A Broadcom Company

technologies

CA Test Data Manager 4.9.1

Table of Contents

Release Notes

Test Data Manager (CA TDM) includes the following capabilities to help optimize the quality of your test data:

- Connect to production data sources
- Profile, subset, and mask production data for testing
- Manipulate the data to meet coverage and test matching requirements
- Publish the data to a test data warehouse
- Request the data for testing from an on-demand interface

This document summarizes product enhancements and defect fixes that are provided in the latest product releases.

Release Comparison

This table compares the new features in the latest Test Data Manager releases.

Key Features	4.9.1	4.8.1	4.7	4.6	4.5	4.4	4.3
Data Modeling							
PII Scan of discovered Data Model	Yes	Yes	Yes	Yes	No	No	No
The Data Model in CA TDM Portal	Yes	Yes	Yes	Yes	Yes	No	No
PII Audit Using CA TDM Portal	Yes	Yes	Yes	Yes	Yes	Yes	No
Configure Dynamic Test Data Reservation Service in Portal	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Prepare Test Data for Relational Databases	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Prepare Test Data for Non Relational Data Sources - GT Excel	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Prepare Test Data for Non Relational Data Sources - CSV	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Prepare Test Data for Non Relational Data Sources - XSD, XML, WSDL, JSON and RR Pair	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Create and Manage Projects and Versions in Portal	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Data Generation							
Create and Manage Generator Configurations in Portal	Yes	Yes	Yes	Yes	Yes	Yes	Yes
New Publish Service for Data Generation in Portal	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Synthetic Data Generation in Portal	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Tester Self Service							
Create and Edit a Find & Reserve Model	Yes	Yes	Yes	Yes	No	No	No
Create and Edit a Find & Reserve Model	Yes	Yes	Yes	Yes	No	No	No
Find and Reserve - View Data From Related Columns	**Yes**	No	No	No	No	No	No
Find and Reserve Test Data Interactively in Portal	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Reserve Test Data with Self Service Catalog Forms in Portal	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Orchestration							
Configure Javelin Programs in Portal	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Data Masking							
Scalable masking with Docker	Yes	Yes	No	No	No	No	No

Mask Data with CA TDM Portal	Yes	Yes	Yes	Yes	No	No	No
Mask Production Data in TDM Datamaker	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Data Subset							
Create Subsets of Production Data in TDM Datamaker	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Virtual Test Data Management (vTDM)							
Install and Register the vTDM Appliance	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Copy Data from vTDM Supported Data Sources	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Consume Gold Copy Clones with vTDM	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Security Enhancements							
Integrated Security for Portal	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Active Directory Configuration in Portal	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Encrypt Passwords in Portal	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Configuration Enhancements							
Configure Telemetry	Yes	Yes	Yes	Yes	No	No	No
Data Masking Support for Hadoop Hive	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Create and Manage Users and User Groups in Portal	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Create and Manage Connection Profiles in Portal	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Create an Advanced File Layout (AFL) with File Definition Manager	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Mail Server Configuration in Portal	Yes	Yes	Yes	Yes	Yes	Yes	Yes
IMS Database Copybook Parser for Data Masking	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Integration Enhancements							
TDM Portal in Linux	Yes	Yes	Yes	No	No	No	No
CA Service Virtualization Integration	Yes	Yes	Yes	Yes	Yes	Yes	Yes
CA Agile Central Integration	Yes	Yes	Yes	Yes	Yes	Yes	Yes
HP ALM Integration	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Installation Enhancements							
Database Installer Utility to Install the TDM Repository	Yes	Yes	Yes	Yes	Yes	Yes	Yes

New Features

The following new features available in the 4.9.1 release fall under the following categories:

TDM 4.9.1 Patch

- ARD self-service flows in the Self-Service Catalog now have persistence.
  In the Self-Service Catalog, the TDM Portal now remembers parameter values and choices of previous requests. The values persist locally in the browser. When Testers create a request, they are prompted to choose whether they want to use the "last used values" or "default values". If any "last used values" have been stored, they are highlighted in the form. For more information, see Reserve Data with Self Service Catalog Forms.
- When you reserve data in the Self-Service Catalog, you can now define an expiration date.
  For more information, see Find and Reserve Test Data Interactively.
- On the Generator detail page, TDM Portal now displays more user-friendly validation error messages for column references.

Columns can reference columns in other tables that themselves contain complex expressions; references can be nested or even be circular. The validation error messages now provide additional detail about the root of the validation problem. The column that caused the error is highlighted with an error icon. Parent columns that are referencing the error are highlighted with a warning icon. Tooltips guide you to the next referenced column with an error or warning. The error and warning icons remain visible even when you browse to the next 10 rows of data, or when you deactivate or activate certain tables in the generator details screen.

- The Generators page now offers a Copy Generator action. You can only copy generators within the same project version.
- In a Data Model scan, you can now also see table views in addition to tables.
  You recognize table views by an icon that looks like a table with an eye. You can click Details to learn more about a View. When you select a View for the first time, it prompts you to provide a primary key. The primary key column is marked with a key icon.
  Notes: All Data Prefetch Modes support table views. You cannot do a PII scan on table views, nor can you mask data in table views.
- On the LDAP configuration doc page, we added the following LDAP attributes: Custom User Filter, User Container, and Group Container.
  For more information, see LDAP Integration with the CA TDM Portal.
- SAP HANA is now supported as a data source in TDM Portal for Data Generation and Data Masking; and also in Datamaker for Data Generation; in Fast Data Masker for Data Masking; for Data Subsetting; and for Data Modelling and PII Audit.

Resolved Issues

Support Case no	Description
32145079	TDM Find and Reserve - TimeStamp not compatible for DB2 on z/OS
	Add TDM version number on main page
32104080	VALIDSIN function was not displaying for number data type
	SAP HANA - gtsubset - additional param is getting misplaced
32090323	Generators: Generating SQL file with quotes in string
32087183	Encrypt the Master Key in FDM for FORMATENCRYPT1
31927351	Show asterisks instead of plain master key in FDM and Portal for FormatEncrypt1 function
	FDM does not work with DB2 when using additional parameters
31955516	Encrypt the Master Key in FDM for FORMATENCRYPT1
31917127	WSDL file upload fails due to multipart.max-file-size
	Issues with evaluating @luhn expression when there is a large value
	FDM need additional parameters field to pass extra options for DB connection param
31837883	TDM Confirm table error in Portal 4.9 due to the large number of table IDs in request header
	Issues Connecting Subset to DB2 Database, added additional parameter field
31855578	Enhancement request for Masking features through Portal
	Allow FDM to mask multiple jsons in one file
	Do not generate empty zip file from generate constraint file option during masking process from Portal
20127699	CA TDM Portal AD Integration - Multiple user OU, Group OU container support; Custom ldap user filter support
31857427	DataReservationService log doesn't show the count of records returned through Query execution and the Actor doesn't capture which user it is retrieving data for
	FDM - Added new formatdecrypt1 masking function based on masterkey parameter

Support Case no	Description
31854388	SEQLOV DataGeneration function not generating correct data with 'S' (Shuffle) option
31889454	FindReserveService log doesn't print the username who triggered the find query and also doesn't print the count of records returned upon query execution
31881933	Error deleting a registered csv object in Portal
31903712	Publish xml - Avoid generating empty xml elements when there is no data for it
31901967	Portal not importing csv file correctly (Registering an object of CSV type with data)
31915579	Logout API doesn't terminate the user session
31956416	Masking fails when LargeTableSplit option is enabled (LargeTableSplit=Y) due to a redundant OrderBy clause being appended to the query
31957442	LDAP users are taking too long to log in and perform operations
32041785	XML element attribute with type as XML schema namespace instance is ignored in the published output xml
32067017	Publish XML with periods in element names
31934159	Security Access function to hide configuration menu
32113168	Failure to view the self service catalog tile details created by user in other projects
32097826	Security Error - User session is terminated before the configured timeout period
	Javelin 64-bit support
20104817	Launch one thread per Javelin flow
31843261	Create Javelin extensions folder (ProgramData) in the same drive where Javelin is installed
20104817	Create copy of flow for loading
32060895	Unable to read Excel files created by Javelin in ExportDataTableActivity
31697169	Spaces in column values from the source table are trimmed while saving in dvid table in gtrep
32090319	Configuration to skip Temporary Tables in Oracle while profiling (tdmweb.profiling.oracle.skip.temporarytable=true)
	FORMATENCRYPT, NFORMATENCRYPT, FORMATENCRYPT1, NFORMATENCRYPT1 support in Hadoop Masking
20258257	FORMATENCRYPT1 is not consistent when using ignore first/last characters
	FDM cannot be executed as standalone jar file
	Implementation of new masking function HashCard1
	Missing option in UI for generating multiple FD files
	Long running test match
20304155	No schema used in TRUNCATE commands
20309826	Query error when using FASTXREF and WHERE condition with DB2 database
20305129	Publishing to table in DB2 fails if column with mixed-case names are used
20310508	Error with DATA MODEL
20310864	Test Data Manager: "The job with id: 289 is not a type which can be cancelled"
20307270	Manage logging activity (gtrep web log tables filling up)
20305974	If evaluation when using dynamic rows in tables
20314297	Data Generator using TDM Portal: repeat count value is not an integer
20302172	Tables locked after executing DML from DataMaker
20304153	TDM Error while copying a version using Remote copy option in Datamaker

Support Case no	Description
20320359	Masking jobs time out in Portal if transfer of audit files takes long time
20320649	Mixed-case column names are not delimited in Javelin flows for DB2
20314575	Create new type of variables to create password type input field - Datamaker
20321641	Missing separator if the whole data row is empty
	TDM Portal LDAP authentication is too slow
20306511	Not able to generate multiple files for a feed without header and trailer
31722651	Unable to cast object of type 'IBM.Data.DB2Types.DB2String'
31790151	Numeric values are stored as text during Excel masking
31792173	Missing index column RD_INDEX in seed table
31794380	Audit password is not secured in FDM UI
31792879	Field properties are reset to default values when they are missing in the update request
31792027	Fields are sorted in wrong order on numeric properties
31795229	DM crashing when recalculating table order with many (thousands) tables
31821754	CA TDM - Linux Portal - Heap Size issue
31824307	Port and additional connection properties are not used when testing db activity
31823761	Masking service fails to update the job status with Oracle repository
31826048	SQLFUNCTION silently ignored in file mode
31819069	Duplicate user relationships not shown
	HASHLOV function is not returning the same value for FDM, DM, and Portal
31843385	The output file is not updated during Excel masking. StringIndexOutOfBoundsException thrown when reading SQL files
31846000	Performance issue with GTEDI Import
31837895	PII column classification skipped if counting rows in a table fails
	Combine the REPORT csvs generated during the Masterflow for the same table into the same file
31851395	Javelin limitation with password encryption
	Column reference with no row number fails validation
	Added possibility to store relationships with context
31856117	OrientDB configuration file corrupted after upgrade
20104817	Unable to publish more than 20-25 jobs at one time.
31872807	The portalEndpoint parameter is required although documentation states optional
31869165	Datamaker / AD integration issue.
31866538	Datamaker - Copy variables between project/versions, Error Saving Variable
31884905	Allow Nullable false fields to have ~NOVALUE~ when used in an if statement
31881975	Javelin64b error when encrypting the Oracle password
	Only target schema can be replaced with variable
31898708	The restart column is not indexed
31930027	Unable to register z/OS layout file
31937935	Query dumps contain passwords even with EnableGlobalDecrypt=false
31943344	Issue with double quotes in input file
31938047	Javelin flow does not consistently display the SourceSchema parameter

Support Case no	Description
31937959	Log file is sent as large attachment when workflow publish completes
31971102	Instance & integrated security not displayed correctly after saving connection profile for SQL Server
31965911	Doubled columns after discovery scan, non-matching entities shown in search
	SAP HANA support proof of concept
	Not registering zOS_AFL elements close to 30 characters max correctly
31957440	WHERE condition not supported for Excel masking
32022065	ARD unable to connect to TDMService
32060384	Unable to drop restart column with SQL Server database
32068020	Version not displayed on connect dialog of FDM Mapper
	formatencrypt1 support in datamaker (transformation map)
32068549	Javelin crashes during startup under restricted user
31913635	Validation of the expression with 13+ column references in DataPainter can fail in IE
32089536	Fast Data Masker errors when attempting to select an XML root in a DB2 CLOB field
32090331	Default value in generator could contain enclosed quotes
	Mandatory indicator is not displayed for SQL Server identity columns
32089910	Index creation for F&R tables could fail with many columns selected as model keys
32079632	GT Datamaker crashes when trying to edit data pool
	Append Export Datatable activity does not work for Excel
32113032	FORMATENCRYPT1 generate duplicates
	Datamaker: remote copy project loses ref to publish in ARD flow
32121683	Custom masking functions don't work in DB mode
	The password encryption/decryption could fail
32119922	The value containing comma is quoted even when different delimiter is used
32119968	Default charset is not used when counting lines in the input file
	<LF> is not recognized as line ending
32158080	Index issue with gtrep_reference_data
32158852	Javelin Clone import incorrect column relationships
32170377	Execution of post/pre scripts don't show number of affected rows
	Masking is not stopped if discrepancy between head and function rows is detected
32119986	TDM FDM - FORMATHASH: High-Leading and trailing spaces in masked field are removed after masking
32146251	TDM: Unable to connect DB from portal using self-service
32160986	After applying TDMWeb-4.9.85.0 we noticed Publish is not working
32106726	Sybase database masking using FDM
32134718	Tiles fail to load after 4.9.71.0 Portal upgrade
31816698, 3182635	ARD to TDM integration is not working using TDMService
31784109	TDM - Default Report keys functionality is not working in new TDM release 4.8.100
31825983	Datamaker Hashlov is not working
20317089	HASHCARD1 issue
20309598	formatencrypt1 exclude characters not implemented for ascii character set
20317746	Password with special characters fails oracle DB activity

Support Case no	Description
20282626	Column referred in generator gives ""Invalid column referenced"" error
20313232	Customer needs help with TDM
20300937	TDM repository DB GTREP table gtrep_su_log logs user password in the clear in
20132191	Error using LARGETABLESPLITENABLED option with masking function SQLFUNCTION
20111316	HEX values change after the load from DB2 table to DB2 table
20110869	Review transpose and translate FDM options
20100014	Masking on fixed width file
20060835	PII Audit and Masking jobs does not get aborted on cancelling
20077309	Issue with speed of masking in FDM
20061801	We cannot change the Default Masking Function from IE 11
20078209	Steps to revert CA TDM Portal version 4.8.102.0 to 4.8.100.3
20053469	Error with FDM when using the DBUPDATES = P in the option tab with SQL SERVER
20029298	Issue in opening .csv and .xlsx in Javelin after patch shared in defect DE421849
20043253	Javelin fails with invalid username/password
20038978	TDM: Masking w/ Portal or FDM doesn't honor option to use mask values
20016503	Fast Data Masker and Sybase 15.7
1349754	FDM and Excel
20000527	A Corrupted Xlsx File is generated when running a Javelin Tool
20017287	FDM and Excel
20018242	Javelin Generated Xlsx Files of 0kb, throws an error when opening the file
1355139	Cannot Validate Referenced Fields: SHRED_ID and SHRED_GROUP_ID: No errors in the logs
20011350	Fields with functions using exesql failed validations in the Portal
20009857	Portal Tiles Not working after 4.8 Upgrade
	FDM not working with comma delimited file masking
1373200	Unable to delete the Environment from an existing Project in Portal
31924051	Slowing to finish submitted request on TDM portal
32087199	TDM 4.9.53 log running completed Job is not being communicated to other parts of services and jobs are queueing
31924112	Slowing in the animation when clicking on made request from tdm portal
32072410	Data Not Generating For Delimited File when Values Passed from Variable File in Portal
31825983	Orai18n support into portal
	Extra connection parameters not passed to the FDM Docker engines
31890880	TDM: Masking function uses the same seedtable value for every row of the table
32107720	TDM: ~User~ being converted to 'Integrator' in 4.9.69
32092341	DVID tables not synchronizing for TDM Find and Reserve with back end tables for caching Duplicate key errors
32119149	TDMService is passing incorrect information
32153752	'All Projects' check box on Self Service Catalog page not selected by default
31819834	Oracle connection profile missing additional parameters
31801207	Modify FORMATLUHN to exclude the first character
20309598	Checksum fail with Oracle connection profile in Portal - Subset Version

Support Case no	Description
20309598	Checksum fail with Oracle connection profile in Portal
20309598	Checksum fail with Oracle connection profile in Portal - FDM Version
	Masking credit card numbers enhancements
	Multithread large table processing
	Hashlov 1 add new parameter
20048718	FORMATENCRYPT1 problem
	DBUPDATES = P not generating the code for Unique index
20048718	Enable excluded Characters Field in the Portal
	FORMATENCRYPT support for french accent characters on LUW and Mainframes
	Enable FORMATENCRYPT support for french accent characters on Portal 4.8
1312597	Flatfile masking performance issue

TDM 4.9

General Features

- All license key checks were removed from the product. For more information, see Activate Test Data Manager.
- The publish job engine has been tuned to support peak load requests without timing out. We are now able to handle thousands of publish job requests within a short span of time.
- Miscellaneous customer fixes and enhancements

TDM Portal

Find & Reserve enhancements:

- **Data Prefetch: Off**
  Up to TDM 4.8, when you created a Find & Reserve model, TDM always prefetched data into the TDM gtrep repository (Data Synchronization). From 4.9 on, you have the option to switch Data Prefetch off, and let TDM query your source database directly. This functionality supports only a single data source and requires you to create an additional reservation table in your source database.
  For more information, see Data Prefetch.
- **Related Tables**
  You can now view columns from related tables when finding & reserving data. You can also export these related tables as part of the CSV file.
  For more information, see Find and Reserve Test Data Interactively.
- **Linked Form Fields**
  For new data models, related Find & Reserve form fields are now linked so they do not show inapplicable combinations. For example, after selecting a "country", the "city" form only lists entries related to that country.

Fast Data Masker

- **Scaled Masking – Large Table Support**
  We have enhanced the scaled masking functionality to support very large tables. The configuration parameters now allow a large table to be split into blocks for processing by the FDM engine. Each block is then processed by a separate CPU thread, allowing for maximum parallelization.
  For more information, see Fast Data Masker Best Practices and Masking Options.
- **RC Extract - TDM on Mainframe Integration**

CA RC/Extract for Db2 for z/OS and CA Test Data Manager allows you to elevate the security of your mainframe Db2 data through intelligent masking capabilities. Mainframe Db2 data must frequently be extracted from one environment to another (such as from production to test). This new capability ensures that the extracted data remains secure through intelligent data masking -- a unique method of creating a structurally similar but inauthentic version of an organization's data. With this new capability, you can specify the columns to be masked either in flight or in place. Additionally, you can define and subset Db2 data as part of the data masking process.
For more information, see Integrate with CA Test Data Manager for Data Masking in the CA RC/Extract™ for DB2 for z/OS documentation.

- **New Masking Functions**
  We have added several masking functions including Formatluhn and Hashsin for the Mainframe, and Formatencrypt1 that lets you specify custom keys for masking.
  For more information, see Masking Functions and Parameters

Patch Releases

A component patch release is typically denoted by four numbers, for example, 4.4.0.43 (patch release for Datamaker component). A new component patch includes resolved issues that were shipped as part of previous component patches. For example, Datamaker patch release 4.4.0.40 includes all previously resolved issues that were shipped as part of patches 4.4.0.x, where x is less than 40.

If you are using Test Data Manager on the previous release, or the one before that, use this table to find out if your patch is included in the next GA release.

The Test Data Manager 4.9.1 release includes the following latest patch versions:

Component Name	Patches based on TDM 4.9	Patches based on TDM 4.8.1
Datamaker	4.9.16.0	4.8.24.0
CA TDM Portal	4.9.87.0	4.8.217.0
Javelin	4.8.137.0	4.8.137.0
GT Subset	4.9.17.0	4.9.17.0
Fast Data Masker	4.9.52.0	4.9.52.0
TestMatch	4.8.100.51	4.8.100.51

The Test Data Manager 4.9 release includes the following latest patch versions:

Component Name	Patches based on TDM 4.8.1	Patches based on TDM 4.7
Datamaker	4.8.14.0	N/A
CA TDM Portal	4.8.171.0	4.7.123.0
Javelin	2.0.713.0	2.0.713.0
GT Subset	4.8.16.0	4.8.16.0
Fast Data Masker	4.8.153.0	4.8.153.0

Acknowledgments

The document #unique_48 attached to this page, contains license agreement information for third-party software that is used in Test Data Manager.

Product Accessibility Features

CA Technologies is committed to addressing user accessibility in the development of its products and documentation to help all customers, regardless of ability, to accomplish vital business tasks.

Keyboard Support for the CA TDM Portal

This section lists the keyboard navigation support provided in the CA TDM Portal.

Edit Generator Table

The following keyboard navigation support is available while creating data generation rules:

- **Tab**
  Moves the focus to the next UI field.
- **Shift + Tab**
  Moves the focus to the previous UI field.
- **Ctrl + Space Bar**
  Opens Data Painter window for a focused cell in a table.
- **Escape**
  Exits from the current focused UI field.
- **Up and Down Arrow**
  Toggles between the options in an expanded list items or the toolbar options of a cell.
- **Left and Right Arrow**
  Toggles between the options in expanded list items or the toolbar options of a cell.
- **Enter**
  Performs the action relevant to focused UI field.

Test Data Model Creation

The following keyboard navigation support is available while creating test data models:

- **Tab**
  Moves the focus to the next UI element.
- **Shift+Tab**
  Moves the focus to the previous UI element.
- **Spacebar**
  Displays the items available in the selected list element or selects/deselects the focused check box.
- **Escape**
  Closes the opened flyout or pop-up dialog.
- **Up and Down Arrows**
  Navigates through the items available in the selected list element.
- **Enter**
  Performs the action relevant to the focused UI element.

Find and Reserve Test Data Interactively

The following key board navigation support is available while finding and reserving test data interactively:

- **Tab**
  Moves the focus to the next UI field.
- **Shift + Tab**
  Moves the focus to the previous UI field.
- **Space Bar**

Displays the items available in the selected list element or selects/deselects the focused check box.

- **Enter**
  Performs the action relevant to focused UI field.
- **Up and Down Arrow**
  Navigates through the items available in the selected list element.
  Moves the focus to the next row in a table.
- **Left and Right Arrow**
  Moves the scroll horizontally within a table.
- **Escape**
  Moves the focus from inside the table to the next UI field outside the table.

Getting Started

This section includes information about Test Data Manager (CA TDM) concepts, features, architecture, and components. Review the information in this section before you start with the product.

View this short video for a summary of how implementing test data management practices into your development lifecycle can reduce development time and increase test data quality:

Role in the Continuous Delivery Ecosystem

CA offers a collection of integrated software planning, development, testing, and delivery tools that create a complete Continuous Delivery Ecosystem. This ecosystem helps you overcome the complexities and obstacles of these new demands and puts innovation in the hands of your customers faster, at lower cost, and with reduced risk. The strategic integration points between these tools help you manage your entire software development lifecycle, end-to-end.

Test Data Manager (CA TDM) is one product in the ecosystem. It provides end-to-end test data management capabilities, including masking, subsetting, profiling, data generation, on demand data provisioning, and more. CA TDM integrates with other products as follows:

- **Test Data Manager with Rally**
  Lets you export the matched data to a CSV file that is directly attached to the test cases in Rally (formerly CA Agile Central).
- **Test Data Manager with CA Agile Requirements Designer**
  Lets you use the generated synthetic data in the test cases that you create from your requirements flows.
- **Test Data Manager with CA Service Virtualization**
  Lets you generate realistic virtual data that covers the full range of possible scenarios for effective service virtualization.
- **Test Data Manager with CA Release Automation**
  Lets you inject synthetic test data into your schema when deploying and testing a new build.

The following video demonstrates how the products in the Continuous Delivery Ecosystem work together in a sample scenario.

Key Use Cases

CA Test Data Manager eliminates friction in your development process related to the availability and quality of test data. The product gives test data engineers the tools to ensure that developers and testers always have the data they need to perform high quality tests and do not have to perform lengthy manual work to get that data.

The test data management discipline spans multiple capabilities spanning data compliance, volume, quality, availability, and portability. CA Test Data Manager provides these capabilities in the form of the following key use cases:

Synthetic Data Generation: Generate all the Data Needed for Rigorous Testing

CA Test Data Manager combines powerful synthetic data generation with sophisticated coverage analysis, enabling organizations to create the smallest set of data needed to for comprehensive testing.

You can model data stored across an entire enterprise. You can then generate realistic synthetic data to fill in the gaps and provide full test coverage. The outcome is smaller richer sets of generated data for complete testing, while avoiding prohibitively high infrastructure costs. Combinable data generation functions, bulking scripts and substitution variables automatically create millions of rows of complex, up-to-date data as fast as the database infrastructure can handle. This

includes future scenarios and unexpected results, so that testers have the right data to test boundary conditions and avoid the spiraling costs and delays caused when critical bugs are detected too late.

NOTE

Generate Synthetic Test Data

Data Generation Functions and Parameters

Store Data Centrally in a Test Data Warehouse

You can import existing data and flat files stored across an organization. All imported, generated, and augmented data is stored in a central repository. you can then extract, clone, and deliver re-usable subsets of data to test teams on demand, eliminating data dependencies and maximizing the value of work already done.

Sophisticated version control means that relevant data sets automatically reflect and replicate changes made to system requirements across projects and versions. Test and development teams can work with the most up-to-date data, developing multiple releases in parallel, from stable, isolated environments.

NOTE

Defining Test Data Using the CA TDM Portal

Configure Test Data Reservation Service

Data Reservation: The Right Data, to the Right Place, at the Right Time, in the Right Format

Data reservation means that testers receive the data they need from multiple back-end systems in minutes, eliminating the time otherwise wasted looking for or preparing data, or creating it where none exists. They can request the exact data they need, and have it automatically delivered from the central repository in parallel and on demand.

Dynamic form building enables testers to mine data using pre-defined criteria, rather than fixed values, using Tester Self-Service TDM Portal functionality. This data is "matched" to the exact tests it can run, allowing testers to perform more stable tests, earlier and with greater repeatability. Test teams can work in parallel, detecting defects earlier and accelerating the delivery of fully tested software at less cost to the business.

NOTE

Configure Test Data Reservation Service

Tester Self-Service

Data Subset: Copy and Extract Coherent Subsets of Data

As data sets are provisioned, they are automatically "cloned" so that the original data remains intact. This means that teams can work in parallel, while avoiding the slow and expensive process of manually copying and moving large copies of production data. Testing using intelligent subsets of data further avoids prohibitively high infrastructure costs, while delivering "cloned" data to isolated test environments prevents bug scenarios being lost when one team makes a change. Data is likewise preserved during a database refresh, meaning that multiple test runs can be performed in parallel, delivering valuable software to market earlier and at less cost.

NOTE

Subset Production Data

Data Masking: Compliant Data in Test Environments

With CA Test Data Manager, you can secure existing data in minutes, mitigating risk while reducing the cost of compliance to the business.

CA Test Data Manager provides powerful data profiling algorithms that automatically discover potentially sensitive information stored enterprise-wide. Over eighty combinable masking functions and four high-performance native masking engines secure large, complex sets of data in minutes. Referential integrity and complex relationships are maintained, producing data with all the characteristics of production but none of the sensitive content. You can also mask data in-flight for secure service and message virtualization, allowing teams to work in parallel, free from dependencies and constraints.

NOTE

Mask Production Data with Fast Data Masker

Data Discovery and Profiling Using Datamaker

Realistic Service and Message Virtualization

CA Test Data Manager helps organizations provide testers and developers with the environments they need to deliver quality software earlier, and at lower cost. From a message specification which has been analyzed in CA Service Virtualization, you can generate realistic request-response pairs and inject them into a deployed virtual service. This data is synchronized across the interdependent databases and services that exist in complex applications, providing testers with access to otherwise unavailable or incomplete components.

NOTE

Defining Test Data Using the CA TDM Portal

Integration with CA Service Virtualization

Mainframe Test Data Management

CA Test Data Manager provides a robust framework for managing test data on mainframe and distributed platforms. A single UI and repository is used to define test data engineering tasks, using mainframe batch operations and engines to execute them in mainframe runtime environments. You can therefore leverage the reliability of the mainframe without investing in multiple tools.

All of CA Test Data Manager's capabilities are available across a broad range of platforms. Data profiling builds a multi-dimensional image of complex data stored across mainframe and legacy platforms. This includes referential information not otherwise available from mainframe sources so that the referential integrity needed for testing is retained even when faced with massive technical debt and minimal documentation.

For more information, see Mainframe.

Key Components

The following key components provide the primary Test Data Manager capabilities:

NOTE

There are other installable components that provide peripheral functionality, such as ALM integrations, or superseded functionality, such as Test Data on Demand. For a complete list of installable components, see Installing.

Repository

The Repository is a database that stores product meta data, registered data, projects, and more.

CA TDM Portal

The CA TDM Portal provides a modern web interface for key capabilities such as project and data management, data registration, data generation, and data reservation. CA TDM Portal gains new functionality each release. For more information about the capabilities available in the CA TDM Portal, see CA TDM Portal.

Datamaker

Datamaker provides a core project and data management interface and core data registration and generation capabilities. Datamaker also links to several other components that provide profiling, subsetting, and other capabilities.

NOTE

Many Datamaker capabilities are now available in the CA TDM Portal, including:

- Connecting to data sources
- Registering non-relational data
- Data generation

Where possible, use the CA TDM Portal for capabilities generally available in both tools for the best user experience and continued maintenance.

Fast Data Masker

Provide core data masking capabilities. From Fast Data Masker, you can connect to a data source, select columns to mask, define masking rules, and mask the selected data to ensure data compliance.

Data Subset

Provide an interface for identifying criteria for taking meaningful subsets of a larger data source for testing.

Data Profiler

Data Profiler is an interface within Datamaker that lets you sample your data and better understand its characteristics. For example, you can use Data Profiler to help identify personally identifiable information that requires masking.

Test Data Visualizer

Test Data Visualizer is an interface within Datamaker that gives you a graphical view of data coverage. When connected to your test data warehouse, Data Visualizer can show identify gaps in your data when you might need to generate synthetic data to ensure appropriate testing coverage.

Javelin

Javelin is a test data orchestration tool that lets you model complex test data provisioning workflows for automated execution.

Mainframe

Mainframe components provide the artifacts to let you work with mainframe data within the core Test Data Manager components.

CA Agile Requirements Designer

CA Agile Requirements Designer is a separate product that is used for the construction of flows for data generation and test matching. You receive a license for CA Agile Requirements Designer when you purchase Test Data Manager. For more information, see Install CA Agile Requirements Designer.

Resources

This section contains links to other useful CA Test Data Manager (CA TDM) resources. Use these resources to discover more about CA TDM.

User Community

The TDM Community is the place to share ideas, tips, information, insights, and more with your business peers and CA Technologies experts. The community provides a unique opportunity to network and help you maximize your software investment by tapping into a community of expertise, open 24/7.

Knowledge Base (KB) Articles

CA TDM has a vast Knowledge Base to help you identify workarounds for known issues or resolve popular issues with your implementation.

Learning Paths

The CA TDM Learning Paths contains recommended courses. You can click a course from the learning path to immediately get started with your learning experience. Courses are offered in a variety of self-paced delivery options, as well as traditional instructor and virtually led classes.

Videos

Watch the CA TDM videos available on YouTube from CA Technologies to increase your CA TDM knowledge.

CA TDM Tutorial Videos

The following tutorial videos provide extra information, context, and examples that augment the product documentation.

End-to-End Integration

The following videos complement the Getting Started documentation:

CA Continuous Delivery Ecosystem

This video shows how you use Test Data Manager, CA BlazeMeter, CA Release Automation, CA APM, CA Agile Central, CA Agile Requirements Designer, CA Service Virtualization, and CA Application Test together to create a continuous delivery chain.

Data-Enable Virtual Services

A core guide to the functionality available within CA TDM. See how CA TDM and CA Service Virtualization work hand in hand to provision data to virtual services. For more information, see Integration with CA Service Virtualization.

CA Service Virtualization Integration

Integrate Test Data Manager with CA Service Virtualization to generate and inject realistic virtual data into a running virtual service, and to execute varied testing scenarios for effective service virtualization. For more information, see Integration with CA Service Virtualization.

Find and Reserve Test Data

The following video complement the *find and reserve* documentation:

CA TDM Portal—Find and Reserve Test Data Interactively

This video demonstrates how testers use the dynamic self-service forms to interactively find, view, analyze, and reserve the test data. The CA TDM Portal helps organizations manage the full life-cycle of test data reservation. It lets TDEs create test data models, which simplify the overall data reservation process by encapsulating the automatic creation and management of test data marts. Furthermore, by sharing the test data models with testers as dynamic self-service forms, the CA TDM Portal also ensures that the data becomes available to testers in minutes. For more information, see Find and Reserve Test Data Interactively.

CA TDM Portal—Reserve Data Using CA ARD Forms

This video demonstrates how testers request and reserve test data using the CA Agile Requirements Designer (CA ARD) forms in the CA TDM Portal. Test data engineers (TDEs) can create self-service forms using CA ARD and expose them to the CA TDM Portal. These CA ARD-based self-service forms then become available to testers as forms in the Self-Service Catalog interface. As a tester, you can consume these CA ARD-based forms to publish the data, to perform the test match, and to attach the test data to HPALM or CA Agile Central test cases. For more information, see Reserve Data Using CA ARD Forms in the CA TDM Portal.

Provision Test Data

The following videos complement the Provisioning Test Data documentation:

CA TDM Portal—Create Test Data Models

This video describes how test data engineers (TDEs) use the CA Test Data Manager (CA TDM) Portal to create test data models that facilitate the data reservation process for testers. Created test data models are then shared with testers as dynamic self-service forms in the Self-Service Catalog interface of the Portal. Testers can use these forms to interactively find, view, analyze, and reserve the test data. For more information, see Configure Dynamic Test Data Reservation Service.

CA TDM Project Setup and Walkthrough

A 101 guide on how you set up a new project workspace within Test Data Manager, followed by a high-level walkthrough of the main features. We show you around the product to find the quickest ways to complete your tasks. For more information, see Create and Edit Projects.

CA TDM Test Data Profiling

A 101 introduction to data profiling with Test Data Manager. Understand the key reasons why profiling your data estate is important, and learn best practices, tips, and tricks to make this an easy process. For more information, see Profile (or Sample) Your Data.

CA TDM Test Data Visualization

A quickstart guide to launching and using CA TDM Test Data Visualizer. Analyse your test data coverage, and spot the gaps in your test systems. For more information, see Visualize Test Data Coverage.

CA TDM Diagrammer

The following video shows how to use GT Diagrammer to visualize a Database Schema into an Entity-Relationship Diagram. For more information, see GT Diagrammer.

CA TDM Test Data Subsetting

A 101 introduction to CA TDM Data Subset and its major functionalities. Learn how to move data from database to database whilst maintaining referential integrity with ease, repeatability, and efficiency. For more information, see Subset Production Data.

CA TDM Test Data Masking

A 101 guide on setting up a new Masking routine and obfuscating data. Learn how to build, validate and run masking jobs to secure your data from threats. For more information, see Mask Production Data with Fast Data Masker.

CA TDM Test Data Cloning

A 101 introduction to basic data cloning. Learn how to use Test Data Manager to clone copies of your data in bulk from very simple beginnings, and create an army of clones to test with. For more information, see Subset Production Data.

CA TDM Test Data Generation

A 101 introduction guide to basic a data generation for beginners. Learn how to build structured rules and routines that generate data as and when its required. See the core data generation functionality and see how easy it is to do. For more information, see Generate Synthetic Test Data.

CA TDM Virtual Test Data Mangement (vTDM)

CA TDM Data Discovery and Profiling

This video describes how Test Data Engineer (TDE) use the CA TDM Portal to scan Data Sources for PII data against one or more Classifier Packs, confirm the findings, and create a draft report to be signed off. An Internal Data Controller reviews the findings in the draft report and signs off. An Internal Auditor can download and review the final Audit report and a Management User or an External Auditor can request the Audit Report from the TDE. For more information about Data Discovery and Profiling, see PII Audit Using CA TDM Portal.

CA Test Data Manager Education and Training

This document summarizes the most current and relevant learning resources available for Test Data Manager, providing learning paths based on your desired level of knowledge. Just one hour of training saves five hours of lost productivity. Trained users make 35% fewer support calls. A well-trained team also delivers more value from their technology investments.

Supported Business Outcomes

The Test Data Manager training resources included in this document support the following key business outcomes:

- Create an agile business (re-orgs, acquisitions, partnerships, resource planning)
- Enable rapid development and releases of high-quality applications (SDLC optimization)
- Maximize application performance and availability
- Support regulatory compliance and security
- Provide workforce with rapid access to resources required to do their job
- Reduce risks and threats to the business
- Provision test data in support of agile objectives, shift left and in-sprint testing
- Improve development and testing efficiencies by making the right test data available when needed in a self-service model
- Improve quality through meaningful testing early and increased test data coverage
- Reduce costs through automation, reduced environment costs, and increased productivity

Enterprise Software Training Tiers

Broadcom learning resources for Enterprise Software are available in three categories:

Tier 1: Onboarding (Free)

Free web-based training to teach you how to use the product and ensure a smooth onboarding experience.

Tier 2: Scaled Adoption (Paid)

Instructor-led training to help you apply what you've learned in a more immersive environment with expert instructors and hands-on labs.

Tier 3: Certified Expert (Paid)

Advanced instructor-led courses and certification exams to help you become a trusted product expert.

Note: The course links in this document require you to log in to Learning@Broadcom as an enterprise customer for them to work. If the links do not work or you are an internal employee, log in to Learning@Broadcom and copy the Course Code into the search window. For more information about access, see the Appendix in this document.

Tier 1: Onboarding Training (Free)

Register for free at BlazeMeter University and sign up for the free TDM course.

Title	Length	Description
Test Data Manager Fundamentals	3 hours	You will learn how to profile, mask, generate, and publish your test data in the TDM Portal web interface, as well as set up TDM itself. By passing the course exam, you will attain a certificate.

For TDM, the Learning@Broadcom onboarding tier consists of a free curriculum of online courses that cover the most common TDM features and use cases:

Title	Course Code	Object Type	Length	Description
Test Data Manager Onboarding Training Curriculum	88TDM20615	Curriculum	20 hours	Curriculum object that contains all courses in the Onboarding tier. Register for this curriculum to get quick access to all free courses.
Test Data Manager 4.6: Overview	88TDM10120	Web-based Training	2 hours	Provides an understanding of the basic tools found in Test Data Manager used to manage, transform, and generate test data for software development.
Test Data Manager 4.6: Data Discovery	88TDM20570	Web-based Training	2 hours	Describes how Test Data Manager collects, manages, and profiles source data to provide a structured and centralized approach to test data management.

Test Data Manager 4.6: Data Subsetting	88TDM20580	Web-based Training	4 hours	Describes the fundamentals of subsetting to create smaller collections of data to use for testing.
Test Data Manager 4.6: Data Masking	88TDM20590	Web-based Training	4 hours	Describes how to mask data using Fast Data Masker.
Test Data Manager 4.6: Data Generation	88TDM20600	Web-based Training	4 hours	Describes how to use the data generation functionality in the TDM Portal to create data generation rules.
Test Data Manager 4.6: Tester Self Service	88TDM20610	Web-based Training	4 hours	Describes how to use the TDM Portal to build a self-service environment, create test data models, and make those models available for reservation by testers.

For the onboarding training tier, you can take all courses in order for a full overview of the main Test Data Manager features. Or, if your organization only plans to leverage certain features, you can take any of the individual courses on a standalone basis.

Tier 2: Scaled Adoption Training (Paid)

For TDM, Tier 2 consists of an instructor-led training and other assets that can enrich the knowledge gained from Tier 1:

Title	Course Code	Object Type	Length	Description
CA Test Data Manager 4.4: Foundations 200 (ILT)		Instructor-led Training	4 days	Learn how to discover, subset, mask, generate, and publish data for testing using the foundational features of Test Data Manager. Includes live instruction and lab exercises.
CA Test Data Manager 4.4 Dynamic Lab Bundles	-Explore Architecture: 88TDM2037S -Discover Data: 88TDM2038S -Generate Data: 88TDM2041S -Provision Data: 88TDM2039S -Transform Data: 88TDM2040S -Tester Self-Service: 88TDM2042S	Hands on lab environment and exercises	30 hours	Lab Environments to learn about each major feature of the product. Bundles include lab guides with detailed exercises for each feature. Divided into 6 different courses, all listed in the Course Code column.

The Test Data Manager Foundations training is an intensive 4 day instructor-led course that introduces you to all of the major features of the product using traditional instruction and hands on lab exercises. The Dynamic Lab Bundles provide self-service lab environments for additional hands-on product interaction.

For information about pricing, availability, and registration for Tier 2 training, contact HCL.

Tier 3: Certified Expert (Paid)

For TDM, Tier 3 consists of a certification exam:

Title	Course Code	Object Type	Length	Description
Proven Professional: CA Test Data Manager Implementation Certification	Click Here	Exam		Detailed exam that grants you official Test Data Manager Certification when passed.

The Test Data Manager Certification Exam tests you on what you've learned from previous tiers of training.

Additional Resources

The following additional resources are available to help increase your knowledge of Test Data Manager:

Title	Object Type	Length	Description
Test Data Manager YouTube playlist	Videos	6 hours	A collection of TDM videos. Includes overviews of new features, technical how to videos, and more
Test Data Manager Blogs	Blogs		Blogs about test data management and other disciplines supported by TDM, on https://www.continuoustesting.com

Recommended Learning Paths

We recommend that you start with the free onboarding training, taking all courses to get a good understanding of the primary TDM features. If you want a more interactive experience, progress to Tier 2 and Tier 3 offerings.

Appendix: Learning@Broadcom Access

Access to training courses requires an Enterprise account with Broadcom. Here are basic instructions for creating an account:

1. Click a training URL, or access the Learning@Broadcom home page.
   If you are not already logged in to Learning@Broadcom, a login page appears.
2. Click '**Do not have an account. Register here**.'
3. Select the following highlighted items on the registration page:
   - **Registration Type:** Enterprise
   - **Product Preference:** CA Technologies Software Solutions
   - **Support Access Information:** CA Standard
4. Enter the required information and submit the form.
5. Use the received confirmation email to activate the account and create a Broadcom password.
6. Click the training URL or Learning@Broadcom again, and log in with your enterprise account information.

Architecture Overview

Test Data Manager (CA TDM) acts as a one-stop-shop for all of your data needs. You can store, manage, edit, find, mask, subset, and make test data that is *fit for purpose*. With CA TDM, you no longer wait or search for the right test data.

Delays in software delivery often occur because testing requires a large portion of the development. Testers are regularly challenged to get test data that is comprehensive enough to cover various testing scenarios. With manually created test data, components often do not fit, and data coverage is not exhaustive enough for full testing. CA TDM provides various data manipulation and data management capabilities that help address such scenarios. With CA TDM, you can easily and quickly access the test data. This ability lets testers test earlier in the project cycle and reduce project delivery time. You are able to deliver high quality, rigorously tested applications to production on time and on budget with no bugs.

At a high level, CA TDM provides the functionality to achieve the following objectives:

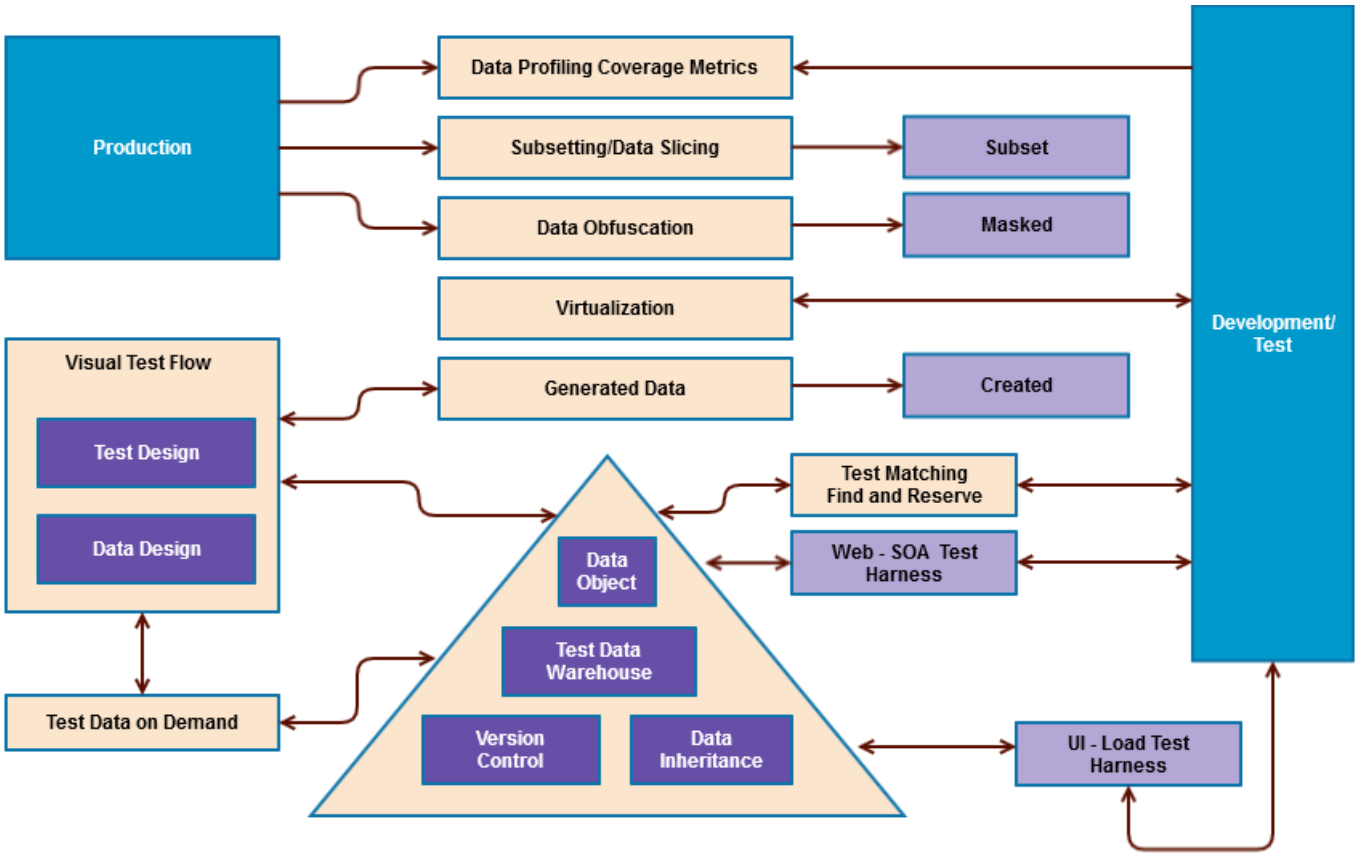
- Connect to production data sources
- Profile, subset, and mask production data for testing
- Manipulate the data to meet coverage and test matching requirements
- Publish the data to a test data warehouse
- Request the data for testing from an on-demand interface

This article includes the following sections:

Component Overview

The following illustration shows the overall CA TDM component architecture:

Figure 1: TDM_Component_Overview



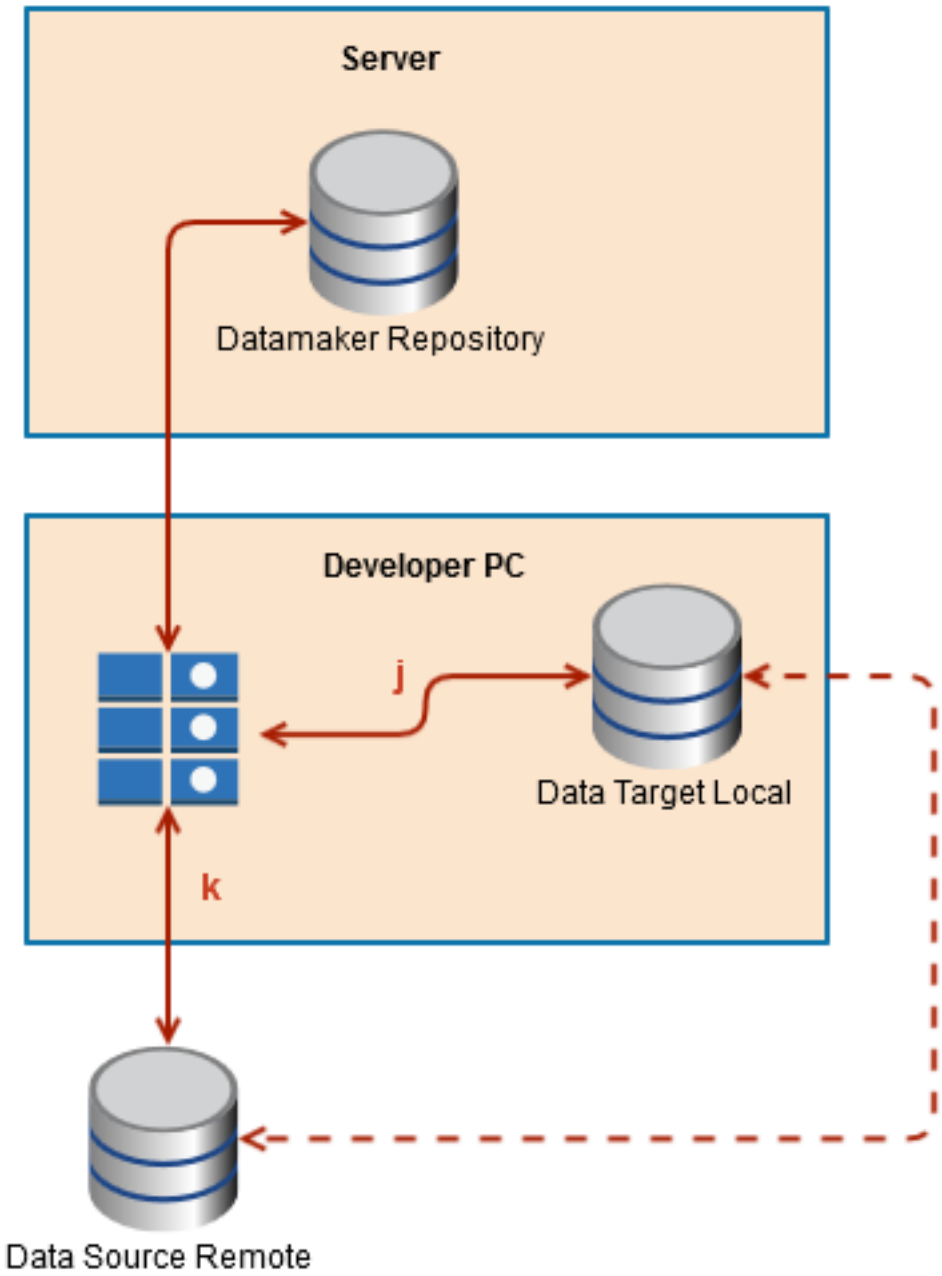
High-Level Architecture

CA TDM follows the client-server architecture. You use the client to define data extracts, data masking, and data relationships. As required, some tasks are performed on the server. However, these tasks are often performed using native database utilities to migrate the data. For example, on OS/390, JCL is generated to extract and load the DB2 data.

The two architecture diagrams in this section illustrate the architecture.

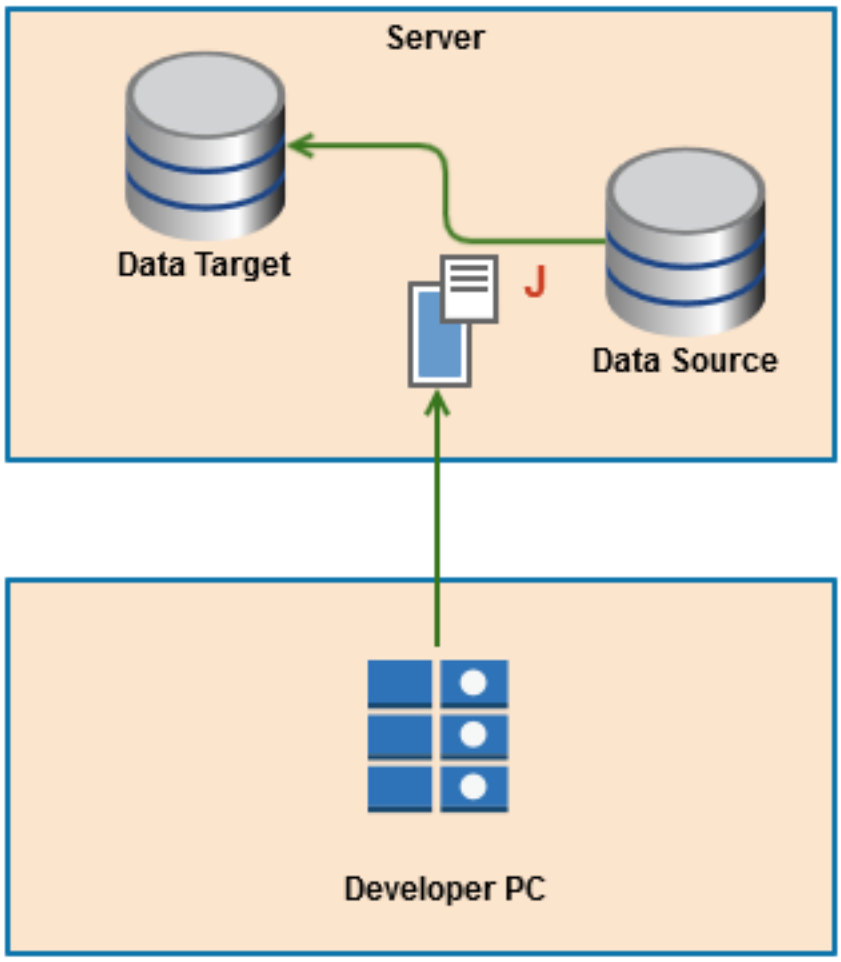
The following diagram shows the client-server architecture:

Figure 2: Client_Server_Architecture



The following diagram shows how generated scripts run on the server:

Figure 3: Server_DeveloperPC_Interaction



The notations that are used in the illustrations represent the following information:

1. **j:** The client can connect to multiple data sources and targets.
   In this illustration, you have a local development database. You can publish data from the central test data repository. You can also copy your own data into the central test data repository for editing and later publishing.
2. **k:** You can also connect to a remote schema and perform the same tasks.
3. **J:** The scripts are generated and are moved to the server to perform the data migrations.

Considerations—Architecture

Review the following CA TDM architecture-related items:

- The central test data repository can be an Oracle or a Microsoft SQL Server schema.
- The repository kit provides starter databases that contain a small training project and a starter project.

NOTE

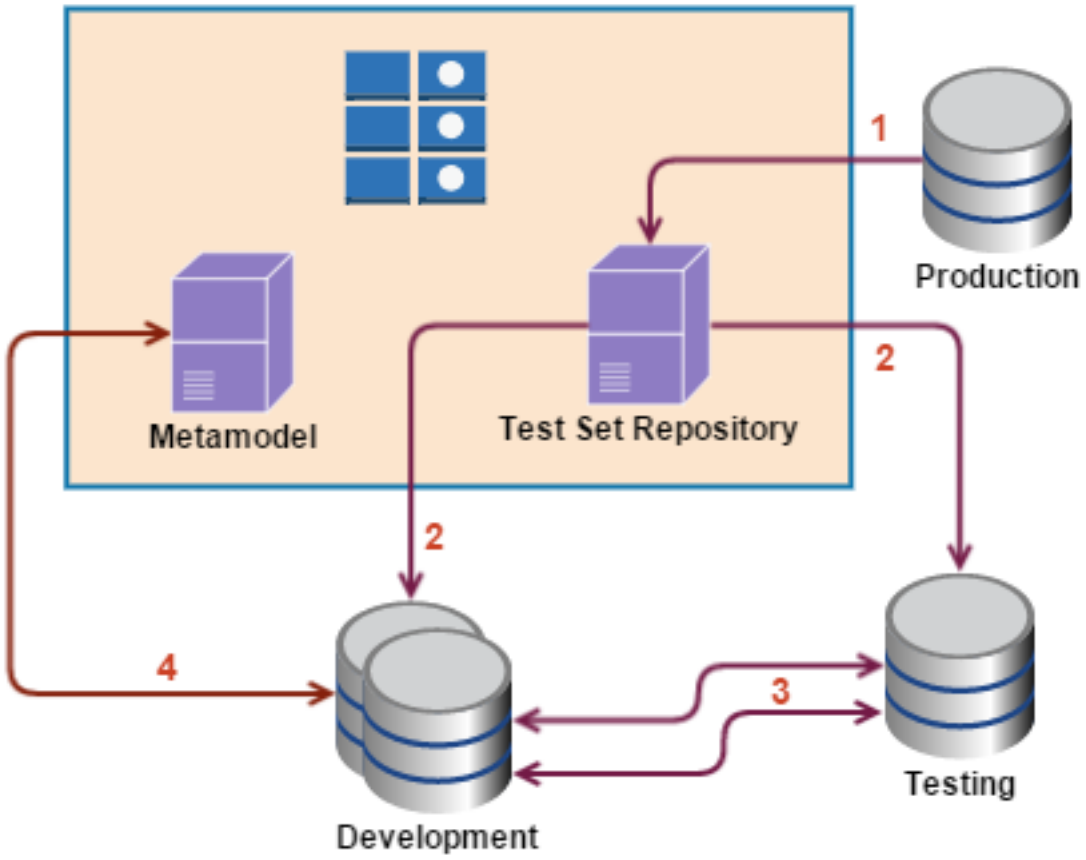
For more information about connection profiles or connecting to a profile or project, see Installing.

- Connections to data sources use normal client connectivity. For example, Oracle, SQL*Net, ODBC, and JDBC.
- CA TDM accesses the source and target database catalog in advance to identify the tables and their underlying structures.
- When directly editing test data, the data is retrieved to the CA TDM server. The data is modified and updates are applied through the client connection to the database.
- When data is published directly from CA TDM to a data source, insert statements are issued through the client to the database.
- CA TDM can publish data to multiple formats, including Microsoft Excel, CSV (Comma-Separated Values), and text. CA TDM can also publish data through connectors to external tools.
- CA TDM can publish custom scripts using native database utilities to move the test data.

Information Flow

CA TDM lets you edit test data directly or copy any related data to a Test Data Repository. You can then edit the data, obfuscate (scramble or mask) the data, or convert the data to metadata. After the data is manipulated, you can publish the data into your development and testing environment. The following diagram shows the data flow with tasks performed at various stages:

Figure 4: Data_Flow_Tasks



The preceding diagram shows the following processes:

1. Copy test data from production, then model and mask.
2. Publish test data to Development and Testing using project parameters.
3. Copy data between Development and Testing and edit the data.
4. Use the meta-model to validate deployment and test databases.

TDM Portal

The TDM Portal modernizes and simplifies tasks that previously required Datamaker, Fast Data Masker and Test Data on Demand. For a guide to your first experience with TDM Portal, see Getting Started with TDM Portal.

TDM Portal functionality is also available as a Docker container, which you can use in Linux. For more information, see Using TDM Portal in Linux.

> **NOTE**
>
> From TDM 4.8, License management and repository maintenance does not require Datamaker. For this reason, you can use TDM Portal without the Windows components (i.e. the components that **GT Server** installs).

The TDM Portal delivers the following features:

- **Mask Data in TDM Portal**
  As a Test Data Engineer, you can mask data in the TDM Portal with the Fast Data Masker engine.
- **Horizontally Scalable Masking with Docker**
  From TDM 4.8, You can use the new Masking and Messaging Docker containers, to scale your masking jobs. For more information, see Scalable masking with Docker.
- **Tester Self Service**
  Testers can request and reserve test data. This was previously done with Test Data on Demand.
- **Administration**
  You can create and manage projects, and various configurations (for example, email server and integration configurations).
- **Data Modeling**
  You can perform tasks on files that you previously performed using the CA TDM Shredder utility, including file registration, import of file data into a relational database, and export of data back into files or to a virtual service. You can also register relational database tables and perform data manipulation operations on the relational data. For more information, see Create a Data Model and Audit PII Data.

  WARNING

  The CA TDM Shredder utility is deprecated and is no longer available in CA Test Data Manager. The same functionality of preparing test data using non-relational sources—XML, XSD, JSON, WSDL, RR pair—is now available in the CA TDM Portal. Moving forward, use the CA TDM Portal for all XML, XSD, JSON, WSDL, and RR pair file registration and data generation needs.
- **Data Generation**
  You can generate smaller, richer, and more sophisticated sets of test data that you previously generated using Datamaker. The CA TDM Portal includes a new publish engine that provides faster publishing times. For more information, see Generate Synthetic Test Data.
- **Virtual Test Data Management**
  vTDM enables you to rapidly create and access lightweight copies of test data sets. Testers reserve individual copies of test data through a simple self-service interface without locking the data for their colleagues. Using vTDM minimizes test duration, storage requirements, compute overhead, and therefore, overall costs. For more information, see Virtual Test Data Management (vTDM).

Getting Started with TDM Portal

When you log into Test Data Manager Portal as an Administrator for the first time, the **TDM Portal Home page** opens. In the top-right corner of the page, you can switch between this view, and the **Insights Dashboard**.

Home Page

From here, you can find links to the most common tasks in TDM Portal. These include:

- **Create a new Project**
  Opens the New Project dialog.
- **Connect to your data**
  Opens the Add New Connection Profile page.
- **Create a Data Model**
  Opens the Data Model page.

  NOTE

  You can click 'Do not show again' to hide these top steps. To reset this option (i.e. to show the top steps), it is necessary to clear the cookies/site data in your browser.
- **Generate Test Data**
  Opens the Generators page.
- **Mask your Data**

Opens the Data Masking page.

- **Find & Reserve**
  Opens the Models for Find & Reserve page**.**
- **vTDM**
  Opens the Virtual Test Data Management page.

Insights Dashboard

In this view, you can track the following metrics that TDM collects:

- Recently Created Generators
- Recently Created Connection Profiles
- Recently Created Projects
- Running Jobs
- Job Queue Time
- Find & Reserve Totals
- Jobs Type Chart (Last 30 Days)
- Users By Login (Last 30 Days)

To change the metrics that display on this page, and the order in which they display, click the **gear** icon to open the **Insights Configuration** dialog.

Using TDM Portal in Linux

You can use features of Test Data Manager Portal in Linux, using the **Docker** application. This instance of TDM Portal runs in a browser, as in Windows.

From TDM 4.8 on, License management and repository maintenance does not require Datamaker. For this reason, you can use TDM Portal without the Windows components (i.e. the components that **GT Server** installs).

For more information, see Install TDM Portal for Docker.

> **NOTE**
>
> This only allows **TDM Portal** to run on the Linux machine. The GT Server components (Datamaker, Fast Data Masker, GT Subset etc), are still only available in a Windows environment. For more information, see Features not available in TDM Portal in Docker.

Datamaker Concepts and Features

This page summarizes the capabilities provided by Datamaker.

Over time, these functionalities will also be available in the new CA TDM Portal UI. For more information, see CA TDM Portal.

Data Editing

Once you connect through a user profile using the SQL window, CA TDM lets you edit data directly. The RDBMS normal security controls the changes to the data you are allowed to make. If you can update a row using any other SQL tool, you can update the row with CA TDM. When you edit a row, the row is highlighted blue. Rows must be saved to be updated.

When you create test data, CA TDM lets you have multiple SQL windows and tabs open simultaneously to perform multiple updates. The product also lets you have two connections active simultaneously, a target and a source. The two connections let you the edit and copy data from one connection to another.

Data Copying

When you copy data between target and source connections, you can copy one or multiple tables at a time. If you copy specific test cases from one connection to another, copy a group of related data instead of one table at a time. For example, a Meter Reading and the associated production Meter, Meter Adjustments, and Billing Data cause a batch process failure. You can copy the Meter Reading and associated tables from a source (production) to a target (development). You can then use the copy to recreate the problem.

You can use Data Subset to copy subsets of data from production to development with the CA TDM defined relationships. You can also copy data from the target and source connections to the central repository.

Table Relationships

When you edit data, it is useful to know how tables are related to each other. You can add and configure missing relationships using the following rules:

- Foreign keys
- Naming Standards
- CASE tools
- DDL
- By direct entry

After you enter a rule, you can select rows in one table and can edit related rows in related tables. When you copy data, you can use entered relationships to identify related rows and create test data in the test data repository.

Entered table relationships are related to a specific project. You can use rules across project versions. You can also copy selected rules from one project to another. With Check Data Integrity, you can verify the actual target data in target and source integrity.

Projects and Project Versions

Before you begin, create a project and project version. If you have a simple application, you can work with one project with one version. For more complex applications, work with multiple project versions.

Save your data definitions against an initial version of the project. Then save (register) changes or new tables against a new project version. A table that is not saved against the current version implies that you are working with a prior table version.

This method lets you identify changes from version to version, instead of saving the definitions of all tables. If you are uncertain about saved definitions, you can save the definition of all tables against the new release.

The following table provides an example of a project and its associated project versions:

Project	Project Version	Tables
Payroll	5.0	Employee, Department, Hours, Bonus, Salary, Grade
Payroll	5.1	Overtime_Rates (New Table), Bonus (New Column)
Payroll	6.0	Supervisor (New Table), Grade, Salary (New Columns)

From this example, you can infer that the Payroll Project Version 5.1 contains the following tables:

- Employee
- Department
- Hours
- Bonus (with new column)
- Salary
- Grade
- Overtime_Rates

Test Case Repository

The data is copied into the test case repository. After the data is copied, you can scramble data or create specific test cases as required. When you edit the data, you can substitute wildcard variables. After you update the data, copy (publish) the data to the target or source connection. You can also publish the data to multiple file formats.

Substitution Variables

When you edit test data in the repository, you can enter wildcard variables. These variables are substituted when you publish the test data. Any value that is entered in the format '~variable~' is substituted when the data is published. *Standard* and *User-Defined* are the two types of substitution variables:

- ***Standard —*** Standard functions that are used to manipulate data when you publish.
  Example: Enter the value '~CDATE~'. This value identifies the current data that is defined in the published connection, and substitutes that value.
- ***User-Defined -*** Created by the user. Allows specific application variables to be substituted.
  Example: Create a variable that is named DEPTCODE. Add this variable to the data in the form of '~DEPTCODE~'. When the data is published, you are prompted for a value for DEPTCODE. This value lets you create the data that includes these user-defined variables that are substituted at the time of publishing.

Tilde variables are substituted when they are published into the target or source connection. You can create custom Tilde variables at each level of the project tree. Any substitution variables are also made available lower down the tree, but not at higher project levels.

Embedded Substitution Variables

You can also embed Tilde variables in other Tilde variables; for example:

You can define ~ACCID~ as ~CY~/~DEPT~/~ROWNUM~.

When you publish, ~CY~ resolves to the year (for example, 2015). ~DEPT~resolves to your department value (for example, 01), and ~ROWNUM~ to the row number. The following data results:

- 2015/01/0001
- 2015/01/0002
- 2015/01/0003
- 2015/01/0004
- 2015/01/0005

Various standard substitution variables are also available. For example, ~ROWNUM~ (the row number), ~NEXT~ (the next highest value in the publish connection).

Test Cases and Data Objects

After you create your project and version, CA TDM lets you create and edit the data. This function lets you create data objects such as test sets and test steps. A project contains the following levels:

- Version
- Data set
- Data pool (or test case).

Data is stored at the bottom level (data pool). You can publish data at the data pool and data set levels. To define the uniqueness of the entity level, define the Key Order as NAME (Character) or SEQ (Numeric).

Note: The number of levels and the name of each level are configurable.

The following example refers to data objects, test sets, and test steps. These items group different test cases together. The data objects are designed to separate different components of an application. This deployment lets different teams and users work on the same project and versions.

For example, a Hospital Billing Application has two development teams. One team works on Hospital Providers and the other team on Claims. The tables are interrelated, but the testing is done independently. This example could result in three different Test Sets:

- Provider Test Cases
- Claims Test Cases
- Claims Provider Interface Cases

Within each test set, you can create test steps that hold the actual test metadata that is published. Each test step holds data across multiple tables that are based on the previously entered table relationships.

When you create test data, it is useful to split groups of data to publish the groups separately. Use the Provider Test Case test set to create various test steps as follows:

- Standard Reference Tables
- Hospitals and Clinics
- Doctors and Ancillary Staff
- High Dependency Units - Area Specific

The first three steps are basic test data. The fourth step is an example of a specific test case. This case is tested and created for specific billing areas as required. You are prompted for the specific area as the data is published.

You can publish all the data for each test set. Any substitution variables that are held in the test case repository are prompted for the appropriate values. You can publish the same data multiple times. This feature lets you create multiple test cases.

Version Control, Upgrading Test Data

Test data is created in data objects. Specific project versions own the data objects. As your project progresses, you can publish test cases from earlier releases to later application releases. When you publish, additional columns in tables are reconciled and default values are inserted into the tables. To set default values, register the table. You can also assign sequences or identity columns to default values.

As your application moves up through releases, you can also upgrade created test cases to the current release. The upgrade of a particular version copies the old test data to the new release (version). The upgrade also identifies new or changed columns and prompts you for changes. You can delete the original test case from the previous release. Earlier releases might be the current production release while development and testing can be several releases advanced. For this reason, retain test cases for earlier releases of an application.

Data Scrambling and Security

Using production data to test or debug applications in development increases the risk of data and security breaches. CA TDM lets you copy specific data pools from production into specific data objects. Data pools are closed from public view until you edit the data and obfuscate Personally Identifiable Information (PII) or private records.

When you know that the data is clean, you can open the data pool to general view. A general view allows users to publish data from the pool and copy data to other data pools. You can audit changes to a data pool to ensure sufficient control and ensure that the data is properly masked. This methodology lets you easily update real test cases that have with specific data nuances to simulate production problems. This methodology eliminates the need to copy an entire production and perform complex key manipulations to de-sensitize the data.

Standard Data (Seed Data)

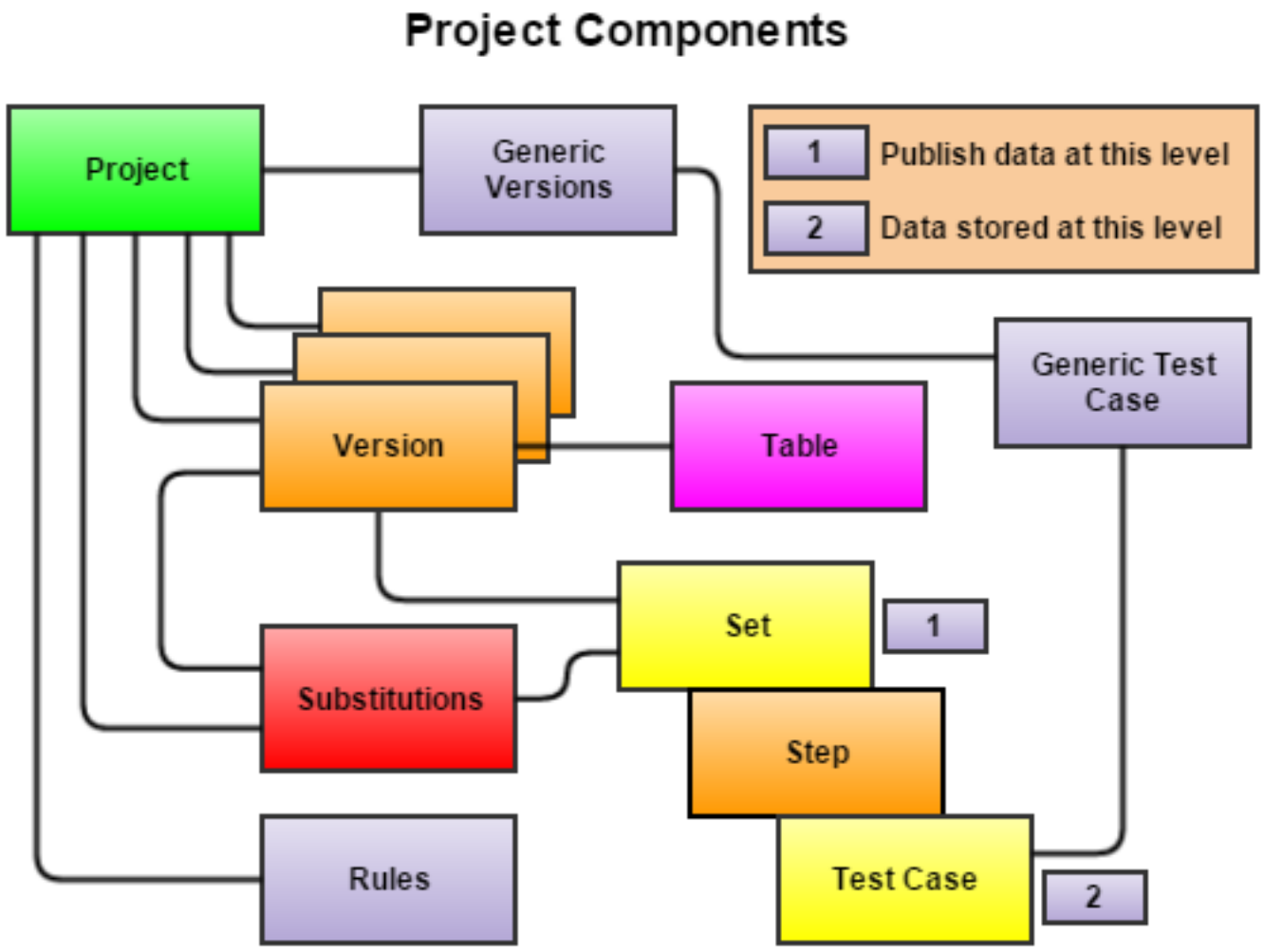
The CA TDM server repository contains a table that stores standard lists of values. Use this table to randomize columns. Examples include, UK Counties, US States, and Random Text. To customize this table and add extra columns, use the Randomize option when you edit

Project Components

The number of levels where test case data is stored is configurable at installation.

The following diagram is an example of a default three-level configuration:

Figure 5: Level three project component configuration



The following level names in the example are configurable:

- Interface
- Test Set
- Test Step
- Test Case.

In these examples, the main component relationships indicate the following guidelines:

- A project can have multiple versions.
- A project can have only one generic version.
- Table definitions are version-specific.
- A later version can use early versions of tables.
- Rules are associated with projects and can apply (if the table and columns exist) across versions.
- Substitution variables are defined within a project and then linked to the project or the publish level.

Getting Started with Fast Data Masker

With greater emphasis on data privacy and compliance regulations, it becomes imperative that organizations ensure that their testing data is devoid of any sensitive information. However, organizations often ignore this basic, yet integral, aspect of testing. By exposing production data to non-production environments, they increase the risk of a data breach, with fines averaging millions of dollars and loss of reputation.

Once personal information is found across production databases, organizations can mask that information to make the data compliant for use in non-production environments. But, manual data masking and other in-house methods to obfuscate sensitive data are slow, adding further delay in providing fit-for-purpose data to testers.

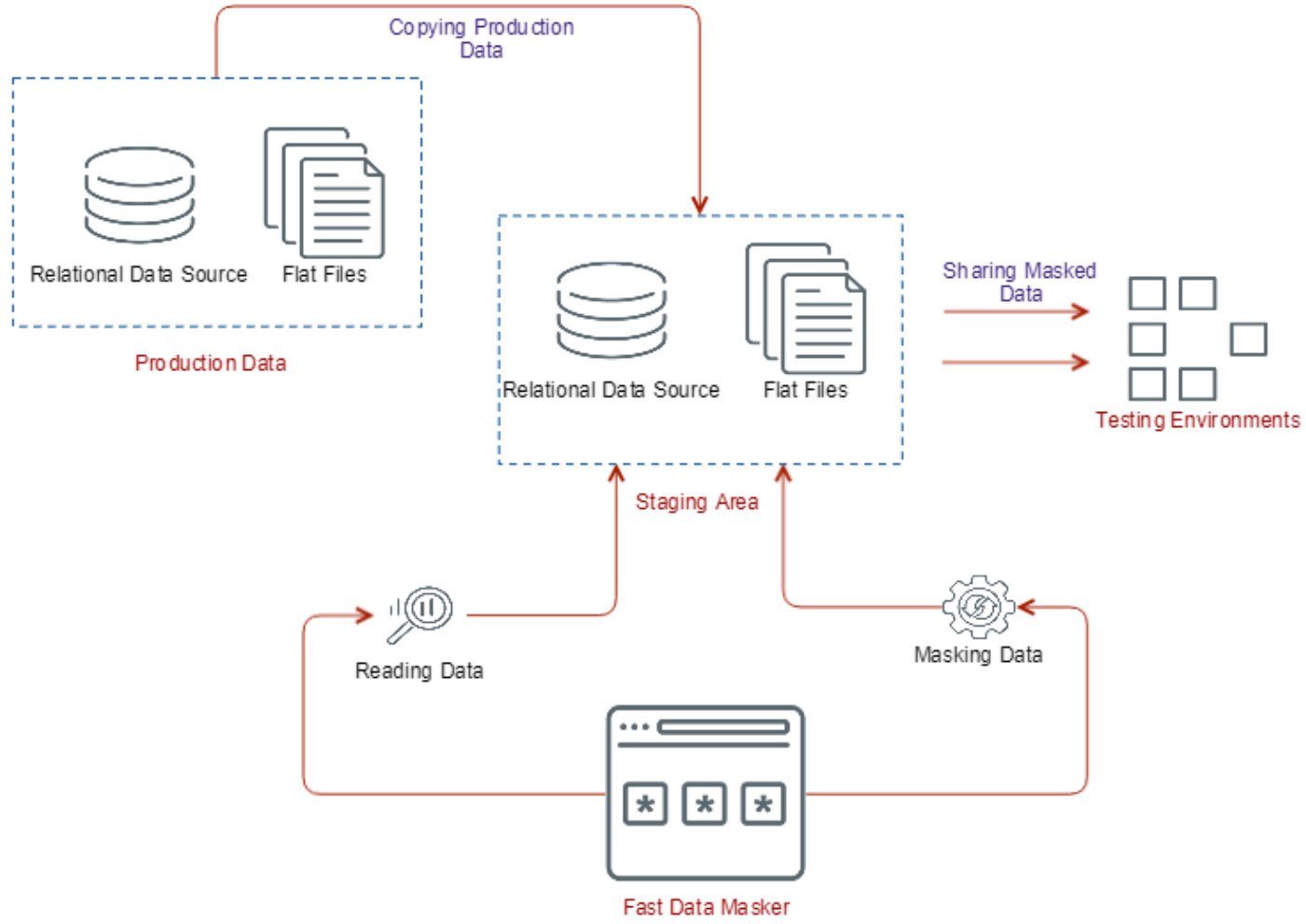
To help organizations address such challenges, CA Test Data Manager provides Fast Data Masker—a high-performance masking application. Fast Data Masker can aid compliance efforts by masking millions of rows of complex sensitive information in minutes. Personal data is replaced with realistic but fictitious values, while maintaining the referential integrity needed for testing across each system. This means that testers and developers do not need to use sensitive content.

The following topics cover the information:

***In-Place* Masking**

Fast Data Masker performs *in-place* masking. The following diagram outlines a typical in-place masking scenario where Fast Data Masker is used:

Figure 6: Getting started FDM main scenario



1. Copy production data to a staging area.
2. Connect Fast Data Masker to the staging database.
3. Mask the identified personal information in the data (staging database).

   NOTE
   For more information about how to discover personally identifiable information in the data, see Data Discovery and Profiling Using Datamaker.
4. Copy the masked, compliant test data to different testing environments as required.

How to Mask Data Using Fast Data Masker

Fast Data Masker can mask the data stored in the following types of data sources:

- Relational data sources
- Flat files

For easier understanding, you can consider masking in Fast Data Masker as a three-step process. Each step in turn includes detailed steps that are relevant to it:

1. **Input:** Connect Fast Data Masker to the data source.

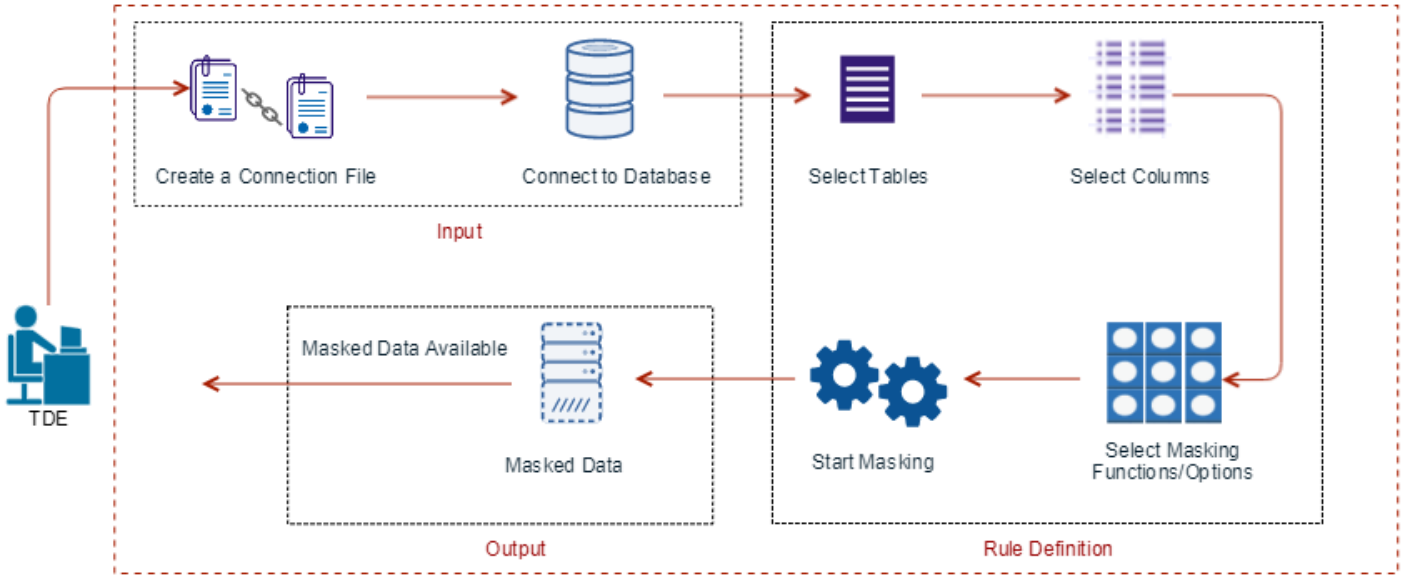
To get started with the masking process in Fast Data Masker, you must first connect your Fast Data Masker instance to the data source that contains the data you want to mask. You establish this connection with the help of a connection file. This connection file includes all the relevant information about the data source. You create this connection in the Fast Data Masker UI. After you create this file, you can use it to connect to the data source whenever you want.

2. **Rule Definition:** Define masking rules and run masking.
   You define all the masking rules by using various available masking functions and options in the Fast Data Masker UI. After you define the rules, you run the masking job to mask the data stored in the data source.
3. **Output:** Verify the masked data.
   You access the data source and verify the output; that is, the masked data. Ensure that you note the pre-masked data before you run the masking job. This allows you to verify the result (masked data) against the original values.

Masking Data in Relational Data Sources

The following diagram shows the detailed step-by-step tasks that a TDE performs while masking the data stored in relational data sources:

Figure 7: Getting started FDM relational data source



Detailed steps are as follows:

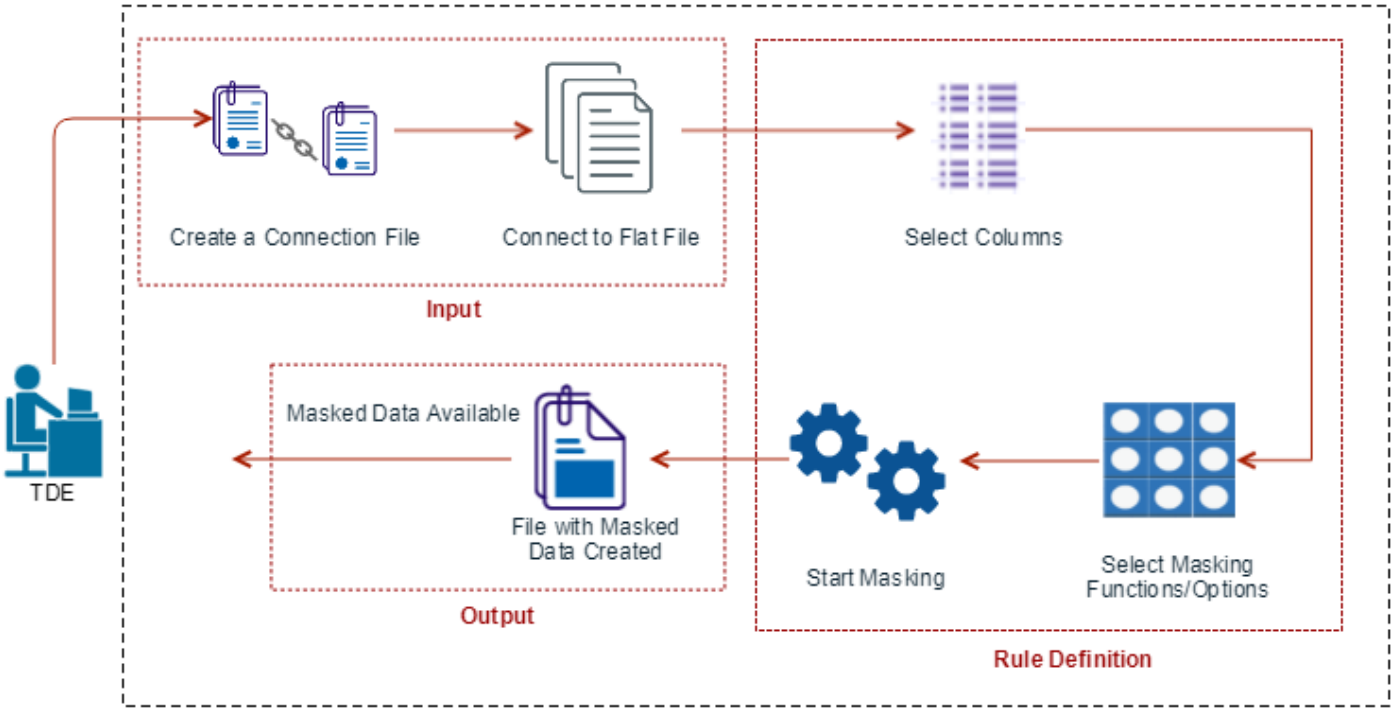
1. Connect Fast Data Masker to the data source.
   - Use existing connection files.
   - Create a new connection file.
   - Manage connection files.
2. Define masking rules and run masking.
   a. Select tables to mask.
   b. Select columns to mask.
   c. Select the mask type.
   d. Select the option to resume masking.
   e. Select other masking options.
   f. Run the masking.
3. Verify the masked data.

For more information about how to perform these tasks, see Mask Data Stored in Relational Databases.

Masking Data in Flat Files

The following diagram shows the detailed step-by-step tasks that a TDE performs while masking the data stored in flat files:

Figure 8: Getting started FDM flat files



Detailed steps are as follows:

1. Connect Fast Data Masker to the data source.
   - Use existing connection files.
   - Create a new connection file.
   - Manage connection files.
2. Define masking rules and run masking.
   a. Select columns to mask.
   b. Select the mask type.
   c. Select masking options.
   d. Run the masking.
3. Verify the masked data.

For more information about how to perform these tasks, see Mask Data Stored in Flat Files.

Specific Use Cases

This section lists specific use cases that are related to Fast Data Masker:

Mask Data in Hadoop

You can use some of the Fast Data Masker masking functions as Hive user-defined functions (UDFs) to mask the structured data stored in Hadoop. You do not use the Fast Data Masker UI in this case. CA TDM provides a JAR file that includes Hive UDFs, which are developed based on a standalone Java masking library. The Java masking library includes Fast Data Masker masking functions. When you execute these Hive UDFs in your Hadoop environment, they perform the defined masking operations and mask the data.

The process to mask structured data stored in Hadoop by using the provided JAR files includes the following steps:

1. Review the files in the masking package.
2. Review the supported masking functions.
3. Deploy the required JAR files and register provided Hive UDFs on the system where Hive is already present.
4. Execute the appropriate Hive UDFs using the Hive query language.

For more information about how to use supported masking functions to mask data stored in Hadoop, see Mask Data Stored in Hadoop.

Use Fast Data Masker in IBM DB2/400 iSeries

To work with Fast Data Masker in iSeries, follow these steps:

1. Understand the key files needed for Fast Data Masker.
2. Verify the JRE setup.
3. Set up the Fast Data Masker directory and files.
4. Create the license file (lic.dat).
5. Run data masking using Fast Data Masker.

For more information, see Work with Fast Data Masker in iSeries (DB2/400).

More Information

This section lists additional resources where you can find related information:

- Supported Data Sources
- Install Product Components
- Install Fast Data Masker on Linux
- Mask Production Data with Fast Data Masker
- Fast Data Masker Best Practices
- Fast Data Masker Troubleshooting

Installing

This section describes how to install Test Data Manager.

The product consists of the following high-level components:

- **CA TDM Portal**
  The CA TDM Portal provides a web interface for many product capabilities. It is available in the following configurations:
  - **A Windows application**
    For more information, see Install TDM Portal for Windows.
  - **An application that runs inside a network of Docker containers**
    Benefits of TDM Portal in Docker include:
    - Simplified deployment, configuration and execution of applications within CA TDM.
    - Scalability. See Scalable masking with Docker.
    - Easier implementation of disaster recovery procedures, due to the fact that its state persists in volumes independent of the Docker container. For more information, see Install TDM Portal for Docker.

      NOTE

      From Test Data Manager 4.8, the TDM Portal service installs the **gtrep** repository tables (if they are not present) or updates **gtrep** (if it is from a previous version) when you start the service. You still need to create the gtrep repository and user before you can install TDM Portal.
- **Repository**
  A database (called **gtrep**) that Test Data Manager uses to store product data. Use the database management tools or the database installer (**ca-tdm-db-installer.exe**) to install this repository.
  For more information, see Install the Repository.

  NOTE

  For Docker-only installations, you can create the **gtrep** repository in a Docker container. This is an option for advanced users. For more information, see TDM Portal Oracle database container.
- **GT Server**
  Container for the primary Test Data Manager components. Use the GT Server installer to install these components.
  For more information, see Install Product Components.

  WARNING

  From Test Data Manager 4.8, Datamaker does not update the gtrep repository. It is therefore necessary to install TDM Portal, or to use the database installer (**ca-tdm-db-installer.exe**) to update gtrep.

  - **Datamaker**
    The primary interface of Test Data Manager that provides a core project and data management interface and several core capabilities, including:
    - Data Discovery
    - Data Subset
    - Data Profiling
    - Data Generation
    - Data Reservation
  - **Fast Data Masker**
    Provides an interface for masking sensitive production data so that you can use it for testing.
  - **Javelin**

Javelin is a workflow engine.

- **Remote Publish**
  The Remote Publish engine handles remote publish jobs instrumented from outside the Datamaker UI. This component is required for many capabilities, including data reservation.
- **Group Job Executor**
  Group Job Executor handles the requests submitted by the testers for data modelling, publishing, testmatching, etc., in batch mode.
- **Test Data Visualizer**
  Provides a customizable graphical representation of your test data coverage. Use this component to find gaps in your coverage that you can fill using data generation and other capabilities.
- **GT EDI**
  Provides an interface for importing and exporting EDI files so that you can work with the data using components like Datamaker and Fast Data Masker.
- **HP ALM Service**
  Provides an integration point for HP ALM.
- **ALM Batch Service**
  Provides integration point for HPALM in batch mode.
- **GTRallyBatch**
  Provides integration point for CA Agile Central in batch mode.
- **Test Data on Demand**
  Deprecated. Use the tester self-service capabilities provided by the CA TDM Portal instead.
- **Portus Job Processor**
  Deprecated. Use the file shredding capabilities provided by the CA TDM Portal instead.

NOTE

These components are **not available** as Docker containers.

- **Mainframe**
  The product mainframe component. To work with mainframe data, use the mainframe integration artifacts to install the mainframe component.
  For more information, see Mainframe Installation and Upgrade.

Supported Data Sources

Test Data Manager supports a wide variety of data sources, both relational and non-relational, for each of its capabilities. This topic summarizes certified and supported data sources by type and capability.

Certified

Indicates that the data source has been fully tested and validated to work.

Supported

Indicates that the data source is expected to work but has not been through a full testing cycle. CA support will strive to resolve any issues which may arise with this data source, but cannot guarantee resolution of issues found.

Here you can find out more about the different data sources that CA Test Data Manager supports:

Relational Data Sources

All database access permissions must be controlled on the DBMS server itself.

Test Data on Demand (TDoD) only supports DSN-less connection profiles.

For more information about supported data types for Dynamic Test Data Reservation, see Configure Dynamic Test Data Reservation Service - Considerations.

For more information on implementation with specific data sources, see Notes on Implementation with Specific Data Sources (links under the **Notes** column also redirect to sections of this page).

Microsoft SQL Server

Database	Dynamic Test Data Reservation - TDM Portal	Data Generation - TDM Portal	Data Generation - Datamaker	Data Masking -TDM Portal	Data Masking - Fast Data Masker	Data Masking - Datamaker	Data Subsetting	Test Match	Virtual Test Data Management	Data Modelling and PII Audit	Distributed with Driver?	Notes
Microsoft SQL Server 2008	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Not Supported	Supported	Yes	Notes
Microsoft SQL Server 2012	Certified	Certified	Certified	Certified	Certified	Supported	Certified	Certified	Certified	Certified See below	Yes	
Microsoft SQL Server 2014	Certified	Certified	Certified	Certified	Certified	Supported	Certified	Certified	Certified	Certified See below	Yes	
Microsoft SQL Server 2014 Express Edition	Certified	Certified	Certified	Not Supported	Certified	Supported	Certified	Certified	Not Supported	Not Supported	Yes	Note on TCP/IP
Microsoft SQL Server 2016	Certified	Certified	Certified	Certified	Certified	Supported	Certified	Certified	Certified	Certified See below	Yes	
Microsoft SQL Server 2016 Express Edition	Certified	Certified	Certified	Not Supported	Certified	Supported	Certified	Certified	Not Supported	Not Supported	Yes	Note on TCP/IP
Microsoft SQL Server 2019	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Yes	
Microsoft Azure SQL	Not Supported	Supported	Not Supported	Supported	Supported	Not Supported	Not Supported	Not Supported	Not Supported	Supported	Yes	

Oracle

Database	Dynamic Test Data Reservation - TDM Portal	Data Generation - TDM Portal	Data Generation - Datamaker	Data Masking -TDM Portal	Data Masking - Fast Data Masker	Data Masking - Datamaker	Data Subsetting	Test Match	Virtual Test Data Management	Data Modelling and PII Audit	Distributed with Driver?	Notes
Oracle 11g (11.2.5, 11.2.0.2)	Certified	Certified	Certified	Certified Mixed-Case masking not supported	Certified Mixed-Case masking not supported	Supported	Certified	Certified	Certified	Certified See below	Yes	Note on Oracle databases
Oracle XE	Certified	Certified	Certified	Not Supported Mixed-Case masking not supported	Not Supported Mixed-Case masking not supported	Supported	Not Supported	Certified	Not Supported	Supported See below	Yes	Note on Oracle databases
Oracle 12c (12.1.1)	Certified	Certified	Certified	Certified Mixed-Case masking not supported	Certified Mixed-Case masking not supported	Supported	Certified	Certified	Certified	Certified See below	Yes	Notes Note on Oracle databases
Oracle 18c	Supported	Supported	Supported	Supported	Supported Mixed-Case masking not supported	Supported	Supported	Supported	Not Supported	Supported	No	Note on Oracle databases
Oracle RAC 11g	Certified	Supported	Supported	Supported Mixed-Case masking not supported	Supported Mixed-Case masking not supported	Supported	Supported	Not Supported	Not Supported	Supported See below	No	Note on Oracle databases
Oracle RAC 12c	Certified	Supported	Supported	Supported Mixed-Case masking not supported	Supported Mixed-Case masking not supported	Supported	Supported	Not Supported	Not Supported	Supported See below	No	Note on Oracle databases
Oracle Cloud	Supported	Supported	Supported	Supported	Supported	Supported	Not Supported	Not Supported	Not Supported	Supported	No	Note on Oracle databases

DB2

Database	Dynamic Test Data Reservation - TDM Portal	Data Generation - TDM Portal	Data Generation - Datamaker	Data Masking -TDM Portal	Data Masking - Fast Data Masker	Data Masking - Datamaker	Data Subsetting	Test Match	Virtual Test Data Management	Data Modelling and PII Audit	Distributed with Driver?	Notes
IBM DB2/400 iSeries V7R1	Certified	Certified	Certified	Certified Mixed-Case masking not supported	Certified Mixed-Case masking not supported	Supported	Certified	Certified	Not Supported	Supported See below	No Note on IBM systems	Notes
IBM DB2 11 for z/OS	Certified	Supported	Supported	Supported	Supported	Supported	Supported	Not Supported	Not Supported	Supported	No Note on IBM systems	Notes
IBM DB2 12 for z/OS	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Not Supported	Not Supported	Supported	No Note on IBM systems	
IBM DB2 UDB 11.1	Certified	Supported	Supported	Supported	Supported	Supported	Supported	Not Supported	Not Supported	Supported See below	No Note on IBM systems	Notes

Teradata

Database	Dynamic Test Data Reservation - TDM Portal	Data Generation - TDM Portal	Data Generation - Datamaker	Data Masking -TDM Portal	Data Masking - Fast Data Masker	Data Masking - Datamaker	Data Subsetting	Test Match	Virtual Test Data Management	Data Modelling and PII Audit	Distributed with Driver?	Notes
Teradata 15.10	Not Supported	Certified	Certified	Not Supported	Not Supported	Certified Note on Teradata databases	Certified	Not Supported	Not Supported	Not Supported	No	Notes
Teradata 16.10	Not Supported	Supported	Certified	Not Supported	Not Supported	Certified Note on Teradata databases	Certified	Not Supported	Not Supported	Not Supported	No	Notes
Teradata 16.20	Not Supported	Supported	Supported	Not Supported	Not Supported	Certified Note on Teradata databases	Supported	Not Supported	Not Supported	Not Supported	No	Notes

Sybase

Database	Dynamic Test Data Reservation - TDM Portal	Data Generation - TDM Portal	Data Generation - Datamaker	Data Masking -TDM Portal	Data Masking - Fast Data Masker	Data Masking - Datamaker	Data Subsetting	Test Match	Virtual Test Data Management	Data Modelling and PII Audit	Distributed with Driver?	Notes
Sybase-IQ 16.0	Not Supported	Certified	Certified	Not Supported	Certified	Not Supported	Not Supported	Not Supported	Not Supported	Not Supported	Yes	
Sybase ASE (SAP ASE)	Not Supported	Not Supported	Supported	Not Supported	Supported	Not Supported	Supported	Not Supported	Not Supported	Not Supported	No	Notes

PostgreSQL

Database	Dynamic Test Data Reservation - TDM Portal	Data Generation - TDM Portal	Data Generation - Datamaker	Data Masking -TDM Portal	Data Masking - Fast Data Masker	Data Masking - Datamaker	Data Subsetting	Test Match	Virtual Test Data Management	Data Modelling and PII Audit	Distributed with Driver?	Notes
PostgreSQL 9.5.4	Not Supported	Supported	Not Supported	Not Supported	Supported	Not Supported	Not Supported	Not Supported	Not Supported	Not Supported	No	Notes
PostgreSQL 10.5	Not Supported	Certified	Not Supported	Supported	Supported	Not Supported	Not Supported	Not Supported	Not Supported	Supported	No	Notes

Other Database Management Systems

Database	Dynamic Test Data Reservation - TDM Portal	Data Generation - TDM Portal	Data Generation - Datamaker	Data Masking -TDM Portal	Data Masking - Fast Data Masker	Data Masking - Datamaker	Data Subsetting	Test Match	Virtual Test Data Management	Data Modelling and PII Audit	Distributed with Driver?	Notes
CA IDMS 19.0	Not Supported	Supported	Supported	Not Supported	Supported	Not Supported	Supported	Not Supported	Not Supported	Not Supported	No	
SAP Hana	Not Supported	Supported	Supported	Supported	Supported	Not Supported	Supported	Not Supported	Not Supported	Supported	No	Notes
Adabas 8.2	Not Supported	Not Supported	Supported Notes	Not Supported	Supported Notes	Not Supported	Not Supported	Not Supported	Not Supported	Not Supported	No	
Informix 12.1	Not Supported	Not Supported	Supported	Not Supported	Supported	Not Supported	Supported	Not Supported	Not Supported	Not Supported	No	Notes
Ingres 10.2	Not Supported	Not Supported	Supported	Not Supported	Supported	Not Supported	Supported	Not Supported	Not Supported	Not Supported	No	Notes
MySQL 5.6	Not Supported	Supported	Supported	Supported	Supported	Not Supported	Supported	Not Supported	Not Supported	Certified See below	No	Notes

MariaDB 10.2.6	Not Supported	Certified	Not Supported	Supported	Supported	Not Supported	Supported	Not Supported	Not Supported	Certified See below	No	Notes
IMS 14	Not Supported	Not Supported	Supported	Not Supported	Supported	Not Supported	Supported	Not Supported	Not Supported	Not Supported	No	
Netezza 7.2	Not Supported	Not Supported	Not Supported	Not Supported	Supported	Not Supported	Not Supported	Not Supported	Not Supported	Not Supported	No	Notes
DataDirect Shadow 7.3	Not Supported	Not Supported	Not Supported	Not Supported	Supported	Not Supported	Supported	Not Supported	Not Supported	Not Supported	No	Notes
SQL Anywhere 17	Not Supported	Not Supported	Not Supported	Not Supported	Supported	Not Supported	Supported	Not Supported	Not Supported	Not Supported	No	Notes
Derby 10.12.1.1	Not Supported	Not Supported	Not Supported	Not Supported	Supported	Not Supported	Not Supported	Not Supported	Not Supported	Not Supported	No	

Notes on Relational Data Sources

Data Discovery and Profiling

Valid on Microsoft SQL Server 2012 / Microsoft SQL Server 2014 / Microsoft SQL Server 2016 / Microsoft SQL Server 2019:

When creating Connection Profiles for Data Discovery and Profiling, leave the Database Name and Schema Name parameters blank to scan all databases.

Valid on Oracle 11g / Oracle XE / Oracle 12c / Oracle RAC 11g / Oracle RAC 12c:

When creating Connection Profiles for Data Discovery and Profiling, leave the Schema Name parameter blank to scan all schemas.

Valid on IBM DB2400 iSeries V7R1 / IBM DB2 UDB 11.1 / MySQL 5.6 / MariaDB 10.2.6:

When creating Connection Profiles for Data Discovery and Profiling, you must enter the specific Database Name you want to scan.

Note on IBM systems

Javelin does not support DB2 drivers on IBM AS-400 and z/OS.

Note on Oracle databases

For all Oracle databases, we only support a user with schema owner permissions.

Note on Teradata databases

For all Teradata databases, you can mask data using GT Subset scripts in Datamaker.

Note on Masking Mixed-Case data sources in TDM Portal / FDM

Masking in Portal / FDM does not support mixed-case data sources (i.e. names of database/schema/table/column) on the following databases:

- All Oracle databases
- DB2 databases on AS-400

Non-Relational Data Sources

The CA TDM Portal lets you work with various file objects and create test data that applications can use to conduct varied testing scenarios. For more information about preparing test data for non-relational data sources, see Prepare Test Data for Non-Relational Data Sources.

Data Source	**Dynamic Test Data Reservation - TDM Portal**	Data Generation - TDM Portal	Data Generation - Datamaker	Data Masking	Data Subsetting	Test Match	Virtual Test Data Management	Data Modelling and PII Audit	Notes
SQL Files	Not Supported	Certified	Certified	Not Supported	Not Supported	Not Supported	Not Supported	Not Supported	
CSV Files	Not Supported	Certified	Certified	Certified	Certified	Not Supported	Not Supported	Not Supported	
Fixed Definition Files	Not Supported	Certified	Certified	Certified	Certified	Not Supported	Not Supported	Not Supported	
XML Files	Not Supported	Certified	Certified	Certified	Not Supported	Not Supported	Not Supported	Not Supported	XML file shredding requires a MS SQL Server connection profile.
Excel Files	Not Supported	Supported	Supported	Supported	Supported	Not Supported	Not Supported	Not Supported	
HTML Files	Not Supported	Not Supported	Supported	Not Supported	Not Supported	Not Supported	Not Supported	Not Supported	
TXT Files	Not Supported	Supported	Supported	Supported	Supported	Not Supported	Not Supported	Not Supported	
VSAM/ ISAM	Not Supported	Not Supported	Supported	Supported	Supported	Supported	Not Supported	Not Supported	
JSON Files	Not Supported	Supported	Supported	Supported	Not Supported	Not Supported	Not Supported	Not Supported	JSON file shredding requires a MS SQL Server connection profile.
Hadoop (Hive)	Not Supported	Not Supported	Not Supported	Certified	Not Supported	Not Supported	Not Supported	Not Supported	

Note: For more information about what is supported in masking for Hadoop, see Mask Data Stored in Hadoop.

Notes on Implementation with Specific Data Sources

This page details important notes on the implementation of CA Test Data Manager with the specified data sources.

Microsoft SQL Server 2008

Microsoft SQL Server 2008 support is based on the Proof of Concept (PoC) executed with customers and supports limited use cases only.

Microsoft SQL Server Express 2014/2016

WARNING

TCP/IP is **not** enabled in Microsoft SQL Server Express (2014 and 2016) by default.

Ensure that you use the SQL Server Configuration Manager to enable the TCP/IP protocol before use with CA TDM. You should use a known port for connection - if you do not have another installation of SQL Server on your machine, use **1433** (default SQL Server port for many applications).

Oracle 12c

vTDM supports the following Oracle 12c databases for automatic attachment of clones:

- **Non-Container Database (CDB)** Supported same as Oracle 11g (Linux) Enterprise edition.
- **Single tenant CDB configuration**
  For a container database with a single pluggable database, you can create clones in the pluggable database (PDB) only.
- **Multitenant CDB configuration**
  For a container database with multiple pluggable databases, you can create clones in the pluggable databases (PDBs) only.

NOTE

Creating clones in system database and root container is not supported.

IBM DB2400 iSeries V7R1

For masking, ensure that the **jt400.jar** file is available at the following location:

- `C:\Program Files\Grid-Tools\FastDataMasker\lib\`

For subsetting, ensure that the **jt400.jar** file is available at the following location:

- `C:\Program Files (x86)\Grid-Tools\GTDatamaker\lib\`

For CA TDM Portal, ensure that the **jt400.jar** file is available at the following location:

- `C:\Program Files\CA\CA Test Data Manager Portal\tomcat\jdbc-drivers`

IBM DB2 11 for z/OS

Download the file **Common.jar** from the vendor and place it in the following folders:

- **CA TDM Datamaker**
  `C:\Program Files (x86)\Grid-Tools\GTDatamaker\dplib\`
- **CA TDM Portal**
  `C:\Program Files\CA\CA Test Data Manager Portal\tomcat\jdbc-drivers`

If you are using **CA TDM version 3.2.1** or later, replace **db2jcc.jar** file with **db2jcc4.jar** file.

- If both db2jcc.jar and db2jcc4.jar files are present, DB2 will default to the older **db2jcc.jar**.
- The older driver only works with **Java 1.7** and below.

If you are using **CA TDM version 3.2** or earlier, ensure **db2jcc.jar** file is available at the below path:

- **CA TDM Datamaker**
  `C:\Program Files (x86)\Grid-Tools\GTDatamaker\lib\`
- **CA TDM Portal**
  `C:\Program Files\CA\CA Test Data Manager Portal\tomcat\jdbc-drivers`

For masking, ensure that the **db2jcc4.jar**, **db2jcc4_license_cu.jar**, and **db2jcc4_license_cisuz.jar** files are available at the following location:

- `C:\Program Files\Grid-Tools\FastDataMasker\lib\`

For subsetting, ensure that the **db2jcc4.jar**, **db2jcc4_license_cu.jar**, and **db2jcc4_license_cisuz.jar** files are available at the following location:

- `C:\Program Files (x86)\Grid-Tools\GTDatamaker\lib\`

For CA TDM Portal, ensure that the **db2jcc4.jar**, **db2jcc4_license_cu.jar**, and **db2jcc4_license_cisuz.jar** files are available at the following location:

- `C:\Program Files\CA\CA Test Data Manager Portal\tomcat\jdbc-drivers`

Run the db2licm command from the command prompt to apply the license as follows:

```
db2licm -a "C:\Users\Username\Desktop\10.5_DB2ConnectEE_License\db2consv_ee.lic"
```

> **NOTE**
>
> Ensure that the license file path is local and does not point to a network drive.

IBM DB2 UDB 11.1

For more information about the JDBC drivers, refer to Supported Data Sourcesnotes.

Sybase ASE (SAP ASE)

Download the *jconn4.jar* driver from https://support.sap.com/en/my-support/software-downloads.html

> **NOTE**
>
> Sybase ASE support is based on the Proof of Concept (PoC) executed with customers and supports limited use cases only.

Teradata 15.10 / Teradata 16.10 / Teradata 16.20

If you create a Teradata connection profile in CA TDM Portal, and you publish from Datamaker using this connection profile (the Enterprise or Remote Enterprise publish option), then only the Teradata ODBC driver is supported.

Download the files **terajdbc4.jar** and **tdgssconfig.jar** from the vendor and place it in the following folder:

- `C:\Program Files (x86)\Grid-Tools\GTDatamaker\lib\`

When using the JDBC driver to create a connection profile in the CA TDM Portal, grant the Teradata user the following additional permissions:

```
GRANT Select ON "UDTInfo" TO "<username>";
```

Adabas 8.2

For Data Generation in Datamaker and Data Masking, we support use cases on a per Proof of Concept (POC) basis.

Informix 12.1

For Informix to work, you must set the following environment variable on the system where Datamaker is installed: **DELIMIDENT=y**.

This setting causes the Informix driver to properly interpret double quotes as delimiters.

Ingres 10.2

Download the file **edbc.jar** from http://esd.actian.com and place it in the following folder:

```
C:\Program Files (x86)\Grid-Tools\GTDatamaker\lib\
```

PostgreSQL

Download the latest driver from https://jdbc.postgresql.org/. Place it in one or both of the following folders:

- For use with **TDM Portal**: `C:\Program Files\CA\CA Test Data Manager Portal\tomcat\jdbc-drivers`
- For use with **Fast Data Masker**: `C:\Program Files\Grid-Tools\FastDataMasker\lib\`

MySQL 5.6

For CA TDM Portal, download the driver **mysql-connector-java-5.1.38-bin.jar** from http://dev.mysql.com and place it in the following folder:

```
C:\Program Files\CA\CA Test Data Manager Portal\tomcat\jdbc-drivers
```

MariaDB 10.2.6

The CA TDM Portal is certified with **connector-java-2.0.3** driver. Download the driver from https://downloads.mariadb.org/connector-java/ and place it in the following folder:

- `C:\Program Files\CA\CA Test Data Manager Portal\tomcat\jdbc-drivers`

  NOTE

  Find and reserve is not supported for MariaDB.

Netezza 7.2

Download the driver from https://github.com/dbfit/dbfit/releases.

DataDirect Shadow 7.3

Download the file **base.jar** from https://www.progress.com and place it in the following folders:

- `C:\Program Files (x86)\Grid-Tools\GTDatamaker\dplib\`
- `C:\Program Files (x86)\Grid-Tools\GTDatamaker\lib\`
- `C:\Program Files\Grid-Tools\FastDataMasker\lib\`

SQL Anywhere 17

Download the driver from http://scn.sap.com.

For more information about supported data sources in CA Test Data Manager, see Supported Data Sources.

SAP Hana

Download the driver from developers.sap.com.

For more information about supported data sources in CA Test Data Manager, see Supported Data Sources.

Install Test Data Manager

You can use Test Data Manager in the following ways:

- **Windows**
  - **Full Application suite**
    Install Test Data Manager as a windows desktop application to get its full functionality. See Windows Installation Procedure.
  - **TDM Portal only**
    From TDM 4.8, you can install the TDM Portal only, for the fastest access to the core Test Data Manager functionality. For more information, see Install TDM Portal for Windows.
- **As a docker container** (TDM Portal only)
  Use the docker application to allow you to run Test Data Manager Portal as a container within this environment. See TDM Portal in docker Installation Procedure.

Windows Installation Procedure

Follow these steps:

1. Review and meet system requirements
2. Install the repository
3. Install TDM Portal for Windows
4. Install product components
5. Connect Datamaker to the repository
6. Obtain and activate product licenses
7. Perform repository maintenance
8. Connect to other data sources
9. (Optional) Secure your Test Data on Demand configuration
10. (Optional) Enable Integrated Security for TDM Portal
11. (Optional) Install mainframe components
12. (Optional) Scalable masking with Docker

TDM Portal in Docker Installation Procedure

Follow these steps:

1. Install Docker application (see https://docs.docker.com/install/overview/)
2. Download the appropriate Docker installation package from support.ca.com. This can be either:
   - Script package - run a script to download Docker images and load them into Docker.
   - Image Bundle - run these images to start TDM Portal services.
   - Image Kit - use this kit to build your own images.
3. Install the gtrep repository

   NOTE

   You can skip this step and create a TDM Portal Oracle database container. This is for Advanced users, and this container should not be used with production data.
4. **(Image Kit only)** Build the Docker images you need. For more information, see Build your own Docker images.
5. Configure your docker-compose files. For more information on the TDM Portal Docker installation process, see Install TDM Portal for Docker.

System Requirements

Test Data Manager has the following installation prerequisites:

Hardware Requirements

Test Data Manager supports the following hardware:

- Physical and virtual Windows machines.
- 32-bit systems
  Note: Service layer components require 64-bit. These components include CA Agile Requirements Designer.
- 64-bit systems
  Note: To create DSN entries on 64-bit systems, use C:\Windows\SysWOW64\odbcad32.exe.
- Fast Data Masker supports installation on Red Hat Linux 6.0 to 7.0

A centralized server that supports multiple simultaneous terminal servers (RDS) simplifies maintenance. Alternatively, each user can install the software on a laptop or workstation and can connect to a shared repository. For production environments, a DBA managed centralized database server is best for the test data repository. For training and trial purposes, you can use a locally managed database.

The following list shows system requirements for various Test Data Manager server types:

- **General Workstation**
  OS: Windows 7 Professional or higher
  Memory: 3 GB
  Free Space: 20 GB
  CPU: > 2 GHz
  Network: 100 Mbit
- **Server** (medium usage, up to 5 concurrent users)
  OS: Windows Server 2012 R2 Standard Edition or higher
  Memory: 16 GB
  Free Space: 20 GB
  CPU: 1x Quad core
  Network: 1 Gbit
- **Server** (high volume or terminal services)
  OS: Windows Server 2012 R2 64-bit Standard Edition or higher
  Memory: 32 GB
  Free Space: 20 GB
  CPU: 2x Quad Core, 2.7GHz, 6 MB L3 Cache, 1 GHz HyperTransport or equivalent
  Network: 1 Gbit

Test Data Manager Operating System Support

Test Data Manager (TDM Portal, Repository, GT Server components) is tested on the following operating systems:

- Microsoft Windows Server 2019
- Microsoft Windows Server 2012 R2 (64-bit)
- Microsoft Windows Server 2016
- Microsoft Windows 10 64-bit
- Microsoft Windows 7 64-bit

The Portal may work on other versions, but they have not been tested.

Database Requirements

Test Data Manager requires a repository to store product metadata. You can also use the repository to store test data. The repository can be on the same server as Test Data Manager, or can be a remote server. If you use remote server, ensure that the server has the memory requirement as recommended in the Hardware Requirements section. For production environments, we recommend a dedicated database server for the repository that is backed up and monitored.

Test Data Manager supports the following databases for the repository:

Database	CA Datamaker	CA TDM Portal	Notes
Microsoft SQL Server 2012, 2014, 2016, 2017, 2019 Microsoft SQL Server Express 2014, 2016, 2017, 2019	YES	YES	-
Oracle 11g, 12c, 18c	YES	YES	-
Oracle 11g XE	YES	YES	Oracle 11g XE does not support Java. If you want to use the scramble functionality of Data Subset, Oracle 11g standard version or higher is required.
Oracle RAC	YES	NO	-

The repository database and server also requires the following:

- Allow 5 GB of disk space for the repository database installation. Growth depends on the volume of test data that you store in the repository. Assume 5 GB to 20 GB in the first year.
- For database access from a remote system, Oracle databases require an Oracle client with the TNSNAMES.ora file. All other database platforms require an ODBC driver software and system DSN entries.

  NOTE
  On 64-bit systems, use C:\Windows\SysWOW64\odbcad32.exe to create DSN entries.
- Oracle 12c can be installed in one of two ways:
  - As a plain database instance similar to 11g.
  - As a container database.
    A container database is used to create a multi-tenant database system. A container contains a set of "pluggable" databases which can be run inside the container as if they are separate instances. DBAs can move pluggable databases from container to container in a simple manner when they need to move to different hardware or storage. It also simplifies administration in various ways. See Introduction to the Oracle Multitenant Architecture for more details.
    Always install CA TDM databases and the repository in a pluggable database and not in the "root" or container database itself. From the point of view of CA TDM, a pluggable database is a regular instance of Oracle with its own service name. For example, if you create a pluggable database called "plug1" then it will appear as a service called "plug1.mydomain.com" with a tnsname of "plug1". The container itself has a tnsname of "orcl" by default. To install gtrep on plug1, edit the install script (repository.bat) and set the destination (environment variable GT_TNS) to "plug1.mydomain.com". Similar considerations apply to the sample databases and scramble.
- For Microsoft SQL Server, ensure that you do the following when you install the database or afterwards as configuration changes:
  - Use mixed mode authentication
  - Enable TCP-IP
  - Enable and start the SQL Server Browser service

Test Data Manager supports additional databases as source and target connections to connect to and publish test data. See Supported Data Sources.

NOTE
All database access permissions must be controlled on the DBMS server itself.

Software Requirements

The following software is required for Test Data Manager installation.

- Java JRE: AdoptOpenJDK JRE 8
- .Net framework 4.5.2 (64-bit) or higher
- Microsoft Visual C++ 2010 SP1 32-bit and 64-bit
- Microsoft Visual C++ Redistributable for Visual Studio 2012 32-bit and 64-bit 11 or higher
- Microsoft SQL Server Native Client 2012 64-bit
- ODP4.NET Deployer 32-bit and 64-bit 4.112.3 or higher

All software prerequisites are listed in the GT Server installer. The installer checks your system for each prerequisite and installs any software that is not present.

Browser Support

CA TDM Portal

The CA TDM portal supports the following browsers:

- Microsoft Internet Explorer 11 or higher
- Google Chrome 57 or higher
- Mozilla Firefox 54 or higher

Data Source Support

CA TDM requires specific JAR files to support different types of data sources. After successfully installing Test Data Manager, based on the data source you use, you must copy the required JAR files to the installation folder to fully support various functionality with in CA TDM (Data Profiler, Data Subset and Fast Data Masking).

The Test Data Manager Installer does not include the following JAR files and need to manually place in the below specified folders to support the corresponding data source:

- **base.jar**
  This JAR file is required if you are using Data Direct. Copy this file to the following folders:
  - C:\Program Files (x86)\Grid-Tools\GTDatamaker\dplib\
  - C:\Program Files (x86)\Grid-Tools\GTDatamaker\lib\
  - C:\Program Files\Grid-Tools\FastDataMasker\lib\
- **terajdbc4.jar and tdgssconfig.jar**
  These JAR files are required if you are using TeraData. Copy this file to the following folder:
  - C:\Program Files (x86)\Grid-Tools\GTDatamaker\lib\
- **Common.jar**
  This JAR file is required if you are using IBM DB2. Copy this file to the following folders:
  - C:\Program Files (x86)\Grid-Tools\GTDatamaker\dplib\
- **db2jcc.jar**
  This JAR file is required if you are using IBM DB2 and **JRE 1.7** or earlier. Copy this file to the following folder:
  - **C:\Program Files (x86)\Grid-Tools\GTDatamaker\lib**
- **db2jcc4.jar**
  This JAR file is required if you are using IBM DB2 and **JRE 1.8** or later. Replace **db2jcc.jar** file with **db2jcc4.jar** file in the following folder:
  - **C:\Program Files (x86)\Grid-Tools\GTDatamaker\lib**
    Note: If both db2jcc.jar and db2jcc4.jar files are present, DB2 will default to the older db2jcc.jar. The older driver only works with Java 1.7 and earlier.
- **edbc.jar**
  This JAR file is required if you are using Ingres. Copy this file to the following folders:
  - C:\Program Files (x86)\Grid-Tools\GTDatamaker\lib\

- **K9sybase.jar**
  This JAR file is required if you are using Sybase. Copy this file to the following folders:
  – C:\Program Files (x86)\Grid-Tools\GTDatamaker\lib\

For more information, see Supported Data Sources.

Integration Support

Integration with the following versions of different applications is supported:

CA Service Virtualization

- 10.1

CA Agile Requirements Designer

- 2.5.5

HP Application Lifecycle Management (HP ALM)

- 12.2
- 12.5

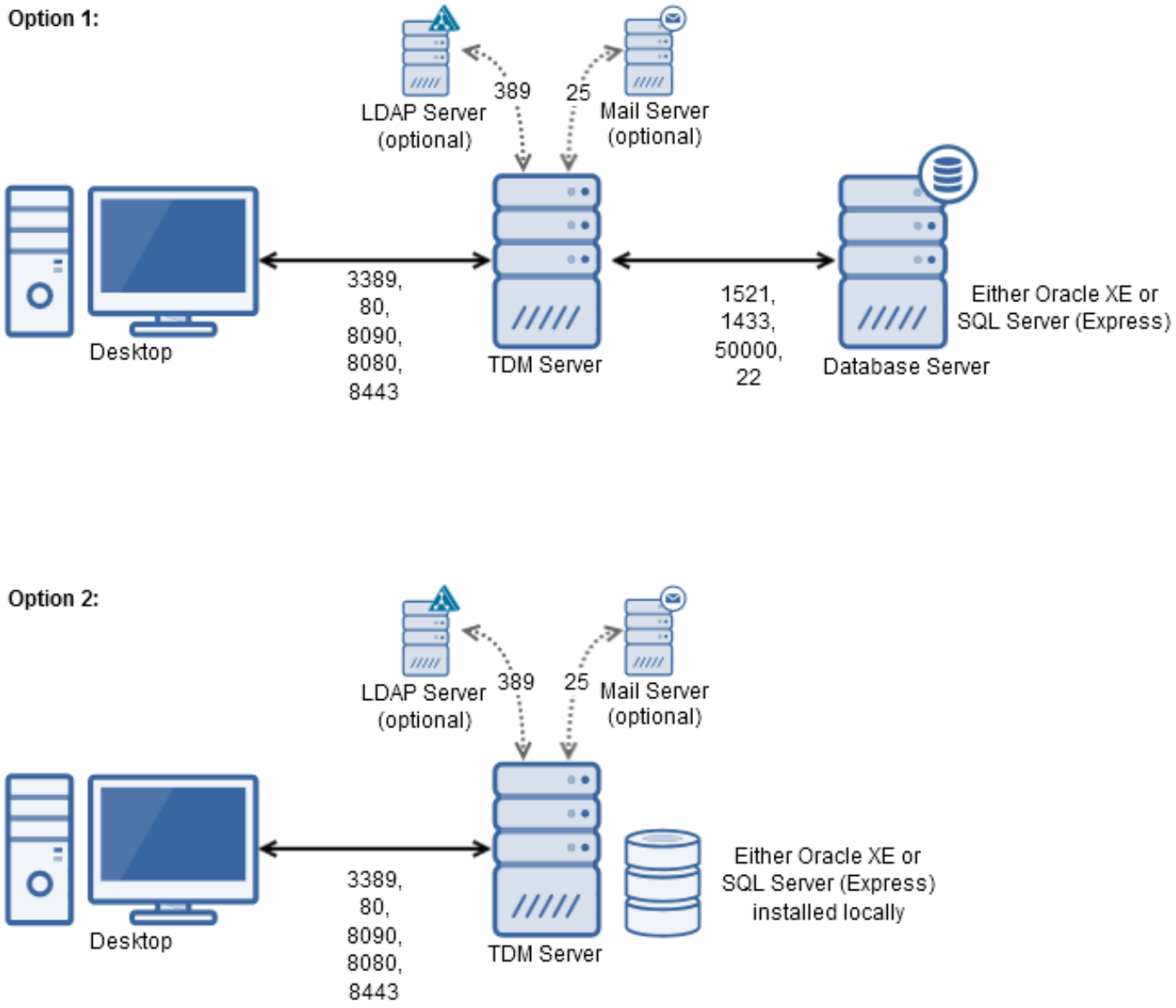
CA Agile Central

- Cloud instance

Open Ports

Either install Oracle XE or SQL Server Express locally on the TDM Server; or open ports between the TDM Server and the database server.

Figure 9: TDM server and database server architecture options



Communications between the Desktop and the TDM Server

Open the following ports:

- RDP: Port 3389
- Browser: Ports 80, 8090, 8080, 8443

Communications between the TDM Server and the Database Server

Open the following ports:

- Oracle: 1521 or alternate port specified by DB Server
- SQL Server: 1433 or alternate port specified by DB Server
- DB2: 50000 or alternate port specified by DB Server
- Other DBs: Refer to Database provider documents or organizational architecture standards documents
- File transfer between Mainframe and Windows: port 21 for FTP or 22 for SFTP

Optional Communications ports

Open the following ports, if applicable:

- Email notifications: SMTP port 25 from TDM Server to Mail Server
- LDAP configuration: LDAP port 389 from TDM Server to LDAP Server

Minimum Resolution

The minimum resolution for the CA TDM UI, including CA TDM Portal and Fast Data Masker, must be greater than 1600x900 pixels.

Server Privileges

The following privileges are required to manage Test Data Manager servers:

- Administrator privileges on the installation server
- Privileges to copy installation files from the Internet or from other network devices

Language Support

CA Test Data Manager supports the use of data in any language that can be represented in UTF-8 character encoding. UTF-8 support requires that the repository is configured for UTF-8 encoding instead of ASCII.

Install the Repository

The Repository (**gtrep**) is a database for storing Test Data Manager data. The Repository is necessary for TDM to function.

You can install the repository in one of the following ways:

- Use the repository installer utility.
- Use native database management capabilities.
- (Oracle on Linux only) Use the TDM Portal Tools container in Docker.

  TIP

  We recommend that you use the repository installer utility to install the repository.

Use your **database** credentials to install the repository. Do not use your Windows credentials.

Use the Database Installer Utility to Install the Repository

The database installer utility provides an automated method for installing various Test Data Manager databases, including the repository. The utility is available in the repository installation kit that you download as a part of the product image.

1. Download and extract the files in the repository installation kit.
2. *Open a command prompt, and navigate to \DB-install-kit-<version>\ca-tdm-db-install-kit.*
3. Run a command similar to the following using valid values for your database server:
   Microsoft SQL Server

```
ca-tdm-db-installer install --dbname gtrep --dbms sqlserver --server mysqlserver.acme.com --dbmsuser sa --
dbmspassword gtsecret123 --dbuser gtrep --dbpassword 12ABcd!%
```

Oracle

```
ca-tdm-db-installer install --dbname gtrep --dbms oracle --server myoracle.acme.com --dbmsuser system --
dbmspassword gtsecret123 --servicename  myoracle.acme.com --dbuser gtrep --dbpassword 12ABcd!%
```

The tool installs the main repository database on the database server that you specify. In this example, it installs the database with a new user gtrep. The --force flag is not set, so if the database already exists, the tools fails and returns `1` . You can use the --testdb command to test for database existence first.

Database Installer Utility Command Line Syntax

The following parameters are available when you use the install action:

ca-tdm-db-installer install

[--server server[\instance]]

[--dbms <*oracle* / *sqlserver*>]

[--dbmsuser name] [--dbmspassword pw]

[--dbname <name of db to install>]

[--dbuser name] [--dbpassword pw]

[--force]

[--servicename <oracle service name>] [--port n]

[--role <login role>] [--tnsname <tns name>]

[? help]

server
Specifies the server name where you want to install the repository. If you omit this property, the installer uses the local host by default.
dbms
Specifies the database type for the repository.
Values: oracle, sqlserver.
dbmsuser, dbmspassword
Specifies a user with administrative rights to access the database where you want to install the repository. If you omit this property, the installer attempts to log in to the database with the current user privileges.
Example: If you configure Microsoft SQL Server to use Windows authentication while logged in as an administrator, the current user has the required privileges. Enter the password for this user in the dbmspassword property.

NOTE

Unified login only works for Microsoft SQL Server and not Oracle.

WARNING

For Oracle, verify that the specified `dbmsuser` has the following roles and privileges:

Roles

- SELECT_CATALOG_ROLE

System privileges

- ALTER ANY PROCEDURE
- ALTER ANY TABLE
- CREATE ANY INDEX
- CREATE ANY PROCEDURE
- CREATE ANY SEQUENCE
- CREATE ANY TABLE
- CREATE ANY VIEW
- CREATE SESSION
- CREATE USER
- DROP USER
- GRANT ANY PRIVILEGE
- GRANT ANY ROLE
- INSERT ANY TABLE
- LOCK ANY TABLE

dbuser, dbpassword

Specifies the user name for the database to be created. Enter the password for this user in the dbpassword property. For Oracle, the dbuser user is usually the same as the database name, but need not be. In fact, several repositories may be installed under different user names.

NOTE

Microsoft SQL Server and Oracle users differ as follows

- In Microsoft SQL Server, a user is an independent entity which can access databases depending on permissions granted by the dba. For example, fred and jim are users and gtrep1 and gtrep2 are databases.
- In Oracle, a user owns tables and other entities. Users and schemas are synonymous. For example, users fred and jim can both own separate repository tables.

dbname
Specifies the name of the directory in the kit that contains the repository, which also becomes the name of the TDM repository (the name of the repository folder in the kit is **gtrep**).

WARNING

For SQL Server installations (i.e. where the value of --dbms is **sqlserver**), this value must be **gtrep**.

force
Drops and re-installs the database if the database already exists. Omit this property unless you upgrade or replace an existing repository.The force option must never be used to upgrade an existing repository because its purpose is to destroy the existing one and replace it. The installer does not support upgrade. Upgrade your repository through TDM Portal.

tnsname

(Oracle only) Specifies the tns name of the Oracle database. This is used as an alternative to server, service and port.

role

(Oracle only) Specifies the role that the Oracle user is to take. If the user logs as system then this can be omitted as system already has the necessary roles. If the user logs as some other user, then specify --role sysdba.

servicename
(Oracle only) Specifies the target Oracle database service name.
port
Specifies the port for the target Oracle or Sqlserver database:
Default: 1521 (Oracle)

Default: 1433 (Microsoft SQL Server)

Install the Repository Manually

You can install the repository manually. To achieve this, you need to create the **gtrep database**, and the **gtrep user** for the gtrep database.

> **NOTE**
>
> To make changes to a database, you must be logged in as an administrator (default: **sa** for SQL Server, **system** for Oracle)

Install the Repository Manually for Microsoft SQL Server

Prerequisites

Ensure that you have an installation of Microsoft SQL Server and that you have privileges to create the following items:

- A database
- Tables
- Views
- Functions
- Procedures
- Indexes
- Primary and foreign keys
- Constraints

Also ensure that you have access to Microsoft SQL Server Management Studio or Enterprise Manager. If you have fulfilled these conditions, you can create **gtrep** manually.

Installation

Complete the following steps to install the repository on a Microsoft SQL Server machine. You must be logged in as an administrator to complete these steps.

Follow these steps:

1. Download and extract the files in the repository installation kit. For this example, we assume that the extracted files are in the directory **C:\TDM\Repository Installer**
2. Open Microsoft SQL Server Management Studio.
3. Create the **gtrep database**
   a. Open the file **C:\TDM\Repository Installer\ca-tdm-db-install-kit\sqlserver\gtrep\gtrep-schema.sql** in Microsoft SQL Server Management Studio.
   b. Click **Execute**.
   The query creates the empty database **gtrep**.
4. Create the **gtrep user**
   a. Open the file **C:\TDM\Repository Installer\ca-tdm-db-install-kit\sqlserver\create-db-user.sql** in Microsoft SQL Server Management Studio.
   b. Define the variables **dbuser**, **dbpassword** and **dbname**.

      > **NOTE**
      >
      > Variable **dbname** must be **gtrep**, to match the database created in Step 3. The recommended value for dbuser is also **gtrep**.

   c. Click **Execute**.

The query creates the user **gtrep** for the database **gtrep**.

NOTE

Other sample databases are available in the TDM file package you download. For information on how to install these databases, see Install sample databases.

Install the Repository Manually for Oracle

This section describes how to configure and install the repository on an Oracle database on a Windows or Linux system.

Prerequisites

Ensure that you have installed Oracle and SQL Plus (sqlplus), and that you have privileges to create the following items:

- Databases
- Tables
- Views
- Functions
- Procedures
- Indexes
- Primary and foreign keys
- Constraints

Windows-specific Requirements

A mismatch of Oracle versions is a common problem during the initial setup of the repository on Windows. Use the following steps to check for an Oracle instance installed on the path:

- From a command prompt, issue the command PATH. Look for an existing Oracle in the path variable.
- Issue a sqlplus command and see if you are prompted for a username and password.
- Issue a tnsping command to see if you can ping an existing Oracle instance.

You must meet the following requirements for an Oracle Windows repository:

- Test Data Manager requires a 32-bit client called Oracle Instant Client. This is provided with the product installation.
- Modify the TNSNAMES.ORA file to include the databases that you plan to connect to. Alternatively, you can use EZConnect strings (//server:port/dbservice).
- A 32-bit client is required for ODBC. The manager for this is located in %windir%\sysWOW64.
- DB2 Connect on 64-bit sets up ODBC sources that do not work with Test Data Manager. These sources must be recreated with a unique name using %windir%\syswow64\odbcad32.exe, instead of %windir% \system32\odbcad32.exe. After you create the source, move over all of the advanced settings.
- In some cases, the sqlnet.ora file needs to be updated to work with the installation batch scripts. Locate and open the sqlnet.ora file in the oracle directory. Change the SQLNET.AUTHENTICATION_SERVICES = (NTS) to SQLNET.AUTHENTICATION_SERVICES = (NONE).

Installation (Windows and Linux)

Complete the following steps to install the repository on an Oracle (Windows or Linux) machine. You must be logged in as an administrator to complete these steps.

1. Download and extract the files in the repository installation kit. For this example, we assume that the extracted files are in one of the following directories:
   - Windows:**C:\TDM\Repository Installer**
   - Linux:**\home\Repository Installer**

2. Define the following parameters in the files **\Repository Installer\ca-tdm-db-install-kit\oracle\create-db-user.sql** and **\Repository Installer\ca-tdm-db-install-kit\oracle\gtrep\gtrep-pre-config.sql**:
   a. **dbusername** (recommended value: **gtrep**)
   b. **dbpassword**
3. Perform the following operations from the Oracle SQL command line (sqlplus):
   a. To create the **gtrep user**, execute the file **\Repository Installer\ca-tdm-db-install-kit\oracle\create-db-user.sql**
   b. To create the **gtrep database**, execute the file **\Repository Installer\ca-tdm-db-install-kit\oracle\gtrep\gtrep-pre-config.sql**.

      NOTE

      Other sample databases are available in the TDM file package you download. For information on how to install these databases, see Install sample databases.

Install Sample Databases

Test Data Manager file packages include sample databases **creditcard**, **orders**, **travel** and **scramble**, which you can use to experiment with TDM's functionality, and which are necessary for some worked examples in the documentation. This page explains how to restore these databases from backup files.

Install sample databases in Microsoft SQL Server

Prerequisites

Before you start, ensure that you have an installed Microsoft SQL Server and that you have privileges to create the following items:

- A database
- Tables
- Views
- Functions
- Procedures
- Indexes
- Primary and foreign keys
- Constraints

Also ensure that you have access to Microsoft SQL Server Management Studio or Enterprise Manager.

Restore a sample database

To install a sample database (i.e. to Restore a database from a backup file), follow these steps:

1. Download and extract the files in the repository installation kit. For this example, we assume that the extracted files are in **C:\TDM\Repository Installer**Each database's backup files are in their own subdirectory under **C:\TDM\Repository Installer\ SQL_Server_Install_Kit\Databases**
2. For the database you want to install (for example, **Credit_Card**), navigate to the relevant directory (for example, **C:\TDM\Repository Installer\ SQL_Server_Install_Kit\Databases\ Credit_Card**).
3. Copy the .bak file for that database (for example, **creditcard.bak**), to your SQL Server file system backup directory, which by default is:**C:\Program Files\Microsoft SQL Server\MSSQL<version>.MSSQLSERVER\MSSQL\Backup**
4. Open Microsoft SQL Server Management Studio.
5. In the **Object Explorer** pane, right-click **Databases** and select **Restore Database**.
   The **Restore Database** dialog opens.

6. Under the **Source** section, select **Device**, and click the **'...'** button to the right of the field.
   The **Select backup devices** dialog opens.
7. From the **Backup media type** drop-down, select **File**.
8. Click **Add**.
   The **Locate Backup File - <server_name>** dialog opens. By default, it opens in the directory where you coped the .bak file in step 3 (i.e. **C:\Program Files\Microsoft SQL Server\MSSQL<version>.MSSQLSERVER\MSSQL \Backup**).
9. Click **OK**.
   The dialog closes.
10. Select the database you want to restore, from the **Database** drop-down under **Device** (in the **Source** section).
11. Select the database into which you want to restore the backup, from the **Database** drop-down in the **Destination** section.
12. Check that the **Restore** check-box (next to the database name, in the **Restore plan section**) is checked.
13. Click **OK**.
    SQL Server Management Studio restores the selected database. The database appears in the **Object Explorer** pane, under **Databases**.
14. Return to the backup directory (for example, **C:\TDM\Repository Installer\ SQL_Server_Install_Kit\Databases \ Credit_Card**), and run the <database>_user.sql script (for example **creditcard_user.sql**) in SQL Server Management Studio.

Install sample databases in Oracle

Install sample databases in Oracle on Windows

Prerequisites

Before you start, ensure that you have an installed Oracle, SQL Plus (**sqlplus**), and Data Pump Import (**impdp**) and that you have privileges to create the following items:

- A database
- Tables
- Views
- Functions
- Procedures
- Indexes
- Primary and foreign keys
- Constraints

Restore a sample database

To install a sample database (i.e. to Restore a database from a backup file), follow these steps:

1. Download and extract the files in the repository installation kit. For this example, we assume that the extracted files are in **C:\TDM\Repository Installer**Each database's backup files are in their own subdirectory under **C:\TDM \Repository Installer\Oracle_Install_Kit\Databases**
2. For the database you want to install (for example, **Credit_Card**), navigate to the relevant directory (for example, **C: \TDM\Repository Installer\ Oracle_Install_Kit\Databases\ Credit_Card).**
3. In the file **<database_name>.bat** (for example, **creditcard.bat**), change the following variables if necessary:
   – **GT_SYS_USER**
     Specifies the system user for the Oracle instance.
   – **GT_SYS_PASSWORD**

Specifies the password for the Oracle system user.

– **GT_TNS**
Specifies the TNS alias for the target database.

4. Run the **<database_name>.bat** file.
The batch file creates the database in your Oracle instance.

Install sample databases in Oracle on Linux with Docker

To install sample databases on an Oracle database, use the Sample Database Creation Tool, part of the TDM Portal Tools container for Docker. This requires the use of Docker for Linux.

Install TDM Portal for Windows

As a systems administrator, review this article to understand how to install the CA TDM Portal. The CA TDM Portal is an intuitive browser-based user interface that lets you configure and execute test data management tasks. The CA TDM Portal installer installs and configures the Portal to connect to and use an existing CA TDM repository. After you complete the installation, you can log into the CA TDM Portal and start using the CA TDM functionality.

Installation Prerequisites

Review the following prerequisites:

- Verify that the CA TDM repository is already available, and that you have appropriate credentials to access it. For more information, see Install the Repository.

  WARNING

  Warning: Create a GTREP database for each CA TDM Portal Installation. Do not use the same repository for multiple CA TDM Portal installations.
- Verify that your system meets all requirements. See System Requirements.

Install the CA TDM Portal

Follow these steps:

1. Ensure that the TDM Repository installation is complete. For more information, see Install the Repository.
2. Double-click the CA TDM Portal installer executable file.
The CA Test Data Manager Portal welcome dialog opens.
3. Click **Next**.
4. Accept the end-user license agreement and click **Next**.
5. Specify whether to use HTTPS protocol for the CA TDM Portal:
Default: https://localhost:8443/
   – **Secure with HTTPS**
   Specifies that the CA TDM Portal can be accessed using HTTPS. This uses a self-signed certificate by default. You can replace the certificate with your own certificate. For more information, see Manage Certificates.
   Default: Enabled

     WARNING

     You should send encrypted web tokens over HTTPS connections, rather than over HTTP. The HTTPS protocol helps prevent interception of data by unauthorized users.
   – **Port**
   Specifies the port where you want to run the CA Test Data Manager Portal service. The default ports are 8080 for HTTP and 8443 for HTTPS.

   Specify whether to automatically generate the keystore password:

- **Autogenerate keystore password**
  Specifies whether you want to automatically generate the keystore password. Select this option to do so.
  If you do not want to automatically generate the keystore password, provide the required information in the **Keystore Password** and **Confirm Keystore Password** fields.
- **Keystore Password**
  Specifies the password you want to use to generate the keystore.
- **Confirm Keystore Password**
  Verifies that the keystore password is entered correctly. Keystore password and confirm keystore password values must be the same.

6. Click **Next**.
   The **Configure Installation Details** dialog opens.
7. Perform the following actions:
   a. Browse to the folder location where you want to install the CA TDM Portal.
      Note: The default install path is C:\Program Files\CA\CA Test Data Manager Portal.
   b. Select the type of the CA TDM repository database from the **Repository database type** drop-down list. The supported database types are Microsoft SQL Server and Oracle.
8. Click **Next**.
9. Enter the connection information for your CA TDM repository as follows:
   - **Server**
     Specifies the name of the server where the database is located. Specify the server name in the following format:
     - SQL Server
       <Server Name> or <Server Name\Instance Name>
     - Oracle
       <Server Name/Fully Qualified Service Name>
   - **Port**
     Specifies the port where the database is available.
     Default: 1433 for MS SQL Server and 1521 for Oracle
   - **Database**
     (Microsoft SQL Server only) Specifies the repository database name. For example, gtrep.
   - **Integrated Security**
     (Microsoft SQL Server only) Lets you use the Windows integrated security authentication to access the database. For more information, see Enable Integrated Security for CA TDM Portal.
   - **Username**
     Specifies the user allowed to access the database.
     Default: sa for SQL Server and gtrep for Oracle
   - **Password**
     Specifies the password associated with the database user.
10. Click **Next**.
    The **Orient DB** dialog opens.
11. Enter information for creating the OrientDB database as follows:
    Note: OrientDB is a distributed graph database that the CA TDM Portal uses to model projects.
    - **Username**
      Specifies the name of the user having access to the Orient database.
      Default: root
    - **Password**
      Specifies the password for the user name provided in the **Username** field. Also, confirm the password.
    - **Binary Port**
      Specifies the binary port number where you want to run the Orient database.
      Default: 2424
    - **HTTP Port**

Specifies the HTTP port number where you want to run the Orient database.
Default: 2480

12. Click **Next**.
    The **ALM Service** dialog opens.
13. Specify the following information to connect to an installed ALM service:
    - **Server**
      Specifies where the ALM service is installed.
    - **Port**
      Specifies the ALM service port number.
      Default: 8095
14. Click **Next**.
    The **Send Usage Data** dialog opens.
15. Select **Share usage data with CA Technologies** if you want to send usage information to CA Technologies.
    For more information, refer to Configure Telemetry.
16. Enter your support credentials.
    - **Username**
      Specifies the user allowed to access CA Support.
    - **Password**
      Specifies the password associated with the support user.
17. Click **Next**.
    The **Ready to Install** dialog opens.
18. Click **Install** to start the CA TDM Portal installation.
    The **Installing CA Test Data Manager Portal** dialog opens and displays the installation status.
19. Click **Finish** when the installation completes.
20. Open the Windows **Services** dialog (Start, services) and verify that services named **CA Test Data Manager Portal** and **OrientDB** are available and are running.
    You have successfully installed the CA TDM Portal.
    After the installation, the CA TDM Portal generates logs for each task that you perform. The log files are available at `%ProgramData%\CA\CA Test Data Manager Portal\logs\`. The full default path is `C:\ProgramData\CA\CA Test Data Manager Portal\logs`.
21. (Optional) Specify Find & Reserve Prefetch Databases
    The TDM Portal Find and Reserve feature lets you cache data to improve performance (see Specify where prefetched databases are stored).

    TIP

    If you want to manually specify a location for this data, we recommend that you specify this location before your first use of CA TDM Portal. If you specify a location after the first prefetch (to the default gtrep database), follow the Remove prefetched data from gtrep database instructions to remove cached databases from the gtrep schema/database.

Modify your CA TDM Portal installation

If you run the CA TDM Portal installer executable on a machine with an installation of CA TDM Portal, the installer displays the **Modify, Repair or Remove installation** dialog (after the welcome dialog). Select the appropriate option (**Modify, Repair,** or **Remove**) and proceed with the associated steps.

Uninstallation of CA TDM Portal

You can uninstall CA TDM Portal from the Control Panel or from the Start menu (Start, All Programs, CA, CA Test Data Manager Portal, Uninstall CA Test Data Manager Portal).

NOTE

For reinstallation of CA TDM Portal, we recommend you first back up your OrientDB database. Failure to do so may result in loss of data from OrientDB database - see possible outcome at CA TDM Portal Troubleshooting.

For more information, see OrientDB Backup and Restore.

Install the CA TDM Portal using Command Line Interface

Follow these steps:

1. Ensure that the CA TDM Repository installation is complete.
2. Copy the CA TDM Portal installer executable file to the server where you want to install the CA TDM Portal. For example, *C:\Users\Administrator\Downloads*
3. Open Command Line Interface and change the drive to the folder where you copied the CA TDM Portal installer. For example, to change the directory to the *C:\Users\Administrator\Downloads,* run the following command:
   cd C:\Users\Administrator\Downloads
4. Run the following command:

```
C:\Users\Administrator\Downloads>"setup_CA Test Data Manager Portal<version>.exe" /
quiet SERVER_PROP=<database server name> PORT_PROP=<Port> DATABASE_PROP=<Repository
 Database Name> USERNAME_PROP=<Database User Name> PASSWORD_PROP=<Database
 User Password> TDMWEB_SERVICE_PORT=<Port> HTTPS_ENABLED=<true/false>
 ODBC_RES_PROP="<SQL Server/ORACLE>" ORIENTDB_PASSWORD=<Password>
 ORIENTDB_BINARYPORT=<Port> ORIENTDB_HTTPPORT=<Port> ALM_PORT_PROP=<Port>
 ALM_SERVER_PROP=<ALM Server Name> AUTO_GENERATE_KEYSTORE_PASSWORD=<true/false>
 TDMWEB_KEYSTORE_USER_PASSWORD=<password> /L*V "<Log File Name.log>"
```

WARNING

- You must separate each parameter with a space. If any parameter value has spaces, then use " " (double quotes).
- Ensure that you specify the parameter values correctly. If there are any syntax errors or invalid parameter values, the installation will fail. In such case the roll back of the installation is not supported.

– **SERVER_PROP**
Specifies the name of the server where the database is located. Specify the server name in the following format:
  - SQL Server
    <Server Name> or <Server Name\Instance Name>
  - Oracle
    <Server Name/Fully Qualified Service Name>

– **PORT_PROP**
Specifies the port where the database is available.
Default: 1433 for MS SQL Server and 1521 for Oracle

– **DATABASE_PROP**
(SQL Server only) Specifies the repository database name. For example, gtrep.

– **USERNAME_PROP**
Specifies the user allowed to access the database.
Default: sa for SQL Server and gtrep for Oracle

– **PASSWORD_PROP**
Specifies the password to authenticate the database user.

– **ORIENTDB USER**
Specifies the name of the user having access to the Orient database.

Default: root

- **ORIENTDB_PASSWORD**
  Specifies the password for the Orient database user.
- **ORIENTDB_BINARYPORT**
  Specifies the binary port number where you want to run the Orient database.
  Default: 2424
- **ORIENTDB_HTTPPORT**
  Specifies the HTTP port number where you want to run the Orient database.
  Default: 2480
- **TELEMETRY_ENABLED**
  Specifies whether you want to send your CA TDM usage information to CA Support. The default value is false. This is an optional parameter.
- **TELEMETRY_EMAIL**
  Specifies the email of your CA Support account.
  Note: This parameter is mandatory if TELEMETRY_ENABLED is set to true.
- **TELEMETRY_PASSWORD**
  Specifies the password of your CA Support account.
  Note: This parameter is mandatory if TELEMETRY_ENABLED is set to true.
- **ODBC_RES_PROP**
  Specifies whether the ODBC is SQL Server or ORACLE.
- **ALM_SERVER_PROP**
  Specifies where the ALM service is installed.
- **ALM_PORT_PROP**
  Specifies the ALM service port number.
  Default: 8095
- **TDMWEB_SERVICE_PORT**
  Specifies the TDM web service port number.
  Default: 8443
- **HTTPS_ENABLED**
  Specifies whether to enable the "https" protocol or not. The value "true" indicates to enable, and "false" indicates to disable the "https" protocol.
- **AUTO_GENERATE_KEYSTORE_PASSWORD**
  Specifies whether to automatically generate the keystore password. The default value is true. This is an optional parameter.
- **TDMWEB_KEYSTORE_USER_PASSWORD**
  Specifies the password you want to use to generate the keystore. This parameter is applicable only when AUTO_GENERATE_KEYSTORE_PASSWORD is set to false. This parameter then becomes a mandatory parameter.
- **<Log File Name.log>**
  Specifies the name of the log file that you want to create. After installing CA TDM Portal, you can access the log file under the path: <*install-dir*>\logs\.

Below is the command with parameter values for example:

```
C:\Users\Administrator\Downloads>"setup_CA Test Data Manager Portal4.4.0.0.exe" /
quiet SERVER_PROP=localhost\TDM PORT_PROP=1433 PASSWORD_PROP=abcCY@123
 DATABASE_PROP=gtrep USERNAME_PROP=sa TDMWEB_SERVICE_PORT=8081 HTTPS_ENABLED=false
 ODBC_RES_PROP="SQL Server" ORIENTDB_PASSWORD=root ORIENTDB_BINARYPORT=2425
 ORIENTDB_HTTPPORT=2481 ALM_PORT_PROP=8092 ALM_SERVER_PROP=mat01-05
 AUTO_GENERATE_KEYSTORE_PASSWORD=false TDMWEB_KEYSTORE_USER_PASSWORD=Abc@123 /L*V
 "CATDMPortalInstallation.log"
```

The CA TDM Portal is successfully installed.

5. View the log file for detailed information. After successful installation, the log file shows the following message at the end of the file:
   "CA Test Data Manager Portal -- Installation completed successfully."

Install Additional Database Drivers

When you try to connect to a database without having required drivers installed, an error message will tell you that the respective JDBC driver class was not found. After you install CA TDM Portal, add JDBC drivers for additional databases that you are using.

To add JDBC Drivers to a CA TDM Portal installation:

1. Download the JDBC driver from the respective company's support portal.
   - MySQL JDBC driver - mysql-connector-java-5.1.38-bin.jar
   - MariaDB JDBC driver - mariadb-java-client-2.0.3.jar
   - Sybase JDBC driver - jconn4.jar
   - DB2 JDBC driver - db2jcc4.jar
   - Teradata JDBC drivers - tdgssconfig.jar and terajdbc4.jar
   - DB2 AS/400 JDBC driver - jt400_jdk8-*xx.y*.jar (for example, jt400-jdk8-10.4.jar)
2. Copy the jar files into the following CA TDM folder:
   `C:\Program Files\CA\CA Test Data Manager Portal\tomcat\jdbc-drivers`
3. Restart the CA Test Data Manager Portal service.

   NOTE

   If you want to connect to the CA TDM Repository or source and target data sources with Integrated Security, Enable Integrated Security for CA TDM Portal.

How to Install the CA TDM Portal and Datamaker on Separate Servers

When Datamaker and CA TDM Portal are installed on the same machine with default settings, no additional configuration is required.

If you have customized the URL parameters of CA TDM Portal during installation (for example, you changed the protocol or port to be different from the default, https://localhost:8443), or, if you have installed the Portal on a different machine than Datamaker, tell Datamaker where you have installed CA TDM Portal.

1. Browse to the Datamaker installation directory:
   `C:\Program Files (x86)\Grid-Tools\GTDatamaker`
2. Open the file `RESTutil.properties` in a text editor.
3. Configure the `server.url` property to point to your custom CA TDM Portal server and port.

Authentication and Security

Datamaker's RESTutil uses the default CA TDM login endpoint (/TestDataManager/user/login) to authenticate users. Proper security for the REST client can only be ensured when the CA TDM Portal uses the HTTPS protocol. When the Portal uses HTTP, all the requests, including login and password, are sent unencrypted in open text.

The RESTutil supports two levels of TLS certificate trust. Choose *one* of the following options:

Trust Server Certificate

Enable the SSL Trust property in your RESTutil.properties file:

```
security.ssl.trust-server-certificate=true
```

WARNING

Trusting the server certificate is generally an unsafe option when the RESTutil and Portal are installed on separate machines, as it does not prevent man-in-the-middle attacks.

Trusting the server certificate is acceptable in case of single-machine installations, which is the default installation option for Portal and Datamaker.

Verify Server Certificate

Disable SSL Trust and configure the server certificate in the RESTutil.properties file.

If you use the default self-signed certificate in CA TDM Portal:

1. Open the Portal login page in a browser.
2. Save the SSL certificate as a file to the following path:
   `C:\Program Files (x86)\Grid-Tools\GTDatamaker\server.cer`
3. Configure your certificate file `server.cer` in the RESTutil.properties file:

```
security.ssl.trust-server-certificate=false

      security.ssl.certificate-file=server.cer

      security.ssl.trust-store=.truststore

      security.ssl.trust-store.password=
```

Alternatively, you can configure it to use your Java Trust Store.

Install TDM Portal for Docker

From Test Data Manager 4.7, TDM Portal functionality is available as a collection of Docker images. This functionality requires you to use the Docker software, available for Linux. For more information on the use of Docker, see https://docs.docker.com/.

NOTE

Docker is also available for Windows. For this implementation, we recommend the use of the **Docker Toolbox**. For more information, see https://docs.docker.com/toolbox/toolbox_install_windows/.

Before you install TDM Portal (or any TDM components), it is necessary to install the **gtrep** repository (i.e. to create the database and **gtrep** user). For more information, see Install the Repository.

From Test Data Manager 4.8, the first time you run TDM Portal in Docker, it creates the necessary tables in your **gtrep** repository (or upgrades these tables to the same version as TDM Portal, if they already exist).

WARNING

After you upgrade **gtrep**, you must also upgrade other product components for them to work with the repository.

Some functionality is not yet available in TDM Portal in Docker. For a full list of features that are not available, see Features not available in TDM Portal in Docker.

Required Elements

To use TDM Portal in Docker, you need a minimum of 3 active elements:

- **TDM Portal Docker container** (**docker-compose.yml** starts this container)
- **OrientDB Docker container** (**docker-compose.yml** starts this container)
- **Oracle database**TDM Portal uses the **gtrep** repository for its internal operation. TDM Portal in Docker only supports Oracle databases for this repository.

  NOTE

  If you cannot make an Oracle database available to your Docker network, you can Create an Oracle database container. This requires you to build the image with Oracle's Offical Docker container, and is an Advanced feature.

Optional containers

In addition, the following containers are available:

- **TDM Portal Tools container**
  Use this container to:
  - Encrypt a password.
  - Generate a JWT shared secret.
  - Create a **gtrep** user on your Oracle gtrep database.
  - Create sample databases on your Oracle database, from the sample databases supplied with CA TDM.
- **TDM Portal REST ActionService container**
  Use instances of this container to allow you to execute Actions from TDM Portal in Docker (one Action per container).
- **TDM Portal Masking containers**
  You can use these containers to mask data.

  TIP

  You can also use these containers to mask data independently of TDM Portal, including masking jobs sent from a Windows installation of Test Data Manager.

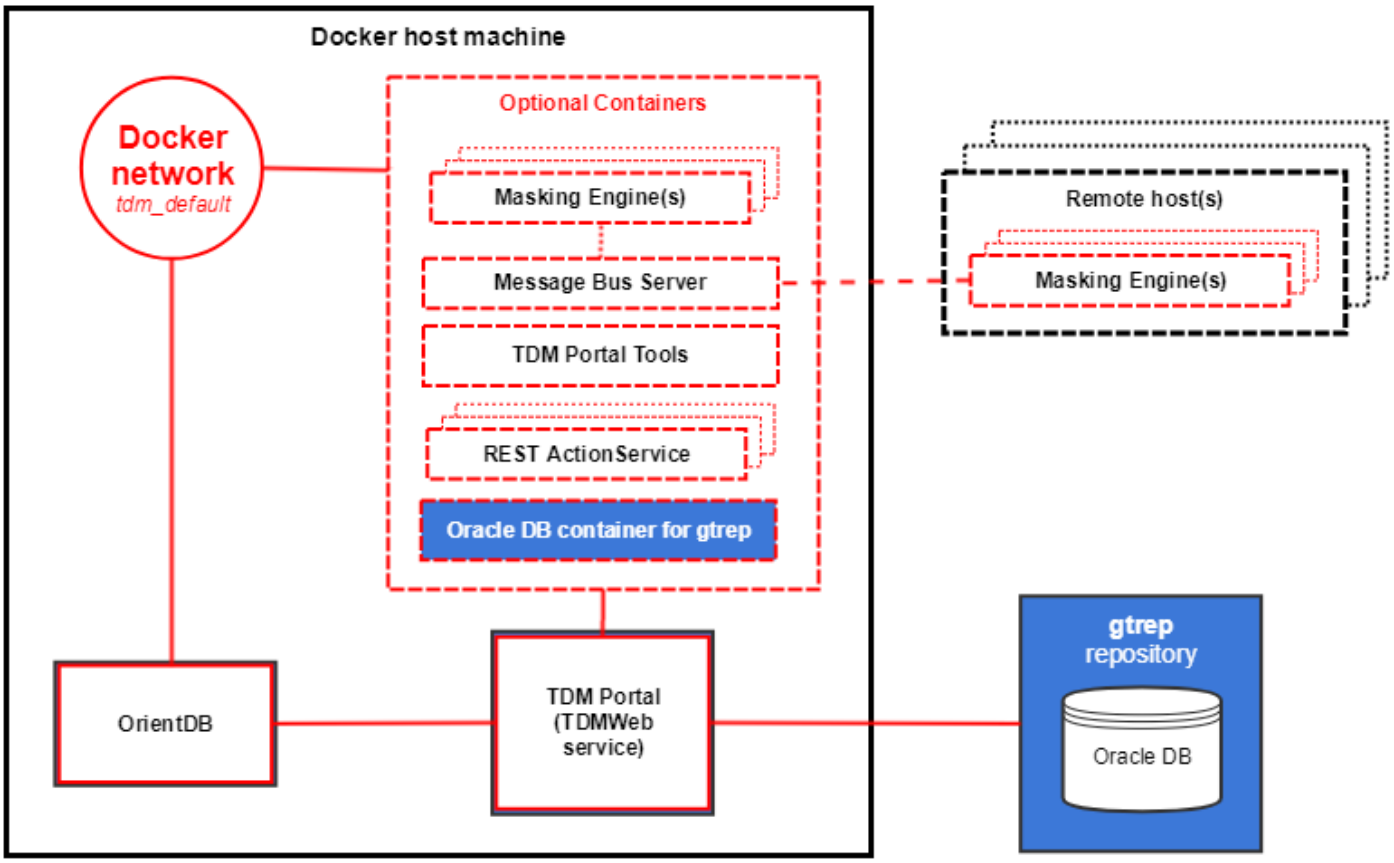
  - **Message Bus Server container**
    The TDM Portal service (in Windows or Docker) sends masking jobs (split into tasks) to this container, which then distributes these tasks to masking engine containers.
    Use the file **docker-compose-messaging.yml** to start this container.
  - **Masking engine container**
    This container performs masking tasks with the Fast Data Masker engine.
    Use the file **docker-compose-masking.yml** to start instances of this container.

    TIP

    Use the `--scale` flag to create multiple instances of this container. For more information, see Scalable masking with Docker.

This diagram shows how these containers interact with each other:

Figure 10: multiple actions



How to use Docker containers

Download Docker images

TDM Portal Docker images, and other files with which you can build your own Docker images, are available from Support. Docker images are built on Ubuntu 16.04 and Java 8u202.

For more information, see File Packages available.

Start a Docker network with docker-compose files

To start the containers necessary to use TDM Portal in Docker, we recommend that you use the `docker-compose up` command on a docker-compose.yml file, or multiple docker-compose.yml files. This .yml file starts each container from a Docker image, as a service on the same Docker network.

For more information on the docker-compose files available and how to use them, see Docker-compose files.

NOTE

You can start all containers from a Docker image individually with the **docker run** command, however we recommend the use of docker-compose files. Use of the docker run command is for Advanced users.

For more information, see Start Containers with the 'docker run' Command.

(Optional) Build your own Docker images

We recommend that you create Docker containers from the pre-built Docker images available. However, you can build your own Docker images from TDM source files and Dockerfiles. This method is more complex than the use of pre-built images, and is for Advanced users.

For more information, see Build your own Docker images.

Add Docker containers to your Docker network with the 'docker run' command

It is only necessary to start the **TDM Portal REST ActionService container** while you need them, and the containers should stopped when no longer in use. They are not included in any docker-compose.yml files.

> **TIP**
>
> Ensure that the `--network` flag for any containers you run with docker run, matches the `network` value for your Docker network.

If you choose to use the TDM Portal Oracle database container, it is necessary to build and run this separately. This container is not included in any docker-compose.yml files.

Configuration of Docker containers

Docker creates containers from the images that you supply as entries under the `services` section of your docker-compose.yml file (see Docker-compose files for the sample files available). To configure these containers, define environment variables under the `environment` section for each container within the docker-compose.yml file, or with the -e flag in a `docker run` command.

> **WARNING**
>
> You cannot change these environment variables while a container is active. If you make changes to variables in a docker-compose.yml file, the changes do not take effect until the next time you execute the file with the `docker-compose up` command.

Example:

This section of a docker-compose.yml file starts a TDM Portal container (as the service `tdmweb`) on the Docker network. It defines the environment variables **GTREP_HOST**, **GTREP_SERVICE_NAME**, **GTREP_PASSWORD** and **ORIENTDB_PASSWORD**:

```
services:
  ...
  tdmweb:
    image: tdm.packages.ca.com/tdm/tdmweb:<version>
    hostname: tdmweb

  environment:
      -
'GTREP_HOST=database'
      -
'GTREP_SERVICE_NAME=orcl'
      -
'GTREP_PASSWORD=Gridt00ls'
      - 'ORIENTDB_PASSWORD={cry}tHpzgrvNhtVu6uHGNd9EdlAuwMR30OL0sAXhBWdgM3Md'
  ...
```

State Persistence in Volumes

All Docker containers can be stateless - their state can persist in the external volumes you specify under a container's `volumes` section. If you do not specify any external volumes for a container, the container stores this data internally and it is not accessible from outside of the container.

> **WARNING**
>
> If you do not specify external volumes for a container, and you then use a new version of the container (i.e after a patch or upgrade), the data from your first container is not available in this new container.

The use of external volumes to persist the state of a container has the following advantages:

- You do not lose your data when you upgrade your version of Test Data Manager.
- You can expose JDBC drivers to your Docker network. For information on JDBC drivers you might need, see Notes on Implementation with Specific Data Sources.
- You can expose your log files, so that they are accessible from outside of your Docker network.

TDM Portal container

This page explains how to start the TDM Portal Docker container and customize its environment.

Environment Variables

You can define the following variables in the `environment` section of the **tdmweb** service, in the docker-compose file that you use to start the container:

> **NOTE**
>
> You can pass encrypted passwords to the TDM Portal container. Encrypted passwords must begin with `{cry}` .
>
> For example: `'ORIENTDB_PASSWORD={cry}tHpzgrvNhtVu6uHGNd9EdlAuwMR30OL0sAXhBWdgM3Md'`

Basic

- **GTREP_DB_TYPE**
  Gtrep database type. Default: '`oracle` '.

  > **WARNING**
  >
  > The only allowed value is '`oracle` '. You do not need to set this parameter.
- **GTREP_SERVICE_NAME**
  SID for oracle database. Default: '`orcl` '.

  > **NOTE**
  >
  > This only applies to **GTREP_DB_TYPE**='`oracle` '.
- **GTREP_PORT**
  Gtrep database port. Default '`1521` ' for **GTREP_DB_TYPE**='`oracle` '.
- **GTREP_HOST**
  Gtrep hostname. Default: '`oracle` '.
- **GTREP_DATABASE**
  Gtrep database name. Default: '`gtrep` '.

  > **NOTE**
  >
  > The TDM Portal Docker container installs the **gtrep** repository on the Oracle database that you specify.
- **GTREP_USER**
  Gtrep database user name. Default: '`gtrep` '.

TIP

We recommend that you do not create a **gtrep** user manually on the Oracle database. Instead, use the user creation tool within the TDM Portal Tools container.

- **GTREP_PASSWORD**
  Password for gtrep database. Default: '$PASSWORD'.
- **PORTAL_HOST**
  TDMWeb hostname. Default: hostname of the running system.
- **PORTAL_PORT**
  TDMWeb http port number. Default: '8080'.
- **PORTAL_ALM_PORT**
  ALM http port number. Default: '8095'.
- **ORIENTDB_HOST**
  OrientDB hostname. Default: 'orientdb'.
- **ORIENTDB_PASSWORD**
  OrientDB password. Default: '$PASSWORD'.
- **KEYSTORE_PASSWORD**
  Password for certificate keystore. Default: '$PASSWORD'.
- **INTEGRATION_USER**
  Name of integration user. Default: 'integrator'.
- **INTEGRATION_PASSWORD**
  Integration user password. Default: '$PASSWORD'.
- **ENABLE_SSL**
  SSL is enabled by default. Default: 'true'. To disable SSL use 'false'.

NOTE

All passed passwords are automatically encrypted in the relevant .properties files.

Quick setup

The following is an example of the section of docker-compose.yml file, that starts the TDM Portal container.

```
services:

tdmweb:
    image:
tdm.packages.ca.com/tdm/
tdmweb:<version>
    hostname:
tdmweb


    environment:


      - 'GTREP_HOST=
oracle'
```

```
      - 'GTREP_DATABASE=
gtrep'

      -
 'GTREP_SERVICE_NAME=orcl'

      - 'GTREP_PASSWORD=
Gridt00ls'
      - 'ORIENTDB_PASSWORD=
{cry}tHpzgrvNhtVu6uHGNd9EdlAuwMR30OL0sAXhBWdgM3Md
'

    volumes:


      - 'tdmweb_logs:/mnt/
logs'



      - 'tdmweb_storage:/mnt/storage'


      - 'tdmweb_fdmconfig:/mnt/
fdmconfig'
    depends_on:
      -
orientdb
    ports:

      -
 '8080:8080'

      - '8443:8443'
```

Advanced setup

You can pass the following variables to the docker-compose.yml file, to customize properties within configuration files. Delimit multiple properties with the | (pipe) character.

- **APPLICATION_PROP**
  Customize properties in file: `/opt/tdm/conf/application.properties`
- **DATARESERVATION_PROP**
  Customize properties in file `/opt/tdm/conf/tdmdatareservation.properties` .

Custom application.properties configuration

The following is an example of the section of a docker-compose.yml file that creates a TDM Portal container with a customized `application.properties` file:

```
services:
  tdmweb:
    image: tdm.packages.ca.com/tdm/
tdmweb:<version>
    hostname: tdmweb


    environment:


      - 'GTREP_HOST=oracle'

      - 'GTREP_DATABASE=gtrep'
      - 'GTREP_SERVICE_NAME=orcl'
      - 'GTREP_PASSWORD=Gridt00ls'
      - 'ORIENTDB_PASSWORD={cry}tHpzgrvNhtVu6uHGNd9EdlAuwMR30OL0sAXhBWdgM3Md'

      - 'APPLICATION_PROP="
spring.datasource.url=jdbc:oracle:thin:@oracle:1521:orcl|\



            spring.datasource.username=gtrep|\



            spring.datasource.password=Gridt00ls|\



            spring.datasource.driver-class-name=oracle.jdbc.OracleDriver|\



            spring.jpa.database=ORACLE|\



            file.resource.loader.path=/opt/tdm/Mail Templates/|\
```

```
            almservice.endpoint.url=http://localhost:8095/ALMService|\


            spring.datasource.tomcat.validationQuery=select 1 from dual|\


            tdmweb.security.integration.userName=integrator|\


            tdmweb.security.integration.password={cry}KSiF
+XH5JYSieHLkUkA5LdMvEk81XBHt-3OZVSKmM8LZ
"'
    volumes:
      - 'tdmweb_storage:/mnt/storage'
      - 'tdmweb_logs:/mnt/logs'
      - 'tdmweb_fdmconfig:/mnt/fdmconfig'
    depends_on:
  - orientdb
    ports:
      - '8080:8080'
      - '8443:8443'
```

Data Volumes

CA TDM Portal in Docker requires data volumes, in which to store logs and production data. These volumes include:

- **tdmweb_logs**

Contains the TDMWeb application Apache Tomcat .log files under the /tdm folder.

- **tdmweb_storage**
  Contains files related to TDMWeb user created projects and publish actions from the generators. This volume is further divided as follows:
  - **/Jobs**
  - **/objects**
  - **/ssl-cert**
  - **/jdbc-drivers**
- **tdmweb_fdmconfig**
  Volume to contain updated fdm-config.xml file

You can pass Docker your own volumes under the `volumes` section of the docker-compose.yml file.

Example:

The following is an example of the section of the **docker-compose.yml** file, which creates a TDM Portal container with external volumes for **storage, logs** and **fdmconfig:**

```
services:

  ...
  tdmweb:
    image: tdm.packages.ca.com/tdm/
tdmweb:<version>
    hostname: tdmweb


    environment:


      - 'GTREP_HOST=oracle'


      - 'GTREP_PORT=1521'

      - 'GTREP_DATABASE=gtrep'

      - 'GTREP_DB_TYPE=oracle'

      - 'GTREP_SERVICE_NAME=orcl'

      - 'ORIENTDB_PASSWORD=marmite'

    ...

volumes:



 - 'tdmweb_
storage:/mnt/storage'
```

```
  - 'tdmweb_
logs:/mnt/logs'

      - 'tdmweb_fdmconfig:/mnt/fdmconfig'
```

If you do not supply the TDM Portal container with volumes, it creates these volumes within the TDM Portal container. By default, these volumes are stored physically in the `/mnt` partition, at `/mnt/logs` and `/mnt/storage` (as in the example above).

Apache Tomcat logs

TDM Portal in Docker uses Apache Tomcat to run the Java code without the Windows environment.

Tomcat's logs (catalina and access-log) are printed to the Docker console. To check them at any time, run:

```
docker logs <tdmweb_container_name>
```

By default, Tomcat stores logs at `/mnt/logs` .

Certificates in CA TDM Portal in Docker

TDM Portal in Docker handles security with certificates.

Disable SSL

By default, the TDM Portal Docker container has SSL enabled. To disable SSL, pass command line argument `ENABLE_SSL=false` to your docker-compose.yml file when you create the Docker container.

Use a third-party certificate

If you wish to use your own third-party certificates, you must copy the relevant certificate and key file to the **storage** volume that you define under the `volumes` section of your docker-compose.yml file. TDM Portal imports these files to the Java key store the first time it runs.

Rename the files within the **storage** volume as follows, to allow TDM Portal to detect them:

Certificate file path:

`/ssl-cert/tdm-site.pem`

Key file path:

`/ssl-cert/tdm-site.key`

TIP

The private key you supply cannot be password protected. You can use the following command in your Linux environment to remove password protection:

```
openssl rsa -in encrypted-tdm-site.key -out tdm-site.key
```

Create a self-signed certificate

If there is no certificate and key file present in the storage volume, the TDM Portal Docker container creates a self-signed certificate when you run it.

Troubleshooting

Exit Code 137

Symptom:

If your instance of the TDMWeb container closes with Exit Code 137, this indicates that your machine does not have sufficient memory.

Solution:

Assign more memory to your machine (either physical or virtual).

TDM Portal OrientDB container

The TDM Portal Docker container requires an active OrientDB container. The TDM Portal Docker container requires a TDM Portal OrientDB Docker container to be active, in order to operate.

> **TIP**
>
> The docker-compose.yml file that starts the TDM Portal Docker container, also starts the TDM Portal OrientDB container. We recommend the use of docker-compose to start your TDM Portal Docker network.
>
> For more information, see Docker-compose files.

The OrientDB Docker image is available as part of the file packages available from support.ca.com.

Environment variables

- **ORIENTDB_ROOT_PASSWORD**
  The password for this OrientDB container.

  > **NOTE**
  >
  > The environment variable **ORIENTDB_PASSWORD** in the TDM Portal service, must match this value.

Example:

The following is an example of the section of docker-compose.yml file, that starts the OrientDB container with the password `marmite` :

```
services:


  orientdb:

     image: tdm/
orientdb:2.2.33


    hostname: orientdb
```

```
    environment:

      - 'ORIENTDB_ROOT_PASSWORD=marmite'
```

Data Volumes

The OrientDB Docker container requires the following 3 volumes:

- **orientdb_backup**
- **orientdb_databases**
- **orientdb_config**

Specify your own volumes under the `volumes` section of your docker-compose.yml file.

Example:

The following is an example of the section of docker-compose.yml file, that starts the OrientDB container and defines the volumes **orientdb_backup**, **orientdb_databases** and **orientdb_config**:

```
services:

  orientdb:
    image: tdm/orientdb:2.2.33



hostname: orientdb
    environment:
    - 'ORIENTDB_ROOT_PASSWORD=marmite'



volumes:

    - 'orientdb_backup:/orientdb/backup'


    - 'orientdb_databases:/orientdb/databases'

    - 'orientdb_config:/orientdb/config'
```

If you do not specify your own volume, Docker creates volumes in the container at the locations above.

TDM Portal Tools container

To use TDM Portal in Docker, it is necessary to create a user for the gtrep repository, which the TDM Portal container creates on your Oracle database. The TDM Portal Tools container includes utilities to make this process and others easier.

NOTE

This container requires the use of the `docker run` command. The Docker network on which you wish to make changes must be active when you run this container.

Using the Tools container

You can run the Tools container in the following two modes:

- **Interactive mode**
  To run the tool in Interactive mode, supply no environment variables to the `docker run` command. When you run this container in **Interactive** mode, the **Action selection** menu displays, from which you can run the **encryption utility** (in 'encrypt password' and 'generate JWT shared secret' modes), the **user creation tool** and the **sample database creation tool**. The tool prompts you to input values for each parameter.
  Syntax:

```
docker run --rm -it --name=TDMWeb_Tools --hostname=TDMWeb_Tools --
network=tdm_default tdm/tdmtools:<version> [--encryption-util -p]
```

  TIP

  To go directly to the Tool you need, add ***--<tool-name>*** to the end of the command.
- **Batch mode**
  To run the tool in Batch mode, pass all the values you want to define, to the container as command line parameters.

  NOTE

  To run each tool in Batch mode, you must supply all necessary environment variables to the `docker run` command. You can omit optional variables, in which case its value is that variable's default value.

Features

The TDM Portal Tools container includes the following features:

- **EncryptUtil**
  This utility functions the same as the Encryption Utility in a normal Windows installation. For more information, see Use the Encryption Utility to Encrypt Passwords.

  NOTE

  The Password encryption utility is not available in **Batch** mode.
- **User creation tool**
  If you have Administrator access to the Oracle database where your gtrep repository is installed, you can use this tool to create gtrep users on this database.
- **Sample database creation tool**
  With this tool, you can create tables from the sample databases supplied with TDM.

Password encryption utility

The Password encryption utility is only available in **Interactive** mode.

Run the encryption utility in 'encrypt password' mode

```
docker run --rm -it --name=
TDMWeb_Tools --hostname=

TDMWeb_Tools
 --network=tdm_default tdm/tdmtools:<version> --encryption-util -p
```

This tool prompts you to enter a password, and returns the encrypted value.

Run the encryption utility in 'generate JWT shared secret' mode

```
docker run --rm -it --name=
TDMWeb_Tools --hostname=
TDMWeb_Tools
 --network=tdm_default tdm/tdmtools:<version> --encryption-util -s
```

When you run this tool, it generates a JWT shared secret. For more information, see TDM Portal Password Management.

Gtrep user and schema creation tool

Create a user and schema for your **gtrep** repository. Before you run this tool, the gtrep repository is an empty database.

Run the user and schema creation tool in Interactive mode

Syntax:

```
docker run --rm -it --name=TDMWeb_Tools --hostname=TDMWeb --network=tdm_default tdm/
tdmtools:<version> --create-db-user
```

This command prompts you to enter each parameter necessary to create a database user.

Run the user and schema creation tool in Batch mode

Syntax:

```
docker run --rm --name=TDMWeb_Tools --hostname=TDMWeb --network=tdm_default \

-e SYS_DB_USER="system" \

-e SYS_DB_PASSWORD="system_password" \

-e DB_HOST="ora11g" \
```

```
-e DB_SERVICE_NAME="orcl" \

-e DB_USER="gtrep" \

-e DB_PASSWORD="gtrep_password" \


tdm/
tdmtools:
<version> --create-db-user
```

Where:

- **SYS_DB_USER**
  Existing user with administrator access to your Oracle database
- **SYS_DB_PASSWORD**
  Password for this existing user.
- **DB_HOST**
  Hostname of database server.
- **DB_SERVICE_NAME**
  Oracle service name.
- **DB_USER**
  The username you want to create for the new gtrep user.
- **DB_PASSWORD**
  The password you want to create for the new gtrep user.

  NOTE

  This password must be **unencrypted**.

Sample database creation tool

Create a database on your Oracle database, from the sample databases included with CA TDM.

Run the Sample database creation tool in Interactive mode

Syntax:

```
docker run --rm -it --name=
TDMWeb_Tools --hostname=TDMWeb --network=tdm_default tdm/tdmtools:<version> --create-
sample-db
```

This command prompts you to enter each parameter necessary to create a database from the sample databases provided with CA TDM.

Run the Sample database creation tool in Batch mode

Syntax:

```
docker run --rm --name=TDMWeb_Tools --hostname=TDMWeb --network=tdm_default \
```

```
-e SYS_DB_USER="system" \

-e SYS_DB_PASSWORD="system_password" \

-e DB_HOST="ora11g" \

-e DB_SERVICE_NAME="orcl" \

-e DB_NAME="orders" \

-e DB_PASSWORD="gtrep_password" \

-e DB_PORT="1521"

tdm/tdmtools:<version> --create-sample-db
```

Where:

- **SYS_DB_USER**
  Existing user with administrator access to your Oracle database
- **SYS_DB_PASSWORD**
  Password for this existing user.
- **DB_HOST**
  Hostname of database server.
- **DB_SERVICE_NAME**
  Oracle service name.
- **DB_NAME**
  Name of the sample database to create. This is also the name of a new user for this database.
  The available sample databases are: `creditcard, creditcard_e, orders, orders_e, scramble, travel, travel_e.`
- **DB_PASSWORD**
  Password to associate with the **DB_NAME** user.
- (Optional) **DB_PORT**
  Oracle database port. Default value is `1521`.

TDM Portal REST ActionService container

You can execute **REST** and **SQL** Actions (of type **Publish** and **Table**) with the TDM Portal in Docker.

For more information about Actions in Test Data Manager, see Create and Manage Publish and Table Actions.

> **WARNING**
>
> Actions of type **HOST** and **WORKFLOW** are not available within the Docker container environment.

To create and execute REST actions in TDM Portal in Linux, it is necessary to run a TDMWeb ActionService container, for each Action you wish to execute. This container must run concurrently with the TDMWeb Portal container.

Syntax for the ActionService container:

```
docker run --rm -it --network=tdm_default --name=echo-action --hostname=echo-action
 -e PUBLISH_ACTION="/bin/sh script.sh" -e ACTION_SECRET=secret -p hostport:8443 tdm/
action-service:<version>
```

Where:

- **--name** = A name that you give this container for your reference.
- **--hostname** = The name of this service by which other services can connect with it.
- **--network** = The name of the Docker network on which your containers run.
- **-e** = Environment variables.
  - **PUBLISH_ACTION** = the command line action you want to execute. This can be in the form of a script, that should be stored and executable on the **action-service** Docker container.
  - **ACTION_SECRET** = this value must match the Secret value that you enter when you create the Action in the UI. For more information, see Create and Manage Publish and Table Actions.
- **host** = the port number on which the REST API listens for input. This must match the port to which you map the REST URL when you create a REST Action.

  NOTE

  This parameter is only necessary if you need to access the Action on the Docker host.

REST Actions

For each REST Action you wish to execute, it is necessary to write a command or script, that you pass to the ActionService container with the docker run command. This command/script can contain variables that you define in the UI on the Generators > Action page.

NOTE

It is also necessary to add any standard variables to this field, if you want to use them in your script.

The first number after the **-p** parameter defines the port on which the container listens for the REST action. This must correspond to the address you define in the Action page of the UI. The number that follows, is the application's default port.

The `ACTION_SECRET` environment variable that you pass to the ActionService container, must match the value of the **Action Secret** field for the REST Action you create on the Actions page of the TDM UI.

Example of an Action that uses a script:

```
docker run -e PUBLISH_ACTION='/bin/sh -c "/tmp/testscript.sh $param1 $param2"' -e
 ACTION_SECRET='secret' -p 8444:8443 tdm/action-service:<version>
```

We provide a sample script, which includes one sample action.

Action response Logs

- TDM logs Actions that you execute *ad-hoc* (i.e. with the Execute function in the Portal UI) in the `TDMGeneratorService.log` file.
- TDM logs Actions that you execute as part of the Publish process (i.e. Actions that run before or after the publish) in the `TDMPublish.log` file.

In a default installation, these log files are located in `C:\ProgramData\CA\CA Test Data Manager Portal\logs`.

NOTE

If the action succeeds, TDM logs the response with 'debug' level. To see these logs, add the following line to `C:\Program Files\CA\CA Test Data Manager Portal\conf\logback-tdm.xml` (in a default installation):

```
<logger name="com.ca.tdm.servicecommon.publishaction" level="DEBUG" />
```

For error responses, this step is not necessary.

Sample TDM REST Action containers

The following REST Action sample images are available, from which you can customize your own Action containers. Each one is supplied in the form of file folders that include `build` and `run` scripts - these scripts include the `docker build / docker run` commands and the necessary environment variables (included in Dockerfiles).

These folders are available from https://github.com/CATechnologies/tdm-custom-actions.

- Echo
- Download and Copy
- DB Log

Echo

This Action's folder can be found at https://github.com/CATechnologies/tdm-custom-actions/ echo.

This action echoes parameters passed by TDM.

Environment Variables

This image inherits environment variables from the base TDM image.

When you register this Action in TDM, you can pass custom parameters param1 and param2 to the Action, and the Action also echoes these parameters.

`PUBLISH_ACTION`
is set in the Echo Dockerfile to the `echo.sh` script.

`ACTION_SECRET`
must match the Secret that you set when you create this Action in TDM Portal.

Build image

Run script **Docker.build.sh**, in the **tdm-custom-actions/tree/master/echo/** folder.

If you want to change the version of TDM to use for the base image, amend the `tdmVersion` parameter in the **tdm.version.sh** file.

Run image

Run script **Docker.run.sh**, in the **tdm-custom-actions/tree/master/echo/** folder.

Download and Copy

This Action's folder can be found at https://github.com/CATechnologies/tdm-custom-actions/tree/master/download-and-copy.

This action performs the following tasks:

1. Downloads a .zip file generated by a Generator (see Generate Data Using the CA TDM Portal), with a REST call.
2. Unzips this .zip file.
3. Securely copies the unzipped data to a target host/folder with the SCP command.

> **NOTE**
>
> The SCP command requires SSH configuration. Follow the procedure to Add your own SSH configuration, **after** you start this Action container.

TDM Variables

Pass the following variables to the Action when you create it in TDM:

- **targetHost**
  Hostname of your target.
- **targetFolder**
  Folder on the target host into which to copy data.
- **targetUser**
  Target host user whose identity will be used to connect to the target host.

Environment Variables

This image inherits parameters from the base TDM image.

`PUBLISH_ACTION`
is set in the Download-and-Copy Dockerfile to the `download.sh` script.

`ACTION_SECRET`
must match the Secret that you set when you create this Action in TDM Portal.

Build image

Run script **Docker.build.sh**, in the **tdm-custom-actions/tree/master/download/** folder.

If you want to change the version of TDM to use for the base image, amend the `tdmVersion` parameter in the **tdm.version.sh** file.

Run image

Run script **Docker.run.sh**, in the **tdm-custom-actions/tree/master/download/** folder.

Add your own SSH configuration

To use the SCP command, it is necessary to add your own SSH configuration to the container, while it is running.

Follow these steps:

1. Attach the Action to the running container.

```
docker exec -it download-action /bin/bash
```

2. Run `ssh-keygen` (confirm default, including empty passphrase).
3. Copy the files that `ssh-keygen` generates(**id_rsa**, **id_rsa.pub**) to **conf** folder.
4. Add content of your public key (**id_rsa.pub**) to the target host, to `.ssh/authorized_keys`
5. Create the file **know_hosts** in the **conf** folder and add public key of your target host (**id_rsa.pub**) to this file.

Uncomment SSH configuration commands in the Dockerfile.

DB Log

This Action's folder can be found at https://github.com/CATechnologies/tdm-custom-actions/tree/master/db-log.

This action logs invocation of itself in a database table. When you register this action as a pre/post build action, it stores information about its invocation in the database.

> **NOTE**
>
> The action assumes that the log table already exists in the database. Use **log-table.ddl** to create the relevant table before you run the Action for the first time.

This action is based on **Oracle Instance Client** container. After accepting the license, you can get the container free.

Node.js is part of the container to run the action script. The action script is implemented in TypeScript and uses Node.js Oracle DB client (**node-oracledb**). The documentation for node-oracledb is available at https://oracle.github.io/node-oracledb/doc/api.html.

Environment Variables

This image inherits parameters from the base TDM image.

`PUBLISH_ACTION`
is set in the DB-Log Dockerfile to the `db-action.js` script, which the TypeScript compiler creates from `db-action.ts`.

`ACTION_SECRET`
must match the Secret that you set when you create this Action in TDM Portal.

The following variables define the connection to the database, where this Action logs its invocation:

- **dbHost**
  Hostname of the database host, e.g. "database".
- **dbService**
  Name of the Oracle service, e.g. "orcl".
- **dbUser**
  Name of the Oracle user, e.g. "TRAVEL".
- **dbPassword**
  Password for the user defined by **dbUser**.
- **logTableName**
  Table into which Action logs events. Default: `TDM_ACTIONS_LOG_TABLE`.

Build image

Run script **Docker.build.sh**, in the **tdm-custom-actions/tree/master/db-log/** folder.

Run image

Run script **Docker.run.sh**, in the **tdm-custom-actions/ tree/master/db-log/** folder.

TDM Portal Masking containers

From Test Data Manager 4.8, Fast Data Masker's masking engine is available in the TDM Portal Docker container.

There are also the following two new Docker containers, with which you can scale masking jobs to run across multiple instances of the FDM masking engine concurrently:

- TDM Portal Message Bus Server container
- TDM Portal Masking Engine container

For more information about how to use these containers, see Scalable masking with Docker.

TDM Portal Message Bus Server container

The Message Bus Server container (or Messaging container) receives lists of masking tasks from the TDM Portal Masking Engine container.

For more information on this process, see Scalable masking with Docker.

How to use the Messaging container

We recommend that you start the TDM Portal Masking Engine container with the supplied file **docker-compose-messaging.yml** (expand below). For more information, see Docker-compose files.

Alternatively, you can start the container with the docker run command (this is for Advanced users).

docker-compose-messaging.yml

```
version: '3.5'


networks:
  default:
    name: tdm_default
services:
  messaging:
    image: tdm/
messaging:<version>
    hostname: messaging
    ports:
      # Expose rabbitmq port in order to make it possible to scale with remote masking
 engines
      - '5671:5671'
      # Expose port 15671 to allow HTTPS access to the rabbitmq management console.
      #- '15671:15671'
    environment:
      - RABBITMQ_LOG_BASE=/var/log/rabbitmq/log
      - RABBITMQ_LOGS=/var/log/rabbitmq/log/rabbitmq.log
      - RABBITMQ_SASL_LOGS=/var/log/rabbitmq/log/rabbitmq_sasl.log
      # Specify your messaging credentials here
      # These values should match the credentials which have been specified
```

```
      # in the tdmweb and masking containers above. These credentials will
      # be used to create a user on the message broker.
      - 'DEFAULT_USER=Admin'
      - 'DEFAULT_PASS={cry}1hY5pZrm87PWjgPdmypDbVZnL4a108lxy8YLuUVRMCr8'
      # SSL selfsigned certificate only
      - RABBITMQ_SSL_CACERTFILE=/home/testca/cacert.pem
      - RABBITMQ_SSL_CERTFILE=/home/server/cert.pem
      - RABBITMQ_SSL_FAIL_IF_NO_PEER_CERT=false
      - RABBITMQ_SSL_KEYFILE=/home/server/key.pem
      - RABBITMQ_SSL_VERIFY=verify_none
    volumes:
      - 'messaging_rabbitmqdb:/var/lib/rabbitmq'
volumes:
  messaging_rabbitmqdb:
```

Environment Variables

You can define the following variables in the `environment` section of the **messaging** service, in **docker-compose-messaging.yml.**

- **RABBITMQ_LOG_BASE**
  Defines the location at which to store RabbitMQ's RABBITMQ_LOG_BASE file. Default: */var/log/rabbitmq/log*
- **RABBITMQ_LOGS**
  Defines the location at which to store RabbitMQ's RABBITMQ_LOGS file. Default: */var/log/rabbitmq/log/rabbitmq.log*
- **RABBITMQ_SASL_LOGS**
  Defines the location at which to store RabbitMQ's RABBITMQ_SASL_LOGS file. Default: */var/log/rabbitmq/log/rabbitmq_sasl.log*
- **DEFAULT_USER**
  Specifies the username from which to accept masking tasks. Must match the value of **messaging.username** (Portal in Windows) or **MESSAGING_USER** (Portal in Docker), and **MESSAGING_USER** in the Masking Engine container.
- **DEFAULT_PASS**
  Specifies the password for the user from which the service accepts masking tasks. Must match the value of **messaging.password** (Portal in Windows) or **MESSAGING_PASS** (Portal in Docker), and **MESSAGING_PASS** in the Masking Engine container.
- **RABBITMQ_SSL_CACERTFILE**
  Location of Certificate Authority certificate. Default: */home/testca/cacert.pem*
- **RABBITMQ_SSL_CERTFILE**
  Location of certificate. Default: */home/testca/cert.pem*
- **RABBITMQ_SSL_FAIL_IF_NO_PEER_CERT**Default: *false*
- **RABBITMQ_SSL_KEYFILE**
  Location of key file. Default: */home/testca/key.pem*
- **RABBITMQ_SSL_VERIFY**
  Default: *false*

Data Volumes

You can use the following volume to persist data from the Masking Engine container:

- **messaging_rabbitmqdb**
  Stored by default at `/var/lib/rabbitmq`

Create your own Signed Certificate

The Messaging container ships with an out of the box certificate, for SSL encryption in RabbitMQ. If you want to provide your own certificate, see Create a Certificate for the Messaging container for more information.

Create a Certificate for the Messaging container

The messaging container ships with an out of the box certificate, for SSL encryption in RabbitMQ. If you want to provide your own certificate, follow these instructions to create your own SSL certificate.

> **NOTE**
>
> Certificates must be signed by a Certificate Authority (CA) for RabbitMQ. Out of the box, the messaging container includes a certificate which can be used as a CA. You can use the out of the box Certificate Authority, or replace it with your own.

Prerequisites

- This process assume you are using **openssl** to perform PKI operations.
- Some of these commands use the file **openssl.cnf** for configuration. The contents of this file as are as follows:

  openssl.cnf

```
[ ca ]

default_ca = testca




[ testca ]

dir = .

certificate = $dir/cacert.pem

database = $dir/index.txt

new_certs_dir = $dir/certs

private_key = $dir/private/cakey.pem

serial = $dir/serial




default_crl_days = 7
```

```
default_days = 365

default_md = sha256




policy = testca_policy

x509_extensions = certificate_extensions




[ testca_policy ]

commonName = supplied

stateOrProvinceName = optional

countryName = optional

emailAddress = optional

organizationName = optional

organizationalUnitName = optional

domainComponent = optional




[ certificate_extensions ]

basicConstraints = CA:false




[ req ]

default_bits = 2048
```

```
default_keyfile = ./private/cakey.pem

default_md = sha256

prompt = yes

distinguished_name = root_ca_distinguished_name

x509_extensions = root_ca_extensions




[ root_ca_distinguished_name ]

commonName = hostname




[ root_ca_extensions ]

basicConstraints = CA:true

keyUsage = keyCertSign, cRLSign




[ client_ca_extensions ]

basicConstraints = CA:false

keyUsage = digitalSignature

extendedKeyUsage = 1.3.6.1.5.5.7.3.2




[ server_ca_extensions ]
```

```
basicConstraints = CA:false

keyUsage = keyEncipherment

extendedKeyUsage = 1.3.6.1.5.5.7.3.1
```

Create your own Certificate Authority

The messaging container already has a Certificate Authority certificate, created at `/home/testca/cacert.pem`.

You can create your own with the following command:

```
# Prepare the certificate authority (self-signed).

cd /home/testca

mkdir private

mkdir certs

touch index.txt




# Create a self-signed certificate that will serve a certificate authority (CA).

# The private key is located under "private".

openssl req -x509 -config openssl.cnf -newkey rsa:2048 -days 1825 -out cacert.pem -
outform PEM -subj /CN=MyTestCA/ -nodes




# Encode our certificate with DER.

openssl x509 -in cacert.pem -out cacert.cer -outform DER
```

The output of this command is the **cacert.pem** file. You will use this to sign the certificate file, generated in the next step.

Create your own certificate

The Messaging container includes a certificate which has been signed using the local certificate authority certificate. These are the instructions which can be used to create and sign a certificate for use with the Messaging container.

Create a server key and sign it with the Certificate Authority certificate

Substitute **<hostname>** in the command below, with the hostname of your Messaging container (default: *'messaging'*).

```
cd /home/server




# Generate a private RSA key.

openssl genrsa -out key.pem 2048




# Generate a certificate from our private key.

openssl req -new -key key.pem -out req.pem -outform PEM -subj /CN=$(<hostname>)/
O=server/ -nodes




# Sign the certificate with our CA.

cd /home/testca

openssl ca -config openssl.cnf -in /home/server/req.pem -out /home/server/cert.pem -
notext -batch -extensions server_ca_extensions -create_serial
```

Create a keystore that contains this certificate

```
cd /home/server

openssl pkcs12 -export -out keycert.p12 -in cert.pem -inkey key.pem -passout
 pass:roboconf
```

Make your certificate, CA certificate and keystore available to the messaging container.

Externalise volumes of the container

To allow your messaging container to use the certificate, CA certificate and KeyStore file from the previous steps, you must externalise the following directories of the container:

- /home/server
- /home/testca

To do this, add the following lines to the **volumes:** section of the file **docker-compose-messaging.yml**, that you use to start the Messaging container:

```
volumes:
# Externalise the Certificate, CA Authority and Keystor files
# If you want to create your own SSL certificate un-comment the following lines so that
 the
# server Certificate and Certificate Authority Certificate and Keystore file can be
 provided externally.
  - "./rabbitmqServerCert:/home/server"
  - "./rabbitmqCACert:/home/testca"
```

> **WARNING**
>
> If you upgrade to a later version of TDM Portal in Docker, you need to repeat this step in the new **docker-compose-messaging.yml** file.

Copy the new files to these volumes

Copy the Certificate Authority Certificate, Certificate and Key file to the following locations:

- `./rabbitmqCACert/cacert.pem`
- `./rabbitmqServerCert/cert.pem`
- `./rabbitmqServerCert/key.pem`

Now when you start the Messaging container with **docker-compose-messaging.yml**, the container uses your Certificates.

TDM Portal Masking Engine container

The TDM Portal Masking Engine container performs masking tasks with the Fast Data Masker engine. It receives tasks from the TDM Portal Messaging container, which receives tasks from the TDM Portal container.

For more information on this process, see Scalable masking with Docker.

How to use the Masking container

We recommend that you start the TDM Portal Masking Engine container with the supplied file **docker-compose-masking.yml** (expand below). For more information, see Docker-compose files.

Alternatively, you can Start Containers with the 'docker run' Command (this is for Advanced users).

docker-compose-masking.yml

```
version: '3.5'


networks:
  default:
    name: tdm_default
services:
  masking:
    image: tdm/
masking:<version>
    environment:
      - 'FDM_LICENSE=<Paste your FDM license here>'
      - 'MESSAGING_SERVER=messaging'
      - 'MESSAGING_PORT=5671'
      - 'MESSAGING_USER=Admin'
      - 'MESSAGING_PASS={cry}1hY5pZrm87PWjgPdmypDbVZnL4a108lxy8YLuUVRMCr8'
    volumes:
      - 'masking_storage:/mnt/storage'
      - 'masking_logs:/mnt/logs'
      - 'masking_seedtables:/mnt/seedtablesCustom'
volumes:
  masking_logs:
  masking_storage:
  masking_seedtables:
```

You can add this container (or multiple instances of this container) to an active Docker network, to distribute your masking jobs across more instances of the Fast Data Masker engine concurrently. For more information, see Scale the masking service.

Environment Variables

You can define the following variables in the `environment` section of the **messaging** service, in **docker-compose-masking.yml.**

- **MESSAGING_SERVER**
  Specifies the **hostname** of the Messaging container. Default: *messaging*.
- **MESSAGING_PORT**
  Specifies the port number of the Messaging container. Default: *5671*.
- **MESSAGING_USER**
  Specifies the username with which the service accepts tasks from the Messaging container. Must match the value of **messaging.username** (Portal in Windows) or **MESSAGING_USER** (Portal in Docker), and **DEFAULT_USER** in the Messaging container.
- **MESSAGING_PASS**
  Specifies the password for the user with which the service accepts tasks from the Messaging container. Must match the value of **messaging.password** (Portal in Windows) or **MESSAGING_PASS** (Portal in Docker), and **DEFAULT_PASS** in the Messaging container.

Data Volumes

You can use the following volumes to persist data from the Masking Engine container:

- **masking_storage**
  Stored by default at `/mnt/storage`
- **masking_logs**
  Stored by default at `/mnt/logs` This volume is divided into two subdirectories:
  - **/fdm**
    Masking Service logs
  - **/tdm**
    Messaging Service logs
- **masking_seedtables**
  You can use your own custom seedtables for masking data. These custom seedtables must be in this volume for the Masking Engine to be able to use them.

Use Custom Seedtables with the Masking Engine container

To use custom seedtables, we recommend that you use the `docker cp` command to copy all the seedtables you want to use, into the volume you define with the environment variable **masking_seedtables**.

Follow these steps:

1. Identify the name of the Masking Engine container to which you want to make your custom seedtables available.
   Example: *tdm_masking_1_a3189d18d209*

   TIP

   Use the `docker ps` command to see a list of the containers in your Docker network.
2. Copy all the seedtables you want to make available to your TDM Portal Masking container, into a local directory on your machine (for example **/home/mySeedTables**). Navigate to this directory.
3. From this directory, execute the `docker cp .` command to copy the entire contents of the directory to the **masking_seedtables** volume on your Masking Engine Docker container (defined as `/mnt/seedtablesCustom` in **docker-compose-masking.yml**).

   ```
   $ docker cp . tdm_masking_1_a3189d18d209:/mnt/seedtablesCustom
   ```

   `docker cp` copies the contents of the directory to the **masking_seedtables** volume on your Masking Engine Docker container, and the seedtables are available for the container to use.

File Packages available

To use TDM Portal on Linux or other platforms, it is necessary to execute a **docker-compose.yml** file (or multiple **docker-compose*.yml** files), that creates a Docker network of containers. For more information, see Docker-compose files.

You need either to have an Oracle database available to your Docker network, or to Create an Oracle database container.

Download or create your own Docker images

There are 3 zipped file packages available for installation. From each one, you can get or build Docker images.

WARNING

The process to build Docker images is for Advanced users.

For more information, see Build your own Docker images.

Note on EULA

In each scenario below, it is necessary to unzip the relevant **.zip** file, and untar the **.tgz** file that the zip file contains. You should then run **install.sh** to agree with the EULA and extract the contents of **install.zip**. You cannot extract the contents of **install.zip** manually.

WARNING

Execution of the **install.sh** script constitutes agreement with the terms of this EULA.

File Packages Available

The following three file packages are available, from which you can get or build Docker images:

- **Packages with pre-built Docker images**
  - script **- CA TEST DATA MANAGER PORTAL FOR DOCKER <*version_number*>**
    This package contains the following files:
    - EULA (in .txt and .rtf formats).
    - Shell script to pull images from the CA Docker Registry.
    - Sample Docker compose file
    - Readme file
  - **Image Bundle Package - CA TEST DATA MANAGER PORTAL FOR DOCKER (IMAGE BUNDLE) <*version_number*>**
    This package contains the following files:
    - EULA (in .txt and .rtf formats).
    - Zipped Docker images
    - Sample Docker compose files
    - Readme file
- **Package to build your own Docker images**You can use package file **CA TEST DATA MANAGER PORTAL FOR DOCKER (IMAGE KIT) <*version_number*>** to build your own TDM Portal Docker images. This process is for Advanced users.
  For more information, see Build your own Docker images.

How to use packages with pre-built Docker images

Script Package

This package contains a script that you can run to pull Docker images from the CA Docker Registry.

Follow these steps:

1. Download **CA TEST DATA MANAGER PORTAL FOR DOCKER <*version_number*>** from support.ca.com.
2. Unzip the file package you download. This contains the file **TDM_Portal_docker_pull-<*version_number*>.tgz.**
3. Untar TDM_Portal_docker_pull-**<*version_number*>**.tgz with the following command:

```
tar -xzf TDM_Portal_docker_pull-<version_number>.tgz
```

4. Run **install.sh**.
   A prompt asks you to accept the license agreement.
5. To accept the license agreement, press **y**.
   The script extracts the contents of **install.zip**.

6. Run the shell script to log in to tdm.packages.ca.com and download all TDMWeb images.

   TIP

   Execute the command `docker images` to check that the images are in your Docker registry.
7. After a successful pull operation, you can customize and execute the **docker-compose.yml** file from this package.

Image Bundle Package

Download Docker images, and make these Docker images available to the Docker network (with the `docker load` command).

Follow these steps:

1. Download **CA TEST DATA MANAGER PORTAL FOR DOCKER (IMAGE BUNDLE) <*version_number*>** from support.ca.com.
2. Unzip the file package you download**.** This contains the file **TDM_Portal_docker-<*version_number*>.tgz.**
3. Untar TDM_Portal_docker-**<*version_number*>**.tgz with the following command:

   ```
   tar -xzf TDM_Portal_docker-<version_number>.tgz
   ```

4. Run **install.sh**.
   A prompt asks you to accept the license agreement.
5. To accept the license agreement, press **y**.
   The script extracts the contents of **install.zip**.
6. Load each Docker image to the local Docker image repository, with the following commands:

   ```
   gunzip -c ./TDM_images/orientdb/orientdb-2.2.33.tgz | docker load

   gunzip -c ./TDM_images/tdmtools/tdmtools-<version>.tgz | docker load

   gunzip -c ./TDM_images/tdmweb/tdmweb-<version>.tgz | docker load

   gunzip -c ./TDM_images/action-service/action-service-<version>.tgz | docker load

   gunzip -c ./TDM_images/masking/masking-<version>.tgz | docker load

   gunzip -c ./TDM_images/messaging/messaging-<version>.tgz | docker load
   ```

7. Now that the Docker images are available to Docker, you can customize and execute one of the following docker-compose files:
   - **docker-compose.yml**
     Use this docker-compose.yml file if you have an Oracle database on which to store the **gtrep** repository.
   - **docker-compose-ora.yml**
     Use this docker-compose.yml file if you wish to create an Oracle container on your Docker network.

Docker-compose files

With Docker, you can:

- Create Docker containers from Docker images.
- Build Docker images from Dockerfiles and source files.

In both cases, we recommend that you synchronize these commands with a **docker-compose.yml** file. You can use the `docker-compose up` command to start one or more docker-compose.yml files; all the services (i.e. containers) you start with the `docker-compose up` command exist on the same Docker network, which means that they can communicate with each other. Docker services identify themselves by their **hostname** parameter.

How to use docker-compose files

To execute a docker-compose.yml file, use a `docker-compose up` command similar to this one:

```
docker-compose -f docker-compose.yml -f docker-compose-messaging.yml -f docker-compose-masking.yml up -d [--scale masking=3]
```

Where:

- **-f**Each **-f** flag defines another docker-compose.yml file to add to the network. The `docker-compose` command adds services from each docker-compose.yml file to your Docker network.

  WARNING

  The command adds or reconfigures services in the order in which you supply docker-compose.yml files to the command.

  For example, If you define a service (e.g. TDMWeb) in **docker-compose.yml**, and then define it again in **docker-compose-ora.yml**, the second set of parameters overwrites the first set.
- (Optional) **--scale** *masking=n*
  This creates *n* number of instances of the service *masking* (i.e. the Scalable masking with Docker.

The example above starts a Docker network with the following services:

- OrientDB (from **docker-compose.yml**)
- TDMWeb (from **docker-compose.yml**)
- Message Bus Server (from **docker-compose-messaging.yml**)
- 3 Masking Engines (from **docker-compose-masking.yml**, **--scale masking=3**)

Customize the docker-compose.yml files you need, to reflect your configuration (for example, parameters of your **gtrep** repository).

TIP

Tip: See the **README-RUN.md** and **README-BUILD.md** files for details on how to combine these docker-compose.yml files for different use cases.

Name of the Docker network

To start other Docker containers on your Docker network, either TDM Portal REST ActionService container) or with re-execution of the `docker compose up` command, the **--network** parameter must match the name of your Docker network's name.

NOTE

In all the docker-compose*.yml files we provide, the network name is ***tdm_default*** .

To change the name of the Docker network you create with your docker-compose.yml file, amend the **name** parameter in the **networks** section of each docker-compose.yml file you call in your `docker-compose` command:

```
networks:
```

```
  default:

    name:
tdm_default
```

Available docker-compose files

TDM Portal's functionality in Docker is available in the following docker-compose*.yml files, available from http://casupport.broadcom.com:

- Files available to **start services from images:**
  - **docker-compose.yml**
    The base for a TDM Portal environment. This starts the OrientDB services.
  - docker-compose-messaging.yml
    This starts the Message Bus Server service, necessary for scalable masking.
  - docker-compose-masking.yml
    This starts the Masking Engine service, for scalable masking.
  - **docker-compose-ora.yml**
    This starts the Oracle database service.

    WARNING

    It is necessary to build this container yourself before you can use **the docker-compose-ora.yml** file. It is intended for Advanced users and should not be used in a production environment.
- Files available to **build images:**
  - **docker-compose-build.yml**
    This builds the following images:
    - Java and Tomcat images, necessary to build further images.
    - TDM Portal (TDMweb)
    - OrientDB
    - REST Action Service
    - TDM Portal Tools
    - Message Bus Server
    - Masking Engine
  - **docker-compose-build-ora.yml**
    This builds the same images as **docker-compose-build.yml** above, and in addition images to create:
    - Oracle database with gtrep user. This Oracle database should be used for demo purposes only.

      WARNING

      Before you execute this docker-compose file, it is also necessary to build the official Oracle Docker container. For more information, see TDM Portal Oracle database container.

  For more information, see Docker-compose Files to Build Images.

Features not available in TDM Portal in Docker

Some features are not available in TDM Portal in Linux.

The following features are not supported in TDM Portal 4.7 for Docker:

- **Javelin integration**
- **ALM integration**
- **Rally integration**
- **Test Match**
- **Windows-specific Actions** (of type HOST and WORKFLOW)
- **Pre/Post Publish Actions** that are implemented as Windows batch scripts (see TDM Portal REST ActionService container for alternative solutions)
- **GT Service Layer integration**

Advanced Use of TDM Portal in Docker

This section contains information on Advanced uses of the Docker software. We do not recommend these use cases for standard users of TDM Portal.

- **Create an Oracle database container**
  This is necessary if you do not have an Oracle database available for installation of the gtrep repository.
- Start Containers with the 'docker run' Command
  This is an alternative way to start Docker containers (instead of docker-compose).
- **Build your own Docker images**
  If you wish to build TDM Portal Docker images based on, for example, an alternative Linux distribution, you can do so with the Image Kit available.

TDM Portal Oracle Database Container

If you are not able to provide an Oracle database in which TDM Portal in Docker can create the **gtrep** repository, you can run a Docker container to provide this service on your TDM Portal Docker network. This Oracle database includes a user called *gtrep*.

> **WARNING**
>
> The TDM Portal Oracle database container is for demo purposes only. For production use, the container would require further configuration from your database administrator.

The Docker image to create this container is **not** available with the Image Kit - you must build this image yourself.

> **NOTE**
>
> To build an Oracle database Docker image, you need to download the official Oracle Database image kit from Oracle. Please review Oracle's License Agreement before you use this software.

Steps to create Oracle Database Docker image

To build a Docker image, from which you can start an Oracle Database container, it is necessary to:

1. Download, modify and build the official Oracle Database Docker image
2. Download and build the TDM Portal Oracle Database Docker image

Download, modify and build the official Oracle Database package

The Dockerfile that creates a TDM Portal Oracle Database container, requires a slightly modified version of **Oracle Database 11g Release 2 (11.2.0.2) Express Edition** in order to build.

Follow these steps:

1. Download **Oracle Database 11g Release 2 (11.2.0.2) Express Edition**.

You can download this package from https://github.com/oracle/docker-images/tree/master/OracleDatabase/SingleInstance.

2. In the Dockerfile from this package(`11.2.0.1/Dockerfile.xe`), remove or comment out the following line:
   `VOLUME ["$ORACLE_BASE/oradata"]`
3. Build the image `oracle/database:11.2.0.2-xe` , with the following command:
   `./buildDockerImage.sh -v 11.2.0.2 -x`

You are now ready to build the TDM Portal Oracle Database Docker image.

Download and build the TDM Portal Oracle Database Docker image

After you complete the procedure above, you can build the TDM Portal Oracle Database image.

Follow these steps:

1. Download the TDM Oracle Database Dockerfile (`officialoracle-gtrep`) from support.ca.com.
2. Build the image `tdm/officialoracle-gtrep:11.2.0.2-xe` . You can do this one of two ways:
   a. Execute the file **docker-compose-build-ora.yml**, to build TDM Portal Oracle Database Docker image, and all other images.
   For more information, see Docker-compose Files to Build Images.
   b. Build the TDM Portal Oracle Database Docker image with the following `docker build` command:
   ```
   docker build . -t tdm/officialoracle-gtrep:11.2.0.2-xe -f Dockerfile.gtrep --no-cache --build-arg dbUserPassword='Gridt00ls' --shm-size='1GB'
   ```
   Optional build argument:
   - **dbUserPassword**Password for the user *gtrep*, that the command creates. Default: '`Gridt00ls` '.

   NOTE

   The `docker build` command may take some time.

The TDM Portal Oracle Database image is now ready to use to create an Oracle Database container on your TDM Docker network.

Start the Oracle database container

To add the TDM Portal Oracle database container `officialoracle-gtrep` to your Docker network, you can either:

- Execute **docker-compose-ora.yml** with the `docker-compose up` command.
  For more information, see Docker-compose files.
- Start the container with the docker run command.
  For more information, see Start Containers with the 'docker run' Command.

Start Containers with the 'docker run' Command

We recommend that you use the `docker-compose up` command to start the containers necessary to run TDM Portal in Docker, because this method creates all the Docker containers on the same Docker network.

Alternatively, you can start Docker containers from Docker images with the `docker run` command. This page explains how to use this command to start Docker containers.

NOTE

All Docker containers you wish to run together must have the same **network** parameter. This parameter is not necessary with the `docker-compose` command.

Syntax

To start a Docker container from a Docker image with the `docker run` command, it is necessary to create a command similar to the following:

```
docker run --name=<container_name> --network=<my_TDM_network> \

          -h <host_name> -v <volume:volume_location> \

          -e <environment_variable1="value_a"> \

          -e <environment_variable2="value_b"> \
          "path_to_docker_image"
```

Where:

- **--name**
  Defines the name by which you can identify this container. This value is equivalent to the name of the `service` in a docker-compose.yml file.
- **--network**
  Defines the network on which this container runs. For more information, see Define the name of the Docker network in your docker-compose.yml file.
- **-h**
  Defines the hostname on which this container runs, i.e. the name with which other containers can interact with this container. This value is equivalent to the parameter `hostname` in a docker-compose.yml file.
- **-v**
  Defines volumes in which this container stores data. See State Persistence in Volumes.
- **-e**
  Defines environment variables for the container. Each variable is equivalent to an entry under the `environment` section of a docker-compose.yml file.

Example:

The following `docker run` command creates a TDM Portal container with the name and hostname `tdmweb` on the Docker network `tdm-net` from the image **tdm/tdmweb:4.7.0.14**:

```
docker run -it -p 8080:8080 -p 8443:8443 \
    --name=tdmweb --network=tdm-net \
    -v storage:/mnt/storage \
    -h tdmweb \
    -e GTREP_PASSWORD="marmite"  \
    -e GTREP_HOST="my_oracle" \
    -e ORIENTDB_HOST="my_orientdb" \
    -e GTREP_SERVICE_NAME="gtrep" \
    "tdm/tdmweb:4.7.0.14"
```

For more information on this container's environment variables, see TDM Portal container - Environment Variables.

Build your own Docker images

It is possible to build your own TDM Portal Docker images, from source binaries and Dockerfiles. The Image Kit is available for this purpose.

Follow these steps:

1. Download **CA TEST DATA MANAGER PORTAL FOR DOCKER (IMAGE KIT) <*version_number*>** from support.broadcom.com.
2. Unzip the file that you download. This contains the file **TDM_Portal_docker_src-<*version_number*>.tgz.**
3. Untar **TDM_Portal_docker_src-<*version_number*>.tgz** with the following command:
   ```
   tar -xzf TDM_Portal_docker_src-<version_number>.tgz
   ```
4. Run **install.sh**.
   A prompt asks you to accept the license agreement.
5. To accept the license agreement, press **y**.
   The script extracts the contents of **install.zip**.
6. **Download Oracle client**

   NOTE

   Due to licensing policy, you need to download additional software from Oracle. Please confirm that your Oracle license permits you to use this software under the terms of the License Agreement.

   a. Go to **Instant Client Downloads for Linux x86-64 (64-bit)** at https://www.oracle.com/database/technologies/instant-client/linux-x86-64-downloads.html and download the following files:
      - instantclient-basic-linux.x64-12.2.0.1.0.zip
      - instantclient-sqlplus-linux.x64-12.2.0.1.0.zip
      - instantclient-tools-linux.x64-12.2.0.1.0.zip

   b. Copy these files to `./target/CATDMWebDocker/instantclient`
   c. Go to Java SE 8 Archive Downloads at https://www.oracle.com/technetwork/java/javase/downloads/java-archive-javase8-2177648.html and download the file `jre-8u172-linux-x64.tar.gz`
   d. Copy the file to `./target/java`
7. **Build Docker images**
   To build Docker images from Dockerfiles, you can either:
   – **(Recommended)** Customize and execute one of the following Docker-compose Files to Build Images:
      - **docker-compose-build.yml**
        Use this docker-compose file if you have an Oracle database for Docker to write to.
      - **docker-compose-build-ora.yml**
        Use this docker-compose file if you wish to create an Oracle container on your Docker network.

        WARNING

        This Oracle database should only be used for demo purposes. Before you execute this docker-compose file, it is necessary to build the official Oracle Docker container. For more information, see TDM Portal Oracle Database Container.

   – Build each image separately.
     Follow these steps:
     a. Change current directory to location where the contents of **TDM_Portal_docker_src-<*version_number*>.tgz** are unpacked.
     b. Run the following commands, in this order:
        a. `docker build -t tdm/java:8u172 -f Dockerfile.java .`
        b. `docker build -t tdm/tomcat:8.5.32 -f Dockerfile.tomcat .`
        c. `docker build -t tdm/orientdb:2.2.33 -f Dockerfile.orientdb .`

d. `docker build -t tdm/tdmweb:<version> -f Dockerfile.tdmweb .`
e. `docker build -t tdm/tdmtools:<version> -f Dockerfile.tdmtools .`
f. `docker build -t tdm/action-service:<version> -f Dockerfile.action .`
g. `docker build -t tdm/messaging:<version> -f Dockerfile.messaging .`
h. `docker build -t tdm/masking:<version> -f Dockerfile.masking .`

Docker-compose Files to Build Images

Once you download the necessary TDM Portal Docker Image Kit and Oracle client, you can build Docker images from which to create TDM Portal containers. We recommend that you build these images with one of the following docker-compose.yml files.

Build images (excluding Oracle database image)

Execute this docker-compose.yml file to build images from Dockerfiles and binaries from the Image Kit. This method assumes that you have an Oracle database to use for your **gtrep** repository.

docker-compose-build.yml

```
version: '3.5'




services:

  java:

    build:

      context: .

      dockerfile: Dockerfile.java

    image: tdm/java:8u202

  tomcat:

    build:

      context: .

      dockerfile: Dockerfile.tomcat

    image: tdm/tomcat:9.0.12

    depends_on:
```

```
    - java
orientdb:
  build:
    context: .
    dockerfile: Dockerfile.orientdb
  image: tdm/orientdb:2.2.33
  depends_on:
    - java
tdmweb:
  build:
    context: .
    dockerfile: Dockerfile.tdmweb
  image: tdm/tdmweb:<version>

  depends_on:
    - tomcat
action:
  build:
    context: .
    dockerfile: Dockerfile.action
  image: tdm/action-service:<version>

  depends_on:
    - java
tdmtools:
  build:
```

```
      context: .
      dockerfile: Dockerfile.tdmtools
    image: tdm/tdmtools:<version>

    depends_on:
      - java
  messaging:
    build:
      context: .
      dockerfile: Dockerfile.messaging
    image: tdm/messaging:<version>

  masking:
    build:
      context: .
      dockerfile: Dockerfile.masking
    image: tdm/masking:<version>

    depends_on:
      - tomcat
```

Build images (including Oracle database image)

Execute this docker-compose.yml file to build images from Dockerfiles and binaries from the Image Kit. This method also creates an image from which you can start an Oracle database container for your **gtrep** repository.

> **WARNING**
>
> Before you execute this docker-compose file, it is also necessary to build the official Oracle Docker container. For more information, see TDM Portal Oracle database container.

docker-compose-build-ora.yml

```
version: '3.5'
```

```
services:
  java:
    build:
      context: .
      dockerfile: Dockerfile.java
    image: tdm/java:8u202
  tomcat:
    build:
      context: .
      dockerfile: Dockerfile.tomcat
    image: tdm/tomcat:9.0.12
    depends_on:
      - java
  orientdb:
    build:
      context: .
      dockerfile: Dockerfile.orientdb
    image: tdm/orientdb:2.2.33
    depends_on:
      - java
  tdmweb:
    build:
```

```
    context: .
    dockerfile: Dockerfile.tdmweb
  image: tdm/tdmweb:<version>

  depends_on:
    - tomcat
action:
  build:
    context: .
    dockerfile: Dockerfile.action
  image: tdm/action-service:<version>

  depends_on:
    - java
tdmtools:
  build:
    context: .
    dockerfile: Dockerfile.tdmtools
  image: tdm/tdmtools:<version>

  depends_on:
    - java
messaging:
  build:
    context: .
    dockerfile: Dockerfile.messaging
  image: tdm/messaging:<version>
```

```
  masking:
    build:
      context: .
      dockerfile: Dockerfile.masking
    image: tdm/masking:<version>

    depends_on:
      - tomcat
  officialracle-gtrep:
    build:
      dockerfile: Dockerfile.gtrep
      shm_size: '1G'
    image: tdm/officialoracle-gtrep:11.2.0.1
```

Install Product Components

The GT Server installer installs all Test Data Manager components and their prerequisites.

> **NOTE**
>
> The GT Server installer **does not** install the TDM Portal. For TDM Portal installation, see Install TDM Portal for Windows.

When you launch the installer, it lists all prerequisites and components and lets you choose which ones to install.

Notes on Installation

We recommend the following strategies for installing the product components:

- Keep all of the prerequisites selected, even if you are planning a distributed installation of the product components. The complication of tracking which prerequisites belong with which components has a high margin for error.
- Install only the product components that you need. However, consider that there are dependencies between components that you might not be aware of. The Datamaker UI provides links to several other components, such as GT Subset and Javelin. If you did not install these components, the links will not work.

  - Example: If you configure TDM Portal to use Datamaker for Self-Service publishing, ensure that the Portal and Datamaker are installed on the same server.
  - Example: If you want to publish from Javelin, ensure that Datamaker is installed on the same machine as Javelin.
- You can install all product components on a single server as long as it meets the System Requirements. However, if you distribute the installation across servers, consider the aforementioned dependencies and only distribute the components that can run independent of other components.

Installation Process

Follow these steps:

1. Download and extract the files in the installation media.
2. Right-click the setup_GTServer_*version*.exe file and select **Run as Administrator** to launch the installer.
   The installation wizard opens to the **Prerequisites Wizard** page.
3. Click **Next**, accept the license agreement, and click **Next** again.
   The Prerequisites page appears. This page contains all prerequisites and product components. If you already have any of the prerequisites or components installed, the installer detects this and clears the check box.
4. Select the items to install on this server and click **Next**.
   The GT Server setup installs each prerequisite and component in the order listed.
5. Consider the following points as the GT Server progresses through the installation:
   - When one installation finishes, click **Finish** on that installer and the GT Server automatically launches the next installer.
   - For typical installations, you can simply leave the defaults selected and move through each installer without changing any information.

   After all installers are complete, you have successfully installed Test Data Manager.
6. Verify the existence of desktop icons for the components Datamaker, EDI, Javelin and Fast Data Masker.

The GT Server installer creates installation logs in the Temp folder (%TEMP%). You can find the log files during both fresh installation and upgrade cases. A typical log file name has the following format:

<componentname_version.log>

For example, the **GT HP ALM Service** version **1.2.3.4** creates a file named "`GT HP ALM Service_1.2.3.4.log` " in the Temp folder during installation.

Install Fast Data Masker on Linux

You can install Fast Data Masker on Red Hat Enterprise Linux 6.0 and 7.0 if you want to use Fast Data Masker in a Linux environment. You can install only one instance of Fast Data Masker on a computer. After you install the application, you can launch, uninstall, upgrade, or reinstall it, as appropriate.

Prerequisites

1. Verify that Java version 1.8 or higher is already installed on the computer on which you want to install Fast Data Masker.
2. Copy the Fast Data Masker installer from the installation media to your Linux computer.
3. Navigate to the location (on your Linux computer) where you copied the Fast Data Masker installer. For example, `# cd /root/FDM`
4. Locate the FDM.bin file:
   ```
   # cd FDM_Installer_Linux/Disk1/InstData/NoVM
   # ls
   ```
5. Give the binary file execute permission:
   ```
   # chmod 755 FDM.bin
   ```

Install Fast Data Masker in GUI Mode

If your Linux system supports graphical user interfaces, use the GUI-based installer. Alternatively, use console mode for installation.

1. Run the FDM.bin file to install Fast Data Masker on Linux as follows:
   `# ./FDM.bin`
   The command extracts the installation resources from the installer package, configures the installer for the environment, and launches the installer GUI.
2. Click **Next** on the Welcome dialog.
3. Read and accept the license agreement and click **Next**.
4. Browse to the directory location where you want to install the application and click **Next**.
   Default: `/opt/CA/FastDataMasker`
5. Enter the default shell path and click **Next**.
   Default: `/bin/bash`
6. Specify whether and where you want to create symbolic links (symlinks) after the installation:
   - **In your home folder**
     Lets you create symbolic links in the home directory after successful installation.
   - **Other**
     Lets you browse to the location where you want to create symbolic links after successful installation.
   - **Don't create links**
     Specifies that you do not want to create symbolic links after successful installation.
7. Review the installation summary and click **Install** to start the installation.
   A dialog displaying installation progress opens. When the installation is done, the **Install Complete** dialog opens.
8. Click **Done** to finish the installation.
   You have successfully installed Fast Data Masker in your Linux environment.

You can now verify the installation by launching the Fast Data Masker GUI. You can also uninstall, upgrade, and reinstall the application, if necessary.

Install Fast Data Masker in Console Mode

If your system does not support graphical user interfaces, use the console installer. Alternatively, use GUI mode for installation.

> **TIP**
>
> To change something on a previous step, type 'back'. To cancel this installation at any time, type 'quit'. To proceed, press Enter.

1. Run the FDM.bin file to install Fast Data Masker on Linux as follows:
   `# ./FDM.bin`
   The command extracts the installation resources from the installer package, configures the installer for the environment, and launches the installer in console mode.
2. Read the Welcome screen and press Enter.
3. Read the license agreement. Press Enter repeatedly to scroll down. Enter Y to accept the license.
4. Type the absolute path of the directory where you want to install the application, and press Enter.
   Default: `/opt/CA/FastDataMasker`
5. Type the Unix default shell path and press Enter.
   Default: `/bin/bash`
6. Specify whether and where you want to create symbolic links (symlinks) after the installation:
   - **Default:** `/root`
     Creates the default symbolic links in the top level directory.
   - **In your home folder**

Lets you create symbolic links in the home directory after successful installation.

- **Choose Another Location**
  Lets you browse to the location where you want to create symbolic links after successful installation.
- **Don't create links**
  Specifies that you do not want to create symbolic links after successful installation.

7. Press Enter to start the installation.
   You have successfully installed Fast Data Masker in your Linux environment.

You can now verify the installation by running a sample masking operation. You can also uninstall, upgrade, and reinstall the application, if necessary.

NOTE

You cannot launch the Fast Data Masker GUI from the console.

Launch the Application

After you install the application, launch it to verify that the installation has completed without any issue. You cannot launch the application from a console.

1. Navigate to the location (for example, `/opt/CA/FastDataMasker`) where you installed the application:
   `# cd /opt/CA/FastDataMasker/`
2. List the contents of the directory:
   `# ls -l`
   All the installed files are available in the location.
3. Locate and run the FastDataMasker file:
   `# ./FastDataMasker`
   The Fast Data Masker UI opens. You can use the UI to get started with the data masking process.

Note: If you created symbolic links in your home location at the time of installation, you can launch Fast Data Masker from that location as follows:

```
# cd /root

# ./FastDataMasker
```

Uninstall the Application

If you no longer want to use installed Fast Data Masker instance, you can uninstall the application.

1. Navigate to the location (for example, `/opt/CA/FastDataMasker`) where you installed the application:
   `# cd /opt/CA/FastDataMasker/`
2. List the contents of the directory:
   `# ls -l`
   All the installed files are available in the location.
3. Locate and navigate to the `_Fast Data Masker_installation` directory:
   `# cd _FastDataMasker_installation/`
4. List the contents of the directory:
   `# ls -l`
5. Locate and run the following file to uninstall the application:
   `# ./Uninstall_FastDataMasker_Installation`
   The command opens the **Uninstall Fast Data Masker Installation** dialog.
6. Click **Uninstall** to start the uninstallation.
   A dialog displaying uninstall status opens. When the uninstallation is complete, the **Uninstall Complete** dialog opens.
7. Click **Done** to finish the uninstallation process.

You can navigate to the installation directory and can verify that only the `Logs` directory is available. All other files and directories are removed from the install location.

Note: If you created symbolic links in your home location at the time of installation, you can launch Fast Data Masker uninstallation from that location as follows:

```
# cd /root
# ./Uninstall_FastDataMasker_Installation
```

Upgrade to a Newer Version

If you want to upgrade the application to a newer version, you can do so by using the appropriate new installer.

1. Run the `FDM.bin` file for the new installer as explained in the installation section.
2. Click **Next** on the Welcome dialog.
   A dialog displaying an upgrade message opens.
3. Review the message and click **Next** to upgrade your existing instance.
4. Verify the auto-populated default shell path and click **Next**.
5. Verify the application links information and click **Next**.
6. Review the summary and click **Next**.
7. Click **Done** when the upgrade is done.
8. Launch the Fast Data Masker application as explained in the application launch section.
9. Verify the new functionality in the interface to confirm that you have successfully upgraded to a new version of the application.

Reinstall the Application

If you want to reinstall Fast Data Masker for any reason, you can do so by using the same installer that you used for the initial installation.

1. Run the `FDM.bin` file as explained in the installation section.
2. Click **Next** on the Welcome dialog.
   A dialog displaying a reinstall message opens.
3. Review the message and click **Next** to reinstall the application.
4. Verify the auto-populated default shell path and click **Next**.
5. Verify the application links information and click **Next**.
6. Review the summary and click **Next**.
7. Click **Done** when the upgrade is done.
8. Launch the Fast Data Masker application as explained in the installation section.
9. Verify the functionality in the interface to confirm that you have successfully reinstalled the application.

Activate Test Data Manager

From Test Data Manager 4.9 on, you can activate the product in one of the following two ways:

- **Broadcom Portfolio License Agreement (PLA)**
  This license gives you access to all Continous Delivery products. You are charged by usage of the products.

  WARNING

  Under the terms of the Broadcom PLA, it is mandatory to send telemetry data for your product usage. For more information, see Configure Telemetry.
- **Standard TDM License**
  The standard license is a stand-alone license for the Test Data Manager product only. Telemetry is optional.

This page contains information on the following topics:

Activate TDM and complete Telemetry Configuration

When you activate TDM, you also need to provide details for the storage of telemetry. If you use TDM under the terms of a PLA, TDM sends this telemetry information to Broadcom.

> **WARNING**
>
> If you use TDM under the terms of a Broadcom Portfolio Licensing Agreement (PLA), this telemetry information is necessary for TDM to function.

If you do not use TDM under the terms of a PLA, TDM stores your telemetry information for your reference, but does not send it to Broadcom.

Broadcom Portfolio License Agreement

You can activate the software with details of your Broadcom Portfolio License Agreement (PLA).

First time Activation

When you log into CA TDM Portal as an Administrator for the first time, TDM prompts you to activate the software. This procedure is specific to activation under the terms of a Broadcom PLA.

Follow these steps:

1. Launch Test Data Manager Portal.
2. Log in to CA TDM Portal with your Administrator username and password.

   > **NOTE**
   >
   > If you attempt to log in with a standard username and password and your CA TDM license is invalid, the page reloads with an error message to inform you that the license is invalid.

   The CA TDM Portal home page opens, with the **Activate Product** dialog active.
3. On the **Licensing Model** page, select **Yes** to the question "Is this install or upgrade related to new or additional planned usage under a Portfolio License Agreement?".
4. Click **Next**.
   The **Telemetry Configuration** page opens.
5. Enter the following:
   - **Company Domain**The last part of your company's e-mail address (e.g. **broadcom.com**)
   - **Enterprise Site ID**This is a 4 to 9 digit number. You can find this on your License Agreement or on the CA Support Portal.
   - (Optional) **Internal Identifier**
     This is for your own reference, to track usage of TDM.

   If the **Company Domain** and **Enterprise Site ID** are valid, the **Next** button becomes active.
6. Click **Activate**.
   If the License is valid, the **Product Activation Successful** dialog opens.
7. Click **Done** to go to the TDM Portal home page.

Standard TDM License

From version 4.8, you must apply a standard TDM license in CA TDM Portal - in previous versions, it is also possible in Datamaker.

NOTE

If you applied your CA TDM license in Datamaker in a previous version, this license continues to apply until it expires.

First time Activation / Invalid License

When you log into CA TDM Portal as an Administrator for the first time (or after your license expires), the software prompts you to enter a valid activation key. This procedure is specific to activation with a standard TDM license.

Follow these steps:

1. Launch Test Data Manager Portal.
2. Log in to CA TDM Portal with your Administrator username and password.

   NOTE

   If you attempt to log in with a standard username and password and your CA TDM license is invalid, the page reloads with an error message to inform you that the license is invalid.

   The CA TDM Portal home page opens, with the **Activate Product** dialog active.
3. On the **Licensing Model** page, select **No** to the question "Is this install or upgrade related to new or additional planned usage under a Portfolio License Agreement?".
4. Click **Next**.
   The **Telemetry Configuration** page opens.
5. Enter the following:
   – **Enterprise Site ID**This is a 4 to 9 digit number.

     NOTE

     TDM populates this field automatically, based on your License Key.
   – (Optional) **Internal Identifier**
     This is for your own reference, to track usage of TDM.

   If the **Enterprise Site ID** is valid, the **Next** button becomes active.
6. Click **Next**.
   The **Review Settings** page opens.
7. If you are sure that the details are correct, click **Activate**.
   If the License is valid, the **Product Activation Successful** dialog opens.
8. Click **Done** to go to the TDM Portal home page.

License Management

You can review the details of a valid license in the CA TDM Portal.

From the CA TDM Portal home page, click **Configuration** in the left pane. Click **License Details** from the new options that appear. Here you can see the following details:

- Time until current license's date of expiration.
- Licensee name.
- Site ID.

TDM Portal in Docker

Activation of TDM Portal in Docker behaves the same as in a standard Windows installation. Therefore, activation instructions on this page also apply to activation of TDM Portal in Docker either under the terms of a Broadcom PLA, or with a standard TDM license.

For more information, see TDM Portal container.

Connect Datamaker to the Repository

After you install the product, you must configure a connection to the repository for the product to function.

Connect Datamaker to a Microsoft SQL Server Repository

To connect Datamaker to a Microsoft SQL Server repository, you configure an ODBC connection.

Follow these steps:

1. Launch Datamaker with the **GT Datamaker** icon on the Desktop.
   The first time you launch Datamaker, the '**Create connection profile for Test Data Repository**' dialog should open automatically. If it does not, click the Yellow button on the main Datamaker dialog.
2. Select '**I need to provide credentials for my Test Data Repository**', select Microsoft SQL Server, and click the blue arrow at the bottom right.
   Unless you have already created an ODBC connection, a Not Found dialog opens.
3. Click **OK**.
4. Click the icon next to the **ODBC Sources** drop down list to create a new ODBC connection for the repository.

   WARNING

   If you are using a 64-bit Windows system, launch the ODBC administrator separately from C:\Windows\SysWOW64\odbcad32.exe.

   The ODBC Data Source Administrator dialog opens.
5. Select the System DSN tab and click **Add**.
   The **Create New Data Source** dialog opens.
6. Select SQL Server Native Client 10.0 and click **Finish**.
   The **Create a New Data Source to SQL Server** dialog opens.
7. Give the repository a name and a description, select the SQL Server that contains the repository, and click **Next**.
8. Select the SQL Server authentication option, enter valid login credentials to the database server, and click **Next**.
   Note: You can also use an Active Directory (AD) account for connecting to the Microsoft SQL Server repository in Datamaker. For more information about setting up integrated security to connect to the Microsoft SQL Server repository, see Enable Integrated Security for Repository Access in Datamaker.
9. Specify gtrep as the default database, select both ANSI check boxes, and click **Next**.
10. Leave the defaults on the final page and click **Finish**.
    A setup page opens that summarizes your settings.
11. Click **Test Data Source** and confirm that the connection is successful.
12. Click **OK**, verify that the repository data source you created appears on the **System DSN** tab of the **ODBC Data Source Administrator**, and click **OK** again.
    You are back to the Database Details tab where you have to select the ODBC connection for the repository.
13. Click the Refresh button next to the drop-down list, select the ODBC source you just created, and click the blue arrow at the bottom right.
    The **User Details** tab opens.
14. Select '**Use specified login details**', enter valid database credentials, select **Store Password**, enter the default schema (gtrep), and click the green button at the bottom right.
    This runs a connection test.
15. After you get a successful connection test, click the green check mark at the bottom right.
16. Enter a name for the repository that you will remember, such as Repository or Test Repository, and click **OK.**
17. Click **Yes** to confirm your profile name**.**
    The repository profile you created should appear on the main Datamaker page.

Connect Datamaker to an Oracle Repository

To connect Datamaker to an Oracle repository, you provide valid database connection credentials to Datamaker.

Follow these steps:

1. Launch Datamaker from the **GT Datamaker** icon on the Desktop.
   If you are launching Datamaker for the first time, the '**Create connection profile for Test Data Repository**' dialog should open automatically. If it does not, click the Yellow button on the main Datamaker dialog.
2. Select '**I need to provide credentials for my Test Data Repository**', select Oracle, and click the blue arrow at the bottom right.
3. In the **Database Details** tab, select your database from the drop down list, or use Search button to browse.
4. In **User Details** tab, select **'Use specified login details'**, enter valid database credentials, select **Store Password**, enter the default schema (gtrep), and click the green button at the bottom right.
   This runs a connection test.
5. After you get a successful connection test, click the green check mark at the bottom right.
6. Enter a name for the repository that you will remember, such as Repository or Test Repository, and click **OK.**
7. Click **Yes** to confirm your profile name**.**
   The repository profile you created should appear on the main Datamaker page.

Perform Repository Maintenance

Your repository is where Datamaker stores product data. If you encounter any database-, repository-, or license-related errors, a good first troubleshooting step is to perform Repository Maintenance.

We recommend you run Repository Maintenance after, for example, the following errors:

- `Violation of PRIMARY KEY constraint 'gtrep_project_pk'. Cannot insert duplicate key in object 'dbo.gtrep_project'`
- `Database Error Message: ORA-00001: unique constraint`

Access Repository Maintenance

You can perform Repository Maintenance from the Maintain Connections window in Datamaker.

Follow these steps:

1. Launch Datamaker as administrator.
2. Log in using the default administrator credentials:
   User name: administrator
   Password: marmite
3. Access the main Datamaker window without connecting to a data source.
4. Do one of the following:
   - Press **Ctrl + Alt + M** and enter your credentials.
     The **Maintain Connections** dialog opens.
   - Enable **Maintenance Mode**.

     NOTE

     Maintenance Mode is available for as long as you are logged in. After you exit, maintenance mode is disabled. Re-enable it the next time you need it.

     To enable Maintenance Mode, follow these steps:
     a. Click **Help, About CA Test Data Manager**.
     b. Click the Red Toolbox icon in the bottom left. Close the About screen.
        Datamaker enables 'Maintenance Mode'.

c. Click **Settings**, **Maintain Schemas**.
The **Maintain Connections** window opens.

To access Repository Maintenance, expand **Datamaker Connection Maintenance**, **Datamaker Test Data Repository.**

Perform Maintenance

If repository maintenance is necessary, you can perform the following actions:

- Click **Remote Publish Authorization** and click **Switch Off**, if it is switched on.
- Click **Reset Sequences** and confirm by clicking **Yes**.

TIP

We recommend you perform this action if you see the following error message in Datamaker or in the TDM Portal logs: `"Violation of PRIMARY KEY constraint 'gtrep_project_pk'. Cannot insert duplicate key in object 'dbo.gtrep_project'"`

After you make any changes, close and restart Datamaker. Repository maintenance is complete.

Connect Datamaker to Test Data Source and Target Databases

Test Data Manager allows for three simultaneous database connections:

- Repository
- Data source
- Data target

The data source and target are data sources that you can use to store and publish test data. The source is typically the source of the original data, and the target is where you publish the transformed data. The distinction between source and target does not have to be this clear. If it is not, simply configure the database connections you need and define them as source and target at a later time.

Test Data Manager provides several sample data sources that you can use to familiarize yourself with the product or to serve as real data sources. For them to be available as source and target data sources, you must first install them using the instructions in Install the Repository. Alternatively, you can use existing databases or databases you have created yourself.

You can connect to data sources using Datamaker or the CA TDM Portal. The CA TDM Portal uses JDBC to connect to databases. Datamaker uses ODBC by default. JDBC connections are visible across both interfaces, while ODBC connections are only visible in Datamaker. If you are defining connections in Datamaker, establish a DSN-less connection to be able to work with the connection in the CA TDM Portal.

Follow these steps:

1. Log in to Datamaker.
2. From the main connection screen, ensure that **Repository** is selected under '**Get profiles from**'.

NOTE

We recommend that you store connection profiles in the repository, not in the registry.

3. Click the **Create New Profile** button (yellow folder icon).
The **Create Connection Profile for Data Target** dialog opens.
4. Select the Database Type to which you want to connect. Click the **Forward** button (blue arrows).
The **Create New Profile in Test Data Repository** dialog opens.
5. Enter a name for the profile in the **Profile** field.

NOTE

The **Profile Name** field has a 30 character limit.

6. Select the type of driver you want to use to connect to this database, from the **DBMS** dropdown list.
   Different options appear, depending on what type of DBMS connection you choose.
7. (**DSN-Less ODBC** only) These options vary further, depending on which ODBC driver you select.

NOTE

The **Other Parameters** field is for additional database connection properties. These properties are specific to your database type. See your specific database documentation for a comprehensive list of available properties.

8. When all fields are complete, the **Test Connection** button ('cog' icon) and **Create New Profile** (tick icon) become available.
   When complete, the new connection profile appears on the main connection screen.
9. Click the green **Connect** button to connect to Datamaker using the selected data sources.

You can add new connections at any time, and you can also specify which is the source and which is the target before you connect.

You can edit or delete connection profiles at any time by right-clicking a profile. You can also copy an existing profile so that it can serve as a template for a new profile.

Secure Your TDoD Configuration

If your site uses SSL certificates, you can optionally choose to configure the Test Data on Demand (TDoD) component for secure SSL environments.

Run the following command line to add the SSL certificate binding for an IP Address and port. The certhash value is specific to the certificate that is being created for the install.

```
netsh http add sslcert ipport=0.0.0.0:8090 certhash=0000000000003ed9cd0c315bbb6dc1c08da5e6
 appid={00112233-4455-6677-8899-AABBCCDDEEFF}
```

The following codeblock shows the relevant sections to change in the configuration file:

```
<bindings>
...
  <webHttpBinding>
        <!-- Default (without a name), required since this is a bug in the WCF server? -->
        <binding closeTimeout="00:01:00" openTimeout="00:01:00" receiveTimeout="00:10:00"
 sendTimeout="00:01:00" allowCookies="false"
        bypassProxyOnLocal="false" hostNameComparisonMode="StrongWildcard" maxBufferSize="2147483647 "
 maxBufferPoolSize="2147483647" maxReceivedMessageSize="2147483647" useDefaultWebProxy="true">
          <readerQuotas maxDepth="32" maxStringContentLength="2147483647" maxArrayLength="2147483647"
 maxBytesPerRead="2147483647" maxNameTableCharCount="2147483647" />
          <security mode="Transport">
            <transport clientCredentialType="Windows" proxyCredentialType="None" realm="" />
          </security>
        </binding>
   </webHttpBinding>
</bindings>
...
<serviceBehaviors>
        <behavior name="TDMoD">
          <dataContractSerializer maxItemsInObjectGraph="2147483647" />
```

```
          <!-- To avoid disclosing metadata information, set the value below to false and remove the metadata
  endpoint above before deployment -->
          <serviceMetadata httpsGetEnabled="true" />
          <!-- To receive exception details in faults for debugging purposes, set the value below to true.
  Set to false before deployment to avoid disclosing exception information -->
          <serviceDebug includeExceptionDetailInFaults="true" />
        </behavior>
 </serviceBehaviors>
 ...
 <services>
       <service name="GTWCF.GTService" behaviorConfiguration="TDMoD">
         <endpoint address="GTService" binding="basicHttpBinding" contract="GTWCF.GT_I"
  behaviorConfiguration="TDMoDFaultBehavior">
         </endpoint>
         <endpoint address="TDoDREST" binding="webHttpBinding" contract="GTWCF.GT_I"
  behaviorConfiguration="webHttpBehavior">
         </endpoint>
         <endpoint address="" binding="webHttpBinding" contract="GTWCF.IPolicyRestriction"
  behaviorConfiguration="webHttpBehavior" />
         <host>
           <baseAddresses>
             <add baseAddress="https://*:8090/" />
           </baseAddresses>
         </host>
       </service>
 </services>
```

For more information about how to get the certificate thumbprint, see how to retrieve the thumbprint of a certificate (microsoft.com)

Access the CA TDM Portal

After you successfully install the CA TDM Portal, you can log in and access its functionality. You access the CA TDM Portal through a web browser.

Your security group membership determines what functionality you can access in the CA TDM Portal. For example, if you are a member of the Tester group, you see only the Self-Service Catalog and Submitted Requests options. If you are a member of the Admin group, you see the complete functionality, including Modeling, Generators, Self-Service Catalog, Submitted Requests, and Configuration. For more information about security groups, see Groups.

Follow these steps:

1. Do *one* of the following:
   - Open your CA TDM Portal service URL in your web browser. Contact your Test Data Engineer to obtain the URL.
   - Click **Start, All Programs, CA, CA Test Data Manager Portal, Launch CA Test Data Manager Portal**.
     The CA TDM Portal login page opens.
2. Enter login credentials in the **Username** and **Password** fields and click **Sign In**.
   Default: Username `administrator` , password `marmite` .
   The welcome page opens.
3. (After an upgrade only) Use the Refresh functionality of your browser to ensure that you are looking at the upgraded version of the web portal.
4. Verify your existing connection profiles. If you have created a connection profile in CA TDM Datamaker that is not compatible with the CA TDM Portal, recreate the profile in the Portal.
5. Navigate the user interface to perform appropriate tasks.

NOTE

You can access the repository through the CA TDM Portal using a database user. The CA TDM Portal does not support Windows authentication.

Enable Integrated Security for CA TDM Portal

By default, you specify a user and credentials for each connection. Enable Integrated Security to instead allow the respective logged on user to connect to data sources and targets, and the application GT repository. If the currently logged on user has permissions to access the Test Data Manager repository, or a target or source database, follow these steps here to enable connection as this AD user.

Enable Integrated Security Patch for TDM Portal

1. Complete the installation of CA TDM Portal.
2. Open the Windows **Services** control panel and stop the Test Data Manager Portal service. Wait for the Portal service to stop.
3. Right click the portal service and click **Properties**.
4. Click the **Log on** tab in the **Properties** dialog.
5. Select the **This account** option, and enter the domain account that you want to be used.
6. Enter the password for this user, and confirm the password. Click **OK**.
7. Open the Windows **Services** control panel and start the Test Data Manager Portal service. Wait for the Portal service to restart.
8. Verify that you can connect at one of these respective URLs, depending on whether you use HTTPS security:
   - http://localhost:8080
   - https://localhost:8443

Create a Connection Profile

1. Launch CA Datamaker, and log in.
2. Click **Create new profile** in the **Profiles** dialog.
3. Select **Other** when asked "What type of database do you want to connect to?"
4. Complete the profile dialog with your details.
5. Enable the option **No login required**, and leave the username and password blank.
6. Specify a default schema.
7. Click **Test** and **Save**.
   The new profile is visible and usable from CA Datamaker and from CA TDM Portal.

When you enable integrated security on database screen for server type SQL server, the TDM Portal service does not start automatically. Log on with an account that has Windows authentication to the SQL Server, and start the service manually.

(Optional) Use Integrated Security to Connect to the GTREP Repository

After you have enabled Integrated Security, you can also configure CA TDM Portal to use Integrated Security when connecting to the GTREP application repository.

1. Browse to the following directory In Windows
   `C:\Program Files\CA\CA Test Data Manager Portal\conf`
2. Open the file `application.properties` in a text editor.
3. Append `integratedSecurity=true` to the `datasource.url` property.
4. Comment out or delete the `datasource.username` and `datasource.password` fields.

```
 # Database
spring.datasource.url =jdbc:sqlserver://
localhost:1433;database=gtrep;integratedSecurity=true;
#spring.datasource.username=
#spring.datasource.password=
```

5. Open the Windows **Services** control panel.
6. Restart the Test Data Manager portal service.

NOTE

More Information:

- SQL Server Security Modes on (msdn.microsoft.com)

Enable Integrated Security for Repository Access in Datamaker

As a TDE, you want to be able to use integrated security when you connect to the Microsoft SQL Server repository in Datamaker. You have a set of Active Directory (AD) users that is allowed to log onto the Microsoft SQL Server repository. Microsoft SQL Server authentication to the repository is not allowed.

For example, you want to allow all AD users in the "`GT TDM Admin` " group to access the repository. The "`GT TDM Admin` " group is in the `MYCO` domain. Prepare the following information:

- The Microsoft SQL Server repository name. In the example, this is `gtreptest` .
- The group login is called `MYCO\GT TDM Admin` .

To configure the repository, replace the repository and group names in the following script with yours, and run the script in Microsoft SQL Server Management Studio:

```
USE gtreptest;

CREATE LOGIN "MYCO\GT TDM Admin" FROM WINDOWS WITH DEFAULT_DATABASE=[master];

CREATE USER [MYCO\GT TDM Admin] FOR LOGIN [MYCO\GT TDM Admin];

ALTER USER [MYCO\GT TDM Admin] WITH DEFAULT_SCHEMA=[dbo];

ALTER ROLE [db_owner] ADD MEMBER [MYCO\GT TDM Admin];



USE master;

GRANT VIEW SERVER STATE to "MYCO\GT TDM Admin"
```

NOTE

- Connect Datamaker to the Repository

Install CA Agile Requirements Designer

CA Agile Requirements Designer is a test case design and optimization tool that lets you do the following tasks:

- Design requirements using interactive flow charts
- Convert requirements to an optimized set of test cases
- Integrate with ALM tools to import requirements and export generated test cases
- Build your automation framework into test cases

This release of Test Data Manager supports CA Agile Requirements Designer 2.7.0. CA Agile Requirements Designer integrates with Test Data Manager to help you build test data into your test cases. It also plays a vital role in the definition of flows for test matching. When you purchase Test Data Manager, you also receive a license for CA Agile Requirements Designer. This license entitles you only to create Self-Service Catalog forms.

To download CA Agile Requirements Designer, access the CA Support Online Download Center and select **CA Agile Requirements Designer Workgroup - MULTI-PLATFORM**. For installation, licensing, and setup instructions, see the CA Agile Requirements Designer documentation.

NOTE

When CA ARD prompts you to connect to a CA TDM connection profile, configure CA ARD to use the service layer of CA TDM, GT Service:

`http://your_TDM_hostname_or_IP:8090/GTService`

Mainframe Installation and Upgrade

To integrate with the z/OS mainframe component of Test Data Manager, you must install additional product components. These components execute on the z/OS mainframe against the mainframe data sources to ensure processing occurs as close to the data source as possible.

This section covers both installation and upgrade. An upgrade requires you to install the new version of the z/OS mainframe component.

Document intended audience

- Mainframe Installation Audience covers the skill sets and knowledge required for installing or upgrading the mainframe components.

Pre-Requisites

- System Requirements for Mainframe Installation covers the pre-requirements for the z/OS mainframe component of Test Data Manager both on the mainframe and on the TDM server.

Mainframe tasks

- Install Mainframe Components (v5.4.*) covers the installation and naming standards

NOTE

More information:

- TDM Mainframe Toolkit Best Practices (PDF)

Mainframe Installation Audience

Readers of these installation instructions must have knowledge in the following areas:

- JCL
- TSO/ISPF
- z/OS environment and installing software in this environment
- Your organization IT environment, enterprise structure, and region structure

Consult with the following personnel, as required:

- DB2 administrator
- Systems programmer for z/OS and VTAM definitions
- Storage administrator for DASD allocations

System Requirements for Mainframe Installation

Verify the following prerequisites and system requirements before you install Test Data Manager mainframe components:

General

General system requirements for mainframe access on the Test Data Manager server are as follows:

- COBOL/LE runtime libraries
- Access to site JCL standards to customize delivered JCL to site requirements
- FTP access to the Mainframe to move both installation files and generated objects between the Mainframe and the distributed platforms where the Test Data Manager GUI runs
- User privileges to perform TSO RECEIVE on the XMIT files and allocate the required libraries in the chosen dataset High Level Qualifiers (HLQ)
- Permission to submit and manage tasks and jobs through SDSF panels or the equivalent, or to have permission to issue operator commands to purge (/P) or start (/S) tasks/jobs

z/OS DB2 Access

General requirements for integrating with a DB2 database are as follows:

- DB2 v10 or higher subsystem (with active WLM address space for standard ODBC / JDBC calls)
- ODBC connectivity to z/OS DB2
- JDBC Type 4 connectivity to z/OS DB2
- Access to run DB2 programs through TSO program IKJEFT01 or the equivalent, and DB2 utility program DSNTIAUL
- Installation user privileges to create tables, bind DBRMs, import data, and grant access to users and programs

Data storage requirements for integrating with a DB2 database are as follows:

- Data is stored as Single Byte Character Set (SBCS). This can be either EBCDIC or UNICODE (providing each character is stored in a single byte).
- Data stored in Double Byte Character Sets (DBCS), for example Graphic fields, is not supported.
- Data stored in Multi Byte Characters Sets (MBCS / UNICODE) that have characters stored in more than 1 byte, is not supported.
- Data stored in LOBs, CLOBs, BLOBs, are not natively supported. (Potential for unloading structured LOB data to a flat file for processing.)

DB2 Database Driver Installation

CA Technologies does not provide ODBC / JDBC drivers for connection to DB2. Install, license and configure the following drivers on the Test Data Manager server.

- DB2 Connect for connectivity to z/OS DB2 data from the Test Data ManagerGUI.
  Note: No license is required for the IBM Data Server Client Version 10.5 Fix Pack 5 client software to fetch data from DB2 LUW. However, an IBM DB2 Connect Enterprise Edition 25 Authorized User License is required for this client software to connect to DB2 on mainframe. The IBM part number for this Enterprise Edition license is D58FILL. The ODBC and .NET drivers provided with this licensed client software are used by Test Data Manager and by the TDoD Web Service, respectively.
- DB2 Type 4 JDBC for connectivity to z/OS DB2 from the GTSubset and FDM UIs.
- If you are using ODBC or the DSNless ODBC connection type, and your ODBC driver is version 10.5 or below, and Datamaker encounters problems connecting to DB2 on mainframe, we recommend you upgrade the ODBC driver to 11 or the latest version.

Access to VSAM, Sequential Files, and Other z/OS Data

Test Data Manager requires access to copybooks that describe the VSAM, PS files, IMS DL1 segments, or access to other z/OS data unloaded to file for processing.

IMS Database Access

The Test Data Manager Mainframe IMS Add On package includes an entitlement to a restricted version of CA File Master Plus for IMS 10.0. This version of CA File Master Plus for IMS lets test data engineers perform following actions:

- Browse IMS database content
- Filter IMS content and view or update a record layout
- Extract IMS database records to a sequential file
- Reload IMS database content from a sequential file
- Automate the IMS data extract and load process using batch jobs

Other CA File Master Plus for IMS activites are restricted, and use of the product entitlement is only allowed for test data management activities. However, if you already own the full version of CA File Master Plus for IMS, you do not need to install this restricted version. You can use an existing installation of CA File Master Plus for IMS for test data management activities as well.

See the following pages in the CA File Master Plus for IMS documentation for installation and configuration instructions:

- Installation: https://techdocs.broadcom.com/content/broadcom/techdocs/us/en/ca-mainframe-software/devops/ca-filemaster-plus/10-0/installing.html
- User Interface: https://techdocs.broadcom.com/content/broadcom/techdocs/us/en/ca-mainframe-software/devops/ca-filemaster-plus/10-0/using-ispf/ispf-user-interface-for-ca-file-master-plus-for-ims.html
- Browse IMS Database Content: https://techdocs.broadcom.com/content/broadcom/techdocs/us/en/ca-mainframe-software/devops/ca-filemaster-plus/10-0/using-ispf/ispf-user-interface-for-ca-file-master-plus-for-ims/browsing-ims-databases.html
- Filters: https://techdocs.broadcom.com/content/broadcom/techdocs/us/en/ca-mainframe-software/devops/ca-filemaster-plus/10-0/using-ispf/ispf-user-interface-for-ca-file-master-plus-for-ims/using-filters.html
- Extract IMS Content: https://techdocs.broadcom.com/content/broadcom/techdocs/us/en/ca-mainframe-software/devops/ca-filemaster-plus/10-0/using-ispf/ispf-user-interface-for-ca-file-master-plus-for-ims/using-utility-functions.html
- Reload Extract Files into an IMS Database: https://techdocs.broadcom.com/content/broadcom/techdocs/us/en/ca-mainframe-software/devops/ca-filemaster-plus/10-0/using-ispf/ispf-user-interface-for-ca-file-master-plus-for-ims/using-utility-functions/reload-utility.html
- Generate Extract File Record Layouts: https://techdocs.broadcom.com/content/broadcom/techdocs/us/en/ca-mainframe-software/devops/ca-filemaster-plus/10-0/using-ispf/ispf-user-interface-for-ca-file-master-plus-for-ims/using-utility-functions/extract-layout-utility.html
- Manage Record Layouts: https://techdocs.broadcom.com/content/broadcom/techdocs/us/en/ca-mainframe-software/devops/ca-filemaster-plus/10-0/using-ispf/ispf-user-interface-for-ca-file-master-plus-for-ims/working-with-record-layouts.html

You can also use a third-party product that provides the following functionality to access IMS data for CA TDM activities:

- Ability to subset IMS data on segment or field level, or both (semi-complex subset rules)
- Extract to a flat file structure
- Records in the flat file are defined by a valid copybook structure (Test Data Manager uses it to access the fields in the file)
- Ability to Load or Replace segments from the "masked" flat file into IMS
- Copies of DBD and PSBs for the segments being processed

You can use a third-party product, but CA does not provide any support for any third-party product for IMS data access.

Other Prerequisites and Considerations

Before installation, review the following requirements with the relevant sections of your organization:

- Mainframe High Level Qualifier (HLQ) to RECEIVE the XMIT files. The default is GRIDT01, and the JCL uses this HLQ. If the HLQ is changed to another name, amend the provided JCLs.
  Note: If you use HLQ, ensure that HLQ is defined within SMS ACS. The installation is received into its own distinct HLQ directories and does not affect system directories or files. You also require access to the HLQ file structure for the required IDs.
- Review JCL job names, message classes, and job classes for the JCL submissions such as masking and subsetting. Ensure that the relevant IDs have access to submit and manage the results. Also ensure access to issue operator commands to purge (/P) or start (/S) jobs and tasks using SDSF or the equivalent.
- Seed data can be held within either:
  – DB2 tables — This option is preferred for all sites with DB2 installed. It is required for masking and subsetting of DB2 data.

Before you install, determine the subsystem and schema name for the product reference tables. GRIDT01 is the default schema.

- VSAM files — This option is for clients with no DB2 installation and no requirement for masking or subset of DB2 data.

- The best roles to complete the noted tasks are DBAs or Systems, Programmer, or Operations Analyst. Plan and raise change control well before the installation date.
- Minimal disk space (<1 cylinder) is required for each library. A few megabytes are required for the VSAM or DB2 seed data.

NOTE

For assistance, contact product support.

Install Mainframe Components (v5.4.*)

This section contains documentation about Datamaker Mainframe installation. This article applies to all service packs of v5.4 of the Mainframe installers on support.broadcom.com. The installation procedure varies depending on your specific requirements and intended use.

- DB2 to hold the reference data - preferred option, and required for masking or subsetting of DB2 tables.
- VSAM KSDS files to hold the reference data - use this option where *no* DB2 installation is available to hold the reference data

NOTE

All reference data, JCL, parameters, executable load modules, and run components are supplied in TSO XMIT format data sets (LRECL=80, RECFM=FB, BLKSIZE=3120).

The **Ref Data** column on the XMI table identifies which XMIT files are required for which installation type. Pre-allocate the required data sets on the mainframe before you transfer the XMIT files to the mainframe. See Appendix A for a JCL example. Transfer the XMIT files to z/OS without character set conversion. For most sites, an FTP (Binary) transfer is satisfactory. For other sites, FTP options Mode B or Type E might be required depending on the architecture and setup.

Once the XMIT files are transferred to the mainframe, they are received into appropriately specified data sets. The source data set names that are used to create the XMIT files have a high-level qualifier of GRIDT01. You can follow any naming convention for the target data set names. If the names differ from the source data set names, edit the supplied JCL members to reflect the new names. For details of the TSO RECEIVE command, see the IBM TSO User Guide and the TSO Command reference. The following members contain JCL to define the data sets. These data sets are required for the installation and to execute the RECEIVE commands:

```
"GRIDT01.LIB.RUNJCL(RECEIVE)"(DB2 install)
"GRIDT01.LIB.RUNJCL(RECEIVEV)"(VSAM install)
```

Edit data set names in this member to appropriate values. Appendix B contains JCL to define and receive data into GRIDT01.LIB.RUNJCL.

After you supply the data sets in z/OS that are specific to your installation, complete the other installation and test steps. These steps are detailed in sections specific to each installation. If you have problems, including issues with the job output from any failing jobs, contact product support.

NOTE

- Appendix A - JCL to Allocate the XMIT Datasets
- Appendix B - JCL to Load GRIDT01.LIB.RUNJCL

Install DB2 Reference Data

Use the following steps to complete and validate your DB2 Reference Data installation:

1. Create DB2 Tables?
2. Load Seed Data
3. Bind DB2 Plans
4. Load Message Data
5. Validate the Installation

Create DB2 Tables

Define the following reference data tables before masking:

- GTSRC_REFERENCE_LOV1
  Note: This table requires a page size of at least 8 KB
- GTSRC_XREF
- GTSRC_SUBSET

NOTE

For details of these tables, see:

- PDS GRIDT01.LIB.SPUFI
  : Members use a qualifier of GRIDT01 in the table definitions. You might want to edit this qualifier to name a different schema.
- GTREF
- GTXREF
- GTSUBSET

Load Seed Data

Before you load DB2, define seed data table GTSRC_REFERENCE_LOV1 to DB2. For more information, see Create DB2 Reference Tables.

GRIDT01.LIB.RUNJCL(SEEDLOAD) contains a job to populate GTSRC_REFERENCE_LOV1 with seed data using the DB2 Load utility. This job reads GRIDT01.SEED.CARDS. These cards specify the target table to be loaded INTO GRIDT01.GTSRC_REFERENCE_LOV1. If you define GTSRC_REFERENCE_LOV1 in a schema other than GRIDT01, edit this line.

To change the job card, make the following changes:

- Change the PROCLIB data set name. This name is initially set to DSN810.PROCLIB.
- Change the SYSTEM parameter. This parameter in initially set to DB8G).

If you are not sure of the correct values to use for PROCLIB and SYSTEM, consult a DB2 DBA. The files that contain the seed data are input to the job against DD name SYSREC01. The data has been split between seven files.

The SEEDLOAD job completes with a condition code no higher than 4. When the job is complete, run the SQL provided in GRIDT01.LIB.SPUFI(SEEDLIST). This SQL shows the seedlists that are inserted, and the number of entries for each seedlist.

Bind DB2 Plans

The Bind JCL template is located in GRIDT01.LIB.RUNJCL(BIND). Amend the JOBLIB DD statements in this job as required for your z/OS environment. If you are not sure of the required DD statements, consult a DBA who is familiar with your environment.

The SYSTSIN parameters are located in GRIDT01.LIB.PARM, members BGTXDMP, BGTXMSK, BGTXMSKF, and BGTXMSKL. You can edit these members to suit your environment. You might want to change the following lines:

- DSN SYSTEM(DB8G)
- The lines that contain the QUALIFIER (GRIDT01)

The following programs are used to bind DB2 plans:

- Two DB2 masking programs
- Dump data from DB2 tables
- Flat file masking

Load Message Data

Edit the dataset names in GRIDT01.LIB.RUNJCL(MSGLOAD) as required. For example, if dataset GRIDT01.MSG.SOURCE is renamed when it is received from MsgData.xmi, then rename this dataset. Also rename GRIDT01.MSG.KS as required.

> **NOTE**
> If you rename GRIDT01.MSG.KSd, make corresponding changes to the JCL procedures in GRIDT01.LIB.PROCLIB.

Submit the job to complete with a condition code no higher than 4.

Validate the DB2 Installation

DB2 Masking

Follow these steps:

1. Run the SQL in GRIDT01.LIB.SPUFI(TEST) to create and populate table GT_TEST.
   Note: If you changed the schema that contains table GT_TEST: change the schema in which the table is defined. This schema is initially set to GRIDT01.
   a. Edit the line SCHEMA=GRIDT01 in job GRIDT01.LIB.RUNJCL(GTXMSK) to name your schema.
      Note: TARGETSCHEMA is an alias for SCHEMA.
   b. Edit the line SCHEMA=GRIDT01 in job GRIDT01.LIB.RUNJCL(GTXMSKL) to name your schema.
2. Submit the job in GRIDT01.LIB.RUNJCL(GTXMSK). This job should complete with a condition code no higher than 4.
3. Submit the job in GRIDT01.LIB.RUNJCL(GTXMSKL). This job should complete with a condition code no higher than 4.

File Masking

Follow these steps:

1. Run job GRIDT01.LIB.RUNJCL(GTXPRT). This job should complete with a condition code no higher than 4.
2. Run job GRIDT01.LIB.RUNJCL(GTXGEN). This job should complete with a condition code no higher than 4.
3. Run job GRIDT01.LIB.RUNJCL(GTXMSKF). This job should complete with a condition code no higher than 4.

Install VSAM Reference Data

Use the following VSAM reference data installation steps to complete and validate your installation:

Load Seed Data

Use the following information to find and populate a VSAM KSDS file to hold the lookup data.

Edit dataset names in GRIDT01.LIB.RUNJCL(KSDSSEED) and in VOLUME as required. This job creates and populates VSAM (GRIDT01.VSEED) with seed data from the dataset GRIDT01.SEED.DATA.

The following JCL parameters control the process that loads the seed data:

- **STEP03.PARMCD**
  Lists the available seed lists held in GRIDT01.SEED.DATA.
  Note: Seeds that are prefixed with "—" are loaded into the VSAM lookup KSDS. To select the seedlists to load, delete "—". These seedlists are available for lookup with the functions HASHLOV, RANDLOV, and SEQLOV.
- **STEP05.PARMCD**
  Add the seedlists to be available for lookup with the functions HASHLOV1, RANDLOV1, SEQLOV1.
  Note: Select these seedlists to load in STEP03.

Verify that the VOLUME information is correct for the allocation of the KSDS. Edit the information as needed.

The supplied JCL loads the US STATE ZIP CITY COUNTY seed data (STEP03 and STEP05). Because this data is used in the installation test, keep this data for the test.

Note: Run this step as required to refresh the seedlists that are available in the lookup VSAM.

Load Message Data

1. Edit the dataset names in GRIDT01.LIB.RUNJCL(MSGLOAD) as required. For example, if dataset GRIDT01.MSG.SOURCE is renamed when it was received from MsgData.xmi, rename this dataset.
2. Verify that the VOLUME information is correct for the allocation of the KSDS. Edit the information as needed

Also rename GRIDT01.MSG.KSDS as required. If this dataset is renamed, make corresponding changes to the JCL procedures in GRIDT01.LIB.PROCLIB and submit the job. The job should complete with a condition code no higher than 4.

Create XREF VSAM KSDS (Optional)

1. Edit dataset names in GRIDT01.LIB.RUNJCL(KSDSXREF) as required.
2. Verify that the VOLUME information is correct for the allocation of the KSDS.

This job creates an empty, usable VSAM (GRIDT01.VXREF) file to store and lookup XREF data masking.

Create Subset VSAM KSDS (Optional)

1. Edit the dataset names in GRIDT01.LIB.RUNJCL(KSDSSUB) as required.
2. Verify that the VOLUME information is correct for the allocation of the KSDS.

This job creates an empty, useable VSAM (GRIDT01.VSUBSET) file to store and lookup Subset data.

Validate the Installation

Flat File Masking:

1. Run job GRIDT01.LIB.RUNJCL(GTXPRT). This job should complete with a condition code no higher than 4.
2. Run job GRIDT01.LIB.RUNJCL(GTXGEN). This job should complete with a condition code no higher than 4.
3. Run job GRIDT01.LIB.RUNJCL(GTXMSKVS). This job should complete with a condition code no higher than 4

This test uses seedlist US STATE ZIP CITY COUNTY. If this list is not loaded into the VSAM KSDS seedlist, the test fails with return code 8.

XMI Files

In the following table, the Ref Data column identifies which XMI files are required for DB2 / VSAM installation. The Member # column is only populated for PDS datasets:

XMI Name	Source DSN	Ref Data	Member #	Approx space (KB)	Format
libdbrm	GRIDT01.LIB.DBRM	DB2	14	72	RECFM=FB,LRECL=80
libdef	GRIDT01.LIB.DEFCSV	DB2 / VSAM	2	9	RECFM=FB,LRECL=120
libload	GRIDT01.LOADLIB	DB2 / VSAM	14	3,439	RECFM=U,LRECL=80
libmap	GRIDT01.LIB.MAPCSV	DB2 / VSAM	2	17	RECFM=FB,LRECL=255
libparm	GRIDT01.LIB.PARM	DB2 / VSAM	7	21	RECFM=FB,LRECL=80
libproc	GRIDT01.LIB.PROCLIB	DB2 / VSAM	11	78	RECFM=FB,LRECL=80
libjcl	GRIDT01.LIB.RUNJCL	DB2 / VSAM	19	104	RECFM=FB,LRECL=80
libspufi	GRIDT01.LIB.SPUFI	DB2	9	18	RECFM=FB,LRECL=80
msgdata	GRIDT01.MSG.SOURCE	DB2 / VSAM		112	RECFM=FB,LRECL=140
seedcard	GRIDT01.LOAD.CARDS	DB2		6	RECFM=FB,LRECL=120
seeddat1	GRIDT01.LOAD.SEED1	DB2		133,000	RECFM=FB,LRECL=2329
seeddat2	GRIDT01.LOAD.SEED2	DB2		199,000	RECFM=FB,LRECL=2329
seeddat3	GRIDT01.LOAD.SEED3	DB2		125,000	RECFM=FB,LRECL=2329
seeddat4	GRIDT01.LOAD.SEED4	DB2		159,000	RECFM=FB,LRECL=2329
seeddat5	GRIDT01.LOAD.SEED5	DB2		161,000	RECFM=FB,LRECL=2329
seeddat6	GRIDT01.LOAD.SEED6	DB2		161,000	RECFM=FB,LRECL=2329
seeddat7	GRIDT01.LOAD.SEED7	DB2		252,000	RECFM=FB,LRECL=2329
seeddata	GRIDT01.SEED.DATA	VSAM		55,204	RECFM=VB,LRECL=16384
testdata	GRIDT01.TEMP.TEST	DB2 / VSAM		20	RECFM=VB,LRECL=258

GRIDT01 PDS/PDSE Packages for Mainframe Installation

The following list shows the GRIDT01 PDS contents:

GRIDT01.LIB.DBRM (DB2)

This PDSE contains member Names and descriptions for all modules that use DB2.

GRIDT01.LIB.DEFCSV (DB2 / VSAM)

This PDS contains flat file record definitions. The following definitions correspond to Datamaker record definition files that are suffixed DM.txt:

- **TEST**
  Flat file definition that corresponds to file GRIDT01.TEMP.TEST. Used to validate the installation for DB2 and VSAM installations
- **TEST1**
  Flat file definition that is used to validate the installation for DB2 and VSAM installations

GRIDT01.LOADLIB (DB2 / VSAM)

This PDSE contains executable programs.

GRIDT01.LIB.MAPCSV (DB2 / VSAM)

This PDS contains the following transformation mapping files:

- **TESTDB**
  Transformation mapping file for masking table GT_TEST. This file is used to validate the DB2 installation.
- **TESTF**
  Transformation mapping file for masking file GRIDT01.TEMP.TEST . This file is used to validate the DB2 and VSAM installation.

GRIDT01.LIB.PARM (DB2 / VSAM)

This PDS contains parameters to program. The supplied JCL references the following members:

- **BGTXDMP**
  Bind parameters for program GTXDMP (DB2 dump executable)
- **BGTXMSK**
  Bind parameters for program GTXMSK (DB2 masking executable)
- **BGTXMSKF**
  Bind parameters for program GTXMSKF (flat file masking executable DB2)
- **BGTXMSKL**
  Bind parameters for program GTXMSKL (DB2 unload and mask executable)
- **BGTXSHD1**
  Bind parameters for program GTXSHD1 (read shredded file data from DB2)
- **PROS1**
  Profiling job sort parameters
- **PROS2**
  Profiling job sort parameters
- **QUDBL**
  Parameter for CSVs with double quotes
- **QUSGL**

Parameter for CSV with single quotes
- **SEEDS1**
  Sort Card for Loading Seed data into VSAM KSDS
- **TEMPL**
  DDL Template
- **SHDPARM**
  PARMCD settings for program GTXSHD

GRIDT01.LIB.PROCLIB (DB2 / VSAM)

This PDS contains the following JCL procedures:

- **GTDMP**
  JCL proc used by GRIDT01.LIB.RUNJCL(GTXDMP)
- **GTGEN**
  JCL proc used by GRIDT01.LIB.RUNJCL(GTXGEN)
- **GTMSKDB**
  JCL proc used by GRIDT01.LIB.RUNJCL(GTXMSK)
- **GTMSKF**
  JCL proc used by GRIDT01.LIB.RUNJCL(GTXMSKF)
- **GTMSKFV**
  JCL proc used by GRIDT01.LIB.RUNJCL(GTXMSKV)
- **GTMSKVS**
  JCL proc used by GRIDT01.LIB.RUNJCL(GTXMSKVS)
- **GTMSKL**
  JCL proc used by GRIDT01.LIB.RUNJCL(GTXMSKL)
- **GTPRO**
  JCL proc used by GRIDT01.LIB.RUNJCL(GTXPRO1)
- **GTPRT**
  JCL proc used by GRIDT01.LIB.RUNJCL(GTXPRT)
- **GTSEEDV**
  JCL proc used by GRIDT01.LIB.RUNJCL(KSDSSEED)
- **GTSHD**
  JCL proc used by GRIDT01.LIB.RUNJCL(GTXSHD)
- **GTTMT**
  JCL proc used by GRIDT01.LIB.RUNJCL(GTXTMT)
- **GTUSHD**
  JCL proc used by GRIDT01.LIB.RUNJCL(GTXUSHD)

GRIDT01.LIB.RUNJCL (DB2 / VSAM)

PDS contains the following Member. The members are categorized by Ref Data:

DB2 Ref Data Members

- **BIND**
  Job to bind DB2 programs.
- **GTXDMP**
  Runs program GTXDMP to dump data from DB2 tables.
- **GTXMSK**

Runs program GTXMSK to mask DB2 tables. Seedlist data, cross-reference tables, and subsetting lists are held in DB2 tables.

- **GTXMSKF**
  Runs program GTXMSKF to mask flat files. Seedlist data, cross-reference tables, and subsetting lists are held in DB2 tables.
- **GTXMSKL**
  Runs program GTXMSKL to mask DB2 tables. Seedlist data, cross-reference tables, and subsetting lists are held in DB2 tables.
  Note: Masked data is written to flat files in a format suitable for to load into DB2 with the load utility. The tables being masked are not updated in place.
- **GTXMSKL2**
  Runs the load utility to load masked data into DB2.
- **GTXMSKV**
  Runs program GTXMSKF to mask VSAM files. Seedlist data, cross-reference tables, and subsetting lists are held in DB2 tables.
- **GTXSHD**
  Job to define DB2 tables for file shredding
- **GTXUSHD**
  Job to unshred file data from DB2 tables
- **GTXSHDL**
  Runs the load utility to load shredded file data into DB2
- **RECEIVE**
  Deletes and defines datasets for product installation, and runs the TSO RECEIVE commands to load the defined datasets from the supplied XMIT format files.
- **SEEDLOAD**
  Runs the DB2 load utility to populate GTSRC_REFERENCE_LOV1 with seed data.

VSAM Ref Data Members

- **GTXMSKVS**
  Runs program GTXMSKF to mask flat files, Seedlist data, cross-reference tables and subsetting lists are held in VSAM KSDS file.
- **RECEIVEV**
  Delete and defines datasets for product installatio,n and runs the TSO RECEIVE commands to load the defined datasets from the supplied XMIT format files.

DB2/VSAM Ref Data Members

- **GTXGEN**
  Runs program GTXGEN to transform flat files from one format to another.
- **GTXPRO**
  Runs programs GTXPRO1 and GTXPRO2 to produce profiling data for a flat file.
- **GTXPRT**
  Runs program GTXPRT1 to print the contents of flat files.
  Note: File layouts are described by a file definition CSV.
- **GTXTMT**
  Runs program GTXTMT to extract Test Mart data from flat files.
- **KSDSSEED**
  Runs IDCAMS to define and populate VSAM KSDS that holds Seed data that is used by product programs.
- **KSDSSUB**

Runs IDCAMS to define and populate VSAM KSDS holding Subset data that is used by product programs.

- **KSDSXREF**
  Runs IDCAMS to define and populate a VSAM KSDS holding Xref data that is used by product programs.
- **MSGLOAD**
  Runs IDCAMS to define and populate a VSAM KSDS holding message data that is used by the product programs.

GRIDT01.LIB.SPUFI (DB2)

This PDS contains SQL to define DB2 tables and query tables:

- **GTREF**
  Table definition for the seed list table GTSRC_REFERENCE_LOV1.
- **GTSUBSET**
  Table definition for the flat file subset driving table GTSRC_SUBSET.
- **GTXREF**
  Table definition for the cross-reference table GTSRC_XREF.
- **SEEDLIST**
  Query to list all the seed lists that are loaded into table GTSRC_REFERENCE_LOV1.
- **SEEDROW**
  Query to show data in a GTSRC_REFERENCE_LOV1 row.
- **TEST**
  Create and populate the table GT_TEST.
- **XREFDATA**
  Query to show data in a GTSRC_XREF row.

GRIDT01.LOAD.* (DB2)

These datasets contain seedlist data. For more information, see Populate Seed Data.

GRIDT01.TEMP.TEST (DB2 / VSAM)

Used to validate the installation.

Appendix A - JCL to Allocate the XMIT Datasets

```
//*-------------------------------------------------------------------

//* TSO ALLOCATE JCL

//*     EDIT SRCHLQ TO THE HIGH LEVEL QUALIFIER OF THE XMIT DATASETS

//*-------------------------------------------------------------------

//DEF1    EXEC PGM=IEFBR14

//DD01    DD DISP=(NEW,CATLG,CATLG),

//        DSN=SRCHLQ.GRIDT01.LIBJCL.XMI,
```

```
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(5,2))
//DD02     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.LIBPARM.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(5,2))
//DD03     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.LIBDBRM.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(5,2))
//DD04     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.LIBDEF.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(5,2))
//DD05     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.LIBLOAD.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(5,2))
//DD06     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.LIBMAP.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(5,2))
//DD07     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.LIBPROC.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
```

```
//         SPACE=(CYL,(5,2))
//DD08     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.LIBSPUFI.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(5,2))
//DD09     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.MSGDATA.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(5,2))
//DD10     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.SEEDCARD.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(5,2))
//DD11     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.SEEDDAT1.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(50,35))
//DD12     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.SEEDDAT2.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(50,35))
//DD13     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.SEEDDAT3.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
```

```
//         SPACE=(CYL,(50,35))
//DD14     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.SEEDDAT4.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(50,35))
//DD15     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.SEEDDAT5.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(50,35))
//DD16     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.SEEDDAT6.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(50,35))
//DD17     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.SEEDDAT7.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(50,35))
//DD18     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.TESTDATA.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(5,2))
//DD19     DD DISP=(NEW,CATLG,CATLG),
//         DSN=SRCHLQ.GRIDT01.SEED.DATA.XMI,
//         DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120,DSORG=PS),
//         SPACE=(CYL,(50,25))
```

```
/*
//
```

Appendix B - JCL to Load GRIDT01.LIB.RUNJCL

```
//*-------------------------------------------------------------------
//* TSO RECEIVE JCL
//*     EDIT SRCHLQ TO THE HIGH LEVEL QUALIFIER OF THE XMIT DATASETS
//*     EDIT TGTHLQ TO THE HIGH LEVEL QUALIFIER OF THE TARGET DATASETS
//*-------------------------------------------------------------------
//DEL1    EXEC PGM=IEFBR14
//DD01    DD DISP=(MOD,DELETE),DSN=TGTHLQ.GRIDT01.LIB.RUNJCL,
//        SPACE=(TRK,0)
//* ----------------------------------------------
//DEF1    EXEC PGM=IEFBR14
//DD01    DD DISP=(NEW,CATLG,CATLG),DSN=TGTHLQ.GRIDT01.LIB.RUNJCL,
//        DCB=(LRECL=80,RECFM=FB,BLKSIZE=27920),
//        SPACE=(TRK,(5,5,5))
//*-------------------------------------------------------------------
//STEP07 EXEC PGM=IKJEFT01,REGION=512K
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
 RECEIVE INDATASET('SRCHLQ.GRIDT01.LIBJCL.XMI')
 DA ('TGTHLQ.GRIDT01.LIB.RUNJCL')
/*
```

```
//
```

Upgrade From v5.4.* to 5.4.9 or later

This article applies to all service packs of v5.4 of the Mainframe installers on support.broadcom.com. Most changes are to the various masking and subset programs. Therefore for an upgrade it is typically valid to update only the following two libraries:

- HLQ.LIB.DBRM
- HLQ.LOADLIB

HLQ is the High Level Qualifier where you have installed your CA TDM mainframe product.

> **NOTE**
>
> These two libraries HLQ.LIB.DBRM and HLQ.LOADLIB have changed from PDS to PDSE, as part of the CA Endevor process. The remaining libraries are still PDS for the time being.

To perform the update:

1. Back up the existing libraries, for example, by renaming them.
   - HLQ.LIB.DBRM
   - HLQ.LOADLIB
2. Allocate the XMI file (HLQ.XMI) with DCB properties:
   (RECFM=FB, LRECL=80, BLKSIZE=3120, SPACE>10 CYL)
3. Allocate the HLQ.LOADLIB PDSE with DCB properties:
   (DSNTYPE=LIBRARY RECFM=U, LRECL=0, BLKSIZE=27998, SPACE>250 BLOCKS)
4. Allocate the HLQ.LIB.DBRM PDSE with DCB properties:
   (DSNTYPE=LIBRARY RECFM=FB, LRECL=80, BLKSIZE=32720, SPACE>7 BLOCKS)
5. Binary FTP the libload.xmi file into the HLQ.XMI file.
6. Receive the loadlib file by:
   ```
   TSO Option 6:
   RECEIVE INDSN('HLQ.XMI')
   DA ('HLQ.LOADLIB')
   ```
7. Binary FTP the libdbrm.xmi file into the HLQ.XMI file:
8. Receive the dbrm file by:
   ```
   TSO Option 6:
   RECEIVE INDSN('HLQ.XMI')
   DA ('HLQ.LIB.DBRM')
   ```
9. Bind the new versions of the programs by executing JCL HLQ.LIB.RUNJCL(BIND).
10. Re-load the messages VSAM file by executing HLQ.LIB.RUNJCL(MSGLOAD).
11. Test the programs.

Upgrade Product Components

This section provides an overview on how to upgrade an existing Test Data Manager installation. The following upgrade paths are supported:

- 4.0 and 4.1 to 4.2
- 4.1 and 4.2 to 4.3
- 4.2 and 4.3 to 4.4
- 4.3 and 4.4 to 4.5
- 4.4 and 4.5 to 4.6
- 4.5 and 4.6 to 4.7

  NOTE

  For upgrades to versions earlier than v4.8, please refer to this page in the specific version you need.
- 4.6 and 4.7 to 4.8
- 4.7 and 4.8.1 to 4.9

  WARNING

  Follow a supported upgrade path to avoid repository and licensing compatibility issues.

Change to upgrade procedure with version 4.8

Test Data Manager 4.8 is the first version in which you **cannot** upgrade the **gtrep** repository from within the Datamaker software. It is now necessary to upgrade the TDM Portal in order to upgrade **gtrep**.

To upgrade Test Data Manager Portal, follow the instructions to Upgrade TDM Portal in Windows.

Download Media

Download the media from Broadcom Support. Select 4.9 and download the required files as per your environment requirement:

- CA Test Data Manager Full Package – For both DataMaker and TDM portal upgrade on Windows.
- CA Test Data Manager Portal for Docker – TDM portal on docker

Upgrade Options

The following 2 installers are available:

- **TDM Portal Installer**
  This installer upgrades the TDM Portal software to the latest version (4.8). See Upgrade TDM Portal in Windows for more information.
  The first time you start the TDM Portal service after installation, the service also upgrades the **gtrep** repository. See Upgrade gtrep manually for information on an alternative method.

  WARNING

  If you upgrade **gtrep**, GTServer applications from a previous version **do not function** with the upgraded gtrep repository! For all TDM applications, the **gtrep** repository version must match the application version.
- **GTServer Installer**
  This installer upgrades the non-Portal components of TDM (Datamaker, GT Subset, etc). This installer **does not** upgrade gtrep.

  WARNING

  If you run this upgrade **before** the TDM Portal service upgrades gtrep, the upgraded GTServer applications **do not function**! For all TDM applications, the **gtrep** repository version must match the application version.

Backup gtrep repository

We recommend that you backup your **gtrep** repository before you upgrade the CA TDM software.

This process varies depending on your database. Check with your Database Administrator for more information on how to proceed.

Upgrade the gtrep repository manually

From CA TDM 4.8, the TDM Portal service can manage the **gtrep** repository upgrade automatically.

> **TIP**
>
> To disable the TDM Portal service's automatic upgrade of the **gtrep** repository, uncheck the **Start CA Test Data Manager Portal service** option in the TDM Portal installer.

When you first run the TDM Portal service after installation, the service upgrades the **gtrep** repository automatically if the repository is not for the correct version, or installs **gtrep** if it is not present.

If you prefer to use the **Schema Management** tools provided to upgrade this repository yourself, refer to this guide.

Follow these steps:

1. Ensure that the TDM Portal service is not active.
   You can check which services are active from the Windows **Services** dialog (click Start, Services).
2. (Optional) To examine the files that the upgrade tool executes:
   a. Navigate to `C:\Program Files\CA\CA Test Data Manager Portal\schema-management\lib\` (in a default installation).
   This directory contains the compressed file `TDMGtrepSchema-version.jar` .
   b. Unzip `TDMGtrepSchema-version.jar` to a temporary location.
   This compressed file contains the directory `/db/migration/gtrep/[oracle/SQLserver]` (oracle/SQLserver depends on which kind of database you use for **gtrep**).
   This directory contains the individual files, that the `schema-management.bat` file executes.
3. To execute the **gtrep** upgrade:
   a. Navigate to `C:\Program Files\CA\CA Test Data Manager Portal\schema-management\bin\` (in a default installation).
   This directory contains the file `schema-management.bat` . You can use this batch file to migrate your existing gtrep entries to a new database.
   b. You can execute `schema-management.bat` with the following arguments:
      - **-m**
        Migrate the data from your existing **gtrep** repository to the upgraded repository.
      - **-i**
        Shows what has been applied, and what will be applied by migration.
      - ***<none>***
        Without any argument, execution of `schema-management.bat` displays help for the user.

Schema Management tool logs

The Schema Management tool logs changes to **gtrep**. These logs are located at the following locations:

- When the tool executes **at TDM Portal startup**:
  `C:\ProgramData\CA\CA Test Data Manager Portal\logs\TDMSchemaManagementStartup.log`
- When you execute the tool **manually**:
  `C:\ProgramData\CA\CA Test Data Manager Portal\logs\TDMSchemaManagement.log`

Upgrade product components with the GTServer Installer

Prepare for Upgrade

Before you upgrade TDM components, we recommend that you backup your **gtrep** repository. We also recommend that you backup configurations for each TDM component.

Back up all enabled Test Data Manager configurations as follows, using the TDoD Config Editor as an example:

1. Navigate to TDM_HOME\Grid-Tools\TDoD\TDoD_Config Editor and run TDoDConfigEditor.exe.
2. On the Configure Server tab, click Backup.
   The config editor creates a backup of the configuration file, and a confirmation message appears.
3. Back up other enabled config editors using the same process.

Other config editors that support backup are:

- Remote Publish Engine
- Rally Batch Service
- HP ALM Service
- Group Job Process Executor
- HP ALM Batch Configuration Service

Perform the Upgrade

To perform the upgrade, it is necessary to download the software and install it on all systems where Test Data Manager components exist. If you have Test Data Manager components that are installed on multiple systems, you must run the upgrade on each system.

1. Download the latest release from the Download Center on Broadcom Support.
2. Extract the downloaded zip file on the systems where you want to perform the upgrade.
3. Run setup_GTServer_*version*.exe:
   - The GT Server installer detects the existence of all prerequisites and clears all prerequisite check boxes.
   - Select all Test Data Manager components that exist on the current system.
   - The GT Server installer upgrades each component by running the different component installers. For components that store configuration settings, the installer automatically takes a backup of your current settings and restores those settings after the upgrade.
4. Repeat Step 3 on all systems that contain Test Data Manager components.

The GT Server installer creates installation logs in the Temp folder (%TEMP%). You can find the log files during both fresh installation and upgrade cases. A typical log file name has the following format:

<componentname_version.log>

For example, the GT HP ALM Service version 1.2.3.4 creates a file named `GT HP ALM Service_1.2.3.4.log` in the Temp folder during installation.

Upgrade Test Data Manager Portal

The upgrade process for TDM Portal is separate to the process to install the other TDM components (Datamaker, Fast Data Masker, GT Subset etc).

From TDM version 4.7, TDM Portal is also available as a Docker image. For this reason, there are are 2 possible ways to upgrade TDM Portal:

Upgrade TDM Portal in Docker

From Test Data Manager 4.7, TDM Portal is available as a collection of Docker images. You can use TDM Portal for Docker as stand-alone software, or you can use TDM Portal for Docker with the individual TDM components (Datamaker, Fast Data Masker, GT Subset etc).

To upgrade to a new version of TDM Portal for Docker, it is only necessary to download the Docker images for that version, and use the **docker-compose.yml** file appropriate to that version.

WARNING

To use a full installation of Test Data Manager and an instance of TDM Portal in Docker with the same data, it is necessary that both versions refer to the same **gtrep** repository.

Notes on Upgrade of gtrep

- From TDM 4.7, the TDM Portal upgrade procedure upgrades the **gtrep** repository. In previous versions, Datamaker manages the gtrep upgrade process. For more information, see Upgrade Product Components.
- From TDM 4.8, the first time TDM Portal (in Windows or Docker) runs, it upgrades the **gtrep** repository if gtrep already exists, or installs gtrep if it does not already exist.
- From TDM 4.8, Datamaker no longer upgrades **gtrep**.

WARNING

If you upgrade the **gtrep** repository, you must upgrade all other product components to the same version as the software that upgraded gtrep.

Upgrade paths for TDM Portal in Docker

From Test Data Manager 4.8, there are two possible paths to upgrade TDM Portal in Docker.

- From TDM Portal in Windows to TDM Portal in Docker.

  NOTE

  You can still upgrade TDM Portal in Windows to v4.8.
- From version 4.7 of TDM Portal in Docker, to version 4.8 of TDM Portal in Docker

Upgrade from TDM Portal in Windows to TDM Portal in Docker.

This upgrade path applies if you currently use TDM Portal in Windows (either with or without the individual Test Data Manager product components).

Follow these steps:

1. (Optional) **Upgrade Test Data Manager product components (Datamaker, GT Subset etc) to the target version.**This applies to users who use TDM Portal in conjunction with TDM product components that the GT Server installer installs. The version of TDM to which you upgrade, **must** correspond with the version of TDM Portal for Docker that you want to run.

   WARNING

   The first execution of TDM Portal upgrades the **gtrep** repository. The TDM components will not function, until you upgrade the repository.

   For more information on the upgrade process, see Upgrade Product Components.

2. **Migrate OrientDB databases from TDM Portal for Windows**It is necessary to transfer the databases that OrientDB uses, from their location in a Windows installation of TDM Portal (`C:\programdata\CA\CA Test Data Manager Portal\orientdb\databases` by default), to the OrientDB container.

   NOTE

   TDM Portal uses the following OrientDB Databases:

   - ReservationDB
   - ModelingDB
   - StagingDB

   Not all of these Databases may exist on your system. The creation of these databases is dependent upon the usage of certain features.

   Follow these steps:

   a. Open a command prompt (cmd.exe) and navigate to the `\orientdb\bin` directory in your TDM Portal installation folder (`C:\Program Files\CA\CA Test Data Manager Portal\orientdb\bin` by default).
   b. Start the OrientDB console, with the command `orientdb.bat` .

      TIP

      It is necessary to expose port **2424** of the OrientDB container, to connect to it from the OrientDB console. To do this, add the following line to your docker-compose file:

      ```
      ports:
        - '2424:2424'
      ```

   c. For each of the databases that are present on your system, perform the following steps:
      a. **Export the OrientDB database to the machine's local hard drive.**
         a. To connect to the local OrientDB databases, type the following in the OrientDB console:
            `connect remote:localhost/<DatabaseName> root <rootpassword>`
            For example:
            `connect remote:localhost/ReservationDB root myrootpwd`
         b. To export a database to a location where you have write access, type the following in the OrientDB console:
            `export database <local-path>`
            For example:
            `export database c:/users/username/ReservationDB`
            This command exports the contents of the database to the file `c:/users/username/ReservationDB.json.gz`
      b. **Import the exported databases to the OrientDB Docker container.**
         a. To connect to each database on the OrientDB container, type the following in the OrientDB console:
            `connect remote:<dockerhost>/<DatabaseName> root <rootpassword>`
            For example:
            `connect remote:mydockerhost.com/ReservationDB root myrootpwd`
         b. To import each database you exported in step i, type the following in the OrientDB console:
            `import database <local-path-to-file.gz>`
            For example:
            `import database c:/users/username/ReservationDB.json.gz`
            This command imports the database to your OrientDB container. Repeat this process for all databases exported in step i.
            Your OrientDB databases are now available for your OrientDB Docker container to use.
3. **Download, setup and run the TDM Portal for Docker containers.**

You can download these images (or download Dockerfiles and source binaries to build these images yourself) from the files available from https://support.ca.com/. When you execute the docker-compose file that contains the TDMWeb container, TDM Portal upgrades the gtrep repository.
For more information, see Install TDM Portal for Docker.

Upgrade from version 4.7 of TDM Portal in Docker, to version 4.8 of TDM Portal in Docker

This upgrade path applies if you currently use TDM Portal in Docker 4.7 (either with or without the individual Test Data Manager product components), and you want to upgrade to TDM Portal in Docker 4.8. The following process backs up the volumes that the TDM Portal and TDM Portal OrientDB containers need to operate.

NOTE

This process requires the use of **scripts backup-volumes-4.7.sh** and **restore-volumes-from-4.7-backup.sh**. These are included in both the file packages that contain Docker images - for more information, see File Packages available.

Follow these steps:

1. **Identify TDM Portal containers on your Docker network.**
   Do this with the `docker ps` command.

   ```
   $ docker ps
   ```

   This command lists all containers active in docker.
   Sample response:

   ```
   CONTAINER ID        IMAGE                                          COMMAND
     CREATED             STATUS              PORTS
         NAMES
   9fc3226587b2        tdm.packages.ca.com/tdm/tdmweb:4.7.0.14    "/opt/tdm/bin/tdmweb…"
     9 seconds ago       Up 7 seconds        0.0.0.0:8080->8080/tcp, 0.0.0.0:8443->8443/
   tcp   upgrade_tdmweb_1
   0f1f4c0769bb        tdm.packages.ca.com/tdm/orientdb:2.2.33    "/opt/tdm/bin/orient…"
     10 seconds ago      Up 8 seconds        2424/tcp, 2480/tcp
         upgrade_orientdb_1
   ```

   You may have other TDM Portal containers active, but you do not need to backup or restore these containers.

   WARNING

   If you execute your **docker-compose.yml** file from within a directory, Docker containers have the prefix "<directoryName>_" (e.g. "upgrade_" in the sample response above).

   TIP

   You can see a list of all Docker volumes active on your Docker network, with the following command:

   ```
   $ docker volume ls
   ```

   If you used a **docker-compose*.yml** file from TDM Portal 4.7, and did not create named volumes, the response to the above command is something similar to this:

   ```
   DRIVER              VOLUME NAME
   local
    1ca35ea30f627c3641ae04dc213295182637b31cae2438756b5da919848b392e
   local
    2a301be43b1ba2a797f0bfdb788e0403c9d615580365cfb7ff9e16285ce7bc52
   ...                 ...
   ```

2. **Stop active TDM Portal Docker containers**
   Stop containers with the following command:
   ```
   $ docker-compose stop
   ```
   This stops all active containers in your Docker network.
   Sample response:
   ```
   Stopping upgrade_tdmweb_1 ... done
   Stopping upgrade_orientdb_1 ... done
   ```
3. **Backup your TDM Portal containers' volumes.**
   The bash script `backup-volumes-4.7.sh` is available for this purpose.
   Syntax:
   ```
   $ backup-volumes-4.7.sh <orientdb_container> <tdmweb_container> [file]
   ```
   Where
   - **<orientdb_container> / <tdmweb_container>**
     Specifies the name of each of these containers (as in the response to the `docker ps` command).
   - (Optional) **file**
     Defines the relative or absolute path to a file, in which to store the output. Default: *tdm-volumes-backup-4.7.tar.gz*

   Example:
   ```
   $ backup-volumes-4.7.sh upgrade_orientdb_1 upgrade_tdmweb_1
   ```
   This creates the archive **tdm-volumes-backup-4.7.tar.gz**. This archive contains all volumes for the OrientDB and TDMweb containers.
   Sample response:
   ```
   INFO: 'tdm-volumes-backup-4.7.tar.gz' archive created
   ```
4. **Create volumes on your Docker network**
   For the **restore-volumes-from-4.7-backup.sh** script to function, it is necessary to create the following volumes on your Docker network:
   - orientdb_backup
   - orientdb_config
   - orientdb_databases
   - tdmweb_logs
   - tdmweb_storage

   WARNING

   The volumes you create must have the exact names above, for the script in Step 5 to function.

   You can create each volume with the following command (other methods are also possible):
   ```
   $ docker volume create <volume_name>
   ```

   NOTE

   At this stage, you may wish to add other attributes to your Docker volumes, for example with the **--driver** and **--opt <option>** flags.

   You are now ready to restore your backups into these new volumes.
5. **Restore your TDM volumes**
   The script `restore-volumes-from-4.7-backup.sh` is available for this purpose.
   Syntax:
   ```
   $ restore-volumes-from-4.7-backup.sh <archive>
   ```
   Where
   - **archive**
     Specifies the name of the tar.gz file, created by the backup-volumes-4.7.sh script.

Example:

```
$ restore-volumes-from-4.7-backup.sh tdm-volumes-backup-4.7.tar.gz
```

This restores the volumes that you backed up in Step 3, to the volumes you created on your Docker network.

6. **Rename your new TDM volumes (if necessary**)
   If your container names from Step 1 include a prefix (e.g. "upgrade_"), it is necessary to rename the TDM volumes you created in Step 4, to include this prefix.
   This command removes the existing container `<old_volume>` , and recreates it with the new name `<new_volume>` :

   ```
   $ docker run --rm -it -v <old_volume>:/from -v <new_volume>:/to busybox ash -c "cd /
   from ; cp -av . /to"
   ```

7. **Modify your docker-compose file, to make the new volumes accessible to the TDM Portal and OrientDB containers**
   For each volume that you created (and renamed if necessary), add the attribute "`external: true` ".The end of **docker-compose.yml** should look like this:

   ```
   volumes:
     upgrade_orientdb_backup:
       external: true
     upgrade_orientdb_config:
       external: true
     upgrade_orientdb_databases:
       external: true
     upgrade_tdmweb_logs:
       external: true
     upgrade_tdmweb_storage:
       external: true
     upgrade_tdmweb_fdmconfig:
       external: true
   ```

   This prevents docker-compose from creating new volumes. Instead, it refers to your existing volumes.

You can now execute a TDM Portal 4.8 docker-compose.yml file, and TDMWeb and OrientDB services will use your data persisted from TDM Portal 4.7.

Upgrade TDM Portal in Windows

This section provides an overview on how to upgrade an existing CA Test Data Manager Portal installation. The TDM Portal installer verifies whether an existing version of TDM Portal is already available on the computer. If the version exists, it starts the upgrade process; otherwise, it starts the new installation.

The following upgrade paths are supported:

- 3.5 to 3.6
- 3.5 and 3.6 to 3.8
- 3.6 and 3.8 to 4.0
- 3.8 and 4.0 to 4.1
- 4.0 and 4.1 to 4.2
- 4.1 and 4.2 to 4.3
- 4.2 and 4.3 to 4.4
- 4.3 and 4.4 to 4.5
- 4.4 and 4.5 to 4.6
- 4.5 and 4.6 to 4.7

NOTE

For upgrades to versions earlier than v4.8, please refer to this page in the specific version you need.

- 4.6 and 4.7 to 4.8
- 4.7 and 4.8.1 to 4.9

Follow a supported upgrade path to avoid repository and licensing compatibility issues.

Note on Upgrading to version 4.9

From Test Data Manager version 4.8 on, the TDM Portal service also upgrades the **gtrep** repository.

It is no longer possible to upgrade **gtrep** through Datamaker.

NOTE

Due to the new upgrade procedure, it is now necessary to install TDM Portal for the purpose of upgrading **gtrep**, even if you do not use TDM Portal as part of your workflow.

Upgrade Test Data Manager Portal

The installation procedure for TDM Portal does not upgrade other TDM components. To upgrade other components, see Upgrade GTServer Components.

You can upgrade an existing installation of TDM Portal.

Follow these steps:

1. Download the TDM Portal installer 4.9 from https://support.broadcom.com/.
2. Double-click the `setup_CA Test Data Manager Portal<version>.exe` file.
   A welcome dialog opens.
3. Click **Next**.
   The **End User License Agreement** dialog opens.
4. Accept the license agreement and click **Next**.
5. Enter your support credentials and click **Next**.
   - **Username**
     Specifies the user allowed to access Support.
   - **Password**
     Specifies the password associated with the support user.
6. Review the upgrade information that is displayed on the dialog.
7. Click **Next**.
   A message appears, to request confirmation that the installer can stop the CA Test Data Manager service to proceed with the upgrade.
8. Click **Yes**.

A message appears, to request confirmation that the installer can stop the OrientDB service to proceed with the upgrade.

9. Click **Yes**.
   The **Ready to Upgrade** dialog opens.
10. Click **Upgrade**.
    A dialog opens and displays the upgrade status. When the upgrade process is complete, the **Start CA Test Data Manager Portal service** checkbox appears.

    WARNING

    If you check this option, the TDM Portal service upgrades the gtrep repository **immediately**. If you wish to upgrade this yourself manually, do not tick this checkbox, and instead follow the procedure to Upgrade gtrep manually.
11. Click **Finish** when the upgrade completes.
12. (Optional) If you chose to **Start CA Test Data Manager Portal service** (step 10), open the Windows **Services** dialog (Start, Services) and verify that the CA Test Data Manager Portal and OrientDB services are available and are running.
13. Open the TDM Portal and verify the version number. If it does not display the newly installed version number, clear the browser cache, restart the TDM portal service, and refresh the page.

You have upgraded your TDM Portal installation.

Best Practice considerations

Review the following considerations.

OrientDB uninstallation removes artifacts

If you uninstall the existing TDM Portal installation and install the latest version, the uninstall process removes the existing OrientDB database, and installs the one that comes with the version of TDM Portal you install. This can create issues, because the uninstall process deletes all OrientDB artifacts created in the previous TDM Portal installation and stored in the associated OrientDB database.

Therefore, we recommend that you simply **upgrade** the TDM Portal.

TIP

If you must uninstall the TDM Portal, ensure that you Backup OrientDB databases prior to uninstallation, so that you can preserve your work artifacts.

Active Directory user migration

When you upgrade from a previous TDM Portal release, all existing Active Directory users are automatically migrated to this release of the TDM Portal. You do not need to perform this task manually.

Uninstall Product Components

Uninstalling Test Data Manager is a manual process that requires you to remove each component individually.

1. From the Windows Start menu, go to the Control Panel and open the **Programs and Features** dialog in Windows Vista/7/8, or **Apps and Features** in Windows 10.
2. Right-click each Test Data Manager component and select Uninstall.

   NOTE

   We recommend you back up your OrientDB database before removal of CA Test Data Manager Portal. Failure to do so may result in loss of data from OrientDB database - see possible outcome on CA TDM Portal Troubleshooting page.

For more information, see OrientDB Backup and Restore.

3. (Optional) Remove the repository database from your database server.

Manage Certificates

CA TDM Portal provides a self-signed certificate that is preconfigured for use. You can manage the CA TDM Portal certificate in any of the following ways:

- Create your own self-signed certificate with a provided utility, encrypt the keystore password, update the properties file with the keystore location, encrypted password, and key alias.
- Obtain a certificate from a recognized Certificate Authority. Update the properties file with the keystore location, encrypted keystore password, and key alias.

Manage certificates involve the following procedures:

- Install the Predefined Certificate
- Create and Implement a Self-Signed Certificate
- Use a Certificate from a Third-Party Certificate Authority

Install the Predefined Certificate

If you access CA TDM Portal with a URL that uses the HTTPS protocol, the browser checks for a certificate issued by a Certificate Authority. If you are using the CA Technologies self-signed certificate when you launch the TDM Portal, the browser displays a warning that the certificate is not trusted.

Follow these steps:

1. Open a browser, enter the URL for the CA TDM Portal, and log in.
2. If a Security Alert appears, click View Certificate.
3. Click Install Certificate and click OK.
4. Finish the wizard.
   The next time you log in, no Security Alert is presented.

Troubleshooting

If you get a Mismatched Address error, the browser believes that the security certificate presented by this website was issued for a different website's address.

Follow these steps:

1. Click the error message in the browser.
2. Click **View certificate**.
3. In the **Certificate** window, click **Details**.
4. In the **Details** tab, click **Copy to file...** in the bottom right-hand corner.
5. In the **Certificate Export Wizard**, under **Export File Format**, enable the "DER encoded binary X.509 (.CER)" radio button. Click **Next**.

The certificate is exported.

1. Double-click the saved certificate file.
2. In the **General** tab, click **Install Certificate...**.
3. In the **Certificate Import Wizard** window, set the **Store Location** to **Local Machine**. Click **Next**.
4. In the **Certificate Import Wizard** under **Certificate Store:**
   a. Enable the **Place all certificates in the following store** radio button and click **Browse**.
   b. Choose the **Trusted Root Certificate Authorities** option and click **OK**.

Add the self-signed certificate on any end-user machine that needs to access the portal.

TIP

To browse the certificate store and verify that the certificate was successfully registered, you can use the Certificates Snap-In through the Microsoft Management Console. For more information, see https://docs.microsoft.com/en-us/dotnet/framework/wcf/feature-details/how-to-view-certificates-with-the-mmc-snap-in.

Create and Implement a Self-Signed Certificate

You can replace the self-signed certificate that comes with CA TDM Portal. The predefined certificate is configured in the application.properties file. When you create your own self-signed certificate, update this properties file and restart CA Test Data Manager Portal service.

Before you create your own certificate, plan values for the keystore path and key alias. You enter these values when you run the keytool and when you update the properties file.

You use the following files and utilities to implement your self-signed certificates:

- keytool utility

  NOTE
  For details about this Java utility, browse for keytool - Key and Certificate Management Tool.
- EncryptionUtil.bat
- application.properties file, specifically, the following three parameters:
  - **tdmweb.keystorePath=**
    Default: *Self-signed certificate key store path.* For example, *install_dir\conf\.keystore*.
  - **tdmweb.keystorePassword =**
    Default: {cry}7i6EOsWzUxSm+tnSov-7cbTZs2TE0uAuXRxl4G+cG6O5Wn3aM8gz.
    Run the EncryptionUtil.bat file, enter the keystore password. The batch program generates the encrypted password on the console, which you specify here as the new value.
  - **tdmweb.keyAlias =**
    Default: Test Data Manager

Follow these steps:

1. Using administrator credentials, log in to host where TDM Portal is installed.
2. Stop the CA Test Data Manager Portal service.
3. If you plan to reuse the current alias name for the key, remove this alias before continuing.
4. Run the following command to generate a key pair with the Java keytool. Specify your own values for aliasname and for keystore_name. If you do not enter a path for keystore, the current path is used.

   ```
   keytool -genkey -alias "aliasname" -keyalg RSA -keystore "keystore_path\.keystore"
   ```

   For example, accept the default keystore path and enter:

   ```
   keytool -genkey -alias "Test Data Manager" -keyalg RSA
   ```

   Prompt to enter and confirm a password for keystore appears.
5. Enter the same keystore password in response to both the prompts. (Remember this password for later entry into an encryption utility.)
6. Respond to prompts with the requested distinguished name information as follows:
   a. Enter your first and last name.
   b. Enter the name of your organizational unit.

c. Enter your organization name.
d. Enter the name of your city or locality.
e. Enter the name of your state or province.
f. Enter the two-letter country code for your organizational unit.

A confirmation of your entries appears in the format, Is CN=value, OU=value, O=value, L=value, ST=value, C=value correct?

7. Review the entries and if correct, enter yes. (If incorrect, enter no and respond to the prompts again.)
8. Prompt for the key password for aliasname appears. Press Enter to use the keystore password as the alias password. A new keystore is created in the current directory.
9. (Optional) Move this keystore to another path.
10. Encrypt the keystore password you entered in Step 5.
    a. Change directories to the *install_dir**service**bin* directory.
    b. Run EncryptionUtil.bat
    c. Enter the keystore password in response to the prompt.

    The utility encrypts the entered keystore password and displays the result on the console.
11. Back up the application.properties file. (*install_dir**conf**application.properties*)
12. Update the application.properties file as follows:
    a. For tdmweb.keystorePath=, enter the absolute path to the keystore, using "/" rather than "\", for example, C:/*keystore_path*/keystore.
    b. For tdmweb.keystorePassword=, copy and paste the encrypted keystore password generated in Step 9.
    c. For tdmweb.keyAlias=, enter the alias name specified in the keytool command in Step 4.
13. Start the CA Test Data Manager Portal service.

Use a Certificate from a Third-Party Certificate Authority

CA TDM Portal supports third-party security certificates for HTTPS web access. Use your own resources to obtain a trusted TLS certificate from the Certificate Authority of your choice.

The use of third-party security certificates requires the use of third-party tools. The set-up process also requires manual changes to the application.properties file available at **C:\Program Files\CA\CA Test Data Manager Portal\conf** by default. Before you begin, become familiar with the basic concepts of security certificates and keystores and the keytool utility provided with the Java JDK.

Implementing third-party security certificates requires updating values for three parameters in the application.properties file:

- "tdmweb.keystorePath"
  Default: The keystore path for the self-signed certificate. For example: install_dir/conf/.keystore
- "tdmweb.keystore.Password"
  Default: {cry}7i6EOsWzUxSm+tnSov-7cbTZs2TE0uAuXRxl4G+cG6O5Wn3aM8gz
- "tdmweb.keyAlias"
  Default: Test Data Manager.

NOTE
To use a key alias that duplicates an existing alias, remove the existing alias before adding a new instance.

Follow these steps:

1. Decide on a certificate password and obtain a security certificate from a Certification Authority.
2. Using the instructions provided by the Certification Authority, import the certificate into a keystore.
   Generally you use a command similar to keytool - import - alias *myalias* - file certfile - keystore "path_and_file_specification_for_keystore".
   Make sure that the private key of the obtained certificate is also available in the specified keystore.

3. For the keystore password, enter the certificate password decided earlier in Step 1.
4. Obtain an encrypted version of the keystore password.
   a. Navigate to *install_dir**service**bin*.
   b. Run the encryption utility and supply the password to encrypt as argument.
      `EncryptionUtil.bat -p passwordtoencrypt`
   c. Save the encrypted value returned for entry in the properties file.
5. Stop the CA Test Data Manager Portal service.
6. Back up and edit the application.properties file to add or update the following:
   a. tdmweb.keystorePath to the location of the keystore using the fully qualified path and file name for the keystore file.
   b. tdmweb.keystorePassword with the encrypted keystore password (do not surround encrypted password value with quotes)
   c. tdmweb.keyAlias to the alias used to reference the certificate in the keystore (myalias in the examples).
7. Start the CA Test Data Manager Portal service.

Deploy CA TDM in a Security Zone

You can install CA TDM in a security zone, where there are untrusted and semi-trusted segments. When you deploy Test Data Manager in a security zone, we recommend that you split the Web Server from the App Server layer. The installation and configuration of Apache HTTPd Server is not covered in this documentation.

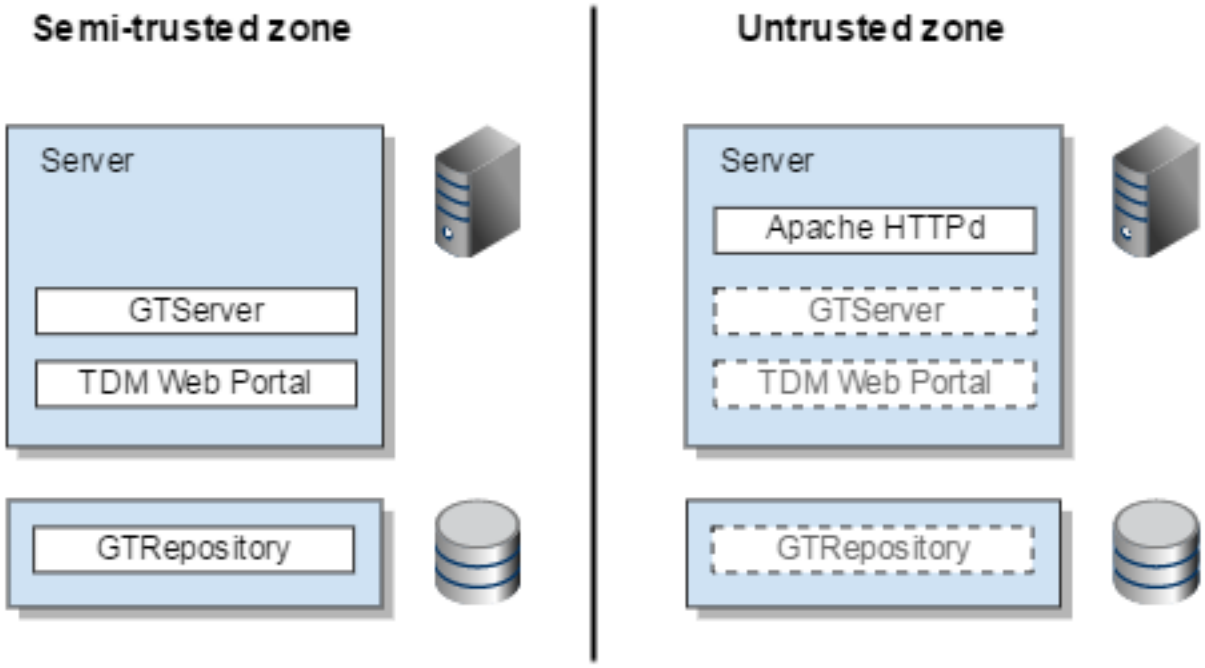
Architecture

This process assumes the existence of two Windows servers.

- The first server is in the untrusted zone, and serves static HTML and JS content for the Test Data Manager Web Portal application.
- The second server is in the semi-trusted zone, and serves the application APIs.

Connections between the trusted and untrusted zone go through a firewall, which is only open to the address of the untrusted zone server.

Figure 11: Install TDM components in the semi-trusted and untrusted zones



Install Software Components

1. Install the folllowing components in both the semi-trusted and untrusted zones.
   - CA TDM Repository
   - CA TDM GTServer
   - CA TDM Portal
2. Install the following component only in the untrusted zone.
   - Apache HTTPd server — including Visual Studio 2015 and latest Windows updates
3. Verify that DataMaker and CA Agile Requirements Designer are licensed on both servers.
4. Verify that the CA TDM Portal can be accessed on port 8080 of both servers.

In the untrusted zone, you require the GTServer, GT Repository, and WebPortal only during the installation process. After Installation, you can remove these applications from the untrusted zone.

1. Log on to the server in the untrusted zone.
2. Open the web apps directory.
   `C:\Program Files\CA\CA Test Data Manager Portal\tomcat\webapps`
3. Delete all the `TDM*.war` files, and folders whose names begin with `TDM`.
4. Keep the folder and WAR file for TestDataManager.
5. Stop the DBMS service.
6. Restart the CA Test Data Manager Portal service.

Configure Apache HTTPd for reverse proxy

On the untrusted server, you configure the Apache HTTPd server for reverse proxy by applying the following changes. Note: CA may extend this list of URL endpoints in the future as further CA TDM API services are made available.

1. Edit the Apache httpd config file.
   `C:\Apache24\conf\httpd.conf`
2. Uncomment the following lines in the apache httpd config file to enable reverse proxy:

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_connect_module modules/mod_proxy_connect.so
LoadModule proxy_http_module modules/mod_proxy_http.so
LoadModule rewrite_module modules/mod_rewrite.so
```

3. Add the following rules to the httpd.conf file to route the API traffic to the semi-trusted server:

```
ProxyRequests off


ProxyPass        /TestDataManager/api http://semi-trusted.domain.com:8080/
TestDataManager/api
ProxyPassReverse /TestDataManager/api http://semi-trusted.domain.com:8080/
TestDataManager/api



ProxyPass        /TestDataManager/user http://semi-trusted.domain.com:8080/
TestDataManager/user
ProxyPassReverse /TestDataManager/user http://semi-trusted.domain.com:8080/
TestDataManager/user



ProxyPass        /TestDataManager http://untrusted.domain.com:8080/TestDataManager
ProxyPassReverse /TestDataManager http://untrusted.domain.com:8080/TestDataManager



ProxyPass        /TDMConnectionProfileService http://semi-trusted.domain.com:8080/
TDMConnectionProfileService
ProxyPassReverse /TDMConnectionProfileService http://semi-trusted.domain.com:8080/
TDMConnectionProfileService



ProxyPass        /tdmJobEngineService http://semi-trusted.domain.com:8080/
tdmJobEngineService
ProxyPassReverse /tdmJobEngineService http://semi-trusted.domain.com:8080/
tdmJobEngineService
```

```
ProxyPass        /TDMProjectService http://semi-trusted.domain.com:8080/
TDMProjectService
ProxyPassReverse /TDMProjectService http://semi-trusted.domain.com:8080/
TDMProjectService


ProxyPass        /TDMService http://semi-trusted.domain.com:8080/TDMService
ProxyPassReverse /TDMService http://semi-trusted.domain.com:8080/TDMService


ProxyPass        /tdmwebModelingService http://semi-trusted.domain.com:8080/
tdmwebModelingService
ProxyPassReverse /tdmwebModelingService http://semi-trusted.domain.com:8080/
tdmwebModelingService
```

4. Substitute the hostname `semi-trusted.domain.com` in these rules with the hostname or address of your semi-trusted server.
5. Substitute the hostname `untrusted.domain.com` in these rules with the hostname or address of your untrusted server.
6. Save the configuration file and restart the Apache server.

Verify the Configuration

1. Open a browser to connect to the Apache server on port 80 of the untrusted host.
   The logon page displays.
2. Create a connection profile through the CA TDM Portal as a test.
3. Verify that this connection is created in the gtrep_profile table of the *semi-trusted* repository.
4. Verify that this connection is *not* created in the gtrep_profile table of the *untrusted* repository.

Remove API Services From the Untrusted Server

1. Open the Windows Control Panel, click Services, and stop the CA Test Data Manager Portal service.
2. Open the Windows File Explorer, and navigate to the Tomcat webapps folder at
   C:\Program Files\CA\CA Test Data Manager Portal\tomcat\webapps
3. Delete the following folders and WAR files:
   - TDMConnectionProfileService
   - TDMJobEngineService
   - TDMProjectService
   - TDMPublisherService
   - TDMService
   - TDMWebModelingService
4. Return to the Services control panel and restart the CA Test Data Manager Portal service.
   The API services are no longer accessible on the untrusted server.

Create rep.xml File to Store Repository Credentials

The rep.xml file lets you store repository credentials so you do not have to connect to the repository every time after launching Datamaker. You may want to create a new rep.xml file to allow this functionality or because you are experiencing error messages like 'System.Exception: rep.xml information not found in DM directory" and want to resolve them.

Prerequisites

1. Datamaker has been installed, licensed, and the appropriate repository profile connection has been created.
2. The TDoD service has been installed, configured, and currently running.
3. A user with administrative privileges is necessary to create and apply the rep.xml file.

Create rep.xml File

1. Launch the CA TDM Datamaker and login as an Administrator.
2. Click the 'Security' tab in the toolbar, and click 'Users and Groups' from the drop down list.
3. Provide your Administrator credentials in the CA TDM Datamaker Administrator Logon dialog.
4. In the Maintain Security window, click the 'XML' button in the top, right-hand corner.
5. Click OK in the Create XML Repository Profile dialog.
6. Go to C:\Users\<users>\AppData\Roaming\Grid-Tools (for example, C:\Users\Administrator\AppData\Roaming\Grid-Tools.
7. Copy the rep.xml file from the above folder to C:\Program Files (x86)\Grid-Tools\GTDatamaker.
8. Ensure that the rep.xml file is now available in both the above mentioned folders.
9. Re-start the CA TDM Datamaker.

Publishing Performance Example

The development team performed testing to show performance benchmarks for publishing to CSV and XLSX files. You can use these benchmarks to help tune the performance of your system.

Environment Details

The following environment was set up to gather performance data for publishing to CSV and XLSX files.

Machine specs

The machine has 16 GB physical memory, 4 vCPU, and runs Microsoft Windows Server 2012 R2 DataCenter. The repository was on a local SQL Server 2016.

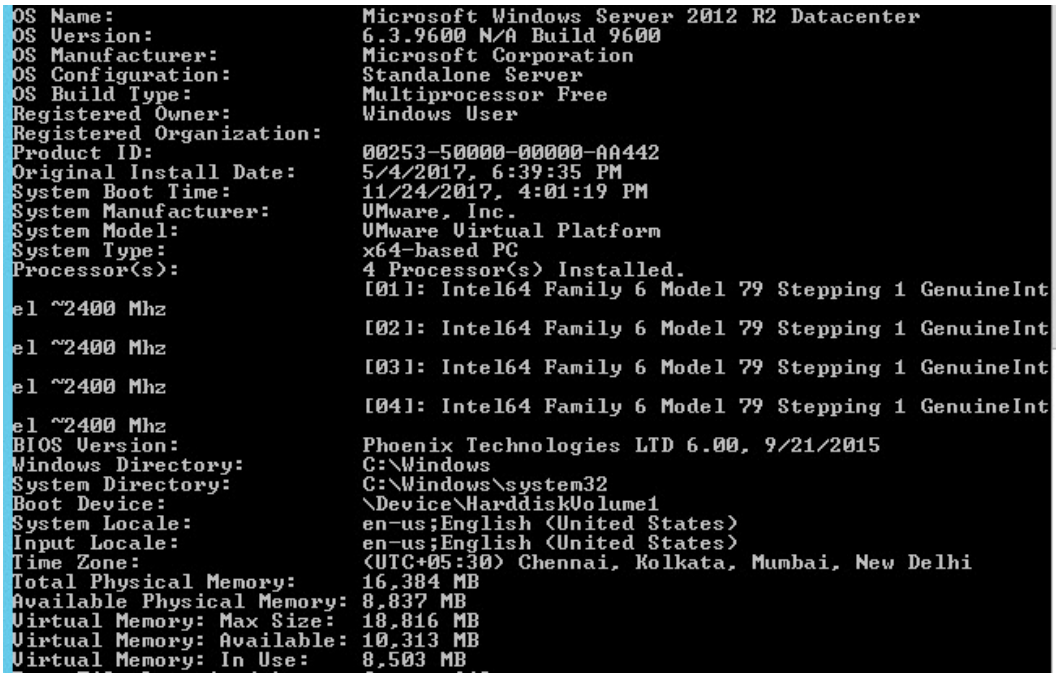


Table used for the performance publish

```
CREATE TABLE [dbo].[CARD_ACCOUNT](
    [CARD_ID] [int] NOT NULL,
    [CARD_BA_ID] [int] NOT NULL,
    [CARD_RA_ID] [int] NOT NULL,
    [CARD_BCH_ID] [int] NULL,
    [CARD_ICH_ID] [int] NULL,
    [CARD_NO] [varchar](16) NOT NULL,
    [CARD_EXP_DATE] [varchar](4) NOT NULL,
    [CARD_VALID_DATE] [varchar](4) NOT NULL,
    [CARD_NAME] [varchar](30) NOT NULL,
    [CARD_CVV] [decimal](4, 0) NOT NULL,
    [CARD_PRI] [varchar](3) NULL,
    [CARD_SUP] [varchar](3) NULL,
    [CARD_ADD] [varchar](3) NULL
)
```

Test 1 - Hard-coded Data

No expressions were used. All data were hard-coded in the generator.

Publish to CSV

repeater	performance based on number of rows
10,000	0.5 seconds
100,000	1.2 seconds
1,000,000	7.6 seconds
10,000,000	1 min 08 seconds

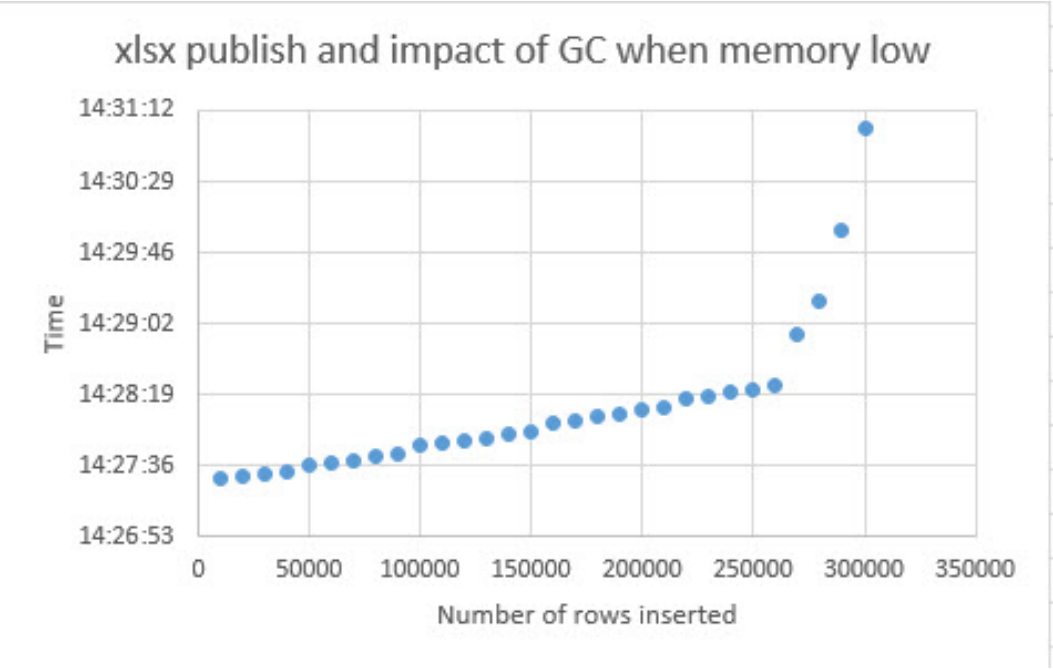
100,000,000	11 min 48 seconds

Publish to XLSX

Publish to XLSX is memory intensive compared to publish to CSV. Using the default CA TDM Portal configuration, the publish hit a wall around 300,000 counts when the CPU usage went high and stayed high. In fact, performance started degrading around 260,000 counts.

This behavior is caused by the GC (Garbage collector) going overdrive when trying to clean up some memory to make sure that the Portal application does not crash with an out-of-memory exception.

The following graph outlines the impact of garbage collection on the Portal:



TIP

Before starting a high-volume publish to XLSX, make sure you increase the memory that is allocated to the Portal.

Edit the YASJW config file called `wrapper.conf` located under `CA\CA Test Data Manager Portal \service\conf` . You can set either `maxmemory` or `maxmemory.percent` . With `maxmemory.percent` , the maximum allocated memory is calculated from the number that was set, times the physical memory.

repeater	performance based on number of rows
100,000	32 seconds
200,000	59 seconds
300,000	1 min 30 seconds
400,000	1 min 49 seconds
500,000	2 min 40 seconds
600,000	3 min 36 seconds

Test 2 - One Expression

We publish using one expression ~NEXT~ in the generator.

Publish to CSV

repeater	performance based on number of rows
1,000,000	14 seconds
10,000,000	1 min 29 seconds
100,000,000	12 min 38 seconds

Publish to XLSX

repeater	performance based on number of rows
100,000	24 seconds
200,000	46 seconds
300,000	1 min 18 seconds
400,000	1 min 47 seconds
500,000	2 min 17 seconds
600,000	2 min 41 seconds
700,000	3 min 16 seconds

SQL Server Target publish (default config)

repeater	performance based on number of rows
100,000	4 min 18 seconds
200,000	8 min 16 seconds
400,000	17 min 36 seconds
(portal restarted) 800,000	23 min 12 seconds

SQL Server Target publish (iterationsBeforeCommit=20000)

tdmweb.publish.batchCommit=true

tdmweb.publish.iterationsBeforeCommit=20000

repeater	performance based on number of rows
800,000	35 seconds
10,000,000	6 min 33 seconds
100,000,000	1 hour 6 min 46 seconds

SQL Server Target publish (iterationsBeforeCommit=50000)

tdmweb.publish.batchCommit=true

tdmweb.publish.iterationsBeforeCommit=**5 0000**

repeater	performance based on number of rows
1,000,000	42 seconds
10,000,000	6 min 31 seconds

Test 3 - With Expressions

We publish using several expressions.

Table DDL Used

```
CREATE TABLE equifax_records (
"update_period" numeric (38, 0) ,
"peer" varchar (20) ,
"state" varchar (20) ,
"county" varchar (20) ,
"product" varchar (20) ,
"vintage" varchar (20) ,
"originalrisk" varchar (20) ,
"currentrisk" varchar (20) ,
"term" varchar (20) ,
"smallbusinessownerflag" varchar (20) ,
"mortgageindicator" varchar (20) ,
"consumer_age" varchar(20) ,
"edti" varchar (20) ,
"pim" varchar (20) ,
"n_cur" numeric (38, 0) ,
"n_030" numeric (38, 0) ,
"n_060" numeric (38, 0) ,
"n_090" numeric (38, 0) ,
"n_120" numeric (38, 0) ,
"n_svr" numeric (38, 0) ,
"n_bkr" numeric (38, 0) ,
"n_misc" numeric (38, 0) ,
"n_closed_pos" numeric (38, 0) ,
"bal_cur" numeric (38, 0) ,
"bal_030" numeric (38, 0) ,
"bal_060" numeric (38, 0) ,
"bal_090" numeric (38, 0) ,
"bal_120" numeric (38, 0) ,
"bal_svr" numeric (38, 0) ,
"bal_misc" numeric (38, 0) ,
"bal_closed_pos" numeric (38, 0) ,
"pmt" numeric (38, 0) ,
"hc" numeric (38, 0) ,
"bal_bk" numeric (38, 0) ,
"n_fcs" numeric (38, 0) ,
"bal_fcs" numeric (38, 0) ,
"n_pos_bal" numeric (38, 0) );
```

Expressions Used

Table Name	Column Name	Definition	Data Type
EQUIFAX_RECORDS	n_pos_bal	@randlov(0,@perclist(90%@randrange(1,2)@,5%0,5%@randrange(3,10)@)@)@	NUMBER(38)
EQUIFAX_RECORDS	vintage	Q@randlov(0,@list(1,2,3,4)@)@@string(@randdate(2005/01/01,~YEAR~/01/01)@,YYYY)@	VARCHAR(20)
EQUIFAX_RECORDS	peer	@randlov(0,@list(GM,CM,CP,NC,OT)@)@	VARCHAR(20)
EQUIFAX_RECORDS	bal_closed_pos	@if(^bal_cur^=0,@randrange(0,60000)@,0)@	NUMBER(38)
EQUIFAX_RECORDS	n_bkr	@randlov(0,@perclist(1%1,99%0)@)@	NUMBER(38)
EQUIFAX_RECORDS	state	@seqlov(0,@seedlist(State_County,S)@,2)@	VARCHAR(20)
EQUIFAX_RECORDS	bal_120	@randlov(0,@perclist(99%0,1%@randrange(0,30000)@)@)@	NUMBER(38)
EQUIFAX_RECORDS	n_120	@randlov(0,@perclist(1%1,99%0)@)@	NUMBER(38)
EQUIFAX_RECORDS	edti	@randrange(0,9)@	VARCHAR(20)
EQUIFAX_RECORDS	n_svr	@randlov(0,@perclist(1%1,99%0)@)@	NUMBER(38)
EQUIFAX_RECORDS	bal_060	@randlov(0,@perclist(99%0,1%@randrange(0,60000)@)@)@	NUMBER(38)
EQUIFAX_RECORDS	hc	@if(^bal_cur^=0,0,@addrand(^bal_cur^,0,20000)@)@	NUMBER(38)
EQUIFAX_RECORDS	update_period	~YEAR~@randlov(0,@list(01,02,03,04,05,06,07,08,09,10,11,12)@)@	NUMBER(38)
EQUIFAX_RECORDS	originalrisk	@randrange(0,14)@	VARCHAR(20)
EQUIFAX_RECORDS	smallbusinessownerflag	@randlov(0,@perclist(15%1,85%~EMPTY~)@)@	VARCHAR(20)
EQUIFAX_RECORDS	n_misc	@randlov(0,@perclist(1%1,98%0,1%-1)@)@	NUMBER(38)
EQUIFAX_RECORDS	bal_030	@randlov(0,@perclist(97%0,3%@randrange(0,90000)@)@)@	NUMBER(38)
EQUIFAX_RECORDS	bal_cur	@randlov(0,@perclist(20%@randrange(100000,900000)@,40%@randrange(0,9000)@,35%@randrange(10000,90000)@,5%@randrange(1000000,2000000)@)@)@	NUMBER(38)
EQUIFAX_RECORDS	bal_fcs	0	NUMBER(38)
EQUIFAX_RECORDS	bal_misc	@randlov(0,@perclist(99%0,1%@randrange(0,20000)@)@)@	NUMBER(38)
EQUIFAX_RECORDS	n_060	@randlov(0,@perclist(25%1,70%0,5%2)@)@	NUMBER(38)

EQUIFAX_RECORDS	n_090	@randlov(0,@perclist(1%1,99%0)@)@	NUMBER(38)
EQUIFAX_RECORDS	n_030	@randlov(0,@perclist(45%1,35%0,15%2,1%3,1%4,1%5,1%10,1%25)@)@	NUMBER(38)
EQUIFAX_RECORDS	bal_bk	0	NUMBER(38)
EQUIFAX_RECORDS	pmt	@randlov(0,@perclist(80%@randrange(0,1000)@,20%@randrange(1000,10000)@)@)@	NUMBER(38)
EQUIFAX_RECORDS	county	@upper(@seqlov(0,@seedlist(State_County,S)@,5)@)@	VARCHAR(20)
EQUIFAX_RECORDS	pim	@randrange(0,9)@	VARCHAR(20)
EQUIFAX_RECORDS	n_fcs	0	NUMBER(38)
EQUIFAX_RECORDS	mortgageindicator	@randlov(0,@perclist(45%1,55%0)@)@	VARCHAR(20)
EQUIFAX_RECORDS	n_cur	@randrange(1,6)@	NUMBER(38)
EQUIFAX_RECORDS	term	@randrange(0,11)@	VARCHAR(20)
EQUIFAX_RECORDS	n_closed_pos	@randlov(0,@perclist(10%1,89%0,1%2)@)@	NUMBER(38)
EQUIFAX_RECORDS	bal_090	@randlov(0,@perclist(99%0,1%@randrange(0,60000)@)@)@	NUMBER(38)
EQUIFAX_RECORDS	currentrisk	@randrange(0,14)@	VARCHAR(20)
EQUIFAX_RECORDS	product	@randlov(0,@list(AB1,AF2,AF1,AB2)@)@	VARCHAR(20)
EQUIFAX_RECORDS	bal_svr	@randlov(0,@perclist(99%0,1%@randrange(0,60000)@)@)@	NUMBER(38)
EQUIFAX_RECORDS	consumer_age	@randrange(0,7)@	VARCHAR(20)

Publish to CSV

repeater	performance based on number of rows
100,000	14 min 51 seconds

Administrating

As a product administrator you are responsible for the configuration and maintenance of Test Data Manager. This section includes administration information for the following components:

- Repository
- CA TDM Portal
- Datamaker

Repository Administration

This section provides information about maintaining the repository and working with remote repositories.

For common database maintenance procedures such as backup, follow the best practices established by the database vendor.

Copy Remote Repository

The Remote Repository Copy Functions in Datamaker enable connections between two repositories and make it possible to copy data from one repository to another. An administrator accesses the Remote Copy Functionality from the Project Manager window from any of the following levels:

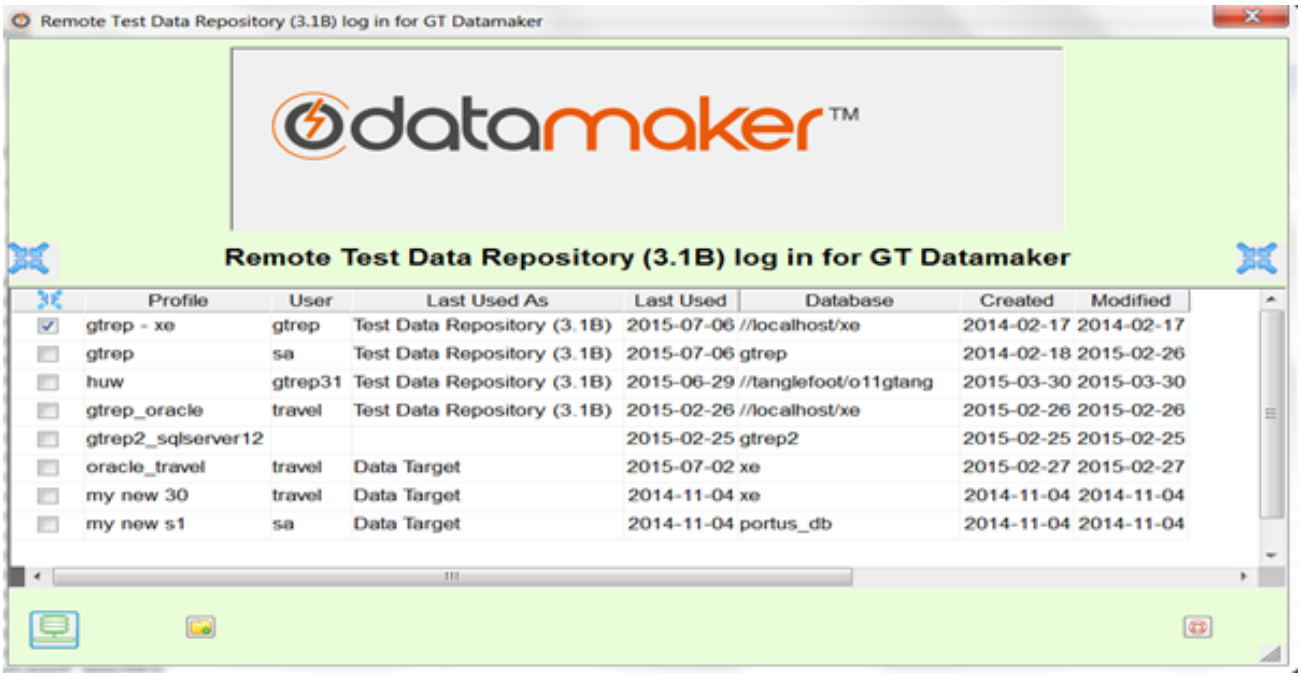
- Top level
- Project level
- Version level
- Any level below a version level

NOTE
The versions of the remote repositories must be the same.

You can perform the following actions:

Connect Remote Repository

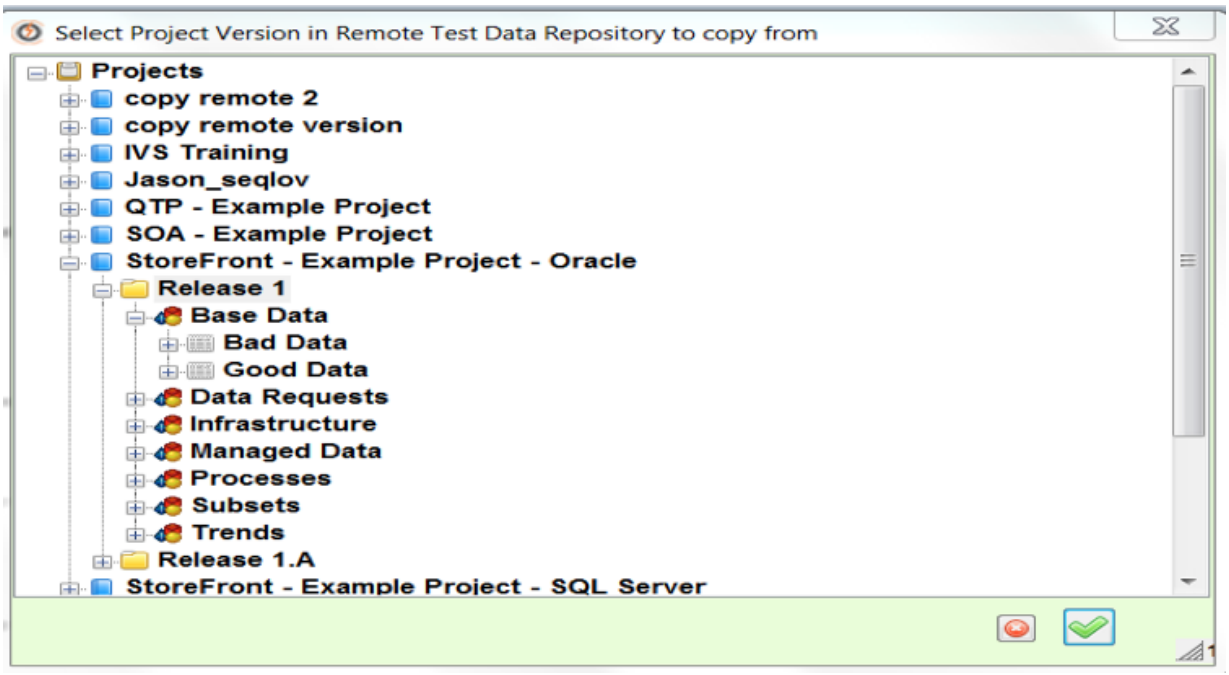
Select the remote repository for connection:



Select Remote Projects

Follow these steps:

- Select remote functionality to display the Remote Projects window:



- Select the remote project
- Select the right remote level for the menu option
- Click the green arrow

Register Tables

Follow these steps:

- Select a remote version in the Remote Projects window and click the green arrow.
- Select a table for copying the remote definition from in the List of registered tables in the selected remote version. If any selected tables are already registered in the current version, a warning appears asking to continue or not. Cancel the operation by selecting No.

Copy Connection Profile

You can copy remote connection profiles to both SQL server and Oracle repositories. If the owner or group of the profile does not exist, the profile is not copied.

Follow these steps:

1. Verify that the owner or owning group of the connection profile exists in the local repository.
2. Connect to the repository to which you want to copy connection profiles from a remote repository.
3. Expand the project tree and right-click the **Projects** parent node. Click **Remote actions from different repository**, then **Copy Profiles**.
4. If prompted, connect to the remote repository.
5. Wait for the copy to finish and review the summary dialog.

Copy Variables

Follow these steps:

1. Select any project/level to copy remote variables for the selected remote level to the current project level in the Remote Projects window. A list of all the variables available to the remote project level are displayed.
2. Select the variables to copy to the current project. The top left four buttons provide access to:
   - Check if the variables are used
   - Variables preview value
   - Select and deselect the variables
3. Select which remote level variable to copy to which current level. By default the equivalent levels are displayed, but you can select all the remote level variables to be copied to the version level, etc.
4. Select Yes or No to proceed or to cancel copying the variables at this stage. If any variables are already defined in the current project/level, you are asked if you want to overwrite the values.

Copy Data

Follow these steps:

1. Select a data group in the current project to be able to copy remote data. Warning is given if a data group is not selected.
2. Select a data group to copy the data in Remote Projects window. Click the green arrow. A new window pops up to display the list of the tables that are used in the selected remote data group.
3. Select the tables from which to copy the data. The system informs the user if the selected tables are already registered in the current version with the same definition.
4. Select to copy the variables of the remote data group when the remote data are copied.

Note: In case one or more remote selected tables are not registered in the current version or their definition is different, warning is given and user is asked to select the tables to register. Select the tables to register/reregister or to proceed with the ones which are already registered with the same definition.

If yes is selected, used variables in the data pool are copied to their equivalent level. A project level, or version level or data level variable that is used in the data pool is copied to the current project level, the version level or data level respectively. The remote publish level variables are copied to the publish level of the current project.

If any variables are already defined in the current project, the system queries overwriting of the existing variables.

Copy Version

When copying a remote version, all the registered tables of the remote version are registered with the same definition of the new version. The following are also copied:

- Variables (optional)
- Data pools
- Actions
- Subsets
- Flowcharts
- Test Matches, are copied if the publish level of the project of the new version is a Data level.

Otherwise a warning message is given to inform the user of:

- Default Publish Jobs (optional)
- Transformation Maps (optional)

In Remote Projects window, select a remote version to copy to the current project and click the green arrow.

To simplify:

- Remote Project --> remote-project
- Current Project --> to-project
- Remote Version --> remote-version
- New Version --> to-version

Specify the name and the description of the new version that the selected remote version is copied to. Choose to copy project level variables to:

- The version level
- The project level of to-project

Note: You can also not copy.

Depending on the first publish level and the data level of the remote and the current projects, some of the definitions of the variables with the same name might be lost. In this case, a message is given to indicate the level at which this might occur.

Subsets, Actions, Flowcharts, and Tags of the remote version are copied to their equivalent level. Select either to copy the remote tags or not.

Case where remote-project has the same levels as to-project

- Each level of remote-version is copied to its equivalent level in to-version
- Variables of remote-levels are copied to its equivalent to-level, considering the first publish level of to-version.

Example: If the first publish level of to-version is level 2, then remote-version level 1 variables are copied to level 1 of to-version and all the variables of remote-version of level2 and above will be copied to level 2 of to-version. Hence the variables with the same name in those levels of the remote-version are copied with their definition in the lowest level. A variable warning is given in this case.

Case where remote-project has fewer levels than to-project

- Data level of the remote-project is copied to the data level of to-project and the remote levels before the data level are copied to the levels before the **to-project** data level. Extra missing levels are created by default.

Example: If remote-project has 2 levels and to-project has 6 levels, then:

- – Remote-version level 2 which is the data level is copied to level 6 of to-version
  – Remote-version level 1 is copied to level 5 of to-version
  – Extra levels, level 1, level 2, level 3, and level 4 are created by default.

Note: Generally the variables of each remote level are copied to its equivalent to-level by considering the first publish level of to-project. If necessary a variables Warning is given due to first publish level.

Case where remote-project has more levels than to-project

- Data level of the remote-version is copied to the data level of to-version. The levels before the data level are then copied respectively until level 1 of to-version is populated.

Example: If remote-project has 6 levels and to-project has 2 levels then:

- – Remote-version level 6 which is a data level is copied to level 2 of to-version which is a data level
  – Remote-version level 5 is copied to level 1 of to-version
  – No action is taken for Levels 1, 2, 3 and 4 of remote-version

Note1: The variables of remote-levels are copied to its equivalent to-level. considering. The first publish is taken into consideration and if necessary a Variables Warning is given due to first publish level. The missing levels variables are either copied to to-version level or the level one of to-version according to user selection. Subsets, Actions, Tags, Flow Charts of the missing levels are not copied.

Copy Project

When copying a remote project, all the versions and hence the following items for each version are copied:

Registered tables for each version:

- Variables (optional)
- Data pools
- Actions
- Subsets
- Flowcharts
- Test Matches
- Default Publish Jobs (optional)
- Transformation Maps (optional)

The following items are also copied:

- Project Tags (optional – user is asked)
- Project security groups (optional - user is asked)

In the Remote Projects window, select a project to copy to the current repository and click on the green arrow.

Click the double arrow to view project properties. The level details are disabled and cannot be modified. However, the rest of the details can be modified for the new project. All the versions and details of the remote project are copied to the new project in the current repository.

Copy Branches of Project Tree

When copying a branch of the project tree, the following items will be copied for the levels and the data levels of the tree:

- Variables (optional)
- Data pools
- Actions
- Subsets
- Flowcharts
- Test Matches, are copied if the publish level of the current project is a Data level. Otherwise a warning message is given.
- Default Publish Jobs (optional)

Follow these steps in Remote Projects window:

- Select the right level of a remote project (according to the selected level of the current project)
- Click the green arrow to copy the branch under the selected level of the current project.

Considering that remote project data level are copied to the current project data level, to copy:

(Current project data level – Current project selected level) levels,

Select the remote level as:

(Remote project data level - remote project selected level) – 1

Example: If the current project and remote project both have three levels and if the current project selected level is 1, select level 2 of the remote project to copy level 2 and 3 of the remote project under the current level 1.

If the selected level name already exists, the application adds _1 or _2 and so on.

A confirmation to copy the variables is displayed.

If any tables used in a data group are not registered in the current version or their definition is different, a warning is given. The user is also prompted to select the tables to register.

Copy of Generic versions

When copying a project, all the versions and their details including Generic versions are copied.

The following functionalities are **not** applied for Generic versions:

- Copy a Generic version (Although the Generic version is copied, not all Remote Copy functionality is available)
- Copy Data belonging to a Generic version, or Copy Data to a data group belonging to a Generic version
- Copy a branch of the project tree belonging to a Generic version, or Copy a branch of a project tree to a Generic version
- Copy Transformation maps

The rest of the copy remote functionalities which can be applied to an ordinary version can be applied to a Generic version.

Copy of Default Publish Jobs

Default Publish Jobs are copied for the following actions:

- Copy Projects
- Copy Versions
- Copy branches of a project tree

If a default publish job is using **values for variables from a Data Pool**, the job is copied without the link for the **values for variables from a Data Pool** which could be added manually by user.

Also if the default publish job is **Publish to Data Pool,** the job is copied as **Publish to file (CSV)** instead of publishing to a Data Pool which can be modified manually.

Copy Functions (Remote Repository)

Copy Selected VTF (Remote Repository)

The menu option option is available when you select a project level in the Maintain Project window. A selected VTF can be copied only to a CA Agile Requirements Designer type level. You are informed if the current level is not a CA Agile Requirements Designer.

In the Remote Projects window, select a VTF. Click the green arrow to copy the selected VTF to the current selected level.

Copy Selected Transformation Map (Remote Repository)

Follow these steps:

1. Select a version in Remote Projects window which displays the list of the Transformation Maps to select for copy.
2. Select a map and click the green arrow.
3. Enter the name of the new transformation map.

The new transformation map, which is a copy of the selected remote Transformation Map, is populated for the current version.

Copy Selected Subset or Action (Remote Repository)

Follow these steps:

1. Select a Subset or action to copy to enable the green arrow in Remote Projects window.
2. Click the green arrow to copy the remote action or subset to the current level. If the action is a stored program, the program is copied too and linked to the copied action.

Copy Selected SQL Program (Remote Repository)

Follow these steps:

1. Select an SQL Program to copy in the Remote Projects window.
2. Click the green arrow to copy the remote SQL Program to the current version. If the current version has an SQL Program with the same name, by default the application adds _1 or _2 and so on.

Copy Selected Tag (Remote Repository)

Follow these steps:

1. Select a project with Tags in Remote Project window to show a list of all the Tags. **Example**: Tag Type: Table and Tag Type: Column lists Table Tags and Column Tags respectively.
2. Select a Tag.
3. Click the green arrow to copy the selected Tag to the current project with the name entered by user.

If the name already exists, user is asked to migrate the Tag to the existing one, or to cancel. The existing Tag is compared with the selected Tag to copy and informs the user if they are both the same. Otherwise, the Tag parts which are missing are copied.

Copy Functions (Same Repository, Different Project)

These Copy options are available for:

- Flow Charts
- Actions
- Transformation Maps
- SQL Programs
- Tags

Copy Selected VTF (same Repository, different Project)

Follow these steps:

1. Select a VTF from Project Manager window
2. Right click the menu option *Copy …* which opens the project window to select an CA Agile Requirements Designer level. Select the right level
3. Click on the green arrow to copy the selected VTF in the current project to the selected location.

Copy Selected Action (same Repository, different Project)

Follow these steps:

1. Select an Action from the Project Manager window
2. Right click the menu option Copy … which opens the project window to select a level for copying the action
3. Click on the green to arrow to copy the selected Action in the current project to the selected location

Copy Selected Transformation Map (same Repository, different Project)

Follow these steps:

1. Select a Transformation Map from the Project Manager window for the current project version
2. Right click on the menu option *Copy to the Different Project* which opens the project window to select a version for copying the Transformation Map.
3. Select a project version to enable the green arrow.
4. Click on the green arrow to copy the selected Transformation Map in the current project to the selected version.

Copy Selected SQL Program (same Repository, different Project):

Follow these steps:

1. Select a SQL Program from Manage Save SQL Program window(Menu option Tools/ Manage Save SQL Program)
2. Right click menu option *Copy …* which opens the project window for user to select a version for copying the SQL Program
3. Select a project version
4. Click on the green arrow to copy the selected SQL Program in the current project to the selected location

Copy Selected Project Tag (same Repository, different Project)

Follow these steps:

1. Select a Tag From Maintain Object tags window (Menu option Tools/ Maintain Object tags)
2. Click on the Picture button *Copy to a Different Project* to open the project window
3. Select a project for copying the Tag
4. Select a project version to enable the green arrow
5. Click on the green arrow to copy the selected Tag in the current project to the selected location. For more information please refer to Copy Selected Tag (Remote Repository)

CA TDM Portal Administration

This page lists administration tasks that you can perform from the CA TDM Portal. As an administrator of the Portal, you are responsible for the following tasks:

- User management
- Connections to data sources and external servers
- Other global settings
- Basic troubleshooting

Most of these tasks are available from the Configuration menu in the CA TDM Portal.

LDAP Integration with the CA TDM Portal

LDAP enables your security teams to authenticate user access and privileges from a central location. The CA TDM Portal lets you integrate with the following LDAP implementations:

- Microsoft Active Directory (MS AD)
- Oracle Directory Services

Where examples on this page refer to integration with Active Directory (AD), integration with other supported LDAP implementations requires the same process in TDM Portal.

The following topics cover the integration-related information:

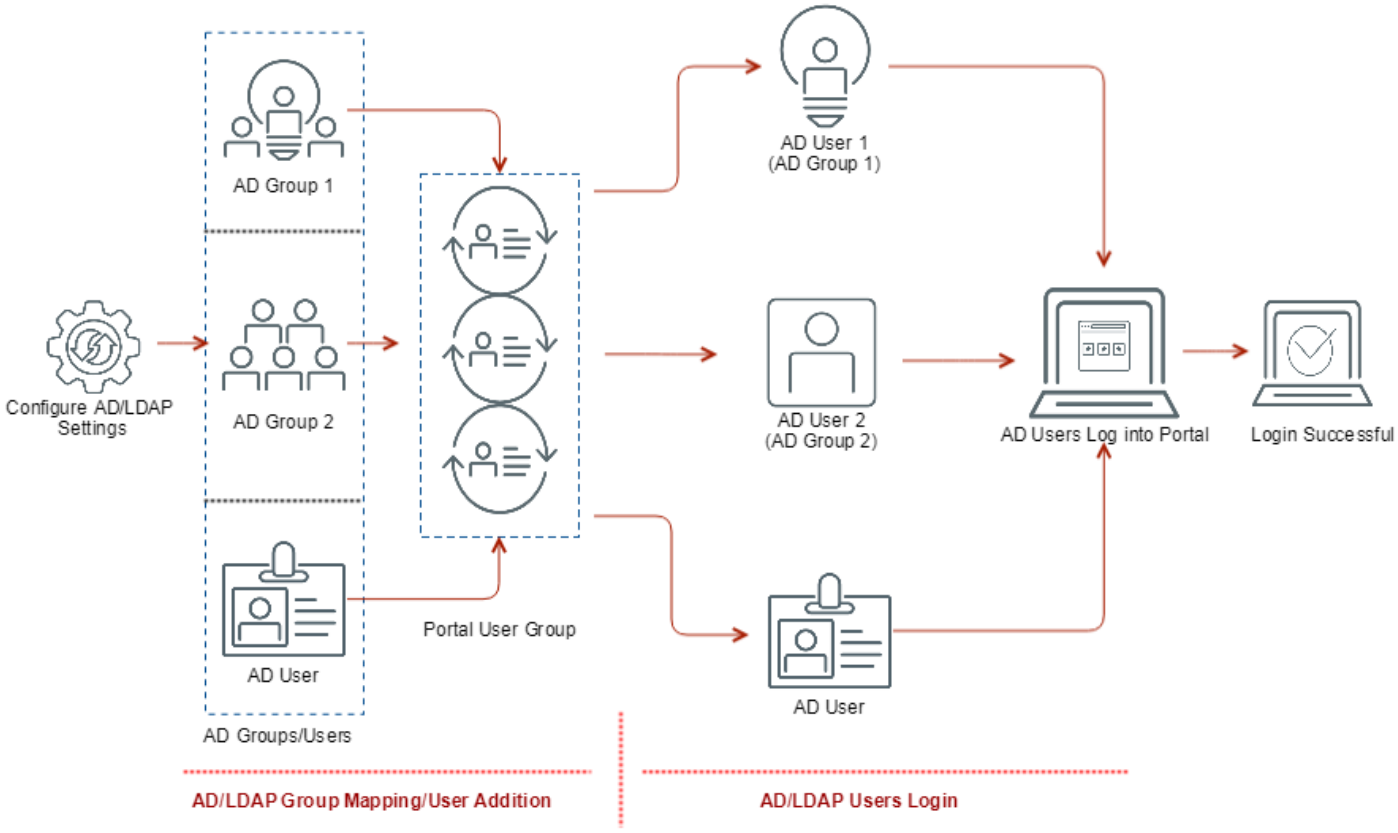
Tutorial Video

Watch the "Integrate Active Directory with the CA TDM Portal" Youtube video for a visual walk-through of a use case of integrating AD with the CA TDM Portal.

Integration Flow

The following diagram shows a simplified version of the integration:

Figure 12: Active Directory Integration with the CA TDM Portal



Considerations

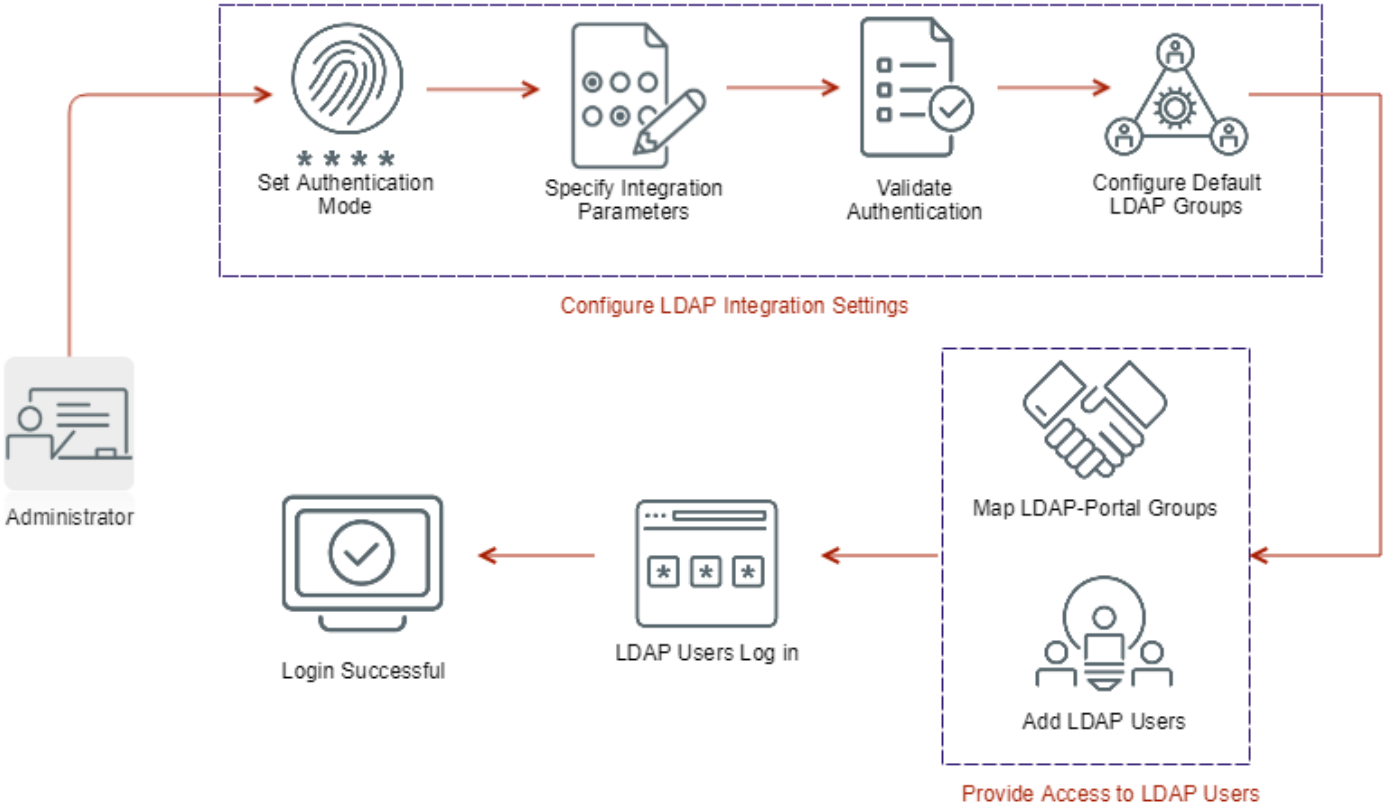
Review the following considerations:

- Only a single Active Directory and Active Directory with a sub-Active Directory (child) are supported. No disjoint Active Directories are supported.
- When you upgrade from a previous CA TDM Portal release (based on the supported upgrade path) to this release, all existing Active Directory users in the previous Portal release are automatically migrated to this release. You do not need to perform this task manually.
- Administrators can decide to hide Native Users in LDAP Mode.

Process

The following diagram shows the detailed process steps:

Figure 13: AD integration process steps



To allow appropriate LDAP users to access the CA TDM Portal, ensure that you perform the following tasks:

1. **Configure the LDAP Integration Settings**.
   a. Set the authentication mode.
   b. Specify the Integration parameter values.
   c. Validate the authentication.
   d. Configure the default LDAP groups.
2. **Provide Access to LDAP Users**.
   – Map LDAP groups to the CA TDM Portal user groups.
   – Add LDAP users to the CA TDM Portal user groups.
3. **Log in LDAP Users.**

Configure the LDAP Integration Settings

The first step in integrating LDAP with the CA TDM Portal is to specify appropriate LDAP integration settings. The settings include selecting AD/LDAP as the authentication mode, providing values for the related parameters, and specifying default LDAP groups.

Follow these steps:

1. Access the CA TDM Portal as an administrator (super administrator).
2. Click **Configuration, Authentication** in the left pane.
   The **Authentication** page opens.
3. Configure the following parameters to integrate LDAP with the CA TDM Portal:

– **Source**
Specifies the type of authentication that you want to use—Active Directory authentication or native authentication:
  - **AD/LDAP**
    In Active Directory authentication, the user authentication happens against Active Directory. Select **AD/LDAP** as the authentication mode to integrate LDAP with the CA TDM Portal, and proceed to specify information for the remaining fields in this procedure.
  - **Native TDM**
    In native authentication, the CA TDM repository is used to verify whether a specific user is present in the repository. If the user is present, the user is authenticated and is allowed to log into the application. For native authentication, select **Native TDM** and click **OK**.

  Note: You do not need to restart the CA Test Data Manager Portal service when you change the authentication mode.

The following are the basic settings:

– **Host Name**
Specifies the host name or IP address of the computer where LDAP is available.
Example: `192.168.255.255`

– **Port Number**
Specifies the port where LDAP is listening.
Example: `389`

– **Use LDAPS**
Specifies whether the AD/LDAP server is configured with LDAPS mode.

– **User DN**
Specifies the distinguished name of the user to use when connecting to the LDAP server.
Example: `CN=administrator,CN=users,DC=ca,DC=com` or `user@domain.name`

– **Password**
Specifies the password that is associated with the user specified in the **User DN** field.
Example: `P@ssword01`

The following are the additional settings:

– **Referral Strategy**
Specifies whether you want to Follow or Ignore the reference to another source if the user in one group is also a part of the other group. Select the respective option from the drop-down list.

– **Base DN**
Specifies the base distinguished name to use for searching users and groups in the LDAP server.
Example: `DC=ca,DC=com` or `CN=users,DC=ca,DC=com`

– **User Class**
Specifies the name of the LDAP user object class to use when loading users.
Example: `person` or `organizationalPerson` .

– **User ID Attribute**
Specifies the attribute field to use when loading the user name. Based on the setting configured on your LDAP server, use the related attribute to uniquely identify users. For example, your LDAP server can use `CN` , `mail` , `uid` , or `userPrincipalName` to identify users.
Example: `CN`

– **User Container**
Enter the RDN of the container to use when loading the users. Delimit multiple RDNs with a '|' character.
Example: `cn=users|cn=admins`

– **Custom User Filter**
Additional LDAP filter for filtering searched users. Make sure it starts with '(' and ends with ')'

– **Group Class**
Specifies the name of the LDAP group object class to use when loading groups.

Example: `group`

- **Group ID Attribute**
  Specifies the attribute field to use when loading the group name.
  Example: `CN`
- **Group Container**
  Enter the RDN of the container to use when loading the groups. Delimit multiple RDNs with a '|' character.
  Example: `cn=group1|cn=group2`
- **Group Member Attribute**
  Specifies the attribute field to use when loading group members from the group.
  Example: `member`

4. Click **Test** to verify that the configuration details are valid and are working. This testing also verifies the availability of LDAP users and LDAP groups in the specified LDAP configuration. That is, at least one LDAP user must be present and at least one LDAP group has one LDAP user.
   A success message indicates that the details are valid.
5. Click **OK**.
6. Click **Next**; the following actions happen:
   - All specified details are saved.
   - The **Default Group Configuration** page opens. This page lets you define the default LDAP groups so that they can get the admin and tester access. This setting is optional.
     By default, when a project is created in the CA TDM Portal, two default CA TDM Portal user groups—ADMIN and TESTER—are also created. The CA TDM Portal also lets administrators select specific LDAP groups as the default LDAP groups for the admin and tester access. The CA TDM Portal achieves this by mapping the selected LDAP groups to the ADMIN and TESTER user groups. This mapping ensures that the two LDAP groups are automatically mapped to the ADMIN and TESTER user groups whenever a new project is created in the Portal. These mapped LDAP groups then become the default LDAP groups for the created project. This ability eliminates the administration overhead of manually mapping the default LDAP groups every time a new project is created. Users belonging to the default LDAP groups get the appropriate privileges depending on the mapped default Portal user group.
7. To configure the default LDAP groups, do the following:
   - **Select default AD group(s) for ADMIN access**
     Lets you search for and select the required LDAP group to which you want to provide the administrator access. The selected LDAP group is mapped to the ADMIN user group. All members of the LDAP group get the administrator access for the created project.
   - **Select default AD group(s) for TESTER access**
     Lets you search for and select the required LDAP group to which you want to provide the tester access. The selected LDAP group is mapped to the TESTER user group. All members of the LDAP group get the tester access for the created project.
8. Click **Finish**.
   A message states that the authentication settings are configured successfully.
9. Click **OK**.

You have successfully configured the LDAP integration settings. You can now proceed to provide access to the LDAP users.

Provide Access to LDAP Users

Users who are members of the default LDAP groups have access to the CA TDM Portal. But, they get access to all the projects that are created after the configuration. In your organization, you might have users whom you do not want to include in these default LDAP groups. You want to give them access based on the business requirements; for example, give them access only to a specific project. To do so, you can use the following methods:

- Map LDAP Groups to CA TDM Portal User Groups.
- Add LDAP Users to CA TDM Portal User Groups.

Map LDAP Groups to CA TDM Portal User Groups

You can map LDAP groups to the CA TDM Portal user groups. With this mapping, when users belonging to a mapped LDAP group try to log into the CA TDM Portal for the first time, they are automatically added to the CA TDM repository. You do not need to add them explicitly to the CA TDM Portal user group. Such users can then log into the CA TDM Portal using their LDAP credentials. They get access to the same resources that are available to the other users who are already members of the mapped CA TDM Portal user group.

You can complete this mapping from the following places in the CA TDM Portal:

- From the edit projects page
- While performing group management

Add LDAP Users to CA TDM Portal User Groups

You can directly add LDAP users to the appropriate CA TDM Portal user groups. When you add LDAP users to the CA TDM Portal user groups, they are automatically added to the CA TDM repository. They can then log into the CA TDM Portal using their LDAP credentials. This ability helps you avoid overhead tasks that are associated with the manual process of adding LDAP users to the repository.

For more information about how to add LDAP users to the CA TDM Portal user group, see User and Group Management.

Log in LDAP Users

After you configure the LDAP integration settings and provide access to LDAP users, all relevant LDAP users can then log in to the CA TDM Portal using their LDAP credentials. The logged in LDAP users can perform all the required operations depending on their association with the CA TDM Portal user group.

Example

The Example: Active Directory Integration article includes an example scenario that explains the complete end-to-end integration.

Troubleshooting

Review the following troubleshooting information:

Some Valid LDAP Users are Unable to Log in

Symptom:

In my organization, some valid LDAP users are unable to log into the CA TDM Portal; whereas, some other LDAP users can without any issue. How can I resolve this problem?

Solution:

If users are spread across different organizational units, you must ensure that the LDAP server host name is configured to point to the Global Catalog server instead of the specific LDAP server. For example, Joe belongs to HRGroup and John to FinanceGroup. You have configured the LDAP server to point only to HRGroup. In this scenario, Joe can log in without any issue. However, John cannot log in, because the LDAP server host information is not configured for FinanceGroup. To ensure both the users belonging to different groups can log into the application, point the LDAP server to the Global Catalog server that covers both the groups. This allows users who are members of different groups to log into the application.

Example: Active Directory Integration

This article includes an example scenario that explains how a CA TDM Portal administrator (Joe) can integrate Microsoft Active Directory (AD) with the CA TDM Portal and achieve the following objectives:

- Configure AD integration settings.
- Set default AD groups.
- Map AD groups to the CA TDM Portal user groups.
- Add AD users to CA TDM Portal user groups.

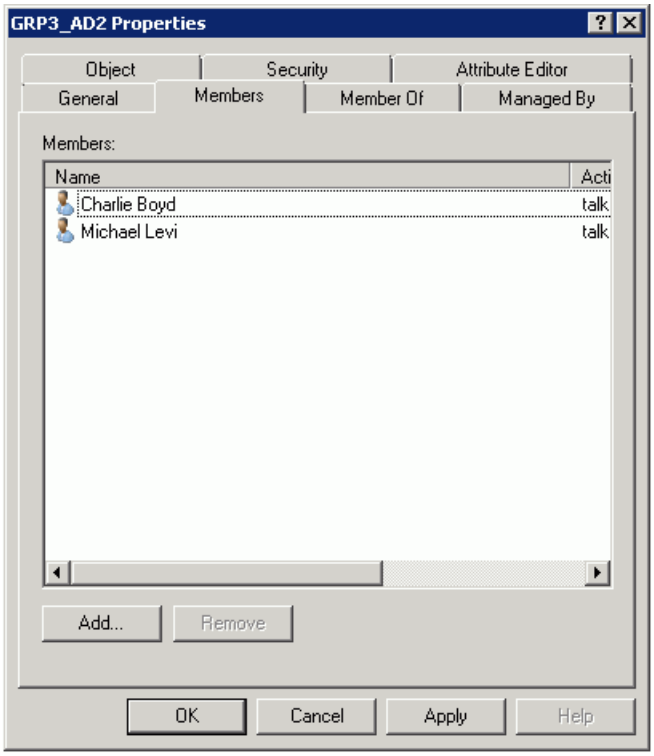
The following topics provide the complete information:

Prerequisites

Review the following prerequisites:

- Ensure that appropriate AD groups and AD users are already available on the AD server.
- Ensure that appropriate AD users are already added to the relevant AD groups.
- Ensure that you have already noted the required AD groups and users that you want to add to the CA TDM Portal.

The following example screen shot shows the AD users (Charlie Boyd and Michael Levi) present in the AD group GRP3_AD2:



Scenario

Joe is a CA TDM Portal administrator. Joe has been asked to allow AD users in his organization to access the CA TDM Portal. This access would enable AD users to log into the CA TDM Portal and perform relevant operations. The business requirement is as follows:

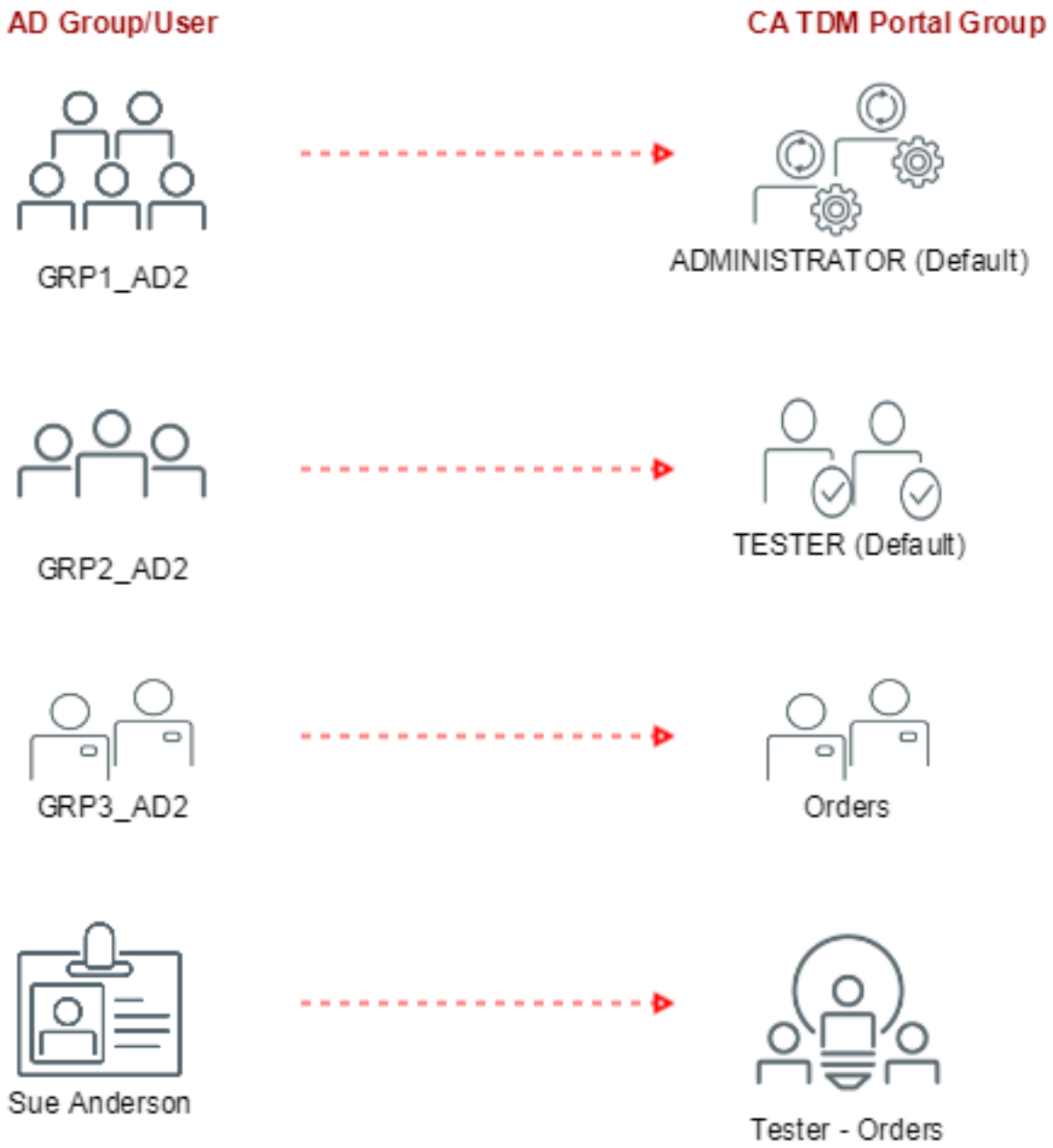
- All users in the AD group GRP1_AD2 must get administrator access for any newly created project in the CA TDM Portal.

Lynn Parker and Marge Walton are members of this group.

- All users in the AD group GRP2_AD2 must get tester access for any newly created project in the CA TDM Portal.
  Cathy Dimitri and Paul Martin are members of this group.
- All users in the AD group GRP3_AD2 must get the same privileges that users of a specific CA TDM Portal user group are getting.
  Charlie Boyd and Michael Levi are members of this group.
- AD user Sue Anderson must be explicitly added to the CA TDM Portal user group (Tester - Orders) so that she can get the same privileges.

The following diagram shows the mapping:

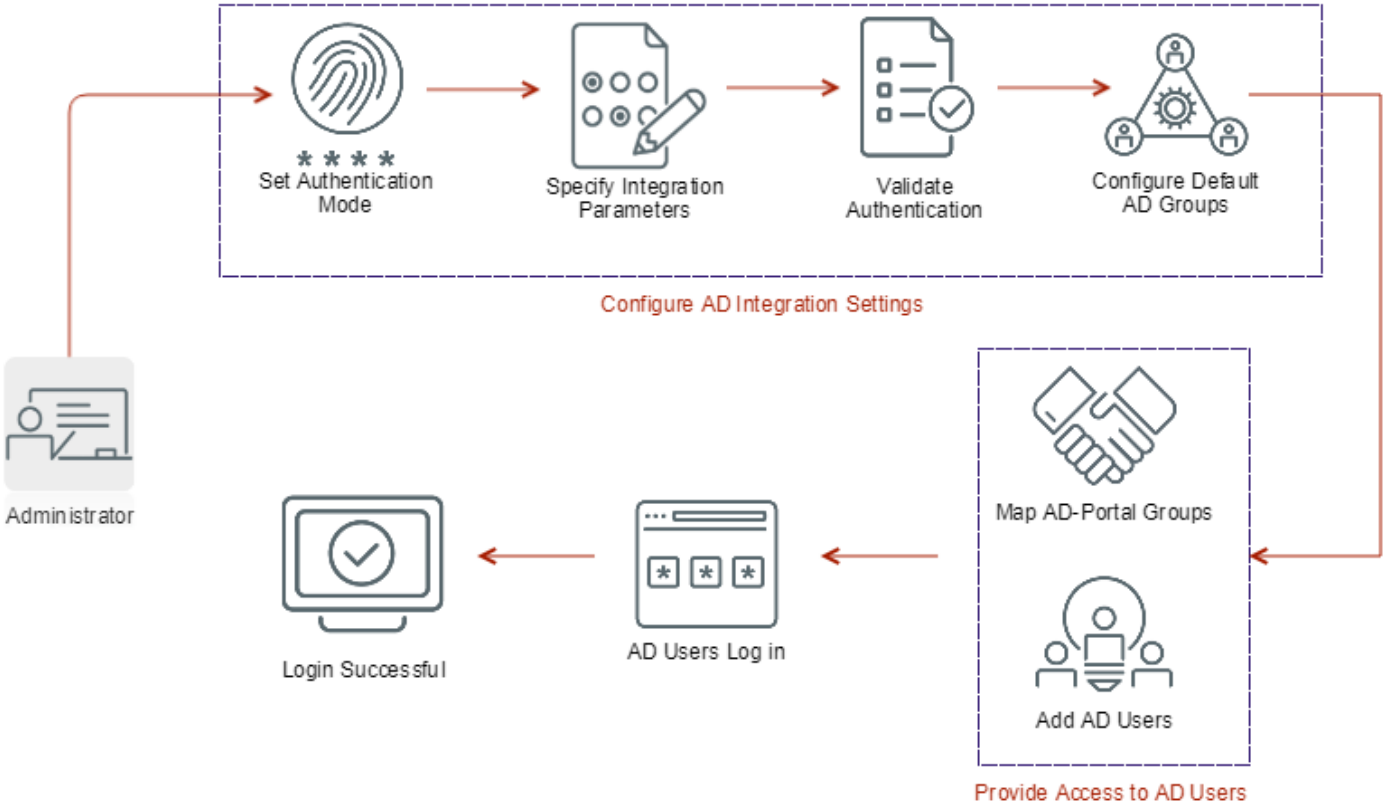
Figure 14: AD and Portal group mapping



Process

The following diagram shows the process steps:

Figure 15: Active Directory Integration Process Steps



Joe follows the following process to meet the requirement:

1. Configure the AD integration settings, which includes the following subtasks:
   a. Set the authentication mode to AD.
   b. Provide the AD integration parameter values.
   c. Test the connection.
   d. Configure the default AD groups for the administrator and tester access, which includes the following subtasks:
      a. Map the AD group GRP1_AD2 to the default CA TDM Portal user group ADMINISTRATOR.
      b. Map the AD group GRP2_AD2 to the default CA TDM Portal user group TESTER.
2. Map the AD group GRP3_AD2 to the CA TDM Portal user group Orders.
3. Add the AD user Sue Anderson to the CA TDM Portal user group Tester - Orders.
   This mapping allows the AD user Sue to get the same privileges that other users of the user group are getting. Note that Sue is not part of the already mapped default AD group GRP2_AD2.

By following this process, Joe can provide appropriate access to all the AD users. This allows them to log into the Portal and perform their operations.

Configure the AD Integration Settings

Configure the AD integration settings to specify the authentication mode, provide values for the integration parameters, and specify default AD groups.

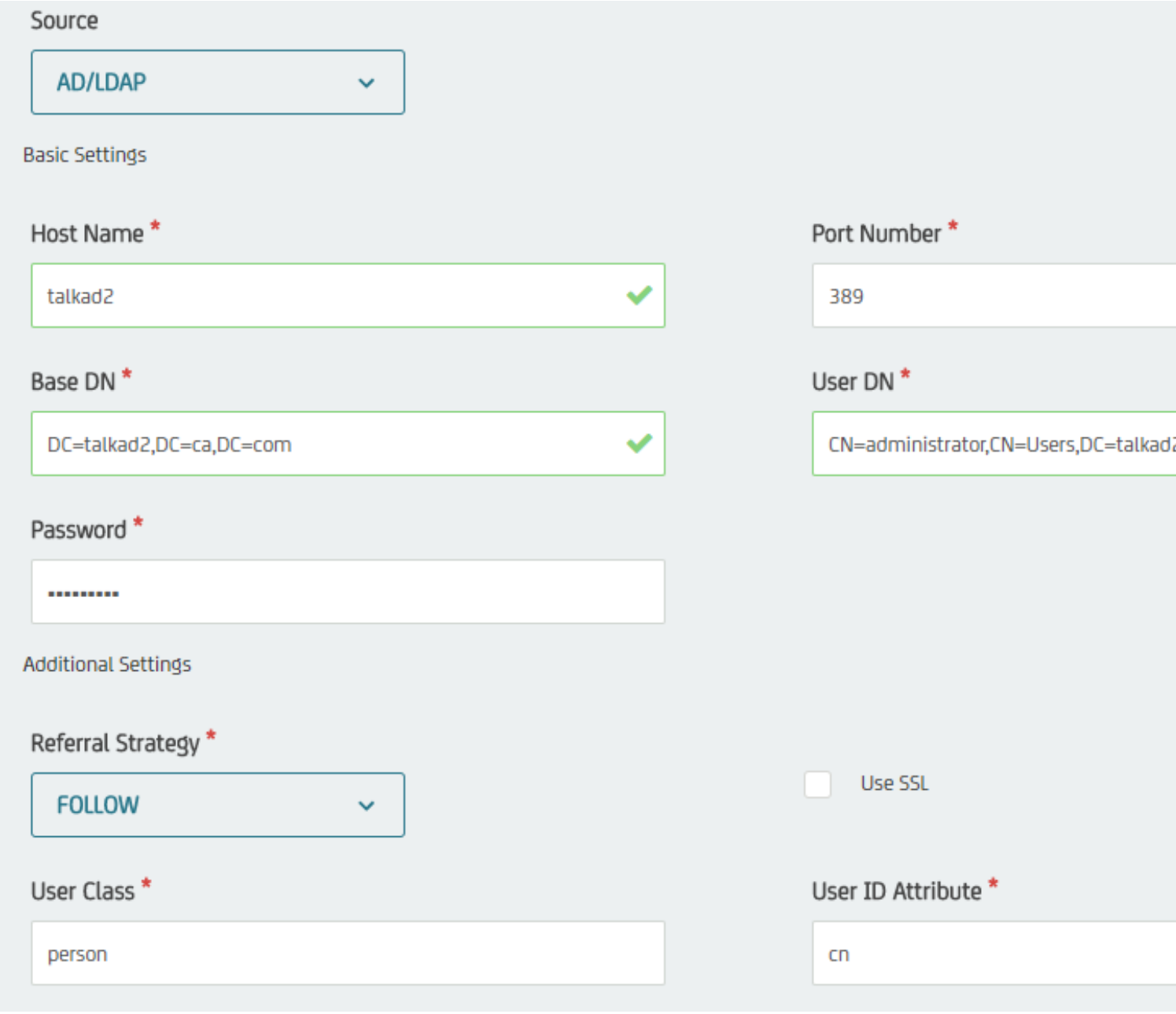
Follow these steps:

1. Access the CA TDM Portal by using your administrator credentials.

2. Expand **Configuration** in the left pane and click **Authentication**.
   The **Authentication** page opens.
3. Enter information in the following fields; example values are provided:
   - **Source:** AD/LDAP

   The following are the basic settings:
   - **Host Name:** talkad2
   - **Port Number:**389
   - **Base DN:**DC=talkad2,DC=ca,DC=com
   - **User DN:**CN=administrator,CN=Users,DC=talkad2,DC=ca,DC=com
   - **Password:** Abc@123

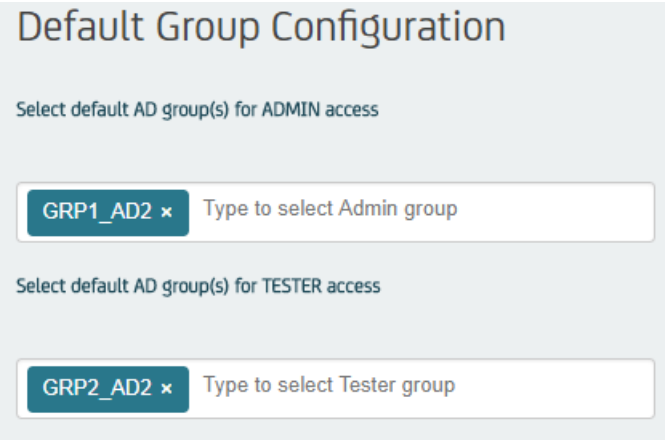
   The following are the additional Settings:
   - **Referral Strategy:** Follow
   - **Use SSL:** No
   - **User Class:**person
   - **User ID Attribute:**cn
   - **User Organization:**cn=Users
   - **Group Object Class:**group
   - **Group ID Attribute:**cn
   - **Group Organization:**cn=Users
   - **Group Member Attribute:**member
4. Click **Test**.
   The CA TDM Portal successfully establishes connection with the AD server, verifies that AD users and AD groups (with users) are present in the specified configuration. The following screen shot shows some of the configured settings:



5. Click **OK**.
6. Click **Next** to configure the default AD groups. In this example, GRP1_AD2 and GRP2_AD2 are identified as the default AD groups.
   Note: This settings is applicable only for those projects that you create after completing the configuration.
   – Enter GRP1_AD2 in the **Select default AD group(s) for ADMIN access** field to search for it. Select the group when it is displayed. This AD group gets the administrator access.
   This mapping makes GRP1_AD2 as the default AD group with the administrator access for any new project that is created. All members of GRP1_AD2 get the administrator access for the created project. Therefore, Lynn Parker and Marge Walton become administrators for the newly created projects.
   – Enter GRP2_AD2 in the **Select default AD group(s) for TESTER access** field to search for it. Select the group when it is displayed. This AD group gets the tester access.

This mapping makes GRP2_AD2 as the default group with the tester access for any new project that is created. All members of GRP2_AD2 get the tester access for the created project. Therefore, Cathy Dimitri and Paul Martin become testers for the newly created projects.

The following screen shot shows the selected default AD groups:



7. Click **Finish**.
   A message appears stating that the authentication settings are configured successfully.
8. Click **OK**.

Joe has successfully set the authentication mode as AD, provided the integration settings, and specified the default AD groups.

Map the AD Group to the CA TDM Portal User Group

Joe also needs to map the AD group GRP3_AD2 to the CA TDM Portal user group (Orders) for the selected project. With this mapping, all users (Charlie Boyd and Michael Levi) in the GRP3_AD2 get the same access as other users of the mapped Orders user group. This access is applicable only for the project associated with the Orders user group.

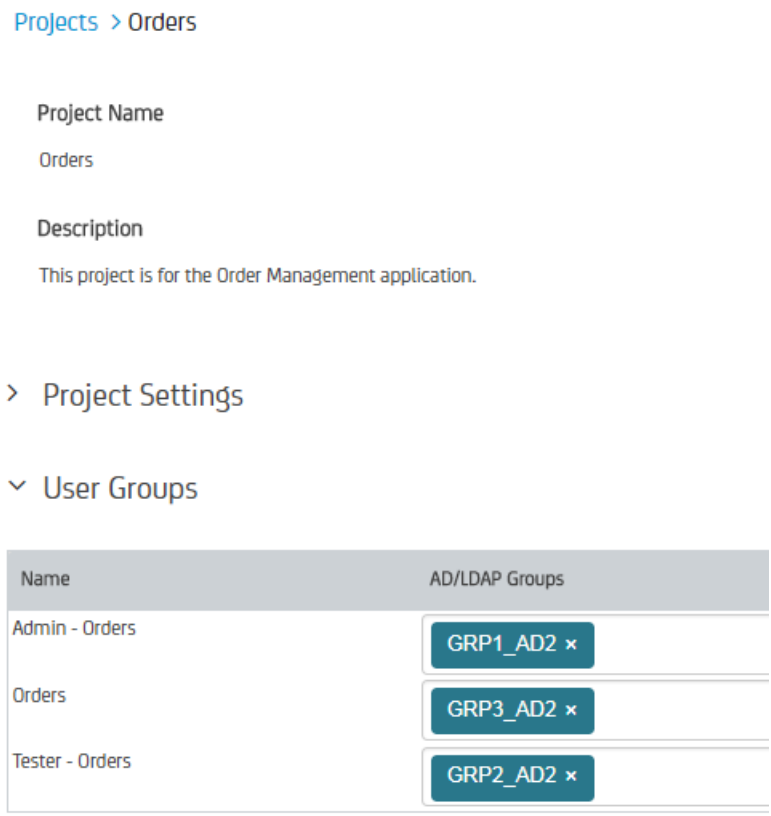
Joe can perform this mapping from two places in the CA TDM Portal—project management or user management page. This procedure shows the steps for the project management page.

Follow these steps:

1. Access the CA TDM Portal by using your administrator credentials.
2. Click the Project Manager icon (gear icon) in the top-blue bar.
3. Create the **Orders** project and click it.
   The **Orders** dialog opens.
4. Expand the **User Groups** section.
   Three CA TDM Portal user groups are assigned to this project. Admin - Orders and Tester - Orders are the default CA TDM Portal user groups. Orders is the third group that is assigned to this project.
5. Search for and enter GRP3_AD2 in the field next to Orders and select the AD group when it is displayed.
   The AD group GRP3_AD2 is added to the field and is mapped to the Orders CA TDM Portal user group.
6. Close the dialog.

Joe has successfully mapped the required AD group to the CA TDM Portal user group. The AD users Charlie Boyd and Michael Levi get the same privileges that are available to others users of the Orders group for the Orders project.

The following screen shot shows GRP3_AD2 mapped to the Orders group. Also, note the presence of two default AD groups. These groups were automatically defined when the Orders project was created:



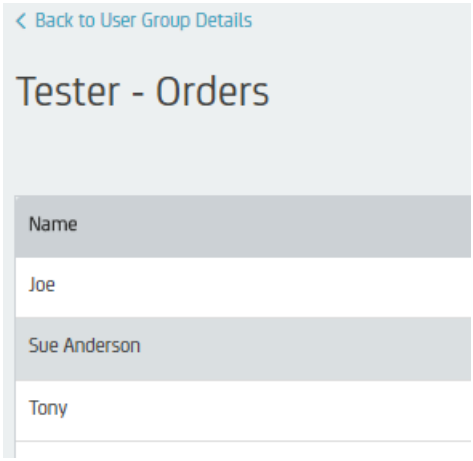
Add the AD User to the CA TDM Portal User Group

The final requirement that Joe has to complete is to add a specific AD user Sue Anderson to the Tester - Orders group, which is a default CA TDM Portal user group with tester access for the Orders project. After Sue is added to the Portal user group, she gets the same tester privileges that other uses of this group are having.

Follow these steps:

1. Access the CA TDM Portal by using your administrator credentials.
2. Click **Configuration, Access Control, User Groups** in the left pane.
   The **User Groups** page opens.
3. Locate and click the **Tester - Orders** CA TDM Portal user group. This is the group to which you want to add the AD user.
   The **Tester - Orders** page opens.
4. Click **Users**.
5. Click **Add User**.
   The **LDAP Users** dialog opens.
6. Enter Sue Anderson in the search field and select the name when it is displayed.
7. Click **Add**.
   The AD user Sue Anderson is added to the list of users for the CA TDM Portal user group Tester - Orders.

The following screen shot shows the AD user Sue Anderson in now present in the list of users added to the Portal group:



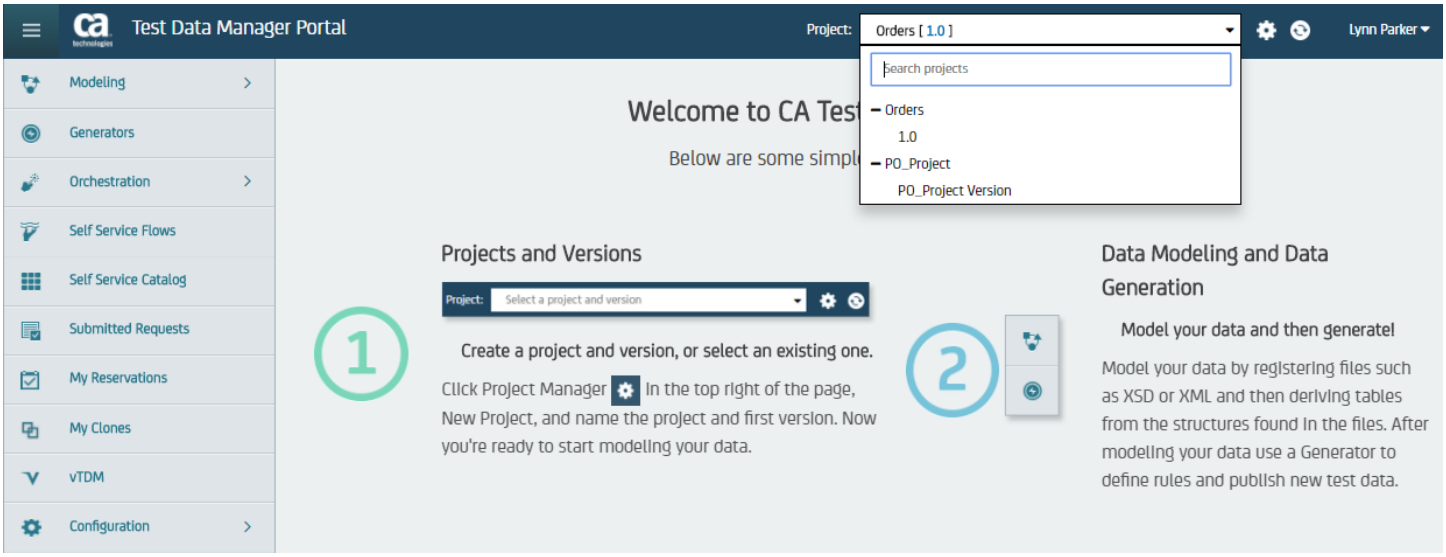
Verify the Added Users/Groups

After completing the user group mapping, all appropriate AD users must be able to log in to the CA TDM Portal. They should also get the same privileges that other Portal users are having. This procedure verifies the same.

GRP1_AD2 Mapping

Lynn Parker and Marge Walton are members of the AD group GRP1_AD2. This AD group is mapped to the default Portal user group ADMINISTRATOR. Therefore, Lynn Parker and Marge Walton must have the same privileges when they log in to the Portal. Also, they must have administrator access to all the projects that are created after completing the default AD group mapping.

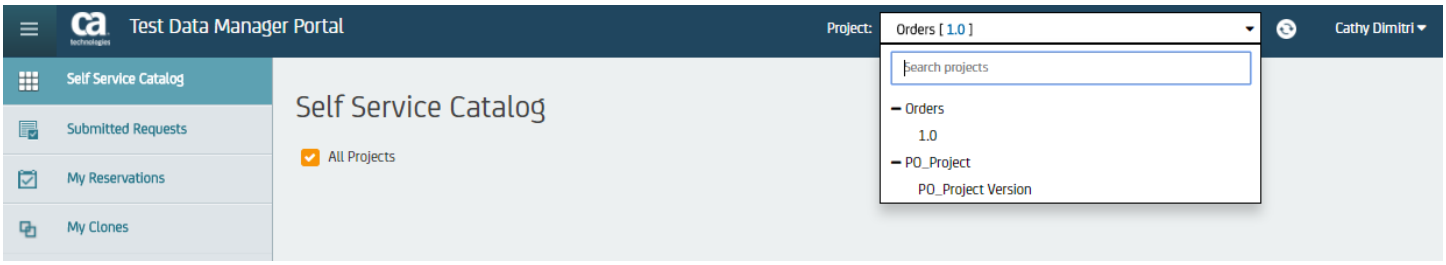
The following example screen shot shows that Lynn Parker has successfully logged in to the Portal. She has also received the appropriate administrator privileges for the two projects—Orders and PO_Project. By default, when the two projects were created after completing the default AD group configuration, the default admin AD group was automatically created for the two projects. All these privileges are as expected:



GRP2_AD2 Mapping

Cathy Dimitri and Paul Martin are members of the AD group GRP2_AD2. This AD group is mapped to the default Portal user group TESTER. Therefore, Cathy Dimitri and Paul Martin must have the same tester privileges when they log in to the Portal. Also, they must have tester access to all the projects that are created after completing the default AD group mapping.

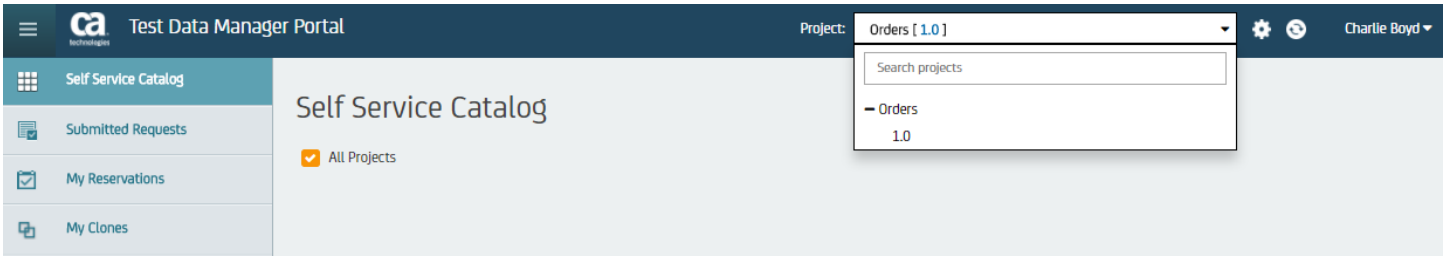
The following example screen shot shows that Cathy Dimitri has successfully logged in to the Portal. She has also received the required tester privileges for the two projects—Orders and PO_Project. All these privileges are as expected:



GRP3_AD2 Mapping

Charlie Boyd and Michael Levi are members of the GD3_AD2 AD group. This AD group is mapped to the Portal user group Orders. Orders has tester privileges in the Portal. Therefore, both the AD users must get the same privileges that users of the Orders group get.

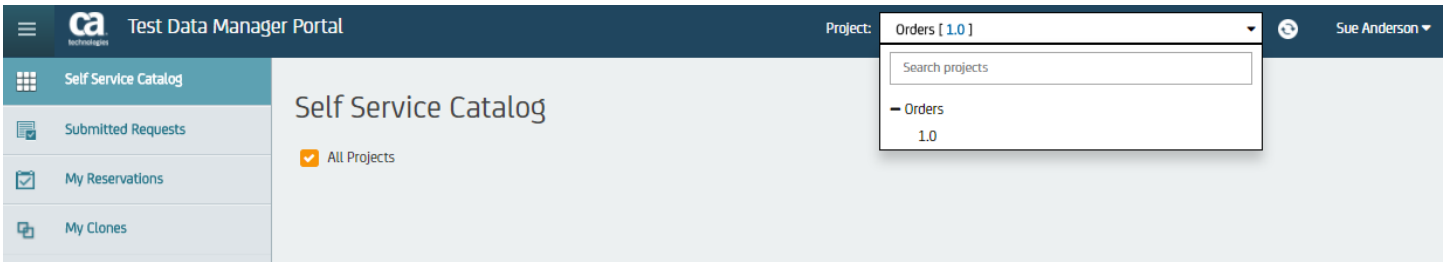
The following example screen shot shows that Charlie Boyd has successfully logged in to the Portal. He has received tester privileges only for the Orders project, not for PO_Project, which is correct:



User Addition

The AD user Sue Anderson is added to the Portal user group Tester - Orders. She must get the privileges based on the Tester - Orders user group.

The following screen shot shows that she has successfully logged in to the Portal. She has tester privileges in the Portal and she can access only the Orders project. All these privileges are as expected:



Disable Native Users in AD/LDAP Mode

Native users are CA TDM Portal-specific users that are created in the repository. By default, native users are visible in the Portal when Active Directory (AD)/LDAP is selected as the authentication mode. Also, the native super administrator login works in the Portal.

However, administrators can decide to hide native users and disable the native super administrator login when the authentication mode is set to AD/LDAP. To do so, administrators configure a property in the `application.properties` file. When the property is set to `true` , no native users are visible in the Portal and native super administrator login is also disabled. In this case, only AD/LDAP users are visible and only they can log in.

Follow these steps:

1. Navigate to the `C:\Program Files\CA\CA Test Data Manager Portal\conf` folder.
   Note: This procedure uses the default CA TDM Portal installation location. If you have installed the CA TDM Portal at a different location, navigate to that location.
2. Locate and open the `application.properties` file in an editor.
3. Remove the comment symbol and set the value of the `tdmweb.security.disableNativeUsers` property to `true` . The default value is `false` .
4. Restart the CA Test Data Manager Portal service.

You have successfully disabled native users in the CA TDM Portal for the AD/LDAP mode.

Configure the Security Token Expiry

The authentication security token that is generated after successful authentication of the user remains valid only for a specific period. After the specified duration is over, the authentication token expires and you cannot use it to perform operations.

You can configure the expiry duration of the authentication token based on your security policies.

1. Navigate to the TDM_HOME\conf location.
   Note: TDM_HOME represents the location where you installed the CA TDM Portal, by default that's `C:\Program Files\CA\CA Test Data Manager Portal\` .
2. Locate and open the application.properties file in a text editor.
3. Specify an appropriate value for the following parameter:
   jwt.expiryInSecs
   Specifies the duration (in seconds) for which you want the authentication token to remain active after it is generated.
   Example: `jwt.expiryInSecs=18000`
4. Save your changes.
5. Restart the service.
   You have successfully configured the expiry duration of the authentication security token.

Configure the Email Server

As an administrator, configure the email server with appropriate information so that the CA TDM Portal is able to send emails to appropriate recipients. For example, all password- and publish job-related emails are sent to intended recipients through the configured email server.

Note: Other than administrators, users having access to the "Settings" security function can also configure the email server.

1. Access the CA TDM Portal by logging in as an administrator.
2. Expand the **Configuration** option in the left pane.

Note: If the left pane is hidden, click the icon (represented by three horizontal bars) in the top left corner to view the pane.

3. Click the **Mail Server Configuration** option in the expanded list.
4. Provide the following information:
   - **Protocol**
     Lets you specify the email protocol that you want to use to exchange the information to and from the email server. This release supports only SMTP.
   - **Host Name**
     Lets you specify the name of the email server.
   - **Port**
     Lets you specify the port number that the email server uses for communication.
   - **From Address**
     Lets you specify the email address from which you want to send the emails.
   - **Require Authentication**
     Lets you provide a specific user name and password that is required to authenticate emails that are sent to the recipients. This option helps ensure that only authorized users can send emails to the recipients. If you enable this option, the **Username** and **Password** fields are displayed. Provide the required credentials in these fields.
5. Click the **Save** icon.
   You have configured the email server. All the emails originating from the CA TDM Portal are now sent using the configured email server.

Next steps:

- Ensure that relevant users have email addresses defined. For more information, see User and Group Management.
- (Optional) Browse to the following directory and edit the provided email templates:
  `C:\Program Files\CA\CA Test Data Manager Portal\Mail Templates`

Configure Data Reservation Email Properties

Administrators can configure the CA TDM Portal to send email notifications to testers about the state of the reservation. This functionality enhances the user experience, because instead of manually trying to check the reservation status, testers can directly receive the notification whenever the process completes.

The email notifications are sent in the following scenarios:

- When a reservation succeeds or fails
- When a reservation is released
- When an environment that is associated with the reservation is deleted
- When a test data model that is associated with the reservation is deleted
  Note: When a project version is deleted, associated environment and test data model are also deleted.

The email notification includes appropriate details about the reservation; for example, it includes the following type of information:

- Name of the reservation
- Environment in which reservation has been made
- Test data model that is associated with the reservation
- Project and version that are associated with the reservation
- Number of resources that have been reserved
- Comments about any failure in the reservation
  For example, if the reservation fails due to already reserved resources, the Comments section includes an appropriate message. Furthermore, those blocked resources are listed in the email and also in the CSV attachment (if configured).
- Status of the reservation

To configure the test data reservation email properties, you update the required parameters in the `tdmdatareservation.properties` file. The default location of the file is `C:\Program Files\CA\CA Test Data Manager Portal\conf`.

Follow these steps:

1. Navigate to the `C:\Program Files\CA\CA Test Data Manager Portal\conf` location.
2. Open the `tdmdatareservation.properties` file in a text editor.
3. Specify appropriate values for the following parameters:
   - **reservation_notification_enabled**
     Sends email notifications to the user who has made the test data reservation. To enable this parameter, set the value to `true`. If you do not want to send the email, set the value to `false`.
     Default: `true`
   - **reservation_notify_all_users**
     (Optional) Includes all the users in the project (which is associated with the reservation) while sending the email. For example, a project can have multiple users. If one user reserves the data, the administrator might want to notify all other users in the project about that reservation.
     To enable this parameter, set the value to `true`. Otherwise, set it to `false`.
     Default: `false`
   - **reservation_notification_with_attachment**
     (Optional) Includes a CSV file as an attachment while sending the email. This CSV file contains the resources (model keys) that have been reserved.
     To enable this parameter, set the value to `true`. Otherwise, set it to `false`.
     Default: `true`
4. Review and save the configuration.
5. Start the CA Test Data Manager Portal service.

You have successfully configured the test data reservation email properties.

Note: You can also customize the test data reservation email template (`ReservationEmailTemplate.vm`) based on your unique requirements. You can find the email template at `C:\Program Files\CA\CA Test Data Manager Portal\Mail Templates`.

Configure CA Service Virtualization Details

To integrate the CA TDM Portal with CA Service Virtualization, perform the appropriate configuration in the CA TDM Portal. The CA TDM Portal uses this configuration information when you try to export the data directly into a virtual service in CA Service Virtualization.

1. Access the CA TDM Portal.
2. Click **Configuration** in the left pane.
   All available options are displayed.
3. Click the **DevTest Portal** option.
4. Enter information in the following fields:
   - **Protocol**
     Specifies the protocol that the DevTest Portal uses. For example, HTTPS or HTTP.
   - **Host Name**
     Specifies the name of the server where the DevTest Portal is running.
   - **Port**
     Specifies the registry web service port number (not CA DevTest Portal port number). By default this is 1505.
   - **Username**
     Specifies the user having access to the DevTest Portal.
   - **Password**

Specifies the password for the specified the DevTest Portal user.

5. Click the Save icon.
   The configuration is saved.

Configure the New Publish Service for CA TDM Portal

The CA Test Data Manager Portal allows you to submit the data generation requests using Generators or Self-Service Catalog. You can configure the CA TDM Portal to use either New Publish Service or Remote Publish to handle the publish requests.

You can modify the application.properties file for the following two parameters to switch between New Publish Service and Remote Publish based on the use cases. The application.properties file is typically available at *C:\Program Files\CA\CA Test Data Manager Portal\conf*.

- ***tdmweb.tdmJobEngineService.useDatamakerToPublish=false|true***
  Default: false
- ***tdmweb.testerSelfService.useDatamakerToPublish=false|True***
  Default: false

Use Cases

1. To run all the publish requests either from Generators or from Self-Service Catalog using New Publish Service use the parameters as follows:
   – *tdmweb.tdmJobEngineService.useDatamakerToPublish=false*
   – *tdmweb.testerSelfService.useDatamakerToPublish=false*
2. To run all the publish requests either from Generators or from Self-Service Catalog using Remote Publish use the parameters as follows:
   – *tdmweb.tdmJobEngineService.useDatamakerToPublish=true*
   – *tdmweb.testerSelfService.useDatamakerToPublish=true/false*
   Note: When configured the Generators to use Remote Publish, Self-Service Catalog cannot use New Publish Service. That means the value of the parameter "tdmweb.testerSelfService.useDatamakerToPublish" is not considered. So the value can be either "true" or "false".
3. To run the publish requests from Generators using New Publish Service and run the publish requests from Self-Service Catalog using Remote Publish use the parameters as follows:
   – *tdmweb.tdmJobEngineService.useDatamakerToPublish=false*
   – *tdmweb.testerSelfService.useDatamakerToPublish=true*

Synchronize Requests to Execute Sequentially

Concurrent requests to the same database target or from the same generator may cause a request to fail with a database integrity violation, because the same key could be generated by different jobs. To prevent concurrent requests to the same database target, or from the same generator, configure Test Data Manager to synchronize the requests. Synchronized requests run sequentially, but possibly more slowly. This configuration enables synchronization for Publish, Testmatch, Rally and HP ALM requests.

Follow these steps:

1. Navigate to the `TDM_HOME\conf` folder.
2. Open the `application.properties` file in a text editor.
3. Specify the following parameters:
   – **tdmweb.tdmJobEngineService.queueConcurrentJobs=true|false**
   Specifies whether you want to synchronize the requests or not. To synchronize and execute the requests sequentially, leave this parameter set to TRUE. Set this parameter to FALSE to run jobs concurrently.

Default: true

- **tdmweb.tdmJobEngineService.queueJobsOn=DP_SOURCE_AND_TARGET | CP_TARGET**
  Specifies at which level to queue the requests. Set this to DP_SOURCE_AND_TARGET to synchronize the requests at the generator (data pool). Set this to CP_TARGET to synchronize the requests at database target.
  Default: DP_SOURCE_AND_TARGET

  NOTE

  In case of synchronizing requests at target, currently the identity of the database target is determined only by the comparison of profile names. If you rename a copy of a profile, and don't change the database target, it is not identified as being the same.
- **tdmweb.tdmJobEngineService.queuePublishOnDPSource=true|false**
  This parameter specifies whether you want to synchronize the *publish* requests at the generator or not. Set this parameter to **true**, to synchronize the publish requests. To run the publish requests concurrently, set this parameter to **false**.This parameter is applicable only, if the **tdmweb.tdmJobEngineService.queueJobsOn=** is set to **DP_SOURCE_AND_TARGET.**
  Default: false
- **tdmweb.tdmJobEngineService.queuedJobTimeout=*n***
  Specifies the maximum number of seconds a request can wait for the previous request to complete. After this time period has passed, the job does not run. The job status is set to "failed" with the output message "The job was queued for too long and has timed out." To disable the timeout, set this value to **0** seconds.
  Default: 3600 seconds

4. Save the file.

   TIP

   You can view the submitted requests, scheduling times, and statuses, in the TDM Portal under Requests.

   You can restart and stop the job engine without disrupting the synchronized requests. On startup, the TDM continues to process the requests that were previously in Queued state. All the jobs that were in Running state are marked as failed.

Configure Access to Requests Results

When you submit a request in the CA TDM Portal, a job is created. You can see the job status on the Requests page. Once the job is completed the request results are available for download against the respective job on the Requests page. These jobs are listed on the Requests page only for a specified period of time. After the specified period the jobs are cleaned up. You can specify the clean-up time period as per your enterprise requirements.

Follow these steps:

1. Open the application.properties file typically available at:
   C:\Program Files\CA\CA Test Data Manager Portal\conf\
2. Go to the end of the file and add the following statements:
   #To enable the job cleanup, set the values of these properties in hours.
   tdmweb.jobs.cleanupInterval=<value>
   tdmweb.jobs.cleanOlderThan=<value>
   Notes:
   - The <value> in tdmweb.jobs.cleanupInterval=<value> indicates the frequency to perform the clean-up. You must specify the value in hours. For example, 12.
   - The <value> in tdmweb.jobs.cleanOlderThan=<value> indicates how old the jobs should be before they are cleaned up. You must specify the value in hours. For example, 720
3. Restart the CA TDM Portal Service.
   Removal of the requests submitted for Projects, Generators and Data Catalog as per the specified parameters is now enabled.

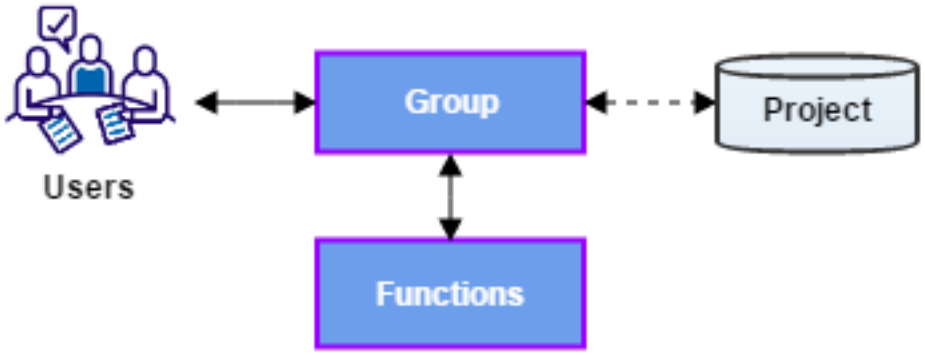
User and Group Management

As an administrator, you can manage users and groups in the CA TDM Portal. Groups are the central component of the CA TDM Portal security model. Group membership determines the projects and functions that users can access. The security model is based on groups and has the following features:

- Users must be associated with a group to access projects and security functions.
- Each group belongs to one project and has associated security functions.
- If a group is associated with all projects, the group is a Super Administrator.

The following diagram shows the security model:

Figure 16: User_Group_Management



Note: Other than administrators, users having access to the "Users and Groups" security function can also manage users and groups.

Manage Groups

You can manage groups as follows:

- Create a group.
- Map AD/LDAP groups to CA TDM Portal user groups.
- Assign functions to a group.
- Add users to a group.
- Delete a group.

Create a Group

Create groups to connect to team projects and security functions.

Follow these steps:

1. Access the CA TDM Portal by logging in as an administrator.
2. Click the **Configuration** option in the left pane to expand it.
   Note: If the left pane is hidden, click the icon (represented by three horizontal bars) in the top left corner to view the pane.
3. Click the **Access Control** option to view the available options.
4. Click the **User Groups** option.
   The **User Groups** page opens.
5. Click **Create User Group**.
   The **Create New User Group** page opens.

6. Enter information in the following fields:
   - **Name**
     Specifies the name of the group that you want to create.
   - **Description**
     Specifies the relevant description about the group.
   - **Project**
     Lets you select an appropriate project from the drop-down list. The created group is associated to the selected project.
7. Click **Save**.
   The created group is added to the table on the **User Groups** page. This table lists all the groups that you create. The table also includes information about the projects that are associated with the group.

You have successfully created a user group and associated it to a project. You can now add security functions to the group or add new users to the group. You can use the search field at the top-right corner to search for a specific user group in the list.

Note: You must map appropriate user groups to those connection profiles that the groups are allowed to use. For more information, see Create and edit Connection Profiles.

Map Active Directory/LDAP Groups to the CA TDM Portal User Group

You can map Active Directory (AD)/LDAP groups to the CA TDM Portal user group. With this mapping, all users from the mapped AD/LDAP groups get the same privileges that users of the associated CA TDM Portal user group have.

This procedure is applicable only when the authentication mode is set to AD/LDAP.

Follow these steps:

1. Follow Step 1 through Step 4 as mentioned in the Create a Group procedure.
2. Click the required CA TDM portal user group to which you want to map the AD/LDAP group.
   The **<User_Group_Name>** page opens.
3. Search for and select the required AD/LDAP groups by using the **Select AD/LDAP group(s)** field.
   The selected AD/LDAP groups are added to the list.

You have successfully mapped the AD/LDAP groups.

Note: You can also do this mapping from the project management page.

Assign Functions to a Group

When you create groups, they have no assigned access functions by default. You must assign access functions to a group. These functions provide required privileges to the users who are added to the group.

Follow these steps:

1. Follow Step 1 through Step 4 as mentioned in the Create a Group procedure.
2. Click the required user group to which you want to assign security access functions.
   The **<User_Group_Name>** page opens. This page lists all the available security functions that you can assign to the group.
3. Expand the **Granted Functions** section and select the appropriate security functions available under different categories. For more information about security functions applicable for the CA TDM Portal, see CA TDM Portal Security Functions.
   The functions are assigned to the group.
   Note: If you select the **Select All Functions** option, that group is made the administrator group. Because all the security functions are assigned to this group, this group has the highest level of privileges in the CA TDM Portal.

You have successfully added security functions to a group. You can now add users to the group.

Add Users to a Group

When you add users to a CA TDM Portal user group, they get privileges to perform actions in the CA TDM Portal based on the security functions and projects that are associated with the group.

The CA TDM Portal also allows you to add AD/LDAP users to the CA TDM Portal user group. The Portal adds the selected AD/LDAP user to the CA TDM repository and maps that AD/LDAP user to the CA TDM Portal user group.

Follow these steps:

1. Follow Step 1 through Step 4 as mentioned in the Create a Group procedure.
2. Click the required user group to which you want to add users.
   The **<User_Group_Name>** page opens.
3. Click the **Users** button next to the group name.
4. Click the **Add User** button.
   Depending on the authentication mode, **Users** or **AD/LDAP Users** dialog opens.
5. Perform the appropriate task based on your authentication:
   - **Users**
     Lets you add users that are already present in the repository to the selected CA TDM Portal user group. Search for and select the native user that you want to add and click the **Add** button.
   - **AD/LDAP Users**
     Lets you add AD/LDAP users to the selected CA TDM portal user group. Search for the required AD/LDAP user, select the user, and click **Add**. The Portal adds the selected AD/LDAP user to the repository and maps that user to the selected CA TDM Portal user group.

   The user is added to the CA TDM Portal user group.
6. Review that the user is now available in the table.

You have successfully added a user to a CA TDM Portal user group.

Delete a Group

If you do not have requirements to use a specific group, you can delete that group from your environment.

Follow these steps:

1. Follow Step 1 through Step 4 as mentioned in the Create a Group procedure.
2. Locate the group that you want to delete.
   Note: You cannot delete the Admin user group that is added by default. You can identify this by observing that the Admin user group row does not have the Delete Group icon (X icon).
3. Click the Delete Group icon (X icon) in the row corresponding to the identified group.
4. Confirm the deletion.
   The group is removed from the list.

You have successfully deleted a group.

Default Groups

When you create a project in the CA TDM Portal, the CA TDM Portal automatically associates two default groups (Admin and Tester) with the project. These default groups contain specific security functions. Users who are assigned to the default groups can access only that functionality in the project that the associated default groups support.

- Admin Group
  This default group includes all the security functions. You can designate a group as an Admin group with all functions granted.
- Tester Group
  The following security functions are assigned to the default Tester group:

- Tester Self-Service
- Publish Data
- Test Match

Note: Existing TDoD users can access the **Self Service Catalog** option in the CA TDM Portal if they are part of the Tester group for a project or they have access to the previously mentioned access functions through any other security group.

Manage Users

You can create users and can add them to groups. Group membership gives users access to projects and functions that are associated with the group. You can manage users as follows:

- Create a user.
- Edit a user.
- Delete a user.

Create a User

When you create a user, you also specify the group with which you want to associate the user. The user can then access security functions and projects that are linked to the selected group.

Follow these steps:

1. Access the CA TDM Portal by logging in as an administrator.
2. Click the **Configuration** option in the left pane to expand it.
   Note: If the left pane is hidden, click the icon (represented by three horizontal bars) in the top left corner to view the pane.
3. Click the **Access Control** option to view the available options.
4. Click the **Users** option.
   The **Users** page opens.
5. Click the **Create User** button.
   The **Create New User** page opens.
6. Enter the following information:
   - **User Name**
     Specifies the name of the user that you want to create.
   - **Email Address**
     Specifies the email address of the user that you want to create. The CA TDM Portal sends all user-related emails to this email ID.
   - **Full Name**
     Specifies the full name of the user that you want to create.
   - **Location**
     Specifies the location of the user that you want to create.
   - **Extension**
     Specifies the phone number of the user that you want to create.
   - **Group Membership**
     Lets you select appropriate groups from the list of available groups. The created user is added to the selected groups. You can also search for a specific group by using the **Search User Groups** field.
7. Click **Save**.
   The user is added to the **Users** page. This table lists all the users that you create. An email to set the password is also sent to the email ID of the user (if the email server is configured correctly). The email includes the user name and a link to set the password. The user follows the link and sets the password. The user can then use the same credentials to log into the CA TDM Portal.

Note: You can customize the default email templates based on your unique requirements. The location of all the default emails is <install_drive>:\Program Files\CA\CA Test Data Manager Portal\Mail Templates.

You have successfully created a user and added it to a group.

Edit a User

After you create a user, if you have a requirement to update user-related information, you can do so. You can reset the password of the user, add the user to an additional group, or remove a group that is associated with the user.

Follow these steps:

1. Follow Step 1 through Step 4 as mentioned in the Create a User procedure.
2. Click the appropriate row corresponding to the user for which you want to update the information.
   Note: You can also use the search field at the top-right corner to search for a specific user in the list.
   The **<User_Name>** page opens with all the details about the user.
3. Update the information as required:
   - Reset the password
     Click the **Reset Password** button to send a reset password email to the email ID of the user. The user can follow the link in the email and can reset the password. This email is sent if the email server is configured correctly; otherwise, you need to copy the link and send it to the user.
     Note: The **Reset Password** button is disabled if the **Email Address** field is blank.
   - Add to a group
     Click the **Add to Group** button to select a group to which you want to add the user. When you click the button, the **Group Membership** dialog opens. You can select and add the appropriate group from this dialog.
   - Remove a group that is associated with the user
     Click the Remove icon (X icon) corresponding to the group from which you want to remove the membership of the user. When you confirm the deletion, the group is removed from the list. The user is no longer a part of the removed group.
4. Review your changes.

You have successfully edited the user information.

Delete a User

If you no longer want a specific user in your environment, you can delete that user.

Follow these steps:

1. Follow Step 1 through Step 4 as mentioned in the Create a User procedure.
2. Locate the user that you want to delete.
   Note: You cannot delete the Administrator user that is added by default. You can identify this by observing that the Administrator user row does not have the Delete User icon (X icon).
3. Click the Delete User icon (X icon) in the row corresponding to the user that you want to delete.
4. Confirm the deletion.
   The user is removed from the list.

You have successfully deleted a user.

CA TDM Portal Security Functions

To access the functionality that is supported in this release of the CA TDM Portal, a user must have access to the following security functions:

- Project Manager
- Maintain Project
- Register Tables
- Actions on Registered Tables
- Data Definition
- Edit Object
- Publish Data
- Delete Object
- Settings
- Users and Groups
- Tester Self Service

The CA TDM Portal has the following additional Profiling security functions for Data Profiling:

- **Execute Data Profiling**User Groups with this security function can:
  - Execute a Data Profiling scan job
  - Review the scan results
  - Submit reports for sign off

  To grant Execute Data Profiling permissions to a user, grant Execute Data Profiling and Job Management permissions to the associated user group for a specific project.
  Note: Defining the Execute Data Profiling permission against a user group for All Projects results in a user with the capability to initiate profiling jobs; however, the user is unable to view the status of the profiling jobs until the job enters the Ready to Review state.
- **Report Sign Off**
  User Groups with this security function can review a Data Profiling scan report and sign off.

For more information about Data Profiling, see PII Audit Using CA TDM Portal.

You can use the Security Functions.

TDM Portal Password Management

The CA TDM Portal lets you manage your password as follows:

Understand the CA TDM Portal Password Policy

The password policy in the CA TDM Portal defines a set of rules and restrictions that determine how passwords are created. This policy ensures that the Portal is able to provide a high level of security to all user accounts. Passwords in the CA TDM Portal adhere to the following password policy:

- Your password must contain at least 8 characters.
- Your password must not contain more than 256 characters.
- Your password must contain at least 1 uppercase letter, 1 lowercase letter, 1 special character, and 1 digit (numeric).
- You can include these special characters (@#$%*^&!~) in your password.
- You cannot use your previous CA TDM Portal passwords.

Retrieve Password (Forgot Password)

If you forgot your CA TDM Portal password, you can set a new password by using the **Forgot Password** link.

1. Access the CA TDM Portal.
2. Click the **Forgot Password** link.
3. Enter your CA TDM Portal user name in the **Username** field.

4. Click **Recover Password**.
   An email is sent to your registered email address. This email includes a link that lets you set a new password.
5. Click the link in the email.
6. Enter and confirm the new password.
7. Click **Save**.
   The new password is saved.

You have successfully specified a new password. You can now use your new password to log into the CA TDM Portal.

Change Password

As part of your profile management, change your password periodically to ensure that your account remains secure.

1. Access the CA TDM Portal by using your credentials.
2. Click **<*User_Name*>** in the top-right corner of the page.
3. Click the **Change Password** link.
4. Enter your old password, new password, and confirm the new password.
5. Click **Save**.
   The new password is saved.

You have successfully changed your password.

Generate JWT Shared Secret for the Portal

If you install the CA TDM 4.2 Portal for the first time, it generates a 512-bit random shared secret key for JWT token generation. If you upgrade an existing installation of any version of the CA TDM portal to 4.2, it keeps your existing shared secret and does not generate a new one. If you require consistent operation of security users across multiple TDM Portals, ensure that they all use the same 'jwt.sharedSecret' value in the Portal's application.properties configuration file.

1. Open the `CA Test Data Manager Portal\service\bin` directory.
2. Execute the EncryptionUtil tool with the '-s' option to generate a new shared secret.

```
C:\Program Files\CA\CA Test Data Manager Portal\service\bin>EncryptionUtil.bat -s
generating key with keysize (bits): 512
jwt.sharedSecret=Iw8i77+977+9C0Z1Bu+/vQYfDu+/vXXvv70Q77+9Uxzvv73vv701Y++/vWrvv70qSe+/
vRzvv73vv71uMe+/ve+/vRvvv73vv73vv73vv71Rbmlb77+9Bho/77+9F3jvv71td++/vUbvv705O++/vQg=
```

3. Open the CA TDM Portal's `application.properties` configuration file in a text editor.
4. Copy and paste the `jwt.sharedSecret` property into the `application.properties` file and save it.
5. Restart the Portal service.

Set Up Passwordless Tester Access

As an administrator, I want to be able to configure the Test Data Manager Portal so that a special user named `tester` can access the Portal without providing a password.

To enable the tester autologin:

1. Open the CA TDM Portal. Click Configuration, Access Control, Users.
2. Create a user with name `tester` .
   a. Assign the appropriate `tester` role to the new user.
   b. Reset the password, for example to `Tester@123` .
   c. Encrypt the password and keep it in your clipboard.

For more information, see Use the Encryption Utility to Encrypt Passwords.

3. Stop the CA TDM Portal.
4. Go to the Portal installation directory, open the `conf` directory.
5. Open the `application.properties` file in a text editor and configure the following properties:
   - `tdmweb.tester.autologin.enable=true`
     Specifies whether the tester autologin is enabled (true). By default, it is disabled (false).
   - `tdmweb.tester.autologin.userid=tester`
     Defines the login name of the tester autologin account. Default: tester
   - `tdmweb.tester.autologin.password={cry}+rESco4uTvy28xgtOYCev6+NzJw6Hh6j7nxLnGM1Lkj0ZDkDM9Hv`
     Defines the password of the tester autologin account. Paste the encrypted password string that you have generated.
6. Restart the CA TDM Portal.

To use the tester autologin:

1. The user opens the CA TDM Portal at the following URL:
   `https://servername:port/TestDataManager/index.html#?user=tester`
2. CA TDM reads the password for the `tester` account from the `application.properties` file and authenticates the tester.

Location to Store User-Specific Data

When you install the CA TDM Portal, the installation creates a separate location to store user-specific data. This location is provided as a value to the environment variable `CATDMWEB_APPDATA` . The default value that is assigned to `CATDMWEB_APPDATA` is *CommonAppFolder*`/CA/CA Test Data Manager` (for example, `C:/ProgramData/CA/CA Test Data Manager`). This approach allows users to start the CA TDM Portal services even if they (users) do not have access to the files installed in the `..\Program Files\CA\CA Test Data Manager` location. However, users must have write permission on the location that is provided in the `CATDMWEB_APPDATA` environment variable.

After successful installation, the following folders are created in the *CommonAppFolder*`/CA/CA Test Data Manager` location by default:

- `logs`
- `orientdb`
- `tomcat`

When you start using the CA TDM Portal, additional folders are created in the *CommonAppFolder*`/CA/CA Test Data Manager` location; for example, `jobs` and `objects` .

An administrator can also change the default value of the environment variable. If you do so, ensure that you perform the following tasks so that all the required parameters point to the updated location:

Note: A new property (`tdmweb.appdata.folder)` is available in the `..\Program Files\CA\CA Test Data Manager Portal\conf\application.properties` file. The value of this property is set to the environment variable as `tdmweb.appdata.folder=${CATDMWEB_APPDATA}` .

- Change the path of the work directory and access logs in the `..\Program Files\CA\CA Test Data Manager \tomcat\conf\server.xml` file by changing the default value:

```
...
<Host name="localhost" appBase="webapps" unpackWARs="true" autoDeploy="true" workDir="C:\ProgramData\CA/CA
 Test Data Manager Portal/tomcat/work">
...
<Valve className="org.apache.catalina.valves.AccessLogValve" directory="C:\ProgramData\CA/CA Test Data
 Manager Portal/tomcat/logs" prefix="localhost_access_log" suffix=".txt" pattern="%h %l %u %t &amp;quot;
%r&amp;quot; %s %b"/>
```

```
...
```

- Change the path of the `java.io.tmpdir` property in the `..\Program Files\CA\CA Test Data Manager \service\conf\wrapper.conf` file by changing the default value:

  ```
  ...
  wrapper.java.additional.4=-Djava.io.tmpdir="C:/ProgramData/CA/CA Test Data Manager Portal/tomcat/temp"
  ...
  ```

  Note: Use double quotes if your folder name includes spaces.
- Change the OrientDB log location in the `..\Program Files\CA\CA Test Data Manager\orientdb\config \orientdb-server-log.properties` file by changing the default value:

  ```
  ...
  java.util.logging.FileHandler.pattern=C:/ProgramData/CA/CA Test Data Manager Portal/orientdb/log/orient-server.log
  ...
  ```
- In the new location, ensure that the `logs` folder and the `orientdb\log` folder are already present.

Note: The location of the `jobs` , `logs` , and `objects` folders is automatically updated when the value of the environment variable is changed. No explicit action is required from your side.

Manage Audit Logs

When you execute multiple repeated publish jobs, the logs in the CA TDM Service audit table can use up the available memory space quickly. By default, CA TDM Portal truncates some of the audit information to save space. Watch this page to be notified about updates to audit log management functionality of Test Data Manager.

Enable Full Audit Log

As a Test Data Engineer for CA TDM Portal, you can enable the full, untruncated audit log.

Follow these steps:

1. Navigate to the directory where you installed the CA TDM Portal, and open the `conf` subdirectory.
2. Locate and open the `application.properties` file in a text editor.
3. Configure the log level to control memory usage.
   `WebLogFullDetail = true|false`
   - **false** — The audit log does not include full details, which saves space. API request/response messages in the audit log are truncated to 30 characters. This is the default.
   - **true** — The audit log includes full details of the request/response messages.

     WARNING

     With full logging enabled, you may encounter issues with memory in the gtrep, as the log files can grow very big. As a Test Data Engineer for CA TDM Portal, set up a cleaning task on your database to regularly manage and clean table gtrep_web_log in gtrep. You will need to do this in accordance with your country's legal requirements.

Manage Portal Log Files

CA TDM Portal writes logs files and other content to a local folder under `C:\ProgramData\CA\CA Test Data Manager Portal\.`

To change the default log path, follow these steps:

1. Modify the value of the environment variable `CATDMWEB_APPDATA` and set it to your desired location, for example, `Z:\CA Test Data Manager Portal\.`
   Default: `C:\ProgramData\CA\CA Test Data Manager Portal\`
2. Change to your custom log directory in your file explorer, in this example, `Z:\CA Test Data Manager Portal\.`
3. Create an `"orientdb"` folder. Inside the folder, create a `"log"` folder.
   In this example, you create the following directory structure: `Z:\CA Test Data Manager Portal\orientdb\log`
4. Restart the Test Data Manager Service.
5. Restart the orientDB service.

Configure CA TDM Portal for Deleting the Purged Reservations

In CA TDM Portal, when you perform the model-based test data reservation, you can release the reservation. When you release the reservation, the reservation model keys are deleted. But the deleted reservation request remains in the purged state for a specified period of time. To identify whether the specified period of time is lapsed for the purged reservation requests, a job runs at a specified interval. You can configure the values of these parameters as per your enterprise requirement.

Follow these steps:

1. Navigate to the `TDM_HOME\conf` folder on the CA TDM installed server.
2. Open the `tdmdatareservation.properties` file in a text editor.
3. Search for the following parameters and modify as below:
   - **reservation_housekeeping_frequency_minutes = <xxx>**
     Specifies the interval time between each time the job runs to identify whether any purged reservations are older than the specified period to permanently delete them.
     Modify the value to change the interval time between each time the job runs. You must specify the value in minutes.
     Default: 72
   - **reservation_purge_timeout_days = <xxx>**
     Specifies the number of days for the reservations to remain in purged state before they are permanently deleted.
     Modify the value to change the number of days to keep the reservations in purged state. You must specify the value in days.
     Default: 30
4. Save the file and restart the CA TDM Portal Service.

CA TDM Portal Troubleshooting

If you installed the TDM Portal in the default location, the path is `C:\Program Files\CA\CA Test Data Manager Portal\`. If you installed the portal in a non-default location, adjust the paths on this page accordingly.

TDM portal seems to hang when I confirm large tables on the Data Model page

Symptom:

When I view the PII scan data of a large Data Model (around 2000 tables), and click the Confirm button on the data model page to confirm all tables, the Portal becomes unresponsive.

TDM Portal passes the table identifiers through a request header, and due to the large number of tables, the request header size surpasses the default size that is defined for the Apache Tomcat server. The server rejects such a large request and it the Portal becomes unresponsive.

Solution:

Increase the maxHttpHeaderSize setting in Tomcat's server.xml.

1. From Windows Explorer, open the file `C:\Program Files\CA\CA Test Data Manager Portal\tomcat\conf\server.xml` in a text editor.
2. To add the `maxHttpHeaderSize` attribute, do *one* of the following tasks:
   – If you are using HTTPS or port 8443, then add `maxHttpHeaderSize="3000000"` as attribute of the `<Connector port="8443">` element.
   Example:

```
<Connector port="8443" protocol="HTTP/1.1"
    SSLEnabled="true" scheme="https" secure="true"
    clientAuth="false" sslProtocol="TLSv1.2"
    keystoreFile="${tdmweb.keystorePath}"
    keystorePass="${tdmweb.keystorePassword}"
    keyAlias="${tdmweb.keyAlias}"
    ciphers="TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256, TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384,
 TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,
 TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,TLS_DHE_RSA_WITH_AES_256_GCM_SHA384,TLS_DHE_RSA_WITH_AES_128_GCM_SHA256"
    connectionTimeout="60000"
    maxHttpHeaderSize="3000000" />
```

   – If you are using HTTP or port 8080, then add `maxHttpHeaderSize="3000000"` as attribute of the `<Connector port="8080">` element.
   Example:

```
<Connector port="8080" protocol="HTTP/1.1"
    connectionTimeout="60000"
    maxHttpHeaderSize="3000000"/>
```

3. Restart the TDM Portal service.

Jobs artifacts not being downloaded in Portal

Symptom:

My jobs artifacts are not being downloaded in the CA TDM Portal.

Solution:

Verify whether you have multiple portal instances pointing to one repository (GTREP). If yes, create a separate GTREP database for each CA TDM Portal Installation, and reinstall the CA TDM Portal instances so that each has a 1:1 mapping to its own GTREP database.

Decrease High CPU Utilization

> **NOTE**
>
> It is normal behavior that CPU usage runs high for a few minutes when the TDM Portal starts while the system is initializing. After the initialization is complete, the system load returns to normal. Monitor the `startup.log` file to determine when initialization is complete.

Symptom

CPU runs at abnormally high levels even after initialization is complete.

Reason

Missing QRTZ* tables or lost connections to the Repository DB cause the TDM Portal to go into an infinite loop, which causes high CPU load.

Solution

Take the following diagnostic steps to end the loop:

1. Verify that the Repository DB has the QRTZ* tables.
2. If the QRTZ* tables are not present, uninstall and reinstall the TDM Portal.
3. If the QRTZ* tables are present, but the CPU is still high, restart the TDM Portal to re-establish connections to the DB.

Configure Resource Constraints for Oracle Repositories

Symptom

You observe the following error when you start or use the CA TDM Web Portal, or other CA TDM components:
`ORA-12516, TNS:listener could not find available handler with matching protocol stack`

Solution

Configure the system to limit the connections that are requested by the TDM Portal, and to allow more connections for Oracle databases.

Configure Oracle Resource Limits

1. Open a sqlplus window, and connect to your repository DB using a system account as *sysdba*.

   NOTE

   The default password that TDM uses for the system account is "manager", but you set the actual password when you install Oracle. If you do not have access to the system account, request that a DBA execute the statements.

   For example, for a local XE installation, you connect as follows:
   `SQL> CONNECT system@XE/manager as sysdba`
2. Run the following commands to verify the parameters values:
   ```
   show parameter processes
   show parameter session
   show parameter transactions
   ```
3. The following are the recommended values. If your values are less than the recommended values, use the following commands:
   ```
   alter system set processes=300 scope=spfile;
   alter system set sessions=300 scope=spfile;
   alter system set transactions=330 scope=spfile;
   ```
4. Save the file.
5. Restart Oracle and the TDM Portal.

If the same error reoccurs, work with the support team for assistance.

Unable to log in to the CA TDM Portal

Symptom

When I try to log in to the CA TDM Portal, I receive the following error in the Portal despite entering correct credentials:

```
Incorrect username and password
```

Also, when I review the
`%ProgramData%\CA\CA Test Data Manager Portal\logs\TdmWeb.log`
file, I find the following entry in the log file:

```
Could not write content: An attempt was made to write more data to the response headers
than there was room available in the buffer. Increase maxHttpHeaderSize on the connector
or write less data into the response headers
```

Solution

If you try to log in to the Portal that includes a large number of projects, you can receive this error and the log entry, despite providing correct credentials.

To address this issue, increase the maximum size of the request and response HTTP header (`maxHttpHeaderSize`) as follows:

1. Navigate to the `C:\Program Files\CA\CA Test Data Manager Portal\tomcat\conf` location.
2. Open the `server.xml` file in a text editor.
3. Add the `maxHttpHeaderSize` parameter to the following section:
   `<Connector port="8443" protocol="HTTP/1.1".....maxHttpHeaderSize="2000000"/>`
   The value
   `2000000`
   bytes is provided as an example. You can enter the value depending on your requirements.

   NOTE
   The default value of the request and response HTTP header is 8000 bytes.
4. Save your changes.
5. Restart the CA Test Data Manager Portal service.

Unable to Publish Data in the CA TDM Portal

Symptom

I am unable to publish the data in the CA TDM Portal.

Solution

One of the reasons could be that you do not have access to the appropriate security functions. Verify with your CA TDM Portal administrator about whether your user profile has privileges to publish the data.

Unable to Install the Portal After Manual Uninstall

Symptom

I uninstalled the CA TDM Portal by manually deleting the files. Now, when I am trying to install it again, I am unable to do so.

Solution

We recommend that you do not manually uninstall the CA TDM Portal. In the case of manual uninstall, it is possible that the Portal fails to uninstall properly, leaving behind a few registry entries or services. To correct this failure scenario:

- Ensure that Windows services has no CA TDM Portal service in it. If it exists, remove it.
- Ensure that no related keys are present in the registry entries. For example, if the following keys exist, the installer picks them to verify the old products. Therefore, delete them:
  HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\ComputerAssociates\CA Test Data Manager Portal
  HKEY_CURRENT_USER32\Software\Caphyon\Advanced Installer\LZMA\{<*ID*>}
  HKEY_LOCAL_MACHINE32\Software\Caphyon\Advanced Installer\LZMA\{<*ID*>}

Receiving Insufficient Privileges Error

Symptom

I am getting the following error in the CA TDM Portal:

```
Access Denied. You do not have sufficient permissions to access this content.
```

Solution

If a user receives this type of error message in the CA TDM Portal, it might be the case that the UI session of the user does not have the updated privileges to the resources. In such cases, users must log out from and then log in to the CA TDM Portal to get the updated privileges and continue their work. If you still receive this error, contact the CA TDM Portal administrator to understand whether you have sufficient privileges to access the resource.

Find Log Files

The log file location has been moved from the TDM Portal installation folder to the following ProgramData folder:

```
%ProgramData%\CA\CA Test Data Manager Portal\logs\
```

> **TIP**
>
> The easiest way to monitor the log files is to download a free utility, for example, BareTail.
> Another option is to use a Powershell script with the following properties:
>
> ```
> $APPROOT = $Env:CATDMWEB_APPDATA $WEBLOGS = "$APPROOT\logs"
> Get-Content -Path "$WEBLOGS\tdmweb.log" -Wait
> ```

Enable Debug Logging

If you are experiencing any issues with the CA TDM Portal, and want more information of the most probable cause, review the log files for details. You can set the log level to DEBUG. This can be very helpful to find out where things are going wrong.

The logs of all the Portal services running in the background are created as soon as the Portal is up and running. The Portal supports these log levels: TRACE, DEBUG, INFO, WARN, and ERROR. Out of these levels, the DEBUG level is best suited to get as much information as possible in the log files.

Follow these steps:

1. Navigate to the `C:\Program Files\CA\CA Test Data Manager Portal\conf` directory.
2. Open the `logback-tdm.xml` file in a text editor.
3. Find the commented-out log statement below the appender section.
4. Uncomment the statement to enable debug logs. Do one of the following:
   a. Enable all debug logs.
      `<logger name="com.ca.tdm" level="DEBUG" />`
   b. Modify the statement to log only a subset of the application.
      `<logger name="com.ca.tdm.jobengine" level="DEBUG" />`
5. Restart the CA Test Data Manager Portal service.

Monitor the JVM Activity of the TDM Portal

You can observe memory, threads, and other performance characteristics of the TDM Portal JVM. To monitor the JVM activity of the TDM Portal, use JMX to connect to the JVM.

To enable JMX, follow these steps:

1. Navigate to the `C:\Program Files\CA\CA Test Data Manager Portal\service\conf` folder.
2. Open the `wrapper.conf` file in a text editor.
3. Add the following lines to the Java Additional Parameters section:
   `wrapper.java.additional.7=-Dcom.sun.management.jmxremote`

```
wrapper.java.additional.8=-Dcom.sun.management.jmxremote.port=31417
wrapper.java.additional.9=-Dcom.sun.management.jmxremote.authenticate=false
wrapper.java.additional.10=-Dcom.sun.management.jmxremote.ssl=false
```

4. Adjust the numbered suffixes in the parameter names to match your parameter list.
5. Modify the port number as needed for your requirements.
6. Save the file.
7. Restart the TDM Portal service.
8. Use JConsole to connect on the port that you just specified as additional parameter.

Publish Data to Sybase IQ Database

To successfully publish data to Sybase IQ database from TDM Portal, ensure that you have modified the corresponding database configuration file (.cfg) in the Sybase IQ installed server for the following parameters:

- Modify the parameter **-c 48m** to **-c 64m**
- Modify the parameter **-gm 10** to **-gm 30**
- Add the parameter **-gn 45** at the end of the file

After modifying the configuration file (.cfg) of the corresponding database, restart the Sybase IQ database.

Submitted Requests in TDM Portal Waits for 30 Seconds to Start Running

Symptom

When I submit a request in CA TDM Portal, the status of the request remains as "Not Started" for 30 seconds. Despite refreshing the page, the status does not change to "Running" for 30 seconds.

Reason
This happens because the repository database has a Quartz Configuration of Idle Wait Time property set to 30000 milliseconds by default. Idle Wait Time is the amount of time in milliseconds that the scheduler will wait before picking up a request from the queue.

Solution
You can edit the Quartz Configuration to set the Idle Wait Time that suits your requirements.

Follow these steps:

1. Open the *quartz_oracle.properties* file or the *quartz_sqlserver.properties* file based on the repository database you are using. These files are typically available under the path *C:\Program Files\CA\CA Test Data Manager Portal\tomcat\webapps\TDMJobService\WEB-INF\classes*.
2. Find the parameter *org.quartz.scheduler.idleWaitTime=30000* in the file. If the parameter is not available add the same at the end of the file.
3. Modify the value 30000 to a lesser value that you want to set as Idle Wait Time in milliseconds. Save the File.
4. Restart the CA TDM Portal Service.
   Notes:
   a. Values less than 5000 milliseconds are not recommended as it will cause excessive database querying. Values less than 1000 are not legal.
   b. Any modifications to the Quartz configuration are not retained after upgrading the CA TDM Portal. You must modify the values manually after the upgrade.

   For more information see Quartz Configuration Reference documentation currently available at http://www.quartz-scheduler.org/documentation/quartz-2.1.x/configuration/ConfigMain.html.

Previous Version Artifacts Not Working in the Latest Version

Symptom

I uninstalled my existing CA TDM Portal instance and then installed the latest version. Now, my artifacts that I created in my previous version (uninstalled now) are not working in the latest version. For example, when I try to perform the export RR (request-response) pair operation in the CA TDM Portal 4.0 for the work done in the CA TDM Portal 3.8, I receive a NULL pointer exception.

Solution

If you uninstall the existing CA TDM Portal installation and install the latest version, the existing OrientDB database is also uninstalled and the one that comes with the latest CA TDM Portal version gets installed. This creates issues because all the artifacts created in the previous CA TDM Portal installation and stored in the associated OrientDB database are lost. These artifacts, therefore, no longer work in the latest installation, resulting in an additional effort for you to recreate them.

To overcome this issue, you can simply upgrade the CA TDM Portal. This way, you can avoid the unnecessary work of recreating the artifacts that you created for use in the previous release of the CA TDM Portal.

However, if you must uninstall the CA TDM Portal, ensure that you take a backup of the OrientDB database before uninstallation, so that you can preserve your work artifacts.

NOTE

More information:

- Uninstall Product Components
- Install TDM Portal for Windows

Connection Manager Service Failing (404 Not Found Error)

Symptom

When I try to perform a publish operation in the CA TDM Portal, I get the following error:

```
ERROR: Publish failed for job ***, Call to ConnectionManagerService
failed::HttpClientErrorException: 404 Not Found
```

Solution

Create the associated connection profiles in the CA TDM Portal. Additionally, associate them with the user group that is connected to the CA TDM Portal (when publishing the data).

Connection Manager Service Failing (403 Forbidden Error)

Symptom

When I try to perform a publish operation in the CA TDM Portal, I get the following error:

```
Call to ConnectionManagerService failed::HttpClientErrorException: 403 Forbidden
```

Solution

This error is linked to the fact that you did not authorize your user to use this connection profile. For this, you must associate your user group with the connection profile used by your user.

TCP/IP Connection to the Host Failed

Symptom

I am receiving the following error in the CA TDM Portal:

```
The TCP/IP connection to the host ***, port 1433 has failed.
Error: "connect timed out.
Verify the connection properties. Make sure that an instance of SQL Server is running on
the host and accepting TCP/IP connections at the port.
```

```
Make sure that TCP connections to the port are not blocked by a firewall.
```

How can I address this issue?

Solution

Check your Microsoft SQL server connection and Windows or network firewall.

If you are using SQL Server Express, then this error could be because this server uses dynamic ports by default. In this case, you can try to solve the error by following one of the following ways:

- Specify the instance and no ports in the CA TDM Portal configuration.
- Configure the static port in your SQL Server Express configuration manager instead of the dynamic ones.

Configure Telemetry

Test Data Manager includes the capability to store your usage information, and (in the case of customers who use TDM under the terms of a Broadcom Portfolio Licensing Agreement) to send this information to Broadcom.

> **WARNING**
>
> If you use TDM under the terms of a Broadcom Portfolio Licensing Agreement (PLA), this telemetry information is necessary for TDM to function. For more information, see Activate Test Data Manager.

The following sections of this page detail how you can configure the Telemetry service:

Configure Telemetry at product installation or upgrade

After you install or upgrade TDM Portal, the first time you log in as an Administrator, the **Activate Product** dialog opens. On the **Telemetry Configuration** page of this dialog, enter the following information:

- (PLA customers only) **Company Domain**The last part of your company's e-mail address (e.g. **broadcom.com**).
- **Enterprise Site ID**This is a 4 to 9 digit number. You can find this on your License Agreement or on the CA Support Portal.
- (Optional) **Internal Identifier**
  TDM reports some metrics per instance. You can set this value for your own reference, to track your organization's usage of TDM by instance.

For more information, see Activate TDM and Configure Telemetry.

Configure Telemetry in TDM Portal

To make changes to how TDM reports your Telemetry data, follow these steps:

1. Open the TDM Portal.
2. Click **Configuration**, then click **Telemetry**.
   The **Telemetry Configuration** page opens.
3. Amend your **Company Domain** (PLA customers only), **Enterprise Site ID** and **Internal Identifier** as necessary.
4. Click **Save**.
   TDM saves your new Telemetry preferences.

Export Telemetry Data

TDM always collects telemetry data, even if it does not send it to Broadcom (in the case of customers who do not use TDM under the terms of a PLA). TDM stores this telemetry data locally.

You can download all the usage information data that TDM collects, as a CSV file.

Follow these steps:

1. Open TDM Portal.
2. Click **Configuration**, then click **Telemetry**.
   The **Telemetry Configuration** page opens.
3. Click **Export Metrics**.
   TDM downloads your Telemetry Data as a JSON file.

Usage metrics that TDM collects

TDM Portal reports the following usage metrics:

- **Data Profiling, PII Data Discovery and Masking**
  - Number of user sessions involved in PII Data Discovery and Profiling activities per day.
  - Number of user sessions involved in creation or editing masking definitions, transformation maps and masking configuration options, per day.
  - Number of user sessions involved in running masking jobs per day.
- **Subsetting**
  - Number of user sessions involved in creation or editing subset definitions per day.
  - Number of user sessions involved in generating subset scripts per day.
- **Synthetic Data Generation**
  - Number of user sessions involved in creation or editing data generation definitions and data pools per day.
  - Number of user sessions involved in running data generation (publish) jobs per day.
- **Find & Reserve**
  - Number of user sessions involved in creation or editing of Find & Reserve model per day.
  - Number of user sessions involved in running Find & Reserve search per day.
- **Test Match**
  - Number of user sessions involved in creation or editing Test Match definitions per day.
  - Number of user sessions involved in Test Match execution per day.
- **Data sources**
  - Number of database instances and data source types in use in Masking, Subsetting, Synthetic Data Generation, Find & Reserve and Test Match functionalities, per day.
  - Information whether flat files are used as source or target in the TDM functionalities above.

(PLA customers only) Set up an HTTP Proxy for Telemetry

If you are unable to successfully send data, it may be because you need an HTTP proxy. You can define an HTTP proxy for telemetry in the `application.properties` file.

NOTE

In a default installation, TDM saves `application.properties` in the folder `C:\Program Files\CA\CA Test Data Manager Portal\conf`.

The lines in `application.properties` that define the HTTP proxy are the following:

```
# Proxy settings

# tdmweb.proxy.type:

# DIRECT - Represents a direct connection, or the absence of a proxy.
```

```
# HTTP - Represents proxy for high level protocols such as HTTP or FTP.

# SOCKS - Represents a SOCKS (V4 or V5) proxy.


tdmweb.proxy.type=DIRECT


tdmweb.proxy.hostname=localhost


tdmweb.proxy.port=8000


tdmweb.proxy.username=


tdmweb.proxy.password=
```

Notes on HTTP Proxy setup

- If
  `tdmweb.proxy.type=DIRECT`
  (i.e. no HTTP proxy is active), the values of properties
  `tdmweb.proxy.hostname`
  and
  `tdmweb.proxy.port`
  are not used. However, these properties must be present, and they must have values associated.
- The password (
  `tdmweb.proxy.password`
) that you enter for the HTTP proxy can be either:
  - **Plain text** (not recommended)
  - **Encrypted**

    NOTE

    CA TDM provides an encryption utility. For more information, see Use the Encryption Utility to Encrypt Passwords.
- Changes that you make to the
  `application.properties`
  file do not take effect until you restart TDM Portal.

Reporting Telemetry in Fast Data Masker and GT Subset

Fast Data Masker (FDM) and GT Subset send Telemetry data to the TDM Portal service, which aggregates Telemetry data from individual TDM components. If FDM or GT Subset are unable to connect to the TDM Portal service at startup, a warning message displays.

To set up this connection to the TDM Portal service successfully, it is necessary to define the operating system environment variable **TDM_PORTAL_URL** on the machine where FDM or GT Subset is installed. You should define this variable as the URL where you access the Portal.

Example:

TDM_PORTAL_URL=*https://TDM_Portal_Host:<port_number>*

NOTE

If TDM Portal is unavailable (e.g. it is an earlier version than FDM, or the service is inactive)**,** you may experience a delay at startup.

For information on how to disable this connection, see Disable Fast Data Masker's connection to TDM Portal.

Backup OrientDB databases

CA TDM Portal uses OrientDB to store information about your installation. We recommend that you backup these databases periodically, to minimize data loss in the case of software or hardware malfunction.

Backup from OrientDB command line

To backup each database from the OrientDB command line, you can follow the instructions at https://orientdb.com/docs/2.2.x/Backup-and-Restore.html.

However, we recommend that you follow the alternative backup instructions below (to backup while OrientDB is inactive), for the following reasons:

- To backup all databases from the OrientDB command line, it is necessary to open each database individually.
- TDM creates these OrientDB databases as they are required. Therefore, it is still necessary to check which OrientDB databases exist, when you want to backup these databases.

Backup while OrientDB is inactive

To backup all OrientDB databases at once, you can do so from the Windows command line. Follow these steps:

1. Open a command prompt window (run **cmd.exe** from the Start menu).
2. Enter the following command in the command prompt:

   ```
   net stop OrientDB
   ```

   The OrientDB service is inactive.

   WARNING

   Schedule backup of OrientDB databases for a time when TDM is not in use.
3. Make a copy of the whole OrientDB databases directory, located here in a standard installation:

   ```
   C:/ProgramData/CA/CA Test Data Manager Portal/orientdb/databases/
   ```

4. Enter the following command in the command prompt:

```
net start OrientDB
```

The OrientDB service is active.

You can copy the contents of the directory that you backed up, to the same directory in your new installation of CA TDM Portal.

Datamaker Administration

This section provides information about how to maintain and configure Datamaker and its sub-components.

Security

As an administrator, you use the Test Data Manager security features to manage users and user groups.

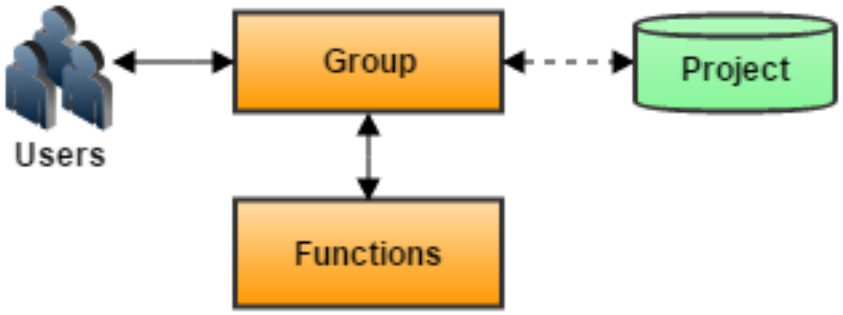
Note: To access the CA Test Data Management security menu, you must be logged in as an administrator.

The security model is based on groups and has the following features:

- To access projects and functions, users must be associated with a group.
- Each group belongs to one project and has associated functions.
- If a group is not associated with a project, the group is a Super Administrator.

The following diagram shows the security model:

Figure 17: Security Model



Groups and Users

Use the Test Data Manager security menu to manage users and groups to give users access to projects and functions.

Note: To access the Test Data Manager menus security menu, you must be logged in as an administrator.

Follow these steps:

1. In the administrator page, click the **Security** tab, and select **Users and Groups**.
   The Maintain Security login window opens.
2. Log in as Administrator
   The security window opens. The window has the following panels
   – **Left panel**
   Shows Groups and User levels.
   – **Right panel**
   Contains a series of tabs specific to your selection in the left Pane.

For more information about managing groups and users, see the Users sections.

Groups

Groups are the central component of the Test Data Manager security model. Group membership determines the projects and functions that users have access to.

Manage Groups

You can manage groups at the group and at the individual groups levels. Right click on a group in the Maintain Security left panel to open a drop-down with the following options:

- Add Group
- Edit Group
- Delete Group
- Copy Group

Add Groups

Add groups to connect to team projects and functions, and to provide access.

Follow these steps:

1. In the **Maintain Security** window, right-click a group in the left panel.
2. Select **Add Group** in the pop-up list.
   The **New User Group** configuration window opens
3. Complete the **Group Name** and **Group Description** fields.
4. From the drop-down list in the **Project** field, and select a Project**.**
5. Click the check mark icon.
6. Click **OK** in the New User Group pop-up.

The new group is created and is shows in the left pane. You can add users and can assign functions to the group.

To assign users, see Assign users to groups.

Edit Groups

You can edit the details of existing groups.

Follow these steps:

1. Right-click the group name in the left panel.
2. Edit the details, and click the green check mark icon.
3. Click **OK.**
   The changes to the group are saved.

Delete Groups

To remove a group from the system, delete the group.

Follow these steps:

1. Right-click the group you want to delete.
   The Delete Group pop-up opens.
2. Click **Yes** to delete the group. Click **No** to cancel and return to the admin window.
3. Click **OK** to verify the action.

The group is deleted from the system.

Copy Groups

You can copy existing groups to use templates to create new groups.

Follow these steps:

1. Right-click a group in the admin window.
2. Select **Copy Group** from the drop-down list.
   The Group window opens. The new user name is **Copy of group_<name of copied group>**
3. Edit the detail for the new group and click the check mark icon.
4. Click **OK**.
   A new group is created with the same properties as the copied group.

Assign Functions to a Group

Groups have no assigned functions by default. You can assign functions to groups either in the **Group Detail** page, or in the left panel of the **Maintain Security** window.

Follow these steps:

1. Click and highlight a group
   The **Available Functions** and **Granted Functions** panels open in the right panel.
2. Select a function from the **Available Functions** panel.
3. Click the move document down icon between the panels.
   The function is moved to the **Granted Functions** panel.
4. Click the save Changes icon.
   The changes are saved and the functions are assigned to the group.

To select and move multiple functions, use the normal CTRL, shift, or CTRL/A keyboard functions.

To remove functions form the Granted Functions pane, highlight the function and click the move document up icon.

Administrator Group

You can designate a group as an Admin Group with all functions granted.

Follow these steps:

1. Click and highlight a group in the left panel.
2. Click the **Admin Group** check box above the **Available Functions** panel.
   The **Available Functions** panel closes and the **Granted Functions** panel is populated with the functions available to the Administrator group.
3. Click the save icon above the **Granted Functions** panel and click **OK**.
   The changes are saved and the group is granted Administrator functions.

Tester Group

A group named *Tester* is available by default in the Datamaker UI. You can copy this group and assign it to your projects as required. You can also add users to this group. The following access functions are assigned to the Tester group:

- Tester Self-Service
- Publish Data
- Test Match

Users who are part of this Tester group can access only the aforementioned functions for associated projects.

Note: Existing TDoD uses can access the **Tester Self-Service** option in the CA TDM Portal if they are part of the Tester group for a project or they have access to the above-mentioned access functions through any other security group.

Users

After you create a group, create users and add the users to groups. Group membership give users access to projects and functions that are associated with the group.

Manage Users

To manage users, right-click a user in the left panel menu tree to open a drop-down with the following options:

- Add User
- Edit User
- Delete User
- Change Password
- Copy User

Add Users

To add users to groups and provide access to projects and functions, create new users.

Follow these steps:

1. Right-click **a user** in the left panel of the **Maintain Security** window.
2. Select **Add User** from the pop-up list.
   The **New User** window opens.
3. Complete following the fields in the **New User** window
   - **User Name**
     User ID used as login credentials
   - **Full Name** (Optional)
     Name of the user
   - **Password**
     User password that is used as login credentials
     - For then system to generate a Password, click **Generate Password**.
     - If the user enters an incorrect the password three times, the system checks the **Locked** box. The **Locked** box can only be unchecked by an administrator.
   - **Expire Password** (optional)
     Date that the use password expires. If you select this option, also specify a **Password Expiry Date.**
     Date Format: yyyy-mm-dd**Notes:**
     - If **Expire Password** is checked, the user must change the password on the first login.
     - If **Expire Password** is unchecked, the user will never have to change the password.
   - **Location** (Optional)
     Location of the user in your organization
   - **Extension** (Optional)
     Telephone extension of the user
   - **email**
     The user e-mail address
   - **Access End Date**
     Date that user access is terminated
   - **Cloud Trial**

4. Click the check mark icon to save the user.
5. Verify the user details in the pop-up window and click **OK**.
   The user is added to the **Users** list and can be assigned to groups. Users are not assigned to any groups by default.

Edit Users

1. Right-click the user name in the admin window and select **Edit User** from the drop-down list
   The **New user** pop-up opens.
2. Edit the user and click the check mark icon.
3. Click **OK.**

Delete Users

1. Right-click the user in the left panel and select **Delete User** from the drop-down list
   The **Delete user** pop-up opens
2. Click **Yes** to delete the user. Click **No** to cancel and return to the admin window.
3. Click **OK**.

Change User Password

1. Right-click the user in the left panel and select **Change Password** from the drop-down list.
   The Change Password for User:<user name> pop-up opens.
2. Enter and confirm the new password and click **OK**.
   Note: Click the question mark icon to the right of the password field to auto-generate a password.
3. Click **OK**.

Copy Users

You can copy existing users to use as templates to create new users.

Follow these steps:

1. Right-click the user in the admin window.
2. Select **Copy a User** from the drop-down list.
   The New User window opens. The new user name is **Copy of user_<**name of user that was copied**>**
3. Enter or generate a password and Edit the user details.
4. Click the check mark icon.and click **OK**.
   A new user is created with the same properties as the copied user.

Add Users to a Group

To give a user access to projects and function, add groups to the user profile.

Follow these steps:

1. Click on a user in the left panel of the Maintain Security window.
   The following panels open:
   Available
   List of available groups
   Member of
   List of groups that the user is a member of
2. Click and highlight the groups in the **Available** panel that you want the user to be a member of.
   Note: Use Ctrl-click and Shift-click to select multiple groups.
3. Click the move document down icon.

4. Click the save icon on the right.
5. Click **OK.**
   The user is assigned to the selected groups and has access to the associated projects and functions.

To remove groups from the user profile:

1. Click and highlight the groups.
2. Click the move document up icon.
3. Click the save icon and click **OK**.
   The groups are moved out of the user profile.

Active Directory Integration

Active Directory (AD) enables your security teams to authenticate and authorize Test Data Manager user access and privileges from a central location.

Enable Active Directory Integration in Datamaker

In Datamaker, click **Security, Users and Groups**, and open the **System Settings** tab, to define the **AD group** name.

> **TIP**
>
> Administrators can hide the CA TDM "Administrator" account when using Datamaker with AD or LDAP authentication. After an administrator enables **Settings, Hide Admin Access on AD Login**, AD users do not see an option to select an admin user when they log in.

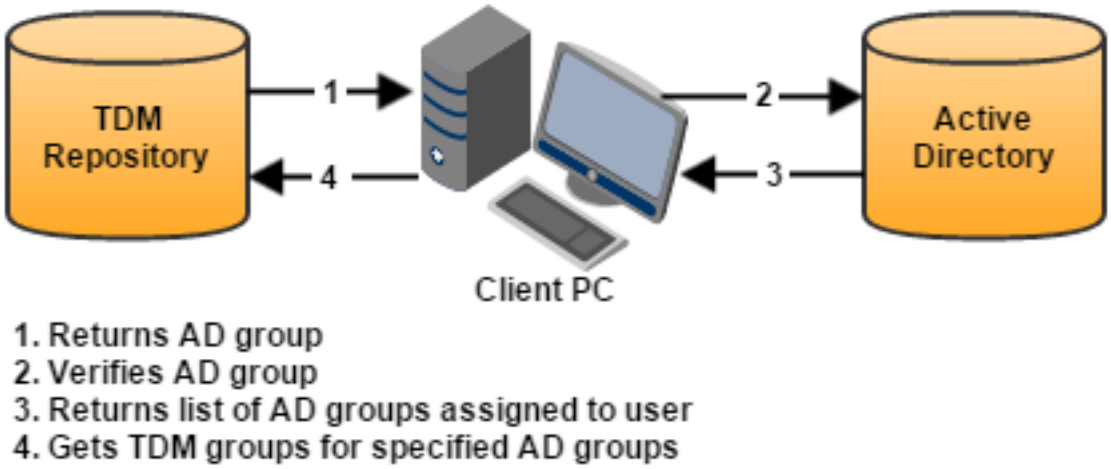
log on to"Test Data Manager using AD

To log in to the Test Data Manager repository, you need a controlling Active Directory (AD) group name. This name must be specified in the Test Data Manager security settings page as outlined above. AD Authentication consists of the following steps:

1. At start-up, the client gets the name of the controlling AD group from the repository.
2. The client checks that the AD group exists with the AD controller.
   Note: If the AD group specified does not exist, the Test Data Manager client denies access to the repository.
3. When the AD that is specified is verified, the client retrieves a list of AD groups to which the user is assigned.
   User membership in the specified AD group is verified.

The following diagram shows the Active Directory configuration:

Figure 18: Active Directory Configuration



Activate Active Directory Integration

To enable Active Directory authentication, activate AD integration.

Follow these steps:

1. Find the full Active Directory (AD) domain name.
   Note: Your AD administrator can provide the AD name, or you can also run the command whoami/UPN in a command line.
2. Make sure that the Test Data Manager users defined in the security screen match AD usernames. For example, if the AD username = ankur@tdm.com or int\ankur, the Test Data Manager security screen username = Ankur.
3. The Test Data Manager administrator must provide ALL ADMIN privileges to the AD user in the Test Data Manager security screen.
   Notes:
   – AD administrator must create a dedicated AD group, for example, GT_DM_ACCESS, and must add all Test Data Manager users directly. You cannot use indirect membership through another AD group.
   – Individual Test Data Manager users can use the *username*/*domain* command to confirm membership in the required AD group.
4. Start Datamaker. Because your username is populated in Datamaker, you enter only your AD password.
5. If you are using TDoD (Test Data on Demand), open the TDoD configuration editor. Set authentication type = AD and domain = domain from step 1. Save and restart TDoD.
6. If you are using the Remote Engine, open the Remote Engine configuration engine. Set AD domain = domain from Step 1. Now save and restart the Remote Engine service.

If you cannot access Datamaker, you can revert the integration.

Follow these steps:

1. In an SQL Window in another application, log in with the repository user name and password.
2. Run the following commands:

```
Delete from gtrep_clob where clob_id < 0;
Commit;
```

3. Restart Datamaker.
   You can use Administrator credentials and other Test Data Manager credentials to log in.

Licensee Administration

The information and functionality on the Licensees Tab has been deprecated. It will be removed in a later release.

Security Functions

The Test Data Manager security functions are found in the Functions tab on the Maintain Security window. The following security functions are available:

Initial Windows

- **Initial Windows**
  Windows accessible at start-up before security checked.

Unsecured Windows

- **Unsecured Windows**
  Windows always accessible when required.

Connections

- **Test Data Repository Connection**
  Windows required to alter the Test Data Repository connection.
- **Test Data Repository SQL Window**
  SQL Window for the Test Data Repository connection
- **Data Source Connection**
  Windows required to alter the Data Source connection
- **Data Source SQL Window**
  SQL Window for the Data Source connection
- **Data Target Connection**
  Windows required to alter the Data Target connection
- **Data Target SQL Window**
  SQL Window for the Data Target connections

Datamaker

- **Check Relationships**
  Check Relationships against data
- **Generate Relationships**
  Generate Relationships between tables
- **Data Definition**
  Define data within a Test Case
- **Data Design**
  Data Design
- **CA Agile Requirements Designer**
  CA Agile Requirements Designer
- **Maintain Object Tags**
  Maintain Tags for grouping Registered Tables and Columns together
- **View Publish Logs**
  View logs for Publish Jobs
- **Publish Data**

Publish a Publish Level of Test Cases to File or Database

- **Maintain Permitted Values**
  Maintain a list of Permitted Values for a Column
- **Actions on Registered Tables**
  Allows actions on Registered Tables
- **Register Tables**
  Store details of selected tables, columns, indexes, and foreign keys in the Test Data Repository

Policy

- **Maintain Policy**
  Maintain Selection Policy for Archiving

TDMWeb

- **Tester Self Service**
  Allows access to the Tester Self-Service functionality in the CA TDM Portal. This security function is applicable only for the CA TDM Portal.

Project

- **Add Links to All Levels From Publish**
  Create a link between a variable group and any data hierarchy object below the publish level (Publish Level) within a Project.
- **Add Link to Project**
  Create a link between a variable group and a Project.
- **Add Link to Publish Level**
  Create a link between a variable group and a publish level (Publish Level) data hierarchy object within a Project.
- **Copy Object**
  Copy data hierarchy objects within a Project.
- **Delete Object**
  Delete data hierarchy objects from a Project.
- **Delete Project Link**
  Delete the link between a variable group and a Project.
- **Edit Object**
  Create and modify data hierarchy objects within a Project.
- **Maintain Project**
  Create, modify and delete Projects and versions.
- **Project Manager**
  Select current context.
- **Repair Object**
  Repair the selected data hierarchy object.
- **Upgrade Object**
  Upgrade the selected data hierarchy object.
- **Verify Object**
  Verify that the selected data hierarchy object is valid.
- **Subset**
  Work with data subsets.
- **Transformation Maps**
  Transformation maps for columns that contain sensitive information.
- **Advanced Layouts**

Manage advanced file layouts.

Security

- **Users and Groups**
  Maintain Groups and the functions that the groups can access, and Users and the Groups they are in
- **Create XML REP Profile**
  Create a rep.xml file to control logging in to the Test Data Repository
- **Authorize Remote Publish**
  Authorize submitted Publish Jobs

Settings

- **Set Support Email Address**
  Set the address to which Support Emails are sent
- **Settings**
  Settings that affect the behavior of Test Data Manager

Utilities

- **Data Tools**
  Compare data in tables.
- **Oracle**
  Oracle utilities.
- **Attach File**
  Attach a file.
- **Reconcile Tables**
  Reconcile table and index definitions.
- **All Pairs**
  Insert data using "All Pairs."
- **Reference Data**
  Maintain reference data.
- **Data Functions**
  Maintain data functions.
- **Repository Maintenance**
  Maintain repository.
- **Test Match**
  Test Matching
- **Data Profiler**
  Data Profiler
- **ER Diagrammer**
  Diagrammer
- **ALM Integration**
  Integration with ALM.
- **REST**
  Integration with REST.

Internal

- **Internal**
  Internal

Authentication Event Logs

The following authentication events are logged in the Test Data Manager repository in the noted tables:

- All user and system authentication attempts:
  GTREP_SU_LOG
- All successful and unsuccessful authentication events (log in and log out):
  GTREP_SU_LO

Authorize Publish Jobs

When a remote publish job is requested, Test Data Manager submits a request for remote publishing authorization. This process lets administrators control remote publishing privileges.

Configure Data Subset

Set Row Fetch Size

Large data retrievals require large amounts of memory and long execution times. For this reason, Data Subset defaults to a maximum of 40 rows for each result set. You can change this value to set the maximum rows. When the SQL query is re-executed, the maximum number of result set rows that are returned equal the value entered.

SQL Tuning Options

This facility is designed for user-defined options when you generate Oracle scripts.

Use Selected Schema for Rules

This option qualifies table names in your rules file with the schema from the **Select Schema** drop-down list in the main design window.

Setting the Default Rule File

The Data Subset default is to use a rules file with the same name as the current connection profile name. This file contains the rule definitions in XML format. A profile connection named user01 Local Data Subset searches for a file that is named user01_Local.xml to use for the rule definitions. The search is in the same directory as DataSubset.exe . You can override this behavior.

Follow these steps:

1. Go to **Configuration**, **Set Rule File Location**.
2. Check the **Use Default** button to use the default rules file. This file has the same name as the connection profile.
3. To use a different rules file, click the **Always Use File** button and browse to a rules file.
   The rules file that is used shows in the status bar at the bottom of the **Data Subset** screen.

Change Rule File Object Schema

You can use an existing rule file against another schema. Change an existing rules file to incorporate your new schema names. The same objects exist in the alternative schema.

Follow these steps:

1. Go to **Configuration**, **Set Rule Schema**.
2. Browse to an existing rules file name.
   The unique list of schema names that are used in the rules file is displayed.

3. Enter the new schema names to use.
4. To save the new rules file with a different file name, change the **Save As** file name.
5. To use this new rules file, select the **Set as default rules file** box.
   The saved file substitutes old schema names with new names for qualifying objects (tables and views) and joining table SQL.

Allow Same Rule Multiple Times

When you design subsets, Data Subset only displays a relationship between two tables in the tree once. For example, consider a relationship between Customer and Orders in the tree. When the Orders tree node (Orders back to Customers) is expanded, Data Subset does not display the rule in the opposite direction. To view all relationships at a given level, choose this option.

Exclude Foreign Key Relationships

If your source schema contains no foreign key relationships, and all table joins are user-defined, select this option. This option prevenst Data Subset from searching for foreign key relationships when tree nodes are expanded in the extract designer.

Retrieve Views for Selected Schema

By default, Data Subset does not display views or make views available for use. To display and enable Views in Data Subset, register the views that you want to use in Test Data Manager. Then set the Transformation Map against the view, not against the table. A large schema such as Oracle E-Business suite has many views and tables. For connections to a large schema, set configuration values so that Data Subset retrieves only tables on application start-up and on schema change.

Follow these steps

1. Select **Configuration**, **Retrieve Views for Selected Schemas**.
2. Check **Retrieve Views**, and click **OK** if you to see and mask Views. The View is visible in Data Subset. The masking update script works against the View and updates the underlying table.

Note: Enabled Views retrieve both tables and views, which slows the application start-up.

Retrieve Data on Tree Select

By default, when a tree node is selected, the data for that object is retrieved based on the data relationships in the tree. Designing extracts against a database with large data volumes is time consuming and unnecessary.

To turn off dynamic data retrieval, follow these steps:

1. Choose **Configuration**, **Retrieve data on tree selec**t.
2. Clear the check box in the resulting dialog.

Set Extract Directory

Use this option to set the default directory where generated scripts are saved. If this parameter is not set, scripts are placed in the same directory as the extract definition file (*.ext).

Set Color for Row Select

For dynamic data retrieval, you can select specific rows from data in the driving table. To display selected rows, change the default color.

Configure the Remote Publish Engine

The Remote Publish Engine is a 32-bit Windows service that can process remote jobs that are submitted through Datamaker, Test Data On Demand, and other TDoD service clients like CA Agile Requirements Designer. The Remote Publish Engine installation supports 64-bit OS versions of Windows (7 Professional, 8 Professional, 2008 R2 and 2012).

If you kick off a publish from Datamaker without a setting of immediate or kick off publish activities from other tools, the Remote Publish Engine executes the jobs. Jobs are picked up in such a manner that their **authorization** status and their **dependency on other jobs** is respected.

Introduction

Currently, the Remote Publish Engine supports processing the following job types:

- PUBLISH (Regular and Subset)
- TESTMATCH
- ALM job types
- Group jobs which can be a combination of PUBLISH, TESTMATCH, and ALM job types

Each type of job has its own executor. For example, Datamaker executes PUBLISH jobs, and Test Matching executes TESTMATCH jobs. When a job completes or triggers errors, a notification email containing a zip file is sent to the user who submitted the job. The zip file contains the job xml, logs and job output files.

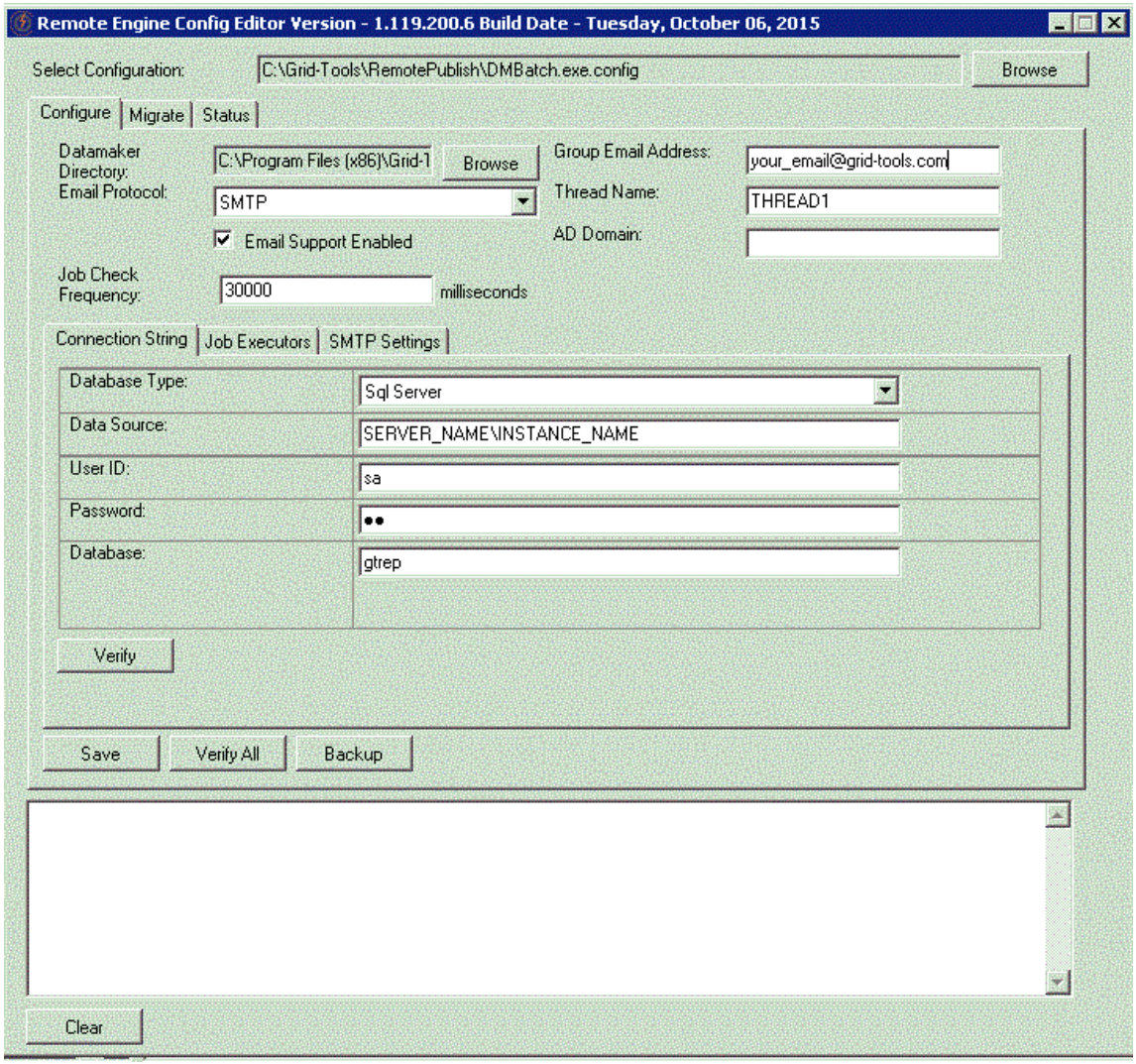
Configure the Remote Publish Engine

The Remote Publish Engine ships with a configuration editor where you configure all required settings.

> **NOTE**
>
> The Remote Publish Engine Windows Service must be configured to run under a real user account instead of a local system account. For Windows, to configure the Remote Publish Engine to run under a real user account, navigate to **Control Panel**, **Administrative Tools**, **Services**, **CA Remote Publish**, **Properties**, **Log On**. Specify a user account and password and restart the service.

Follow these steps:

1. Launch the Configuration Editor from C:\Grid-Tools\RemotePublish\RemotePublish_ConfigEditor \RemotePublishConfiguration.exe.
   If not populated by default, use the **Browse** button in the Configuration Editor to open the remote engine configuration file at C:\Grid-Tools\RemotePublish\DMBatch.exe.config.



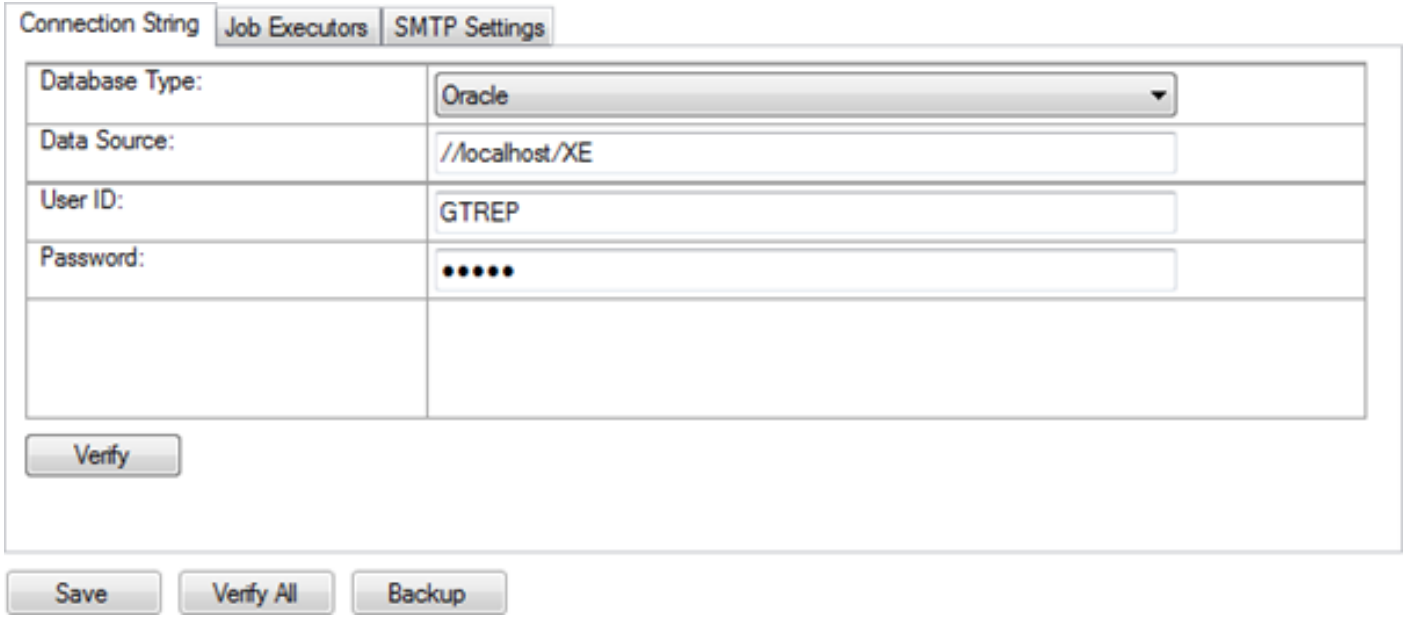
2. Populate the settings in the Configure tab as follows:
   - **Datamaker Directory**
     This setting stores the path of the Datamaker directory. If the path listed is incorrect, click the **Browse** button to navigate and store the correct path.
   - **Email Protocol**
     A list of email protocols that can be used for the mailing of notifications. The selection determines if there will be a tab after **Job Executor** labeled **SMTP Settings** or **Exchange Settings**.
   - **Email Support Enabled**
     This setting enables email notifications when checked. If not, email notifications are stored at C:Grid-Tools \RemotePublish\emails.
   - **Group Email Address**
     This is an email address that also receives email notifications in addition to the email address supplied for the published jobs. Also, for awareness, notifications are sent to this email address when the CA Remote Publish service has been started or stopped.
   - **Thread Name**

The thread name is used as an identifier by the engine. Thread name can be a comma (,) separated list, The Remote Publish Engine launches one executor per thread name and all jobs having the same thread name are executed by executor for that thread. Jobs with thread name **ANY** are picked by any executor.

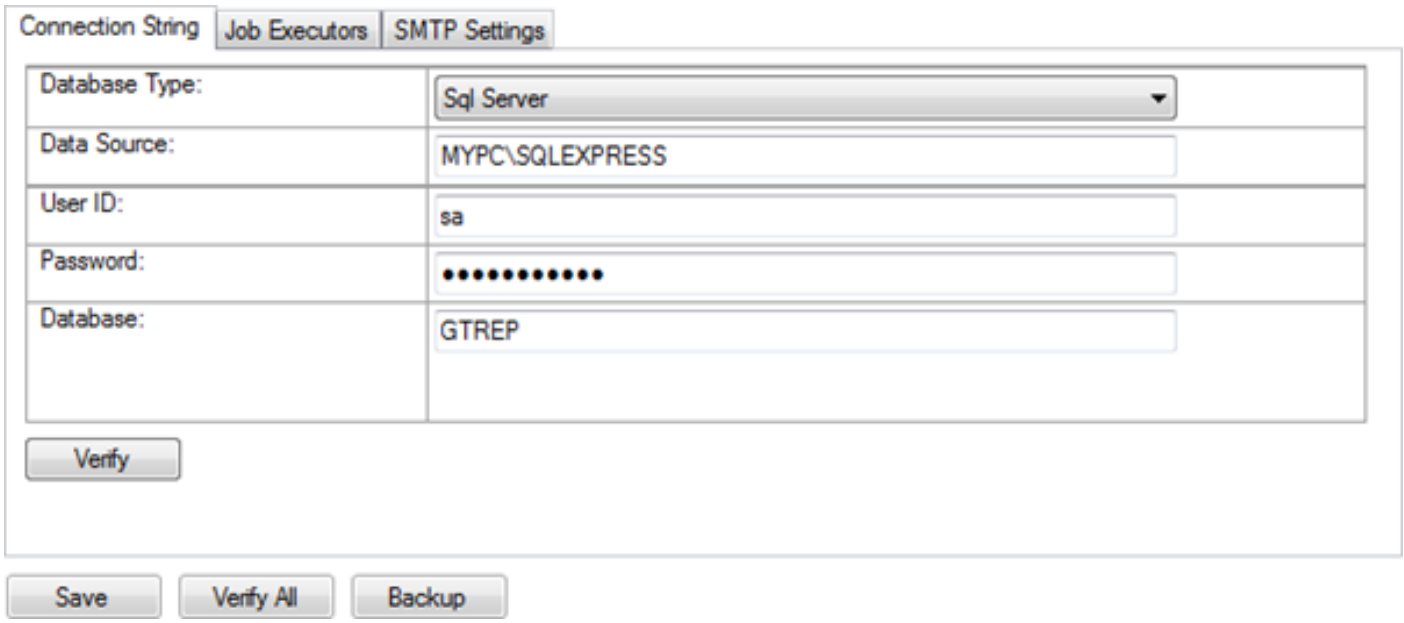
- **AD Domain**
  This field is populated with the Active Directory name, only, when the Datamaker license was issued for an Active Directory.
- **Job Check Frequency**
  Enter a value in this field to check for the new jobs at specified interval in milliseconds.

3. Populate the settings in the **Connection String** tab as follows:
   - **Database Type**
     You have the option of either Oracle or SQL Server. Your selection is based on where the repository database was installed.
   - **Data Source**
     If the repository database is installed on SQL Server, then this is set to the fully qualified SQL Server instance name.
     If the repository database is installed on Oracle, this is set to the TNS_ALIAS specified in the tnsnames.ora file.
   - **User ID**
     A user that can connect to the repository database.
   - **Password**
     The password of the specified user.
   - **Verify**
     Used to verify the supplied connection details.

ORACLE EXAMPLE



SQL SERVER EXAMPLE



4. Populate the settings in the **Job Executors** tab as follows:
   - **Enabled**
     Allows the option of enabling or disabling jobs from executing.
   - **Name**
     The name that is shown in both the logs and email notifications.
   - **Directory**
     Stores the full path of where the **Job Executor** executable is located.
     Example: C:\Program Files (x86)\Grid-Tools\GTDatamaker for gtdatamaker.exe
   - **ExeName**
     The name of the executable file.
     Example: gtdatamaker.exe, testmatch.exe
   - **JobName**
     The name of the Job.
     Example: PUBLISH, TESTMATCH, ALM
   - **Verify**
     Used to verify any corrections or changes that are made to listed jobs.
5. Populate the settings in the **SMTP Settings** tab as follows:
   - **Host**
     SMTP Host.
   - **Port**
     SMTP Port.
   - **Enable SSL**
     Check this setting if SSL is enabled on the host.
   - **Username**
     Depending on the configuration of the SMTP server, this setting is the authentication email address.

     NOTE

     If you are using a gmail account, the following settings prevent the error message "The SMTP server requires a secure connection or the client was not authenticated. The server response was: 5.5.1 Authentication Required".

     Follow these steps:

- Login to the Google account.
- Go to My Account page.
- Click Sign-in & security.
- Go to the Connected apps & sites section.
- Turn on the Allow less secure apps setting.

– **Password**
The authentication email account password.
– **Auth Domain**
The Domain which authentication is performed.
– **Use Default Credentials**
Choose this if using local IIS.
– **From Address**
Depending on the configuration of the SMTP server, this field can be configured in one of two ways. If the SMTP server requires a username/password, this field can remain blank. If the SMTP server does not require a username/password, this is populated with an email address who is viewed as the sender.
– **Display Name**
See **From Address** on whether this is populated with the name of the sender.
– **Reply To Name**
Username that is associated to the reply email.
– **Reply To Address**
Email address that is used to reply to.
– **Timeout**
This value is the duration that the program tries to send an email before exiting.
– **Verify**
Used to verify the supplied email details. A test email is sent for confirmation.

6. Populate the settings in the **Exchange Settings** tab as follows:
   – **Domain**
   An optional parameter to specify the domain name.
   – **Username**
   The email address from which the email notification is sent from.
   – **Password**
   The email account password.
   – **Auto Discover URL**
   Select this checkbox if the Exchange server supports Auto URL Discovery.
   – **Exchange Service URL**
   If the Exchange server does not support Auto URL Discovery and you have not selected the **Auto Discover URL** checkbox, provide the Exchange asmx service URL.
   – **Verify**
   Used to verify the supplied email details. A test email is sent for confirmation.
7. Do any of the following to complete the configuration:
   – Click Verify All to verify all changes that you made.
   – Click Save to save the configuration
   – (Optional) Click Backup to create a backup of the most currently saved configuration.

Migrate Configuration Settings

To import a backup of your Remote Publish Engine settings, click the **Browse** button and navigate to the location where the backup is stored. This usually can be found at C:\Grid-Tools\RemotePublish\DMBatch.exe.config.bak.

After selecting the appropriate file, click the **Open** button from the file browser dialog box. The settings have been loaded into the editor for review. If the settings are correct, click the **Save** button found in the **Configure** tab.

Manage Service Status

From the **Status** tab in the Configuration Editor, you can manage the service as follows:

Get Service Status

Gets Current Service Status whether it is running or stopped.

Stop Service

Stops the Remote Publish Engine Service.

Start Service

Starts the Remote Publish Engine Service.

Restart Service

This setting stops and restarts the Remote Publish Engine Service.

Relevant messages are shown in the message area of the editor before and after completion of the operation performed. If there is no message within 2 minutes, check the logs for any errors at C:\Grid-Tools\RemotePublish\logs\dmbatch.log.

The Remote Publish Engine Windows Service must be configured to run under a real user account rather than a local system account. This configuration can be carried out from Control Panel, Administrative Tools, Services, CA Remote Publish, Properties, Log On. Specifying a user account and password and then restart the service.

Use the Encryption Utility to Encrypt Passwords

The CA TDM Portal provides an encryption utility (EncryptionUtil.bat) that lets you generate encrypted passwords. If you want to update an existing password in any configuration file because of changes in your environment or password policy, run the utility to encrypt the password.

The encryption utility is available in the location where you install the CA TDM Portal. For example, C:\Program Files\CA \CA Test Data Manager\service\bin.

1. Open the command prompt, navigate to the location where the utility is available, and run the following command:
   ```
   cmd>EncryptionUtil.bat -p
   ```
2. Enter the password when prompted.
3. Re-enter the password to confirm.
   The utility encrypts the provided password and displays the encrypted value. The following snippet shows an example of the encrypted password:

   ```
   cmd>EncryptionUtil.bat -p
   Enter password:
   Re-enter password:
   Encrypted password: {cry}AAAAEP13ot80EIZAeGosYD7Wao0/rkMRYfo4gI2eubAXw1D
   ```

4. Use the encrypted value in the configuration file.

CA TDM Troubleshooting

This article includes information that can help you troubleshoot issues that you might face while working with CA TDM.

Datamaker

Mismatch Between Datamaker and Repository Versions

Symptom

After I upgrade Datamaker and open it, I receive the following error message:

```
Licence Package version (3.2D) differs from Test Data Repository version (3.2C) error message.
```

The steps that caused this error in my case are as follows:

1. Download CA TDM to a virtual machine and other individual local computers.
2. Execute the setup file (.exe) for upgrading the already installed CA TDM version on the virtual machine.
3. On the local computers, launch Datamaker.
   The following error message is displayed:
   `Licence Package version (3.2D) differs from Test Data Repository version (3.2C) error message.`

In this scenario, CA TDM is installed on the virtual machine where the repository database is installed. Other users have CA TDM installed on their local computers, but they use the repository located on the virtual machine to connect to Datamaker.

Solution

This error message occurs when the repository database is not in sync with, or not upgraded to the same version as, the repository connection you are trying to use on your local machine. From the error message, you can see that the repository is on a different version than the installation of CA TDM.

The error message mentions the License Package, and this refers to the file pk_gtrep_lic.tsq in your Datamaker directory folder (C:\Program Files (x86)\Grid-Tools\GTDatamaker). You can open this file by using an appropriate editor. When you search for "Repository version" in this file, you see a line with the repository version listed. You can then compare this version number to the one in the error message and see that the repository is out of sync with your new or current installation.

The repository database being out of sync with your connection after an upgrade can be caused by either an incomplete upgrade or not performing repository maintenance after upgrading.

You can resolve this issue by re-executing the CA TDM setup installer (in case of an incomplete upgrade) and performing repository maintenance to sync your repository with your newly installed version of CA TDM.

Before you start, verify that you have already performed the following tasks:

- Download the appropriate CA TDM release you are planning to upgrade to on the computer where you have your current CA TDM release installed.
- Note the user name and password of a user in the ADMIN group to perform an upgrade.
- Close all the CA TDM and CA Agile Requirements Designer applications and stop the TDoD and remote publish services before starting the upgrade.
- Review the upgrade documentation.

To re-execute the CA TDM setup installer, follow these steps:

1. Double-click the setup_GTServer_x.x.x.xx.exe file (for example, setup_GTServer_3.6.0.19.exe) to extract all the compressed .zip files.
2. Browse for a destination and extract the files.
3. Double-click the setup_GTServer_x.x.x.xx.exe file.
   The **GTServer Setup** dialog opens.
4. On the **Welcome to the Prerequisites Wizard** page, click the **Next** button.

Note: If you already attempted to upgrade once, you may be prompted with an extra window to repair or remove the CA TDM installation. Click the repair option.

5. Accept the license agreement and click **Next**.
6. On the **Prerequisites** dialog, leave everything checked as is and click **Next**.
7. For each prerequisite and product component, follow the specified prompts and finish the installation.

To finish upgrading the repository, follow these steps:

1. Double-click the Datamaker icon on your desktop to launch the Datamaker UI.
2. Log into Datamaker with your administrator credentials and connect to the repository.
   The following error is displayed:

   ```
   Test Data Repository version specified in database (3.1B) is invalid.
   This version of the software can work with Test Data Repository versions 3.2A and above only.
   In order to run the upgrade you will need to know the user name and password of a user in the ADMIN group.
   Would you like to attempt an upgrade?
   ```

   Note: Your database and repository versions might be different from the example error message.
3. Click **Yes**.
4. Enter the administrator user name and password, and click the green check mark in the bottom-right corner.
   The **CA Test Data Manager - Datamaker Repository Schema - Update Required** dialog opens.
5. Click **Yes**.
   A list of all your connection profiles is displayed.
6. Connect to a profile and click the green button in the bottom-left corner.
   You can now access Datamaker and its toolbar.
   Note: If you receive any error messages regarding the rep.xml, see Upgrade Product Components.
7. Perform the repository maintenance so that everything with your repository and current installation are in sync. For more information about how to perform repository maintenance, see Repository Administration.

After you complete the upgrade and perform the repository maintenance, the version mismatch error message is no longer displayed.

No Valid Project Assigned to the User

Symptom

When I try to log into Datamaker, I get the following error message:

```
FATAL ERROR: CA Test Data Manager-Datamaker User Logon

No valid project (with at least one version) has been assigned to this user!
```

Solution

This error message occurs when one of the user groups to which you have been assigned was incorrectly created or configured. This is why you are unable to log in successfully.

To resolve this issue, you can have another user create a new project and give you administrator rights to the project. This enables you to log in successfully, create sub-projects, and assign users to projects.

Unable to Connect to DB2 Through Datamaker

Symptom

I have access to a DB2 environment. However, when I am trying to connect to the DB2 database (source or target) through Datamaker, I am getting an error. I also tried copying the db2jcc.jar and db2jcc_license_cisuz.jar jar files to the C:\Program Files (x86)\Grid-Tools\GTDatamaker\lib folder without any success.

How can I connect Datamaker (which is installed on Windows computer) to DB2 (which is installed on a Unix computer)?

Solution

The steps that you followed are for a Java application (for example, Fast Data Masker). To connect to IBM DB2 on LUW, z/OS, or iSeries using ODBC, .NET, or JDBC, you need DLLs and JARs that a client application named IBM DB2 Connect (also known as IBM Data Server Client) provides.

Additionally, review the following considerations:

- IBM DB2 Connect used to come in two flavors: Personal Edition and Enterprise Edition. The Personal Edition is discontinued now.
- IBM DB2 Connect EE(Enterprise Edition) is a marketing name, but when it comes to downloading and installing the software, it is called IBM Data Server Client.
- IBM Data Server Client needs to be licensed if you are connecting to DB2 z/OS or iSeries (that is, AS400). For connecting to DB2 on LUW (including AIX), unlicensed version of IBM Data Server Client downloaded from the internet serves the purpose.
- Licensing of IBM Data Server Client to connect to DB2 z/OS or iSeries means two things:
  - If you are using ODBC- or .NET-based application (for example, Datamaker or TDoD service), get the paid license file db2consv_ee.lic and apply the license using the `db2licm -a db2consv_ee.lic` command.
  - If you are using a JDBC application, include the licensed file db2jcc_license_cisuz.jar in the CLASSPATH. For example, for GTSubset.exe, include db2jcc_license_cisuz.jar in the \lib folder. When you download the activation license from the IBM website, db2jcc_license_cisuz.jar is present in the same folder/zip file as db2consv_ee.lic.
- When a 64-bit IBM Data Server Client is installed on Windows, it gets installed in the C:\Program Files\IBM\SQLLIB folder. If you are asked to copy the JDBC drivers from IBM DB2 Connect, you can copy the requested JAR files from C:\Program Files\IBM\SQLLIB\java.
- Each IBM DB2 Connect EE SKU (for example, D58FILL) is a bundle of 25 authorized or floating licensed users. If you buy two D58FILL, it means you are buying license for 50 authorized users to connect to DB2 on LUW, z/OS, or iSeries.
- Find the full list of various versions and fix packs of IBM Data Server Client that are available on a given day at http://www-01.ibm.com/support/docview.wss?uid=swg27016878.

Unable to Use the 32-Bit Version of Datamaker

Symptom

I am unable to use the 32-bit version of Datamaker. Is there a 64-bit version available?

Solution

The CA TDM Portal is a 64-bit component that is now available for use. The Portal currently provides some of the functionality that Datamaker does, and new functionality is being added with each release. For example, the Portal has its own publish engine and enables capabilities such as Tester Self-Service, which the older Test Data on Demand (TDoD) interface used to provide.

For more information about the CA TDM Portal, see CA TDM Portal.

Appending Additional Records to the Same Data Pool

Symptom

I have imported data from a Microsoft Excel file into a data pool. I have additional data in another Microsoft Excel file (with the same format) that I want to append to the same data pool. How can I do this?

Solution

To append the additional records available in a second Micorsoft Excel file to the same data pool, follow these steps:

1. Rename the second Microsoft Excel file (with the additional data) so that the file name matches the first file name that you have successfully registered and imported.
2. Access Datamaker.

3. Register the second file with the same project. On the registration dialog, ensure the following:
   a. The name in the **Workbook Name** field matches the first file name.
   b. The **Import Any Data** option is selected.
   c. The **Register** option is unchecked.
4. Click the **Register/Import** button.
   Additional records in the second file are appended to the same data pool.

Note: For data in a CSV file, right-click on the data pool that has the initial data imported, click **Import External File Data**, and point to the CSV file. In this case, however, it is important that your CSV file does not contain the "Header" row with the column names. Otherwise, that will also get imported as a record in your data pool.

Datamaker Stops Responding During Data Import

Symptom

After I register tables in Datamaker and try to import the data into the data pool, Datamaker stops responding. The data is not imported into the data pool. Some of the tables have more than 25,000 records.

Solution

This is a performance issue with Datamaker. Datamaker 3.5 and prior releases cannot manage data import with tables that have over 25,000 records (approximately). The number of records Datamaker can manage depends on factors such as the size of the record, number of columns, use of data types, and so on. Therefore, you might be able to import more or less depending on these factors.

Although Datamaker cannot typically handle a scale of 25-30 thousand records because it is a 32-bit application, the CA TDM Portal can. The CA TDM Portal can handle a much larger data volume and is the recommended solution for such scenarios. To import the data that has tables with over 25,000 records, we recommend upgrading to Datamaker 3.8 (or later) or using the CA TDM Portal.

Licensing Violation Error

Symptom

When I try to launch Datamaker, I receive the following error message:

```
A licensing violation has occurred - the application will stop!
```

Solution

The licensing violation error is not because of any licensing issue. This error is about how you are accessing Datamaker. If the same user tries to use multiple instances of Datamaker at the same time, this error message is displayed.

To address this issue, ensure that you are using only one Datamaker instance at a time.

Create a Data Model and Audit PII Data

CA TDM contains data sampling and data modeling functionality that enables you, as a Test Data Engineer (TDE), to profile data within a project.

In CA TDM Portal, the **Data Model** section of the UI was previously known as Data Discovery and the **Audit PII Data** section of the UI was previously known as Data Profiling.

Create a Data Model in CA TDM Portal

The Data Discovery process (accessible via the **Data Model** section of the UI) scans your databases to identify tables and relationships to create a reusable Data Model.

Scan the Data Model and Mask sensitive data

You can run a PII Scan on this Data Model, and with the results of this scan, you can mask PII data

that you discover.

Expose your Data to testers

You can create further models from your masked data, to allow testers access to the data without the risk of exposure of sensitive data. See Create and Edit a Find & Reserve Model.

PII Audit in CA TDM Portal

PII Audit performs a PII Scan on your data sources, to identify sensitive data. The identification of such data is key to the mitigation of risk associated with data retention.

NOTE

For more information about Data Discovery and PII Audit, see the following sections:

With CA TDM Portal

- The Data Model in CA TDM Portal
- PII Audit Using CA TDM Portal

With CA TDM Components

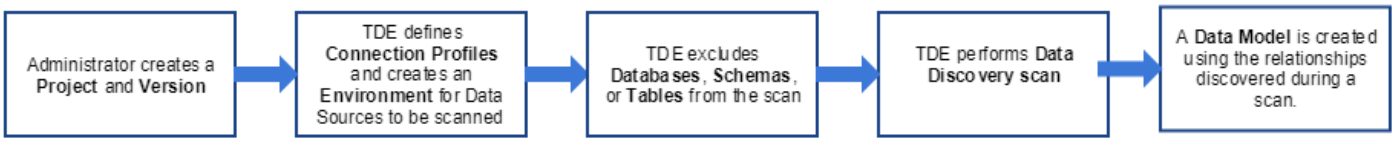
Data Discovery and Profiling Using Datamaker

The Data Model in CA TDM Portal

CA TDM uses its Data Discovery process to create a Data Model, from analysis of your environment.

A Data Model provides a solution to identify all table relationships across multiple data sources in an environment. After you have identified the table relationships, you can add new table relationships and edit existing table relationships. A Data Model is created using the relationships discovered during a scan. This Data Model allows you to visualize the relationships between table and data sources, so you can better understand data usage in your Environment.

Figure 19: Data Discovery flow



An Administrator logs in to CA TDM Portal and creates a project with a specific version for a Test Data Engineer (TDE) to perform Data Discovery.

A Data Discovery scan discovers table relationships based on the following critera:

1. Foreign key relationships between tables.
2. Column name matches along with common values within the tables.

After creation of the Data Model, you can add table relationships manually.

The Data Discovery process excludes some system entities by default, whose contents do not create meaningful data relationships. A full list of these exclusions is available at List of System Exclusions. The Data Discovery process excludes columns that only refer to the creation of data in the database ('Who' columns) from Primary Key relationship discovery - the Data Model only includes these columns' Foreign Key relationships.

Overview of Data Discovery process (creation of Data Model)

As a TDE, you need to perform the following steps in order to begin the Data Discovery process:

- Create Project and Versions.
  See Create and Edit Projects.
- Create Connection Profile(s).
  See Create and edit Connection Profiles
- Create an Environment.
  See Create an Environment.

For a detailed guide to the Data Discovery process for the creation of a Data Model, see End-to-End Scenario for Data Discovery.

After you create a Data Model, you can do the following:

- Perform a PII Scan on the Data Model.
  See Scan Data Model for PII.
- Use the results of the PII Scan to Mask data in the Data Model.
  See Mask Data with CA TDM Portal.
- Expose your Data Model to testers via the creation of Find & Reserve Models.
  See Create and Edit a Find & Reserve Model.

Data Model Terminology

- **Data Model** refers to CA TDM's model of the entities it discovers in your environments through the Data Discovery process.
- **Data Discovery** refers to the process of identifying tables and relationships to create a reusable Data Model.
- **Environment** refers to a list of connection profiles. For more information about connection profiles, see Create and edit Connection Profiles.

NOTE

Connection Profiles are shared between Data Discovery using CA TDM Portal and Test Data Reservation Service. For more information about how to use Connection Profiles with Test Data Reservation Service, see Configure Test Data Reservation Service.

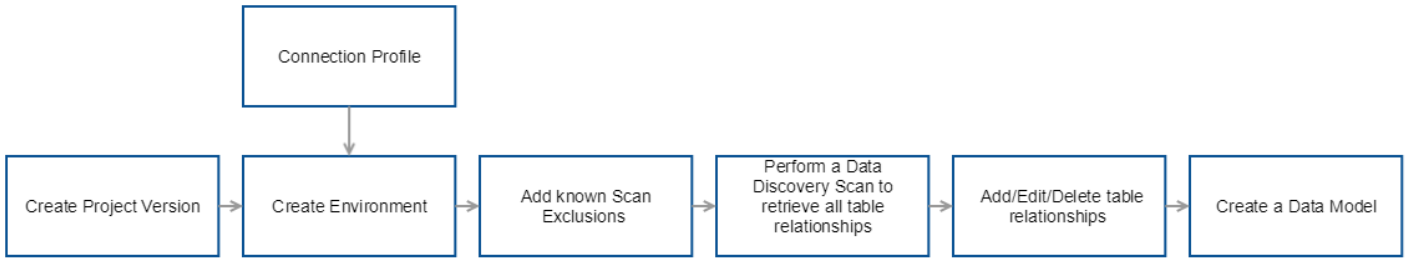
End-to-End Scenario for Data Discovery

CA TDM Data Model scans your Environment to identify tables and table relationships to create a reusable Data Model. You can set Scan Exclusions to ignore unnecessary tables, to tailor your Data Model to your needs. Within a Data Model, you can view an Entity-Relationship diagram for all tables in your environment, edit and add table relationships, perform a PII Scan to identify any PII data (see Scan Data Model for PII), and mask the PII data that you find.

The Data Model PII Scan process is similar to the PII Audit process, but you can use the results of the PII Scan to directly mask data in a Data Model. A guide to how you can do this is available at Mask Data with CA TDM Portal.

The Data Model End-to-End Scenario outlines the step-by-step process for a Test Data Engineer (TDE) to create a Generic Data Model. The basic flow to create a Data Model is as follows:

Figure 20: Data Discovery Detailed Architecture



Setting Up a Data Model

Before you can start with the Data Model creation process,you need to perform the following steps:

- Create Project and Versions.
  See Manage Project Versions.
- Create Connection Profile(s).
  See Create and edit Connection Profiles
- Create an Environment.
  See Create an Environment.

You are now ready to perform Data Discovery to create a Data Model.

Perform Data Discovery to create a Data Model

As a TDE, when you have an Environment ready with Connection profiles, you can create a Data Model.

Follow these steps:

1. Choose an Environment from which to create your Data Model. You can do this in one of two ways:
   a. Method One
      a. Click on the **Modeling, Environments** section of the Portal.
      b. Click on the Environment you want to use to create your Data Model.
      c. Click **Perform Data Discovery**.

The **Scan Options / Scan Exclusions** page of the **Data Model** section of the Portal opens.

b. Method Two
   a. Click on the **Modeling, Data Model** section of the Portal.
   b. Click **Get Started**.
      The **Select Environment** page opens.
   c. Select an Environment from the list in the left pane. Click **Next**.
      The **Scan Options / Scan Exclusions** page of the **Data Model** section of the Portal opens.

2. **Select Scan Level**
   You have the following options here:
   – **Basic - Scan Key relationships only (default)**
   The Data Discovery scans the environment for key relationships (i.e. primary and foreign key relationships defined in the data sources) only.

   NOTE

   This option is faster, but it does not discover relationships unless you define them in the data sources. You can add table relationships manually later - see Make Changes to Table Relationships for more information.

   – **Advanced - Scan Key and Column relationships**The Data Discovery scans the environment for key relationships (i.e. primary and foreign key relationships defined in the data sources), and also scans all columns within your environment for their relationships to each other across data sources.

3. **Select Scan Exclusions**
   Before you perform a Data Discovery scan, you can add a specific Database, Schema, or Table to scan exclusions. The scan excludes the specified Database, Schema, or Table from the Data Discovery scan and does not form part of the Data Model.

   NOTE

   By default, CA TDM already excludes some 'system' entities that do not contain relevant data, from relationship discovery. For more information, see List of System Exclusions.

   You can use basic wild card characters to add scan exclusions such as `*` (used to match zero or more characters) and `?` (used to match a single character).
   For example: for a table exclusion if you enter the value `*sys` , all tables in the environment that end in `sys` are excluded from the scan and no relationships are retrieved for them.

   NOTE

   If you change a scan exclusion on an existing model, you must re-scan the Data Model for the changes to take effect.

   For example, to remove a table from an existing Data Model, you can add the table as an exclusion and perform a re-scan.

4. After you add any necessary exclusions, click **Scan** to perform a Data Modeling scan.

   NOTE

   If you already have a Data Model, CA TDM prompts you to Confirm that you wish to overwrite it.

   The **Data Model** page opens. While the Data Discovery process is in progress, this page displays a summary.
5. When the process completes, the **Data Model** page displays your Data Model in the List View.

You can perform one or more of the following actions on the Data Model:

- Manage Table Relationships.
- Perform a PII Data Scan.
- After you identify PII Data, you can Mask Data.

Manage Table Relationships

After performing a Data Model scan, the Data Model screen opens. This firstly shows the List view, as a hierarchy of table relationships for all tables and columns in the selected environment. At the top-right of the screen, next to the text 'View' there are buttons to switch between List View, Entity Relationship diagram and PII Heatmap view.

List view

Tables that have associated table relationships are identified with a link icon. Expand the tree structure in the List view and select a table. All table relationships for the selected table appears in the right pane. If you click a column, this hides the relationships for all other columns. Clicking again on the table shows the relationships for all columns. This allows you to focus on one particular column without the distraction of all the others.

> **TIP**
>
> Click in the Search field to filter the list of schemas, tables and columns with the text you enter.

You can add, edit, or delete table relationships to adjust table relationships that were automatically discovered. To better understand table relationships, you can access the Details view for a specific table. You can assign tables and columns aliases within the Data Model view.

To perform actions on tables and columns, click the **Edit Relationships**, **Details** and **Edit Alias** buttons in the toolbar at the top of the screen in the Data Model view. These actions are also available from a context menu in the list view - when you hover over a table or column, an ellipsis symbol appears in the Actions column. Click the ellipsis to show these actions for that column or table.

> **TIP**
>
> Assign aliases to a column or table within the Data Model, to make them more accessible or memorable than their real names, as they appear in the database.

The following Actions are available from the Actions drop-down menu (from the **List View** and **Entity Relationship Diagram**):

- **Re-scan Data Model**
  Click here to create a new Data Model. This option overwrites the existing Data Model and its relationships.
- **Run/Re-run PII Scan**
  Click here to begin the PII Scan process. If you re-run the PII Scan, this scan overwrites any custom tags you added to columns in the data model.
- **Register Data Model Tables**
  Click here to registertables. You can use Registered tables for Synthetic Data Generation, and other tasks.

Table Details

To view details of a table's relationships to other tables by column, select a table in the list view and click **Details**.

In the Details view, you can:

- View all tables related to the selected table by column. In the left pane, all columns with at least one relationship are displayed in bold.
- Click a column in the selected table to specifically highlight relationships for that column in other tables. Click a table or column name from the left pane to get an overview of all the table relationships for the selected table. Click the down arrow to expand and view all columns in a table. You can expand only one table at a time. From here you can make changes to table relationships.

Register Data Model Tables

When you click **Register Data Model Tables** from the **Actions** menu, the Register Data Model Tables page opens, with a list of all tables in your Data Model. Click the tick icon to the left of a table name to select that table. Click Register to

register selected tables. When you Register tables, they appear on the **Objects** page under the **Modeling** section. This process is equivalent to the one described at Create and Register Derived Objects.

Tables with duplicate names

Tables with duplicate names from different databases can exist in your Data Model. In the list on the Register Data Model Tables page, these duplicate tables appear in **bold**. You cannot register tables with duplicate names. If you try to register multiple tables with identical names, no tables are registered and you receive the error message "Selected table names must be unique".

> **NOTE**
>
> If you register a table whose name is not unique, the Status of all tables with this name changes to **Registered**. The Differences column displays **Changes Found** for all tables whose contents in the database do not match the registered contents. The Differences column displays **No Changes Found** for the table you registered, unless you subsequently make changes to this table in the database. If you register another of the tables with the same name, this overwrites the originally registered table.

Entity Relationship diagram

Use the icons in the top-right corner to toggle between the List view and Entity-Relationship diagram for all table relationships in an environment. Circles represent tables in the environment, and links between them indicate that the at least one column connects the two tables. All table names are not visible at once in the entity-relationship diagram. When you hover over a circle, this displays the name and alias for the table that circle represents, and all paths to related tables are highlighted. Click a table to view the following details for that table:

- Data Source name
- Database name
- Schema
- Table name
- Alias
- Number of related tables
- Summary of table relationships

You can add, edit, or delete table relationships from the entity-relationship diagram for an Environment and view details for a specific table. For more information about how to add, edit, or delete table relationships, see Add/Edit/Delete Table Relationships.

All Data Sources are color coded in the entity-relationship diagram and listed under Data Model Details. You can filter the data sources that you want to be currently visible. After you select a table you can click **Details** to view the table Details for that specific table.

Make changes to Table Relationships

From the List View and Entity Relationships diagram, you can see and make changes to tables' relationships to other tables.

- From the List View, click a table in the left-hand pane and either click the **Edit Relationships** button in the top bar, or click on the ellipsis icon on the table's row in the list, and click **Edit Relationships**.
  The **Relationships** page opens. This lists all columns with relationships to columns in other tables.
- From the Entity Relationships diagram, click a table (a node on the diagram) and click the **Edit Relationships** button in the top bar.
  The **Relationships** page opens.

Add a Table Relationship

From the **Relationships** page, you can add relationships from the primary (selected) table, to columns in other tables.

Follow these steps:

1. Click **New Relationship** in the top bar.
   A new row appears at the bottom of the list of columns.
2. Select a Column from the drop-down list of columns in that table.
3. Select a **Related Table** from the drop down list of other tables.
4. Select a **Related Column** from the drop down list of that table's columns.

Edit a Table Relationship

From the **Relationships** page, you can edit relationships from the Primary (selected) table, to columns in other tables.

> **NOTE**
>
> You cannot edit table relationships of key columns.

Follow these steps:

1. For the Primary Column whose relationship you want to change, select the appropriate Related Table from the drop-down menu.
2. Select the appropriate Related Column from the drop-down (this contains all of the Related Table's columns).

Delete a Table Relationship

From the **Relationships** page, you can delete relationships from the Primary (selected) table, to columns in other tables.

- To delete a column's relationship, click the delete icon in the row of the Primary Column whose relationship you wish to delete
- To delete all relationships for a table, click **Delete All Relationships.**

Manage Aliases

Assign an alias to a database entity

Tables and columns can have aliases. When you hover over a database entity in the List View or Entity Relationship diagram, a tooltip displays the entity's real name (as it appears in the database), and its alias.

To assign an alias to a table or column, follow these steps:

1. Select a table or column in the List View (or Table Details pane), or a table in the Entity Relationship diagram, and click **Edit alias**.
   A dialog appears, with a field for the alias name. When a table or column has no alias, this field is blank.
2. Click **Save** to assign the alias, or **Cancel** to discard changes.

> **NOTE**
>
> Aliases for database entities are propagated to the Find & Reserve Model. Changes are not limited to the Data Model.

Display aliases or real names of database entities

When a database contains entities with aliases, you can choose whether to view these aliases or the entities' actual names in the list view.

In the Data Model view, there is a **Show Aliases** toggle under Display Settings in the right pane of the list view. If you switch this to Yes, entities' aliases appear, if the entities have aliases. When you hover over a database entity in the list view, a tooltip appears with the entity's real name (as it appears in the database), and its alias.

NOTE

More Information:

- Scan Data Model for PII
- Mask Data with CA TDM Portal

List of System Exclusions

By default, CA Test Data Manager excludes the following schemas and databases during Data Modeling:

Data Source	Excluded Schemas	Excluded Databases
DB2	• SYSCAT • SYSFUN • SYSIBM • SYSIBMADM • SYSIBMINTERNAL • SYSIBMTS • SYSPROC • SYSPUBLIC • SYSSTAT • SYSTOOLS	None
MySQL	• SYS • INFORMATION_SCHEMA • MYSQL • PERFORMANCE_SCHEMA • SAKILA	• SYS • INFORMATION_SCHEMA • MYSQL • PERFORMANCE_SCHEMA • SAKILA

Oracle	• ANONYMOUS • APEX_030200 • APEX_040200 • APEX_PUBLIC_USER • APPQOSSYS • AUDSYS • CTXSYS • DBSNMP • DIP • DVF • DVSYS • EXFSYS • FLOWS_FILES • GSMADMIN_INTERNAL • GSMCATUSER • GSMUSER • LBACSYS • MDDATA • MDSYS • MGMT_VIEW • OJVMSYS • OLAPSYS • ORACLE_OCM • ORDDATA • ORDPLUGINS • ORDSYS • OUTLN • OWBSYS • OWBSYS_AUDIT • SI_INFORMTN_SCHEMA • SPATIAL_CSW_ADMIN_USR • SPATIAL_WFS_ADMIN_USR • SYS • SYSBACKUP • SYSDG • SYSKM • SYSMAN • SYSTEM • WMSYS • XDB • XS$NULL • IX • OE	None
SQL Server	None	• master • model • msdb • tempdb

Scan Data Model for PII

You can run a PII Scan on your Data Model, to discover sensitive data in it. You can then use the results of this scan to produce a masking configuration and mask this sensitive data (see Mask Data with CA TDM Portal).

The PII Scan process is also part of the PII Audit process, but the PII Audit process only produces a report on sensitive data. With your Data Model, you can also mask this sensitive data.

The following PII data scan scenario outlines the step-by-step process for a Test Data Engineer (TDE) to identify any Personally Identifiable Information (PII) data in a Data Model.

Prerequisites

You need to create a Data Model in order to scan this model for PII. For more information, see End-to-End Scenario for Data Discovery.

Options for the PII Scan

Scan Level

You can choose to scan either:

- **Column names only (faster)**
  The PII Scan assigns tags based on comparison of column names, with Classifiers of the type 'column' (see Manage Data Classifiers).
- **Column names and data**
  The PII Scan assigns tags based on comparison of column names and column contents, with Classifiers of types 'column' and 'content' (see Manage Data Classifiers).

Column contents scan level

If you choose to scan 'Column names and data', you have a choice of how many rows to scan from each column. The Scan Level ranges from Basic to All based on the percentage of data you want to scan in your environment.

- **Basic:** Performs a PII data scan on 10 samples of data for each column in a table of your environment.
- **All:** Performs a PII data scan on all columns and rows for all tables in the selected environment.

NOTE

Running a scan at Scan Level **All** on an entire Data Source may take a long time.

(Optional) You can set the maximum number of rows to scan from each column with the **Max Rows** field.

NOTE

This maximum number overrides the value from the slider, only if it is **lower** than the value based on the slider.

Store Matched Samples

When you select **Store matched Samples**, CA TDM collects ten samples of data for each column in a table and stores it in the repository until the Internal Data Controller signs off the report. The collected data is deleted after the Internal Data Controller signs off and no record of the data is preserved in CA TDM. You can remove PII samples at any time from the Heat Map view of the Data Model. Click on **Actions**, and **Remove Matched Samples**, to permanently delete all matched samples.

Include or Exclude Connection Profiles, Schemas, and Tables

You can apply a filter to include or exclude Connection Profiles, Schemas, and Tables to reduce the size of your scan. You can use the basic wild card characters such as * (used to match one or more characters) and ? (used to match a single character) in the search terms to include or exclude any matching connection profile, schema, and table from the scan. For example, when you enter *sys in the tables to be excluded, the scan excludes all tables that end in 'sys'.

NOTE

You can either include a connection profile, schema, and table or exclude it from the scan but not both.

Perform a Data Model PII Scan

You can run a PII Scan on your Data Model to detect sensitive information in the Data Model.

Follow these steps:

1. Click on **Data Model** under the **Modeling** section of the Portal UI.
   The Data Model page opens with the List View active.
2. From the View options in the top-right, select the **Heatmap** icon.
   The Data Model page displays the Heatmap. The Heatmap displays a grid of squares, which represent the tables in the Data Model.
   These squares are blue before you run the PII Scan on your Data Model.
3. Click **Run PII Scan**.

   NOTE

   After the first execution of the PII Scan, this changes to an **Actions** drop-down, with options to **Re-scan the Data Model**, **Re-Run PII Scan** or downloadcsv.

   The Personally Identifiable Information (PII) Data Scanning page opens.
4. Select the Classifier Packs that you want the PII Scan to use to identify sensitive data.
   For more information on Classifiers, see Manage Data Classifiers.
5. Click **Next**.
6. Select the appropriate Scan Level.
7. (Optional) Check **Store Matched Samples** if you want to store matched samples.
8. Click **Next**.
9. Under **Scan Key Columns**, select whether to include Key columns (primary key and foreign key columns) in the PII scan. You can choose whether to include:
   - **String-based Key columns**
     By default, the PII scan includes String-based Key columns.
   - **Numeric-based Key columns**
     By default, the PII scan does not include Numeric-based Key columns.

     WARNING

     You may want to exclude key columns from your PII scan, because masking these columns may cause conflicts with database constraints. However, if you exclude key columns that contain sensitive data (for example, credit card numbers), **you risk the exposure of sensitive data**.

     For this case, you should download and use database constraints scripts (to disable and re-enable constraints before and after the mask job), and mask the data using Fast Data Masker. For more information, see Database Constraints Scripts.
10. Under **Include/Exclude Tables**, select one of the following:
    - **Scan All Tables**Scan executes on all tables in the data sources.
    - **Include / Exclude**'Add Filters' section appears below.
11. Under 'Add Filters', you can do the following:

- Click **New Filter** to add a filter.
- In the fields for each filter, select the appropriate Connection Profile, Schema, and as many Table names as you want. For each field, you can select a value from a dropdown list of existing values, or type your own value (this can include wildcards).

12. Click **Next** to confirm your selection.
    The PII Data Scan Execution page opens. This page lists your choices from the PII Scan process.
13. If you are happy with the details of the Scan to be run, select one of the following Schedule options:
    - **Now**When you click **Profile**, the PII Scan begins.
    - **Schedule** When you click **Profile,** CA TDM schedules the PII Scan to run at the time you specify.
14. Click **Profile** to begin or schedule the scan.
    The Data Model page opens. The page displays the progress of your PII Scan.
15. When the scan completes, the Data Model page opens with the Heatmap View. The Heatmap shows the results of the PII Scan. The colour of tables now represents their risk in terms of PII, according to how many tags the PII Scan identifies in each table (see table below).

Review Scan Results

When CA TDM completes a PII scan of your Data Model, it is available on the Heatmap View on the Data Model page. The Heat Map provides an instant graphical view to identify the total potential risk from PII data that exists within the scanned environment. The colours of the squares (i.e. tables) on the heat map indicates the following numbers of distinct tags present in each table:

Colour	Distinct tags	Risk Level
Red	15+	Very High
Dark Orange	10 - 14	High
Light Orange	5 - 9	Medium
Yellow	1 - 4	Low
Green	0	Very Low

NOTE

Multiple occurences of a single tag within a table only count as one distinct tag. Multiple tags assigned to one column all count as distinct tags.

The top menu bar lists the total number of PII data found within the scanned environment and the number of tables that are marked as confirmed. Each square in the Heat Map represents a table in the Data Source. You can zoom into a specific section of the Heat map to better view the table details.

You can filter tables in the following two ways:

- **Filter Unscanned Tables:**Use the Unscanned tables toggle to view all unscanned tables.
- **Filter Search Tab**:
  Use the Filter tab to search for a table, column, tag, profiles, and schema in the Heat Map
- **Risk Slider**:
  Use the Risk slider to filter tables based on their Risk category

Filter tables

By search term

Type a search term into the **Filter search results** field, to view a drop down with all matches to that term for tables, columns, tags, connection profiles, and schemas. You can use the basic wild card characters such as ***** (used to match one or more characters) and **?** (used to match a single character) in the search terms.

Click one of these matches to make the filter active, and to redraw the Heat Map to only show tables that contain matches to active filters. Active filters are listed under the **Filter** section of the page. You can remove a filter by clicking the **X** next to it.

For example, type in 'CREDIT' to search for all entities that begin with 'CREDIT'. Matches for each type of entity are displayed in the drop down. Click on any of the matches within the drop down to activate that filter and redraw the Heat Map.

CA TDM applies filters with AND logic, therefore more filters results in fewer matches. For example, to search for a string that contains 'CUSTOMER' within the schemas containing 'ACCOUNT' or 'ACCOUNTS', enter the search term '*ACCOUNT*' and select the matching Schema filter from the drop down. Next enter the search term '*CUSTOMER', which shows results for all matches in the remaining tables, columns, tags and profiles. Matching is case insensitive, so you can get results as follows:

```
tables: 'LEGACY_ACCOUNT', 'Account', columns: 'ACCOUNT_ID', 'ACCOUNT_CUSTOMER',
'Active_Account'
```

By row count

You can also filter tables based on column size in rows, and re-draw the Heat Map. The column size range is between small and extra large (relative to the row count of tables across the database).

Use the Risk slider to filter tables based on their Risk category

Drag the edges of the slider over the Risk categories to adjust your selection and redraw the Heat Map for a more specific view of the potential PII data that are identified in the scanned environment. Depending on the number of distinct tags that are identified in a table, the tables are filtered and positioned in the Heat Map as follows:

Distinct tags	Risk Level
15+	Very High
10 - 14	High
5 - 9	Medium
1 - 4	Low
0	Very Low

Manually Review Data within Tables

You can review each table in the Heat Map and further investigate if the data identified as PII is correct. You can select multiple tables to **Confirm** them or to mark them as **Not PII**. Click **Clear Selection** to select no tables. If you change filters or the risk slider, this has the same effect as **Clear Selection**.

Follow these steps:

1. Select a table or tables in the Heat Map.
   Hover your mouse over a table to view a summary of the table details and the tags that were identified as PII data. You can zoom into the Heat Map to view table details.
2. You can perform one of the following actions on this table/tables:

- Click **Confirm** if the tags identified in a table, or tables, are correct.
- Click **Not PII** if you are sure that a table, or tables, do not contain PII data.
- **(Single table only)** Click **Investigate** to see a list view that includes details of columns, tags, and the sample data matched for each column that was identified as PII data.
  From this view, you can do the following:
  - Click **View Random Row** to view a random row from the selected table to get a better understanding of data available in the selected table.
  - Click tags that you confirm are appropriate to the column, to 'pin' the tag. To unpin a tag, click the tag again.
  - Click the plus icon to add tags for columns that should be identified as PII data. The first tag you add is defined as the column's Primary tag.
    When you type in the Tag Name field, a drop-down list of available tags appears. If you add your own custom tag (i.e. not from the drop-down list), the next time you add a new tag, your custom tag is available from the drop-down list of tags.

    NOTE

    The tags that the Audit Scan automatically assigns, already have associated masking functions. User-defined tags do not have associated masking functions, until you define them from the Manage Data Classifiers.
  - Click the **X** icon associated with each tag, to remove the tag from the column. You can click **Remove Unpinned Tags** to remove all tags that are not pinned, from all columns .

    NOTE

    You must provide a reason when you manually add or remove tags from columns. The 'Reason' field automatically populates with your last input value.
  - Click **Confirm And Review Next Table** to automatically review the next table or click **Confirm And Close** to manually select the next table you want to review.
    A tick mark is added to the reviewed and confirmed tables in the Heat Map.

Download PII Scan results as CSV

To better understand the Profiling scan details in a Heat Map, you can download all details of a Heat Map into a CSV file. The CSV file includes details such as Job ID, Job Name, when the scan was initiated, Connection Profile or Environment name that was scanned, all the Heat Map details for matched tags and where they were found.

To download all details of a Heat Map in a CSV file, click **Actions** and select **Download as CSV**.

Mask PII Data

You can use the tags applied to columns in the Data Model, to mask sensitive data in your data sources.

WARNING

The data masking process is irreversible.

For more information, see Mask Data with CA TDM Portal.

'Who column' exclusions

CA TDM includes 'Who columns' (columns that refer to the creation of the data and and therefore do not create meaningful relationships with the data) in the Data Model, but only their foreign key relationships to the rest of the data are included in relationship discovery.

'Who columns' excluded from Relationship Discovery

The following columns refer only to the creation of data. Although CA TDM includes these columns in the Data Model, only their foreign key relationships to the rest of the model are included in relationship discovery:

Excluded columns
CREATIONDATE
CREATION_DATE
CREATED_BY
DATE_CREATED
CREATEDDATE
CREATED_DATE
ENDTIME
ENDDATE
ENDDATETIME
ENDTIMESTAMP
STARTTIME
STARTDATE
STARTDATETIME
STARTTIMESTAMP
DATE_UPDATED
UPDATEDDATE
UPDATED_DATE
LAST_UPDATED_BY
LAST_UPDATE_DATE
LAST_UPDATE_LOGIN
UPDATED_BY
MODIFIEDDATE
MODIFIED_DATE
PROGRAM_CREATED
PROGRAM_UPDATED
WHO_CREATED
WHO_UPDATED

Making changes to the list of 'Who columns'

You can add entries to the `gtrep_datamodel_who_columns` table. CA TDM then excludes these entries from relationship discovery.

> **WARNING**
>
> The `gtrep` depository contains system data. You should only make changes to the gtrep depository if you understand the risks of this action. Contact CA Support if you are unsure how to proceed.

Examples

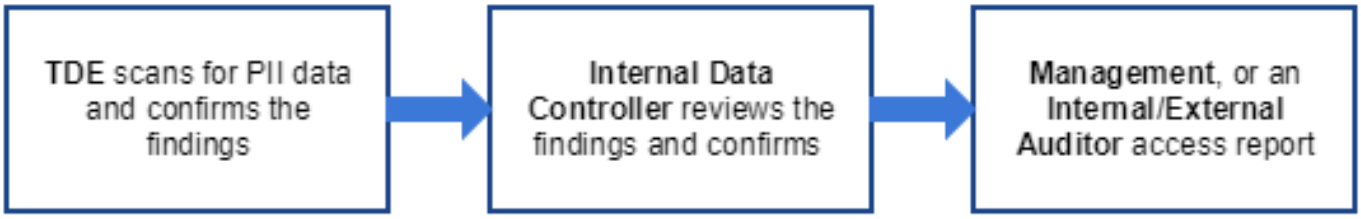
- **Add 'MODIFIED_BY' column to list of WHO columns** `INSERT INTO gtrep_datamodel_who_columns values (15,'MODIFIED_BY');` CA TDM excludes 'MODIFIED_BY' column from relationship discovery.
- **Update 'MODIFIEDDATE' column to 'MODIFIED_DATE'** `UPDATE gtrep_datamodel_who_columns set name = 'MODIFIED_DATE' where name = 'MODIFIEDDATE';` CA TDM excludes 'MODIFIED_DATE' column from relationship discovery.
- **Delete 'MODIFIED_DATE' column from list of WHO columns** `DELETE FROM gtrep_datamodel_who_columns where name = 'MODIFIED_DATE';`
  CA TDM does not excludes 'MODIFIED_DATE' column from relationship discovery.

PII Audit Using CA TDM Portal

PII Audit provides a reliable solution to identify any Personally Identifiable Information (PII) data across multiple data sources in an environment. Once you have identified the PII data, you can make business decisions to secure, encrypt, archive, or delete the PII data. Compliance and adherence to regulations is a critical business requirement to help prevent data breaches and their consequences.

The different persona-based flow for PII Audit is as follows:

Figure 21: Data Profiling role based scenario



A Test Data Engineer (TDE) performs PII data scan as follows:

1. Log in to CA TDM Portal.
2. Create Classifier Packs and import Classifiers into CA TDM Portal. For more information about creating and importing Classifiers, see Manage Data Classifiers.
3. Select one or more Connection Profiles or an Environment.
   For more information about how to create a connection profile, see Create and edit Connection Profiles.
   For more information about how to create an environment, see Create an Environment.
4. Select the appropriate Classifier Packs.
5. Select the PII data scan level.
6. Select Matched Samples option if you want to view matched samples for each column in a table.
7. Include or exclude specific tables from the PII data scan.
8. Perform a scan for PII data.
9. Review scan results in the Heat Map to identify the potential risk from PII data that exists within the selected Environment.
10. Review the results that are found for each scanned table, adjust the tags as required, and confirm the table.
11. Confirm that all tables are reviewed. Preview the draft report and submit it for an Internal Data Controller to review and sign off.

    NOTE

    At any stage in the PII data scan process, you can go back to the initial scan configuration step and you can modify the scanning rules to perform a re-scan.

An Internal Data Controller reviews the findings as follows:

1. Receive an email invitation to review the report.
2. Log in to CA TDM Portal.
3. Review the report and sign off.
4. Download the signed off reports as a PDF, with Executive Summary, Job Summary, and Audit Reports.

A Management user and an external auditor request Audit Report from the TDE. An Internal Auditor logs in to CA TDM Portal to access the final signed off report.

This section contains the following procedures:

- Prepare the Environment for PII Data Scan
- Manage Data Classifiers
- End-to-End Scenario for PII Audit

API Reference material:

- Use APIs to Audit and Mask PII Data
- TDMModelService
- TDMMaskingService

PII Data Scan Terminology

- **PII Scan** refers to the process of applying filters to a set of data sources to discover potential Personally identifiable information (PII). CA TDM can use the results of this scan in two ways:
  - To create a **PII Audit** of data sources (this does not change the source data).
  - To create a **masking configuration**, which can mask PII data that it identifies in a Data Model.
- **Environment** refers to a list of connection profiles. For more information about connection profiles, see Create and edit Connection Profiles.
- **Data Classifiers** refers to a scanning rule for PII data that can be customized as per the user requirement. Examples of basic classifiers are first name, last name, address, telephone number, credit card number, email address, passport ID, government ID.
- **Classifier Packs** refers to a group of different Classifiers. You can create and import additional Classifier Packs into CA TDM Portal. For more information about creating Classifiers, see Manage Data Classifiers.
- **Seedlist** refers to a collection of example data objects (a data dictionary) that are used to match against Data Sources.
- **Scan Report** refers to a scan report generated by different personas who are part of the PII Audit process such as:
  - Test Data Engineer (TDE)
  - Internal Data Controller
  - Management User, Internal Auditor, or External Auditor
- **Tags** refers to a label used to mark different PII data based on the classifier that is used to identify the data as PII; or a TDE manually identifies the data as PII.
- **Heat Map** refers to a graphical view to identify the total potential risk from PII that exists within the selected data sources.
- **Ready for Review** refers to a scan job and report that is ready for review by an Internal Data Controller to confirm before creating a final report that is suitable for being archived.

Prepare the Environment for PII Data Scan

Before you begin with PII data scan, prepare your environment. You must consider the Data Sources and the size of the Data Source to be scanned, the Personally Identifiable Information (PII) data you want to find, and if the current Classifiers meet your PII requirements.

Data Discovery and PII Audit has the following considerations:

- Define one or more Connection Profiles that you want to scan. You can group Connection Profiles in an Environment. For more information about how to create a connection profile, see Create and edit Connection Profiles. For more information about how to create an environment, see Create an Environment.
- The Data Sources and the size of the Data Sources to be scanned. Running a PII data scan job on multiple Data Sources, or Data Sources with large number of tables and columns may take a long time.

TIP

We recommend running a PII data scan job on Data Sources for specific applications to get quicker results and break down PII data scan into manageable operations.

- PII data scan jobs are configured per Project and per Project Version.

TIP

We recommend that you create Projects for each group of Data Sources you want to scan.

- Review the current Classifiers to ensure that they meet your PII requirements.
  Example: A mobile network provider wants to identify which data pack is being used by each customer in a location. The mobile network provider creates a seedlist and a classifier with data pack names and imports the classifiers in to CA TDM Portal. For more information about how to import classifiers, refer to Manage Data Classifiers.
- Ensure that a TDE is a member of the Admin User Group for the required project. The TDE needs to have permissions to perform the following tasks:
  - Define Connection Profiles and Data Sources
  - Assign reviewers
  - Import Classifiers in to CA TDM Portal
- For the required project, create a user group for internal Data Controllers to review and sign off the PII Audit report. Associate the user group with the Report Sign Off function and associate the required users to this user group. For more information about how to create a user group and how to add users to a user group, refer to User and Group Management.

Limitations

Data Discovery and Profiling has the following limitations:

- PII data scan does not scan BLOB objects inside a data source.
- Depending on the selected Scan level, PII data scan examines a specific percentage of your environment. For example, if you set the Scan Level to 10%, CA TDM performs a PII data scan on 10% of your environment only with a minimum of at least 10 rows per table. Subsequent scans on the same tables return slightly different results depending on the range of data values that are found in columns and the coverage in the corresponding matching seedlists.

Manage Data Classifiers

Data Classifiers are a set of rules in JSON format, that CA TDM uses for two purposes:

- **To perform a PII scan**
  For the PII scan, CA TDM compares column values to seedlists or regular expressions within classifiers, to assign a classifier's tag to that column. For more information on the PII Audit procedure, see PII Audit Using CA TDM Portal.
- **To mask data**
  When CA TDM masks data, it uses a classifier's seedlist or regular expression to generate masked data. For more information on the masking process, see Mask Data with CA TDM Portal.

For a list of terms used in the process, see PII Data Scan Terminology.

This page covers the following topics:

The two classes of Data Classifiers are as follows:

- **RegEx**

Classifiers that recognize column names and table content that match the appropriate regular expression.
For example, a UK Postcode Classifier is a RegEx Classifier that contains the appropriate regular expression.

- **SeedList**
  Classifiers that recognize column names and table content that match a sample list of values.
  For example, a UK Given Name Classifier is a SeedList Classifier that contains a sample list of UK given names.

CA TDM loads classifiers at startup. To add your own classifiers to the classifiers that CA TDM loads at startup, see Add Classifiers to CA TDM.

How to create a RegEx or SeedList Classifier

Use the following JSON code to create a customized RegEx or SeedList Classifier.

Syntax

```
{
  "name":"name",
  "description":"description",
  "classifierOrigin":"company name",
  "classifierClass":"com.ca.tdm.profiler.classifiers.RegExClassifier",
  "classifierType":"content",
  "tags":"tag name",
  "config":[
            {
             "name":"name",
             "value":"value"
            }
           ]
}
```

Parameters

- **name**Specifies the name of the Classifier.
- **description**(Optional) Specifies a generic description of the Classifier.
- **classifierOrigin**Specifies the origin of the Classifier. By default this parameter is set to CA.
- **classifierClass**Specifies the class of the Classifier.
  - For RegEx Classifier:
    By default this parameter is set to `com.ca.tdm.profiler.classifiers.RegExClassifier`
  - For SeedList Classifier:By default this parameter is set to
    `com.ca.tdm.profiler.classifiers.SeedListClassifier`
- **classifierType**
  Identifies the type for each Classifier. There are two possible values for classifierType:
  - **content**
    These Classifiers scan the contents of a column (against either a regular expression or seedlist, dependent on the Classifier class)
  - **column**
    These Classifiers only scan the title of a column (against either a regular expression or seedlist, dependent on the Classifier class)
- **tags**Specifies a tag that the Classifier associates with matched columns.
- **config**Specifies the name and value parameters for a Classifier to match content during PII data scan. Choose one of the following:

- For RegEx Classifier:
  Specify the name and enter a Java-compliant regular expression.
- For SeedList Classifier:
  Do not edit the value of the name parameter. The name value in the JSON file is used to match with the name value in the corresponding SEEDLIST file.
  The value parameter specifies the name of the SEEDLIST file.
  Note: Only one config item is allowed in a SeedList Classifier JSON file.

Examples: Create a RegEx and SeedList Classifier

Example 1: Create a RegEx Classifier

This example creates a RegEx Classifier:

```
{
   "name":"IBAN",
   "description":"Classifier to identify an IBAN from the United Kingdom, Germany or
 Sweden",
   "classifierOrigin":"CA Technologies",
   "classifierClass":"com.ca.tdm.profiler.classifiers.RegExClassifier",
   "classifierType":"content",
   "tags":"IBAN",
   "config":[
            {
              "name":"Germany",
              "value":"(?:DE)[\\d]{2}\\s?(?:[\\d]{4}\\s?){4}[\\d]{2}"
            },
            {
              "name":"UK",
              "value":"(?:GB)(?:[\\d]{2})\\s?(?:[A-Z]{4})\\s?(?:[\\d]{4}\\s?){3}[\\d]
{2}"
            },
            {
              "name":"Sweden",
              "value":"(?:SE)[\\d]{2}\\s?(?:[\\d]{4}\\s?){5}"
            }
           ]
}
```

Example 2: Create a Seedlist Classifier

This example creates a SeedList Classifier:

```
{
   "name": "German Given Name",
   "description": "Seedlist classifier for given names (German).",
   "classifierOrigin": "CA Technologies",
   "classifierClass": "com.ca.tdm.profiler.classifiers.SeedListClassifier",
   "classifierType": "content",
```

```
    "tags": "Given Name",
    "config": [
              {
                "name": "name",
                "value": "Given Name (Germany)"
              }
              ]
}
```

Example: Create a SeedList File

This is an example of a SeedList file, used by the SeedList Classifier above:

```
name:Given Name (Germany)
description:Given Name (Germany)
origin:CA Technologies
revision:1
values:
Abbo
Abelard
Achim
Adalgisa
Adelaide
```

Include Masking Functions in a Classifier

You can include one or more masking functions in a RegEx Classifier or a SeedList Classifier. You can use these when you generate a masking configuration for a Data Model. You define a Mask Function Group (maskFunctionGroup), and all masking functions defined within this Mask Function Group are associated with the Classifier.

Each Classifier has a tag, and at least one masking function is associated with the tag. Masking functions have between zero and four parameters.

For more information about all the supported masking functions and their required parameters, see Masking Functions and Parameters.

NOTE

If a classifier has no tag, the Mask Function Group section should be empty in the Classifier JSON file.

Use the following JSON code to customize and add a Mask Function Group to a RegEx or SeedList Classifier.

Syntax

```
{
  "name": "name",
  "description": "description",
  "classifierOrigin": "company name",
  "classifierClass": "com.ca.tdm.profiler.classifiers.SeedListClassifier",
  "classifierType":"content",
  "tags": "tag name",
  "config": [
             {
```

```
                "name": "name",
                "value": "value"
              }
            ],


  "maskFunctionGroup":[
{
 "groupName": "group name",
"maskFunction":
[

 {
                    "functionName": "function name",
                    "displayName": "display name",
                    "notes": "notes",

                    "maskParams":
 [

 {
                                        "paramPosition": "parameter
 position",
                                        "paramValue": "parameter
 value"
                                    },
                                ]

 }
                ]
}
]
}
```

Parameters

- **maskFunctionGroup**Specifies the name for a group of masking functions.
  - **maskFunction**Specifies a list of names and parameters for all the masking functions in this group. Each item within the list has the following parameters:
    - **functionName**Specifies the name of the masking function.
    - **(Optional) displayName**Specifies the user-defined alias for a function name that appears in the TDM Portal.
    - **(Optional) notes**Specifies additional details about the masking function.
    - **maskParams**Specifies the parameters to be used during masking.
      - **paramPosition**Specifies the parameter position of each masking parameter.
        Values: 1,2,3, or 4

NOTE

Ensure that you enter the correct parameter position as supported by the masking function. For more information about the masking functions and their required parameters, see Masking Functions and Parameters.

- **paramValue**Specifies the parameter value of each masking parameter.

Examples: Masking functions in Classifiers

In both of the following examples, masking functions are defined inside a masking function group callled "masking functions".

Example 1: Create a SeedList Classifier with a Masking Function

This example creates a SeedList Classifier with the masking function HASHLOV, with firstname.txt as the argument for HASHLOV's first parameter:

```
{
  "name": "Given Name (UK)",
  "description": "Seedlist matcher for given names (UK)",
  "classifierOrigin": "CA Technologies",
  "classifierClass": "com.ca.tdm.profiler.classifiers.SeedListClassifier",
  "classifierType":"content",
  "tags": "Given Name",
  "config": [
              {
                "name": "name",
                "value": "Given Name (UK)"
              }
            ],
  "maskFunctionGroup":
[
{
  "groupName": "masking functions"
  "maskFunction":[
                   {
                     "functionName": "HASHLOV",
                     "displayName": "Given Name UK",
                     "notes": "Given name derived from a hashed index into a lookup-
table",
                     "maskParams": [
                                     {
                                       "paramPosition": "1",
                                       "paramValue": "firstname.txt"
                                     }
                                   ]
                   }
                 ]
}
```

```
]
}
```

Example 2: Create a SeedList Classifier with multiple Masking Functions

This example creates a SeedList Classifier with the following masking functions:

- HASHLOV, with firstname.txt as the argument for HASHLOV's first parameter
- RANDLOV, with lastnameindian.txt as the argument for RANDLOV's first parameter

```
{
  "name": "Given Name (UK)",
  "description": "Seedlist matcher for given names (UK)",
  "classifierOrigin": "CA Technologies",
  "classifierClass": "com.ca.tdm.profiler.classifiers.SeedListClassifier",
  "classifierType":"content",
  "tags": "Given Name",
  "config": [
              {
                "name": "name",
                "value": "Given Name (UK)"
              }
            ],
  "maskFunctionGroup":
[
{
  "groupName": "masking functions"
  "maskFunction":[
                   {
                     "functionName": "HASHLOV",
                     "displayName": "Given Name UK",
                     "notes": "Given name derived from a hashed index into a lookup-
table",
                     "maskParams": [
                                     {
                                       "paramPosition": "1",
                                       "paramValue": "firstname.txt"
                                     }
                                   ]
                    },
                    {
                     "functionName": "RANDLOV",
                     "displayName": "Last Name India",
                     "notes": "Last name derived from a random index in a lookup-table",
                     "maskParams": [
                                     {
                                       "paramPosition": "1",
                                       "paramValue": "lastnameindian.txt"
```

```
                                        }
                                    ]
                                }
                            ]
  }
  ]
  }
```

Create a Classifier Pack

You can create classifier packs, to add them to CA TDM.

Follow these steps:

1. Download and edit the following Classifier files as required:
   - #unique_250
   - #unique_251
   - Sample Values.seedlist
   - #unique_253
2. Save the Classifier file in a directory within a zip file.
   The hierarchy of the files and directories in the zip file is replicated as the hierarchy of Classifiers in the CA TDM Portal.
   For example, the *classifier-data.zip* file contains a classifier pack directory named Common. The individual classifier files are saved under two group directories, Financial and Personal. When importing the *classifier-data.zip* file in to the CA TDM Portal, the following tree structure appears under the preview in **Classifiers:**

- **Common**
  - **Financial**
    - **Credit Card**
    - **IBAN**
    - **Swift Code**
  - **Personal**
    - **Birth Date**
    - **E-mail**

Import a Classifier pack

You can import classifiers to CA TDM during use. After you import a Classifier in CA TDM Portal, the Classifier remains available in CA TDM Portal after restarting the server.

Follow these steps:

1. Open the CA TDM Portal as administrator.
2. Click **Configuration**, **Classifiers**.
3. Drag and drop the zip file onto the grey 'Drag and Drop File or Click to select' button, or click the button to browse for the zip file, and click **Open** when you locate the zip file. The import process starts automatically.

   NOTE

   If you try to import duplicate classifiers, a warning message to overwrite the existing classifiers appears. Click **Yes** to overwrite the duplicate Classifiers or click **No** to ignore the duplicate Classifiers.

The Classifier files are successfully imported into the CA TDM Portal. A preview of the imported Classifiers appears under **Classifiers**.

Delete a Classifier

You can delete classifiers from CA TDM Portal.

1. Open the CA TDM Portal as administrator.
2. Click **Configuration**, **Classifiers**.
3. In the preview, select one or more classifiers in the tree structure.
4. Click **Delete**.
   A confirmation prompt appears.
5. To confirm the deletion of selected classifiers, click **Delete**.

List of Classifiers

Out of the box, CA Test Data Manager includes the following list of classifiers:

NOTE

You cannot edit these Classifiers as they are pre-loaded in to the system. However, you can delete individual Classifiers from the system and import customized Classifiers to replace them. For more information about creating a Classifier, see Manage Data Classifiers.

Classifier Pack Name	Classifier Name	Classifier Class	Classifier Type	Tag	RegEx or SeedList	Default Masking Function (Includes SeedList file name where appropriate)
Common	Credit Card Column	RegEx	column	Credit Card	.*creditcard.*	GENCARD
Common	Credit Card	RegEx/Luhn	content	Credit Card	4\d{12}	GENCARD
					4\d{15}	
					(4\d{3})[-](\d{4})[-](\d{4})[-](\d{4})	
					4\d{18}	
					(4\d{3})[-](\d{4})[-](\d{4})[-](\d{4})[-](\d{3})	
					3[47]\d{13}	
					3[47]\d{2}[-](\d{4})[-](\d{4})[-](\d{3})	
					(5[1-5]\d{2}|222[1-9]|22[3-9]\d|2[3-6]\d{2}|27[01]\d|2720)[-](\d{4})[-](\d{4})[-](\d{4})	
					(5[1-5]\d{2}|222[1-9]|22[3-9]\d|2[3-6]\d{2}|27[01]\d|2720)\d{12}	

3(?:0[0-5]|[68]\d)\d{11}	
(3(?:0[0-5]|[68]\d)\d)[-](\d{4})[-](\d{4})[-](\d{2})	
6(?:011|5\d{2})\d{12}	
(6(?:011|5\d{2}))[-](\d{4})[-](\d{4})[-](\d{4})	
6(?:011|5\d{2})\d{15}	
(6(?:011|5\d{2}))[-](\d{4})[-](\d{4})[-](\d{4})[-](\d{3})	
(?:2131|1800)\d{11}	
(2131|1800)[-](\d{4})[-](\d{4})[-](\d{3})	
(?:352[89]|35[345678]\d)\d{12}	
(352[89]|35[345678]\d)[-](\d{4})[-](\d{4})[-](\d{4})	
(?:352[89]|35[345678]\d)\d{15}	
(?:352[89]|35[345678]\d)[-](\d{4})[-](\d{4})[-](\d{4})[-](\d{3})	
63[789]\d{13}	
(63[789]\d)[-](\d{4})[-](\d{4})[-](\d{4})	
(?:5018|5020|5038|5893|6304|6759|6761|6762|6763)\d{12}	
(?:5018|5020|5038|5893|6304|6759|6761|6762|6763)[-](\d{4})[-](\d{4})[-](\d{4})	
(?:5018|5020|5038|5893|6304|6759|6761|6762|6763)\d{15}	

					(?:5018|5020|5038|5893|6304|6759|6761|6762|6763)[-](\d{4})[-](\d{4})[-](\d{4})[-](\d{3})	
					(\d{4})[-](\d{4})[-](\d{4})[-](\d{4})	
					(\d{4})[-](\d{4})[-](\d{4})[-](\d{4})[-](\d{3})	
Common	IBAN	RegEx	content	IBAN	(?:DE)[\d]{2}\s?(?:[\d]{4}\s?){4}[\d]{2}	HASHLOV iban-random.txt
					(?:GB)(?:[\d]{2})\s?(?:[A-Z]{4})\s?(?:[\d]{4}\s?){3}[\d]{2}	
					(?:SE)[\d]{2}\s?(?:[\d]{4}\s?){5}	
Common	IBAN Column	RegEx	column	IBAN	.*iban.*	HASHLOV iban-random.txt
Common	SWIFT Code	RegEx	column	SWIFT	.*swi?ft.*	HASHLOV swift-bic.txt
Common	Financial Column	RegEx	column	Financial	.*account.* | .*salary.* | .*revenue.* | .*profit.* | .*sales.* | .*transaction.*	FORMATENCRYPT
Common	Name Column	RegEx	column	Name	.*nm.* | .*nam.*	HASHLOV lastnames.txt
Common	Birth Date	RegEx	column	Birth Date	.*(?:Bi?rth?.*Da?t?e|Da?te.*Bi?rth|Bi?rthda?y).*(?:.*[^a-z]|^)dob(?:[^a-z].*|$)	HASHDOB
					((?:19|20)\d\d)(0[1-9]|1[012])(0[1-9]|[12]\d|3[01])	
					(0?[1-9]|1[012])([- /.])(0?[1-9]|[12]\d|3[01])\2((?:19|20)\d\d)	
					(0?[1-9]|[12]\d|3[01])([- /.])(0?[1-9]|1[012])\2((?:19|20)?\d\d)	
Common	E-mail	RegEx	content	Email Address	[\w-\.]+@([\w-]+\.)+[\w-]{2,4}	HASHLOV emailaddresses.txt

Common	E-mail Column	RegEx	column	Email Address	.*email.*	HASHLOV emailaddresses.txt
Common	Phone Number	Java	content	Phone Number		
Common	Phone Number Column	RegEx	column	Phone Number	.*phone.* | .*number.* | .*mobile.*	FORMATENCRYPT
Common	IPv4 Address	RegEx	content	IPv4 Address	((25[0-5])|(2[0-4][0-9])|([01]?[0-9][0-9]?))\.((25[0-5])|(2[0-4][0-9])|([01]?[0-9][0-9]?))\.((25[0-5])|(2[0-4][0-9])|([01]?[0-9][0-9]?))\.((25[0-5])|(2[0-4][0-9])|([01]?[0-9][0-9]?))	HASHLOV ipv4-random.txt
Common	IPv6 Address	RegEx	content	IPv6 Address	((?:(?:[A-F])|(?:\d)){1,4}::?){4,7}(?:(?:(?:[A-F])|(?:\d)){1,4})(?:(?:(?:%)|(?:/))?(?:[A-F\d]{0,4}))	HASHLOV ipv6-random.txt
Common	Country	SeedList	content	Country	Country	HASHLOV country.txt
Common	Country Column	RegEx	column	Country	.*count.*y.*	HASHLOV country.txt
Common	City Column	RegEx	column	City	.*city.*	HASHLOV usaddressbig.3.txt
Common	Address Column	RegEx	column	Address	.*add.* | .*adr.*	HASHLOV usaddressbig.2.txt
Common	PostCode Column	RegEx	column	PostCode	.*post.*code.*	HASHLOV ukpostcodes-sample.txt
Common	MAC Address	RegEx	content	MAC Address	(?:(?:[A-F\d]{2}(?:-|:)?){6})|(?:[A-F\d]{4}\.?){3}	HASHLOV mac-random.txt
Germany	Post Code (Germany)	RegEx	content	Post Code (Germany)	(?!01000|99999)(0[1-9]\d{3}|[1-9]\d{4})	HASHLOV germanpostalcodes.txt
Germany	Bank Account Number (Germany)	RegEx	content	Bank Account Number (Germany)	\d{10}	FORMATENCRYPT
Germany	Driving License (Germany)	RegEx	content	Driving License (Germany)	[A-Z]\d{3}[A-Z]{3}\d[A-Z]\d{2}	FORMATENCRYPT
Germany	Given Name (Germany)	SeedList	content	Given Name	Given Name (Germany)	HASHLOV firstnamegerman.txt
Germany	Surname (Germany)	SeedList	content	Surname	Surname (Germany)	HASHLOV lastnamegerman.txt
Germany	Title (Germany)	SeedList	content	Title	Title (Germany)	HASHLOV honorific-titles-german.txt

Japan	Post Code (Japan)	RegEx	content	Post Code (Japan)	[\d]{3}-[\d]{4}	HASHLOV japan-zip-codes-full.txt
Sweden	Post Code (Sweden)	RegEx	content	Post Code (Sweden)	(?:(?!99)(?:[1-9]\d))\d\s\d{2}	HASHLOV swedishpostalcodes.txt
Sweden	Bank Account Number (Sweden)	RegEx	content	Bank Account Number (Sweden)	\d{11}(?:\d{5})?	FORMATENCRYPT
Sweden	Driving License (Sweden)	RegEx	content	Driving License (Sweden)	\d{6}-\d{4}	FORMATENCRYPT
Sweden	National ID (Sweden)	RegEx	content	National ID (Sweden)	\d{6}-\d{4}	FORMATENCRYPT
UK	Post Code (UK)	RegEx	content	Post Code (UK)	([Gg][Ii][Rr] 0[Aa]{2})|((([A-Za-z][0-9]{1,2})|(([A-Za-z][A-Ha-hJ-Yj-y][0-9]{1,2})|(([A-Za-z][0-9][A-Za-z])|([A-Za-z][A-Ha-hJ-Yj-y][0-9]?[A-Za-z])))) [0-9][A-Za-z]{2})	HASHLOV ukpostcodes-sample.txt
UK	Sort Code (UK)	RegEx	content	Sort Code (UK)	(?:(?:\d{2})-){2}(?:\d{2})	FORMATENCRYPT
UK	Driving License (UK)	RegEx	content	Driving License (UK)	([A-Z9]{5})(\d)([0156]\d)([0-3]\d)(\d)([A-Z][A-Z9])(\d)([A-Z]{2})\s*(\d{2})?	FORMATENCRYPT
UK	National Insurance Number (UK)	RegEx	content	National Insurance Number (UK)	(?!(?:BG)|(?:GB)|(?:KN)|(?:NK)|(?:NT)|(?:TN)|(?:ZZ))(?:(?:(?!D|F|I|Q|U|V)(?:[A-Z]))(?:(?!D|F|I|O|Q|U|V)(?:[A-Z])))\d{6}(?:(?:[A-D])|\s)	FORMATENCRYPT
UK	National Insurance Number Column (UK)	RegEx	column	National Insurance Number (UK)	.*nat.*ins.* | .*ni.*no.*	FORMATENCRYPT
UK	County (UK)	SeedList	content	County	County (UK)	HASHLOV ukcounties.txt
UK	Town (UK)	SeedList	content	Towns	Town (UK)	HASHLOV uktowns.txt
UK	Ethnicity (UK)	SeedList	content	Ethnicity	Ethnicity (UK)	HASHLOV ukethnicity.txt
UK	Gender (UK)	SeedList	content	Gender	Gender (UK)	HASHLOV ukgender.txt
UK	Given Name (UK)	SeedList	content	Given Name	Given Name (UK)	HASHLOV firstnames.txt

UK	Religion (UK)	SeedList	content	Religion	Religion (UK)	HASHLOV ukreligions.txt
UK	Surname (UK)	SeedList	content	Surname	Surname (UK)	HASHLOV lastnames.txt
UK	Title (UK)	SeedList	content	Title	Title (UK)	HASHLOV honorific-titles.txt
USA	ZIP Code (USA)	RegEx	column	ZIP Code (USA)	.*zip.*	USZIP+4
USA	State (USA)	RegEx	column	State (USA)	.*state.*	HASHLOV usaddressbig.3.txt
USA	Social Security Number (USA)	RegEx	content	Social Security Number (USA)	(?!(?:666)|(?:000))(?:(?!9)(?:\d)(?:\d{2}))(?:-?(?!(?:00))(?:\d{2})){3}	FORMATENCRYPT
USA	Social Security Number Column (USA)	RegEx	column	Social Security Number (USA)	.*ssn.* | .*social.*	FORMATENCRYPT

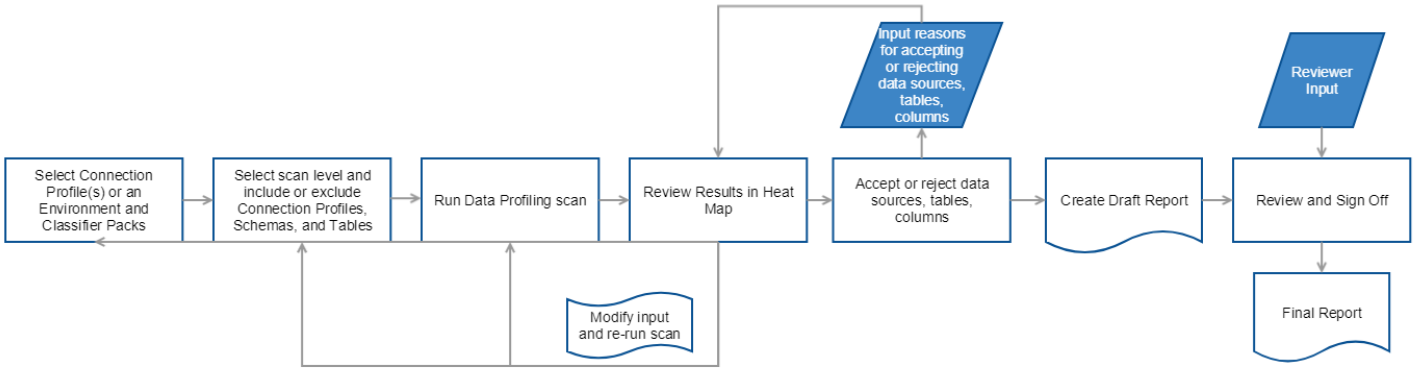
End-to-End Scenario for PII Audit

The PII Audit End-to-End Scenario outlines the step-by-step process for a Test Data Engineer (TDE) to identify any Personally Identifiable Information (PII) data across multiple data sources in an environment.

This video describes how a Test Data Engineer (TDE) uses the CA TDM Portal to scan Connection Profiles for PII data against one or more Classifier Packs, confirm the findings, and create a draft report to be signed off. An Internal Data Controller reviews the findings in the draft report and signs off. An Internal Auditor can download and review the final Audit report and a Management User or an External Auditor can request the Audit Report from the TDE.

The basic flow for PII Audit is as follows:

Figure 22: Data Profiling Detailed Architecture



Overview of PII Audit process

To run a PII Audit on your data, you must execute the following steps:

1. Select the required **Connection Profile** or **Environment**.
   For more information about Connection profile and Environments, see Prepare the Environment for PII Data Scan.
2. Select the **Classifier Packs** against which the PII data is matched.
   For more information about Classifier Packs, see Manage Data Classifiers.

3. Select the required **Scan Level**.
4. Select the **Matched Samples** option if you want to view matched samples for each column in a table.
5. Add known Connection Profiles, Schemas, and Tables to **Include** or **Exclude** in the PII data scan.

Scan Level

The Scan Level ranges from Basic to All based on the percentage of data you want to scan in your environment. For example, if you set the Scan Level to 10%, CA TDM performs a PII data scan scan on 10% of your environment only with a minimum of at least 10 rows per table.

- **Basic:** Performs a PII data scan on 10 samples of data for each column in a table of your environment.
- **All:** Performs a PII data scan on all columns and rows for all tables in the selected environment.

  NOTE

  Running a scan on an entire Data Source may take a long time since every record is read.

Matched Samples

When you select **Store matched Samples**, CA TDM collects ten samples of data for each column in a table and stores it in the repository until the Internal Data Controller signs off the report. The collected data is deleted after the Internal Data Controller signs off and no record of the data is preserved in CA TDM. The tables in the Heat Map include sample data for all columns that are identified as PII data.

Include or Exclude Connection Profiles, Schemas, and Tables

You can apply a filter to include or exclude Connection Profiles, Schemas, and Tables to reduce the size of your scan.

You can use the basic wild card characters such as `*` (used to match one or more characters) and `?` (used to match a single character) in the search terms to include or exclude any matching connection profile, schema, and table from the scan.

For example, when you enter `*sys` in the tables to be excluded, the scan excludes all tables that end in `sys` .

NOTE

You can either include a connection profile, schema, and table or exclude it from the scan but not both.

Perform a PII Audit

Use CA TDM to run a PII Scan as part of the PII Audit process.

Follow these steps:

1. Open the CA TDM Portal as administrator for the Project.
2. Select the required Project in the header bar.
3. Click **PII Audit**.
   The PII Audit section of the UI expands.
4. Click **Get Started**, or click **Set-up** from under the PII Audit section.
   The PII Data Scan Set-up page opens.
5. Select whether you want to run a PII Scan on:
   – An **Environment**. Select from created Environments.
   For more information about how to create an environment, see Create an Environment.
   – One or more **Connection Profiles**. Check all the Connection Profiles you want to scan.
   For more information about how to create a connection profile, see Create and edit Connection Profiles.
6. Select one or more Classifier Packs against which the PII data is matched and click **Next** to confirm your selection.
   For more information about how to create and import classifiers, see Manage Data Classifiers.

7. Choose how much data to scan. You can choose to either:
   - **Scan column names only**
     This only scans the names of columns in your environment. This scan uses only classifiers of the type 'column' (see Manage Data Classifiers).
   - **Scan column names and data**
     This scans the names and contents of columns in your environment. You have two options to set how many rows to scan from each column:
     a. Drag the slider to set the percentage of a column's rows to scan.
     b. (Optional) Set the maximum number of rows to scan for each column with the **Max Rows** field.

        NOTE

        This maximum number only overrides the value from the slider if it is lower than the value based on the slider.
8. (Optional) Select **Store Matched Samples** to store the first 10 samples that triggered each Classifier.

   NOTE

   Data for the samples is copied from the Data Source into the CA TDM repository and deleted after the Internal Data Controller signs off.

   Click **Next**.
9. Under **Include/Exclude Tables**, select one of the following:
   - **Scan All Tables**Scan executes on all tables in the data sources.
   - **Include / Exclude**'Add Filters' section appears below.
10. Under 'Add Filters', you can do the following:
    - Click **New Filter** to add a filter.
    - In the fields for each filter, enter the appropriate Connection Profile, Schema, and a comma separated list of Table names.
11. Click **Next** to confirm your selection.
    The PII Data Scan Execution page opens. This page lists your choices from the PII Audit process.
12. If you are happy with the details of the Scan to be run, select one of the following Schedule options:
    - **Now**
      When you click **Profile**, the PII Scan begins.
    - **Schedule**
      When you click **Profile,** CA TDM schedules the PII Scan to run at the time you specify.
13. Click **Profile** to begin or schedule the scan.
    CA TDM creates a new job under the **Job Requests** tab.

Job Requests

You can review all PII Audit jobs, inlcuding those with the statuses Running, Not Started (scheduled for future start) and Complete.

1. Click the **Job Requests** tab under the **PII Audit** section of the left-hand panel.
2. Click the relevant Job request row to view Additional Information about that job. This shows the **State** of the job, the **Duration** of the job, the **Scan Level**, the number of tables and columns **Scanned**, the number of tables and columns **Classified** as PII data.
   If the job is complete, you can click **Ready for Review** to go directly to the Heat map view for that job.

Review Scan Results

You can view results of completed scan jobs. Click **Ready for Review** under the **PII Audit** section of the left-hand panel.

When CA TDM completes a scan job, the scan job appears in the table on this page. Click on the job's row in the table to see results in detail.

Heat Map view

The Heat Map provides an instant graphical view to identify the total potential risk from PII data that exists within the scanned environment. The top menu bar lists the total number of PII data found within the scanned environment and the number of tables that are marked as confirmed. Each square in the Heat Map represents a table in the Data Source. You can zoom into a specific section of the Heat map to better view the table details.

You can filter tables in the following two ways:

- **filter Search Tab**:
  Use the filter tab to search for a table, column, tag, profiles, and schema in the Heat Map.
- **Risk Slider**:
  Use the Risk slider to filter tables based on their Risk category, according to the PII Scan.

Use the Filter search tab to filter based on a search term

Type in the search term to view a drop down with all matches for tables, columns, tags, connection profiles, and schemas. Click a search result to view a smaller result set and redraw the Heat Map. You can also view the filters that are applied on the Heat Map. For example, type in CREDIT to search for all tables that begin with CREDIT and all matches are displayed in the drop down. By default the first match for any type with matches will be the search criteria itself. Click on any of the matches within the drop down to activate a filter and redraw the Heat Map.

You can use the basic wild card characters such as * (used to match one or more characters) and ? (used to match a single character) in the search terms. All active filter types are 'ANDed' together and matched against all of the remaining types. For example, to search for a string that contains 'customer' within the schemas containing 'account' or 'accounts', enter the search term '*account*' and select the matching schema from the drop down. Next enter the search term '*customer' which will show results for all matches in the remaining tables, columns, tags and profiles. Matching is case insensitive, so you can get results as follows:

```
tables: 'LEGACY_ACCOUNT', 'Account', columns: 'ACCOUNT_ID', 'ACCOUNT_CUSTOMER',
'Active_Account'
```

You can also filter tables based on their size and re-draw the Heat Map. The table size range is between small and extra large.

Use the Risk slider to filter tables based on their Risk category

Drag the edges of the slider over the Risk categories to adjust your selection and redraw the Heat Map for a more specific view of the potential PII data that are identified in the scanned environment. Depending on the number of distinct tags that are identified in a table, the tables are filtered and positioned in the Heat Map as follows:

Distinct tags	Risk Level
15+	Very High
10 - 14	High
5 - 9	Medium
1 - 4	Low
0	Very Low

Manually Review Data within Tables

You can review each table in the Heat Map and further investigate if the data identified as PII is correct.

Follow these steps:

1. Select a table in the Heat Map.
   Hover your mouse over a table to view a summary of the table details and the tags that were identified as PII data. You can zoom into the Heat Map to view table details.
2. You can perform one of the following actions on this table:
   – Click **Confirm** if the tags identified in a table are correct.
   – Click **Not PII** if a table does not contain PII data.
   – Click **Investigate** to see a list view that includes details of columns, tags, and the sample data matched for each column that was identified as PII data.
   From this view, you can do the following:
   - Click **View Random Row** to view a random row from the selected table to get a better understanding of data available in the selected table.
   - Click tags that you confirm are appropriate to the column, to 'pin' the tag. To unpin a tag, click the tag again.
   - Click the plus icon to add tags for columns that should be identified as PII data.
     When you type in the Tag Name field, a drop-down list of available tags appears. If you add your own custom tag (i.e. not from the drop-down list), the next time you add a new tag, your custom tag is available from the drop-down list of tags.

     NOTE

     The tags that the Audit Scan automatically assigns, already have associated masking functions. User-defined tags do not have associated masking functions, until you define them from the Manage Data Classifiers.
   - Click the **X** icon associated with each tag, to remove the tag from the column. You can click **Remove Unpinned Tags** to remove all tags that are not pinned, from all columns .

     NOTE

     You must provide a reason when you manually add or remove tags from columns. The 'Reason' field automatically populates with the last input value.
   - Click **Confirm And Review Next Table** to automatically review the next table or click **Confirm And Close** to manually select the next table you want to review.
     A tick mark is added to the reviewed and confirmed tables in the Heat Map.

To better understand the Profiling scan details in a Heat Map, you can download all details of a Heat Map into a CSV file. The CSV file includes details such as Job ID, Job Name, when the scan was initiated, Connection Profile or Environment name that was scanned, all the Heat Map details for matched tags and where they were found.

To download all details of a Heat Map in a CSV file, click **Actions** and select **Download as CSV**.

Create and Sign Off Report

Depending on the user persona, a scan report summarizes all the scan details. For example, the Job ID, time when the scan was initiated, time when the scan was completed, environment that was scanned, the Classifier Packs that were used for the scan, the tags that were identified during the scan process, and so on.

As a **TDE**, after you review and confirm all the tables, you perform the following steps to create a scan report and request sign-off:

1. Log in to the CA TDM Portal and navigate to **PII Audit**, **Ready for Review**.
2. Click **Create Report**.
   The Submit for Sign-Off page appears.
3. Click **Download/View Report** to download and review the report.
4. Click **Submit Report For Sign-Off**.
   An email notification is sent to all the Internal Data Controllers.

As an **Internal Data Controller**, you perform the following actions when you receive an email notification to review a scan report:

1. Click the URL provided in the email and log in to the CA TDM Portal.
   The PII Audit Sign-Off page appears.
2. Click **Download/View Report** to download and review the report.
3. After reviewing the scan report, click S**ign-**O**ff**.
   The Confirm Sign-Off dialog appears.
4. Enter your comments and click **Sign-Off**.
   The PII Audit Reports page displays a list of PII Audit reports that you have signed off or the reports that are pending approval.

When all the Internal Data Controllers sign off the scan report, an **Internal Auditor** can views the signed-off report. As an Internal Auditor, you perform the following actions to view this report:

1. Log in to the TDM Portal and navigate to **PII Audit**, **Reports.**
2. Click **Audit Report** to download the final signed-off scan report.

To view the final signed-off report, a **Management User** and an **External Auditor** request the Audit Report from a TDE.

Data Discovery and Profiling Using Datamaker

CA TDM contains data sampling and discovery functionality that enables you to profile data within a project. You can use the Datamaker UI to filter the sampled data to determine which tables and columns contain PII (Personally Identifiable Information).

Profile (or Sample) Your Data

To sample or profile the data in a specific project, first ensure that all the tables you want to sample are registered. After you register the tables, follow these steps:

1. Open the Datamaker UI and select **Data Profiler, Sample Table Data** from the main menu.
   The **Perform Actions on Registered Objects** dialog opens. The left pane represents the tree structure of the project. The middle pane contains a list of the registered objects within a project.
2. Select the table that you want to sample from the middle pane.
3. Select **Sample Data** from the drop-down list in the top-right corner and click the forward arrow icon.
   The **Sample Options** dialog opens, allowing you to select the connection (Source, Target, or a Test Case).
4. Enter appropriate information in the fields.
5. Click **Build Scripts** and browse your directory for a batch file to build batch scripts, or click **OK**.
   Clicking **OK** displays the **Data Sample** dialog.
6. Verify that all the requested tables are sampled. These tables are displayed in the tabs at the top of the dialog.
7. (Optional) View the results as XML by selecting the XML icon in the bottom-right corner of the dialog.
   The **Save as XML** dialog opens. You can save these results to a CSV file or to the repository by clicking the relevant button in the XML dialog. You can also generate reports of the sample as CSV files by clicking the Generate reports icon.
8. Select the columns that you want to work with by selecting the option that is associated with that column and clicking the tick mark icon.
9. Select a sample type.

See also:

- Filter Options for Transformation Maps
- Custom Filter Functions for Transformation Maps

Verify Information Using a Filtered Sample

After you filter your data sample, CA TDM provides you with a range of options to help you understand and manage your filtered data.

1. Follow all the steps that are mentioned in the Profile (or Sample) Your Data section to filter the data.
2. Select the parallel bar icon next to the column you want to investigate.
3. The **Information for column <Column_Name>** dialog containing all the information that is related to that column opens. This dialog provides all the relevant range information about the column, including:
   - All keys, indexes, and transformation maps in which the column appears.
   - Up to five previous functions that are performed on the column within the current project.
   - Any previous names for the column and the version in which they existed.
4. Select the parallel bar icon in the top-right corner.
   The **Find similar columns to <table_name>** dialog opens. This dialog accesses a Column-to-Column comparison, in which you can compare relationships between similar columns.
5. Review the information.

Build Custom Sample Criteria

Use this procedure to understand how to build custom sample criteria using regular expressions. The custom criteria tells the sampling utility about what patterns to look for in a column, enabling the identification of specialized forms of Personally Identifiable Information (PII).

1. Open the **Sample Options** dialog as mentioned in the Profile (or Sample) Your Data section.
2. Click the **Use Filters** tab.
3. Click the plus icon to create a new filter.
4. Enter the filter name.
5. Select **Regular Expression** from the drop-down list.
   The filter syntax corresponds to standard regular expressions.
   For example, add ^[0-9] in the **Condition** field after selecting the **Regular Expression** option. This example specifies that it must look for a number (specified by the character class [0-9]) at the beginning of a given string (CA TDM syntax uses the caret (^) character rather than the dollar ($) character).
   The filter is added to the **Filter Name** list.
6. To use the filter, select the created filter name.
   The new filter is added as a custom filter to the list of filters in the **Transformation Maps** dialog.

Define Cube Dimensions and Create the View

Before you can build a cube view, create a subset of your data. To do so, start by designing a transaction, which includes all the tables you want to analyze. For more information about how to do this, see the documentation about Data Subset.

1. Open the Datamaker UI and select **Data Subset, Design Extracts and Transactions** from the main menu.
   The Data Subset UI opens.
2. Extract your subset in the Data Subset UI.
   In the left pane, you can see a chain of all the applicable tables within the transaction.
3. Navigate to the Datamaker UI and select the **Data Profiler, Create Cube View** option from the main menu.
   The **Choose Extract** dialog opens.
4. Click the row of the extract for a check mark to appear, and click **OK**.
   The **Sample Options** dialog opens.

5. Select the connection (source, target, or test case) from where you want to sample the data and click **OK**.
   The **Choose Transaction** dialog opens.
6. Select the columns that you want to analyze.
   These columns are important from a coverage standpoint and are used to drive the subset.
7. After you select the columns, the **Cube Generation Options** dialog opens defining the details and dimensions of the view you are trying to create for your analysis.
   Note: At the top of the dialog, you have the opportunity to limit the number of results for each distinct combination.
8. Click the tick mark icon.
   Two types of SQL are created—one distinct and the other including the details that are visible in the SQL Window (you are taken to the SQL Window automatically).
9. Pressing ALT + A converts the SQL to views within the database.
   For more information about registering these views, see the information about the database cubes in this documentation.

Create Seed Data from a Cube

To create seed data from cube views, first define your definition or transactions using the Data Subset component. For more information, see the documentation about Data Subset.

1. Open the Datamaker UI and select **Data Subset, Design Extracts and Transactions** from the main menu.
   The Data Subset UI opens.
2. Define the definition or transaction in the Data Subset UI.
   After the definition is created, save it to the repository.
3. To save the definition to the repository, select **File, Save Extract to Repository** from the main menu in the Data Subset UI.
   For example, in the following example, the table PRODUCTS_BASE is linked through three tables to the PERSONS table. This definition is created in the Data Subset UI:

4. Navigate to the Datamaker UI.
   You can now use this definition to extract a cube of data using these subset definitions.
5. To find the definition, use the **Project Manager** dialog to navigate to appropriate definition.
6. Right-click the selected definition and select the **Create Seed Data from Cube** option.
   The **Sample Options** dialog opens.
7. Select the appropriate connection to the sample data and click **OK**.
   The **Choose Transaction** dialog opens.
8. Select the columns that you want to include in the data cube by right-clicking on a table and adding columns.
9. Click the tick mark icon when you are done with the task.
10. Enter a name for your cube and click **OK**.
    The **Cube Generation Options** dialog opens. This dialog includes options that help you define how to generate the cube.
    Note: For normal use cases, you can use the default values.
11. Click the tick mark icon.
    A summary of the data is displayed.
    You can save the data as seed data using the save icon. You can also save the SQL and use it to create views or select statements by clicking the SQL icon.

Analyze Your Cube

You can analyze your cube as follows:

Repository Coverage Metrics

1. Open the Datamaker UI and select **Data Profiler, Repository Coverage Metrics** from the main menu.
2. Select the cube that you want to view from the drop-down list.
   This selection automatically places the appropriate columns into the selected columns pane. To deselect/select available columns, click on the column name and use the arrow buttons to move between panes.
3. Click the tick mark icon after you are done with your selection.
   The **Repository Coverage Metrics** dialog opens.
4. Click the tick mark icon in the top-right corner to recalculate the coverage and perform the analysis.
   The top pane provides a statistical summary of the coverage in the chosen cube view. This information includes the number of distinct values, number of possible combinations, and total coverage. The bottom pane of the summary tab identifies coverage weaknesses in specific columns and combinations. The exclamation mark icon identifies normal weaknesses; the cross mark icon represents severe weaknesses.
5. To find out more about each column, click on the corresponding tab at the top of the dialog.
   The top pane shows a statistical summary of distinct values. The bottom pane shows an analysis of the coverage across that column. This analysis shows the total number of paired combinations and the combinations found. This information lets you quickly identify the weaknesses in your coverage for each different combination.
6. You can export a coverage to a CSV file by clicking table icon.
7. Select the reports that you want to generate and then click the tick mark icon button to generate them.
   You can also place constraints on the cube so that only valid combinations of data are considered rather than all possible combinations. For instance, you can have a rule where a person can only have one type of account that is associated with them (and persons with more than one type of account would be considered a data error). To do so, select a constraint from the drop-down list in the top-right corner.

Repository Find Duplicates

1. Open the Datamaker UI and select **Data Profiler, Repository Find Duplicates** from the main menu.
2. Select the cube that you want to view from the drop-down list.
3. After you select all the appropriate columns, click the tick mark icon to find the duplicate values.
   A dialog showing a report about the presence or absence of the duplicates opens.

Work with Transformation Maps

As a test data engineer, you manage databases and file transformations in Test Data Manager. Transformation maps are used to scramble data, condition data, age data, or as part of the data multiplier. You can create new transformation maps, copy existing maps to new maps or a new release, and can compare maps.

Use this scenario to guide you through the process.

Create a Transformation Map

Use transformation maps to scramble, condition, or age data. To define data for use in databases or file transformations, create transformation maps.

Follow these steps:

1. Select **Projects**, **Transformation Maps**, and click the **Plus** button.
2. The transformation map dialog appears.

3. To create a map, click the **Plus** button and a new map is created.
4. To specify the map name, double-click **New Map**.
5. (Optional) To change the DBMS type, click the **DBMS** and select a type from the drop-down.
   Note: The default DBMS type is Oracle.
6. If the content is ordered, enable the **Ordered** checkbox.
7. To add a description, double-click the description field, type a description, and click **Save**.
   The transformation map is created.

Map Columns to Transformation Maps

To map columns to transformation maps,

Follow these steps:

Select a registered object in the left panel, and click on the drop-down list to apply transformations to specific columns.

Note: In the center panel, the first three boxes in the columns are for transformation map review. The three levels of transformation map review are:

- Checked
- Validated
- Approved

If a previously checked and validated map has its Checked attribute removed, the Validated attribute is also removed.

You can map the following columns to the transformation map.

- **Transformation**

Select the function from the drop-down list. To add extra functions, see Maintain Data Functions.

- **Keep Nulls**

If the original value is NULL, the new value remains NULL. The option can be toggled On and Off for each row by selecting the check box.

- **Listcol No**

The LIST function randomly picks up values from the seed table GTSRC_REFERENCE_DATA. The table allows you to store up to nine columns for one row. For example, PERSON NAME has four columns: Full Name, Title, First Name, and Last Name. The column allows you to pick from linked values so that First and Full name match.

- **Fixed Value**

Any fixed value. Enclose any characters in inverted commas, for example, `test `.

- **Cross Ref**

Cross Ref allows you to retain the same transformation for the same input value. For example, if the name Price is converted to Jones, all Prices become Jones.

- **Cross Ref Indent**

Groups the rows in the cross-reference table, allowing you to have several columns which map to a single cross-reference map.

- **Substring Start**

Enter the place value at the start of the substring to mask.
Note: Positioning starts at the value 1, and not at the value 0

- **Sub-string Finish**

Enter the place value at the start of the sub string to mask.**Note:** Positioning starts at the value 1, and not at the value 0

- **Notes**

Use to add comments.

The following columns are only available when you select SDM as the RDBMS for a map:

Note: SDM represents Fast Data Masker.

- **WHERE Clause**

Double-click the cell to enter a WHERE clause. A pop-up window shows all the columns that you can use.

- **Date Format**

Allows you to define the data format to use.

- **Preformat**

Specifies the format of the original data before it is masked.

- **XPath Element**

Specifies where in the XML data you want the masking to take place. For more information about mapping an XML file, see Defining Mapping for Multiple XML Files.

To set related tables with the same function for the column, click the tree icon to the right of the scramble function. You are guided through the model identifying any related columns.

Define Mapping for Multiple WHERE Clauses

To map multiple WHERE files within DataMaker, ensure that the Transformation Map selected is an SDM file.
Note: If you do not have an SDM file, create one. For more information, see Create a Transformation Map.

Follow these steps:

1. Select the SDM file, the column, and the transformation.
2. Double-click the WHERE clause cell, and drag-and-drop available columns into the SQL window.
3. Add text to define the WHERE clause.
   Note: We recommended that you validate your SQL against either the source or target connection.
4. When the validation is finished, click the **Disk** icon to save.
5. The WHERE clause is saved, and a new row opens to add another WHERE clause for the selected column.

Define Mapping for Multiple XML Files

To map multiple WHERE files within DataMaker, ensure that the Transformation Map selected is an SDM file.**Note:** If you do not have an SDM file, create one. For more information, see Create a Transformation Map.

Follow these steps:

1. Select the SDM file, the column, and the transformation.
2. Specify the location of the XML data to mask in the **XPath Element** cell.
3. Click to save.
   A new row opens where you can enter another XML file location.

Copy and Delete Transformation Maps

Copy transformation maps to the current project, to another version within the project, or to an existing project. To copy, edit, or delete transformation maps,

Follow these steps:

1. Select **Projects**, **Project Manager**.
2. In the **Projects** tree, expand a project and a version.
3. Select **Transformation Maps** from the tree structure.
   A list of maps that are associated with that version appears.
4. Right-click on the transformation map, and select a copy type:
   - **Copy to the same project .**
   - **Copy to the different project**.
5. (Optional) To edit or delete a transformation map, follow the same steps, and select **Edit** or **Delete** from the menu.
6. Specify the copy details, and click the **Copy** icon.
   The transformation map is copied.

Upgrade Transformation Maps Between Versions

When you upgrade a version, transformation maps are not copied to the new version by default. To copy a transformation map between versions,

Follow these steps:

1. Right-click on the transformation map, and select **Copy to the same project**.
2. Select the version, specify the name, and click the **Copy** icon.
   The transformation map is copied to the new version.

Reconcile and Merge Transformation Maps

If you have multiple transformation maps defined, you can compare and merge them between different versions. To compare and merge transformation maps between different versions;

Follow these steps:

1. Go to **Project Manager**, expand the project and a version.
2. Select **Transformation Maps**, right-click on the transformation map, and select **Reconcile and Merg**e.
3. Select the two maps to compare from the drop-down lists.
   Note: Differences between the two maps are shown in blue.
4. In the **Action** column, use the drop-down list to select one of the following merge action types.
   - **Copy to Right**
     Copies any changes from the left drop-down to the right
   - **Copy to Left**
     Copies any changes from the right drop-down to the left
   - **No Action**
     No action in either direction occurs.
5. Click the drop-down list in the top-right hand corner, and select one of the following actions;

- **Perform Action**
  Performs all the right to left and left to right actions that are set.
- **Set Copy to Left**
  Forces 'Copy to Left' the columns with differences
- **Set Copy to Right**
  Forces 'Copy to Right' the columns with differences
- **Set All No Action**
  Forces 'No Action' to the columns with differences
- **Set to Equalize Left**

Copies changes 'Left to Right' and equalizes the two lists.

- **Set to Equalize to Right**
  Copies changes 'Right to Left' and equalizes the two lists.
- **Reset**
  Returns selections back to their original state

Note: To create seed data from cube views, define your definition or transactions using Data Subset. For more information, see the Data Subset Reference Guide.

To use the created Data Subset definition to extract a cube of data:

Follow these steps:

1. Go to **Project Manager**, expand the project and a Version.
2. Expand **Subsets**, right-click on the definition, and select **Create Data from Cube**.
3. Specify the data subset information.
   Seed data is created from a cube.
   For more information about creating seed data from a data cube, see Create Seed Data from Cube.

Mask Using Transformation Maps

You can also use transformation maps to mask your production data. You do this through transformation maps and Subset scripts. You first define a transformation map (Oracle or MSSQL) in Datamaker, create masking functions for the columns you want to mask. You then use Subset to create *masked* export scripts. These scripts perform masking as they export the source data to a dump file. You can then import the dump file (which contains the masked data) to the target database.

Furthermore, you can use transformation map files in Fast Data Masker to mask the data. In this case, you export your transformation map into a CSV file and use that CSV file in Fast Data Masker. For more information about how to use transformation map files in Fast Data Masker, see Use Transformation Map Files.

Note: Review the Fast Data Masker and Transformation Maps section to understand the masking approach that you can follow depending on business requirements.

Import or Export Ordered Transformation Maps

When you export an ordered transformation map as a CSV file, the file content is not in the expected order by default.

To ensure that the order is reflected correctly, follow these steps:

1. Set the transformation map to Ordered when you create it.
2. Use the Order column in the transformation map window to specify the order manually.
3. Export the transformation map to CSV.

Similarly, when you import SDM transformation maps from a CSV file, set the Ordered flag in the Import dialog.

The CSV output is based on the following criteria:

1. Ordered by table name.
2. Then ordered by the presence of a where clause. Rows without where clause are exported first.
3. Then ordered by the order number. Rows without an order number are exported first, followed by low to high order value.
4. Then ordered by an internal transformation column ID and where clause sequence number.

Design Transformation Maps for iSeries V7R1 (DB2/400)

DB2/400 users who have access to the CA Test Data Manager Datamaker component are also able to create and design the transformation maps for Fast Data Masker within the Datamaker UI. Once a map is base lined in Datamaker, it can be moved to iSeries server using an FTP client, System i Navigator, or mapped network drive.

Using Datamaker to design your transformation maps has the following benefits:

- Datamaker stores the transformation maps in its Oracle repository, providing long-term storage.
- Datamaker lets you version control your transformation maps.
- Datamaker offers data sampling, which enables you to sample table columns to find any personally identifiable information (PII) you need to mask.
- Datamaker lets you build and store a data model of tables in the project.

Follow the steps mentioned in this article to understand how to use the Datamaker UI to create transformation maps: You can use those transformation maps in Fast Data Masker for masking purpose:

1. Access the Datamaker UI and open the **Maintain Projects** dialog.
2. Right-click the root of the projects tree and select **New Project and Version** from the context menu.
3. Enter a project name, version, and click the Advanced options icon.
4. Edit and set the following three project attributes as a best practice:
   - File Publish DBMS : db2/400
   - Publish to: Data Target
   - Key order of data group, set and pool : SEQ
5. Click the save icon to complete the project creation process.
6. Click on the **Data Source** menu item and confirm that Datamaker is connected to a db2400 database and schema, which has the tables to be masked.
   It is important that this source connection is as good as production. This is because Datamaker makes use of the information in the database catalog of this source connection to determine table relationships and sample values of the tables to be masked.
7. Right-click the project version and select **Register**.
8. Select the **Database Table** option and click the forward arrow.
9. Select the **Register Tables from Data Source** option from the drop-down list.
10. Select tables to be masked from the source schema and click the forward arrow to register. Once the registration is complete, Datamaker may prompt you to choose if you want to calculate the table order of the registered tables. Select **Yes**.
11. Navigate to the **Maintain Projects** dialog, right-click the project version, and select the **Action for Registered Objects** option.
12. Select the **Sample Data** option from the drop-down list and select tables to be sampled.
13. Click the forward arrow to start the sampling process.
14. Select the **Sample from Source Connection** option and also ensure that you select the limited data sampling functionality by choosing a maximum number of rows to be sampled.
    db2400 does not have an inbuilt support for data sampling, so using ‘limited data sampling’ of Datamaker is the quickest and best way to gain information about the tables to be masked.
    This action opens up the data sampling results dialog, which provides data sampling information about every column in the first x number of rows (for example, 100) in the table. This information is auto-saved by Datamaker and is associated with the given project and version. This information is more useful when viewed in Transformation Maps.
15. Navigate to the **Maintain Projects** dialog and highlight the project version in the **Context** field to ensure that Datamaker is aware of the correct context.
16. Click **Projects, Transformation Maps**.
17. Open a list of existing transformation maps and create a new Fast Data Masker map for the given project and version. Ensure that you choose map **DBMS** as **SDM**.

18. In the **Transformation Map** dialog, you can review the following information:
    - See all the column of the registered tables.
    - See the data sampling information about every column.
    - Choose an appropriate data masking function to be applied to sensitive data columns.

    Datamaker automatically discovers the data type of the given column and suggests a list of data masking function for it. Transformation sheets are saved into the Oracle repository of Datamaker using the blue save icon. Transformation sheets are exported to CSV file using the grey save icon. Once exported, you can use them in Fast Data Masker.

Use Personally Identifiable Data in a Transformation Map

You can find Personally Identifiable Information in a Transformation Map, to mask this data.

Follow these steps:

1. Select **Projects, Transformation Maps** to open the **Transformation Maps** dialog.
2. To load your data sample from the CSV or Excel file, click the Import SDM CSV/XLS icon and import the file to the map.
3. To filter your data sample, ensure that the tables you want to search are highlighted with a red square in the tree structure.
4. You can then select your filtering criteria from the drop-down list in the top-right corner of the dialog.

   NOTE

   The **Contains Values from Seed List** filter lets you use CA TDM to find the sensitive data you need to mask from your seed tables. This ensures that you uncover all potentially sensitive records. However, it is recommended that you do not perform this filter for all tables at once to enable better performance.
   For more information about all the filter options, see Filter Options for Transformation Maps.
5. Select the **Contains Values from Seed List** filter.
   The **Select Required Seed List** dialog opens.
6. Select the required seed list and save your changes.
7. After you select your criteria, click the filter icon to perform the filtering.
   All the columns within the specified tables, which meet or contain potentially similar data, are displayed.

Provisioning Test Data

As a Test Data Engineer, you are responsible for ensuring quality, coverage level, and referential integrity of application test data. Team access to the right data at the right time is vital to accelerate development velocity and increase quality. Test Data Manager provides capabilities for test data engineers to maintain and provision test data with simple interfaces and automated processes. These capabilities replace processes that previously required manual database scripting.

Consider the following high-level sequence of CA TDM test data provisioning activities. Not all environments require all activities:

1. Discover your data. Connect CA TDM to production (or copies of production) and testing data sources. These sources can include databases, flat files, mainframe files, and other file formats. Connecting data sources with CA TDM helps you represent the data sources in a relational model. This model lets you analyze, manipulate, and optimize data for testing.
2. Sample and profile your data. CA TDM lets you analyze and sample that data to help you understand required actions. For example, profiling helps you to identify personally identifiable information that requires masking before you use the data for testing.
3. Subset production data for testing. The volume of production data is often too overwhelming to use in a testing database. You can define rules by which CA TDM can extract a subset of a large data source to use for testing.
4. Mask production or data subset for testing. Production data typically contains personally identifiable information that you are forbidden to use in testing environments. You can use masking functions to transform all sensitive information into data acceptable for testing.
5. Visualize test data coverage and identify gaps. Once you have a sanitized set of test data, you can use Data Visualizer to view your overall test coverage and see where you might have gaps.
6. Generate synthetic data to fill test coverage gaps. After you understand what data you need to fill coverage gaps, you can use Datamaker to create rules to generate synthetic data based on column data types. You can then publish that synthetic data to any data source for use in testing.
7. Build a test data repository and configure a data request and reservation system. You can make certain data available for reservation by testers through a portal interface. Test Data on Demand gives testers the assurance that their test data is locked for their use only.

This section describes how you perform all of these operations to provision the right test data to accelerate your testing cycles.

Defining Test Data

Defining test data is the first step in the provisioning of test data in which you expose data and data sources to Test Data Manager. Defining test data refers to the high-level process that includes:

- Connecting to relational data sources
- Creating projects to contain data
- Registering data from relational and non-relational data sources
- Converting non-relational data to relational tables
- Analyzing and editing imported data to prepare for other operations

Test Data Manager supports a wide variety of data sources that you can register, including databases, flat files, mainframe databases, and more. Depending on the data type, you discover the data in one of the following ways:

- Using the CA TDM Portal
- Using Datamaker
- Using format-specific utilities for formats that do not support direct registration by Datamaker or the CA TDM Portal

The following list provides a summary of the different data types you can work with and from where you should start the discovery process for each data type.

Non-Relational Data Sources (Flat Files)

Non-relational data sources, or flat files, are data sources that do not maintain data in relational tables. Common flat file types include, XSD, XML, and CSV. After you register flat files, CA TDM puts the data in a relational table format so that it can perform other operations on the data.

Work with flat files using the CA TDM Portal.

NOTE
Defining Test Data Using the CA TDM Portal

Relational Tables

Register relational database tables to integrate the database data with CA TDM. You can then generate more data, create a subset, and more. A wide range of databases are supported, including common distributed databases, mainframe databases, and big data sources.

Work with relational tables using the CA TDM Portal or Datamaker.

NOTE

- Defining Test Data Using the CA TDM Portal
- Defining Test Data Using Datamaker

EDI Files

EDI files are a record-based file format that covers a large number of record types. There are several different EDI formats, as well. A format-specific utility is required to import the EDI files into a specialized XML file, and then import that file into Test Data Manager.

To work with EDI files, use the GT EDI utility.

NOTE

Working with EDI Files Using the GT EDI Utility

Record-Based Mainframe Files

Several mainframe data sources are indexed, record-based files, such as ISAM and VSAM. The structure of these files is typically described in a COBOL Copybook. A format-specific utility is required to read the copybook data and parse it into a format that CA TDM can understand. You can parse Copybook data into a GT Excel file format, which you can then register in the CA TDM Portal and Datamaker.

NOTE

How to Parse IMS Database Copybooks and Mask Data

After you register data, the following functionality is available to edit the registered data:

- Datamaker and the CA TDM Portal provide the ability to edit table relationships based on the derived tables created during the registration process
- Datamaker includes a utility called GTDiagrammer, which provides a visual representation of your table structure and lets you edit the data in place

When the discovery process is complete, you can progress to common data provisioning operations, such as:

- Profiling the data and creating transformation maps for masking
- Creating data generation rules and generate synthetic data
- Assembling a test data warehouse of registered data that you can use to make the data available for testers

Defining Test Data Using the CA TDM Portal

This section includes information about how you can define test data using the CA TDM Portal. You can use the Portal to perform various operations; for example, create projects, register objects, create and register derived objects, and perform actions on objects.

Complete the following process to model data in the CA TDM Portal:

1. Create and Edit Connection Profiles.
2. Create a project.
3. non-relational data sources.

Create and Edit Projects

To get started with Test Data Manager, create a project. A project must contain at least one version. For simple applications, you can work with a single version. However, if you have a complex application, create multiple project versions to address different scenarios. The version name is usually the name of the current release of your database/ application; for example, 7.A. If you are not sure about the exact version, use Version 1. You can always edit the name later when you are sure of the exact version.

You can also create one generic version within each project in Datamaker; not in the Portal. The generic version stores all generic test cases, which normal test cases can then inherit. For example, in a travel system, you can create a standard trip. The standard trip is then available when you edit the data.

Create a Project

1. Access the CA TDM Portal.
2. Click the Project Manager icon (gear icon) in the top blue bar.
   The **Manage Projects** dialog opens.
3. Click the **New Project** button.
   The **New Project** dialog opens.
4. Specify the following information and click **Save**:
   - **Name**
     Specifies the project name.
   - **Description**
     Specifies a brief description of the project.
   - **Version**
     Lets you specify an initial version of your choice for the project.
   - **Version Description**
     Specifies a brief description of the project version.
   - **All new versions inherit tables from previous version**
     Specifies whether you want a new project version to inherit tables from a previous version.

   The project is created successfully with the provided information.
5. Review the **Manage Projects** dialog to verify that the created project is available in the list.

Use the search feature to find a specific project in the list.

Edit Project Details

After you create a project, you can update the required project information based on your changed requirements. Additionally, you can use the same project details page to map Active Directory (AD)/LDAP groups to the appropriate CA TDM Portal user groups.

Follow these steps:

1. Access the CA TDM Portal.
2. Click the Project Manager icon (gear icon) in the top blue bar.
   The **Manage Projects** dialog opens.
3. Locate the project for which you want to update the relevant information or map AD/LDAP groups.
4. Click the project name in the list.
   Displays the following details of the selected project:
   - **Project Name**
     Specifies the project name. Do the following to modify the project name:
     - Click the Edit icon (pencil).
     - Modify the project name.
     - Click the Save icon. Click the Cancel icon to undo the changes.
   - **Description**
     Specifies the project description. Do the following to modify the project description:
     - Click the Edit icon.
     - Modify the project description.
     - Click the Save icon. Click the Cancel icon to undo the changes.
   - **Project Settings**
     Shows the following information related to the corresponding project:
     - **Project ID**
       Specifies the auto assigned project ID when you create a project.
     - **Inherit Tables**
       Specifies whether table definitions can be inherited from prior versions or not.
       Default: Yes
     - **Date Format**
       Specifies the default date format for publishing date columns.
     - **Project Type**
       Specifies whether the project type is ‘DB (Database)’ or ‘soapUI’. When you create project in Datamaker, you can select the type as 'soapUI' for the projects to be visible in SoapUI.
       Default: DB
   - **User Groups**
     (For AD/LDAP authentication mode) Lets you map Active Directory (AD)/LDAP groups to appropriate CA TDM Portal user groups for the selected project. Search for and select the required AD/LDAP group by entering its name in the **AD/LDAP Groups** field. With this mapping, when users belonging to the mapped AD/LDAP group try to log into the CA TDM Portal for the first time, they are automatically added to the CA TDM repository. Administrators do not need to add them explicitly to the CA TDM Portal user group. Such users can log into the CA TDM Portal using their AD/LDAP credentials; they are authenticated based on the mapped CA TDM Portal user group. They get access to the same resources that are available to other users, who are part of the CA TDM Portal user group (mapped).
     Note: You can map AD/LDAP groups to a CA TDM Portal user group from the user group management page, too. Additionally, for more information about how to integrate Active Directory with the CA TDM Portal, see LDAP Integration with the CA TDM Portal.
5. Review the changed information.

You have successfully updated the project information and mapped AD/LDAP groups to the CA TDM Portal user groups.

Delete a Project

If you no longer need a specific project, you can delete it from your CA TDM Portal environment. You must have appropriate privileges to delete a project.

You cannot delete a project if it includes versions. If you want to delete such a project, you must first delete all the versions that are included in the project.

1. Access the CA TDM Portal.
2. Click the Project Manager icon (gear icon) in the top blue bar.
   The **Manage Projects** dialog opens.
3. Locate the project that you want to delete.
4. Click the Delete Project icon (X icon) in the row corresponding to the identified project that you want to delete.
   A confirmation message appears.
5. Click **Delete** to proceed with the delete process.
   A confirmation message states that the project is successfully deleted.

Last-Accessed Project

The CA TDM Portal remembers the last project that you access before you move out of the Portal. For example, if your browser is closed and you again log into the Portal using the same browser, the Portal selects the last project that was accessed before the browser was closed. You are not required to re-select your last-accessed project every time you log into the Portal.

This behavior is applicable if you use the same browser to access the Portal for your subsequent sessions. If you use a different browser for your subsequent sessions, the Portal does not remember your last-accessed project in the new browser. For example, if you use Google Chrome to access the Portal for one session and later you use Mozilla Firefox for the next session, the last project that you selected in Google Chrome is not automatically selected when you open the Portal in Mozilla Firefox.

Additionally, if you clear the browser cache, the project selection is lost.

NOTE
For more information about project versions, see Manage Project Versions.

Manage Project Versions

Each project that you create in the CA TDM Portal gets associated with at least one version, which is created at the time of project creation. For simple applications, you can work with a single version. However, if you have a complex application, create multiple project versions to address different scenarios. You can add multiple versions to the same project as and when required.

Version Types

The following are the two types of versions:

- **Normal**
  A project can have any number of normal versions. Each normal version includes information that is specific to that project version. This information is not common to all the project versions.
- **Generic**
  A project can have only one generic version. A generic version contains common information that is applicable to all the versions. For example, you can use a generic version to define generic rules that other versions can inherit.
  Note: You cannot create a generic version in the CA TDM Portal. However, if a project in DataMaker includes a generic version, the Portal displays that generic version as a first row in the **Versions** table. Also, no Upgrade Version

icon is available for a generic version (in the Portal). This information helps you distinguish a generic version from a normal version in the Portal.

Version Considerations

Review the following version-related considerations. This information helps you understand and implement the version functionality in an efficient way:

- The order of a version is determined by the sequence in which it is created.
- Tables are always registered against a version.
- A table registered against an older version can be inherited by a newer version if the option to inherit tables is enabled during the project creation.
- Table definitions are specific to a version. However, newer versions can inherit them during the upgrade version process.

Add More Versions to an Existing Project

If you have a requirement to add more versions to a project after the project is created, you can do so.

1. Access the CA TDM Portal.
2. Select the Project Manager icon (gear icon) in the top blue bar.
   The **Manage Projects** dialog opens.
3. Identify the project for which you want to create a version.
4. Click the plus icon (+) in the row corresponding to the project for which you want to create a version.
   The **Create New Version** page opens.
5. Provide the following information for the version that you want to create:
   - **Name**
     Lets you specify the name of the version.
   - **Description**
     Lets you specify the meaningful information about the version.
   - **Upgrade Version**
     Lets you copy information (for example, data definitions, variables) from an older version to the version that you are creating. When you select this option, the following option becomes available in the CA TDM Portal:
     - **Upgrade From**
       Lets you specify the version from which you want to copy the information to the version that you are creating.
6. Click **Save**.
   A message appears after the version is created successfully.
7. Click the forward arrow (>) before the project name (**Manage Projects** dialog).
   The project view is expanded and lists all the applicable versions for the project.
8. Verify that the newly created version is present in the list.
   You have successfully added a new version to an existing project.

Perform Actions on Versions

You can perform the following actions on a version after you create it:

- Edit a Version.
- Upgrade a Version.
- Delete a Version.

Note: For more information about creating variables at a version level, see Create and Manage Variables.

Edit a Version

After you create a version, you can edit its name and description if you want to do so.

1. Access the CA TDM Portal.
2. Select the Project Manager icon (gear icon) in the top blue bar.
   The **Manage Projects** dialog opens.
3. Identify the project that includes the version that you want to update.
4. Click the forward arrow (>) before the project name to view all the available versions for the project.
5. Click the row corresponding to the version that you want to update.
   A dialog with all the version details opens.
6. Move the mouse pointer to the following places in the dialog:
   - **<Version_Name>**
     Click the Edit icon (pencil) and specify a new name for the version. Then, click the Save icon to save the updated name.
   - **Description**
     Click the Edit icon (pencil) and specify a new description for the version. Then, click the Save icon to save the updated description.
7. Review the updated information.
   You have successfully edited a version.

Upgrade a Version

Upgrading a project version copies the following information from an older version (source version) to a newer version (target version):

- Copies data generators if data generators with the same name do not exist in the target version.
- Copies data generation rules.
- Copies variables at various levels if variables with the same name do not exist in the target version.
- Copies CA Agile Requirements Designer flows if flows with the same name in the same hierarchy do not exist in the target version.
- Copies publish actions if the corresponding data generator is not present in the target version.

Upgrade is applicable only for those projects for which the **All new versions inherit tables from previous version** option was enabled during the project creation. This option is available in the project creation page. Additionally, you can upgrade only a normal version, not a generic version.

Note: Registered tables and foreign keys are not copied during the version upgrade.

1. Access the CA TDM Portal.
2. Select the Project Manager icon (gear icon) in the top blue bar.
   The **Manage Projects** dialog opens.
3. Identify the project that includes the version that you want to upgrade.
4. Click the forward arrow (>) before the project name to view all the available versions for the identified project.
5. Identify the version (target version) that you want to upgrade with the information from a previous version (source version).
6. Click the Upgrade Version icon (up arrow) in the row that corresponds to the version that you want to upgrade.
   The **Upgrade Version <Version_Name>** dialog opens.
   Note: No Upgrade Version icon is available for a generic version (if it exists) and for an initial version of a project. For example, if `Version 1` is the first version in the project, the upgrade icon is not available for `Version 1`. Similarly, if `Version 2` is a generic version, then `Version 2` appears before `Version 1` in the versions list and the `Version 2` row does not display the upgrade icon.
7. Select the source version (from which you want to inherit the information) from the **Upgrade From** drop-down list.

This drop-down list displays all versions that you had already created before this version (which you are upgrading). For example, if you are upgrading version `6.0`, then all versions that are created before version `6.0` become available for selection from this drop-down list.

8. (Optional) Select the **Remove Existing Data** option to remove the existing data from the target version; that is, the version that you are upgrading. Otherwise, the data in the target version is merged with the data from the source version.
9. Click the **Upgrade** button.
   A message appears at the top of the dialog. You have successfully upgraded a project version.

Delete a Version

If you no longer need a specific project version, you can delete it from your CA TDM Portal environment. You must have appropriate privileges to delete a version. The CA TDM Portal also deletes data generators that are available under the version when you delete it (version).

1. Access the CA TDM Portal.
2. Select the Project Manager icon (gear icon) in the top blue bar.
   The **Manage Projects** dialog opens.
3. Identify the project that includes the version that you want to delete.
4. Click the forward arrow (>) before the project name to view all the available versions for the project.
5. Click the cross icon (X) in the row that corresponds to the version that you want to delete.
   A confirmation dialog opens.
6. Click **Delete** to proceed with the delete process.
   A message appears after the successful deletion of the version.
   Note: You cannot delete a version if the jobs associated with the version are in the running state.
7. Review the versions list to verify that the deleted version is no longer available in the table.
   You have successfully deleted a version.

Create and Edit Connection Profiles

A connection profile is a way of storing the details about a connection to a database system. Datamaker and other programs in the Test Data Manager suite can connect to any number of databases to manipulate and generate data. A set of profiles allows you to store the details of different connections.

Use the CA TDM Portal to create connection profiles for Microsoft SQL Server, DB2, MySQL, MariaDB, Oracle, Sybase, PostgreSQL, and Teradata databases. CA TDM Portal does not support connection profiles using DSN type that were created in CA TDM Datamaker. Profiles created in the CA TDM Portal are compatible with both TDM Portal and Datamaker.

You can also share a connection profile with a group. When a connection profile is shared with a group, all users associated to that group can then access the shared connection profile.

> **NOTE**
>
> Connection Profiles are shared between Test Data Reservation Service and Data Discovery using CA TDM Portal. For more information about how to use Connection Profiles with Data Discovery, see The Data Model in CA TDM Portal.

Considerations

Review the following considerations for connection profiles:

- Super administrators and users who have access to the "Settings" security function can create a connection profile.
- Only two types of user can edit or delete a connection profile:

  - The owner (i.e. creator) of that connection profile. This user must have the '**Settings**' privilege.
  - Super administrators who are a member of a group with which that connection profile is shared.
- Super administrators, project administrators (users who are part of the "Admin" group for a project), and users who have access to the "Users and Groups" security function can share their connection profiles with user groups that they can manage. Also, only these users can view the **User Groups** section in the Portal; this section is not visible to other users.

  WARNING

  If you are an administrator, connection profiles that you share with a group are available to users, but these users cannot see the connection profiles in the Connection Profiles section of the UI, and are not able to edit or delete them.
- Users can access and use those connection profiles that they have created. Additionally, they can also access those connection profiles that are shared with a group of which they are a member.
- Super administrators and project administrators can create, update, and delete user groups for their project.
- Connection profiles that are already available in Datamaker become accessible in the Portal. Similarly, connection profiles that you create in the Portal are accessible in Datamaker.
- All the group connection profiles in Datamaker become available as shared connection profiles in the Portal; however, they are not editable in the Portal.
- Shared connection profiles in the CA TDM Portal show up as a single user connection profile in Datamaker.
- The CA TDM Portal now enforces that the connection profile names must be unique.

  TIP

  If you created connection profiles in Datamaker and in earlier versions of the CA TDM Portal, you might see two connection profiles with the same name in your list. In that case, rename your connection profile to resolve the conflict.
- In scenarios where a connection profile that is shared with a group is used in both the interfaces—CA TDM Portal and Datamaker, we recommend that you create this connection profile in Datamaker and then use it across both the interfaces, as required.

Create a Connection Profile

You can create a new Connection Profile from the CA TDM Portal. Follow these steps:

1. Access the CA TDM Portal.
2. Expand **Configuration** in the left panel.
3. Click **Connection Profiles**.
   The **Connection Profiles** page opens.
4. Click **New Profile**.
   The **Add New Connection Profile** page opens.
5. Enter required information in the following fields:
   - **Profile Name**
     Specifies a unique name for the connection profile that you are creating.
   - **Description**
     Specifies an appropriate description for the connection profile that you are creating.
   - **DBMS**
     Lets you select the type of the database you want to connect to.
     Note: Input fields vary depending on the type of the data source you select.
   - **Server**
     Specifies the host name or the address of the server where the database is available.
   - **Database**

Specifies the name of the database that you want to use as a source or target.

- **User Name**
  Specifies the name of the user who can access the database.**Note:** For Microsoft SQL Server, you can now select the Integrated Security option instead of providing a user name and password.
- **Password**
  Specifies the password for the database user.
- **Port**
  Specifies the port number where the database is running on the server.
- **Oracle Service Name**
  (For Oracle) Specifies the name of the Oracle service. TNS names are not supported.
- **SQL Server Instance Name**
  (For Microsoft SQL Server) Specifies the name of the Microsoft SQL Server instance.
- **SQL Server Schema Name**
  (For Microsoft SQL Server) Specifies the Microsoft SQL Server database schema name.
- **Additional Connection Properties**
  (For DB2/OS, DB2/AS400 and Teradata) Lets you specify additional connection properties. Add key/value pairs separated by semi colons (;). An equals sign (=) separates each key from its value or values (separate values with commas).For example,

  ```
  libraries=Schema1,Schema2,Schema3
  ```

  causes the Connection Profile only to include the schemas Schema1, Schema2 and Schema3.

  NOTE

  These properties are specific to your database type. See your specific database documentation for a comprehensive list of properties.

6. (Optional) Share the connection profile that you are creating with a group as follows:
   a. Click the **Share to User Group** button.
      The **Select User Groups** dialog opens.
   b. Select an appropriate group from the list and click **Add**. You can also select multiple groups.
      The selected group is added to the **User Groups** table.
   c. To delete a specific group from the connection profile, click the Remove icon (X icon) in the row corresponding to the identified group, and confirm the deletion. The group is removed from the list and is no longer associated with the connection profile.
7. Click **Test** to verify that the connection profile is able to establish the connection.
8. Click **Save** if the test is successful.
   The **Connection Profiles** page opens.

Edit Connection Profile

You can edit Connection Profiles from the CA TDM Portal. Follow these steps:

1. Access the CA TDM Portal.
2. Expand **Configuration** in the left panel.
3. Click **Connection Profiles**.
   The **Connection Profiles** page opens.
4. Click a Connection Profile.The **Editing <Connection Profile name>** page opens. You can now change the following properties:
   - **Description**
   - **Profile properties**You can change the properties of this Connection profile. Different database management systems have different properties. Server and Port are common to all database management systems

WARNING

In Oracle / SQL Server DBMS, you have the option to **Use a specific DBMS Schema**. This property is case-sensitive.

If you click **Test** to test the database connection, this does not test the connection to the specified schema. The test may return the message 'The Connection works OK', but the schema may be invalid.

- **User Groups**

5. Click **Test** to verify that the connection profile is able to establish the connection.
6. Click **Done** if you are happy with the changes.
   The **Connection Profiles** page opens.

Select Source and Target Profiles

The **Connection Profiles** page displays only those connection profiles that you have created. If you are a member of a user group and a connection profile is shared with that group, then you can access that connection profile from all the appropriate places in the Portal. However, that connection profile is not listed on the **Connection Profiles** page for you, because you are not the owner of that connection profile.

A source is typically a database from which you want to model and interact with data. A target is typically a database to which you are publishing data.

- You can only have one source and one target active at the same time.
- Source and target can refer to the same profile.
- You can switch a profile from source to target, or from target to source any time.

The currently selected source and target profiles are displayed at the top of the page. Click the source or target name to jump down to the line in the list where this profile is defined.

To select a different source and target profile, enable the checkbox in the Source or Target column, respectively.

Create an Environment

An environment is a collection of data sources that are associated with an application. Each data source maps to a connection profile. Testers use the environment to find the test data from multiple data sources and then reserve the required test data from the respective data sources.

Follow these steps:

1. Access the CA TDM Portal as a TDE.
2. Ensure that you select the required project and version from the **Project** drop-down list.
3. Click **Modeling** in the left pane.
4. Click **Environments**.
   The **Environments** page opens.
5. Click **New Environment**.
   The **New Environment** dialog appears.
6. Enter the following information:
   - **Name**
     Specifies an appropriate name for the environment.
   - **Description**
     Specifies an appropriate description for the environment.
   - **Add Data Source**
     Click this button to add data sources and specify the following information. You can add multiple data sources to an environment:

- **Data Source Name**
  Specifies the database name that sources the data you want to find. Enter the database name in this text box.
- **Connection Profile**
  Specifies the connection profile name that is mapped to the specified data source. Select an applicable connection profile from the drop-down list.

7. Click **Save**.
   The environment is successfully created and added to the list on the **Environments** page. You can edit or delete an environment as necessary.

Edit an Environment

Identify the environment you want to edit from the list and click the Edit icon in the line corresponding to the environment. Modify the values in the **Edit Environment** dialog and click **Save**.

Delete an Environment

Identify the environment you want to delete from the list and click the Delete icon in the line corresponding to the environment. You cannot undo the delete action; the respective environment is deleted from the **Environments** page under Test Data Models.

Perform Data Discovery on an Environment

Identify tables and table relationships in an environment on which you want to perform a Data Discovery scan. For more information about Data Discovery, see The Data Model in CA TDM Portal.

Register and Manage Relational Schema

In the CA TDM Portal, you register objects so that you can perform various data manipulation operations (for example, data generation) on them. You register objects in context of a project and its version.

The CA TDM Portal supports registering and managing the following object types:

- Relational Schema (Database Tables)
- Non-relational Data Sources (XSD, XML, WSDL, JSON, RR Pair). For more information about how to register non-relational data source objects, see Prepare Test Data for Non-Relational Data Sources.

The high-level process to register different object types remains the same; only a few options differ based on the type you select.

Follow these steps to register Relational Schema:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the Project drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
   Note: If the project is not available, create a new project.
3. Click the **Modeling** options in the left pane to expand it.
   Note: If the left pane is hidden, click the icon (represented by three horizontal bars) in the top left corner to view the pane.
4. Click the **Objects** option to view the available options.
5. Click the **Register New Object(s)** button.
6. The **Register New Object(s)** page opens.
7. Enter information in the following fields:
   - **Object Type**
     Lets you select the type of the object that you want to register. Select Table from the drop-down list.
   - **Connection Profile**

Specifies the connection profile that associates the database to register tables from.

- **Schemas**
  Lets you specify the schema within the database that the selected connection profile points to. Select an appropriate schema.
- **Tables**
  Lets you specify the tables included in the selected schema. Select all the tables that you want to register.
  Note: Registering tables from a Teradata database with special characters in table or column names fails. Ensure that all characters are UTF-8 encoded.

8. Click **Register.**
   The Registered Objects page opens with the newly registered tables added to the list of registered objects.
   You have successfully registered the data tables.
   Note: To delete a registered object, click the cross icon (X) for the required object and confirm the deletion. You can also select multiple objects and delete all of them at once. A delete job is created and is added to the jobs queue. You can view the jobs queue in the Requests table by clicking the request ID in the message that is displayed. When the status of the job is shown as Completed, the registered object is deleted. You can also click the appropriate row to view the additional information about the job. The additional information is displayed in the **Additional Information** dialog.
9. You can now publish the data to relational tables. Review Publish Data Using the CA TDM Portal section, for detailed steps to publish data to target database schema.

Prepare Test Data for Non-Relational Data Sources

As a test data engineer, review this section to understand how the CA TDM Portal helps you generate data to test applications that rely on non-relational data sources. Many applications use various file objects (such as XSD, XML, G-T Excel, CSV, JSON, or Request-Response Pairs) as their data sources, schema definitions, or media formats for data transfer. The CA TDM Portal lets you work with these file objects and create test data that applications can use to conduct varied testing scenarios.

The CA TDM Portal supports the following file objects:

- XSD
- XML
- WSDL
- JSON
- Request-Response Pair (RR Pair)
- G-T Excel
- CSV

NOTE

Ensure that you are aware of the appropriate generator, and connection profile that you want to use to prepare test data for non-relational data sources.

CA Service Virtualization Integration

One key use case for working with non-relational data is using RR pairs to generate data for data-driven virtual services in CA Service Virtualization. For more information about setting up this integration and a detailed example, see Integration with CA Service Virtualization.

Tutorial Video

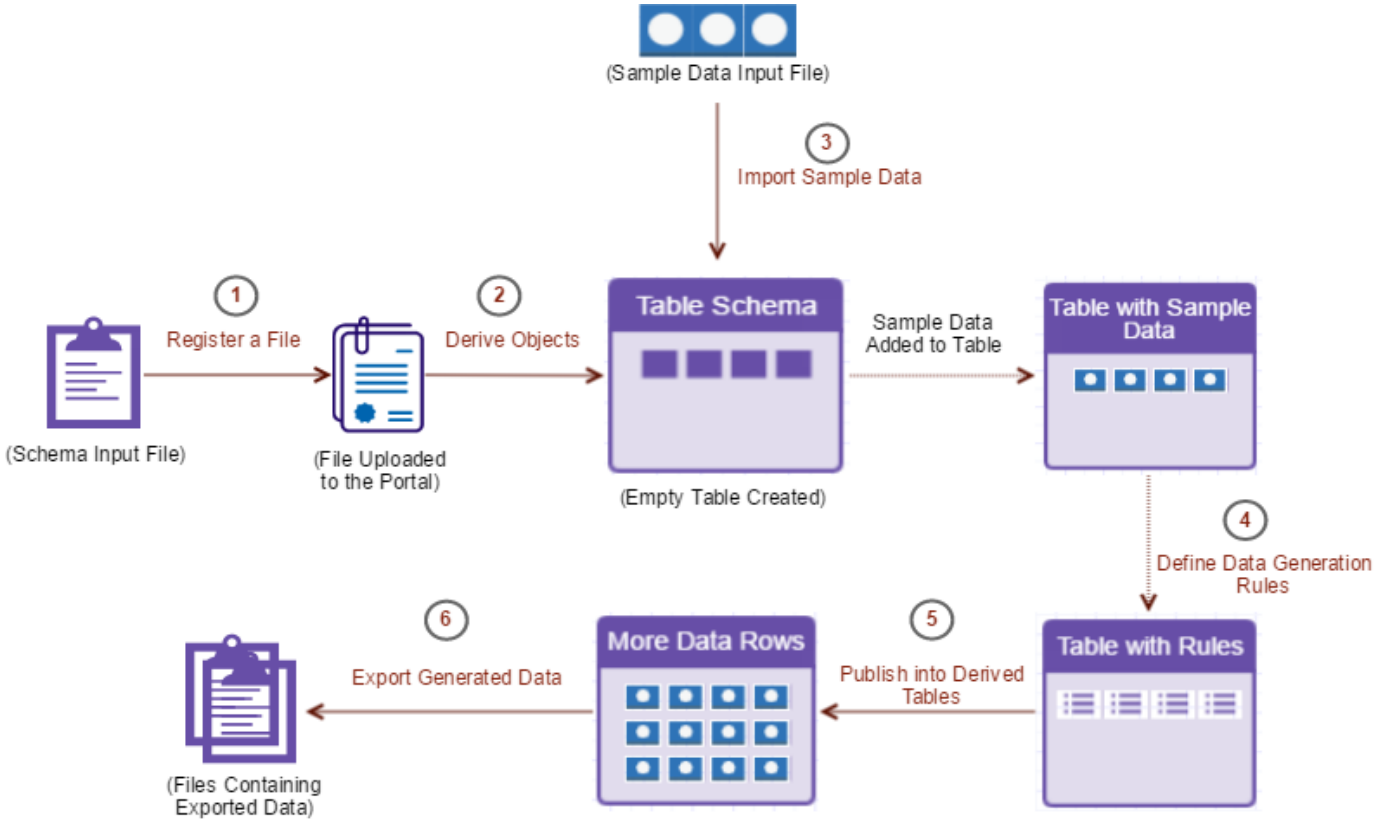
Watch the following video for a visual walkthrough of a common use case of importing an XML schema and generating test data that you export to XML files:

How to prepare XSD, XML, WSDL, JSON, RR Pair File Types

Prepare test data

The following illustration shows the high-level process to prepare test data for XML, XSD, JSON, WSDL, and Request-Response Pair (RR Pair) file object types. This functionality is applicable only for the Microsoft SQL Server connection profiles.

Figure 23: XSD_XML_FileTypes



To prepare test data for XML, JSON, WSDL, XSD, and RR Pair file object types, follow these steps:

1. Register file objects.
2. Create and register derived objects.
3. Import sample data into derived objects.
4. Define data generation rules.
5. Publish data into derived tables.
6. Export the generated data from derived objects into appropriate file formats.

You can also perform other actions on derived objects. You can delete data from all the derived objects or drop all the derived objects.

Request-Response Pairs

An 'RR pair' represents a request-response pair in the form of XML, JSON or text files. You can use the request and response files to create a relational schema. You can then use the same RR pair files to import the sample data into

the relational schema. Additionally, note that RR pair files using .txt extension contain information in the form of HTTP headers and body, thereby providing support for REST format. For more information about the structure of .txt RR files, see the REST RR Pair Format section.

The following video provides an example of this process using an XSD schema to create XML files for testing:

Register File Objects

In the CA TDM Portal, you register file objects so that you can perform various data manipulation operations (for example, data generation) on them. You register file objects in context of a project and its version.

This procedure provides information about how to register the following file object types:

- XSD
- XML
- WSDL
- JSON
- RR Pair

Note: This procedure is applicable only for the aforementioned file object types. For more information about other file object types (for example, GTExcel and CSV), see the appropriate section.

The high-level process to register different file object types remains the same; only options differ based on the type you select.

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
3. Expand **Modeling** in the left pane and click **Objects**.
   The **Objects** page opens. This page lists all the objects that are registered to the selected version and the project. If no object is registered, nothing is listed on the page.
4. Click the **Register New Object(s)** button.
   The **Register New Object(s)** page opens.
5. Select the type of the file object from the **Object Type** drop-down list. You can select from the following file object types:
   – XSD
   – WSDL
   – XML
   – R/R PAIR
   – JSON
   Note: For the RRPAIR file object, you can provide RR pairs in the form of .xml, .json, and .txt files. RR pair files using .txt extension contain information in the form of HTTP headers and body, thereby providing support for REST format. For more information about the structure of .txt RR files, see the REST RR Pair Format section.
6. Enter an appropriate name for the file object that you want to register in the **Name** field.
7. Specify the location from where you want to get the object file. Appropriate fields are displayed depending on the file object type that you select:
   – **File(s) to Upload**
   Lets you specify the local location where the object file is available. You can browse to the location or drag and drop the file.
   This field is displayed for all the object types except RRPAIR.
   Note: For XSD and WSDL object types, you can also specify a .zip file that includes XSD or WSDL files (as appropriate). When you specify the location of a .zip file, the **Root File Name** field is displayed. In this field, you specify the location (relative) of the root file that you want to use for creating and registering derived objects.
   – **Object URL**

(Only WSDL) Lets you specify the remote URI location where the object file is available.

- **Request-File to Upload**
  (Only RRPAIR) Lets you specify the local location where the request object file (.xml, .json, or .txt) is available. You can browse to the location or drag and drop the file.
- **Response-File to Upload**
  (Only RRPAIR) Lets you specify the local location where the response object file (.xml, .json, or .txt) is available. You can browse to the location or drag and drop the file.
  Note: Ensure that the request and response files are of the same type.

8. Click the **Advanced Settings** option and enter appropriate information in the following fields:
   - **File Encoding**
     Specifies the file encoding format that you want to use.
     Default: UTF-8
     Note: The CA TDM Portal supports all encoding formats that Java supports. However, it is tested and certified for these encoding formats:
     For XML, US-ASCII, ISO-8859-1, UTF-8, UTF-16BE, UTF-16LE, UTF-16.
     For JSON, UTF-8, UTF-32BE, UTF-16BE, UTF-32LE, and UTF-16LE.
   - **No Namespace Schema Location**
     (Only for XSD and XML) Specifies the location of the schema definition file (XSD) that does not have a target namespace. This information is used for the noNamespaceSchemaLocation attribute, which references an XML schema document that does not have a target namespace. The following is an example value for this field:
     ```
     http://abx21yz.com/schemas/purchase12.xsd
     ```
     This attribute value is included in the exported XML documents (when you perform the export operation). The following example shows how this attribute is added to the exported XML document:
     ```
     xsi:noNamespaceSchemaLocation="http://abx21yz.com/schemas/purchase12.xsd"
     ```
     Note that the attribute value includes only one part, which is the location of the XML schema.
   - **Schema Location**
     (Only for XSD and XML) Specifies the location of the schema definition file (XSD) that has a target namespace. This information is used for the schemaLocation attribute, which references an XML schema document that has a target namespace. The following is an example value for this field:
     ```
     http://dpou123xy.com/Order http://dpou123xy.com/schemas/order.xsd
     http://dpou123xy.com/schemas/Purchase http://dpou123xy.com/schemas/Purchase.xsd
     http://dpou123xy.com/schemas/Client http://dpou123xy.com/schemas/Client.xsd
     ```
     This attribute is included in the exported XML documents (when you perform the export operation). The following example shows how this attribute is added to the exported XML document:
     ```
     xsi:schemaLocation=
                      "http://dpou123xy.com/Order http://dpou123xy.com/schemas/order.xsd
                       http://dpou123xy.com/schemas/Purchase http://dpou123xy.com/schemas/Purchase.xsd
                       http://dpou123xy.com/schemas/Client http://dpou123xy.com/schemas/Client.xsd"
     ```
     Note that the attribute value includes two parts that are separated by a space. The first part represents the namespace. The second part represents the location of the XML schema that describes the specified namespace.
   - **Namespaces**
     (Only for XSD and XML) Specifies the namespaces defined in the XML schema document that you want to include in the exported XML documents. Use a semicolon (;) to separate multiple values. The following is an example value for this field:
     ```
     http://www.abc90ef.com/store;http://www.abc90ef.com/location
     ```
     This attribute is included in the exported XML document (when you perform the export operation). The following example shows how this attribute is added to the exported XML document:
     ```
     xmlns:tdmns0="http://www.abc90ef.com/store" xmlns:tdmns1="http://www.abc90ef.com/location"
     ```
     Note: The namespaces referred by the XML elements in the exported XML documents are added by default. Use this attribute to mention any explicit namespaces to be included in the XML document.
9. Click the **Save** button.

The **Objects** page opens with the the newly registered object added to the list of registered objects.
You have successfully registered an object.
Note: To delete a registered object, click the cross icon (X) for the required object and confirm the deletion. A delete job is created and is added to the jobs queue. You can view the jobs queue in the requests table by clicking the request ID in the message that is displayed. When the status of the job is shown as **Completed**, the registered object is deleted. You can also click the appropriate row to view the additional information about the job. The additional information is displayed in the **Additional Information** dialog.

For the XML, XSD, WSDL, JSON, and RR Pair file objects that you register, you must perform actions on the data.

Additionally, if you want to create a data generator in context of the selected project and version, click the **Create Generator** button in the **Objects** page and follow the steps to create a generator.

Create and Register Derived Objects

Derived objects represent relational tables that you create out of the registered file objects. File objects (XML, XSD, WSDL, RR Pairs, and JSON) include non-relational data. You cannot directly work with these file objects that contain non-relational data in the CA TDM Portal. To work with them, convert the non-relational model into a relational model and store it in the relational database.

Therefore, by creating derived objects, you achieve the following objectives:

- Convert the non-relational data model into a relational model by creating tables in the relational database.
- Register created relational tables with the CA TDM Portal.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Modeling** in the left pane.
   The **Modeling** option expands.
4. Click **Objects**.
   The **Objects** page opens. This page lists objects that are registered to the selected project and the version.
5. Click the file object that you want to use to create derived tables.
   The **<Object_Name>** page opens. Appropriate options based on the object type that you selected appear in the **Derived Tables** section.
6. Provide information depending on your object type.
   a. Enter information for the following common options that appear for all object types:
      - **Connection Profile**
        Specifies the appropriate connection profile to use for creating derived tables.
        Note: Users can access only those connection profiles that they have created. They cannot access connection profiles created by other users.
      - **Advanced Settings**
        Lets you provide information for the advanced options. Click the **Advanced Settings** option and specify the following information:
      - - **Generate Foreign Key Constraints**
          Lets you generate foreign keys for the derived tables when you select this option. The foreign keys information helps you identify relationships between different derived tables.
        - **Table Prefix**
          Lets you specify a prefix to add to the names of the derived tables that are created in the database.
        - **Duplicate Table Suffix**
          Lets you add a meaningful suffix to derived object (table) names to distinguish among the same names. Whenever multiple derived objects are going to get created with the same name, you can use this

suffix to differentiate among them. For example, if your schema has two billTo tables, then without this option, you get billTo and billTo_1. However, if you enable this option, you get billTo and billTo_tdm_1.
Default: _tdm_1

- **Reconcile tables**
  Lets you resolve conflicts with the existing registered tables. When you slect this option, the CA TDM Portal unregisters the existing conflicting tables from the version and then adds the new derived objects. When the conflicting tables no longer remain registered with the version, the CA TDM Portal lets you proceed with the operation. In this case, all the data generation rules that are added to the existing tables are moved to the derived objects. For more information about resolving conflicts, see Resolve Conflicts (Derived Objects Creation/Registration).

b. Click the appropriate link to provide information for the remaining options that appear only for a specific object type:
   - XSD
   - WSDL
   - XML
   - RRPAIR
   - JSON

7. Click **Create and Register**.
   The CA TDM Portal creates a job for this operation and adds it to the jobs queue. You can view the jobs queue in the requests table by clicking the job ID in the message. The requests table displays all the job requests with their status, date, name, and other relevant information. When the status of the job is shown as Completed, the CA TDM Portal completes the task of deriving tables out of the selected object type and registering them. You can click the required row in the requests table to view the additional information about the job, if necessary. The additional information is displayed in the **Additional Information** dialog.

You can now perform different actions on the derived tables. These actions are available at the top of the list where derived tables are displayed.

Note: You can also create a data generator from this view. The ability to create a data generator in context of a project and its version improves the overall workflow. You no longer need to navigate out of your selected project or version view to create a generator for the same project or version. In this case, click the **Create Generator** button and follow the remaining instructions in Create Data Generator.

XSD

For XSD, provide the following information:

Note: To edit the noNamespaceSchemaLocation, schemaLocation, and namespaces information, expand the **Advanced Settings** option present above the **Derived Tables** section.

- **Root Element**
  Specifies the root element to use for creating derived tables.
- **Advanced Settings**
  Click the **Advanced Settings** option and specify the following information:
  - **Cyclic Recursion Depth**
    Lets you specify the required recursion depth when you have cyclic references in the XSD that you want to use to create derived tables. The supported recursion depth range is from 1 through 32.
    Default: 2

WSDL

For WSDL, provide the following information:

- **WSDL Operation**

Lets you specify the WSDL operation name for which you want to create derived tables.

- **Advanced Settings**
  Click the **Advanced Settings** option and specify the following information:
  - **Cyclic Recursion Depth**
    Lets you specify the required recursion depth when you have cyclic references in the XSD that you want to use to create derived tables. The supported recursion depth range is from 1 through 32.
    Default: 2

XML

For XML, provide the following information:

Note: To edit the noNamespaceSchemaLocation, schemaLocation, and namespaces information, expand the **Advanced Settings** option present above the **Derived Tables** section.

- **Import Object Data**
  Specifies whether you want to import data into the derived tables after they are created. This option uses the same XML file that you use for creating derived objects to import data. The CA TDM Portal populates the relational database with data from the provided XML file.
  When you select this option, the **Document Group ID** field is displayed. Enter the appropriate ID that you want to associate with the data that you are importing into derived objects. This ID helps you group the data, which is useful during the export process. The document group ID that you specify at the time of importing the data into derived objects can be used during the export process to group the data based on the same ID. You can then export only that grouped data. This ability, therefore, helps you filter only the required data during the export process. That is, instead of exporting all the data, you can simply use this ID to group the required data and then export only that set of relevant data.
  For import, the value can be a string or integer without a comma (,) and hyphen (-).
- **Advanced Settings**
  Click the **Advanced Settings** option and specify the following information:
  - **Cyclic Recursion Depth**
    Lets you specify the required recursion depth when you have cyclic references in the XSD that you want to use to create derived tables. The supported recursion depth range is from 1 through 32.
    Default: 2

The CA TDM Portal follows the Russian Doll design approach to convert the registered XML file into relational tables. For more information about the Russian Doll design approach, see Introducing Design Patterns in XML Schemas. Derived tables of already registered XML files that come from the previous CA TDM Portal releases (prior to 4.0) work properly in this release, too. However, if you register the same XML file object to the same project version in this release and derive tables, the structure of the derived tables might change. For example, the number of derived tables or columns in the derived tables might increase or decrease depending on the updated scenario.

RRPAIR

For RR Pair, provide the following information:

- **Import Object Data**
  Specifies whether you want to import data into derived tables after they are created. This option uses the same request and response .xml, .json, or .txt files that you use for creating derived objects to import the data. The CA TDM Portal populates the relational database with the data from the provided request and response .xml, .json, or .txt files.
  Note: RR pair files using .txt extension contain information in the form of HTTP headers and body, thereby providing support for REST format. For more information about the structure of .txt RR files, see the REST RR Pair Format section.
  When you select this option, the following options are displayed:
  - **Document Group ID**

Specifies the appropriate ID that you want to associate with the data that you are importing into derived objects. This ID helps you group the data, which is useful during the export process. The document group ID that you specify at the time of importing the data into derived objects can be used during the export process to group the data based on the same ID. You can then export only that grouped data. This ability, therefore, helps you filter only the required data during the export process. That is, instead of exporting all the data, you can simply use this ID to group the required data and then export only that set of relevant data.
For import, the value can be a string or integer without a comma (,) and hyphen (-).

- **R/R PAIR Link ID**
  Specifies an alphanumeric request-response link ID that identifies the associated request-response pair. This ID establishes a link between the request file and the response file. The value can be a string or integer without a comma (,) and hyphen (-).

- **Advanced Settings**
  Click the **Advanced Settings** option and specify the following information:
  - **Cyclic Recursion Depth**
    Lets you specify the required recursion depth when you have cyclic references in the XSD that you want to use to create derived tables. The supported recursion depth range is from 1 through 32.
    Default: 2

JSON

For JSON, provide the following information:

- **Import Object Data**
  Specifies whether you want to import data into the derived tables after they are created. This option uses the same JSON file that you use for creating derived objects to import data. The CA TDM Portal populates the relational database with data from the provided JSON file.
  When you select this option, the **Document Group ID** field is displayed. Enter the appropriate ID that you want to associate with the data that you are importing into derived objects. This ID helps you group the data, which is useful during the export process. The document group ID that you specify at the time of importing the data into derived objects can be used during the export process to group the data based on the same ID. You can then export only that grouped data. This ability, therefore, helps you filter only the required data during the export process. That is, instead of exporting all the data, you can simply use this ID to group the required data and then export only that set of relevant data.
  For import, the value can be a string or integer without a comma (,) and hyphen (-).
- **Advanced Settings**
  Lets you provide information for the advanced options. Click the Advanced Properties option and specify the following information:
  - **Allow Comments**
    Specifies whether the JSON parser allows the use of Java or C++ style comments (both '/'+'*' and '//' varieties) in the JSON content.
  - **Allow Non-numeric Numbers**
    Specifies whether the JSON parser recognizes a set of "Not-a-Number" (NaN) tokens as valid floating number values in the JSON content.
  - **Allow Back Slash Escaping**
    Specifies whether the JSON parser allows the use of backslash to escape any character in the JSON content. If you do not enable this property, only those characters that JSON specification supports are escaped.
  - **Allow Single Quotes**
    Specifies whether the JSON parser allows the use of single quotes (apostrophe, character '\'') for mentioning strings names and string values in the JSON content. For example, 'name' : 'value'.
  - **Allow Unquoted Control Characters**

Specifies whether the JSON parser allows JSON strings to contain control characters without quotes in the JSON content.

– **Allow Unquoted Field Names**
Specifies whether the JSON parser allows the use of field names without quotes in the JSON content. For example, name : "value".

– **Allow Numeric Leading Zeros**
Specifies whether the JSON parser allows the integer numbers to start with additional zeroes (for example, 005) in the JSON content.

Resolve Conflicts (Derived Objects Creation/Registration)

The CA TDM Portal helps you address table conflicts that come up during the creation and registration of derived objects in the same version.

At the time of creating and registering a derived object, the CA TDM Portal verifies whether any conflict exists with the existing registered tables or derived objects in that version. If a conflict exists, the CA TDM Portal highlights the issue and provides ways to help how you resolve the conflict.

Scenario 1: Conflict with Existing Registered Tables

If a conflict is because of the existing registered tables, you can perform one of the following tasks to resolve the conflict:

- Add a unique prefix to the derived objects that you are creating and registering.
  This approach suggests you to add a unique prefix to the derived objects. The unique prefix resolves the conflict and you can then start the operation.
- Enable the **Reconcile tables** option in the CA TDM Portal.
  This approach unregisters the existing conflicting tables from the version and then adds the new derived objects without any issue. When the conflicting tables no longer remain registered with the version, the CA TDM Portal lets you proceed with the operation. In this case, all the data generation rules that are added to the existing tables are moved to the derived objects.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Modeling** in the left pane.
   The **Modeling** option expands and displays two options: **Objects** and **Variables**.
4. Click **Objects**.
   The **Objects** page opens.
5. Click the **Register New Object(s)** button.
   The **Register New Object(s)** page opens.
6. Follow the steps to register a new file object to the selected project and version.
   The **Objects** page opens. This page now shows the file object that you have registered.
7. Click the file object.
   The **<Registered_File_Object_Name>** page opens.
8. Follow the steps to create and register derived objects.
   A request job is created and is added to the requests table.
9. Click the job ID in the message to view the job status.
10. Click the job row in the table to view the message.
    This job fails if conflicts with the existing registered tables exist.
11. Navigate to the **Objects** page for the version and select the conflicting file object.
12. Click **Advanced Settings** and do one of the following:

- Enter the prefix that you want to add to the derived objects in the **Table Prefix** field.
- Enable the **Reconcile tables** option to reconcile the registered tables.

13. Follow the remaining steps to create and register derived objects without any conflict.

Scenario 2: Conflict with Existing Derived Objects

If a conflict is because of the existing derived objects, you can perform one of the following tasks to resolve the conflict:

- Add a unique prefix to the derived objects that you are creating and registering.
  This approach suggests you to add a unique prefix to the derived objects. The unique prefix resolves the conflict and you can then start the operation.
- Delete the existing derived objects from the version.
  This approach suggests to manually delete the existing conflicting derived objects from the version. When conflicting derived objects are deleted from the version, the CA TDM Portal lets you proceed with the process.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Modeling** in the left pane.
   The **Modeling** option expands and displays two options: **Objects** and **Versions**.
4. Click **Objects**.
   The **Objects** page opens.
5. Click the **Register New Object(s)** button.
   The **Register New Object(s)** page opens.
6. Follow the steps to register a new file object to the selected project and version.
   The **Objects** page opens. This page now shows the file object that you have registered.
7. Click the file object.
   The **<Registered_File_Object_Name>** page opens.
8. Follow the steps to create and register derived objects.
   A request job is created and is added to the requests table.
9. Click the job ID in the displayed message to view the job status.
10. Click the job row in the table to view the message.
    This job fails if conflicts with existing registered tables exist
11. Navigate to the **Objects** page for the version and select the conflicting file object.
12. Click **Advanced Settings** and specify the prefix that you want to add to the derived objects in the **Table Prefix** field.
    If you do not want to add the prefix, delete the existing derived objects to resolve the conflict.
13. Follow the remaining steps to create and register derived objects without any conflict.

Upgrade Inherited File Objects

You can upgrade inherited file objects available in different project versions. When you create a new version after the first (initial) version, the new version inherits data from its previous version if the inherit option is enabled during project creation. In this case, the CA TDM Portal lets you use an updated file (data source) to upgrade the file objects that have been inherited from an earlier version. This scenario is helpful in situations where you want to update your existing schema because of the modifications made to the original file (data source) object. In this scenario, you do not go through the complete process from the start, because the Portal identifies and implements only the changes in the schema, not the complete information.

To achieve this, the CA TDM Portal provides an Upgrade icon in the row that includes the inherited object. When you click this icon, it prompts you to specify the updated file (data source). The CA TDM Portal evaluates the file, reconciles any conflicts, and updates the existing inherited file object with the new information.

For example, consider a scenario where version 1.0 is the first version. In this version, you create and register derived objects by using an XML file object (data source); for example, Abc1.xml. You now create a new version 2.0. The version 2.0 inherits all those derived objects from the previous version 1.0. These inherited objects are now available for the upgrade in version 2.0. That is, if your Abc1.xml file is changed and includes new information that you want to add to the derived objects inherited in version 2.0, you can do so by using the updated Abc1.xml file.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Modeling** in the left pane.
   The **Modeling** option expands and displays two options: **Objects** and **Variables**.
4. Click **Objects**.
   The **Objects** page opens. This page lists all the registered objects that are available for the selected project and version.
5. Locate the inherited file object that you want to upgrade.
   Note: You can identify whether an object is an inherited object by viewing the inherit icon before the object ID in the **ObjectID** column. This icon is represented by two square boxes joined with an arrow.
6. Click the Upgrade Object icon in the row corresponding to the inherited file object that you want to upgrade.
   The **Upgrade Object** dialog opens.
7. Navigate to the location where the updated file is available.
8. Verify the encoding format and change it if necessary.
9. Click the **Upgrade** button.
   A message appears at the top of the page. The inherited file object is upgraded with the new information.

Perform Actions on Derived Objects

After you create derived objects (tables) out of the registered file objects, you can perform the following actions on the derived objects:

Import Data into Derived Objects

Import the sample data into derived objects (tables) that you have created in the relational database. Use an appropriate file to import the sample data into derived objects. You can then add more data into these derived objects by using data generation rules. The import data options that are displayed in the **Import** dialog change based on the object type.

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Modeling** in the left pane.
   The **Modeling** option expands and displays two options: **Objects** and **Variables**.
4. Click **Objects**.
   The **Objects** page opens.
5. Click the file object that you want to use for the data import operation.

   NOTE

   You cannot import the data into inherited file objects.

   The **<Object_Name>** page opens. This page includes the list of derived objects.
6. Click the **Import Data** icon (up arrow).

NOTE

This icon becomes visible only when you create and register derived objects (tables).

The **Import Data** dialog opens.

7. Complete the following fields:
   - **Document Group ID**
     ID to assign to the data you import into derived objects. This value can be a string (excluding hyphens) or integer.

     TIP

     You can use this ID to filter data in derived objects, during the Export process.
   - **Advanced Settings**

   a. • **Connection Profile**
     Specifies the default Connection Profile that you used at the time of creating derived objects, from a drop-down list of available Connection Profiles.

     WARNING

     Only select a Connection Profile different to the one you chose at Step 2, if you are sure that this Connection Profile also contains the tables you want to import, and that you have access to this Connection Profile.
     - **Schema Name**
       Specifies the schema name that you want to use. This drop-down list is populated based on the Connection Profile that you select.
     - **Data Encoding**
       Specifies the encoding format of the XML (for XSD, XML, WSDL, RR Pair) or JSON file that you want to use for importing the sample data.
       Default: *UTF-8*

       NOTE

       TDM Portal supports all encoding formats that Java supports. However, only these encoding formats are tested and certified:

       For XML: US-ASCII, ISO-8859-1, UTF-8, UTF-16BE, UTF-16LE, UTF-16.

       For JSON: UTF-8, UTF-32BE, UTF-16BE, UTF-32LE, UTF-16LE.
   - **Import to Generator**
     Lets you import the sample data into the data generator. Select the required data generator from the **Data Generator Connection** drop-down list. This drop-down list is populated with all the data generator connections available to the selected Project, Project version, and Connection Profile.

     NOTE

     If you import the sample data into the data generator, you can use the **Registered Tables** (**Generators, Data Generators, <Generator_Name>, Select Tables**) dialog to verify whether required tables include the imported data.
   - (XSD, XML, or JSON only) **File(s) to Upload**Specifies the location of the file that contains the sample data that you want to import into the derived objects. You can drag and drop the file or browse to the file location.
   - (WSDL or RR Pair only) **R/R Pair Link ID**Specifies an alphanumeric request-response link ID that identifies the associated request-response pair. For import, the value can be an integer or string (without hyphens). For example, *100* or *RD*.

TIP

You can filter data by **R/R Pair Link ID** when you export derived objects.

- (WSDL or RR Pair only) **Request-File to Upload**Specifies the location of the request data .xml (applicable for WSDL and RR Pair), .json (applicable for RR Pair), or .txt (applicable for RR Pair) file. You use this file to populate the relational database with the sample request data included in the specified file. You can drag and drop the file or browse to the file location.

  NOTE

  Ensure that the request and response pair files are of the same type. In the case of WSDL files, both XML files must be of the same version of SOAP (1.1 or 1.2).
- (WSDL or RR Pair only) **Response-File to Upload**Specifies the location of the response data .xml (applicable for WSDL and RR Pair), .json (applicable for RR Pair), or .txt (applicable for RR Pair) file. You use this file to populate the relational database with the sample request data included in the specified file. You can drag and drop the file or browse to the file location.

  NOTE

  RR pair files using .txt extension contain information in the form of HTTP headers and body, thereby providing support for REST format. For more information about the structure of .txt RR files, see the REST RR Pair Format section.

8. Click the **Import** button.
   The CA TDM Portal creates a job for the import operation and adds it to the jobs queue.

You can view the jobs queue in the Requests table by clicking the job ID in the message. The requests table displays all the job requests with their status, date, name, and other relevant information. You can also click rows in the Requests table, to view additional information about that job in the **Additional Information** dialog.

Create Data Generation Rules

After you import the sample data into derived tables, create data generation rules. These rules help you generate synthetic data that you can publish into the derived tables.

Note: For more information about how to create data generation rules, see the Create Data Generation Rules section.

Publish Data into and Export Data from Derived Objects

You can publish the data into the appropriate target schema based on the defined data generation rules and then export the same data into required formats.

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Modeling** in the left pane.
   The **Modeling** option expands and displays two options: **Objects** and **Variables**.
4. Click **Objects**.
   The **Objects** page opens.
5. Click the file object that you want to use for the data publish and export operations.

   NOTE

   You cannot export data from inherited file objects.

   The **<Object_Name>** page opens. This page includes the list of derived objects.
6. Click the Export Data icon (down arrow).

NOTE

This icon becomes visible only when you create and register derived objects (tables).

The **Export Data** dialog opens.

7. Complete the following fields:
   - **Connection Profile**Specifies the connection profile that you want to use for the publish or export process.

     NOTE

     Users can access only those connection profiles that they have created. They cannot access connection profiles created by other users.
   - **Schema Name**
     Specifies the schema name that you want to use for the publish or export operation. This drop-down list is populated based on the connection profile that you select.
   - **Document Group ID**

   a. ID to assign to the data you import into derived objects. This value can be a string (excluding hyphens) or integer.

      TIP

      You can use this ID to filter data in derived objects, during the Export process.

      You can specify this value in the following ways:
      - Single document group ID. For example: *101*
      - Comma-separated list of document group IDs. Examples: *101,102,103* or *ID1,ID2,ID3*
      - Range of document group IDs. For example, *101-109*. You can use the range method only if your document group IDs are integers.

   - **Publish Files**Lets you generate and add more data to the derived tables in the database based on the defined data generation rules. When you select **Publish Files**, the following options appear:
     - **Data Generator Connection**
       Lets you select the appropriate data generator connection from the drop-down list. This list is populated with all the data generator connections based on the selected project, project version, and connection profile.
     - **No of Files to Publish**
       Lets you specify the number of rows that you want to add to the derived tables in the database.
   - (WSDL or RR Pair only) **R/R Pair Link ID**Specifies an alphanumeric request-response link ID that identifies the associated request-response pair. For import, the value can be an integer or string (without hyphens). For example, *100* or *RD*.

     TIP

     You can filter data by **R/R Pair Link ID** when you export derived objects.
   - (WSDL or RR Pair only) **Update Virtual Service**Specifies whether you want to update a virtual service in a CA Service Virtualization 9.1 environment with sample RR pair data. The integration with CA Service Virtualization helps you take advantage of the large volume of test data in the form of sample RR pairs that the CA TDM Portal generates. You can deploy your virtualized services with that test data and cover a wide range of testing scenarios. Using this integration, you can manage the following use cases:
     - Virtual service on demand
     - Increase test coverage
     - Data synchronized across inter-dependent systems and services
     - Up-to-date virtual services

     When you select the **Update Virtual Service** option, the following fields are displayed:
     - **Virtual Service Environment**

Specifies the name of the virtual service environment (VSE) to which you want to connect to add the sample RR pair data to a virtual service. This list is populated based on the virtual service configuration that you perform in Configure CA Service Virtualization Details.

- **Virtual Service**
  Specifies the name of the virtual service that you want to update with the sample RR pair data. This list is populated based on the virtual service configuration that you perform in Configure CA Service Virtualization Details.

– **Advanced Settings**

- **Data Encoding**Specifies the encoding format of the XML (for XSD, XML, WSDL, RR Pair) or JSON file that you want to use for importing the sample data.
  Default: *UTF-8*

  NOTE

  TDM Portal supports all encoding formats that Java supports. However, only these encoding formats are tested and certified:

  For XML: US-ASCII, ISO-8859-1, UTF-8, UTF-16BE, UTF-16LE, UTF-16.

  For JSON: UTF-8, UTF-32BE, UTF-16BE, UTF-32LE, UTF-16LE.

a. • **Base File Name**
  Specifies the value to prefix to the exported XML file name.
  Default: *CATDMShredder*

– • (XML or XSD only) **Export Into**Specifies whether to export the data into a single XML file or multiple XML files. Supported options are Single File or Multiple Files.

  NOTE

  You can only export XSD or XML data to XML files.

  Default: *Multiple Files*

- (XML or XSD only) **Require Data Indentation**Specifies whether to indent the XML data while exporting.
  Default: *Yes*
- (XML or XSD only) **Include XML Processing Instruction**Specifies whether to include the XML declaration in the exported XML file.
  Default: *Yes*
- (XML or XSD only) **Include Standalone Attribute**Specifies whether to include the generated standalone attribute value in the exported XML file.
  Default: *No*
- (XML or XSD only) **Honor Unqualified forms**Specifies the flag to honor the unqualified form for elements. If 'elementFormDefault' is 'undefined' for a schema (or an element has form=undefined), its XML instance files can have elements without namespace prefixes.
  Default: *Yes*
- *(JSON only)* **Export Into**Specifies whether to export the data into a single file or multiple files. Supported options are Single File or Multiple Files.
  Default: *Multiple Files*
- *(JSON only)* **Escape Non-ASCII**Specifies whether to escape non-ASCII characters at the time of exporting the data into a JSON file. If enabled, this property exports all characters outside the 7-bit ASCII range using the backslash character as an escape character.
  Default: *No*
- *(JSON only)* **Quote Field Names**Specifies whether you want to export JSON object field names using double quotes.
  Default: *Yes*
- *(JSON only)* **Quote Non Numeric Numbers**Specifies whether you want to export float values or double values as strings using double quotes.

Default: *Yes*

- *(JSON only)* **Write Numbers As Strings**Specifies whether you want to export all numbers as JSON strings. Default: *No*
- *(JSON only)* **Pretty Print JSON**Specifies whether you want to format/indent the JSON data while exporting. Default: *Yes*

8. Click the **Publish & Export** button.
   The CA TDM Portal creates a group job for the publish and export operation and adds it to the jobs queue. The group job includes two jobs—publish job and export job.

You can view the jobs queue in the requests table by clicking the job ID in the message. The requests table displays all the job requests with their status, date, name, and other relevant information. When the status of your job is shown as **Completed**, the CA TDM Portal completes the following tasks in sequence:

- Publish the data into the derived tables in the database.
- Export the published data from the derived tables into the specified file format.

You can use these exported files to test applications that rely on non-relational data sources.

Delete Data from Derived Objects

You can delete data from derived tables. Only the data is cleared from derived tables, the derived tables are not deleted from the database.

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Modeling** in the left pane.
   The **Modeling** option expands and displays two options: **Objects** and **Variables**.
4. Click **Objects**.
   The **Objects** page opens.
5. Click the file object for which you want to delete data from derived objects.
   Note: You cannot perform this operation if the file objects are inherited.
   The **<Object_Name>** page opens. This page includes the list of derived objects.
6. Click the Delete Data icon.
   Note: This icon becomes visible only when you create and register derived objects (tables).
7. Click **Delete** on the confirmation dialog.
   The CA TDM Portal creates a job for the delete data operation and adds it to the jobs queue. You can view the jobs queue in the requests table by clicking the job ID in the message. The requests table displays all the job requests with their status, date, name, and other relevant information. When the status of the job is shown as Completed, the CA TDM Portal completes the task of deleting the data from derived tables. You can also click the required row in the requests table to view the additional information about the job, if necessary. The additional information is displayed in the **Additional Information** dialog.

Drop Derived Objects

You can drop derived tables from the database if you do not require them. When you drop derived tables, the tables are unregistered from the repository and then they are deleted from the database.

Note: You cannot drop a single derived table.

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Modeling** in the left pane.

The **Modeling** option expands and displays two options: **Objects** and **Variables**.

4. Click **Objects**.
   The **Objects** page opens.
5. Click the file object for which you want to delete derived objects.
   Note: You cannot perform this operation if the file objects are inherited.
   The **<Object_Name>** page opens. This page includes the list of derived objects.
6. Click the Drop Derived Objects icon.
   Note: This icon becomes visible only when you create and register derived objects (tables).
7. Click **Delete** on the confirmation dialog.
   The CA TDM Portal creates a job for the drop derived objects operation and adds it to the jobs queue. The selected objects are removed from the **Objects** page only when the corresponding delete job is completed. They remain visible on the page till the job is in progress.
   You can view the jobs queue in the requests table by clicking the job ID in the message. The requests table displays all the job requests with their status, date, name, and other relevant information. When the status of the job is shown as Completed, the CA TDM Portal completes the task of removing all derived tables from the database. You can also click the required row in the requests table to view the additional information about the job, if necessary. The additional information is displayed in the **Additional Information** dialog.

Integration with CA Service Virtualization

This article explains the concepts, benefits, use cases, and best practices about integrating the CA Test Data Manager (CA TDM) Portal with CA Service Virtualization (SV integration). This article also includes an end-to-end example that helps you understand the complete SV integration workflow using the CA TDM Portal.

This article covers the following information:

Tutorial Video

Watch the "Integrate the CA TDM Portal with CA Service Virtualization" Youtube video for a visual walk-through of a use case of using RR pairs to generate data for data-driven virtual services in CA Service Virtualization.

How This Integration Benefits Users

Often, organizations face issues when unavailable, unfinished, or constrained components create situations where testers and developers wait for components to become available *upstream*. Many organizations, therefore, use service virtualization to provide parallel, on-demand access to the components that distributed teams need. However, creating realistic virtualized services requires realistic data. You cannot expose live service data to non-production environments, because it increases the risk of data breach. Similarly, manual creation of request-response pairs (sample data) for a virtual service is a time-consuming and error-prone process. This approach often leads to scenarios where the virtual service data is not exhaustive enough for the rigorous testing. Also, it becomes difficult to incorporate new specifications because of the manual effort involved, leading to the risk of data becoming obsolete.

Integrating CA Service Virtualization with the CA TDM Portal helps you address such issues. This integration helps you update running virtual services with realistic, representative virtual data, which lets you cover a wide variety of the possible testing scenarios. You push the generated Request-Response (RR) pairs into a running virtual service, augmenting your virtual service with the synthetic data that is similar to the production data. Some of the benefits that this integration provides are as follows:

- Maintains referential integrity of data by creating it (data) directly from an API specification (for example, WSDL). This approach helps create stable environments that are free from cross-system dependencies and constrains.
- Provides on-demand environments without risking non-compliance, because no live data is exposed.
- Removes the need to create or maintain manual data for virtual services.
- Provides uninterrupted access to the up-to-date environments to the distributed teams. This continuity enables them to deliver fully tested applications.
- Avoids project delays by simulating unavailable or incomplete components.

Use Cases

Using this integration, you can manage the following use cases:

Data Synchronized Across Inter-Dependent Systems and Services

This integration helps you generate and push virtual data with correct referential integrity across inter-dependent services, databases, and components. If your virtual data is referentially intact, your tests do not fail because of data inconsistency.

Consider a scenario where you are testing a customer web portal that is used for ordering items from an online store. The web portal is part of a composite system. When a test order is submitted, the transaction goes through the following steps:

- An order database is searched for the order.
- An inventory service is accessed to update the stock.
- A credit card payment processing service is accessed to charge the credit payment.

The order database is complete and is available for use. However, the inventory service and credit card payment service are constrained. They are unavailable to the teams testing the customer web portal and must be virtualized. To do so, create virtual data so that it is synchronized across the test databases and virtual services. That is, when a test order is submitted, the inventory service and credit processing service must return an inventory item and a credit card. This information must correspond to the order found in the order database. For this scenario, you must push synchronized RR pairs into the inventory service and credit card payment processing service. If the data is not synchronized, tests fail due to data inconsistency. The SV integration lets you manage this scenario as follows:

The CA TDM Portal provides the virtual data that is required to cover every test case. When a test is executed, relevant data in the order database is reserved and RR pairs are generated for the dependent services. Therefore, the correct RR pair is pushed into the virtual inventory service. Simultaneously, the correct data is reserved in the order database and the correct RR pair is pushed into the credit card processing system. The constraints that arise from cross-system data dependencies are therefore removed and teams can cover the complete testing scenario without any manual intervention.

Increase Test Coverage

From a test coverage perspective, you want sufficient coverage in your virtual service. You do not need to guess what possible combinations are available to improve the test coverage. You can push the most important permutations of RR pairs into the virtual service. By linking with CA Agile Requirements Designer, you can determine the different permutations that are available to generate data. Using that generated data of all the important permutations and through the CA TDM and SV integration, you can push the data into a virtual service and can get a better test coverage.

Up-to-Date Virtual Services

With this integration, you can easily maintain your virtual services and provide an up-to-date testing environment to test new scenarios. For example, whenever API specifications change, you can update the virtual service to support the new test scenarios. You can push new parameters into existing virtual services and can leverage the previous effort. This approach also lets you maximize the value of existing virtual data. Testers can then use the up-to-date environment that contains the latest version.

Virtual Services on Demand

With this integration, you can create realistic virtual services without spending manual efforts on generating RR pairs (sample data). You can create realistic sample data for new virtual services directly from API specifications (for example, WSDL/RR pair). You can also support more testing scenarios by pushing new RR pairs into existing virtual services. Distributed teams can then run efficient test and development cycles in parallel, in secure environment that is free from system constraints and dependencies.

Understand the CA TDM Portal and SV Projects

Integrating the CA TDM Portal with CA Service Virtualization involves interaction with different projects. Different projects that are used when you integrate the CA TDM Portal with CA Service Virtualization are as follows:

CA Test Data Manager (CA TDM) Portal Project

You create a CA TDM Portal project (Portal project) by using the CA TDM Portal. All the integration-related operations that you perform in the CA TDM Portal occur in context of a Portal project and its version.

The CA TDM Portal project must also contain at least one version. For simple applications, you can work with a single version. However, if you have a complex application, create multiple Portal project versions to address different scenarios. The version name is usually the name of the current release of your database/application; for example, 7.A. You can also create one generic version within each Portal project. The generic version stores all generic test cases, which normal test cases can then inherit. For example, in a travel system, you can create a standard trip. The standard trip is then available when you edit the data. For more information about how to create Portal projects and work with them, see Create and Edit Projects.

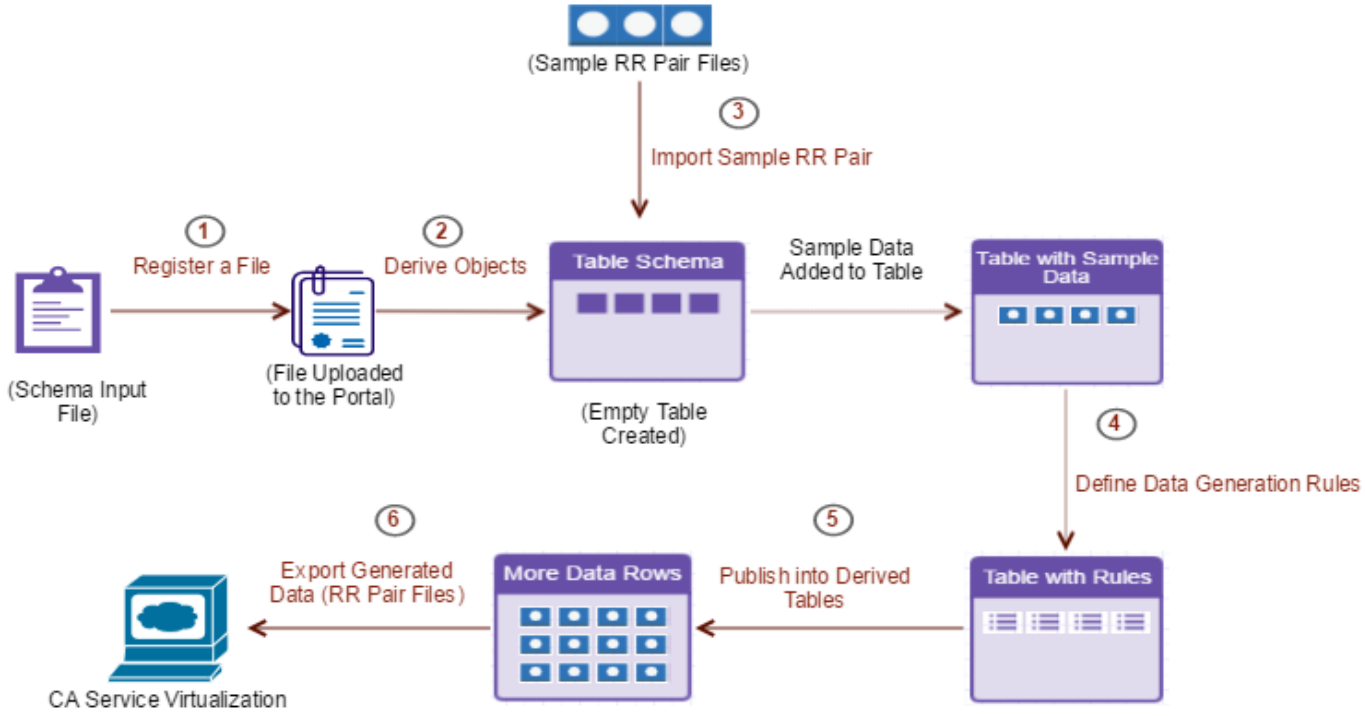
CA Service Virtualization Project

A CA Service Virtualization project (SV project) contains static virtual services, which you can run in a virtual service environment (VSE). After you run the static virtual service, you can update it by using RR pairs (XML or JSON) that the CA TDM Portal has created. For more information about SV projects, see CA Service Virtualization documentation.

How the CA TDM Portal and CA Service Virtualization Interact

The following diagram illustrates the high-level flow of information between the CA TDM Portal and CA Service Virtualization:

Figure 24: SV_Integration_Flow



The following points explain the information flow that is shown in the illustration:

- **Step 1:** Register a file object to the CA TDM Portal.
  For SV integration, the file object must be of type WSDL, XML RR pair, JSON RR pair, or REST RR pair.
- **Step 2:** Create and register derived objects in the target database based on the registered file object.
- **Step 3:** Import the sample RR pair data into the derived objects.
- **Step 4:** Define data generation rules.
  This step adds data generation rules to the derived objects, which help during the process of data generation.
- **Step 5:** Publish the data into derived objects.
  This step adds more data to the derived objects.
- **Step 6:** Export the generated data (as RR pair files) into a virtual service in CA Service Virtualization.

Best Practices

Review the following best practices that you can use while working with the SV integration:

- **Performing Multiple Operations:** (For WSDL) You can use one file object to create and register derived objects only for one operation at a time. To use the same file object for multiple operations, register that file object separately for each operation and create and register derived objects. For example, your WSDL file has two operations (`getAddress` and `getProduct`). You want to derive objects for each operation. In this case, register your WSDL file separately for each operation—first for the `getAddress` operation and then for the `getProduct` operation.
  Another use case about using the same file object for performing multiple operations is as follows:
  If the "create and register derived objects" task fails for one operation in a file object, you can reuse the same registered file object for another operation. If the task succeeds for one operation, drop the derived objects if you want to use another operation for the same file object. For example, your WSDL file has two operations (`getAddress` and `getProduct`). In this case, you register the WSDL file and perform the "create and register derived objects" task for the `getAddress` operation. If the task fails for `getAddress` , you can perform the same task for the second

operation, `getProduct` . However, if the task succeeds for `getAddress` and you want to use the same file object for `getProduct` , drop the derived objects that are created for `getAddress` and proceed with `getProduct` .

- **Using the Same Connection Profile and Schema Name:** When you perform the import and export tasks, we recommend that you do not change the connection profile and schema names that you used at the time of creating and registering derived objects. Though the CA TDM Portal lets you change the connection profile and schema names, we recommend that you do not change them.
- **Identifying Derived Objects:** If you use the same database as a target connection profile to derive objects for the different file objects, it becomes difficult to identify which derived objects belong to which file object. In this case, you can uniquely identify derived objects based on the schema name. The schema name format is `<objectName>_<objectId>` , where `<objectId>` is an integer.
- **Adding Unique Values to Primary Keys:** Every derived object includes a column that is named `Shred_ID` , which must include a unique value. The CA TDM Portal recognizes this ID as an identity of a derived object. Instead of manually adding a unique value to this primary key column, use the **Make All Parents Default** option in the CA TDM Portal to automatically add this value during data generation. For more information, see the step about relational editing in Create Data Generation Rules.
- **Establishing Parent-Child Relationships:** After you create and register derived objects, you can automate the task of ensuring that the foreign key column in the child table refers the primary key column in the parent table. To do so, use the **Make All Children References** option in the CA TDM Portal. This option helps you automatically generate data generation rules for the foreign key column in the child table, which otherwise is a manual and error-prone effort. This reference, therefore, enables you to establish a relationship between the parent and the corresponding child tables. This approach also maintains the referential integrity of the data at the time of publishing. For more information, see the step about relational editing in Create Data Generation Rules.
- **Incorporating Schema Changes:** Consider a scenario where you use one file object to create and register derived objects in context of a specific project. After you do that, you update the same file object because of the additional information that you received. Now, to accommodate the additional schema changes, you want to use the updated file to create and register derived objects. In such scenarios, you do not need to use a different project and perform the "create and register derived objects" task from the start. You can simply use a new version in the same project to create and register derived objects. This approach ensures that the CA TDM Portal identifies and implements only the changes in the schema, not the complete information. For more information, see Upgrade Inherited File Objects.
- **Using Table Relationships:** After you derive objects in the CA TDM Portal, you can view the table relationships between different derived objects. These relationships help you understand how derived objects are related to each other. You can use this information to select appropriate derived objects for which you can write necessary data generation rules. For more information about table relationships, see View Table Relationships.
- **Creating Virtual Services:** In CA Service Virtualization, you can create a virtual service using multiple methods. For example, using recording, RR pair, and JDBC. However, the CA TDM Portal is not dependent on the method that you used to create the virtual service. After you run the virtual service, the CA TDM Portal can update it with generated RR pairs regardless of the method that you used to create the service.
- **Using Correct RR Pair Naming Convention:** Ensure that the RR pair files that you use for uploading the data into a virtual service follow the correct naming convention. Example RR pair file names that follow the correct naming convention are `TDMFile_1655-Req.xml` and `TDMFile_1655-Rsp.xml` .

Differences in Handling XML, JSON, and REST RR Pairs

The SV integration requires an RR pair to update a virtual service. The SV integration supports the following types of RR pairs:

- XML request-XML response (XML RR pair)
- JSON request-JSON response (JSON RR pair)
- REST request-REST response (REST RR Pair)

XML Request-XML Response (XML RR Pair)

XML RR pair supports SOAP, non-SOAP, and REST formats. The RR pair files are in .xml format. For SOAP/non-SOAP, the XML RR pair input includes an XML request body and a corresponding XML response body. The SV integration supports this XML RR pair use case, because it adheres to the RR pair format. However, for REST format in the case of XML RR Pair, different REST methods are available—PUT, POST, GET, and DELETE. RR pair input for PUT and POST requires a request body and a corresponding response bod. Whereas, GET and DELETE do not require any request body. The SV integration supports only PUT and POST methods for the XML RR pair use case.

JSON Request-JSON Response (JSON RR Pair)

JSON RR pair supports REST format. The RR pair files are in .json format. The SV integration supports the PUT and POST methods for JSON RR pairs.

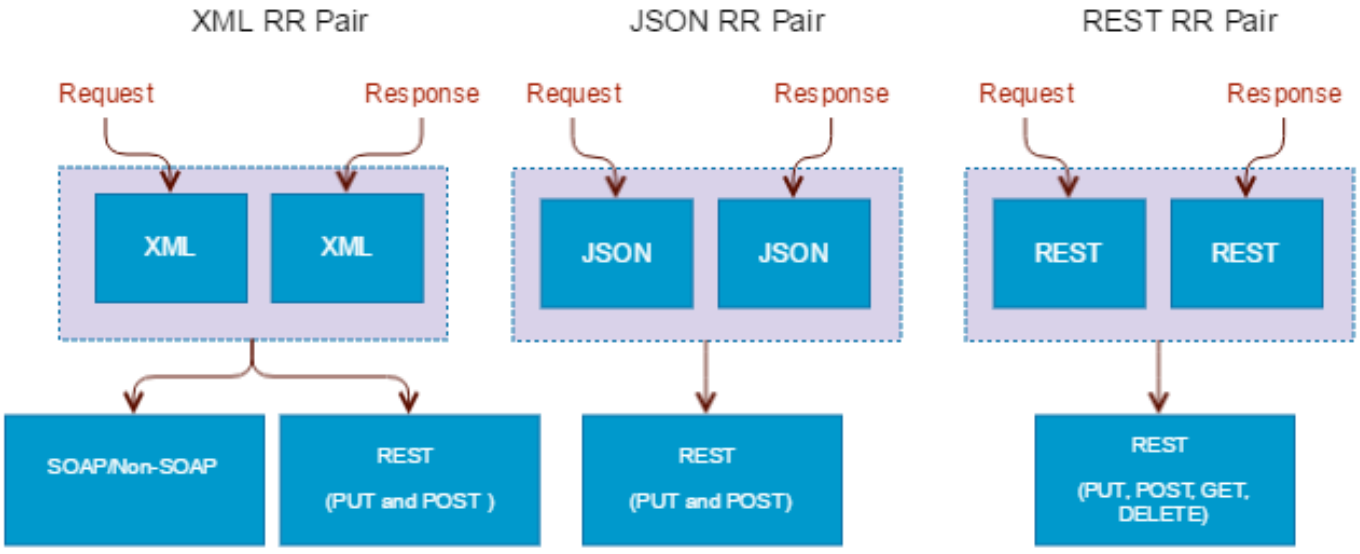
REST Request-REST Response (REST RR Pair)

REST RR pair supports REST format. The RR pair files are in .txt format. The SV integration supports the PUT, POST, GET, and DELETE methods for REST RR pairs. The text files (which include the data for REST RR pair) must include the data in a specific format. For more information about the format, see REST RR Pair Format.

> **NOTE**
> For GET and DELETE, the request text file includes only the URL; the response text file includes the URL and body.

The following illustration further explains the information:

Figure 25: REST_RRPAIR_Formats



Example: Get Supplier Information Based on a ZIP Code

Access to a live environment in your organization is restricted. You want to test your application service with various combinations of data to ensure that the service works as expected. To address such scenarios, you can use CA Service Virtualization to virtualize the actual service and push RR pairs into this virtual service. You, therefore, enhance the virtual service by adding more RR pairs to it, allowing you to rigorously test your application. This approach lets you route your application to the virtual service so that you can test the application without getting blocked because of the restricted access.

In this example, you test a scenario where you virtualize a service to get details about the supplier based on a ZIP code. You have only the ZIP code information. You want to use that ZIP code in a request body to find available suppliers catering to that area in the corresponding response body. To virtualize this service, you must have multiple XML RR pairs that you want to host in that virtual service. For these XML RR pairs, the request XML file must include the ZIP code information and the corresponding response XML file must include the related supplier information. You generate multiple similar XML files, adhering to the same schema defined in the associated WSDL file. All the XML RR pairs are then hosted in the virtual service.

The complete workflow is as follows:

1. Create a Connection Profile.
2. Create a Portal Project.
3. Register a File Object (WSDL).
4. Create and Register Derived Objects.
5. Create a Data Generator.
6. Import the Sample RR Pairs into Derived Objects.
7. Verify Records in the Database After Import.
8. Add Data Generation Rules.
9. Publish More Data into Derived Objects.
10. Verify Extra Records in the Database After Publishing.
11. Create a Virtual Service in CA Service Virtualization.
12. Configure the Virtual Service Connection in the CA TDM Portal.
13. Export the Generated Data (RR Pairs) into the Virtual Service.
14. Verify the RR Pairs Added to the Virtual Service.

> **NOTE**
> You can follow the same process for the other file object types (XML RR pair, JSON RR pair, and REST RR pair) that are applicable for SV integration. As of now, you cannot verify JSON RR pairs and REST RR pairs that are added to the virtual service using a client. However, if you have received a successful message for the export operation, it implies that the virtual service has been updated with the generated RR pairs.

Create a Connection Profile

Create a connection profile to ensure that you store the details about a connection to a target database system (Microsoft SQL Server).

Follow these steps:

1. Access the CA TDM Portal.
2. Click **Configuration** in the left pane.
   All available configurable options are expanded.
3. Click the **Connection Profiles** option.
   The **Connection Profiles** page opens.
4. Click the **New Profile** button.
   The **Add New Connection Profile** page opens.
5. Enter information in the following fields for this example:
   - **Profile Name**
     Specify the profile name as `Supplier_Connection_Profile`.
   - **Description**
     Specify the connection description as `This is a Supplier connection profile.`
   - **DBMS**
     Specify the database type as `SQL Server`.
   - **Server**

Specify the server where the Microsoft SQL Server database is available as `abc01-ip05`.
- **Database**
  Specify the name of the database as `SupplierDB`.
- **Port**
  Specify the port number as `1433`.
- **User Name**
  Specify the user name as `sa`.
- **Password**
  Specify the password as `Xyz123`.

6. Click **Test** to verify the connection.
7. Click **Save** to save the connection profile.
   You have successfully created a connection profile.

Create a Portal Project

Create a Portal project to ensure all the operations are performed in context of that project.

Follow these steps:

1. Access the CA TDM Portal.
2. Select the Project Manager icon (gear icon) in the top blue bar.
   The **Manage Projects** dialog opens.
3. Click **New Project**.
   The **New Project** dialog opens.
4. Enter information in the following fields for this example:
   - **Name**
     Specify the name of the CA TDM Portal project as `Supplier`.
   - **Description**
     Specify the description of the project as `This is a Supplier project`.
   - **Version**
     Specify the project version as `1.0`.
   - **Version Description**
     Specify the project version description as `This is version 1.0 of the project Supplier`.
   - **All new versions inherit tables from previous version**
     Enable this option to ensure that a new version can inherit tables from a previous version.
5. Click **Save** to save the information.
   You have successfully created a Portal project `Supplier` with its version as `1.0`.

Register a File Object (WSDL)

For this example, the file object that you register is WSDL (`content/dam/broadcom/techdocs/us/en/assets/docops/tdm/medicare.wsdl`).

Follow these steps:

1. Access the CA TDM Portal.
2. Select the `Supplier` project and its version `1.0` from the **Project** drop-down list in the top blue bar.
3. Click **Modeling** in the left pane.
   The **Modeling** option expands and displays two options—**Objects** and **Variables**.
4. Click **Objects**.
   The **Objects** page opens.
5. Click **Register New Object(s)**.
6. Select WSDL as the object type from the **Object Type** drop-down list.

7. Enter the object name in the **Name** field. For this example, the value is specified as `Medicare_Supplier`.
8. Drag and drop the `medicare.wsdl` file to the specified area under **File(s) to Upload**.
   The `medicare.wsdl` file is uploaded to the CA TDM Portal.
9. Click **Register** to register the WSDL file object.
   The `Medicare_Supplier` object is added to the list of registered objects.

The next step is to create and register derived objects based on the registered WSDL file object.

Create and Register Derived Objects

When you create and register derived objects, you add a schema to the relational database based on the registered WSDL file.

Follow these steps:

1. Access the CA TDM Portal.
2. Select the `Supplier` project and its version `1.0` from the **Project** drop-down list in the top blue bar.
3. Click **Modeling** in the left pane.
   The **Modeling** option expands and displays two options: **Objects** and **Variables**.
4. Click **Objects**.
   The **Objects** page opens.
5. Click the `Medicare_Supplier` object.
6. Click the WSDL operation `GetSupplierByZipCode` in the table that lists the available WSDL operations for the registered `medicare.wsdl` file.
7. Select the connection profile as `Supplier_Connection_Profile` from the **Connection Profile** drop-down list.
8. Click the **Create and Register** button.
   A message displaying the job ID appears. When the job completes, all the derived objects are displayed in the **Derived Tables** section.
9. Review the derived objects.
   You have successfully created and registered derived objects based on the registered WSDL file.

Create a Data Generator

Create a data generator for your `Supplier` project.

Follow these steps:

1. Access the CA TDM Portal.
2. Select the `Supplier` project and its version `1.0` from the **Project** drop-down list in the top blue bar.
3. Click **Generators** in the left pane.
4. Click the **New Generator** button.
5. Enter information in the following fields for this example:
   - **Name**
     Specify the data generator name as `Supplier_Generator`.
   - **Description**
     Specify the data generator description as `This data generator is for the Supplier project.`
6. Click **Save** to save the information.
   The data generator `Supplier_Generator` is created and added to the data generators list.

Import the Sample RR Pairs into Derived Objects

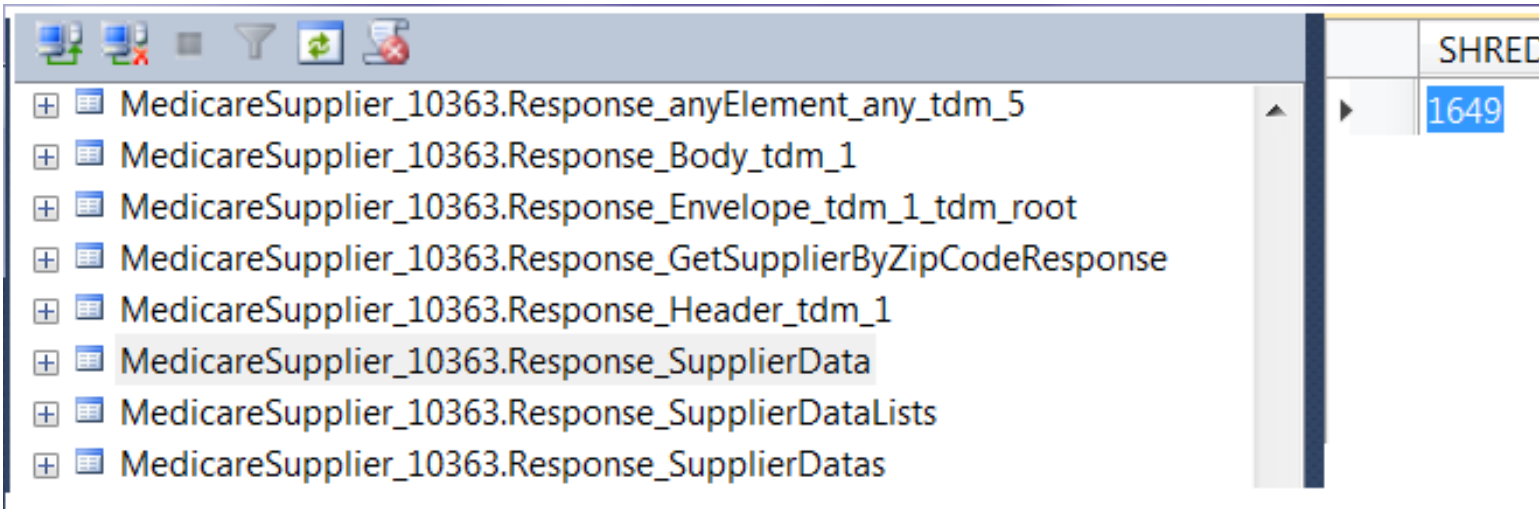
Import the sample data (XML RR pairs) into the `Supplier_Generator` data generator and into the target database schema (`MedicareSupplier_10363`).

Follow these steps:

1. Access the CA TDM Portal.
2. Select the `Supplier` project and its version `1.0` from the **Project** drop-down list in the top blue bar.
3. Click **Modeling** in the left pane.
   The **Modeling** option expands and displays two options: **Objects** and **Variables**.
4. Click **Objects**.
   The **Objects** page opens.
5. Click the `Medicare_Supplier` object.
6. Click the Import Data icon (up arrow).
   The **Import Data** dialog opens.
7. Enter information in the following fields for this example:
   - **R/R Pair Link ID**
     Specify the RR pair link ID as `r1` .
   - **Advanced Settings**
     Specify the connection profile as `Supplier_Connection_Profile` and the schema name as `MedicareSupplier_10363` .
   - **Request-File to Upload**
     Drag and drop the XML request filecontent/dam/broadcom/techdocs/us/en/assets/docops/tdm/medicare_req.xml for the `medicare.wsdl` file (which defines the schema).
   - **Response-File to Upload**
     Drag and drop the XML response file `content/dam/broadcom/techdocs/us/en/assets/docops/tdm/medicare_rsp.xml` for the `medicare.wsdl` file (which defines the schema).
   - **Import To Generator**
     Select this option to import the sample data to a data generator. When you select this option, the following drop-down list appears:
     - **Data Generator Connection**
       Specify the data generator as `Supplier_Generator` .
8. Click the **Import** button.
   A message displaying the job ID appears. When the job completes, the sample data is imported into derived objects in the target database schema and into the data generator.

Verify Records in the Database After Import

Access the Microsoft SQL Server database (`SupplierDB`) to verify that the sample data is added to the derived objects in the database. The following screen shot shows the sample RR pair data added to one of the derived objects after the import process:



Review that the derived object (`MedicareSupplier_10363.Response_SupplierData`) includes only one record, with the ZIP code as `60532` .

Add Data Generation Rules

You add data generation rules to derived objects. These rules help you add more data to your derived objects during the data publishing. For more information about how to add data generation rules, see Create Data Generation Rules.

Follow these steps:

1. Access the CA TDM Portal.
2. Select the `Supplier` project and its version `1.0` from the **Project** drop-down list in the top blue bar.
3. Click **Generators** in the left pane.
4. Click the `Supplier_Generator` data generator that you created for the `Supplier` project.
5. Click the **Select Tables** button to open the **Registered Tables** dialog.
   This dialog lists the registered tables (*used* and *unused*) based on the Supplier project. The Supplier project is associated with the Supplier_Generator data generator.
   a. View the table relationships to understand how tables are related to each other. For more information, see View Table Relationships.
6. Click the **Relational Edit** button.
7. Select the **Make All Children References** and **Make All Parents default** options, and click **OK**.
   The first option automatically adds a data generation expression to table cells. This option ensures that the foreign key column in the child table refers the primary key column in the parent table. Similarly, the second option automatically adds a data generation expression to table cells. This option ensures that all the primary keys in the parent tables are replaced with the valid next sequence. For more information, see the step about relational edit in Create Data Generation Rules.
8. Review the information in the **Relational Editor Results** dialog, and click **OK**.
9. Add data generation rules to the appropriate cells of all the required derived objects.
   Example data generation rules added to the `Response_SupplierData` derived object are as follows:

- SHRED_ID: `@nextval(SHRED_ID_SEQ)@`
- SupplierNumber: `@randlov(0,@seedlist(Car Parts)@)@`
- CompanyName: `@randlov(0,@seedlist(Companies)@)@ LTD`
- Address1: `@randlov(0,@seedlist(US Address Line 1)@,1)@`
- City: `@randlov(0,@seedlist(US City State Zip County)@,1)@`
- Zip: `^Request_GetSupplierByZipCode.zip(1)^`
- ZipPlus4: `@randlov(0,@seedlist(US City State Zip County)@,2)@`
- Telephone: `@randrange(2300,9800)@`
- Description: `(@randrange(110,999)@)@randrange(110,999)@-@randrange(1110,1999)@`
- Response_SupplierDatas_SHRED_ID: `^Response_SupplierDatas.SHRED_ID(1)^`

You have successfully added data generation rules to all the derived objects.

Publish More Data into Derived Objects

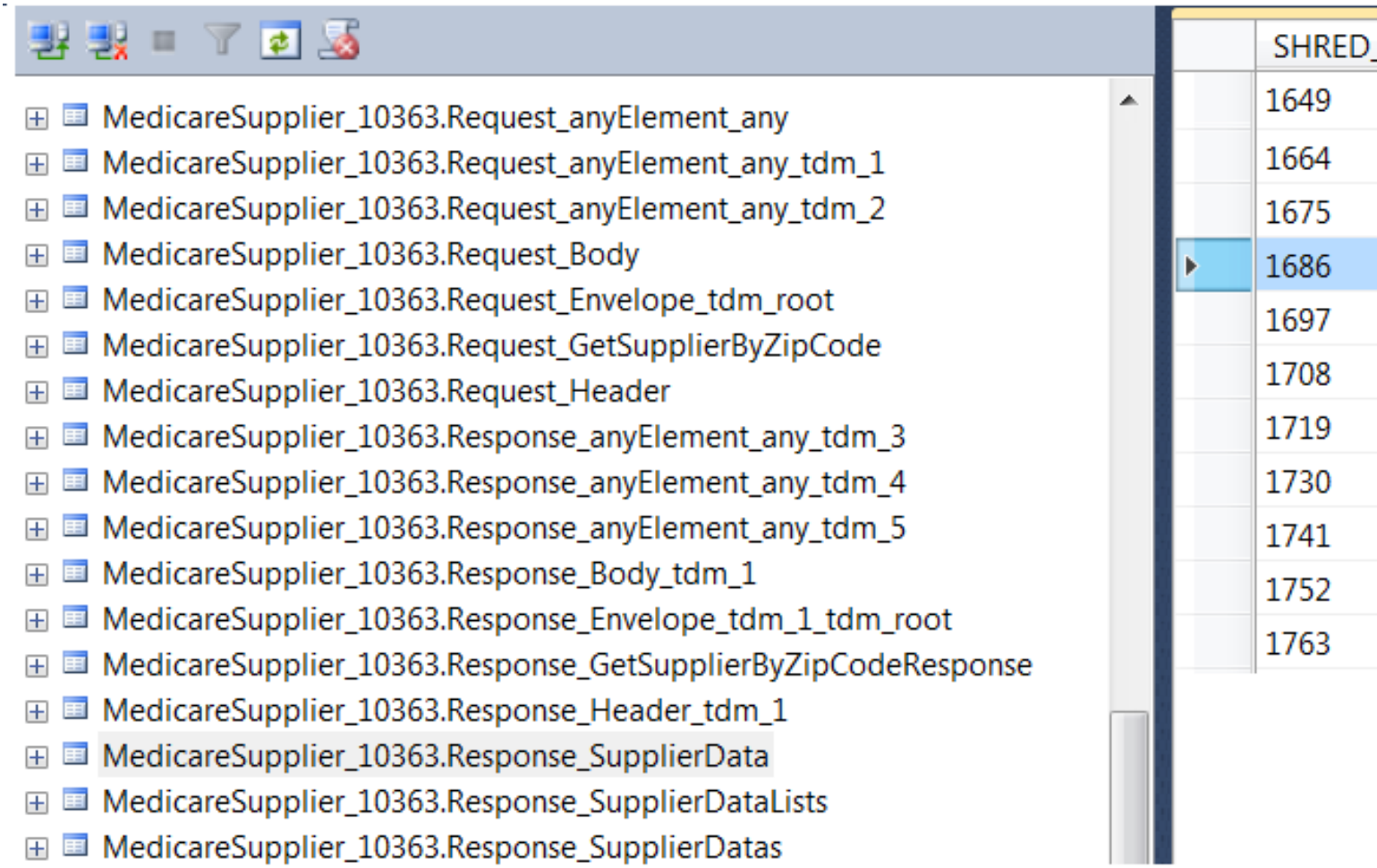
After you create data generation rules, publish the data so that you can add more data to the derived objects in the target database. For more information about how to publish the data, see Publish Data Using the CA TDM Portal.

Follow these steps:

1. Access the CA TDM Portal.
2. Select the `Supplier` project and its version `1.0` from the **Project** drop-down list in the top blue bar.
3. Click **Generators** in the left pane.
4. Click the data generator `Supplier_Generator` that you created.
5. Click the **Select Tables** button to open the **Registered Tables** dialog.
6. Click the rows corresponding to the *used* derived objects that you want to use for the data publishing. For example, *used* derived objects are `Request_GetSupplierByZipCode` , `Response_GetSupplierByZipCodeResponse` , `Response_SupplierDataLists` , `Response_SupplierData` , and so on.
   All derived objects with appropriate rows are listed in the **<Data Generator Name>** page.
7. Review the data generation rules (added to the derived objects in the preceding procedure).
8. Click the **Publish** button.
9. Enter information in the following fields for this example:
   a. **Table Count**
      Specify the table count value as `1` for all the listed derived objects.
   b. **Publish To**
      Specify the connection profile as `Supplier_Connection_Profile` .
   c. **Schema**
      Specify the target schema name as `MedicareSupplier_10363` (for the selected connection profile).
   d. **Repeat**
      Specify the value as `10` to provide the number of rows to add to the derived objects.
   e. **Email**
      Specify the email ID where you want to send the publishing notifications as
      `givore@xyz.com`
      .
   f. **Schedule Publish**
      Select **Now** in this section.
   g. **Include in Publish**
      Select the check box that is available before the table name to include the table in the publish.
10. Click the **Publish** button.
    A message displaying the publish job ID appears. When the job completes, the data is published into the derived objects in the target database schema.

Verify Extra Records in the Database After Publishing

Access the Microsoft SQL Server database (`SupplierDB`) to verify that extra rows are added to the derived objects in the target database.
The following screen shot shows that the same derived object (`MedicareSupplier_10363.Response_SupplierData`) now includes more records as a result of the publish process:



Create a Virtual Service in CA Service Virtualization

Use the CA DevTest Portal to create a virtual service. You use the initial sample RR pair data to create the virtual service. You then add more RR pairs (as part of exporting from the CA TDM Portal) to this virtual service to represent the real-world scenario.

Follow these steps:

1. Log in to the CA DevTest Portal.
2. Click the **Virtualize using RR pairs** option under **Create** in the left pane.
3. Select the virtual service environment server name from the **VSE Server** drop-down list.
4. Enter the virtual service name (`Supplier_Virtual_Image`) in the **Service Name** field.
5. Select the service group name (`VSE`) from the **Service Group** drop-down list.
6. Select `HTTP` as a protocol from the **Protocols** drop-down list.

7. In the **Upload Custom Request/Response Files** area, upload the sample XML request and response files by browsing to the location where the files are available or by dragging and dropping the files into the area.
8. Click the **Allow duplicate specific transactions** option.
9. Click **Create and Deploy**.
   A message displays stating that the virtual service with the name `Supplier_Virtual_Image` is created and is deployed in the provided `VSE` server. The message also provides the HTTP location where the virtual service is available.

Configure the Virtual Service Connection in the CA TDM Portal

To communicate with the required CA Service Virtualization environment, configure the CA TDM Portal with the appropriate virtual service-related information.

Follow these steps:

1. Access the CA TDM Portal.
2. Click **Configuration** in the left pane.
3. Click the **DevTest Portal** option.
4. Enter information in the following fields for this example:
   - **Protocol**
     Specify the protocol value to connect to the virtual service as `HTTP` .
   - **Host Name**
     Specify the name of the host where the virtual service environment is available as `abc01-grica` .
   - **Port**
     Specify the registry web service port number (not CA DevTest Portal port number) where the virtual service is running as `1505` .
   - **Username**
     Specify the name of the user having access to the virtual service environment as `abc` .
   - **Password**
     Specify the password for the user as `xyz123` .
5. Save the configuration details.
   The virtual service connection details are saved.

Export the Generated Data (RR Pairs) into the Virtual Service

After you configure the CA TDM Portal to connect to the virtual service, you can export the generated data into the configured virtual service. The data is exported into the virtual service in the form of XML RR pairs.

Follow these steps:

1. Access the CA TDM Portal.
2. Select the `Supplier` project and its version `1.0` from the **Project** drop-down list in the top blue bar.
3. Click **Modeling** in the left pane.
   The **Modeling** option expands and displays two options: **Objects** and **Variables**.
4. Click **Objects**.
   The **Objects** page opens.
5. Click the `Medicare_Supplier` object that you want to use for the data export operation.
6. Click the Export Data icon (down arrow).
7. Enter information in the following fields for this example:
   - **Connection Profile**
     Specify the connection profile name as `Supplier_Connection_Profile` .
   - **Schema Name**

Specify the schema name for the selected connection profile as `MedicareSupplier_10363` .

– **Update Virtual Service**
Select this option and provide information in the following fields:
  - **Virtual Service Environment**
Specify the name of the virtual service environment as `VSE` . This drop-down list is automatically populated based on the virtual service configuration information that you provide in the CA TDM Portal.
  - **Virtual Service**
Specify the name of the virtual service as `Supplier_Virtual_Image` . This drop-down list is automatically populated based on the virtual service environment that you select in the preceding list.

8. Click the **Export** button.
An export job is created. A message with a job ID appears. When the job completes, all the exported XML RR Pair files are exported into the specified virtual service `Supplier_Virtual_Image` .

Verify the RR Pairs Added to the Virtual Service

After you perform the export operation, all the data available in the target schema is pushed into the virtual service in the form of XML RR pairs. You can verify this information by using any available client that can display a response based on a request for a virtual service.

In this example, CA DevTest Workstation is used to verify the RR pair data in the virtual service.

Follow these steps:

1. Log in to the CA DevTest Workstation.
2. Right-click **Tests** and select **Create New Test Case**.
3. Right-click in the right pane and select **Add Step, Web/Web Services, Web Service Execution (XML)**.
4. Specify the location of the WSDL file (used for deriving objects) and the virtual service endpoint.
5. Paste the request body and provide the ZIP code (for example, `65001`).

> **NOTE**
>
> The ZIP code `65001` was added to the derived objects as a result of the data generation process. This ZIP code was not originally present in the derived objects.

6. Click the arrow icon to execute the request.
The appropriate response body is returned.

The following screen shot shows the request for the ZIP code `65001` :

The following screen shot shows the corresponding response received for the ZIP code `65001` :

In this example, you have successfully exported XML RR pairs that adhere to the schema that is defined in the WSDL file.

G-T Excel File Type

The G-T Excel file is a CA-proprietary format for Test Data Manager. With G-T Excel, users can manage and generate fixed-width flat files in the Test Data Manager Portal. For example, you can convert your Cobol Copybook file into G-T Excel format by using Generate Synthetic Mainframe File Data using File Definitions.

To prepare test data for the G-T Excel file object type, follow these steps:

Register a G-T Excel File Type

In the CA TDM Portal, you register file objects so that you can perform various data manipulation operations (for example, data generation) on them. You register file objects in context of a project and its version. This procedure is applicable only for the G-T Excel file object type. For other file object types (for example, WSDL, XML, CSV, and JSON), see the appropriate section.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
3. Expand **Modeling** in the left pane and click **Objects**.
   The **Objects** page opens. This page lists all the objects that are registered to the selected version and the project. If no object is registered, nothing is listed on the page.
4. Click the **Register New Object(s)** button.
   The **Register New Object(s)** page opens.
5. Select GTExcel as the file object type from the **Object Type** drop-down list.
6. Enter an appropriate name for the G-T Excel file object that you want to register in the **Name** field.

7. Specify the location from where you want to get the G-T Excel file object (<*GTExcelFilename*>.xls) in the **File(s) to Upload** section. You can browse to the location or drag and drop the file.
   Note: Ensure that the extension of the G-T Excel file is .xls (Excel 97-2003 workbook).
8. Click the **Register** button.
   The **Objects** page opens with the the newly registered object added to the list of registered objects.
   You have successfully registered the G-T Excel file object.
   Note: To delete a registered object, click the cross icon (X) for the required object and confirm the deletion. A delete job is created and is added to the jobs queue. You can view the jobs queue in the requests table by clicking the request ID in the message that is displayed. When the status of the job is shown as **Completed**, the registered object is deleted. You can also click the appropriate row to view the additional information about the job. The additional information is displayed in the **Additional Information** dialog.

Additionally, if you want to create a data generator in context of the selected project and version, click the **Create Generator** button in the **Objects** page and follow the steps to create a generator.

Import Sample Data into a Data Generator

Import the sample data into a data generator that is associated to the same project and version that you used for registering the G-T Excel file. To import the sample data, ensure that you use a file that contains the data in the format that adheres to the G-T Excel format.

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Modeling** in the left pane.
   The **Modeling** option expands and displays two options: **Objects** and **Variables**.
4. Click **Objects**.
   The **Objects** page opens.
5. Click the G-T Excel file object for which you want to import the sample data.
   Note: You cannot import the data into inherited file objects.
   The **<Object_Name>** page opens. This page includes the list of tables that are created in the repository by using the G-T Excel file.
6. Click the Import Data icon (up arrow).
   The **Import Data to Generator** dialog opens.
7. Specify the location of the file (for example, SampleData.txt, SampleData.csv, SampleData.dat) that contains the sample data in the **File(s) to Upload** area. The file must contain the sample data in the format that adheres to the G-T Excel fixed-width file format. You can drag and drop the file or browse to the file location.
8. Select the required data generator from the **Data Generator Connection** drop-down list. This drop-down list is populated with all the data generator connections based on the selected project and version.
9. Click the **Import** button.
   The CA TDM Portal creates a job for the import operation and adds it to the jobs queue. You can view the jobs queue in the requests table by clicking the job ID in the message. The requests table displays all the job requests with their status, date, name, ID, and other relevant information. When the status of the job is shown as Completed, the CA TDM Portal completes the task of importing the sample data into the selected data generator. You can also click the required row in the requests table to view the additional information about the job, if necessary. The additional information is displayed in the **Additional Information** dialog.

You have successfully imported the sample data into the selected data generator. You can now write data generation rules.

Define Data Generation Rules

After you import the sample data into a data generator, create data generation rules. These rules help you generate synthetic data that you can publish into the required file format (FD). For more information about how to create data generation rules, see the Create Data Generation Rules section.

Publish the Data into FD Files

You can publish the data into the required file format; in this case, FD (file definition with fixed width). The FD files are in ASCII format. After you publish the data, you can navigate to the requests page and download the .zip file that contains the generated data. The .zip file includes a data file and a log file.

> **NOTE**
>
> Publishing to the FD file works only for the tables that are created by the registration of the G-T Excel or CSV file object. Therefore, ensure that you select only these tables while publishing the data to the FD file. If you try to publish other tables, the publishing fails.

1. Follow Step 1 through Step 11 as described in Publish Data Using the CA TDM Portal.
2. Select the **File** option under the **Publish To** area.
3. Select **FD - Formatted Text Files** as the file type from the **Select File Type** drop-down list.
4. Enter information in other fields as described in Publish Data Using the CA TDM Portal.
5. Click the **Publish** button.
   A message with a publishing job ID is displayed. Display of this message implies that a publishing job is created and is added to the jobs queue in the requests table.
6. Click the job ID in the message.
   The **Submitted Requests** page opens. This page includes all the submitted jobs.
7. Review the status of your job.
   When the job status changes to **Completed**, it implies that the publishing process is done.
8. Click the Download icon to download the .zip file that includes the published data.

You have successfully published the data and downloaded the file that includes the published data.

CSV File Type

In the CA TDM Portal, you register objects so that you can perform various data manipulation operations (for example, data generation) on them. You register objects in context of a project and its version. This procedure is applicable only for the CSV file object type. For other file object types (for example, WSDL, XML, JSON, and G-T Excel), see the appropriate section.

Follow these steps:

1. Access the CA TDM Portal.
2. Select a project and version from the Project drop-down list available in the portal header. Optionally, click the gear icon (next to Project drop-down in the portal header) to manage for a specific project. If you want to create a new project, see Create and Edit Projects.
3. Expand **Modeling** in the left pane and click **Objects**.
   The **Registered Objects** page opens. This page lists all the objects that are registered to the selected version and the project. If no object is registered, nothing is listed on the page.
4. Click the **Register New Object(s)** button.
5. The **Register New Object(s)** page opens.
6. Enter information in the following fields:
   - **Object Type**

Lets you select the type of the object that you want to register. Select DELIM(CSV) from the drop-down list. Currently ANSI and UTF-8 are the supported encoding types of CSV files.

- **Object Location**
  Lets you upload the CSV or ZIP file that you want to register. Drag and drop the file from your local computer. You can upload multiple CSV files but only one ZIP file.
- **Tables**
  Lists the tables that will be created based on the CSV files uploaded. One table will be created for each file you upload.
  - **Import Data**
    Lets you import the data from the CSV file into the data table created for that CSV file. Select the Import Data check box against the tables for which you want to import the data and register to a data generator.
- **Data Generator**
  Specifies the Data Generator (Data Pool) to which you want to import the data from the CSV file. Select a Data Generator that associates to the corrsponding project and version.
  Note: If the data generator is not available, Create Data Generator.
- **Advanced Settings**
  Lets you specify the row number to place the header and the data.
  - **Column names at line**
    Specifies which row in the CSV file should be used as header for each table. This row is used as the header row when the table is created for the CSV file.
    Default: 1
  - **Data Starts At**
    Specifies which row in the CSV file should be used as the first row of the data. All the rows following the first row are used as data rows.
    Default: 2

7. Click **Save.**
   You have successfully registered the CSV file.
   You can now publish the data. For more information about how to publish CSV files, see Publish Data Using the CA TDM Portal.

Note: CSV files that are registered in Datamaker are not supported in the CA TDM Portal.

Defining Test Data Using Datamaker

This section describes how to register and work with data using Datamaker to prepare the data for further test data provisioning operations.

You can use Datamaker to define test data for the following data types:

- Relational tables
- Mainframe files converted by the How to Parse IMS Database Copybooks and Mask Data
- EDI files converted by the GT EDI utility

In addition to basic registration, Datamaker offers several tools for manipulating the registered data:

- A full SQL editor for running queries and making basic edits
- Tools for creating and manipulating table relationships
- The GTDiagrammer utility, which provides a visual editor view

Create a Project

To get started with CA TDM, create a project. A project in CA TDM contains: Version, Data Group, Data Set, Data Pool (or Test Case). The number of levels and the name of each level are configurable. Data is stored at the bottom level (Data Pool) and published at the Data Pool and Data Set level.

A project must also contain at least one version. For simple applications, you can work with a single version. However, if you have a complex application, create multiple project versions to address different scenarios. The version name is usually the name of the current release of your database/application; for example, 7.A. If you are not sure about the exact version, use Version 1. You can always edit the name later when you are sure of the exact version.

You can also create one generic version within each project. The generic version stores all generic test cases, which normal test cases can then inherit. For example, in a travel system, you can create a standard trip. The standard trip is then available when you edit the data.

Follow these steps:

1. Open the Datamaker UI.
2. Select **Projects**, **Project Manager** from the main menu.
3. Right-click the **Project** option in the left pane and select **New Project and Version** from the context menu.
4. Enter information in the following fields and click the save icon:
   - Project Name
   - Project Description
   - Version Name
   - Version Description
5. Verify the level structure and click **OK**.
   A confirmation dialog opens.
6. Click **OK** on the confirmation dialog.
   The project with appropriate version is created. You can add more versions later.
7. (Optional) Click the Advanced options icon (forward arrows) to provide appropriate values in the **Local Settings** and **Project Settings** areas.
   Note: In the **Project Settings** area, the highest time stamp precision value that you can provide in the **Timestamp precision** field is 3. That is, you can provide time stamp precision only up to the third digit.
8. (Optional) To create a generic version for the same project, right-click the created project and select **New Version** from the context menu.
9. Enter the version name, description, and select the **Generic** option.
   The generic version is added to the created project.

Create a data group, data set, and data pool

After you create a project, create a data group, data set, and data pool under the project version.

Follow these steps:

1. Right-click the project version and select **New Data Group** from the context menu.
2. Enter the information for the data group, click the save icon, click **OK** on the confirmation dialog, and click **Yes** to create a data set.
3. Enter the information for the data set, click the save icon, click **OK** on the confirmation dialog, and click **Yes** to create a data pool.
4. Enter the information for the data pool, click the save icon, and click **OK** on the confirmation dialog.
   You have successfully created a data group, data set, and data pool under the available project version.

After your project is configured, you can start to create and publish data at the lowest level. In the **Project Manager** dialog, you can right-click an object in the tree structure to use the additional functionality available; for example, for Data Pool and Data Object. You can also configure the number and names of levels within a version. The following

example has these levels: Project Version (Build 1), Data Group (Base Tables), Data Set (Car Hire), and Data Pool (Availability and Offices). The data is stored in and published from Data Pool:

```
-Training - Travel System

    Variables

   -Build 1

       Variables

      -Base Tables

         Variables

         Car Hire

            Variables

            Availability

              Variables

            Offices

              Variables

...

...
```

This segment represents the tree structure in the **Maintain Projects** dialog in the DataMaker UI of CA TDM.

Note: For more information about working with data objects, see the data-specific sections.

Object Registration

Before you use CA TDM to manage data, use the CA TDM Datamaker UI to register the data definitions to CA TDM. This registration is only required once per version. Re-registration of a table or data definition is only required if the definition changes. We recommend that you create a new version and register the new or changed tables to the new version. If you are not sure, register all tables to the new version.

The registration data entities include the following data types:

- Database Tables
- Flat File Definitions
- XML Files
- Excel Files
- REST Services / SOAP UI Files

WARNING

CA Test Data Manager previously used Portus (Message Gateway Server) from Ostia for generating XML, JSON, and RR pairs. In the earlier releases of CA TDM, this functionality was replaced with the CA TDM Shredder utility. However, with the deprecation of the CA TDM Shredder utility in this release of CA TDM, this functionality is now available in the CA TDM Portal. Moving forward, use the CA TDM Portal for all XML, XSD, JSON, WSDL, and RR pair file registration and data generation needs.

Register Database Tables (and Cubes)

CA TDM tries to extract as much metadata as possible for your database. For example, for all the available tables, CA TDM tries to extract attached columns, indexes, and keys. This extract is like a snapshot of your database schema. This information helps CA TDM understand what to generate.

Use the Datamaker UI to register your database tables and cubes to your project.

1. Open the Datamaker UI.
2. Select the appropriate project context in which you want to register the tables from the **Context** drop-down list.
3. Select **Projects, Register** from the main menu.
   A dialog that lists the possible data object types that you can register opens.
4. Select the **Database Table** option and click the forward arrow button next to **Select Type**.
   Note: For cubes, select the **Database Cube** option instead of the **Database Table** option.
   The **Register Tables from Data Target into Project <project_name>** opens. The left pane includes details about each available schema. The middle pane contains tables under each schema. The right pane lists tags for selecting data, for example, **Registered in current Version** selects all objects that are registered in the current version.
5. Use the drop-down list in the top-right of the dialog to register tables or check constraints from the data target or data source. You can select from the following options:
   - **Register Tables from Data Target**
   - **Register Check Constraints from Data Target**
   - **Register Tables from Data Source**
   - **Register Check Constraints from Data Source**
6. Select the required table (cube) and click the forward arrow button to register the table (cube).

Register Flat Files

Use the Datamaker UI to register the following types of files:

After external file definitions are registered as tables and rules created, import data into these tables from external files.

Register File Definition from G-T Excel File

1. Open the Datamaker UI.
2. Select the appropriate project context in which you want to register the tables from the **Context** drop-down list.
3. Select **Projects, Register** from the main menu.
   A dialog listing all the possible types of data objects that you can register opens.
4. Select the **File Definition from G-T Excel File** option and click the forward arrow button next to **Select Type**.
   The **Register File** tab opens next to **Select Type**.
5. Click the first ellipsis icon in the **Register File** section and locate the file that you want to use.
   Note: If you want to browse and register an entire directory, click the second ellipsis icon.
6. Click the **Show Record Types** button to display the records that are contained in the file.
   Each selected record is shown in the **Record Types** area.

TIP
Click the **FD** icon to view detailed information about the file.

7. Click the select all icon to select all records in the **Record Types** area.
8. Review all the information and click **Register** to register the file.

Register Delimited File from G-T Text File

For *delimited file from G-T text file*, in addition to the normal CSV file used to define your masking mapping, create an additional file that contains the layout of your file to be masked. The normal suffix for this file is DM.TXT.

1. Open the Datamaker UI.
2. Select the appropriate project context in which you want to register the tables from the **Context** drop-down list.
3. Select **Projects, Register** from the main menu.
   A dialog listing all the possible types of data objects that you can register opens.
4. Select the **Delimited File from G-T Text File** option and click the forward arrow button next to **Select Type**.
   The **Register Delimited File** tab opens next to **Select Type**.
5. Click the first ellipsis icon in the **Register File** section and locate the file that you want to use.
   Note: If you want to browse and register an entire directory, click the second ellipsis icon.
6. Click the **Show Record Types** button to display the records that are contained in the file.
7. Each selected record is shown in the **Record Types** area.

   TIP
   Click the **FD** icon to view the detailed information about the file.

   The additional lines contain the name of the column. The name must match with the COLUMN name in the map.csv file (mapping file). If the column is a date field, include the date format in double quotes; for example, "YYYY/MM/DD."
8. Click the select all icon to select all records in the **Record Types** area.
9. Review all the information and click **Register** to register the file. You can then assign normal masking functions in the same way as for RDBMS maps.
   Note: WHERE and SQLFUNCTION functions do not apply to flat file masking; cross-referencing still applies.

Example DM.TXT File

The following code shows an example of a DM.TXT file:

```
BANK_TRANSFER.DM.TXT

HEADER=Y,TRAILER=Y,DELIM=*,DATEQUOTED=,CHARQUOTED=
BUSINESS_UNIT
TRANS_REF_NUM
POSTED_DATE,"YYYY/MM/DD"
JOURNAL_DATE, "YYYY/MM/DD"
ACCOUNTING_PERIOD
FISCAL_YEAR
ACCOUNT
DEPT_ID
PRODUCT
PROJECT_ID
CHARFIELD1
AFFILIATE
FOREIGN_AMOUNT
```

```
FOREIGN_CURRENCY
MONETORY_AMOUNT
CURRENCY_CD
```

The first row of the file gives general details about the file. The following list includes information about the parameter names, description, and values:

- **HEADER**
  Specifies the header record.
  Values:
  – Y, N, or the number of rows
  – Y == 1
- **TRAILER**
  Specifies the trailer record.
  Values:
  – Y, N, or a number
  – Y == 1
- **SIZES**
  Specifies whether field sizes are provided.
  Values:
  – Y or N
- **DELIM, DELIMITER**
  Specifies whether fields are delimited. If yes, by what.
  Values:
  – 'fixed', 'FIXED' – fixed width – no delimiters
  – 'comma', '<COMMA>' – comma delimited
  – 'tab', '<TAB>' – tab delimited
  – 'us', '<US>' – ASCII char (31) delimited
  – Or any character
- **'DATEQUOTED', 'QUOTED_DATES'**
  Specifies whether dates are quoted.
  Values:
  – Y or N
- **'CHARQUOTED', 'QUOTED_STRINGS'**
  Specifies whether strings are quoted.
  Values:
  – Y or N
- **'SUBDELIM', 'SUB_DELIMITER'**
  Are subfields delimited and by what.
  Values:
  – Any character
- **'LINETERM', 'LINE_TERMINATOR'**
  Specifies how lines are terminated.
  Values:
  – 'cr/lf', '<CR/LF>', '<CR><LF>', '<CR>/<LF>' – carriage return and line feed
  – 'cr', '<CR>' – carriage return
  – 'lf', '<LF>' – linefeed
  – 'rs', '<RS>' – ASCII char (30)
  – Or any character
- **'LINESEP', 'LINE_SEPARATOR', 'RECSEP', 'RECORD_SEPARATOR'**

Specifies how records are separated.
Values:
- 'cr/lf', '<CR/LF>', '<CR><LF>', '<CR>/<LF>' – carriage return and line feed
- 'cr', '<CR>' – carriage return
- 'lf', '<LF>' – linefeed
- 'rs', '<RS>' – ASCII char (30)
- Or any character

- **'FILETERM', 'FILE_TERMINATOR'**
  Specifies how files are terminated.
  Values:
  - Any character
- **'TERMTRAIL', 'TERMINATE_TRAILER'**
  Specifies whether the trailer record is terminated.
  Values:
  - Y or N
- **'TRAILDELIM', 'TRAILING_DELIMITERS'**
  Specifies whether trailing delimiters generating output where there is no data.
  Values:
  - Y or N
- **'QUOTESTYLE', 'QUOTE_STYLE'**
  Specifies when to use quotes.
  Values:
  - ALL – everywhere
  - AUTO – when required
  - NONE – not quoted
- **'QUOTEHEAD', 'QUOTE_HEADER'**
  Specifies whether the header is quoted.
  Values:
  - Y or N
- **'QUOTETRAIL', 'QUOTE_TRAILER'**
  Specifies whether the trailer is quoted.
  Values:
  - Y or N
- **'NULLIND', 'NULL_INDICATOR'**
  Writes NULL indicators.
  Values:
  - Y or N
- **'LEFTQUOTE', 'LEFT_QUOTE'**
  Specifies the left quote character.
  Values:
  - Usually " or [but any character allowed
- **'RIGHTQUOTE', 'RIGHT_QUOTE'**
  Specifies the right quote character.
  Values:
  - Usually " or] but any character allowed
- **'STYLE', 'COMPLEXITY'**
  Specifies the complexity.
  Values:

  - SIMPLE – header body and trailer only
  - COMPLEX – multiple record types
- **'OUTMASK', 'FILEMASK', 'OUTPUT_FILE_MASK', 'OUTPUT_MASK'**
  Specifies the mask used to create the output file name.
  Values:
  - For example, Myname~STIME~
- **'OUTEXT', 'FILEEXT', 'OUTPUT_FILE_EXT', 'OUTPUT_EXT'**
  Specifies the output file extension.
  Values:
  - For example, dat
- **'ID_COL', 'RECORD_IDENTIFIER_FIELD'**
  Specifies the column that specifies the record type.
  Values:
  - number
- **'ID_OFFSET', 'ID_FROM', 'RECORD_IDENTIFIER_OFFSET'**
  Specifies the offset where the record type is found.
  Values:
  - number
- **'ID_LEN', 'ID_LENGTH', 'RECORD_IDENTIFIER_LENGTH'**
  Specifies the length of the record identifier.
  Values:
  - number
- **'UNRELATED', 'UNRELATED_RECORD_TYPES'**
  Specifies the record types that are unrelated.
  Values:
  - The numbers of record types that are unrelated.
- **'RELATED', 'RELATED_RECORD_TYPES'**
  Specifies the record types that are related.
  Values:
  - Pairs of record numbers that are linked by – and separated by semi-colons
    For example, 2-3;4-6

Register Capture/Replay Definition

If your object definition type includes capture or replay definition files, follow these steps to register them:

1. Open the Datamaker UI.
2. Select the appropriate project context in which you want to register the tables from the **Context** drop-down list.
3. Select **Projects, Register** from the main menu.
   A dialog listing all the possible types of data objects that you can register opens.
4. Select the **Capture/Replay Definition** option and click the forward arrow button next to **Select Type**.
   The **Register/Import Capture Replay** dialog opens.
5. Click the ellipsis icon to locate the file that you want to use.
6. Click **Show Worksheets** to see the contents of the selected file and perform the following steps as appropriate:
   - If a Microsoft Excel file is chosen, the test name defaults to the Microsoft Excel file name. You can also enter the test name manually. If a CSV file is chosen, enter the test name.

The table is registered as test_name(worksheet name); for example, FF_Payments(Payment301_3).
- If the current context is at a data level, you can use the **Import Any Data** options to import available data. To define the data included, click **Parameters**.
- If you are re-registering a file, table differences are indicated with a question mark icon. Click the icon to show differences. A tick mark icon indicates if the table structures are the same.

7. To Select all the records, click the **Select All** button, or select each record individually. To deselect all records, use the **Deselect All** button.
8. Click **Register** to register the files.
   You can assign normal masking functions in the same way as for RDBMS maps.

Register Excel/CSV Files

If your object definition type includes Microsoft Excel or CSV files, follow these steps to register the files:

1. Open the Datamaker UI.
2. Select the appropriate project context in which you want to register the tables from the **Context** drop-down list.
3. Select **Projects, Register** from the main menu.
   A dialog listing all the possible types of data objects that you can register opens.
4. Select the **Excel/CSV** option and click the forward arrow button next to **Select Type**.
   The **Register Delimited File** tab opens next to **Select Type**.
5. Click the ellipsis icon to locate the file that you want to use.
6. Click **Show Worksheets** to see the contents of the selected file.
   Note: To provide record names in Workbook (Worksheet), select the **Qualify** option.
7. Click the FD icon to view detailed information about the file.
8. To select all records, click the **Select All** button, or by select each record individually. To deselect all the records, use the **Deselect All** button.
9. Click **Register** to register the file.

Register Pervasive Files

Pervasive Data Integrator is an ETL (extract, transform, load) and data management solution. The Data Integrator toolset allows for the definition of a vast array of data sources and targets. These definitions are stored in SML files as Structured Schema (SS) or Document Schema (DS) definitions. These files are stored with a suffix of .ss.xml and .ds.xml respectively. The CA TDM Datamaker UI lets you import definitions in the Pervasive toolset to import, generate, or mask data.

1. Open the Datamaker UI.
2. Select the appropriate project context in which you want to register the tables from the **Context** drop-down list.
3. Select **Projects, Register** from the main menu.
   A dialog that lists all the possible data objects types that you can register opens.
4. Select the **Pervasive file descriptor (SS or DD)** option and click the forward arrow button next to **Select Type**.
   The **Register Pervasive File** tab opens next to **Select Type**.
5. Select the file type **SS** or **DS** in the **Register File** area.
6. Click the ellipsis icon to locate the file.
7. Click **Show Record Types** to see what is contained within records.
8. Select the **Header** or the **Trailer** options. You can select both options.
9. Click **Register** to register the definitions and create test data.

Register XML Files

If your object definition type includes XML files, follow these steps to register them:

1. Open the Datamaker UI.
2. Select the appropriate project context in which you want to register the tables from the **Context** drop-down list.
3. Select **Projects, Register** from the main menu.
   A dialog listing all the possible types of data objects that you can register opens.
4. Select the **Simple XML File** option and click the forward arrow button next to **Select Type**.
   The **Select File to Register** dialog opens.
5. Browse to the XML file that you want to register.
6. Select the file and click **Open**.
   The tree structure of the XML file is displayed.
7. Select a node in the tree that represents a row in your registered table.
   A dialog displays the names of the tables to be registered.
8. Click **OK**.
   Note: You can overwrite tables that are already registered.
9. Click **Register**.

Table Relationships

Table relationships help you understand how tables are related. When you copy data, you can use entered relationships to identify related rows of data. You can also use the relationships to create test data in the test data repository.

You can use the following rules to add and configure missing relationships:

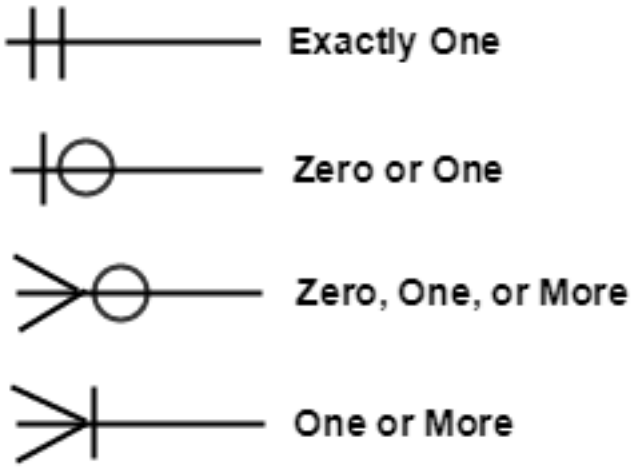
- Foreign keys
- Naming Standards
- CASE tools
- DDL
- By direct entry

After you enter a rule, you can select rows in one table and can edit related rows in related tables. Entered table relationships are related to a specific project. You can use rules across project versions and can selectively copy the rules from one project to another. You can also use these relationships to verify the integrity of the actual data in target or source.

To view existing table relationships, expand the plus symbol next to a table in the **Registered Objects** explorer. Database constraints are displayed with a symbol which indicates a database-enforced foreign key between the related tables. Any other manually entered table relationships are also displayed within the tree structure. You can also view table relationships through an Entity Relationship Diagram using GTDiagrammer in the Datamaker UI.

The cardinality options use the Crows Foot notation, which is shown in the following diagram:

Figure 26: Crows Foot Diagram



Add Relationships

You can manually add a relationship between tables. This ability gives you the flexibility to define relationships that are based on your unique scenarios and use cases. You can also add more SQL statements to relationships.

> **NOTE**
> Table relationships are independent of versions. When you register the same tables in different versions of a project, and add table relationships in one version, these relationships do not propagate to other versions. When you copy projects, relationships are copied, too. When you delete a version, it only deletes relationships for this version's tables. When you delete a project, it deletes all relationships for this project.

In the Datamaker UI, do the following:

1. Choose a project version, and click **Projects, Table Relationships** in the main menu.
   The **Table Relationships** dialog opens.
2. Click the **Registered Tables** tab.
3. Right-click the table in the left pane for which you want to create a relationship, and select **Add Relationship** from the context menu.
   The **Create Relationship from <*table_name*>** dialog opens.
4. Click the **Registered Tables** tab. Select the table with which you want to link your original table from the **To Table** drop-down list. Click the forward arrow icon.
   The **Create Relationship** dialog opens.
5. Select the cardinality of the table relationship from the drop-down list. The list is located above the two columns representing the two tables.
6. Double-click to add columns that link the two tables together.
   The added columns appear in the bottom pane.
   Note: To remove a column from the list, select the added column in the bottom pane and press the Delete key.
7. (Optional) To add more SQL to include in the relationship, select the **SQL** tab and add the appropriate information in the respective table columns.
   a. Click the parallel bar icon to select columns from the list that you want to include in your SQL.
   b. Click the tick mark icon to validate the SQL statement.

For example, to link to the Persons table from the Addresses table, list associated Persons when you select Addresses. To list Persons, include an extra check PRIMARY_IND='Y' (for example) in the *join*.
Note: Once you add a SQL statement, the statement is included as part of the rest of the application UI. For example, you can list the Persons when you select Addresses in the **SQL Window.**

8. Review your changes and click the **OK** button to create the rule.

To modify or delete a relationship,

1. Choose a project version, and click **Projects, Table Relationships** in the main menu.
   The **Table Relationships** dialog opens.
2. Click the **Registered Tables** tab and select a table.
3. Expand the 'Related Tables' Node, right-click the existing relationship, and perform one of the following steps:

- – Select **Edit Relationship** from the context menu to modify the details of a table relationship.
  – Select **Delete Relationship** from the context menu to delete a table relationship.

Import Relationships

Use the following methods to import table relationships into CA TDM:

Import Relationships from a Database

You might have information about table relationships available, but not as foreign keys in your database. To make this information available to CA TDM create a view as follows:

```
CREATE OR REPLACE VIEW gtdb_rel_v (
    rel_parent_owner,
    rel_parent_table,
    rel_parent_cardinality,
    rel_parent_sql,
    rc_seq,
    rc_parent_column,
    rel_child_owner,
    rel_child_table,
    rel_child_cardinality,
    rel_child_sql,
    rc_child_column,
    rel_name,
    rel_desc,
    rel_parent_add_sql,
    rel_child_add_sql
)
AS
SELECT <your columns>
FROM <your table or tables>
```

The following table shows the values to return in this view:

Column	Required	Default if NULL	Comment
rel_parent_owner	No	db user	-
rel_parent_table	Yes	-	The parent table.
rel_parent_cardinality	No	ONE	-

rel_parent_sql	No	-	-
rc_seq	No	0	Use multiple rows for keys with multiple parts. If you know the sequence of the columns in the key, specify the sequence.
rc_parent_column	Yes	-	-
rel_child_owner	No	db user	-
rel_child_table	Yes	-	-
rel_child_cardinality	No	ZERO_OR_MANY	Use ONE_OR_MANY for mandatory children (unusual).
rel_child_sql	No	-	-
rc_child_column	Yes	-	-
rel_name	No	<parent>_<child>_fk<n>	A unique name is created if one is not provided.
rel_desc	No	-	Useful to differentiate similar relationships.
rel_parent_add_sql	No	-	Use this to provide an additional parent query when doing a Join with SQL only.
rel_child_add_sql	No	-	Use this to provide an additional child query when doing a Join with SQL only.

Because all columns must exist in the view, return NULL if you do not have the required data.

Import Relationships from CASE Tools

Many CASE tools create a fully constrained schema from any entered relationships. The schema includes business relationships as normal foreign key constraints. After the CASE tool creates the DDL, you can use the created file to import relationships.

1. Open the Datamaker UI.
2. Select **Projects, Table Relationships** from the main menu.
3. In the relationships dialog, select **Extract from DDL** from the drop-down list in the top-right corner.
4. Click the forward arrow icon and select the file containing the generated DDL statements.
   The **Generated Relationships** dialog opens and displays foreign keys, if any.
5. Add rules that you want to import.

Validate Relationships

After you create relationships between tables, you can validate those relationships by using the following methods:

Check Table Relationships Against a Connection

1. Open the Datamaker UI.
2. Select **Projects, Table Relationships** from the main menu.
3. Select **Check Relationships in Data Source** (or **Check Relationships in Data Target**) from the drop-down list in the top-right corner.
   The **Check Relationships in Data Source** (or **Check Relationships in Data Target**) tab opens next to the existing tabs.

4. Select the tables that you want to validate against the source or target connection. You can use the table tags to select tables, or click each table. Use Ctrl + Click for multiple tables.
5. Click the forward arrow icon next to the drop-down list from where you selected the option in Step 3.
   The **Check Relationships and Constraints** dialog opens.
6. Click **Execute** to run the checks.
   SQL is created and runs against the source or target connection. Errors in the data are displayed in the **Check Relationships in Data Source** (or **Check Relationships in Data Target**) tab.
   Note: You can also decide to build SQL to execute the report later.
7. (Optional) To save the validation report, click the **Save Validation Report** icon. You can fix the erroneous data or can adjust the relationship.

Generate a Report on Relationships

CA TDM lets you produce a report of the relationships between selected tables.

1. Open the Datamaker UI.
2. Select **Projects, Table Relationships** from the main menu.
3. Select **Report on Relationships for tables** (or **Report on Relationships for registered tables**) from the drop-down list in the top-right corner.
4. Select the tables for which you want to create the report.
5. Click the forward arrow icon next to the drop-down list from where you selected the option in Step 3.
6. Report information is displayed in the **Select the Report on Relationships for tables** (or **Report on Relationships for registered tables**) tab.
7. To save the report as a validation report, click the **Save Validation Report** icon in the bottom right corner.
8. Review the report.

Export Relationships to a File

1. Open the Datamaker UI.
2. Select **Projects, Table Relationships** from the main menu.
3. Select **Export Relationships to file** from the drop-down list in the top-right corner.
4. Click the forward arrow icon next to the drop-down list from where you selected the option in Step 3.
5. Save the CSV file to a location in your directory.
6. Click **Open** to open and review the saved file or click **No**.

Create the Table Load Order

When you import data, it is important to load the data in the correct referential sequence. CA TDM maintains a load order that is based on the rules entered. You can rebuild this load order, or the order can be calculated automatically when you publish data.

1. Open the Datamaker UI.
2. Select **Projects, Table Relationships** from the main menu.
3. Select the **Recalculate table order** option from the drop-down list in the top-right corner of the dialog.
4. Click the forward arrow icon next to the drop-down list.
5. Click **Recalculate** to recalculate table order when you publish.
   If the table order already exists, click **Use** to use the existing order.
   Note: For tables with multiple relationships, this process might take a few minutes.

Build Relationships from Naming Standards

To build your relationships based on naming standards, use the **Names Parameters** tab options in the Datamaker UI. This tab helps you find column combinations for a list of tables that might be candidates to related tables.

The following example is a list of potential combinations built-up for tables named A, B, C, and D:

- A->B A->C A->D
- B->A B->C B->D
- C->A C->B C->D
- D->A D->B D->C

So *n* tables times *(n-1)* combinations.

1. Open the Datamaker UI.
2. Select **Projects, Table Relationships** from the main menu.
3. Select the **Generate from Names** option from the drop-down list in the top-right corner.
   The **Names Parameters** tab is added next to the existing tabs.
4. Click the **Names Parameters** tab and edit the following options:
   - **Primary Key Columns**
     Specifies the primary keys. This option identifies the list of columns in the source tables as potential to match with the target tables.
   - **Unique Key Columns**
     Specifies the unique keys. This option identifies the list of columns in the source tables as potential to match with the target tables.
   - **Uniquely Indexed Columns**
     Specifies the uniquely indexed columns. This option identifies the list of columns in the source tables as potential to match with the target tables.
   - **All Indexed Columns**
     Specifies all indexed columns. This option identifies the list of columns in the source tables as potential to match with the target tables.
   - **Add table name to ID**
     Adds the table name to the ID so that foreign keys follow the structure TABLE_NAME (_ID).
   - **Exact Match on Target Column Names**
     Searches only for the identical column names in the source and target table.
   - **If No matching column try without the first part of the name**
     Removes the first part of each name (before the first underscore) in the source and target columns. For example, columns CUST_LOC_ID and SHOP_LOC_ID would generate LOC_ID and the columns would match.
   - **If No matching column try without the last part of the name**
     Removes the last part of each name (before the last underscore) in the source and target columns. For example, columns LOC_ID_CUST and LOC_ID_SHOP would generate LOC_ID and the columns would match.
   - **Target column starts with source column name**
     Specifies that this option is useful if columns in the target table are suffixed by a table identifier; for example, LOC_ID would match with LOC_ID_CUST.
   - **Target column ends with source column name**
     Specifies that this option is useful if columns in the target table are prefixed by a table identifier; for example, LOC_ID would match with CUST_LOC_ID.
   - **Target column contains source column name**
     Matches columns where the source column is embedded within other columns; for example, LOC_ID is contained in the source column CUST_LOC_ID_CURRENT.
   - **Exact Match on Target Column data type**

Specifies that once columns are matched the data type of the columns must be equal.

- **Exact Match on Target Column data length and scale**
  Specifies that once columns are matched the data length and scale of the columns must be equal.
- **Find no more links if a link already exists between tables**
  Specifies that if the candidate tables are already linked by a foreign key or an entered relationship, the tables are ignored.
- **Accept all candidates**
  Auto creates all rules for selected tables and entered parameters. **Important**: Use this option with caution.

5. Select the relevant table for which you want to create a relationship based on the naming standard.
6. Click the forward arrow icon next to the drop-down list from where you selected the **Generate from Names** option.
   The **Generate Relationships from Names** dialog opens.
7. Click **Yes**.
   A dialog opens that displays candidate relationships, if any.
8. (Optional) To view the details of a rule, click on the rule. The matching columns are displayed in the bottom panel.
9. (Optional) To validate the relationship against the data in the target connection, right-click a rule and select **Check Relationships** from the context menu.
   The **SQL** tab in the **Check Relationship** dialog includes the SQL statement.
10. Enter Ctrl-X to execute the SQL statement to see if any invalid data is present in the target connection.
11. Close the **Check Relationship** dialog when you are done.
12. Click **Add All** if you are fine with all the displayed relationships.
    All the displayed rules are added and the next candidate table is displayed.
    Note: To add a single rule, click the **Add** button and then click the **Next Table** button. If **Add All Candidates** and **Add All** are selected, all relationships are added to the rules repository automatically. Use this option with caution.

Create Alias Tables

You can create an *alias* of a table using the Datamaker UI. An *alias* of a table is different from a *copy* of table. A copy is considered as a normal, independent table; whereas, an alias is always linked to the original table. An alias lets you copy foreign keys that are present in the original table to the alias table.

1. Open the Datamaker UI.
2. Select **Project, Project Manager** from the main menu.
3. Right-click the version where the table is registered and select **Actions for Registered Objects**.
4. In the middle pane under **Registered Tables**, select the required table that you want to use to create an alias.
   The selected table is highlighted.
5. Select **New Table** from the drop-down list in the top right.
6. Click the forward arrow icon next to the drop-down list.
7. Enter a name for the alias table and click **OK**.
   The **New Table** dialog opens.
8. Click **Alias**.
   The **New Alias Table** dialog opens prompting you to copy foreign keys.
9. Click **Yes**.
   The alias table is added to the tree hierarchy in the left pane.
10. Click the alias in the tree to display its definition.
    If the table has foreign keys, you are prompted to copy them; otherwise not. However, relationships are not copied. The reason for this is that, whatever you use your alias for, the alias must conform to the database constraint. An alias is created to denote alternative uses of a table. Tables used alternatively have different business rules and require different relationships than the parent table.
11. Verify whether a table is an alias using the following option:

- Right-click the table in the tree, select **Maintain Table**, and note the label **Alias** next to the table type.
- Click the **GT ALIAS** tag in the right pane to highlight all aliases.

12. To copy relationships from a table to its alias; follow these steps:
    a. Select **Projects, Table Relationships** to open the table relationships dialog.
    b. Select the **Export Relationships to file** option from the drop-down list in the top right and click the forward arrow icon.
    c. Edit the relationships file and save the file.
    d. Select the **Import Relationships from file** option from the drop-down list and click the forward arrow icon.
    e. Select the edited relationships file.
    Any relationships that can be imported correctly are selected by default.
    f. Click **Save**, then click **Add All**.
    g. Verify that the relationships are added to the alias.

TIP

To add one or two relationships to a table or an alias, right-click the table and select **Add Relationship** from the context menu.

Understand the Conditional Summation in Tables

If you have two tables, it is possible to do conditional summation. This example shows how to count the number of debits and the number of credits in two separate columns of a second table:

Table 1

The following table includes the individual credit amount, debit amount, and columns to be excluded:

Column Name	Col1	Col2	Credit_Amount	Debit_Amount
Description	Indicator (C or D)	Amount	Excluded column	Excluded column
Example DM Function	@randlov(0,@list(C,D)@)@	@randdigits(3,5)@	@if(^Indicator^=C,^Amount^,0)@	@if(^Indicator^=D,^Amount^,0)@

Table 2

The following table includes the total credit and debit amounts:

Column Name	Total_Credit	Total_Debit
Description	Sum of Credit_Amount of Table 1	Sum of Debit_Amount column of Table 1
Example DM Function	@sum(^EMP_Accounts.Credit_Amount^)@	@sum(^EMP_Accounts.Debit_Amount^)@

In these tables, C stands for credit and D stands for debit.

Two conditions are possible in this example. Col1 in Table 1 can contain either C or D. The conditional summation functions work as follows:

- If C in Col1, then copy Col2 in Credit_Amount.
- If D in Col1, then copy Col2 in Debit_Amount.

Whenever a column is excluded, it is not considered during the publish phase. To exclude a column, follow these steps:

1. Open the Datamaker UI.
2. Select **Projects, Project Manager** from the main menu.

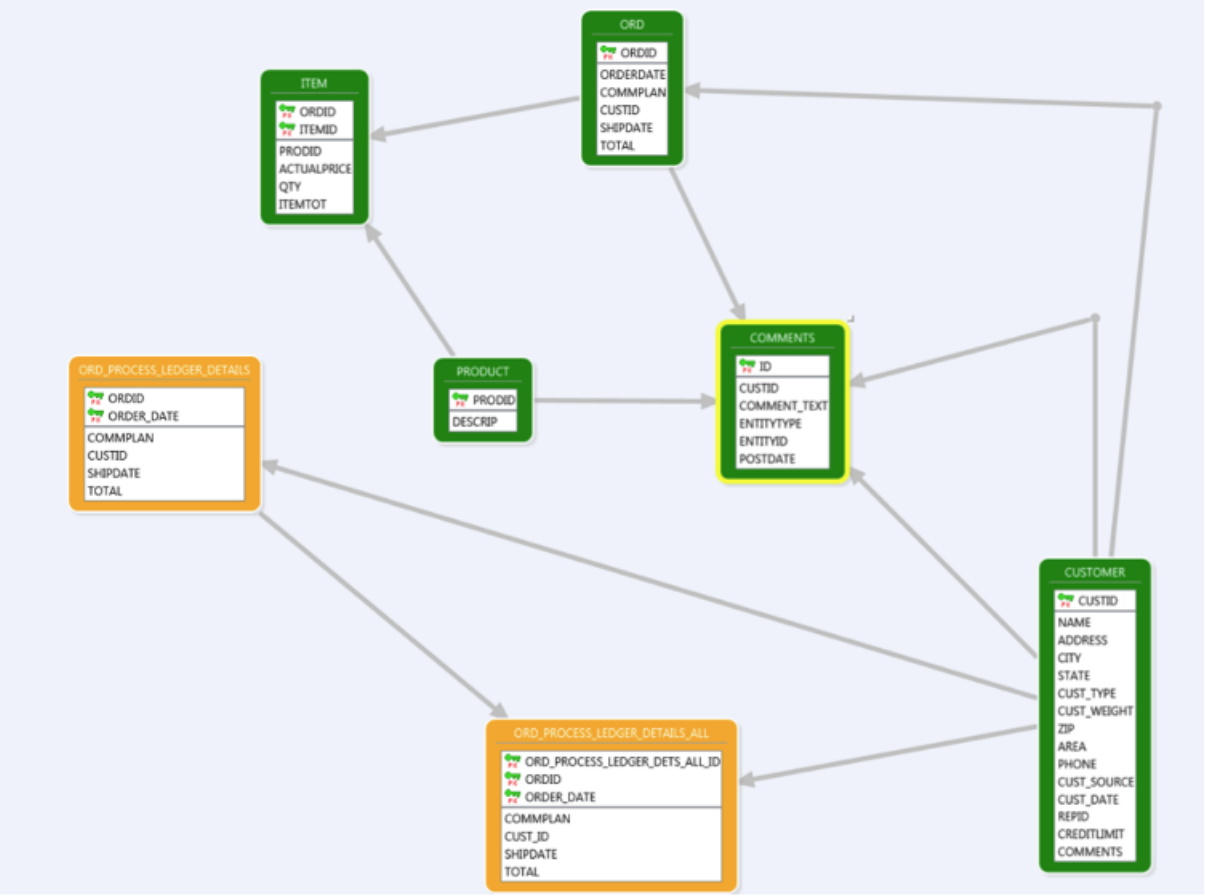
3. Right-click the version where the table is registered and select **Actions for Registered objects** from the context menu.
4. Right-click the table and select **Maintain Table** from the context menu.
5. Click the plus icon and add the column Excluded_Column.
6. Scroll to the right and select **All** from the drop-down list under **Exclude**.

Understand Conditional Relationships and Data Publishing

This section uses an example scenario to explain how to work with conditional relationships and publish relevant data.

Structure of the Tables

For this scenario, consider the tables in the following diagram:



This diagram shows that a foreign key relationship exists between COMMENTS and CUSTOMER. This relationship is represented by the following SQL statement:

```
ALTER TABLE COMMENTS ADD CONSTRAINT COMMENTS_CUSTOMER_FK FOREIGN KEY (CUSTID) REFERENCES
  CUSTOMER (CUSTID);
```

This SQL statement implies that every comment is customer-specific.

Relationships

Also, the following three extra relationships are added between CUSTOMER-COMMENTS, PRODUCT-COMMENTS, ORD-COMMENTS tables:

Details for Relationship CUSTOMER_COMMENTS_fk1

Parent Table: CUSTOMER

- Column: CUSTID

Child Table: COMMENTS

- Column: ENTITYID
- Child Table SQL: SELECT * FROM COMMENTS WHERE ENTITYTYPE = ‘CUSTOMER’

Details for Relationship ORD_COMMENTS_fk1

Parent Table: ORD

- Column: ORDID

Child Table: COMMENTS

- Column: ENTITYID
- Child Table SQL: SELECT * FROM COMMENTS WHERE ENTITYTYPE = ‘ORD’

Details for Relationship PRODUCT_COMMENTS_fk1

Parent Table: PRODUCT

- Column: PRODID

Child Table: COMMENTS

- Column: ENTITYID
- Child Table SQL: SELECT * FROM COMMENTS WHERE ENTITYTYPE = ‘PRODUCT’

These three relationships allow the records in COMMENTS to refer to the PRODUCT, CUSTOMER, and ORDER tables.

Data Translation by the Relational Editor

The relational editor translates the data in the first table to the data in the second table:

The following table includes the original data:

Row	Id	Custid	Comment Text	Entitytype	Entityid	Postdate
1	1	100	Oder 609 Customer 100 comment	ORD	609	2012-06-06 00:00:00
2	2	100	Customer 100 comment	CUSTOMER	100	2013-04-25 00:00:00
3	3	100	Product 100870 Customer 100 comment	PRODUCT	100870	2014-08-13 00:00:00
4	4	101	Order 610 Customer 101 comment	ORD	610	2013-04-22 00:00:00

5	5	101	Customer 101 comment	CUSTOMER	101	2013-01-25 00:00:00
6	6	101	Product 100860 Customer 101 comment	PRODUCT	100860	2013-08-27 00:00:00
7	7	101	Ledger 2 Customer 101 comment	LEDGER	2	2012-01-19 00:00:00
8	8	102	Order 602 Customer 102 comment	ORD	602	2012-02-06 00:00:00
9	9	102	Customer 102 comment	CUSTOMER	102	2014-12-27 00:00:00
10	10	102	Product 100870 Customer 102 comment	PRODUCT	100870	2014-05-10 00:00:00

The following includes the translated data:

Row	Id	Custid	Comment Text	Entitytype	Entityid	Postdate
1	~NEXT~	^CUSTOMER.CUSTID(1)^	Oder 609 Customer 100 comment	ORD	^ORD.ORDID(4)^	2012-06-06 00:00:00
2	~NEXT~	^CUSTOMER.CUSTID(1)^	Customer 100 comment	CUSTOMER	^CUSTOMER.CUSTID(1)^	2013-04-25 00:00:00
3	~NEXT~	^CUSTOMER.CUSTID(1)^	Product 100870 Customer 100 comment	PRODUCT	^PRODUCT.PRODID(3)^	2014-08-13 00:00:00
4	~NEXT~	^CUSTOMER.CUSTID(2)^	Order 610 Customer 101 comment	ORD	^ORD.ORDID(5)^	2013-04-22 00:00:00
5	~NEXT~	^CUSTOMER.CUSTID(2)^	Customer 101 comment	CUSTOMER	^CUSTOMER.CUSTID(2)^	2013-01-25 00:00:00
6	~NEXT~	^CUSTOMER.CUSTID(2)^	Product 100860 Customer 101 comment	PRODUCT	^PRODUCT.PRODID(1)^	2013-08-27 00:00:00
7	~NEXT~	^CUSTOMER.CUSTID(2)^	Ledger 2 Customer 101 comment	LEDGER	2	2012-01-19 00:00:00
8	~NEXT~	^CUSTOMER.CUSTID(3)^	Order 602 Customer 102 comment	ORD	^ORD.ORDID(1)^	2012-02-06 00:00:00
9	~NEXT~	^CUSTOMER.CUSTID(3)^	Customer 102 comment	CUSTOMER	^CUSTOMER.CUSTID(3)^	2014-12-27 00:00:00
10	~NEXT~	^CUSTOMER.CUSTID(3)^	Product 100870 Customer 102 comment	PRODUCT	^PRODUCT.PRODID(3)^	2014-05-10 00:00:00

Note: Because no relationship exists for this row, the LEDGER comment has not been transformed.

Data Generation

You can use this data pool to generate data. You might find it easier to use table aliases to generate the different types of COMMENTS.

In this example scenario, four aliases are created of COMMENTS, one for each related table. A default value for each alias is added to the ENTITYTYPE, and a relationship is created to the parent table.

Aliases Information

This section includes information about the four aliases of COMMENTS.

COMMENTS_CUSTOMER

- The following table includes information for the COMMENTS_CUSTOMER alias:

Name	Type	Nulls/Constraint	Default/Index
ID	number	NOT NULL	~NEXT~
CUSTID	number	NOT NULL	-
COMMENT_TEXT	nclob	NULL	-
ENTITYTYPE	varchar2(254)	NULL	CUSTOMER
ENTITYID	number	NULL	-
POSTDATE	date	NULL	-

The primary key information is as follows:

Name	Type	Nulls/Constraint	Default/Index
ID	Primary Key	SYS_C0027589(#1)	SYS_C0027589

The unique index information is as follows:

Name	Type	Nulls/Constraint	Default/Index
ID	Unique Index	-	SYS_C0027589(#1)

The foreign key information is as follows:

Name	Column	References	Pos
COMMENTS_CUSTOMER_FK	CUSTID	CUSTOMER.CUSTID	1

The relationship information is as follows:

Name	Column	References	Pos
CUSTOMER_COMMENTS_CUSTOMER_fk1	ENTITYID	CUSTOMER.CUSTID	1

COMMENTS_LEDGER

- The following table includes information for the COMMENTS_LEDGER alias:

Name	Type	Nulls/Constraint	Default/Index
ID	number	NOT NULL	~NEXT~
CUSTID	number	NOT NULL	-
COMMENT_TEXT	nclob	NULL	-
ENTITYTYPE	varchar2(254)	NULL	LEDGER
ENTITYID	number	NULL	-
POSTDATE	date	NULL	-

The primary key information is as follows:

Name	Type	Nulls/Constraint	Default/Index
ID	Primary Key	SYS_C0027589(#1)	SYS_C0027589

The unique index information is as follows:

Name	Type	Nulls/Constraint	Default/Index
ID	Unique Index	-	SYS_C0027589(#1)

The foreign key information is as follows:

Name	Column	References	Pos
COMMENTS_CUSTOMER_FK	CUSTID	CUSTOMER.CUSTID	1

The relationship information is as follows:

Name	Column	References	Pos
LEDGER_COMMENTS_LEDGER_fk1	ENTITYID	LEDGER.LEDGERNO	1

COMMENTS_ORD

- The following table includes information for the COMMENTS_ORD alias

Name	Type	Nulls/Constraint	Default/Index
ID	number	NOT NULL	~NEXT~
CUSTID	number	NOT NULL	-
COMMENT_TEXT	nclob	NULL	-
ENTITYTYPE	varchar2(254)	NULL	ORD
ENTITYID	number	NULL	-
POSTDATE	date	NULL	-

The primary key information is as follows:

Name	Type	Nulls/Constraint	Default/Index
ID	Primary Key	SYS_C0027589(#1)	SYS_C0027589

The unique index information is as follows:

Name	Type	Nulls/Constraint	Default/Index
ID	Unique Index	-	SYS_C0027589(#1)

The foreign key information is as follows:

Name	Column	References	Pos
COMMENTS_CUSTOMER_FK	CUSTID	CUSTOMER.CUSTID	1

The relationship information is as follows:

Name	Column	References	Pos
ORD_COMMENTS_ORD_fk1	ENTITYID	ORD.ORDID	1

COMMENTS_PRODUCT

- This heading includes information for the COMMENTS_PRODUCT alias:

Name	Type	Nulls/Constraint	Default/Index
ID	number	NOT NULL	~NEXT~
CUSTID	number	NOT NULL	-
COMMENT_TEXT	nclob	NULL	-
ENTITYTYPE	varchar2(254)	NULL	PRODUCT
ENTITYID	number	NULL	-
POSTDATE	date	NULL	-

The primary key information is as follows:

Name	Type	Nulls/Constraint	Default/Index
ID	Primary Key	SYS_C0027589(#1)	SYS_C0027589

The unique index information is as follows:

Name	Type	Nulls/Constraint	Default/Index
ID	Unique Index	-	SYS_C0027589(#1)

The foreign key information is as follows:

Name	Column	References	Pos
COMMENTS_CUSTOMER_FK	CUSTID	CUSTOMER.CUSTID	1

The relationship information is as follows:

Name	Column	References	Pos
PRODUCT_COMMENTS_PRODUCT_fk1	ENTITYID	PRODUCT.PRODID	1

The data in the COMMENTS tables is now recast as follows:

- **COMMENTS_CUSTOMER**

Row	Id	Custid	Comment Text	Entitytype	Entityid	Postdate
1	~NEXT~	^CUSTOMER.CUSTID(1)^	Customer 100 comment	CUSTOMER	^CUSTOMER.CUSTID(1)^	2012-04-25 00:00:00
2	~NEXT~	^CUSTOMER.CUSTID(2)^	Customer 101 comment	CUSTOMER	^CUSTOMER.CUSTID(2)^	2013-01-25 00:00:00
3	~NEXT~	^CUSTOMER.CUSTID(3)^	Customer 102 comment	CUSTOMER	^CUSTOMER.CUSTID(3)^	2014-12-27 00:00:00

- **COMMENTS_ORD**

Row	Id	Custid	Comment Text	Entitytype	Entityid	Postdate
1	~NEXT~	^CUSTOMER.CUSTID(1)^	Order 609 Customer 100 comment	ORD	^ORD.OTDID(4)^	2012-06-06 00:00:00
2	~NEXT~	^CUSTOMER.CUSTID(2)^	Order 610 Customer 101 comment	ORD	^ORD.OTDID(5)^	2013-04-22 00:00:00
3	~NEXT~	^CUSTOMER.CUSTID(3)^	Order 602 Customer 102 comment	ORD	^ORD.OTDID(1)^	2012-02-06 00:00:00

- **COMMENTS_PRODUCT**

Row	Id	Custid	Comment Text	Entitytype	Entityid	Postdate
1	~NEXT~	^CUSTOMER.CUSTID(1)^	Product 100870 Customer 100 Comment	PRODUCT	^PRODUCT.PRODID(3)^	2014-08-13 00:00:00
2	~NEXT~	^CUSTOMER.CUSTID(2)^	Product 100860 Customer 101 Comment	PRODUCT	^PRODUCT.PRODID(1)^	2013-08-27 00:00:00

3	~NEXT~	^CUSTOMER.CUSTID(3)^	Product 100870 Customer 102 Comment	PRODUCT	^PRODUCT.PRODID(3)^	2014-05-10 00:00:00

- **COMMENTS_LEDGER**

Row	Id	Custid	Comment Text	Entitytype	Entityid	Postdate
1	~NEXT~	^CUSTOMER.CUSTID(2)^	Ledger 2 Customer 101 Comment	LEDGER	2	2012-01-19 00:00:00

At the time of publishing, all these versions of the COMMENTS table are published into the one table. The following table is a representation of the **Publish Data Pool: Row Data to file** dialog:

Table Name	Table Location	Seq	Rows in Data Pool
PRODUCT	COURSE1	15	10
CUSTOMER	COURSE1	18	4
COMMENTS_ORD	COURSE1.COMMENTS	20	3
COMMENTS_LEDGER	COURSE1.COMMENTS	20	1
COMMENTS_CUSTOMER	COURSE1.COMMENTS	20	3
COMMENTS_PRODUCT	COURSE1.COMMENTS	20	3
ORD	COURSE1	22	10
ITEM	COURSE1	24	19

This publishing gives a SQL statement as follows:

```
-- Inserts for table COMMENTS

INSERT INTO COMMENTS ( ID, CUSTID, COMMENT_TEXT, ENTITYTYPE, ENTITYID, POSTDATE )

VALUES (1, 1, N'Order 609 Customer 100 comment', 'ORD', 4, '2012-06-06 00:00:00');

INSERT INTO COMMENTS ( ID, CUSTID, COMMENT_TEXT, ENTITYTYPE, ENTITYID, POSTDATE )

VALUES (2, 2, N'Order 610 Customer 101 comment', 'ORD', 5, '2013-04-22 00:00:00');

INSERT INTO COMMENTS ( ID, CUSTID, COMMENT_TEXT, ENTITYTYPE, ENTITYID, POSTDATE )

VALUES (3, 3, N'Order 602 Customer 102 comment', 'ORD', 1, '2012-02-06 00:00:00');

-- Inserts for table COMMENTS

INSERT INTO COMMENTS ( ID, CUSTID, COMMENT_TEXT, ENTITYTYPE, ENTITYID, POSTDATE )

VALUES (1, 2, N'Ledger 2 Customer 101 comment', 'LEDGER', 2, '2012-01-19 00:00:00');

-- Inserts for table COMMENTS

INSERT INTO COMMENTS ( ID, CUSTID, COMMENT_TEXT, ENTITYTYPE, ENTITYID, POSTDATE )
```

```
VALUES (1, 1, N'Customer 100 comment', 'CUSTOMER', 1, '2013-04-25 00:00:00');

INSERT INTO COMMENTS ( ID, CUSTID, COMMENT_TEXT, ENTITYTYPE, ENTITYID, POSTDATE )

VALUES (2, 2, N'Customer 101 comment', 'CUSTOMER', 2, '2013-01-25 00:00:00');

INSERT INTO COMMENTS ( ID, CUSTID, COMMENT_TEXT, ENTITYTYPE, ENTITYID, POSTDATE )

VALUES (3, 3, N'Customer 102 comment', 'CUSTOMER', 3, '2014-12-27 00:00:00');

-- Inserts for table COMMENTS

INSERT INTO COMMENTS ( ID, CUSTID, COMMENT_TEXT, ENTITYTYPE, ENTITYID, POSTDATE )

VALUES (1, 1, N'Product 100870 Customer 100 comment', 'PRODUCT', 3, '2014-08-13
 00:00:00');

INSERT INTO COMMENTS ( ID, CUSTID, COMMENT_TEXT, ENTITYTYPE, ENTITYID, POSTDATE )

VALUES (2, 2, N'Product 100860 Customer 101 comment', 'PRODUCT', 1, '2013-08-27
 00:00:00');

INSERT INTO COMMENTS ( ID, CUSTID, COMMENT_TEXT, ENTITYTYPE, ENTITYID, POSTDATE )

VALUES (3, 3, N'Product 100870 Customer 102 comment', 'PRODUCT', 3, '2014-05-10
 00:00:00');
```

Understand, Access, and Use the SQL Window

The SQL window in the Datamaker UI lets you run SQL statements against the source data or target data. Each SQL window includes the following tabs:

- SQL
  Lets you enter your SQL.
- Status
  Reports on the status of the SQL, how long it ran, and whether it was successful.
- Data in <table_name>
  Shows your returned data, which you can view and edit.

Above the SQL window, you can find a toolbar that includes various icons. You can use these icons to perform different functions. To know the functionality of each icon, review the tooltip that is associated with the icon.

Note: To perform the steps that are mentioned in this article, ensure that you connect to the target or source schema.

1. Open the Datamaker UI and select **Data Target, Data Target SQL Window** from the main menu.
   Note: For the source schema, select **Data Source, Data Source SQL Window** from the main menu.
   The **SQL Window** dialog opens. The dialog contains a schema explorer in the left pane and a SQL window in the right pane.

In the schema explorer, you can expand schemas, tables, views, and synonyms. For example, clicking on a column displays the columns within that table. You can drag and drop objects from the schema explorer to the SQL window as required.
If you click the Hide/Show details pane icon and click on a table, the **List View** panel is displayed. This panel displays index, data type, null, and foreign key details. Click the same icon to hide the view.

2. In the **SQL** tab, enter the name of a table below the SELECT * FROM by typing it or dragging it from the schema explorer.
3. If you want to create another piece of SQL, select the **New** tab next to the **SQL #1** tab that you are currently using. This action creates another tab where you can enter new SQL. You can have multiple tabs open at once. To switch between tabs, click on the numbered SQL tab to change the focus.
4. To execute the SQL, press Alt-X.
   The **Data in <table_name>** tab includes the result for the SQL statement that you executed.

You can also use the shortcut keys to perform actions. The following shortcut keys are supported:

- **F6**
  If you move the cursor over the * in the SELECT * FROM CUSTOMER and press F6, the * is expanded to include all the columns in the table or view.
- **Alt –X**
  Execute the current SQL.
- **Alt – A**
  Execute all the SQL in sequence.

Modify Data (SQL Window)

You can modify data in the SQL window as follows:

Update Rows in the Data Window

1. Open the Datamaker UI and select **Data Target, Data Target SQL Window** from the main menu.
   Note: For the source schema, select **Data Source, Data Source SQL Window** from the main menu.
   The **SQL Window** dialog opens. The dialog contains a schema explorer in the left pane and a SQL window in the right pane.
2. In the **SQL** tab, enter the name of a table below the SELECT * FROM by typing it or dragging it from the schema explorer.
   If you want to create another piece of SQL, select the **New** tab next to the **SQL #1** tab that you are currently using. This action creates another tab where you can enter new SQL. You can have multiple tabs open at once. To switch between tabs, click on the numbered SQL tab to change the focus.
3. To execute the SQL, press Alt-X.
   The **Data in <table_name>** tab includes the result for the SQL statement that you executed.
4. Edit any column that you want to change and tab out of the cell.
   The color of the line changes.
5. Click the save icon to commit your changes.
6. (Optional) If you want to discard the changes, close the SQL window, re-execute the SQL statement, or click the back arrow icon.
   A confirmation dialog opens prompting you to confirm whether you want to discard the changes.

Delete Rows in the Data Window

1. Open the Datamaker UI and select **Data Target, Data Target SQL Window** from the main menu.
   Note: For the source schema, select **Data Source, Data Source SQL Window** from the main menu.

The **SQL Window** dialog opens. The dialog contains a schema explorer in the left pane and a SQL window in the right pane.

2. In the **SQL** tab, enter the name of a table below the SELECT * FROM by typing it or dragging it from the schema explorer.
   If you want to create another piece of SQL, select the **New** tab next to the **SQL #1** tab that you are currently using. This action creates another tab where you can enter new SQL. You can have multiple tabs open at once. To switch between tabs, click on the numbered SQL tab to change the focus.
3. To execute the SQL, press Alt-X.
   The **Data in <table_name>** tab includes the result for the SQL statement that you executed.
4. Select the row that you want to delete and click the delete icon.
   The row is deleted from the returned set of data.
5. Click the save icon to commit your changes.
   The row is deleted permanently.

Insert Rows into the Data Window

1. Open the Datamaker UI and select **Data Target, Data Target SQL Window** from the main menu.
   Note: For the source schema, select **Data Source, Data Source SQL Window** from the main menu.
   The **SQL Window** dialog opens. The dialog contains a schema explorer in the left pane and a SQL window in the right pane.
2. In the **SQL** tab, enter the name of a table below the SELECT * FROM by typing it or dragging it from the schema explorer.
   If you want to create another piece of SQL, select the **New** tab next to the **SQL #1** tab that you are currently using. This action creates another tab where you can enter new SQL. You can have multiple tabs open at once. To switch between tabs, click on the numbered SQL tab to change the focus.
3. To execute the SQL, press Alt-X.
   The **Data in <table_name>** tab includes the result for the SQL statement that you executed.
4. Click the plus icon to insert rows.
5. Enter the number of rows you want to insert and click the tick mark icon.
   The specified number of rows are added to the table in the UI.
6. Enter appropriate values in the rows and save your changes by clicking the save icon.
   Note: Cells in red are mandatory; whereas, cells in yellow are not.

Perform Tasks Based on the Object Type

1. Open the Datamaker UI and select **Data Target, Data Target SQL Window** from the main menu.
   Note: For the source schema, select **Data Source, Data Source SQL Window** from the main menu.
   The **SQL Window** dialog opens. The dialog contains a schema explorer in the left pane and a SQL window in the right pane.
2. Select an object in the schema explorer.
3. Click the tasks icon at the bottom of the left pane.
   A context menu displays various tasks that are based on the selected object type.
   For example, if you select a table, you can find tasks (for example) count rows, drop indexes, and drop table. You can further perform maintenance tasks on individual tables (for example, add column, modify column, drop columns).

Save the Modified Data in an Appropriate File Format

1. Open the Datamaker UI and select **Data Target, Data Target SQL Window** from the main menu.
   Note: For the source schema, select **Data Source, Data Source SQL Window** from the main menu.
   The **SQL Window** dialog opens. The dialog contains a schema explorer in the left pane and a SQL window in the right pane.

2. In the **SQL** tab, enter the name of a table below the SELECT * FROM by typing it or dragging it from the schema explorer.
   If you want to create another piece of SQL, select the **New** tab next to the **SQL #1** tab that you are currently using. This action creates another tab where you can enter new SQL. You can have multiple tabs open at once. To switch between tabs, click on the numbered SQL tab to change the focus.
3. To execute the SQL, press Alt-X.
   The **Data in <table_name>** tab includes the result for the SQL statement that you executed.
4. Select **SQL -> Save As** from the main menu.
   The **Save As** dialog opens.
5. Navigate to the location where you want to save the file.
6. Enter an appropriate name for the file and select the file format from the **Save as type** drop-down list.
7. Click **Save** to save the file that contains all the data.

Review Column-Editing Functions (SQL Window)

Various column-editing functions are available that you use to perform specific tasks that are related to columns in the SQL window.

1. Open the Datamaker UI and select **Data Target, Data Target SQL Window** from the main menu.
   Note: For the source schema, select **Data Source, Data Source SQL Window** from the main menu.
   The **SQL Window** dialog opens. The dialog contains a schema explorer in the left pane and a SQL window in the right pane.
2. In the **SQL** tab, enter the name of a table below the SELECT * FROM by typing it or dragging it from the schema explorer.
   If you want to create another piece of SQL, select the **New** tab next to the **SQL #1** tab that you are currently using. This action creates another tab where you can enter new SQL. You can have multiple tabs open at once. To switch between tabs, click on the numbered SQL tab to change the focus.
3. To execute the SQL, press Alt-X.
   The **Data in <table_name>** tab includes the result for the SQL statement that you executed.
4. Select a column and right-click on it.
   A context menu with different options opens.
5. Review the options as follows:
   Note: This step includes only those options that are no self-explanatory.
   - **Copy Row Down**
     Copies the value of every column within the row to all rows/new rows directly below it.
   - **Copy Column Down**
     Copies the value of the cell to all rows directly below it within that column.
   - **Copy Column Down with Increment**
     Lets you copy and increase the value from the selected column to the rows directly below it within that column. The increase can only apply to numeric and date type columns. You are prompted to define an incremental value.
   - **Increment**
     Adds a defined value to the value in the specific cell. The increase can only apply to numeric and date type columns.
   - **Randomize – Range**
     Randomizes the value within a defined range for all rows/new rows within a column.
   - **Randomize – Seed Values**
     Uses a random value from a seed list for all rows/new rows within a column. For example, use a random value from the chosen seed list and click the tick mark icon to apply the randomization to the column.
   - **Sequential – Range**

Uses a sequential list of values within a defined range for all rows/new rows within a column. For example, use a sequential list of IDs from 23 through 47. Click **OK** to apply the range to the column.

- **Sequential – Seed Values**
  Uses a sequential list of values from a seed list for all rows/new rows within a column. For example, use first names from the list, in order. Click **OK** to apply the randomization to the column.
- **Find Value in Tables**
  Searches selected tables to find and display a specific value.
- **Set Null**
  Sets the column to null; it is displayed red or yellow depending on the mandatory status.
- **Re-Query**
  Re-executes the current SQL in the **SQL** tab without re-parsing it.
- **Modify Date Format**
  Lets you enter any valid Microsoft Windows date and time syntax.
- **Sort**
  Sorts your result set without having to use ORDER BY in your SQL clause. You can drag and drop the columns as required in the **Specify Sort Columns** dialog.
  Clearing the **Ascending** option causes the sort to be in the descending order. Double-clicking allows you to sort on a function.
- **Filter**
  Lets you filter your search set by entering a further criteria in the **Specify Filter** dialog.
- **Autowidth**
  Expands or contracts the column so that it can accommodate the data in the column. The column cannot be expanded to more than 2/3 the width of the window.
- **Column Details**
  Reports the details of the column definition from the database.
  Right-clicking in the data window outside a column displays a different drop-down list.
- **Datawindow Details**
  Displays the internal information of all the columns within the data window.

Write Data (SQL Window)

After you select some data in the SQL window, you can write that data out to other locations. For example, you can write the data present in the current SQL window to another table using the source or target connection, as appropriate.

1. Open the Datamaker UI and select **Data Target, Data Target SQL Window** from the main menu.
   Note: For the source schema, select **Data Source, Data Source SQL Window** from the main menu.
   The **SQL Window** dialog opens. The dialog contains a schema explorer in the left pane and a SQL window in the right pane.
2. In the **SQL** tab, enter the name of a table below the SELECT * FROM by typing it or dragging it from the schema explorer.
   If you want to create another piece of SQL, select the **New** tab next to the **SQL #1** tab that you are currently using. This action creates another tab where you can enter new SQL. You can have multiple tabs open at once. To switch between tabs, click on the numbered SQL tab to change the focus.
3. To execute the SQL, press Alt-X.
   The **Data in <table_name>** tab includes the result for the SQL statement that you executed.
4. Select the location where you want to copy the data. If you select **Write to Test Case Repository**, you write the data into your current data hierarchy depending on your current context.
   The import of the data starts. After the data is imported, you receive a confirmation message.
5. Click **OK**.

Note: If the target table has an extra column, which is preventing the import from working, you can delete the column using the data editing function.

Import Data from a File (SQL Window)

You can import the data from a tab-separated file into the data window.

1. Open the Datamaker UI and select **Data Target, Data Target SQL Window** from the main menu.
   Note: For the source schema, select **Data Source, Data Source SQL Window** from the main menu.
   The **SQL Window** dialog opens. The dialog contains a schema explorer in the left pane and a SQL window in the right pane.
2. In the **SQL** tab, enter the name of a table below the SELECT * FROM by typing it or dragging it from the schema explorer.
   If you want to create another piece of SQL, select the **New** tab next to the **SQL #1** tab that you are currently using. This action creates another tab where you can enter new SQL. You can have multiple tabs open at once. To switch between tabs, click on the numbered SQL tab to change the focus.
3. To execute the SQL, press Alt-X.
   The **Data in <table_name>** tab includes the result for the SQL statement that you executed.
4. Select **SQL, Import From** from the main menu.
   The **Select Import File** dialog opens.
5. Browse to the location of the file and click **Open**.
   The import process starts.
6. After importing into the data window completes, click the commit icon to save the data into the current connection.

Working with Registered Objects

After you register objects to CA TDM, you can add information to the meta-model repository by performing appropriate actions for registered objects.

Set a Primary Key Descriptor

By setting a primary key descriptor, you can identify which columns describe a table.

1. Open the Datamaker UI and select **Projects, Actions for Registered Objects** from the main menu.
2. Right-click the table in the left pane and select **Set PK descriptor** from the context menu.
   The **Specify Table Primary Key Descriptor** dialog opens.
3. In the **Column** tab, select any column that you want to display with the primary key by dragging the column into the selection area at the bottom of the dialog.
4. If you want to concatenate columns, use the standard SQL syntax in the **SQL** tab. An example is as follows:
   ```
   (SELECT S.NAME ||'-' ||C.NAME||'-'||THIS.name FROM <<CITIES>>C,<<COUNTRIES>>S WHERE THIS.CITY_ID=C.ID AND C.CON_ID=S.ID)
   ```
   The format of the SQL must follow these rules:
   - Begin and end the SQL with brackets '(' and ')'.
   - The current table columns must be prefixed with 'THIS.'
   - Referenced tables must begin and end with angled brackets '<<' and '>>'.
5. Click the tick mark icon to verify the format of the clause against the data target or data source.

Working with Columns

You can work with and can manage registered columns in the Datamaker UI as follows:

Add Lists of Values to Columns

1. Open the Datamaker UI and select **Projects, Actions for Registered Objects** from the main menu.
2. Navigate the table hierarchy in the left pane and select the column to which you want to add permitted values.
3. Right-click the selected column and select **Edit Permitted Values** from the context menu.
   The **Edit Values for: <table_name>.<column_name>** dialog opens.
4. Click the plus icon to add a new permitted value.
5. Select a value (MIN, MAX, VALUE, AVG, STDDEV, and MED) from the **Type** drop-down list.
6. Enter the value in the **Value** field.
7. Select a source (DDL, PROD, DEV, and INVALID) from the **Source** drop-down list. The source is used to group values so that you can select them during data entry. The source of DEV and PROD can be mass populated by extracting data characteristics from development or production databases.
8. For values that require additional information, use the **Description** field to add notes for the value. For example, the card type AX would have the description American Express.

Set Default Values

You can set the default value for a column. Default values are important because they are used to populate NOT NULL columns with a value when you publish the data.

1. Open the Datamaker UI and select **Projects, Actions for Registered Objects** from the main menu.
2. Navigate the table hierarchy in the left pane and select the column for which you want to set a default value.
3. Right-click the selected column and select **Edit Permitted Values** from the context menu.
   The **Edit Values for: <table_name>.<column_name>** dialog opens.
4. Select a value from the list and click the star icon to set that value as a default value for the column.
5. (Optional) To deselect a default value, click the star icon that is marked with a cross.

You can also use sequences or identity columns to set the default value. For example, if you want to use an Oracle sequence when you publish, with a ~NEXT~ variable, set the default value of the column as shown in the following table (for example):

Type	Value	Source	Description
VALUE	SOFTPROJ_SEQ.NEXTVAL	DEFAULT	-

The ~NEXT~ variable identifies the next value from the sequence and uses the value when you publish; any ~PARENT() ~ values referring to this are set to the same value.

Create a Variable from a Column

1. Open the Datamaker UI and select **Projects, Actions for Registered Objects** from the main menu.
2. Navigate the table hierarchy in the left pane and select the column from which you want to create a variable.
3. Right-click the selected column and select **Create Variable From Column** from the context menu.
   The **New variable for Project Version** dialog opens.
4. Select the type of variable from the **Type** drop-down list.
5. Enter the variable length using the **Max Length** and **Min Length** fields.
6. Save your changes.

Rename a Column

At times, you change the name of columns between releases. To carry forward any stored data from a previous release to the later release, update a cross-reference table in the Datamaker UI.

1. Open the Datamaker UI and select **Projects, Actions for Registered Objects** from the main menu.
2. Select the **Rename columns** option from the drop-down list in the top-right corner.
3. Click the forward arrow icon next to the drop-down list.
   The **Rename columns** dialog opens.
4. Click the plus icon to open a new row.
5. Enter the table name, the original column name, the new column name, and select the version from the **Version** drop-down list. Repeat this action for as many columns as you want to rename.
6. To delete a rename added by mistake, use the **Delete row** icon.
7. Click the save icon to save your changes.

Working with Tables

You can work with and can manage registered tables in the Datamaker UI as follows:

Create a New Table

You can create a new table from multiple data objects.

1. Open the Datamaker UI and select **Projects, Actions for Registered Objects** from the main menu for the selected version.
2. Select the tables that you want to use to create the new table from the middle pane.
3. Select **New Table** from the drop-down list in the top-right corner and click the forward arrow icon.
4. Enter the name of the table and click **OK**.
5. Click **Yes** to confirm that you want to create a new table.

Open Tables in a Data Source

1. Open the Datamaker UI and select **Projects, Actions for Registered Objects** from the main menu for the selected version.
2. Select the tables that you want to open in the **Data Source SQL** tab from the middle pane.
3. Select the **Open Tables in Data Source** option from the drop-down list on the top-right corner.
4. Click the forward arrow icon.
   The selected tables open in the **Data Source SQL** tab.

Open Tables in a Data Target

1. Open the Datamaker UI and select **Projects, Actions for Registered Objects** from the main menu for the selected version.
2. Select the tables that you want to open in the **Data Target SQL** tab from the middle pane.
3. Select the **Open Tables in Data Target** option from the drop-down list in the top-right corner.
4. Click the forward arrow icon.
   The selected tables open in the **Data Target SQL** tab.

Register PKs from DDL

You can register primary keys from DDL when constraints are not enabled in the database but fully constrained DDL is available. This is often the case when database design tools have been used.

1. Open the Datamaker UI and select **Projects, Actions for Registered Objects** from the main menu for the selected version.
2. Select the tables to which you want to register the primary keys from the middle pane.
3. Select the **Register PKs from DDL** option from the drop-down list in the top-right corner.
4. Click the forward arrow icon.
5. Browse to the DDL file from which you want to register the primary keys.
6. Select the file and click **Open**.
7. Click **OK** on the confirmation dialog.

Copy Tables to Another Project

If a table in one project version includes information that is applicable to another project version, you can copy the table from one project version to another.

1. Open the Datamaker UI and select **Projects, Actions for Registered Objects** from the main menu for the selected version.
2. Select the tables that you want to copy to another project version from the middle pane.
3. Select **Copy Tables to Another Project** from the drop-down list in the top-right corner.
4. Click the forward arrow icon.
   The **Select required version** dialog opens.
5. Select the appropriate project version and click the tick mark icon.
6. Click **OK** on the confirmation dialog.

Create Hidden Columns for Data Creation

You can create hidden columns in a table that exist exclusively within CA TDM (Datamaker UI) for synthetic data generation.

1. Open the Datamaker UI and select **Projects, Actions for Registered Objects** from the main menu for the selected version.
2. Select the table to which you want to add the hidden column from the left pane.
3. Right-click the selected table and select **Maintain Table** from the context menu.
   The **Maintain Table <table_name>** dialog opens.
4. Click the plus icon to add the new column.
5. Enter the name of the column and update other properties as appropriate.
6. Click the save icon to save your changes.
   The new column is added to the table as a hidden column.

GT Diagrammer

GT Diagrammer allows the visualization of a Database Schema (represented in one of a number of file formats or loaded from a web service) into an Entity-Relationship Diagram. The Diagrammer allows users to add, remove and edit existing tables (referred to as objects) and relationships (referred to as rules) in the context of the diagram, as well as to add labels on top of it.

Navigation through large diagrams is achieved by dragging objects around, panning and zooming. The objects (tables) and labels can be re-positioned by dragging and dropping. Multiple objects can be dragged together by selecting them and dragging one of them. The rules (relationships) will follow the objects they are connected to, should the latter be moved, but they cannot be dragged themselves. Panning is done by clicking the mouse on an empty space (not an object, rule or label) and dragging it.

On the top-left side of the diagram, there is a zoom control with a slider and two buttons, which can be used to zoom in and out of the diagram. The first button on the zoom control will set the zoom ratio to 1:1, while the second will set it to fit the whole diagram in the window. Additionally, holding the Ctrl key and scrolling the mouse wheel will also zoom in and out, while holding the Alt key and clicking and dragging a box with the mouse will zoom in to the outline of that box.

The information contained in the objects can be expanded to three levels: the table name; the table name and all the column names; or the table name, all the column names, their respective SQL type and whether they can be set to NULL or not (or their offset, in the case of a diagram loaded from a DBD file). This is done using the View menu submenus. In addition, hovering the mouse cursor over an object's table name will show the object's properties and hovering over a specific column will show its details. The information about a rule (the columns that are joined) can be seen by hovering the mouse cursor over that rule.

Highlighting and Selecting Diagrams

When the mouse cursor hovers over an object or a rule, it becomes highlighted in purple. Its parent objects (and rules connecting to them) become highlighted in brown and its children objects (and rules connecting to them) become highlighted in blue. Likewise, when a column has been hovered over, it will also be highlighted in purple (in addition to the object it belongs to). All columns in other objects related to the hovered column are highlighted in the corresponding colour - brown if the column is a parent of (is referenced by) the hovered column and blue if the column is a child of (is a reference of) the hovered column. This feature can be disabled, or colors changed, from the View Menu.

Selecting can be done by left-clicking on an item (object, rule, or label). By holding the Ctrl key and left-clicking an item, it will be added to the selected items (if it was not selected before). If it was selected before, it is removed from the selected items. Alternatively, holding the Ctrl key and clicking and dragging a rectangular box selects all objects and labels within that box, and all rules connecting objects that are within the box. In addition, right-clicking an item selects it if it was not selected. If it was already selected (all items that were selected will stay selected), nothing will change. Left-clicking (or right-clicking) empty space unselects all items.

Selected items can be distinguished as follows:

- Objects and labels have their border color changed to the selection color and their border thickness increased.
- Rules have their background changed to the selection color and their thickness increased. The default selection color is yellow, but it can be changed from the View menu.

GT Diagrammer Capabilities

Context Right-click Menus

EMPTY SPACE CONTEXT MENU

When an empty space is right-clicked, the context menu contains three items:

- *Add Object* (keyboard shortcut: *Ctrl+Shift+O*)
  This item has the same functionality as the Tools Menu submenus of the same names (see the Tools Menu section).
- *Add Rule* (*Ctrl+Shift+R*)
  This item has the same functionality as the Tools Menu submenus of the same names (see the Tools Menu section).
- *Add Label Here* (*Ctrl+Shift+L*).

This item allows a label to be added to the right-clicked spot. Once selected, a dialog window opens, asking for the text to appear on the label. If the confirm button is clicked, the label appears at that spot.

LABELS CONTEXT MENU

When a label is right-clicked, the context menu contains two items:

- *Delete Selected* (keyboard shortcut: *Del*)
  Delete all selected items (objects, rules, and labels) from the graph (**NOTE:** All selected items are removed, not only the right-clicked one).
- *Choose Default Color*
  Used to change the default color of all the selected items (**NOTE**: this changes the default color of all selected items, not only the right-clicked one). This refers to changing the default background color of objects and rules and the border color of labels.

RULES CONTEXT MENU

When a rule is right-clicked, the context menu contains three items. The first two items are the same as when a label is right-clicked.

- *Select Connected*
- *Select Ancestors*
  Select all ancestor items (objects and rules) of the right-clicked item.
- *Select Descendants*
  Select all its descendant items.
- *Select All Connected*
  Select all items that are somehow connected to the right-clicked item (**NOTE**: not only its descendants and ancestors but all their descendants and ancestors)

OBJECTS CONTEXT MENU

When an object is right-clicked, the context menu contains five (or six) items. Items 1, 2, and 6 are the same as when a rule is right-clicked.

- *Edit Object*
  This item allows the editing of the right-clicked object. A dialog window opens, from where the table name and default color of the object can be changed, and the list of columns (two lists, one with primary keys and one with non-primary-key attributes) can be added to or removed from. The principle for adding a column is the same as when adding a column to a new object. See Tools Menu section for details. Once the confirm button has been clicked, the changes appear on the edited object.
- *Add New Rule*
  Used to add a rule with the right-clicked object as the parent (see *Add New Rule* in the Tools Menu section). In case a connection to a web service has been established (see the *Connect* subsection of the File Menu section for details)
- *Load Related*
  Used to add tables that are related to the right-clicked object to the diagram by retrieving them from the web service.

File Menu

NEW

This item creates a new, empty diagram and deletes the existing one (unless saved). A dialog asking for confirmation appears before the existing diagram is replaced. The layout is automatically changed to ISOM (see the Layout Menu for details). Keyboard shortcut: *Ctrl+N*.

CONNECT

Connect to a Test Data on-Demand web service. For this option to work, a file with the web service details must have been provided at the application start-up. This happens automatically when the GTDiagrammer has been opened directly from within DataMaker. Once clicked, a dialog asking for confirmation appears.

Note: This option loads a new empty diagram and the previous one is lost if not saved. If confirmed, a list of all tables that have been retrieved from the web service is shown. The user can then select which of the tables to load into the diagram.

When a table from the web service has been loaded as an object on the diagram, right-clicking it has one extra menu option – *Load Related*. Once clicked, this menu option shows a list of tables that are related to the right-clicked object (child tables with yellow background and parent tables with red background), as retrieved from the web service. The user can select which of to load.

Note: All relationships that exist between newly added tables and tables already in the diagram are automatically added as rules when the new object is created. This includes not only the relationship with the object that was right-clicked to add this new table.

LOAD

Load one of four types of files to be used to create a diagram:

- Data files containing database information in the Silwood format (.dat or .txt or any other file format).
- GraphML files (.gml), a type of XML file which has been saved using the GTDiagrammer. If a corresponding label XML file exists, it is loaded too (See Save).
- DBD files (.txt), IMS Database Description files that describe the characteristics of an IMS database.
  If a corresponding .DM.txt file exists in the same directory, it is loaded too. **Note:** Columns in tables that are loaded from a DBD file do not have an attribute showing if they can be set to **null** or **not**. Instead they have an integer offset.
- GT Subset extract files (.ext) produced by the GT Subset application
  Note: Since GT Subset does not track all columns in tables or their types, the diagram that is produced is an incomplete reproduction of the original database. A prompt appears asking whether to save the current diagram before loading the new one (unless the current diagram is empty or no changes have been made since the last save). If Yes was selected, the new diagram is loaded when the old one is saved. Keyboard shortcut: *Ctrl+L*

SAVE

Save the current diagram in the GraphML format (.gml). If there are labels in the current diagram, another file is produced. That file is an XML file with the same name as the GraphML file but with an added *_labels* at the end. If there are no labels in the diagram, no such file is created. **Note**: Positions of objects and rules within the diagram are not saved. The next time a diagram is loaded it might (and most probably will) have slightly different positions for some of its objects. All other data that are associated with objects and rules (including colors), and the positions of labels, are saved. Keyboard shortcut: *Ctrl+S*.

PRINT

Print the current diagram to a printing device or PDF file. The printing is done with the diagram in a 1:1 zoom ratio, splitting it into as many pages as needed. Once the menu option is selected, a print preview window is shown, showing how the printed diagram looks. Keyboard shortcut: *Ctrl+P*.

EXPORT AS IMAGE

Convert the current diagram into an image file with one of four formats: PNG, JPEG, BMP, or GIF. One thing to note is that PNG and GIF images that are produced by exporting the diagram have a transparent background, while BMP and JPEG images have a black background.

EXPORT AS DDL

Convert the current diagram into a DDL file, containing SQL *CREATE TABLE* and *ALTER TABLE* statements, which describe all the Objects and Rules within the diagram. A dialog is shown asking where to save the DDL file.

EXIT

Exit the GTDiagrammer. A prompt appears asking whether to save the current diagram before exiting. If *Yes* was selected, the GTDiagrammer is shut down as soon as the diagram is saved.

View Menu

EXPAND ALL

Each object in the diagram consists of a table name and an expander containing the table columns. The expanders can be collapsed manually and individually at any time. However, the View Menu *Expand All* option also allows the user to expand or collapse all objects on the diagram. Keyboard shortcut: *Ctrl+E*.

Note: If all objects have been manually collapsed (but the *Expand All* menu option is still ticked) and *Expand All* is clicked, all objects are expanded (and the *Expand All* menu option stays ticked).

SHOW DETAILS

The columns in each object all have a name, a type, and an indicator which can be set to NULL or not (or an integer offset in case the diagram was loaded from a DBD file). By default, all but the name is hidden. However, the *Show Details* menu option allows the other column metadata to be shown. Once selected, all objects show the details of their columns. Keyboard shortcut: *Ctrl+D*.

ENABLE HIGHLIGHTING

As mentioned in the Highlighting and Selecting section, highlighting occurs when a rule, object, or a column within an object is hovered over by the mouse cursor. This behavior can be disabled from the View Menu *Enable Highlighting* option. Keyboard shortcut: *Ctrl+H*.

CHOOSE BACKGROUND COLOURS

Choose Background Colors allow the user to change the background colors of objects and/or rules.

Note: Using this method changes the color of all objects and/or rules on the diagram. *Restore Defaults* changes the colors back to their default values (which were set by right-clicking on the items or were loaded with the file, or if unset are Black for objects and Silver for rules).

CHOOSE HIGHLIGHT COLOURS

Choose Highlight Colors can change the colors that are used when highlighting (as discussed in the Highlight and Selection section). The default ones are Purple (for the hovered over item), Blue (for its children) and Brown (for its parents).

CHOOSE SELECTION COLOUR

Choose Selection Color can be used to change the color used for selection (as discussed in the Highlight and selection section). The default color is Yellow.

RELAYOUT DIAGRAM

Relayout Diagram is used to rearrange the objects in the diagram, while keeping the same layout algorithm. It can be useful after using *Expand All* or *Show Details*, because both of them could potentially leave the diagram clustered or scattered. However, objects will most likely have different positions after *Relayout Diagram* has been executed. More details about layout algorithms can be found in the Layout Menu section of this document. Keyboard shortcut: *Ctrl+R*.

ZOOM TO FIT

Zoom to Fit produces the same result as clicking the Fill button on the zoom control in GTDiagrammer – it changes the zoom ratio to fit the whole diagram on the window.

Layout Menu

The Grid-Tools ER Diagrammer can currently use nine different layout algorithms to display the diagram. The user can choose which one to use from the Layout Menu. The supported algorithms are the following:

- Fruchterman – Reingold (FR)
- Bounded FR
- Kamada – Kawai (KK)
- ISOM
- LinLog
- Simple Tree layout
- Simple Circle layout
- Efficient Sugiyama
- Compound graph layout (CompoundFDP) The default one is the Efficient Sugiyama algorithm. However, when the diagram has no rules (which could happen when a new diagram is created, when a diagram with no rules is loaded from a file or from a web service, or when the last remaining rule is deleted from the diagram) the only available option becomes the ISOM algorithm. As soon as a single rule is added to the diagram, all other options become available. FR and Bounded FR have certain conditions to be applicable, so they might not work in some cases.

Tools Menu

ADD NEW OBJECT

Add New Object allows the user to add a new object to the diagram. The new object table name, default color, and list of columns (primary keys and attributes) needs to be provided. Columns can be added by clicking the *Add Column* button. Once clicked, it opens a dialog asking for the new column name, data type and whether it can be set to NULL or not (or its offset if it is a DBD diagram). Data types are selected from a drop-down list. Some data types can have a precision, scale or length that is associated with them (Varchar can have length; Decimal can have precision and scale) If such a data type has been selected, the appropriate text boxes are enabled to allow the user to type an integer value; otherwise they are disabled. When all of the new column details have been filled-in, the column can be added as either a primary key or an attribute (non-primary-key). Primary keys and attributes that have been added can be removed by selecting them and clicking the appropriate *Remove* button. Clicking Confirm creates the new object with the provided data and adds it to the diagram. Keyboard shortcut: *Ctrl+Shift+O*.

ADD NEW RULE

Add New Rule allows the user to add a new rule between two existing objects. The tables and their columns involved in the new rule can be selected from drop-down lists. If the new rule involves multiple columns, the *Add more Columns to Rule* button can be used. This makes it possible to join more than one pair of columns from the same parent and child objects. Clicking *Confirm* creates the new rule. Keyboard shortcut: *Ctrl+Shift+R*.

Note: When a multi-column rule is being added clicking *Confirm* creates a rule containing the list of column joins in the *Rule so far* section AND the currently selected columns.

REMOVE UNRELATED OBJECTS

Remove Unrelated Objects removes all objects that have no rules that are associated with them (no relations to other objects) from the diagram. A dialog asking confirmation is shown before removing these objects.

SEARCH FOR TABLE NAME

Used to search through objects in the diagram (more specifically their table names). Every object whose table name contains the specified string in the search box (ignoring case) becomes selected. Keyboard shortcut: *Ctrl+F*.

Note: All previously selected items are unselected.

DIAGRAM OVERVIEW

The *Diagram Overview* is used to provide a view of where the user has currently zoomed to within the diagram. It shows the whole diagram in a small window and indicates where within it the main window of the GTDiagrammer has been zoomed to. It allows the currently seen data to be shown in relation to the rest of the data, especially useful with large diagrams. Keyboard shortcut: *Ctrl+O*.

CHOOSE LANGUAGE

The GTDiagammer has been localized to several languages and access to the localized versions is done by the *Choose Language* submenu in the *Tools* menu.

Note: Only the interface is translated, the table and column names will not change, nor will the Relationship Information for the Rules. Also, the text on buttons in dialogs (such as OK, Cancel, Save) seen when saving and loading files, printing, or error messages are not changed. They stay in the language of the Operating System installed.

Working with EDI Files Using the GT EDI Utility

This document explains how to use EDI formatted data in the Test Data Management suite. This is done through the functionality explained below for generating scripts and rules, converting to XML and importing/exporting EDI format.

Note: The following x12 EDI transactions are supported from the Insurance Industry. If the processing transaction is outside the list below, a *transaction not supported error* is displayed during the **convert to XML** stage.

270 - 5010X279

271 - 5010X279

276 - 5010X212

277 – 5010X212

820 – 5010X218

820 – 5010X306

834 – 5010X220

835 – 5010X221

837D – 5010X224

837I – 5010X223

837P – 5010X222

Generate Schema and Rules

The process to generate Schema and Rule files for a given EDI set e.g. 837P, 834 should be run only once.

Follow these steps:

1. Go to the first tab Generate Scripts and Rules.
2. Provide the SQL Server database name. Example - EDI.
3. Select the transaction specification for which Schema is required.
4. Click **Generate Scripts** to generate the scripts.
5. A generated Rules file is automatically moved to the rule file folder.

Once schema scripts have been generated, create an MS SQL Server database and execute DDL scripts to generate schema and tables. Do not execute the new primary key – foreign key DDL (example **820X218_DDL_pk_fk.sql),** because creation of Primary and Foreign keys slows down the import XML process.

Convert to XML

The Convert to XML process converts EDI files (*.txt and *.edi) to XML. Maximum file size limit is 60 MB that you can use to convert to XML.

Follow these steps to convert EDI files to XML:

1. Copy the EDI files to an input folder such as C:\Grid-Tools\GTEDI\input.
2. Open the second tab in In GTEDI.exe to convert to XML.
3. Populate the EDI message directory with the pathname specifying where the EDI files are stored, such as C:\Grid-Tools\GTEDI\input.
4. Click the **Convert** button.
5. Now the EDI files in the EDI message directory are converted to XML files and the EDI files.
   are moved to a subfolder named ProcessedEDI. These XML files can be imported to a database in the next step.

Importing EDI Files

Follow these steps:

1. Go to **Import XML** tab after running GTEDI.exe.
2. Input the folder path where XML messages are present such as C:\Grid-Tools\GTEDI\input.
3. Choose the drop down transaction specification of **to-be imported XMLs**.
4. Input the SQL Server Instance name – such as MYVAIOWIN81\EXP2014.
5. Input the SQL Server database name – such as EDI.
6. Do one of the following:
   a. Choose integrated security.
   b. Populate the username and password for SQL Server authentication.
7. Click **Test connection** to test the connection.
8. Click **Import.**

Successfully imported EDI XML are saved in a folder named InsertedXML in the XML Message directory. Example directory pathname: C:\Grid-Tools\GTEDI\input\InsertedXML

The inserted data can now be masked using FDM or manipulated using Datamaker.

Exporting EDI Files

This step reads EDI messages from the SQL Server database. It then converts them into EDI files with the desired file extension and EDI separators.

Follow these steps:

1. In GTEDI.exe, go to **Export EDI** tab**.**
2. Provide the folder path where XML messages are present such as C:\Grid-Tools\GTEDI\input.
3. Choose the drop down transaction type rule file for **to-be exported** EDI messages.
4. The imported EDI separators are used by default, however when using the **overwrite separators** check box, you can chose your own segments, elements, and line separators.
5. If you want to export all EDI messages then put **Export Bundle Id** as *. Otherwise give a specific valid bundle id from the ISA table of the given transaction table in the SQL Server database.
6. Do not change the output directory name, let it be the default. This directory is created relative to location of GTEDI.exe.
7. Populate the SQL Server Instance name – like MYVAIOWIN81\EXP2014.
8. Populate the SQL Server database name – like EDI.
9. Do one of the following:
   a. Choose integrated security.
   b. Populate username and password for SQL Server authentication.
10. Click **T est connection** to test the connection.
11. Click the **E xport** button.

Now the windows explorer is launched to show you the exported files. Exported files are by default stored in a folder such as C:\Grid- Tools\GTEDI\BuildOutput\[transaction_code]. For example if the transaction code is 837DX224, the folder would be C:\Grid- Tools\GTEDI\BuildOutput\837DX224.

Subset Production Data

Data subset is the process of creating a smaller referentially correct copy of a larger database. After subsetting, the cut-down database remains perfectly usable - the data is referentially correct and internally consistent - but total size is much more manageable.

CA TDM Data Subset uses native database utilities to ensure the highest possible performance when extracting small, more intelligent subsets from production. This allows you to quickly provide teams with more manageable sets of consistent, reverentially intact data for testing. It also minimises the risk of exposing sensitive records.

CA TDM Data Subset System Requirements

The CA TDM Data Subset component has the following system requirements. If you install the Data Subset with other components during the TDM installation, ensure that your system meets the requirements.

Operating Systems

Following operating systems are supported:

- Windows 7 Professional or higher
- Windows Server 2012 R2 Standard Edition or higher
- Windows Server 2012 R2 64-bit Standard Edition or higher

Java

CA TDM Data Subset requires a Java Runtime Environment (JRE 1.8 or higher).

Supported Data Sources

For the complete list of supported data sources (for example, database types, mainframe platforms, or file formats) for data subset, see Supported Data Sources.

Additionally, you can review the following points:

- Netezza supports only establishing a connection and saving an extract for data subset.

Installation Considerations

CA TDM Subset can live on the same system as other CA Test Data Manager components. If you installed CA TDM Datamaker, the Subset is already installed where you ran the GT Server.

Subset Stored Data

As a test data engineer, use the CA TDM Data Subset UI to access the larger data set stored in the data source and apply the subset rules to extract small, more intelligent data subsets from production.

Follow these Procedures:

- Establish Database Connection
- Create Extract Definitions
- (Optional) Prepare Subset Schema
- Generate Scripts
- Running Extracts and Imports
- Example: Create a Subset of Data Stored in Relational Database

Establish Database Connection

CA TDM uses the concept of connection profiles. A connection profile is a saved set of parameters that constitute a database connection. Once a profile is saved it may be selected from the Profile list in order to supply the connection parameters.

You can launch CA TDM Subset from CA TDM Datamaker and establish the database connection using the Datamaker connection profile. Also a stand alone CA TDM Subset executable file named GTSubset.exe is available that you can access from the default CA TDM Datamaker installation folder (C:\Program Files (x86)\Grid-Tools\GTDatamaker\).

Launch CA TDM Subset from CA TDM Datamaker

When you access CA TDM Data Subset from CA TDM Datamaker, it connects to the same database that you specified for the connection profile selected while launching the CA TDM Datamaker.

Follow these steps:

1. Launch the CA TDM Datamaker.
2. Go to Projects, Project Manager.
   Maintain Projects window opens.
3. Expand the project and select a version that you want to use for data subset.
4. Click Data Subset, Design Extracts and Transactions from the menu bar.
   CA TDM Subset is launched and the database connection is established successfully.

Launch CA TDM Subset using GTSubset Executable File

When you launch Data Subset using GTSubset.exe file for the first time, it has no saved profile information so you will need to set up a connection profile.

Follow these steps:

1. Run the GTSubset.exe file from the CA TDM installation folder. Typical path for CA TDM installation folder is *C:\Program Files (x86)\Grid-Tools\GTDatamaker*.

2. **GT Subset Professional Logon** window opens. Do the following:
   a. Enter values in the following fields:
      - GT Subset Profile Name
      - User Name
      - Password
      - DBMS
      - Server
      - Port
      - Default DB
      - Default Schema
   b. Click **Save** and click **OK** to confirm the action.Click **Connect**.

   Notes:
   a. The Oracle JDBC Thin Client driver is a 100% pure Java, Type IV driver. As it is written entirely in Java, this driver is platform-independent. It does not require any additional Oracle software on the client side.
   b. If you are connecting to Microsoft SQL Server, verify the following:
      - SQL Server port number (default is 1433, but this can be different). In the configuration manager, double-click on the TCP/IP protocol in the right pane.
      - Check that the port is not being blocked by your firewall.
      - Check that your browser service is running.
      - Check that the server name and instance are spelled correctly.
   c. If you are connecting to Sybase Adaptive Server, enter the Database, Host Name and Port Number (default is 5000). The Host Name and Port number should match the values returned by the Sybase Dsedit utility.
   d. If you are connecting to a MySQL database, enter the Database, Host Name and Port Number (the default is 3306).
   e. If you are connecting to DB2 and the java version is 1.8 or above, db2jcc.jar must be removed from the lib sub directory and db2jcc4.jar present. DB2 will default to the older db2jcc.jar if both are present, the older driver only works with Java 1.7 and below.
   f. If you are connecting to a DB2 database on a mainframe, ensure the appropriate connection software such as DB2 Connect or similar is installed, enter the Database Alias name and the Default Schema.
   g. Javelin supports Oracle, DB2, and Microsoft SQL Server. Subset also supports these data sources. Thus, Subset can generate Javelin workflows for Oracle, Microsoft SQL Server, and DB2.
   h. If you are connecting to a Teradata database, enter the Host Name and Default Schema (Database Name).
   i. If you are connecting to an Informix database, enter the Host Name and Default Schema (Database Name). The default port number is 9088.
3. **Repository Logon** window opens. Do the following:
   a. Enter values in the following fields:

      Profile
      Specifies the name of the Connection Profile you want to connect. The drop-down lists all the connection profiles which you have created using CA TDM Datamaker. Select the connection profile that connects to the CA TDM Respository.

      DBMS
      Specifies the database server where CA TDM Repository is installed. Following are the databases supported for repository:
      - MS SQL Server
      - Oracle

      For more information, see Supported Datasources.
   b. Based on the DBMS you selected some or many of the following options display. Enter values in all the displayed fields.

      Server

Specifies the name of the server where the DBMS is installed. Enter the server name.
Port No
Specifies the port number that listens to the DBMS. Enter the port number.
User Name
Specifies the user name that allows the connection to the CA TDM Repository.
Password
Specifies the password that authenticates the user to connect to the CA TDM Repository.
Default DB
Specifies the name of the database that represents the CA TDM Repository. For example, gtrep.
Default User
Specifies the default user name that connects to the CA TDM Repository. For example, dbo.

c. Click **Save** and click **OK** to confirm the action. Click **Connect**.

Create Extract Definitions

After establishing the database connection, you can select the data available in the database and generate a subset to get a smaller and referentially correct copy of larger database.

Follow these steps:

1. Launch CA TDM Subset using the connection profile that connects to the database you want to use to subset the data.
   CA TDM Subset application opens.
2. Select the **Project** and **Version** that you want to use to subset the data.
3. Select an appropriate schema from the **Select Schema** drop-down list.
4. Go to the **Select Table** drop-down list and select the driving table. Driving table is the table in the database, based on which you want to subset the data.
   Shows the relational tables in the data navigation tree in the left side pane. The data navigation tree includes all the relational tables as exists in the data source by default.
5. Review the table relationships and the child tables, to identify the tables that have the data you want in the subset. You can modify the table relationships to have the child tables related to the parent tables in the way that you require. For more information, see Modifying Table Relationships.
6. Select the driving table from the data navigation tree in the left side pane.
   When you select the driving table or any related table from the data navigation tree, a tab is opened for the respective table in the right side pane. Each tab has three sub-tabs to show the details of SQL, STATUS and RESULTS for the respective table.
   – **SQL** tab shows the default SQL query that runs when you select the table from the data navigation tree in the left side pane.
   – **STATUS** tab shows the information related the time taken to execute the SQL query and the number of rows returned.
   – **RESULTS** tab shows all the rows with data resulted by executing the default SQL query.
7. Select the driving table in the data navigation tree and review the SQL, STATUS and RESULTS in the right side pane under the respective table tab.
8. Under the **SQL tab,** modify the SQL query inserting the conditions you want to apply to filter the data and click **Execute SQL** button.
   The data is filtered based on the SQL query and returns the subset of data. Continue to modify the SQL query till you get the subset of data that fits for your purpose. For more information, see Modify the Driving Table SQL.

   NOTE

   Do not add comments in the SQL statements. The current version of CA TDM Subset does not support the SQL comments in the SQL window.
9. Select the driving table and click the **Preview Rowcounts** button.
   Preview Rowcounts window opens.

10. Click the **Run** button to retrieve the row counts for the extract.

    NOTE

    This will give the actual row counts your extract would produce assuming you are connected to the same connection at extract and design time.
    Data retrieved dynamically in the designer is restricted according to the row count size. This can be user defined and defaults to 40.
11. Go to the **File** menu, and click **Save Extract to Repository** or **Save Extract to File** as per your choice.
    a. Do the following to save the extract to repository:
    a. a. Click the Save Extract button.
       b. Click the browse button and select the directory.
       c. Specify the file name you wish to save the extract to.
       d. Add a description for the newly saved extract.
       e. Click the OK button.The extract is saved to the repository.
       f. (Optional) If you want to modify the extract that is saved to the repository, do the following:
    a. • • Click the Load Extract from Repository button.
       • Select Extract window opens to show the available extracts for the corresponding project version.
       • Select the extract from the list and click OK.
       • The selected extract is loaded.
    b. Do the following to save the extract to file:
    a. a. Click the Save Extract to Repository button.
       b. Connect to a Datamaker repository if not already connected.
       c. Specify the following in Save Extract to Repository dialog and click Save.
          • Extract Name
          • Description
          • Project
          • Version
          • Save Extract To
       d. (Optional) If you want to modify the extract that is saved to a file, do the following:
       a. • Click the Load Extract button.
          Windows explorer opens to show all the available extract files.
          • Select the Extract file you want load for the corresponding project version and click OK.

    The extract definition is saved successfully.

    NOTE

    1. In order to bring in relationships which have been previously removed from the extract, you will need to select a table and click the Refresh Children button.
    2. When a saved extract is re-loaded, Data Subset will attempt to query all nodes in the loaded tree. If a particular node returns no data, subsequent child nodes in that branch will not be queried.

Modifying Table Relationships

You can add the table relationships if the required child tables are not related to the parent in the way that you require. You can also remove the table relationships, if the unwanted child tables are related to the parent in the way that you do not require. If there are no table relationships defined in the data source, you can copy the table relationships from CA TDM Datamaker. Review the following procedures for different operations you can perform on table relationships.

Copy Relationships from Datamaker

Follow these steps:

1. Launch CA TDM Subset.
2. Select the project and version for which you want to get the table relationships.
3. Go to Utilities in the menu bar and click Copy Relationships.
   Copy Relationships from System window opens.
4. Verify the Project Name and Version are selected appropriately.
5. Select the Copy from GT Datamaker to GT Subset Professional radio button.
6. Click Get Relationships and click OK in the confirmation dialog.
   Relationships available in the Datamaker are shown in Available Relationships list box.
7. Select the relationships you want to copy to Subset and click the green down arrow button.
   The selected relationships are moved to Selected Relationships list box.
8. Review the relationships in Selected Relationships list box and ensure that all the relationships you want to copy from Datamaker to Subset are available in Selected Relationships list box.
   Note: Select a relationship and click down arrow or up arrow buttons to move the Relationships between Available Relationships and Selected Relationships.
9. Click Copy.
   The table relationships are successfully copied from TDM Datamaker to TDM Subset.

Edit User Defined Relationships

Follow these steps:

1. Launch CA TDM Subset.
2. Select the project and version for which you want to edit the table relationships.
3. Select a relationship in the Data Navigation tree.
4. Right click to open the context menu and click Edit Link.
   Edit Link window opens with Columns and SQL tabs.
5. Go to the Columns tab that includes the list boxes.
   The top list box contains the list of all columns related the parent and child tables. The lower list box contains the columns that join the two tables.
6. Do the following to edit the relationship:
   - If you want to edit the cardinality of the columns, select the respective columns from the lower list box and select the cardinality from the drop-down list. Click OK to save the changes.
   - If you want to define more columns to join the two tables, select the joining columns from the parent and child tables in the top list box. The selected columns are copied to the lower list box. Select the copied columns in the lower list box and select the cardinality to join them. Click OK to save the changes.
   - If you want to remove any of the joining columns, select the columns from the lower list box and press delete button on the key board. Click OK to save the changes.
7. Click OK to save the changes.
   You have successfully edited the table relationship for the joining columns.

Specify Additional Filter Conditions (SQL Clause) for Child Tables

After modifying the SQL to apply data filter conditions for parent tables (driving table), you can further apply the conditions for the child tables to filter the data based on the columns exist in the child tables.

Follow these steps:

1. Launch CA TDM Subset.

2. Select the project and version for which you want to edit the table relationships.
3. Select the child table for which you want to specify additional filter conditions in the Data Navigation tree.
4. Right click to open the context menu and click Edit Link.
5. Edit Link window opens with Columns and SQL tabs.
6. Go to the SQL tab that includes the SQL for corresponding parent and child tables. Left hand side pane includes the SQL for parent table (corresponding to the child table you selected) and the right hand side pane includes the SQL for child table you selected.
7. Go to the lower text boxes corresponding to the parent and child tables, and enter your SQL conditions which you want to apply to filter the data.
8. (Optional) Select the Join with SQL Only check-box to ignore the join columns specified under Columns tab and to use only the SQL specified under the SQL tab.
9. Click OK.
   The confirmation dialog opens.
10. Click OK and click the Save Extract button available below the Data Navigation tree.
    You have successfully applied the additional SQL conditions for the child table.
    The tables with additional SQL conditions, are marked with * (asterisk) next to the table name in the Data Navigation tree.

Delete User Defined Relationships

This option removes the relationship definition from file and also from the Data Navigation tree. Relationships based on foreign key constraints cannot be deleted.

Follow these steps:

1. Launch CA TDM Subset.
2. Select the project and version to see the existing table relationships.
3. Select the relationship that you want to delete in the Data Navigation tree.
4. Right click to open the context menu and click Delete Link Permanently.
5. Click Yes in the confirmation dialog.
   You have successfully deleted the table relationship.

Remove Links from Extract

This option removes the link from the Data Navigation tree, but it will remain in the extract. You can refresh the tree to recall the removed links, if required.

Follow these steps:

1. Launch CA TDM Subset.
2. Select the project and version to see the existing table relationships.
3. Select a table in the Data Navigation tree for which you want to remove the links.
4. Right click to open the context menu and click Remove Link from Extract.
5. Click Yes in the confirmation dialog.
   You have successfully removed the link from the extract.

Refresh Links

This options refreshes and shows all the removed links from the extract. If you have removed a link while exploring or you have imported an existing extract definition created earlier you may wish to refresh the link.

Follow these steps:

1. Launch CA TDM Subset.

2. Select the project and version to see the existing table relationships.
3. Select a table in the Data Navigation tree for which you want to refresh the links.
4. Right click to open the context menu and click Refresh Children. Alternatively you can also click the Refresh Children button at the bottom of the Data Navigation tree.
   You have successfully refreshed the links for the selected table.

(Optional) Prepare Subset Schema

After creating the extract definitions, you can tag tables in order to simplify the data design process and generate scripts for subset schema.

Follow these steps:

1. Launch **Datamaker**, go to **Data Subset** menu and click **Verify and Prepare Subset Schema**.
   Verify and Prepare Subset Schema window opens.
2. Select the appropriate schema from the **Get Tables For Schema** drop-down list and click the **Go** (green arrow) button.
   Shows the schema tables with their respective row counts. You can set the row count limit to small and large to include the tables that have number of rows smaller or larger than the set limit.
3. Enter the value in **Small Limit** field and click Set.
   Selects tables that have the row count less than the specified value.
4. Enter the value in **Large Limit** field and click Set.
   Selects tables that have the row count more than the specified value.
5. Identify the potential reference tables and select the **Reference** check box in the row corresponding to the identified tables.
6. Identify the tables that you want to ignore, and select the **Ignore** check box in the row corresponding to the identified tables.
   Ignore tables will not be visible in Data Subset, if you connect to a Datamaker repository.
7. Choose one of the below under the **Extract** options:
   a. **Repository**
      Select this radio button, if the stored schema is available in the repository.
      a. Click the **Open Extract** button.
      b. The Choose Extract window opens.
      c. Choose the extract from the list of available extracts.
      d. Click OK.
   b. **File**
      Select this radio button, if the stored schema is available in the form of an extract file (.ext file format).
      a. Click the **Open Extract** button.
      b. the Select Extract File window opens.
      c. Browse the windows explorer and select the extract file.
      d. Click OK.

   Identifies the tables in the schema that are included in the extract.
8. Click **Update Tags**
   Updates the tag information for all the tables in your schema.
9. Click **Verify**.
   Perform a series of checks against your schema in order to see if there are any potential foreign key violations based on your table tags.
10. Review the information in the Validate Schema dialog and click OK.
11. Click the **Generate** button.
    Generates extract scripts for the tables marked as Small or Reference and a set of foreign key enable and disable statements for the schema.
12. Review the information in the **Generate Scripts** confirmation dialog, and click **OK**.

You have successfully generated the Subset Schema.

Generate Scripts

You can generate scripts to move the data depending on the RDBMS you are connected to. Database Actions screen consists of various tabs where script options can be set, dependent on the type of script generation chosen.

Masking scripts for MS SQL Server that are generated from GTSubset now directly join onto the seed data table. You must upgrade the gtsrc_reference_data table in the scramble database. To upgrade the *gtsrc_reference_data table* in the scramble database, drop and recreate the scramble database using the installer provided with the latest release.

NOTE

If you do not upgrade the MS SQL Server scramble database, the new masking scripts generated from GTSubset for MS SQL Server masking do not function.

1. Launch Data Subset.
2. Go to File menu and click Build Database Actions.
   The Database Actions window opens.
3. Go to database actions drop-down (next to Execute SQL icon below the menu bar) and select appropriate action based on the database you are using. For example, if you are using MS SQL Server, select Build MS SQL Server Export/Import.
4. Go to Extract Details tab and enter the Action Name.
5. The action name you enter will be the suffix for the script file names.
6. **(MS SQL Server only)** Specify the following, if you are connected to MS SQL Server:
   - **Functions Database**
     Specifies the name of the database where the scramble components are installed. This is applicable, if you are generating scramble scripts.
   - **Create Table As Select**Select this check box for insert scripts only.
   - **Restrict Driving Table Data**
     Specify only when the driving table has no where clause.
   - **Persist views in functions database**Select this check box to create a BCP masked export for a masked view in the database where scramble components are or in the source database.
   - **Target Schema**Specifies the target schema for insert scripts only.
   - **Source Database**
     Specifies the database from which the data is extracted. Select the database name from the drop-down list.
   - **Target Database**
     Specifies the database from which the data is loaded into. Select the database name from the drop-down list.
   - **Use Link to reference Source tables**
     Select this check box if source tables and target tables are on different servers for insert scripts only.
7. **(Teradata only)** Specify the following, if you are connected Teradata:
   - **Functions Database**
     Specifies the name of the database where the scramble C UDF functions are installed. This is applicable, if you are generating scramble scripts. Select the database name from the drop-down list.
   - **Export Log Database**
     Specifies the database where the export log tables are created. Typically this is the database where the data is extracted from. Select the database name from the drop-down list.
   - **Load Log Database**
     Specifies the database where the load log tables are created. Typically this is the target database where the data will be loaded into. Select the database name from the drop-down list.
   - **Source Database**

Specifies the database from which the data is extracted. Select the database name or the environment variable (name of the variable enclosed in %) from the drop-down list.

- **Target Database**
  Specifies the database from which the data is loaded into. Select the database name or the environment variable (name of the variable enclosed in %) from the drop-down list.
- **Stage Source Database**
  Specifies the database for the stage insert scripts from which the data is selected. Select the database name or the environment variable (name of the variable enclosed in %) from the drop-down list.
- **Stage Target Database**
  Specifies the database for the stage insert scripts into which the data is inserted. Select the database name or the environment variable (name of the variable enclosed in %) from the drop-down list.
- **Log Directory**
  Specifies the directory path where the log files are saved. This is applicable for Windows scripts only. Select the Windows environment variable (name of the variable enclosed in %) from the drop-down list. You can also click the browse button and select a directory path.
- **Extract file directory**
  Specifies the directory path where the fast export data files are saved, and the load scripts are loaded from. Select the Windows environment variable (name of the variable enclosed in %) from the drop-down list. You can also click the browse button and select a directory path.

8. Go to the Extract Tables tab and select the whether the extract definition is available in the repository or file:
   a. **Repository**
      a. Select this radio button, if the saved extract is available in the repository.
         Lists all the available extracts.
      b. Identify the extract from which you want to generate the scripts and select the corresponding check box. Select all the extracts that you want to use to form the subset.
         Adds the tables of the selected extracts to the appropriate list boxes at the bottom of the screen.
   b. **File**
      Select this radio button, if the saved extract is available in the form of an extract file (.ext file format).
      a. Click the **Open Extract Directory** button.
         The **Browse Directory** window opens.
      b. Browse the windows explorer, select the **extract file** and click **OK**.
         Adds the selected file to the extracts list. Add as many files as you want to use to form the subset.
      c. Select the check box corresponding to each extract file which all you want to use to form the subset.
         Adds the tables from the selected extract files to the appropriate list boxes at the bottom of the screen.
9. Review the tables added to Subset Tables and No Data Tables list box and ensure that all the tables which you want to use in subset are available.
   The Subset Tables list box includes the tables that are defined in the saved extract and has some data. The No Data Tables list box includes the tables which will not have any data in the resulting subset.
   Notes:
   a. Select the Use Connection Tables Only check box, if you wish to choose all the tables only from the Subset Tables list. This will move all the tables in the ‘Subset Tables’ list box to ‘All Data Tables’ list. You can then move tables between the All Data Tables list and the No Data Tables list to define which tables will have their data extracted. If you choose the Connection Tables Only option, then all the data will be extracted from those tables in the ‘All Data Tables’ list. This option is most commonly used when data needs to be scrambled for a schema. Tables in the All Data Tables list will have all their data exported. Moving a table name from this list to the No Data Tables list will mean the resulting subset will contain no data for this table.
   b. To have the relationally intact data in the subset, you can remove tables completely from the extract. To remove a table from All Data Tables list or No Data Tables list, select a table and press the delete key on the key board.
   c. Oracle only:
      - For Oracle exports, to exclude No Data Tables from creating data exports select the Include No Rows check-box.

d. Teradata Only
   - For Teradata Windows Exports, you can use the Bind variables when defining extracts against Teradata. The binds are added to the driving table SQL. For example, "*select * from orders.PERSONS where person_id < :id AND FIRST_NAME = :NAME".* To resolve the variables at run time, you must create environment variables with the same name as the bind. Non numeric values must be enclosed in quotes. For example, 'David'.

10. **(Optional)** To define the referential order for the tables in the extract scripts, do the following:
    a. Go to the Table Order tab.
    b. Select the Table Order check box.
       Displays the calculated table order for all tables in the schema.
11. **(Optional) (Teradata only)** Go to the **Extra Scripts** tab and select the following options as necessary:
    a. **Drop/Create Indexes**
       Creates SQL scripts, to drop and recreate the indexes and the primary keys. These scripts are called by the relevant load scripts as Teradata does not support the loads into the tables with secondary indexes.
    b. **Disable/Enable or Drop/Create FKs**
       Select this option to drop the foreign key constraints prior to the data load. The dropped foreign key constraints are recreated after completing the load.
12. Click the Generate button and click OK in Create Export dialog.
    Creates the appropriate export and import scripts to populate the subset schema.
    Notes:
    a. The generated scripts are saved to the same directory from which the extract files are selected using Open Extract Directory button.
    b. When importing into MS SQL Server, it is a prerequisite that the tables to be imported are empty. CA TDM Subset archive produces a historySchema.bat script file in order to create the empty tables.
    c. The extract scripts can be launched from a command prompt (Windows) a terminal window (UNIX) or using z/OS JCL. The corresponding import script is created, so that once the export script is run, the resulting export files may be imported into another database.

(Oracle only) Reformat Tables

For the action type 'Build Oracle Windows Extract / Loader Import' and 'Build Oracle UNIX Extract / Loader Import', you can reformat the SQL Loader generated scripts. This reformat is generally used when there are changes in the versions of subset schema. You must have maintained these versions of schema registered for different versions in the Datamaker. Do the following to reformat the SQL Loader generated scripts:

1. Launch Data Subset.
2. Go to File menu and click the **Build Database Actions**.
3. Go to **Reformat Tables** tab in the Database Actions window.
4. Select your **Extract Version** and **Load Version**.
   a. Click the **Show Differences** button.
      Reflects the differences found between the versions in the SQL Loader control files generated.
      If extra NOT NULL columns exist in tables in the Load version as compared to the Extract Version, adds a default value as specified in Datamaker. If the default value is not specified in the Datamaker, uses the below values:
      - System date for Date fields
      - A space for character fields
      - A '0' for numeric fields
   b. The tables which exists in the Extract Version but not in the Load Version, are ignored in the load process.
   c. The table columns which exists in the Extract Version but not the Load Version, are ignored in the load process.

Samples of Script Files

Batch Files for Extracts on Windows

Oracle

```
@echo off
if "%1"=="" goto error
del xxx_*.dmp
del xxx_*.par
echo tables=USER01.ITEM >> xxx_full.par
echo tables=USER01.PRICE >> xxx_full.par
echo tables=USER01.PRODUCT >> xxx_full.par
exp system/%1 statistics=none constraints=N rows=Y DIRECT=Y file=xxx_full.dmp
 parfile=xxx_full.par
exp system/%1 statistics=none constraints=Y rows=N file=xxx_norows.dmp owner=USER01
exp system/%1 statistics=none constraints=N rows=Y file=xxx_CUSTOMER.dmp
 tables=USER01.CUSTOMER query=\"where ( STATE = 'MN') OR ((CUSTID) in (select (CUSTID)
 from USER01.LEDGER where custid = 104))\" > xxx_CUSTOMER.out
exp system/%1 statistics=none constraints=N rows=Y file=xxx_ORD.dmp tables=USER01.ORD
 query=\"where ((CUSTID) in (select (CUSTID) from USER01.CUSTOMER where STATE = 'MN'))\"
 > xxx_ORD.out
exp system/%1 statistics=none constraints=N rows=Y
 file=xxx_ORD_PROCESS_LEDGER_DETAILS.dmp tables=USER01.ORD_PROCESS_LEDGER_DETAILS query=
\"where ((CUSTID) in (select (CUSTID) from USER01.CUSTOMER where STATE = 'MN'))\" >
 xxx_ORD_PROCESS_LEDGER_DETAILS.out
exp system/%1 statistics=none constraints=N rows=Y
 file=xxx_ORD_PROCESS_LEDGER_DETAILS_ALL.dmp
 tables=USER01.ORD_PROCESS_LEDGER_DETAILS_ALL query=\"where ((CUST_ID)
 in (select (CUSTID) from USER01.CUSTOMER where STATE = 'MN'))\" >
 xxx_ORD_PROCESS_LEDGER_DETAILS_ALL.out
exp system/%1 statistics=none constraints=N rows=Y file=xxx_LEDGER.dmp
 tables=USER01.LEDGER query=\"where ( custid = 104)\" > xxx_LEDGER.out
goto end
:error
echo.
echo ** ERROR SPECIFYING PARAMETERS
echo.
echo ** TO RUN: xxx_export system_password@tnsname
echo.
:end
```

MySQL

```
@echo off
```

```
if "%1"=="" goto error
if "%2"=="" goto error
if "%3"=="" goto error
del xxx_*.sql
echo Full Tables
echo GridTools.product_BDB
mysqldump --user=%1 --password=%2 --host=%3 --single-transaction --extended-insert
 --disable-keys --skip-add-drop-table %4 --result-file=xxx_GridTools_product_BDB.sql
 GridTools product_BDB
echo GridTools.product_csv
mysqldump --user=%1 --password=%2 --host=%3 --single-transaction --extended-insert
 --disable-keys --skip-add-drop-table %4 --result-file=xxx_GridTools_product_csv.sql
 GridTools product_csv
echo No Data Tables
echo GridTools.bonus
mysqldump --user=%1 --password=%2 --host=%3 --no-data --disable-keys --skip-add-drop-
table %4 --result-file=xxx_GridTools_bonus.sql GridTools bonus
echo GridTools.customer_address
mysqldump --user=%1 --password=%2 --host=%3 --no-data --disable-keys --skip-add-drop-
table %4 --result-file=xxx_GridTools_customer_address.sql GridTools customer_address
echo Subset Tables
echo GridTools.customer
mysqldump --user=%1 --password=%2 --host=%3 --single-transaction --extended-insert
 --disable-keys --skip-add-drop-table %4 --result-file=xxx_GridTools_customer.sql --
where="( STATE = 'MN')" GridTools customer
echo GridTools.ord
mysqldump --user=%1 --password=%2 --host=%3 --single-transaction --extended-insert
 --disable-keys --skip-add-drop-table %4 --result-file=xxx_GridTools_ord.sql --
where="EXISTS (SELECT 1 FROM GridTools.customer WHERE custid = GridTools.ord.custid AND
 STATE = 'MN')" GridTools ord
echo Done
goto end
:error
echo.
echo ** ERROR SPECIFYING PARAMETERS
echo.
echo ** TO RUN: xxx username password hostname [extra mysqldump options]
echo.
:end
```

SQL Server/Sybase

```
@echo off
if "%1"=="" goto error
if "%2"=="" goto error
if "%3"=="" goto error
```

```
del abc_*.sql
echo Full Tables
echo marin.dbo.BONUS
bcp "marin.dbo.BONUS" out "BONUS.dmp" -U"%1" -P"%2" -S"%3" -n -k
echo marin.dbo.COMMPLAN_CODES
bcp "marin.dbo.COMMPLAN_CODES" out "COMMPLAN_CODES.dmp" -U"%1" -P"%2" -S"%3" -n -k
echo marin.dbo.CUSTOMER_ADDRESS
bcp "marin.dbo.CUSTOMER_ADDRESS" out "CUSTOMER_ADDRESS.dmp" -U"%1" -P"%2" -S"%3" -n -k
echo Subset Tables
echo marin.dbo.CUSTOMER
bcp "SELECT * from marin.dbo.CUSTOMER where ( state = 'MN')" queryout CUSTOMER.dmp -
U"%1" -P"%2" -S"%3" -n -k
echo marin.dbo.ORD
bcp "SELECT * from marin.dbo.ORD where exists (SELECT 1 FROM marin.dbo.CUSTOMER WHERE
 CUSTID = marin.dbo.ORD.CUSTID AND state = 'MN')" queryout ORD.dmp -U"%1" -P"%2" -S"%3"
 -n -k
echo marin.dbo.ITEM
bcp "SELECT * from marin.dbo.ITEM where exists (SELECT 1 FROM marin.dbo.ORD WHERE ORDID
 = marin.dbo.ITEM.ORDID AND EXISTS(SELECT 1 FROM marin.dbo.CUSTOMER WHERE CUSTID =
 marin.dbo.ORD.CUSTID AND state = 'MN'))" queryout ITEM.dmp -U"%1" -P"%2" -S"%3" -n -k
echo marin.dbo.PRODUCT
bcp "SELECT * from marin.dbo.PRODUCT where exists (SELECT 1 FROM marin.dbo.ITEM WHERE
 PRODID = marin.dbo.PRODUCT.PRODID AND EXISTS(SELECT 1 FROM marin.dbo.ORD WHERE ORDID
 = marin.dbo.ITEM.ORDID AND EXISTS(SELECT 1 FROM marin.dbo.CUSTOMER WHERE CUSTID =
 marin.dbo.ORD.CUSTID AND state = 'MN')))" queryout PRODUCT.dmp -U"%1" -P"%2" -S"%3" -n
 -k
echo marin.dbo.PRICE
bcp "SELECT * from marin.dbo.PRICE where exists (SELECT 1 FROM marin.dbo.PRODUCT
 WHERE PRODID = marin.dbo.PRICE.PRODID AND EXISTS(SELECT 1 FROM marin.dbo.ITEM WHERE
 PRODID = marin.dbo.PRODUCT.PRODID AND EXISTS(SELECT 1 FROM marin.dbo.ORD WHERE ORDID
 = marin.dbo.ITEM.ORDID AND EXISTS(SELECT 1 FROM marin.dbo.CUSTOMER WHERE CUSTID =
 marin.dbo.ORD.CUSTID AND state = 'MN'))))" queryout PRICE.dmp -U"%1" -P"%2" -S"%3" -n -
k
echo marin.dbo.ORD_PROC_LED_DETS
bcp "SELECT * from marin.dbo.ORD_PROC_LED_DETS where exists (SELECT 1 FROM
 marin.dbo.CUSTOMER WHERE CUSTID = marin.dbo.ORD_PROC_LED_DETS.CUSTID AND state = 'MN')"
 queryout ORD_PROC_LED_DETS.dmp -U"%1" -P"%2" -S"%3" -n -k
echo marin.dbo.ORD_PROC_LED_DETS_ALT
bcp "SELECT * from marin.dbo.ORD_PROC_LED_DETS_ALT where exists (SELECT 1 FROM
 marin.dbo.CUSTOMER WHERE CUSTID = marin.dbo.ORD_PROC_LED_DETS_ALT.CUSTID AND state =
 'MN')" queryout ORD_PROC_LED_DETS_ALT.dmp -U"%1" -P"%2" -S"%3" -n -k
echo Done
goto end
:error
echo.
echo ** ERROR SPECIFYING PARAMETERS
```

```
echo.
echo ** TO RUN: abc username password server
echo.
:end
```

DB2

```
@echo off
if "%1"=="" goto error
if "%2"=="" goto error
if "%3"=="" goto error
del user01_*.IXF
del user01_*.log
db2cmd user01_export_ctl.bat %1 %2 %3
goto end
:error
echo.
echo ** ERROR SPECIFYING PARAMETERS
echo.
echo ** TO RUN: user01_export [DB Connect database name] [username] [password]
echo.
:end
```

Batch Files for Imports on Windows

Oracle

```
@echo off
if "%1"=="" goto error
if "%2"=="" goto set1
set targetuser=%2
goto go1
:set1
set targetuser=USER01
:go1
imp system/%1 ignore=Y rows=Y fromuser=USER01 touser=%targetuser% file=xxx_CUSTOMER.dmp
imp system/%1 ignore=Y rows=Y fromuser=USER01 touser=%targetuser% file=xxx_ORD.dmp
imp system/%1 ignore=Y rows=Y fromuser=USER01 touser=%targetuser%
 file=xxx_ORD_PROCESS_LEDGER_DETAILS.dmp
imp system/%1 ignore=Y rows=Y fromuser=USER01 touser=%targetuser%
 file=xxx_ORD_PROCESS_LEDGER_DETAILS_ALL.dmp
imp system/%1 ignore=Y rows=Y fromuser=USER01 touser=%targetuser% file=xxx_LEDGER.dmp
imp system/%1 ignore=Y rows=Y constraints=N fromuser=USER01 touser=%targetuser%
 file=xxx_full.dmp
```

```
imp system/%1 ignore=Y rows=N constraints=Y fromuser=USER01 touser=%targetuser%
 file=xxx_norows.dmp
goto end
:error
echo ** ERROR SPECIFYING PARAMETERS
echo.
echo ** TO RUN: xxx_import target_system_password@tnsname target_user
echo.
:end
if exist %gtfiles%\analyse.pls sqlplus system/%1 @%gtfiles%\analyse.pls %targetuser%
```

MySQL

```
@echo off
if "%1"=="" goto error
if "%2"=="" goto error
if "%3"=="" goto error
if "%4"=="" goto error
echo Full Tables
echo %4.product_BDB
mysql --user=%1 --password=%2 --host=%3 --database=%4 < xxx_GridTools_product_BDB.sql
echo %4.product_csv
mysql --user=%1 --password=%2 --host=%3 --database=%4 < xxx_GridTools_product_csv.sql
echo %4.product_isam
echo No Data Tables
echo %4.bonus
mysql --user=%1 --password=%2 --host=%3 --database=%4 < xxx_GridTools_bonus.sql
echo %4.customer_address
mysql --user=%1 --password=%2 --host=%3 --database=%4 <
 xxx_GridTools_customer_address.sql
echo Subset Tables
echo %4.customer
mysql --user=%1 --password=%2 --host=%3 --database=%4 < xxx_GridTools_customer.sql
echo %4.ord
Data Subset™ User Guide | 157
mysql --user=%1 --password=%2 --host=%3 --database=%4 < xxx_GridTools_ord.sql
goto end
:error
echo ** ERROR SPECIFYING PARAMETERS
echo.
echo ** TO RUN: xxx username password hostname database
echo.
:end
```

MS SQL Server/Sybase

```
@echo off
if "%1"=="" goto error
if "%2"=="" goto error
if "%3"=="" goto error
if "%4"=="" goto error
set GTOWNER=dbo
echo Full Tables
echo %4.BONUS
bcp "%4.%GTOWNER%.BONUS" in "BONUS.dmp" -U"%1" -P"%2" -S"%3" -n -k
echo %4.COMMPLAN_CODES
bcp "%4.%GTOWNER%.COMMPLAN_CODES" in "COMMPLAN_CODES.dmp" -U"%1" -P"%2" -S"%3" -n -k
echo %4.CUSTOMER_ADDRESS
bcp "%4.%GTOWNER%.CUSTOMER_ADDRESS" in "CUSTOMER_ADDRESS.dmp" -U"%1" -P"%2" -S"%3" -n -k
echo Subset Tables
echo %4.CUSTOMER
bcp "%4.%GTOWNER%.CUSTOMER" in "CUSTOMER.dmp" -U"%1" -P"%2" -S"%3" -n -k
echo %4.ORD
bcp "%4.%GTOWNER%.ORD" in "ORD.dmp" -U"%1" -P"%2" -S"%3" -n -k
echo %4.ITEM
bcp "%4.%GTOWNER%.ITEM" in "ITEM.dmp" -U"%1" -P"%2" -S"%3" -n -k
echo %4.PRODUCT
bcp "%4.%GTOWNER%.PRODUCT" in "PRODUCT.dmp" -U"%1" -P"%2" -S"%3" -n -k
echo %4.PRICE
Data Subset™ User Guide | 167
bcp "%4.%GTOWNER%.PRICE" in "PRICE.dmp" -U"%1" -P"%2" -S"%3" -n -k
echo %4.ORD_PROC_LED_DETS
bcp "%4.%GTOWNER%.ORD_PROC_LED_DETS" in "ORD_PROC_LED_DETS.dmp" -U"%1" -P"%2" -S"%3" -n
 -k
echo %4.ORD_PROC_LED_DETS_ALT
bcp "%4.%GTOWNER%.ORD_PROC_LED_DETS_ALT" in "ORD_PROC_LED_DETS_ALT.dmp" -U"%1" -P"%2" -
S"%3" -n -k
goto end
:error
echo ** ERROR SPECIFYING PARAMETERS
echo.
echo ** TO RUN: abc username password server target_database
echo.
:end
```

DB2

```
@echo off
if "%1"=="" goto error
if "%2"=="" goto error
```

```
if "%3"=="" goto error
if "%4"=="" goto error
db2cmd user01_import_ctl.bat %1 %2 %3 %4
goto end
:error
echo.
echo ** ERROR SPECIFYING PARAMETERS
echo.
echo ** TO RUN: user01_import [DB Connect database name] [import_schema] [username]
 [password]
echo.
:end
```

Shell Scripts for Extracts on Linux/Unix

Oracle

```
if [ "$1" = "" ] ; then
echo
echo \#\# ERROR SPECIFYING PARAMETERS
echo
echo \#\# TO RUN: yyy_export system_password[@tnsname] [working directory]
echo
exit -1
fi
# Change to the working directory if given one that is valid!
if [ ! "$2" = "" ] ; then
if [ -d $2 ] ; then
cd $2
fi
fi
rm -f yyy_*.dmp
rm -f yyy_*.par
echo TABLES=USER01.ITEM >> yyy_full.par
echo TABLES=USER01.PRICE >> yyy_full.par
echo TABLES=USER01.PRODUCT >> yyy_full.par
exp system/$1 statistics=none constraints=N DIRECT=Y rows=Y parfile=yyy_full.par
 file=yyy_full.dmp > yyy_full.out 2>&1 &
exp system/$1 statistics=none constraints=Y rows=N owner=USER01 file=yyy_norows.dmp >
 yyy_norows.out 2>&1 &
exp system/$1 statistics=none constraints=N rows=Y file=yyy_CUSTOMER.dmp
 tables=USER01.CUSTOMER query=\"where \( STATE = \'MN\'\) OR \(\(CUSTID\) in \(select
 \(CUSTID\) from USER01.LEDGER where custid = 104\)\)\" > yyy_CUSTOMER.out 2>&1 &
```

```
exp system/$1 statistics=none constraints=N rows=Y file=yyy_ORD.dmp tables=USER01.ORD
 query=\"where \(\(CUSTID\) in \(select \(CUSTID\) from USER01.CUSTOMER where STATE =
 \'MN\'\)\)\" > yyy_ORD.out 2>&1 &
exp system/$1 statistics=none constraints=N rows=Y
 file=yyy_ORD_PROCESS_LEDGER_DETAILS.dmp tables=USER01.ORD_PROCESS_LEDGER_DETAILS query=
\"where \(\(CUSTID\) in \(select \(CUSTID\) from USER01.CUSTOMER where STATE = \'MN
\'\)\)\" > yyy_ORD_PROCESS_LEDGER_DETAILS.out 2>&1 &
exp system/$1 statistics=none constraints=N rows=Y
 file=yyy_ORD_PROCESS_LEDGER_DETAILS_ALL.dmp
 tables=USER01.ORD_PROCESS_LEDGER_DETAILS_ALL query=\"where \(\(CUST_ID\)
 in \(select \(CUSTID\) from USER01.CUSTOMER where STATE = \'MN\'\)\)\" >
 yyy_ORD_PROCESS_LEDGER_DETAILS_ALL.out 2>&1 &
exp system/$1 statistics=none constraints=N rows=Y file=yyy_LEDGER.dmp
 tables=USER01.LEDGER query=\"where \( custid = 104\)\" > yyy_LEDGER.out 2>&1 &
wait
cat yyy_full.out
cat yyy_norows.out
cat yyy_CUSTOMER.out
cat yyy_ORD.out
cat yyy_ORD_PROCESS_LEDGER_DETAILS.out
cat yyy_ORD_PROCESS_LEDGER_DETAILS_ALL.out
cat yyy_LEDGER.out
echo Extract complete
```

MySQL

```
if [ "$1" = "" -o "$2" = "" -o "$3" = "" ] ; then
echo
echo \#\# ERROR SPECIFYING PARAMETERS
echo
echo \#\# TO RUN: yyy_export.sh username password hostname [working directory & extra
 mysqldump options]
echo
exit -1
fi
# Change to the working directory if given one that is valid!
if [ ! "$4" = "" ] ; then
if [ -d $4 ] ; then
cd $4
fi
fi
rm -f yyy_*.sql
echo Full Tables
echo GridTools.customer_type
```

```
mysqldump --user=$1 --password=$2 --host=$3 --single-transaction --extended-insert --
disable-keys --skip-add-drop-table $5 --result-file=yyy_GridTools_customer_type.sql
 GridTools customer_type > yyy_GridTools_customer_type.out 2>&1 &
echo GridTools.dept
mysqldump --user=$1 --password=$2 --host=$3 --single-transaction --extended-insert --
disable-keys --skip-add-drop-table $5 --result-file=yyy_GridTools_dept.sql GridTools
 dept > yyy_GridTools_dept.out 2>&1 &
echo No Data Tables
echo GridTools.Test1
mysqldump --user=$1 --password=$2 --host=$3 --no-data --disable-keys --skip-add-drop-
table $5 --result-file=yyy_GridTools_Test1.sql GridTools Test1 > yyy_GridTools_Test1.out
 2>&1 &
echo GridTools.Test2
mysqldump --user=$1 --password=$2 --host=$3 --no-data --disable-keys --skip-add-drop-
table $5 --result-file=yyy_GridTools_Test2.sql GridTools Test2 > yyy_GridTools_Test2.out
 2>&1 &
echo Subset Tables
echo GridTools.customer
mysqldump --user=$1 --password=$2 --host=$3 --single-transaction --extended-insert
 --disable-keys --skip-add-drop-table $5 --result-file=yyy_GridTools_customer.sql --
where=" ( STATE = 'MN')" GridTools customer > yyy_GridTools_customer.out 2>&1 &
echo GridTools.ord
mysqldump --user=$1 --password=$2 --host=$3 --single-transaction --extended-insert
 --disable-keys --skip-add-drop-table $5 --result-file=yyy_GridTools_ord.sql --
where="EXISTS (SELECT 1 FROM GridTools.customer WHERE custid = GridTools.ord.custid AND
 STATE = 'MN')" GridTools ord > yyy_GridTools_ord.out 2>&1 &
echo Working...
wait
cat yyy_GridTools_customer_type.out
cat yyy_GridTools_dept.out
cat yyy_GridTools_customer.out
cat yyy_GridTools_ord.out
cat yyy_GridTools_customer.out
cat yyy_GridTools_ord.out
echo Extract complete
```

SQL Server/Sybase

```
if [ "$1" = "" -o "$2" = "" -o "$3" = "" ] ; then
echo
echo \#\# ERROR SPECIFYING PARAMETERS
echo
echo \#\# TO RUN: yyy_export.sh username password server [working directory]
echo
exit -1
fi
```

```
# Change to the working directory if given one that is valid!
if [ ! "$4" = "" ] ; then
if [ -d $4 ] ; then
cd $4
fi
fi
rm -f yyy_*.dmp
echo Full Tables
echo marin.dbo.BONUS
bcp "marin.dbo.BONUS" out "BONUS.dmp" -U"$1" -P"$2" -S"$3" -n -k >
 yyy_marin_dbo_BONUS.out 2>&1 &
echo marin.dbo.COMMPLAN_CODES
bcp "marin.dbo.COMMPLAN_CODES" out "COMMPLAN_CODES.dmp" -U"$1" -P"$2" -S"$3" -n -k >
 yyy_marin_dbo_COMMPLAN_CODES.out 2>&1 &
echo marin.dbo.CUSTOMER_ADDRESS
bcp "marin.dbo.CUSTOMER_ADDRESS" out "CUSTOMER_ADDRESS.dmp" -U"$1" -P"$2" -S"$3" -n -k >
 yyy_marin_dbo_CUSTOMER_ADDRESS.out 2>&1 &
echo Subset Tables
echo marin.dbo.CUSTOMER
bcp "SELECT * from marin.dbo.CUSTOMER where ( state = 'MN')" queryout CUSTOMER.dmp -
U"$1" -P"$2" -S"$3" -n -k > yyy_marin_dbo_CUSTOMER.out 2>&1 &
echo marin.dbo.ORD
bcp "SELECT * from marin.dbo.ORD where exists (SELECT 1 FROM marin.dbo.CUSTOMER WHERE
 CUSTID = marin.dbo.ORD.CUSTID AND state = 'MN')" queryout ORD.dmp -U"$1" -P"$2" -S"$3"
 -n -k > yyy_marin_dbo_ORD.out 2>&1 &
echo marin.dbo.ITEM
bcp "SELECT * from marin.dbo.ITEM where exists (SELECT 1 FROM marin.dbo.ORD WHERE ORDID
 = marin.dbo.ITEM.ORDID AND EXISTS(SELECT 1 FROM marin.dbo.CUSTOMER WHERE CUSTID =
 marin.dbo.ORD.CUSTID AND state = 'MN'))" queryout ITEM.dmp -U"$1" -P"$2" -S"$3" -n -k >
 yyy_marin_dbo_ITEM.out 2>&1 &
echo marin.dbo.PRODUCT
bcp "SELECT * from marin.dbo.PRODUCT where exists (SELECT 1 FROM marin.dbo.ITEM WHERE
 PRODID = marin.dbo.PRODUCT.PRODID AND EXISTS(SELECT 1 FROM marin.dbo.ORD WHERE ORDID
 = marin.dbo.ITEM.ORDID AND EXISTS(SELECT 1 FROM marin.dbo.CUSTOMER WHERE CUSTID =
 marin.dbo.ORD.CUSTID AND state = 'MN')))" queryout PRODUCT.dmp -U"$1" -P"$2" -S"$3" -n
 -k > yyy_marin_dbo_PRODUCT.out 2>&1 &
echo marin.dbo.PRICE
bcp "SELECT * from marin.dbo.PRICE where exists (SELECT 1 FROM marin.dbo.PRODUCT
 WHERE PRODID = marin.dbo.PRICE.PRODID AND EXISTS(SELECT 1 FROM marin.dbo.ITEM WHERE
 PRODID = marin.dbo.PRODUCT.PRODID AND EXISTS(SELECT 1 FROM marin.dbo.ORD WHERE ORDID
 = marin.dbo.ITEM.ORDID AND EXISTS(SELECT 1 FROM marin.dbo.CUSTOMER WHERE CUSTID =
 marin.dbo.ORD.CUSTID AND state = 'MN'))))" queryout PRICE.dmp -U"$1" -P"$2" -S"$3" -n -
k > yyy_marin_dbo_PRICE.out 2>&1 &
echo marin.dbo.ORD_PROC_LED_DETS
```

```
bcp "SELECT * from marin.dbo.ORD_PROC_LED_DETS where exists (SELECT 1 FROM
 marin.dbo.CUSTOMER WHERE CUSTID = marin.dbo.ORD_PROC_LED_DETS.CUSTID AND
 state = 'MN')" queryout ORD_PROC_LED_DETS.dmp -U"$1" -P"$2" -S"$3" -n -k >
 yyy_marin_dbo_ORD_PROC_LED_DETS.out 2>&1 &
echo marin.dbo.ORD_PROC_LED_DETS_ALT
bcp "SELECT * from marin.dbo.ORD_PROC_LED_DETS_ALT where exists (SELECT 1 FROM
 marin.dbo.CUSTOMER WHERE CUSTID = marin.dbo.ORD_PROC_LED_DETS_ALT.CUSTID AND
 state = 'MN')" queryout ORD_PROC_LED_DETS_ALT.dmp -U"$1" -P"$2" -S"$3" -n -k >
 yyy_marin_dbo_ORD_PROC_LED_DETS_ALT.out 2>&1 &
echo Working...
wait
cat yyy_marin_dbo_BONUS.out
cat yyy_marin_dbo_COMMPLAN_CODES.out
cat yyy_marin_dbo_CUSTOMER_ADDRESS.out
cat yyy_marin_dbo_CUSTOMER.out
cat yyy_marin_dbo_ORD.out
cat yyy_marin_dbo_ITEM.out
cat yyy_marin_dbo_PRODUCT.out
cat yyy_marin_dbo_PRICE.out
cat yyy_marin_dbo_ORD_PROC_LED_DETS.out
cat yyy_marin_dbo_ORD_PROC_LED_DETS_ALT.out
echo Extract complete
```

DB2

```
if [ "$1" = "" -o "$2" = "" -o "$3" = "" ] ; then
echo
echo \#\# ERROR SPECIFYING PARAMETERS
echo
echo \#\# TO RUN: user01_export database user password [working directory]
echo
exit -1
fi
# Change to the working directory if given one that is valid!
if [ ! "$4" = "" ] ; then
if [ -d $4 ] ; then
cd $4
fi
fi
rm -f user01_*.IXF
rm -f user01_batch_*.sh
echo db2 connect to $1 user $2 using $3 > user01_batch_CUSTOMER.sh
echo db2 EXPORT TO user01_CUSTOMER.IXF OF IXF SELECT \\* FROM USER01.CUSTOMER
 WHERE \\\(CUSTID \\\) IN \\\(SELECT L0.CUSTID from USER01.CUSTOMER L0 \\\) >>
 user01_batch_CUSTOMER.sh
echo db2 connect reset >> user01_batch_CUSTOMER.sh
```

```
echo db2 connect to $1 user $2 using $3 > user01_batch_CUSTOMER_ORDER.sh
echo db2 EXPORT TO user01_CUSTOMER_ORDER.IXF OF IXF SELECT \\* FROM
 USER01.CUSTOMER_ORDER WHERE \\\(CUSTID, ORDID \\\) IN \\\(SELECT L1.CUSTID, L1.ORDID
 from USER01.CUSTOMER_ORDER L1 INNER JOIN USER01.CUSTOMER L0 ON L1.CUSTID = L0.CUSTID \\
\) >> user01_batch_CUSTOMER_ORDER.sh
echo db2 connect reset >> user01_batch_CUSTOMER_ORDER.sh
echo db2 connect to $1 user $2 using $3 > user01_batch_ORD.sh
echo db2 EXPORT TO user01_ORD.IXF OF IXF SELECT \\* FROM USER01.ORD WHERE \\\(ORDID \\\)
 IN \\\(SELECT L2.ORDID from USER01.ORD L2 INNER JOIN USER01.CUSTOMER L0 ON L2.CUSTID =
 L0.CUSTID \\\) >> user01_batch_ORD.sh
echo db2 connect reset >> user01_batch_ORD.sh
echo db2 connect to $1 user $2 using $3 > user01_batch_ORD_PROC_LED_DETS.sh
echo db2 EXPORT TO user01_ORD_PROC_LED_DETS.IXF OF IXF SELECT \\* FROM
 USER01.ORD_PROC_LED_DETS WHERE \\\(ORDER_DATE, ORDID \\\) IN \\\(SELECT L3.ORDER_DATE,
 L3.ORDID from USER01.ORD_PROC_LED_DETS L3 INNER JOIN USER01.CUSTOMER L0 ON L3.CUSTID =
 L0.CUSTID \\\) >> user01_batch_ORD_PROC_LED_DETS.sh
echo db2 connect reset >> user01_batch_ORD_PROC_LED_DETS.sh
chmod +x user01_batch*.sh
echo Starting...
./user01_batch_CUSTOMER.sh > user01_CUSTOMER.out 2>&1 &
./user01_batch_CUSTOMER_ORDER.sh > user01_CUSTOMER_ORDER.out 2>&1 &
./user01_batch_ORD.sh > user01_ORD.out 2>&1 &
./user01_batch_ORD_PROC_LED_DETS.sh > user01_ORD_PROC_LED_DETS.out 2>&1 &
wait
cat user01_CUSTOMER.out
cat user01_CUSTOMER_ORDER.out
cat user01_ORD.out
cat user01_ORD_PROC_LED_DETS.out
echo Exports complete
```

Shell Scripts for Imports on Linux/Unix

Oracle

```
if [ "$1" = "" ] ; then
echo
echo \#\# ERROR SPECIFYING PARAMETERS
echo
echo \#\# TO RUN: yyy_import target_system_password[@tnsname] target_user [working
 directory]
echo
exit -1
fi
# Change to the working directory if given one that is valid!
```

```
if [ ! "$3" = "" ] ; then
if [ -d $3 ] ; then
cd $3
fi
targetuser=$2
elif [ ! "$2" = "" ] ; then
if [ -d $2 ] ; then
cd $2
targetuser=USER01
else
targetuser=$2
fi
else
targetuser=USER01
fi
imp system/$1 ignore=Y rows=Y constraints=N fromuser=USER01 touser=$targetuser
 file=yyy_full.dmp > imp_yyy_full.out 2>&1 &
imp system/$1 ignore=Y rows=Y fromuser=USER01 touser=$targetuser file=yyy_CUSTOMER.dmp >
 imp_yyy_CUSTOMER.out 2>&1 &
imp system/$1 ignore=Y rows=Y fromuser=USER01 touser=$targetuser file=yyy_ORD.dmp >
 imp_yyy_ORD.out 2>&1 &
imp system/$1 ignore=Y rows=Y fromuser=USER01 touser=$targetuser
 file=yyy_ORD_PROCESS_LEDGER_DETAILS.dmp > imp_yyy_ORD_PROCESS_LEDGER_DETAILS.out 2>&1 &
imp system/$1 ignore=Y rows=Y fromuser=USER01 touser=$targetuser
 file=yyy_ORD_PROCESS_LEDGER_DETAILS_ALL.dmp >
 imp_yyy_ORD_PROCESS_LEDGER_DETAILS_ALL.out 2>&1 &
imp system/$1 ignore=Y rows=Y fromuser=USER01 touser=$targetuser file=yyy_LEDGER.dmp >
 imp_yyy_LEDGER.out 2>&1 &
wait
# stream the output
cat imp_yyy_CUSTOMER.out
cat imp_yyy_ORD.out
cat imp_yyy_ORD_PROCESS_LEDGER_DETAILS.out
cat imp_yyy_ORD_PROCESS_LEDGER_DETAILS_ALL.out
cat imp_yyy_LEDGER.out
cat imp_yyy_full.out
# import other objects
imp system/$1 ignore=Y rows=N constraints=Y fromuser=USER01 touser=$targetuser
 file=yyy_norows.dmp
# analyse the user's objects
if [ -d $GTFILES ] ; then
if [ -f $GTFILES/analyse.pls ] ; then
sqlplus system/$1 @$GTFILES/analyse.pls $targetuser
else
echo $GTFILES/analyse not found - no analysis performed
fi
```

```
else
echo GTFILES not set or invalid - no analysis performed
fi
echo Import Complete
```

MySQL

```
if [ "$1" = "" -o "$2" = "" -o "$3" = "" -o "$4" = "" ] ; then
echo
echo \#\# ERROR SPECIFYING PARAMETERS
echo
echo \#\# TO RUN: yyy_import username password hostname target_database [working
 directory]
echo
exit -1
fi
# Change to the working directory if given one that is valid!
if [ ! "$5" = "" ] ; then
if [ -d $5 ] ; then
cd $5
fi
fi
echo Full Tables
mysql --user=$1 --password=$2 --host=$3 --database=$4 < yyy_GridTools_customer_type.sql
 > imp_yyy_GridTools_customer_type.out 2>&1 &
mysql --user=$1 --password=$2 --host=$3 --database=$4 < yyy_GridTools_dept.sql >
 imp_yyy_GridTools_dept.out 2>&1 &
echo No Data Tables
mysql --user=$1 --password=$2 --host=$3 --database=$4 < yyy_GridTools_Test1.sql >
 imp_yyy_GridTools_Test1.out 2>&1 &
mysql --user=$1 --password=$2 --host=$3 --database=$4 < yyy_GridTools_Test2.sql >
 imp_yyy_GridTools_Test2.out 2>&1 &
echo Subset Tables
mysql --user=$1 --password=$2 --host=$3 --database=$4 < yyy_GridTools_customer.sql >
 yyy_GridTools_customer.out 2>&1 &
mysql --user=$1 --password=$2 --host=$3 --database=$4 < yyy_GridTools_ord.sql >
 yyy_GridTools_ord.out 2>&1 &
echo Working...
wait
echo The Data Tables Import
cat yyy_GridTools_customer_type.out
cat yyy_GridTools_dept.out
echo The No Data Tables Import
cat yyy_GridTools_Test1.out
cat yyy_GridTools_Test2.out
echo The Subset Tables Import
```

```
cat yyy_GridTools_customer.out
cat yyy_GridTools_ord.out
echo Import Complete
```

MS SQL Server/Sybase

```
if [ "$1" = "" -o "$2" = "" -o "$3" = "" -o "$4" = "" ] ; then
echo
echo \#\# ERROR SPECIFYING PARAMETERS
echo
echo \#\# TO RUN: yyy_import username password server target_database [working
 directory]
echo
exit -1
fi
# Change to the working directory if given one that is valid!
if [ ! "$5" = "" ] ; then
if [ -d $5 ] ; then
cd $5
fi
fi
export GTOWNER=dbo
echo Full Tables
echo $4.BONUS
bcp "$4.${GTOWNER}.BONUS" in "BONUS.dmp" -U"$1" -P"$2" -S"$3" -n -k > imp_yyy_$4_
${GTOWNER}_BONUS.out 2>&1 &
echo $4.COMMPLAN_CODES
bcp "$4.${GTOWNER}.COMMPLAN_CODES" in "COMMPLAN_CODES.dmp" -U"$1" -P"$2" -S"$3" -n -k >
 imp_yyy_$4_${GTOWNER}_COMMPLAN_CODES.out 2>&1 &
echo $4.CUSTOMER_ADDRESS
bcp "$4.${GTOWNER}.CUSTOMER_ADDRESS" in "CUSTOMER_ADDRESS.dmp" -U"$1" -P"$2" -S"$3" -n -
k > imp_yyy_$4_${GTOWNER}_CUSTOMER_ADDRESS.out 2>&1 &
echo Subset Tables
echo $4.CUSTOMER
bcp "$4.${GTOWNER}.CUSTOMER" in "CUSTOMER.dmp" -U"$1" -P"$2" -S"$3" -n -k > imp_yyy_$4_
${GTOWNER}_CUSTOMER.out 2>&1 &
echo $4.ORD
bcp "$4.${GTOWNER}.ORD" in "ORD.dmp" -U"$1" -P"$2" -S"$3" -n -k > imp_yyy_$4_
${GTOWNER}_ORD.out 2>&1 &
echo $4.ITEM
bcp "$4.${GTOWNER}.ITEM" in "ITEM.dmp" -U"$1" -P"$2" -S"$3" -n -k > imp_yyy_$4_
${GTOWNER}_ITEM.out 2>&1 &
echo $4.PRODUCT
bcp "$4.${GTOWNER}.PRODUCT" in "PRODUCT.dmp" -U"$1" -P"$2" -S"$3" -n -k > imp_yyy_$4_
${GTOWNER}_PRODUCT.out 2>&1 &
echo $4.PRICE
```

```
bcp "$4.${GTOWNER}.PRICE" in "PRICE.dmp" -U"$1" -P"$2" -S"$3" -n -k > imp_yyy_$4_
${GTOWNER}_PRICE.out 2>&1 &
echo $4.ORD_PROC_LED_DETS
bcp "$4.${GTOWNER}.ORD_PROC_LED_DETS" in "ORD_PROC_LED_DETS.dmp" -U"$1" -P"$2" -S"$3" -n
 -k > imp_yyy_$4_${GTOWNER}_ORD_PROC_LED_DETS.out 2>&1 &
echo $4.ORD_PROC_LED_DETS_ALT
bcp "$4.${GTOWNER}.ORD_PROC_LED_DETS_ALT" in "ORD_PROC_LED_DETS_ALT.dmp" -U"$1" -P"$2" -
S"$3" -n -k > imp_yyy_$4_${GTOWNER}_ORD_PROC_LED_DETS_ALT.out 2>&1 &
echo Working...
wait
echo The Data Tables Import
cat imp_yyy_$4_${GTOWNER}_BONUS.out
cat imp_yyy_$4_${GTOWNER}_COMMPLAN_CODES.out
cat imp_yyy_$4_${GTOWNER}_CUSTOMER_ADDRESS.out
echo The Subset Tables Import
cat imp_yyy_$4_${GTOWNER}_CUSTOMER.out
cat imp_yyy_$4_${GTOWNER}_ORD.out
cat imp_yyy_$4_${GTOWNER}_ITEM.out
cat imp_yyy_$4_${GTOWNER}_PRODUCT.out
cat imp_yyy_$4_${GTOWNER}_PRICE.out
cat imp_yyy_$4_${GTOWNER}_ORD_PROC_LED_DETS.out
cat imp_yyy_$4_${GTOWNER}_ORD_PROC_LED_DETS_ALT.out
echo Import Complete
```

DB2

```
if [ "$1" = "" -o "$2" = "" -o "$3" = "" -o "$4" = "" ] ; then
echo
echo \#\# ERROR SPECIFYING PARAMETERS
echo
echo \#\# TO RUN: user01_import import_database user password target_schema [working
 directory]
echo
exit -1
fi
# Change to the working directory if given one that is valid!
if [ ! "$5" = "" ] ; then
if [ -d $5 ] ; then
cd $5
fi
fi
rm -f user01_load_*.sh
echo db2 connect to $1 user $2 using $3 > user01_load_CUSTOMER.sh
echo db2 IMPORT FROM user01_CUSTOMER.IXF OF IXF MESSAGES user01_CUSTOMER_import.log
 INSERT INTO $4.CUSTOMER >> user01_load_CUSTOMER.sh
echo db2 connect reset >> user01_load_CUSTOMER.sh
```

```
echo db2 connect to $1 user $2 using $3 > user01_load_CUSTOMER_ORDER.sh
echo db2 IMPORT FROM user01_CUSTOMER_ORDER.IXF OF IXF MESSAGES
 user01_CUSTOMER_ORDER_import.log INSERT INTO $4.CUSTOMER_ORDER >>
 user01_load_CUSTOMER_ORDER.sh
echo db2 connect reset >> user01_load_CUSTOMER_ORDER.sh
echo db2 connect to $1 user $2 using $3 > user01_load_ORD.sh
echo db2 IMPORT FROM user01_ORD.IXF OF IXF MESSAGES user01_ORD_import.log INSERT INTO
 $4.ORD >> user01_load_ORD.sh
echo db2 connect reset >> user01_load_ORD.sh
echo db2 connect to $1 user $2 using $3 > user01_load_ORD_PROC_LED_DETS.sh
echo db2 IMPORT FROM user01_ORD_PROC_LED_DETS.IXF OF IXF MESSAGES
 user01_ORD_PROC_LED_DETS_import.log INSERT INTO $4.ORD_PROC_LED_DETS >>
 user01_load_ORD_PROC_LED_DETS.sh
echo db2 connect reset >> user01_load_ORD_PROC_LED_DETS.sh
chmod +x user01_load_*.sh
wait
cat user01_CUSTOMER.bout
cat user01_CUSTOMER_ORDER.bout
cat user01_ORD.bout
cat user01_ORD_PROC_LED_DETS.bout
echo Loads complete
```

Using Templates to Generate Scripts

Data Subset supports the processing of xml files to produce tailored scripts. The xml template files reside in the templates sub directory under the Datamaker install directory. Typical installation path of the Datamaker is *C:\Program Files (x86)\Grid-Tools\GTDatamaker*. The templates directory has sub directories corresponding to the RDBMS connection. For example, *C:\Program Files (x86)\Grid-Tools\GTDatamaker\Templates\oracle* or *C:\Program Files (x86)\Grid-Tools\GTDatamaker\Templates\db2*.

Template Syntax

The template files must be in valid xml format. This means that certain characters need to be resolved into their xml compliant aliases as follows:

1. & in xml is &
2. < in xml is <
3. > in xml is >
4. " in xml is "
5. ' in xml is '

Example:

- <Organization>IBM & Microsoft</Organization> is an invalid XML string.
- <Organization>IBM & Microsoft</Organization> is a valid XML string.

Template XML Tags

Data Subset currently processes the following tags which must be in uppercase:-

<FILENAME>

This tag must be the first tag next to the template header tag in the template file. This specifies the output file to be written to.

Example:

<FILENAME> [ACTION NAME]_export.bat </FILENAME>

This will output the template script to a file named [ACTION NAME]_export.bat in the current open directory for the extract and replacing [ACTION NAME] with the action name entered in the Database Actions of Data Subset.

<PARMS>

This tag should appear below the file name tag. It is used to collect user input for one or more parameters before generating the resulting script. The left part of the tag value (before =) corresponds to the name of a special token that can be used anywhere in subsequent tags. The right part of the value (after =) corresponds to the value that is substituted instead of the special token at generation time.

Example:

XML Tag:

<PARMS>UNLOAD JOB CARD=//IDBJOBQX JOBINSOFT,INSOFT,CLASS=E,MSGCLASS=X,MSGLEVEL=(1,1)</PARMS>

Resulting dialog at the time of generation:

UNLOAD JOB CARD

//IDBJOBQX JOBINSOFT,INSOFT,CLASS=E,MSGCLASS=X,MSGLEVEL=(1,1)

The value in the text box will then be substituted wherever the [UNLOAD JOB CARD] special token is encountered while processing the template file.

<FORMAT>

This specifies the format of the output script file (The default is WINDOWS). Windows format ends each line with a new line and carriage return whereas UNIX ends each line with just a newline.

Example:

- – <FORMAT>WINDOWS</FORMAT>
  – <FORMAT>UNIX</FORMAT>

This tag is optional. If used it should be the second tag in the template file.

<TEXT>

Data Subset will output all characters within this tag to a separate line in the output script.

Example:

<TEXT>if "%1"= ="" goto error</TEXT>

Any special tokens will be processed before output.

<NEWFILENAME>

This tag may be used any number of times in the template file. When Data Subset encounters this tag, it will write out it’s processing of previous tags to the file name specified in the <FILENAME> tag and start processing subsequent tags against this new file name.

Example:

<NEWFILENAME> [ACTION NAME]_preview.sql </NEWFILENAME>

SQL Template Tags

[QUERY1]

WHERE clause only, using INNER JOIN syntax.

[QUERY2]

WHERE clause only, using EXISTS syntax.

[QUERY3]

SELECT * FROM owner.table and associated WHERE clause using INNER JOIN syntax. This SQL has the following format:

```
SELECT *

FROM TRAVEL.EMP

WHERE (EMPNO)

IN (

SELECT L3.EMPNO

from TRAVEL.EMP L3

INNER JOIN TRAVEL.CUSTOMER L0 ON L3.EMPNO = L0.CUSTID

AND CUSTID > 90 )
```

The additional IN clause based on primary key or uniquely indexed columns prevents duplicates for many too many table joins. The recommended tag to use is [QUERY3]

Looping XML Tags

The following special tags will process <TEXT> tags in a loop.

<ALLDATA>

This tag processes for all tables in the ‘All Data Tables’ list in the Database Actions in Data Subset.

Example:

<ALLDATA> <TEXT>spool [TABLE].txt</TEXT> </ALLDATA>
This Will output zero or more lines corresponding to the number of tables in the ‘All Data Tables’ list, replacing [TABLE] each time with the table name in the list.

<ALLEXTRACT>

This tag processes for all tables in the ‘Subset’ list in the Database Actions in Data Subset.
Example:

<ALLEXTRACT> <TEXT>spool [TABLE].txt</TEXT> </ALLEXTRACT>
This Will output zero or more lines corresponding to the number of tables in the ‘Subset’ list, Replacing [TABLE] each time with the table name in the list.

<NODATA>

This tag processes for all tables in the ‘No Data Tables’ list in the Database Actions in Data Subset.

Example:

<NODATA> <TEXT>spool [TABLE].txt</TEXT> </NODATA>
This Will output zero or more lines corresponding to the number of tables in the ‘No Data Tables’ list, replacing [TABLE] each time with the table name in the list.

<ALLTABLES>

This tag processes for all tables in the ‘All Data Tables’ , ‘No Data Tables’ and ‘Subset’ lists in the Database Actions in Data Subset.

Example:

<ALLTABLES> <TEXT>spool [TABLE].txt</TEXT> </ALLTABLES>
This Will output zero or more lines corresponding to the number of tables in ‘All Data Tables’ , ‘No Data Tables’ and ‘Subset’ lists, replacing [TABLE] each time with the table name in the lists.

<ALLCOLS>

This tag processes all columns for a particular table, and for this reason must be placed in one of the above table loop tags.

Parameter Tokens

The script generator will substitute certain parameter tokens with their true values. These tokens are enclosed in square brackets in the template file.
The parameter tokens that can be used are as follows:

[ACTION NAME]

This token is replaced with the Action Name entered in the Database Actions in Data Subset.

[GCOUNTER1]

Currently up to four counters can be used in template files (GCOUNTER1 … GCOUNTER4). These tokens are replaced by a numeric value starting at one and incrementing by one each time it is used. For this reason this token should be used inside a looping tag.

[SP]

This replaces the token with a white space – this can be useful when indenting text in the output script, since any leading spaces within <Text> tags are ignored by the XML parser.

[DBDEFAULT]

This token is replaced by the Default Database specified when connecting to MS SQL Server.

[OWNER]

This token is replaced by a table’s owner.

[TABLE]

This token is replaced by a table name.

The above 3 parameters should only be used in Table Looping Tags. For example, <ALLTABLES> , <ALLDATA> , <ALLEXTRACT> or <NODATA>

[DIRECTORY]

This token is replaced with the Extract Directory specified in the Database Actions in Data Subset.

[GTSYMBOL]

The first time this is used it is replaced by ‘>’ any subsequent uses are replaced by ‘>>’.

Output Separate Files in Looping Tags

Adding a <FILENAME> tag as the first tag in a looping tag will output the subsequent text in the <FILENAME> tag to separate files.

Example:

<ALLEXTRACT> <FILENAME>[ACTION NAME]_[TABLE].ndl</FILENAME> <TEXT>import @[TABLE].txt</TEXT> </ALLEXTRACT>

The text in the looping tag would then be ignored by the processing for the previous <FILENAME> or <NEWFILENAME> tags.

The following tags currently have a hard coded output, and must be placed in a table looping tag. For example, <ALLDATA>, <ALLEXTRACT>, <NODATA> or <ALLTABLES>.

<ALLSANDTABCOLS>

This tag if used must be placed in a table looping tag. The output is designed for use with Oracle table types. For all columns for a specific table the following output is produced:

1. If the columns data type is date - to_char([columnname], 'YYYY-MM-DD-HH24.MI.SS') || CHR(9)
2. If the columns data type is number or pls_integer - nvl(ltrim(to_char([columnname])),' ') || CHR(9) || (if column is not nullable) or ltrim(to_char(CUSTID)) || CHR(9) || if the column is nullable.
3. If the columns data type is a character type nvl(rtrim(to_char([columnname])),' ') || CHR(9) || (if column is not nullable) or rtrim(to_char(CUSTID)) || CHR(9) || if the column is nullable.
4. For all other data types nvl(rtrim(to_char([columnname])),' ') || CHR(9) || (if column is not nullable) or rtrim(to_char(CUSTID)) || CHR(9) || if the column is nullable.

<ALLTABCOLS>

This tag if used must be placed in a table looping tag. It has a similar output to the above <ALLSANDTABCOLS> tag, the only difference being that for date column types the output is - to_char([columnname], 'YYYYMMDDSSSSS') || CHR(9).

<ALLTABIMPORTCOLS>

This tag if used, must be placed in a table looping tag, the output is designed for use with Oracle table types. For all columns for a specific table the following output is produced.

1. If the columns data type is date - [columnname] DATE(13) "YYYYMMDDSSSSS"
2. If the columns data type is number or pls_integer - [columnname] char(6)
3. For all other types - [columnname] char([columnwidth])

<ALLCOLS>

This tag if used must be placed in a table looping tag. It processes the column list for the given [TABLE] in the table looping tag. The column name is referenced by the [COLUMN] special token.

Running Extracts and Imports

Create batch files or shell script files to extract and import the data from the database based on the operating system you are using.

MS SQL Server/Sybase

Both Sybase and MS SQL Server support the bulk copy protocol (bcp) extract and import mechanism, and as such the scripts for each database are very similar. With both Sybase and MS SQL Server, the tables must be created in the test database before you export and import the data. Use the available standard utilities to do this.

Run the following command to extract the data:

- **Windows**
  extractname_export.bat username password server
- **Linux**
  extractname_export.sh username password server {working data directory}
  This job will issue bcp commands to extract the data. When the job has completed it will have created files with a suffix of *.dmp* in the local directory or directory specified in the run command. These may be moved to another machine, if required.
  Notes:
  – On Linux issue: *chmod +x *.sh* to make the shell scripts executable.
  – Ensure that the Sybase or MS SQL Server \bin directory is on the path.

Run the following command to import the data:

- **Windows**
  extractname_import targetusername targetpassword targeserver targetdatabase

  NOTE

  To import data to an instance of an MSSQL database, you must include the port number (**-port**) and target schema name (**-target**) parameters. For example:

  extractname_import targetusername targetpassword targetserver targetdatabase -port <*port number*> -target <*schema name*>
- **Linux**
  extractname_import.sh targetusername targetpassword targetserver targetdatabase {.dmp ldirectory location}
  This will issue bcp command to import the data. If you have extracted the data to a different directory during the extract you must add it as an extra parameter.

Oracle

Run the following command to extract the data:

- **Windows**
  extractname_export.bat userid/password[@tnsname]
- **Linux**
  exportconnection_userid/password[@tnsname]
  This command issues Oracle exports to extract the data. It creates the necessary files in .dmp format. You can move these files to another machine as necessary.

  NOTE

  On Linux, run chmod +x*.sh to make the shell scripts executable.

  On Linux, if you wish to extract the data into a specific directory include a working directory name at the command prompt.

Run the following command to import the data:

- **Windows**
  extractname_import.bat userid/password[@target_tnsname] target_user
- **Linux**

extractname_import.sh userid/password[@tnsname] target_user
This command issues Oracle imports to import the data. The import creates the necessary tables and enables the constraints.

NOTE

If you are loading into a different schema, include the target schema name on the command line.

Oracle Database Scheduler

Create a UNIX account to run the jobs, in the account copy up the example shell scripts (.sh) files in the \GTFiles sub folder. Following is a sample file:

```
#!/bin/sh
PATH=/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:.
export PATH
echo $# $0 $* > $0.errlog
if ! [ $# -ge 2 ] ; then
echo \#\# ERROR SPECIFYING PARAMETERS
echo \#\# ERROR SPECIFYING PARAMETERS >> $0.errlog
echo \#\# At least 2 parameters \(Directory and File\) must be specified
echo \#\# At least 2 parameters \(Directory and File\) must be specified >> $0.errlog
 exit -1
fi

if ! [ -d "$1" ] ; then
echo
echo \#\# ERROR SPECIFYING PARAMETERS
echo \#\# ERROR SPECIFYING PARAMETERS >> $0.errlog
echo \#\# First Parameter \"$1\" must be a directory
echo \#\# First Parameter \"$1\" must be a directory >> $0.errlog
exit -1
fi
cd $1/$2
echo $# $0 $* $PWD >> $0.errlog
echo Whoopeee!!!!! >> $0.errlog

ORACLE_SID=o10gamst
ORAENV_ASK=NO
. oraenv
ORAENV_ASK=

fileexport=$2_export.sh
fileimport=$2_import.sh
shift 2
echo $# $0 $* $PWD >> $0.errlog
echo $fileexport $1 >> $0.errlog
$fileexport $1 > $fileexport.log 2>&1
shift 1
```

```
echo $fileimport $* >> $0.errlog
$fileimport $* > $fileimport.log 2>&1
```

- Customize the file as needed for your environment. You can, for example, customize these scripts to email any results file once a job is complete.
- In the Oracle Enterprise manager, create a new job with the parameters required. In this case, the Oracle Export and then the Oracle Import jobs.

MYSQL

The MySQL utilities mysqldump and mysql will be used to extract and load the data. To view the available options for each utility from a command prompt enter:

- mysqldump –help | more
- mysql –help | more

This will display all the available options. It is possible to change the default options used by Data Subset™ by either including the additional options as an extra parameter on the export or by simply editing the generated scripts prior to execution. If there is a particular option you would like added by default please contact support so it can be added.

The Data Subset default is to include a create table statement as part of the extract file, if you wish to:

- Drop and recreate the table include the option –add-drop-table, this will drop the table if it already exists.
- Not include create tables in the extract include the option –no-create-info, this will remove the create table statement from the extract file.

Run the following command to extract the data:

- **Windows**
  extractname_export.bat username password hostname {extra mysqldump parameters}
  If you wish to include a specific mysqldump parameter include it at the end. For example, *yyy_export huw huwpswd huwmachine –add-drop-table.*
- **Linux**
  extractname_export.sh username password hostname {data directory name & extra mysqldump parameters}
  This job will issue mysqldump commands to extract the data. When the job has completed it will have created files with a suffix of .sql in the local directory or directory specified in the run command. These may be moved to another machine if required.
  Notes:

1. a. To specify extra mysqldump options you must include the directory name you wish the data extract files to be written to, for exampleyyy_export.sh huw huwpswd huwhost \home\huw\data –add-drop-table.
   b. Run the command *SET PATH="c:\program files\MySQL\MySQL Server 4.1\bin";%path%* to add the MySQL bin directory to your path.

Run the following command to import the data:

- **Windows**
  extractname_import targetusername targetpassword targethostname targetdatabase
- **Linux**
  extractname_import.sh targetusername targetpassword targethostname targetdatabase {.sqldirectory location}
  This will issue mysql command to import the data. The import will create the table if they do not exist by default. If you have extracted the data to a different directory during the extract you must add it as an extra parameter.

DB2

Any tables in the extract need to be created in the test database prior to exporting and importing the data. Use the standard provided utilities to do this.

Run the following command to extract the data:

- **Windows**
  extractname_export.bat [DB Connect database name] [username] [password]
  Where [DB Connect database name] is the database connection name defined in your DB2 client configuration
- **Linux**
  extractname_export.sh database user password *{working data directory}*
  Where [DB Connect database name] is the database connection name defined in your DB2 client configuration And [username] [password] are the username and password for the connection.

Run the following command to import the data:

- **Windows**
  extractname_import.bat [DB Connect database name] [import_schema] [username] [password]
  Where [DB Connect database name] is the database connection name defined in your DB2 client configuration, import_schema is the database name you will be importing data into and the username and password to connect to the [DB Connect database name].
- **Linux**
  extractname_import.sh import_database user password target_schema [working directory]
  Where import_database is the database connection name defined in your DB2 client configuration and user and password are the username and password for that connection, Target_schema is the database you wish to load data into.
  This will issue bcp command to import the data. If you have extracted the data to a different directory during the extract you must add it as an extra parameter.

Example: Create a Subset of Data Stored in Relational Database

This example use case provides information about how you can subset the data stored in the Travel database using TDM Subset. This example also helps you understand the overall flow and steps that you perform to complete a data subset task.

CA TDM is shipped with some sample databases. This example uses the sample database Travel which is installed on MS SQL Server. For more information about how to install the TDM repository and sample databases, see Install the Repository.

The following topics cover the end-to-end example:

Scenario

John works as a Test Data Engineer in a company. His current assignment is to provide data to test a Travel Management application. The data for the Travel Management application is spread across various tables in Travel database in the Microsoft SQL Server. The number of tables in the database and the amount of the data is so huge to manage for John's current assignment. So he wants to create a subset of the data.

Prerequisites

The following are the prerequisites:

- Verify that you have created source and target database in Microsoft SQL Server. The following values in this example:

  - **MS SQL Server Name:** TravelSystem-DatabaseServer
  - **Port:** 1433
  - **User Name:** sa
  - **Password:** P@ssw0rd
  - **Source Database (Data Source):** Travel (Uses travel.bak and travel_user.sql files which are shipped with the CA TDM product.)
  - **Target Database (Data Target):** Travel_e (Uses travel_e.bak and travel_e_user.sql files which are shipped with the CA TDM product.)

  For more information about how to install target and source databases, see Install the Repository.
- Verify that you have created a connection profile in CA TDM Datamaker to connect to the data source and data target. The connection profile Travel_SQLServer in CA TDM Datamaker is used in this example.
  For more information about how to create a connection profile, see Connect to the Repository.
- Verify that you have created the appropriate project, version and a GT Subset type of Data Set in the CA TDM Datamaker. The following values are used in this example:
  - **Project Name:** TravelSystem
  - **Version:** 1.0
  - **Subset type Data Set:** Travel_Subset

  For more information about how to create a project, version, and data set, see Create a Project.
- Verify that you have registered the required tables from source database to the project version you have created. All the 53 tables from the data source Travel are registered in this example.
  For more information about how to register tables, see Register Database Tables.

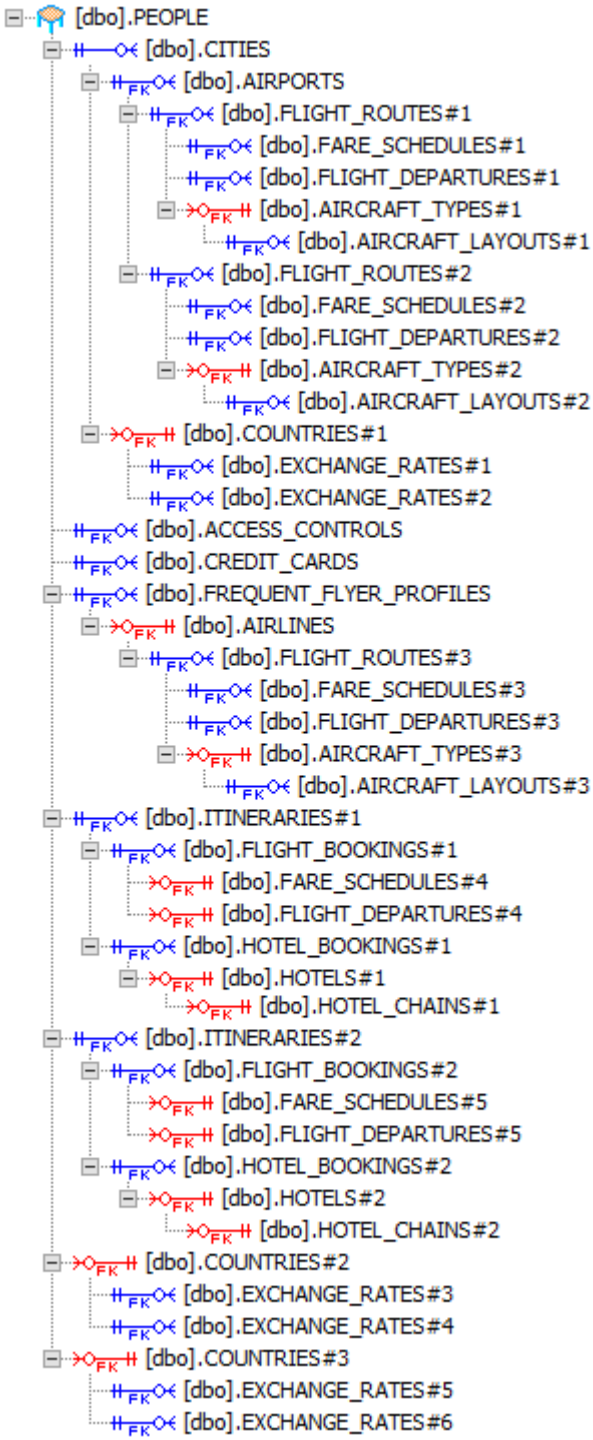
Establish Database Connection

1. Access the CA TDM Datamaker using the connection profile Travel_SQLServer connection profile.
2. Go to the Projects menu and click Project Manager.
3. Expand the projects tree and select the version 1.0 under the project TravelSystem.
4. Go to the Data Subset menu and click Design Extracts and Transactions.
   GT Subset Source Connection window opens.
5. Enter the following and click Save.
   - **GT Subset Profile Name:** Travel_SQLServer
   - **User Name:** sa
   - **Password:** P@ssw0rd
   - **DBMS:** MS SQL Server
   - **Server:** TravelSystem-DatabaseServer
   - **Port:** 1433
   - **Default DB:** gtrep
   - **Default Schema:** dbo

   Database connection is successfully established and CA TDM Subset application is launched.

Create Extract Definition from Data Source

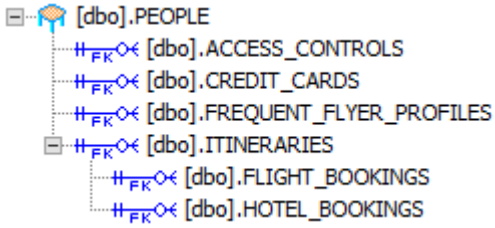
1. Select TravelSystem from the Project drop-down list.
2. Select 1.0 from the Version drop-down list.
3. Select dbo from the Select Schema drop-down list.
4. Select People from the Select Table drop-down list to use as driving table.
   Table relationships are displayed in the data navigation tree as shown in the below figure:



5. Right click on the driving table People and click Remove ‘Red’ Relationships.
   Removes the children that have the red relationship. Red relationship means that the respective table is a parent to the selected driving table PEOPLE.
6. Right click the following tables and click Remove Link from Extract.

- CITIES – This table is removed because there is no data in this table.
- ITINERARIES#2 – This table is removed because this table has two different relationships with the names ITINERARIES#1 and ITINERARIES#2.

7. After modifying the relationships, the table relationships are displayed in the data navigation tree as shown in the below figure:



8. Select the driving table PEOPLE from the data navigation tree, go to SQL tab and modify the SQL as shown below and click the Execute SQL button:

```
select * from [dbo].[PEOPLE]

where ID<30
```

- Under the Status tab the following message is shown to confirm that the SQL is successfully executed:

```
Query executed in 0.7 seconds

22 rows returned
```

- Under the results tab the following data is returned that matches the condition you specified in the SQL.

...	DESIGNATION	FIRST_NAME	LAST_NAME	JOB_TITLE	L...	EMAIL	CONTACT_PHONE	HOME_PHONE
1	MR	AMBER NOEL	GILDENHORN	C1	SAL	AGILDENH@us.oracle.com	201 890 8902	201 567 7890
3	MR	KATHLEEN	BAYYAT	C3	CON	KBAYYAT@us.oracle.com	555-0000	Not Known
6	MR	ERIK	CARDENAS	C3	CON	ECARDENA@us.oracle.com	555-0000	Not Known
9	MR	LAUREN MISS	NISHIOKA	C3	CON	LNISHIOK@us.oracle.com	555-0000	Not Known
10	MR	BENJAMIN	PLUMMER	C3	CON	BPLUMMER@us.oracle.com	555-0000	Not Known
12	MR	MATT	ADKINS	C3	CON	MADKINS@us.oracle.com	555-0000	Not Known
13	MR	ROBERT	RODRIGUZ	C3	CON	RRODRIGU@us.oracle.com	555-0000	Not Known
14	MR	FAWZI	MARJANOVIC	C3	CON	FMARJANO@us.oracle.com	555-0000	Not Known
15	MR	SERGIO	KAVANAUGH	C3	CON	SKAVANAU@us.oracle.com	555-0000	Not Known
16	MR	JOAO	LARSH	C3	CON	JLARSH@us.oracle.com	555-0000	Not Known
17	MR	FRANCISCO	HUMPHREYS	C3	CON	FHUMPHRE@us.oracle.com	555-0000	Not Known
18	MR	ELIZABETH	CORRADO	C3	CON	ECORRADO@us.oracle.com	555-0000	Not Known
20	MR	SHINICHI	REZENDE	C3	CON	SREZENDE@us.oracle.com	555-0000	Not Known
21	MR	EDWARDMR	SHIELDS	C3	CON	ESHIELDS@us.oracle.com	555-0001	Not Known
22	MR	ANIL	FELONG	C3	CON	AFELONG@us.oracle.com	555-0001	Not Known
23	MR	PAUL	WIECZOREK	C3	CON	PWIECZOR@us.oracle.com	555-0001	Not Known
24	MR	ALEJANDRO	NISHIOKA	C3	CON	ANISHIOK@us.oracle.com	555-0001	Not Known
25	MR	JOVAN	JACOBS	C3	CON	JJACOBS@us.oracle.com	555-0001	Not Known
26	MR	STEPHEN L	BRADLEY	C3	CON	SBRADLEY@us.oracle.com	555-0001	Not Known
27	MR	LANCE	BETTIN	C3	CON	LBETTIN@us.oracle.com	555-0001	Not Known
28	MR	MIKE	BARRIERE	C3	CON	MBARRIER@us.oracle.com	555-0001	Not Known
29	MR	THOMAS V	BAYYAT	C3	CON	TBAYYAT@us.oracle.com	555-0001	Not Known

9. Go to the File menu and click Save Extract to Repository.
The Save Extract to Repository window opens.

10. Enter the values in the fields and click Save. Click OK in the confirmation dialog.
    The following values are entered in this example:
    - **Extract Name:** TravelDataSubset
    - **Description:** TravelDataSubset
    - **Save Extract To:** Project:TravelSystem Version:1.0 Set:Travel_Subset

    The extract is successfully saved.
    Note: You can verify the saved extract details in the Datamaker, from the respective datapool (Travel_Subset).

Generate Extract and Import Scripts

1. Select Build MS SQL Server Export/Import from the drop-down list beside the Execute SQL button.
2. The Database Actions - Build MS SQL Server Export/Import window opens.
3. Go to the Extract Details tab and specify the Action Name (TravelSystemDataSubset).
4. Under the Extract Tables tab select the following:
   - Select the From Repository radio button.
   - Select the extract you saved to repository (TravelDataSubset).
5. Click Generate.
6. Select the default directory to save the scripts, if not already done. In this example the folder path *C://TravelSystem/* is selected as default directory.
   Create Export confirmation dialog opens showing the details of the Export Scripts generated. The following files are created in this example:
   - C:/TravelSystem/TravelSystemDataSubset/TravelSystemDataSubset_export.bat
   - C:/ TravelSystem /TravelSystemDataSubset/TravelSystemDataSubset_import.bat
   - C:/ TravelSystem /TravelSystemDataSubset/TravelSystemDataSubset_preview.sql

   Export and Import Scripts are successfully generated.

Run the Export Script to Extract the Data from Data Source

1. Launch the command line interface and change the directory to the folder which includes the export.bat file.
   The directory used in this example is *C:/TravelSystem/TravelSystemDataSubset/.*
2. Run the command to execute the export.bat file in the format "*<file name.bat> <user name> <password> <server>"*.
   The command used in this example is, *TravelSystemDataSubset_export.bat sa techpubs@123 sarsa06-i182691.*
3. The data is successfully exported. Export log files are created in the same folder where export and import script files are saved. In this example, the following files are created:e sample databases. This example uses the sample database Travel which is installed on MS SQL Server. For more information about how to install the TDM repository and sample databases, see Install the Repository.
   - C:/TravelSystem/TravelSystemDataSubset/PEOPLE.dmp
   - C:/TravelSystem/TravelSystemDataSubset/PEOPLE_export.txt
   - C:/TravelSystem/TravelSystemDataSubset/TravelSystemDataSubset_export.txt
   - C:/TravelSystem/TravelSystemDataSubset/TravelSystemDataSubset_export_ok.txt

   Exporting data from data source is successfully completed.

Run the Import Script to Import the Data into Data Target

1. Launch the command line interface and change the directory to the folder which includes the import.bat file. The the directory used in this example is *C:/TravelSystem/TravelSystemDataSubset/*.
2. Run the command to execute the import.bat file in the format "*<file name.bat> <user name> <password> <server> <target database name>"*. The command used in this example is, *TravelSystemDataSubset_import.bat sa techpubs@123 sarsa06-i182691 Travel_e*.
3. The data is successfully imported. Import log files are created in the same folder where export and import script files are saved. In this example, the following files are created:

– *C:/TravelSystem/TravelSystemDataSubset/TravelSystemDataSubset_import_ok.txt*
Importing data into data target is successfully completed.

You have successfully exported a subset of large data from the source database Travel and imported to target database Travel_e.

Subset Data for iSeries V7R1 (DB2/400)

You can use the CA TDM Datamaker UI to subset data for DB2/400. The Subset Production Data section in the CA TDM wiki includes the generic high-level process that explains how you can use Datamaker to subset your data for different databases. This generic process covers the following steps:

- Establish Database Connection
- Select Data to Subset
- Define Relationships
- Save Extract Definitions
- Generate Scripts to Move Data
- Prepare Subset Schema
- Running Extracts and Imports

This process is also applicable for DB2/400. However, some of the options in the Generate Scripts to Move Data section change for DB2/400. Therefore, we recommend you to review the following steps to understand how to use those options in the context of DB2/400:

1. Access the GT Subset interface from the Datamaker UI.
2. Select your source schema, driving (parent) table, and create and execute the SQL statement as appropriate.
3. Select the **Build SQL Insert Script** option from the drop-down list next to SQL and click the Database Actions icon (forward arrow).
   The **Database Actions - Build SQL Insert Script** dialog opens.
4. Click the **Extract Details** tab, enter the action name, select the target database, and perform one of the following actions as applicable:
   – CTAS (Create Table As Select)
   For CTAS subset, only target database instance should be present without having database tables. The generated script out of the subset creates the required database tables into the target database schema and inserts the subset data from source database into the target database instance.
   Select the **Create Table As Select** option and click the **Extract Tables** tab.
   – Non-CTAS
   For non-CTAS subset, the database instance containing the same database schema as in source should be present without any data. The generated script out of the subset inserts the subset data from source database into the target database instance.
   Click the **Extract Tables** tab.
5. Select the transformation map created for the subset and click **Generate**.
   A subset insert SQL script is generated.
   You need to execute the SQL insert query from the Datamaker SQL window of the target DB instance. The query extracts the data from the source database tables, performs the subset based on the transformation map, and inserts the subset data into the target database tables. You perform this action as part of the Running Extracts and Imports section.

Generate Data Definition Expressions for Cloning and Subsetting

In a dataset, create an empty data pool in which you generate your data definition expressions. Provide a subset extract that defines table relationships, and use the information on these relationships to create expressions in table definitions.

These complex expressions are generated automatically by the Data Creation Assistant to allow Datamaker to clone or subset data from source to the target.

1. In Datamaker, click Settings, Enable Data Creation Assistant.
2. Right-click and edit the datapool. The data creation assistant window opens.
3. Choose Subset Data and select your extract.
4. Click Create Data Based on Subset.
5. Choose one of the following Subset Conditions and click the tickmark:
   – Exists Select
   – Inner Join Select
6. Review the row population rules for this condition and click the tickmark to confirm. The Data Definition Row Editor opens.

The table is populated with expressions that correspond to the subset extract used.

Enable Debugging for Subset

You can enable debugging for Subset, to review the issues you are experiencing while using Subset and send the log files to CA Support.

Follow these steps:

1. Ensure that the CA TDM Subset application is closed.
2. Launch the command line interface and change the directory to Subset installation directory folder. The default directory for this is *C:\Program Files (x86)\Grid-Tools\GTDatamaker*.
3. Run the command **java -jar gtsubset.exe**.
   Launches the **Subset** application and records debugging information.
4. In the Subset application UI, establish database connection.
5. Recreate the issue that you experienced while using the Subset application.
6. Review the **stack trace** appears in the command prompt.
   You have successfully enabled the debugging for Subset.
7. Analyze the stack trace and fix the issue accordingly. Alternatively copy the stack trace information to a text file and send to CA Support to fix the issue.

Example: Mask Tables With Linked Seed Data (Teradata)

This scenario shows how you mask multiple columns in a table whose seed data relates to each other. The use case for this masking scenario is that you are masking, for example, an address table where the city, state, and zip code need to match. This example uses a Teradata database.

You want to ensure that the seed data has matching data columns, and that the hash operates on the same columns.

Follow these steps:

1. Issue the following SQL to find which data categories have linked data. Substitute the database name where you have installed the masking objects for FUNCTIONS:
   `SELECT DISTINCT rd_ref_id FROM FUNCTIONS.gtsrc_reference_data WHERE rd_ref_value2 IS NOT NULL`
   The query returns a list of linked `rd_ref_id` s.
2. Pick a linked data category and query it in the seed table. Pick specific columns in the result to use for each column mask. For example:

```
SELECT * FROM FUNCTIONS.gtsrc_reference_data WHERE rd_ref_id = 'US ADDRESS'
```

3. Create a transformation map of type TERADATA.
4. Set the column masks according to the data.
   - Type the data category name if it does not exist in the drop down, but does exist in the seed data table.
     Example: Type `HASHLOV,US ADDRESS` in the Transformation column.
   - Ensure that the List Col# number matches the corresponding column in the seed data table.
   - To link the columns in the table to be masked, ensure that the hash operates on the same column in the table, so that values are picked from the same row in the seed table.
5. Specify the column by putting the column name in the notes section of the transformation map.
   In general, you use a primary key column, or at least a column with as many unique values as possible.
   Example: `ADDRESS_ID`

Transformation	Rel	Keepnulls	List Col#	Fixed Value	Cross Ref	Cross Ref Ident	Substring Start	Substring Length	WHERE Clause (Double-click column to edit)	Notes
HASHLOV,US ADDRESS			2							ADDRESS_ID
HASHLOV,US ADDRESS			6							ADDRESS_ID
HASHLOV,US ADDRESS			1							ADDRESS_ID
HASHLOV,US ADDRESS			4							ADDRESS_ID

Example: Generate Insert Scripts for Oracle from Subset

The fastest way to create a subset of data in GTSubset is to generate direct insert scripts.

In a simple subset design like in the following example, you want to extract out to a file using, for example, DataPump.

- TRAVEL.PEOPLE
  - TRAVEL.ACCESS_CONTROLS
  - TRAVEL.CREDIT_CARDS
  - TRAVEL.FREQUENT_FLYER_PROFILES
    - TRAVEL.AIRLINES
      - TRAVEL.FLIGHT_ROUTES
        - TRAVEL.FARE_SCHEDULES
        - TRAVEL.AIRCRFAT_TYPES
          - TRAVEL.AIRCRAFT_LAYOUTS
        - TRAVEL.AIRPORTS

To get the rows for the table AIRCRAFT_LAYOUTS, the select must join on AIRCRAFT_TYPES, FLIGHT_ROUTES, AIRLINES, FREQUENT_FLYER_PROFILES and finally PEOPLE. Joining 6 tables is an inefficient task for any database.

The direct insert scripts rely on a temporary schema being present next to the schema you are subsetting. This schema can be empty, or contain empty copies of the tables in source.

The inserts load the data sequentially according to the extract design. The rows for PEOPLE are inserted first, then for ACCESS_CONTROL, the select is from source, but joined on the rows already loaded into target in the first SQL insert. Similarly, once ACCESS_CONTROL is loaded, the rows for AIRLINES need only join on the already loaded rows in ACCESS_CONTROL. This way, very large data sets can be subsetted in a very streamlined way.

After the subset is complete, you can perform a full schema datapump extract of the target (temp) schema, and load the corresponding files into a database on another server if required.

To generate these types of scripts, first design and save an extract in the normal way.

1. Select ‘Build SQL Insert Script’ as the script generation option from the drop down.
   The Database Action - Build SQL Insert Script window opens.

2. Fill in the following fields:
   - **Action Name**
     Defines the prefix name for the generated scripts
   - **Max Parallel Processes**
     Defines the Oracle Parallel number to use for the SQL Insert, for example 4.
     `INSERT  /*+ append parallel(TRAVEL_E.PEOPLE, 4)`
   - **Target Database**
     Defines the database schema into which you want to insert the subsetted data.
   - **Create Table As Select**
     Specifies whether generated SQL creates and populates the tables in one SQL statement. This is the fastest option in terms of loading of the data, but the tables are created without any constraints, indexes or keys. Enable this option if your target schema is totally empty.
     Note: If you enable this option, many of the other script options are not necessary, for example, check for constraints, create merge script, or check for existing data.
   - **Check for constraints**
     Specifies whether you want to generate FK disable and enable scripts based on the constraint present in the target schema. Enable this option only if you are loading into a constrained schema, that is, your target schema has some foreign key constraints. Enabling this options prompts you for a connection to your target schema.
   - **Create merge scripts**
     Specifies whether you want to generate two additional script types, insert and upsert merge scripts. Use these if you want to merge data from source to target and if target already contains data.
   - **Check for existing Data**
     Specifies whether to add a ‘NOT EXISTS’ clause to the insert statements, in other words only new (not duplicated) data is inserted. Use this if you did not enable 'Create Merge Script', and your target tables contain data.
   - **Database link**
     Specifies whether to use a link to reference target or source tables, or both. If your target schema does not reside on the same server as source, choose the option to use database links to join the tables. This requires that the database link already exists.
     Note: Joining tables across DB Links is usually pretty slow, so this option is not recommended.
   - **Get Metadata from Repository Connection**
     Subset has to query the Oracle catalog tables to get meta information for all the tables in an extract to generate the scripts. For large schemas this can be quite slow, and the generation process can take several minutes to complete. If you register your source tables in Datamaker, enabling this option will get the meta info from the Datamaker repository instead, making the generation process much quicker.
3. Click generate.

Mask Production Data with Fast Data Masker

Data masking hides or obfuscates sensitive and classified data. The goal is to protect data that is used for purposes such as development, testing, and QA cycles. Data masking is a standard practice that is often required for compliance with national and international data protection legislation.

Test Data Manager Fast Data Masker is a simple-to-use, high performance data masking tool for data protection across multiple projects of any size and complexity. The Fast Data Masker interface lets you create, save, run, and reuse high performance masking definitions. These definitions enable you to secure large volumes of complex data across multiple systems, and comply with data protection regulations.

Various data masking functions provide systematic, repeatable methods to secure personally identifiable information (PII). These functions are defined as rules to mask and anonymize the data. Examples of the available masking rules include the following:

- Using seed tables
- Multi-table column values
- Hashing
- Replacements
- Custom Functions
- Translations
- Offset dates
- Substitution
- Credit Cards
- Random ranges and text
- Numeric variances

For more information about the masking functions, see Masking Functions and Parameters.

Fast Data Masker and Transformation Maps

In addition to Fast Data Masker, you can also use Datamaker transformation maps to mask the data. The approach that you select depends on your business requirements and feasibility. You can adopt one of the following approaches to masking with regards to which stage the data is masked at:

- **Using Fast Data Masker (In-place Masking)**
  In this case, a typical scenario is that the production data is copied over to a staging area. Fast Data Masker points to this staging database and masks the data that resides there. This *masked* data is then copied over to different testing environments as required.
- **Using Transformation Maps (In-flight Masking)**
  In this case, you use Datamaker transformation maps and Subset scripts. You first define a transformation map (Oracle or MSSQL) in Datamaker, create masking functions for the columns you want to mask. You use the Subset interface to create the *masked* export scripts. These scripts perform masking as they export the source data to a dump file. The dump file (which contains masked data) is then imported into the target database. Testers can use the same database, which now includes masked data, for testing.
  Note: For mainframes, use transformation maps to mask the data.

  NOTE

  To use HASHLOV consistent masking with MS SQL Server, download the CA TDM scramble database for MS SQL Server from the repository installation kit. Drop the existing scramble database for MS SQL Server. For more information about data scrambling, see Data Scrambling.

Additionally, you can follow another approach where you convert your transformation map into a CSV file and use that CSV file in Fast Data Masker to do the masking. For more information, see Use Transformation Map Files.

Fast Data Masker System Requirements

The Fast Data Masker component has the following system requirements. If you install Fast Data Masker with other components during the Server installation, ensure that your system meets the requirements.

Operating System

RHEL 6.0, RHEL 7.0, and Windows 7 (or higher) are supported.

Java

Fast Data Masker requires Java Runtime Environment (JRE) version 1.8.

Note: If there is a problem with Java, Fast Data Masker contains a /jre folder that contains a suitable Java version.

Masking Rules and Files

- All masking rules are stored in comma-separated text files.
- Fast Data Masker contains a standard set of seed data files that you can easily add and amend.

Supported Data Sources

For the complete list of supported data sources (for example, database types, mainframe platforms, or file formats) for masking, see Supported Data Sources.

Additionally, you can review the following points:

- For DB2 on z/OS, ensure that the db2jcc4.jar, db2jcc4_license_cu.jar, and db2jcc4_license_cisuz.jar files are available in the ..\Grid-Tools\FastDataMasker\lib folder.
  Note: The default location where Fast Data Masker is installed is C:\Program Files\Grid-Tools\FastDataMasker.
- For DB2 on iSeries (AS400), ensure that the jt400.jar file is available in the ..\Grid-Tools\FastDataMasker\lib folder.
- The following file formats are not supported:
  - HIPAA 40-10, 50-10, X12
  - EDI Files
  - SWIFT
- For Netezza, the following points are applicable:
  - Ensure that you can create external tables; that is, create external table privileges for the login user.
  - Increase the commit size to at least the size of the largest table to be masked.
  - PARALLEL option is not supported for Netezza, as only one external table can be created at a time.
  - If PTS is being used, this has to be run on the Primary.
  - NPS client needs 1 GB of memory per million rows being masked; this is an approximate estimate. The memory requirement depends on the size of the table to be masked (row count) and the number of columns being masked.
  - Copy the nzjdbc.jar file to the ..\Grid-Tools\FastDataMasker\lib folder.

Minimum Resolution

The minimum resolution for the Fast Data Masker UI must be greater than 1024 x 768 pixels.

Maximum Memory Parameter

When you mask large XML files, set the maximum memory (JVM heap size) to eight times more than the XML file. For example, if the XML file is 1 GB, set the maximum memory to 8 GB.

The memory is required primarily to hold the DOM tree structure of the XML document.

If the **namespace aware** is on, set the maximum memory to ten times the size of the XML file.

To set the maximum memory in the FDM Mapper UI, update the **Max Memory (MB)** field at the bottom of the **Summary** tab.

To use the FastDatamasker_flat.jar directly, set the maximum memory in the -Xmx parameter. For example:

```
java -Xms100M -Xmx8000M -jar Fastdatamasker_flat.jar connect_xml.txt xpath.csv
```

Namespaces

By default, XML masking is not namespace aware. The XPath query is used literally with namespace prefixes that are not interpreted. For example, an XPath */bookstore/bk:book/ti:title* looks under *bookstore* nodes for nodes with the name *ti_title*.

XML documents might have parallel or nested namespace declarations where prefixes are not uniquely mapped to namespace URLs. For example:

- The same prefix that is used for more than one namespace
- Different prefixes are used for the same namespace

In these cases, turn on namespace aware. This setting lets your XPath queries select the correct elements to mask in the XML document.

To turn on namespace aware, set the XMLNAMESPACEAWARE options parameter to Y.

Next, provide a mapping of prefixes to use in your XPath queries, to namespaces. Define the prefixes in the XPATHNAMESPACEMAP options parameter.

This parameter has the following format:

```
prefix1=url1; prefix2=url2; ...
```

Example:

```
bk=urn:xmlns:25hoursaday-com:bookstore; inv=urn:xmlns:25hoursaday-com:inventory-tracking
```

The prefixes in the XPath queries might be different from the prefixes that are used in the XML document. You can select the prefixes to use in the XPATH, but ensure that you specify the prefixes in XPATHNAMESPACEMAP.

Note: The prefixes in your document might not map uniquely to namespace URLs.

To scan large XML documents to find the namespace URLs to use in your XPath, run an initial mask run. Use the following values:

- **XMLNAMESPACEAWARE =Y**
- **Empty XPATHNAMESPACEMAP**
- **XPath**
  Example: /a/b

FastDatamasker parses your XML file and lists the namespace declarations. For example:

Namespaces declared in the XML document:

```
=urn:xmlns:25hoursaday-com:bookstore
bk=urn:xmlns:25hoursaday-com:bookstore
bka=urn:xmlns:25hoursaday-com:bookstore
bk=urn:xmlns:26hoursaday-com:bookstore
inv=urn:xmlns:25hoursaday-com:inventory-tracking
```

This example shows that the document has three declarations to the namespace URL `urn:xmlns:25hoursaday-com:bookstore` . The first declaration is the default namespace with an empty prefix. The prefix *bk* is used for two different namespace URLs.

NOTE

For more Information:

- https://msdn.microsoft.com/en-us/library/aa468565.aspx
- https://msdn.microsoft.com/en-us/library/ms950779.aspx
- http://books.xmlschemata.org/relaxng/relax-CHP-11-SECT-1.html

Default Namespace Example

Consider the following XML file:

```
<table xmlns="http://www.w3.org/TR/html4/">
 <tr>
 <td>Computers</td>
 </tr>
</table>
```

In this example, the XPath to modify `Computers` is as follows:

```
/DEFAULT:table/DEFAULT:tr/DEFAULT:td
```

Therefore, for the default namespace, update the XPath for your XML files as shown above. The DEFAULT automatically maps to the default namespace.

Copy `sqljdbc_auth.dll` to Appropriate Locations for Microsoft SQL Server (Integrated Authentication).

If integrated authentication is enabled on the Microsoft SQL Server database, ensure that you copy the required `sqljdbc_auth.dll` file to the appropriate location.

If you are using the Fast Data Masker Mapper, copy the file as follows:

- For 32-bit, copy `sqljdbc_auth.dll` from `C:\Program Files\Grid-Tools\FastDataMasker \SQLSERVER_DLLs\x86` to `C:\Program Files\Grid-Tools\FastDataMasker` (where Fastdatamasker.jar exists).
- For 64-bit, copy `sqljdbc_auth.dll` from `C:\Program Files\Grid-Tools\FastDataMasker \SQLSERVER_DLLs\x64` to `C:\Program Files\Grid-Tools\FastDataMasker` (where Fastdatamasker.jar exists).

Note: The default location where Fast Data Masker is installed is `C:\Program Files\Grid-Tools \FastDataMasker`.

If you are using the batch file that was saved from the Fast Data Masker Mapper, copy the file as follows:

- For 32-bit, copy `sqljdbc_auth.dll` from `C:\Program Files\Grid-Tools\FastDataMasker \SQLSERVER_DLLs\x86` to `%APPDATA%\Grid-Tools\FastDataMasker` (where the batch file exists).
- For 64-bit, copy sqljdbc_auth.dll from `C:\Program Files\Grid-Tools\FastDataMasker\SQLSERVER_DLLs \x64` to `%APPDATA%\Grid-Tools\FastDataMasker` (where the batch file exists).

Limitations

Review the following limitations:

- The use of Fast Data Manager on databases to which the user is connected via a VPN, can cause a 'Socket Error' / 'Socket write error' / 'Socket read timed out' error. We recommend the use of Fast Data Manager only with a local connection to the database.
- The XML parser only supports simple DOCTYPE declarations with no nested groups. The following expressions are not supported in XPATH queries:

  - filter (select left from right)
  - variables
  - number()
  - text()
  - string()
  - name()
  - local-name()
- In the case of XML file masking, if your XML has a large number of nodes (for example, more than 10,000), the XPATH segments are not displayed in the **Add XPATH to mask** drop-down list in the **Masking** tab. You must manually enter the XPATH in **Add XPATH to Mask** and click the **Add** button to proceed with the masking.

Fast Data Masker Best Practices

This article focuses on the best practices that are used to mask tables. Use the Fast Data Masker functions to mask tables, seed tables, work with columns, and for cross-reference mapping.

These best practices make the Fast Data Masker tasks easier to understand and to modify.

Manage Seed Tables

The seedtables subdirectory contains seed data in editable text files. The subdirectory is located in the install directory; the following line is an example of the directory path.

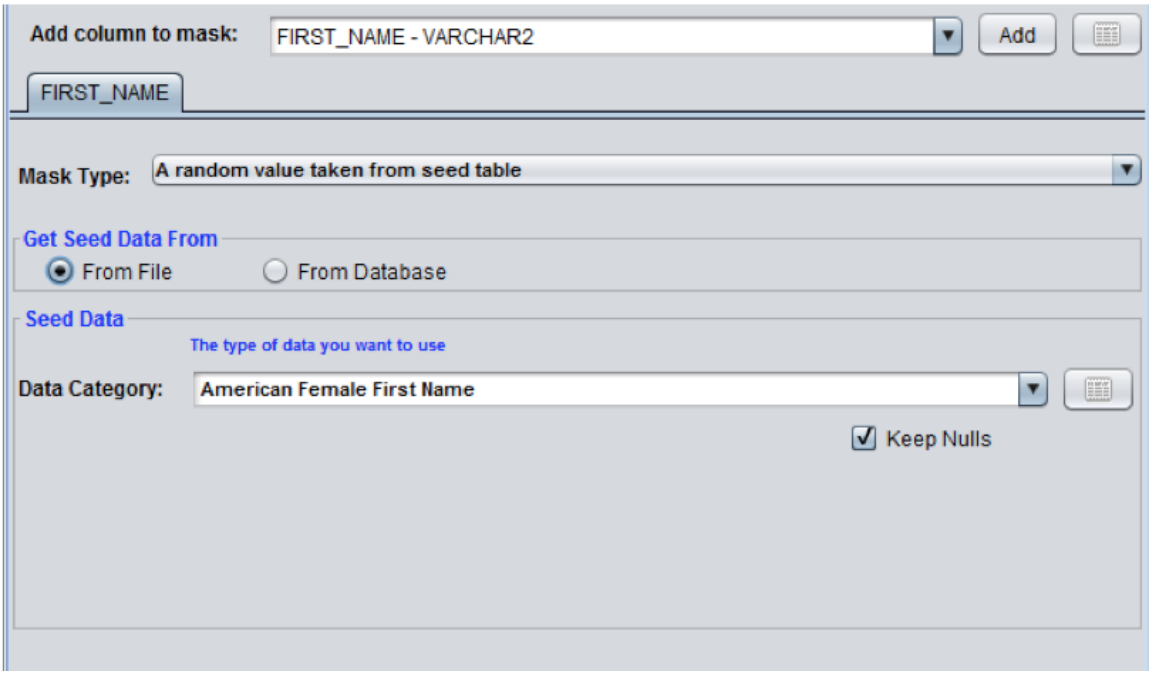
```
C:\Program Files\Grid-Tools\FastDataMasker
```

Each file contains a list of values that you can use to mask the columns.

The following text file shows the contents of the sample seed file *femalenames.txt*:

```
Amanda Amanda Angela Anita Ann Anna Anne Anne Ashley Beate Betty Beverly Bonnie Carla Carla Carol Carol
 Carolyn Cary Casey Cathy Chris Christina Christine Christine Christine Cindy Cindy Claire Clarise Collette
 Connie Cory Courtney Darsha Edith
```

To use the seed file to mask database columns, enter the seed file name with the appropriate masking function. The following example shows how to mask the column, FIRST_NAME of the table PEOPLE:



Split Tables

The Fast Data Masker component uses parallel processes to ensure the highest possible performance. To do so, determine the amount of work that is required to allocate to each parallel thread.

To automatically assign each table its own thread when you mask multiple tables in one masking run, set the **PARALLEL=** option in the **Options** tab. Similarly, when you mask a partitioned table in Oracle, the partitions in the table are automatically assigned to separate threads.

To mask large tables, split the row to be masked for a given thread. The simplest way to split the table is to use the SQL WHERE clauses. When you click on a table in the first tab and center list more than once, the **Add Masking** dialog opens.

Click **Yes**, and multiple tabs are provided for the same table. In this way, you can apply the same mask multiple times but with different WHERE clauses. Fast Data Masker can split the mask work into logical chunks, and can assign a thread for each.

Note: For more information, see Use Parallel Threads to Mask Data. Additionally, when masking Microsoft SQL Server tables, use the PARALLEL option only when the table has a primary key or unique index. If the table does not have a primary key or unique index and you use the PARALLEL option, masking is either slow or does not work.

The following table shows an example that is displayed on the **Summary** tab after the split:

Table	Column	Function	Parm1
PEOPLE		WHERE	where id < 10000
PEOPLE	FIRST_NAME	RANDLOV	firstnamefemaleamerican.txt
PEOPLE		WHERE	where id > 10000
PEOPLE	FIRST_NAME	RANDLOV	firstnamefemaleamerican.txt

Cross-Reference Tables

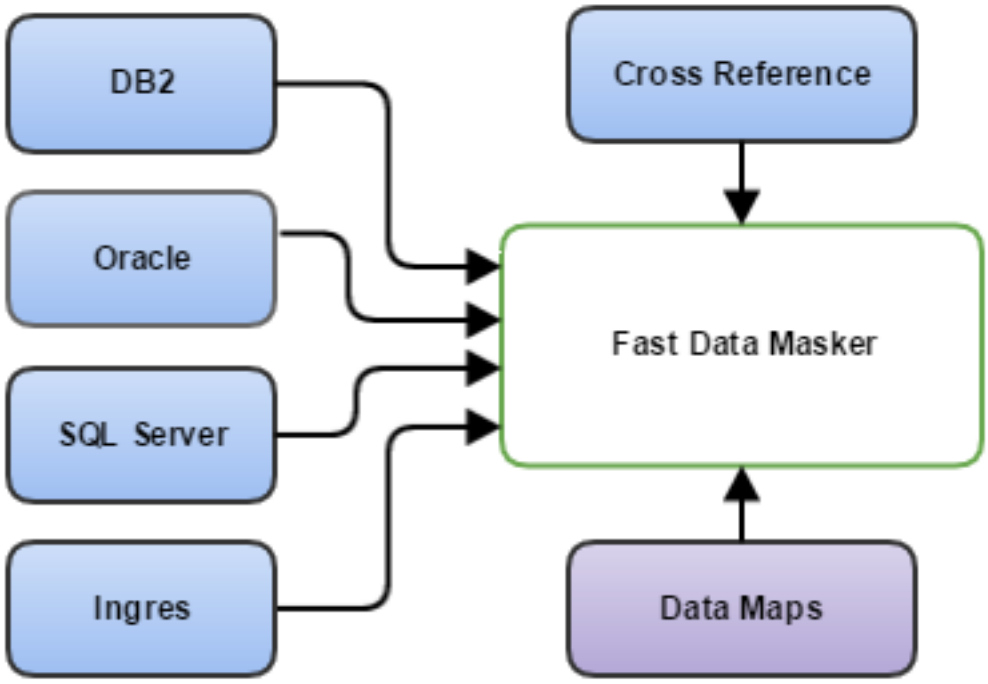
To use the cross-reference functionality, create a table in one of your connections that has the following structure:

```
CREATE TABLE gtsrc_xref (rx_ref_id varchar(254) NOT NULL,
rx_old_value varchar(254) NOT NULL,
rx_new_value varchar(254) );

ALTER TABLE gtsrc_xref
ADD CONSTRAINT gtsrc_xref_pk PRIMARY KEY (
rx_ref_id ,
rx_old_value );
```

If you clear the table down, the cross-reference that is mapped is rebuilt during the next mask run.

Figure 27: Shows the colum masking dialog



Note: Using cross-reference to mask large volumes is time consuming. Each row that is to be masked has to perform a lookup in the cross-reference table. To optimize this functionality, see the FASTXREF scramble option. To maintain consistency across different tables and databases when you use seed values to mask, we recommend that you use the HASHLOV function without cross-reference. The HASHLOV function consistently converts an existing value to an integer value in a given range. For example, if you mask a person LAST_NAME, and use the seed data category LASTNAME (which has 2000 entries), the existing LAST_NAME value is assigned a number from 1 through 2000. This value is used to look up the corresponding value in the seed list. For example, the LAST_NAME Jones is always masked to value Burton.

Add WHERE Clause to a Table

The WHERE clause allows you to restrict your masking to only certain rows in the table. This allows you to mask, for example, male names and female names differently based on the GENDER column.

As an example, consider a scenario where you want to mask the employee first names (FIRST_NAME) in the table EMPLOYEE for those rows where the value in the EMPLOYEE ID column is lesser than 5 (EMPLOYEE_ID<5).
To perform the mask in the table EMPLOYEE, you use the WHERE CONDITION field to specify the condition

(EMPLOYEE_ID<5), select the FIRST_NAME for masking, and choose the appropriate masking options. The following table shows the example:

Table	Column	Function	Parm1
EMPLOYEE		WHERE	EMPLOYEE_ID<5
EMPLOYEE	FIRST_NAME	RANDLOV	firstnamefemaleamerican.txt

The following table shows another example. In this example, all the HELP TEXT columns are masked with random values from companies.txt. The rows with an MSP_CODE starting with 'HSD' has the value huw assigned to CREATED BY. The rows with an MSP_CODE starting with 'OFG' has the value fred assigned to CREATED BY:

Table	Column	Function	Parm1
QMS_MESSAGE_TEXT	HELP TEXT	RANDLOV	companies.txt
QMS_MESSAGE_TEXT		WHERE	MSP_CODE LIKE 'HSD%'
QMS_MESSAGE_TEXT	CREATED BY	FIXED	huw
QMS_MESSAGE_TEXT		WHERE	MSP_CODE LIKE 'OFG%'
QMS_MESSAGE_TEXT	CREATED BY	FIXED	fred

Review the following points:

- To execute the WHERE clause, the WHERE condition for a map is required before subsequent masks for the same table. In other words, the WHERE clause applies to the mask for a table that follows in the CSV file. The other rows for the table specify the specific mask that is applied to the table.
- WHERE clauses are required to contain ANSI standard SQL. You cannot use special characters like ; or /.
- Fixed width records cannot have zero length. Therefore, WHERE COL = "" does not work because the column value is padded out to its fixed width size.
- Fast Data Masker currently supports the following operators: <,<=,=, =>, >, IN and LIKE (LIKE is not supported for flat files. It is only used for database masking).

Apply Multiple Functions to the Same Column

When you apply multiple functions to substrings in the same column, list the functions in consecutive rows in the mapping CSV; for example, as shown in the following table:

Table	Column	Function
PAYMENT_OPTION	ACCOUNT_NUMBER	SEQNUMBER
PAYMENT_OPTION	ACCOUNT_NUMBER	FIXED

No limit is set to the number of mask functions that are performed on each column. However, set KeepNulls the same for all the functions.

Note: All functions are required to have "*substr start*" and "*substr length*" set. Substr specifications for different functions can refer to the same character positions. Also, if you are using cross-references, they (substr specifications) can only be set for the last function used on each column as shown in the following example table:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls	DateFormat	Cross Reference	Overrice SQL	Unique Columns	Xpath Element	Substr start	Substr length
PAYMENT_OPTION	ACCOUNT_NUMBER	SEQNUMBER	10000001		N								7	10

PAYMENT_OPTION	ACCOUNT_NUMBER	FIXED	0		N								7	1

In the preceding example, the mapping CSV shows that a mask of column ACCOUNT_NUMBER in the table PAYMENT_OPTIONS with a zero padded sequence in positions 7 through 16 is expressed as follows:

caaccounts,numberx,SEQNUMBER,1000000001,,,,,,,,,7,10,

caaccounts,numberx,FIXED,0,,,,,,,,,7,1,

NOTE: SEQNUMBER returns a left-aligned non-padded number. The sequence is started at 1000000001 and the lead digit is then set to 0.

In the audit report, the function names are suffixed with start:length parameters and concatenated together. For example, "SEQNUMBER(7:10) FIXED(7:1)".

If you specify cross-referencing, the value that is used to look up and update the cross-reference table is a substring of the value, not the whole column value.

Manage Primary and Foreign Keys

To mask primary keys, unique indexes, or columns that match with foreign keys, you might have to drop them before you mask. You can reapply them once the mask is complete.

If you have the DBUPDATES=P option when you run the mask, no masking takes place. But, Fast Data Masker checks if any FK constraints or triggers exist for the tables to be masked. Pre- and post-step scripts are produced to disable or drop the constraints and re-enable or create them after the mask.

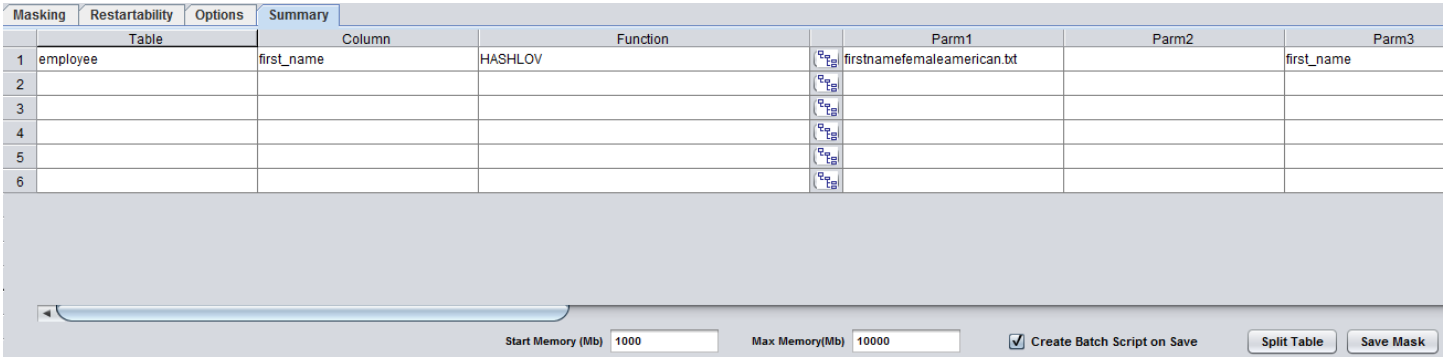
WARNING
It is easy to create duplicate values when you mask primary key columns. For example, you have ID values of 2,5,7,8,100, and apply a SEQNUMBER masking function starting at 100, the value 2 is updated to the value 100 which creates a duplicate.

If you have any questions before you begin, contact CA Support.

Manage Large Tables and Seed Tables

To mask large or multiple tables and seed tables, we recommend that you increase the memory that is allocated to the Fast Data Masker executable.

To increase the allocated memory, specify the appropriate values in the **Start Memory (Mb)** and **Max Memory (Mb)** fields in the **Summary** tab while defining the masking. When you save the defined masking information, Fast Data Masker creates a batch file. This batch file includes the values that you specify for the memory in the **Summary** tab. The batch file also includes other required information (for example, location of the connection file, masking file, and options file). The following screenshot shows the increased memory values:



The default values are 100 MB and 1000 MB. In this screenshot, the values are increased to 1000 MB and 10000 MB.

The following snippet shows the example content included in the batch file that is generated after you save the masking. Review that the parameters `Xms1000M` and `Xmx10000M` represent the increased memory values:

```
java -Djava.util.logging.config.file="C:/Program Files/Grid-Tools/FastDataMasker/
logging.properties" -Xms1000M -Xmx10000M -jar "C:/Program Files/Grid-Tools/FastDataMasker/
Fastdatamasker.jar" "C:/Program Files/Grid-Tools/FastDataMasker/doc/connectSQLSERVER.txt"
"C:\Users\<username>\AppData\Roaming\Grid-Tools\Fastdatamasker\MyMask.csv"
```

When masking a very large table with a primary key or a unique index, you can improve performance by using the following options. These options apply only to tables in Oracle and SQL Server databases.

- **LARGETABLESPLITENABLED**
  Enables large tables processing. Set this parameter to Y to enable, and to N to disable.
  Default: N
- **LARGETABLESPLITSIZE**
  Defines the minimal number of rows for FastDataMasker to start using large table processing.
  Default: 1000000

With this setting, FastDataMasker processes large tables by generating several blocks, with each block containing LARGETABLESPLITSIZE rows to be processed.

The existing option PARALLEL defines the number of threads that can run concurrently to process the blocks. If the PARALLEL option is not set, and you enable LARGETABLESPLITENABLED, then PARALLEL is set to 10 by default. If there are more blocks than threads, then remaining blocks are queued for processing and wait for a thread to become available.

Write Logs to a Local Drive

As a best practice, we recommend that you write the logs to a local drive, not a network drive. Audit/application logs on remote drives significantly affect the masking performance.

Configure the Fast Data Masker Logs Location

By default, all Fast Data Masker logs are located in %AppData%\Grid-Tools\FastDataMasker\Logs (for example, C:\Users\<*user_name*>\AppData\Roaming\Grid-Tools\FastDataMasker\logs). If you are unable to use this folder and want to change the location where Fast Data Masker logs are stored, you can do so in the Fast Data Masker Mapper.

Follow these steps:

1. Click **Start, All Programs, FastDataMasker, FastDataMasker** to open the Fast Data Masker Mapper.
2. Click the **Options** tab.
3. Locate the row that contains the LOGDIR option.
4. In the **Value** column, enter the file path where you want to save the the Fast Data Masker log files.

5. Complete your masking rules and run a masking job.
6. Access the file path that you specified in the LOGDIR option.
   Your log files are now available in this location.

Hash on the Value of a different XML tag when using HASHLOV for XML files

When masking XML files using XPATH elements, you may want to define a custom hash column so you can hash on a different XML element than the default, which is to hash on the current value being masked. For example, when you use an XML input file to update information of a pre-existing user in the system, the masked values in the file must match those in the database. If you use a member ID for hashing in the database which is also in the XML file, you want to use that ID as the hash column.

You can customize the **Relative XPATH to hash on** at the step where you configure seed data for HASHLOV. Enter a tag name using XPATH syntax, for example, `/preceding-sibling::id`.

Use Parallel Threads to Mask Data

Fast Data Masker lets you use parallel threads to mask large tables. The PARALLEL option enables you to run *n* concurrent threads.

To apply parallel threads, Fast Data Masker must split the work to be done into separate chunks. Fast Data Masker can manage this task in one of the following ways:

- A regular mask of multiple tables where none of the tables has a large amount of data. Fast Data Masker automatically sets a thread for each table.
- A large table that is not partitioned. In this case, split the table using the *where* clauses. For this to work, the following must apply:
  - The masking CSV must only contain mask for a single table.
  - The *where* clauses must not overlap; for example, if two or more SQL Where clauses select the same rows in the table to be masked, then the mask causes row lock errors.
- For an Oracle-partitioned table, Fast Data Masker automatically assigns a thread to each underlying partition. This cannot be combined with the *where* clauses, and as in point two above, it is applicable to a mask for a single large table.

Note: The number of parallel threads that you can execute concurrently is constrained by the number of physical cores and/or processors available. If the parallel number specified in the options is greater than the number of cores, then some of the threads are held in a queue until resources become available.

To split a table with the *where* clause, use Fast Data Masker as follows:

1. Access the Fast Data Masker UI.
2. Use the required connection file to connect to the database.
   Note: When masking Microsoft SQL Server tables, use the PARALLEL option only when the table has a primary key or unique index. If the table does not have a primary key or unique index and you use the PARALLEL option, masking is either slow or does not work.
3. Select the **Masking** tab and perform the following steps:
   a. Select the table (for example, **PERSONS**) in which you want to mask the data.
   b. Select the columns (for example, **FIRST_NAME** and **LAST_NAME**) that you want to mask.
   c. Provide appropriate information (for example, **HASHLOV** masking type, **FULL NAME** data category, and so on) in the relevant fields.
4. Select the **Options** tab and specify the number of threads (for example, 4) for the **PARALLEL** option.

5. Select the **Summary** tab and review the information. The following table shows the example information that is displayed before you split the table:

Table	Column	Function	Parm1	Parm2
PERSONS	FIRST_NAME	HASHLOV	FULL NAME	3
PERSONS	LAST_NAME	HASHLOV	FULL NAME	4

6. Click the **Split Table** button.
   The **Split Table** dialog opens.
7. Enter the following information and click **OK**:
   a. Enter the number of threads; for example, 4.
   b. Select the numeric column (for example, **PERSON_ID**) that you want to use for the split.
      We recommend that you choose an indexed column or a column that is a primary key. That is, the column must contains unique values.
8. Review the updated information in the **Summary** tab. The following table shows the example information after you split the table:

Table	Column	Function	Parm1	Parm2
PERSONS		WHERE	PERSON_ID < 153	
PERSONS	FIRST_NAME	HASHLOV	FULL NAME	3
PERSONS	LAST_NAME	HASHLOV	FULL NAME	4
PERSONS		WHERE	PERSON_ID BETWEEN 153 AND 205	
PERSONS	FIRST_NAME	HASHLOV	FULL NAME	3
PERSONS	LAST_NAME	HASHLOV	FULL NAME	4
PERSONS		WHERE	PERSON_ID BETWEEN 206 AND 258	
PERSONS	FIRST_NAME	HASHLOV	FULL NAME	3
PERSONS	LAST_NAME	HASHLOV	FULL NAME	4
PERSONS		WHERE	PERSON_ID BETWEEN 259 AND 313	
PERSONS	FIRST_NAME	HASHLOV	FULL NAME	3
PERSONS	LAST_NAME	HASHLOV	FULL NAME	4

9. Click the **Save & Run Mask** button.
10. Enter the name for the masking file that includes all the mapping information and click **Save**.
11. Enter the name for the options file that includes the selected options information and click **Save**.
12. Click **OK**.
    A dialog opens that displays the masking progress.
13. Review the information and save or close the dialog when masking is done.

You have successfully used parallel threads to mask the data.

Run Fast Data Masker Scripts Remotely

When you save the defined masking information in Fast Data Masker, Fast Data Masker creates a batch file. This batch file includes information about the location of the connection file, masking file, options file, and other related information (for example, start memory). You can use this batch file to run from a remote location where Fast Data Masker is not installed. This is helpful in scenarios where you are facing performance issues on your server because of different components installed on the server. And, to improve the performance, you want to run the batch script from a different server.

To run this batch file from a remote location, edit the batch file and update the paths to the Fast Data Masker .jar, connection file, masking file, and options file. Ensure that they all point to valid locations. You can then run the file from that remote location.

The following example snippet shows the contents of a masking batch file; update the required file locations based on your requirement:

```
java -Djava.util.logging.config.file="C:/Program Files/Grid-Tools/FastDataMasker/
logging.properties" -Xms1000M -Xmx10000M -jar "C:/Program Files/Grid-Tools/FastDataMasker/
Fastdatamasker.jar" "C:/Program Files/Grid-Tools/FastDataMasker/doc/connectSQLSERVER.txt"
"C:\Users\<username>\AppData\Roaming\Grid-Tools\Fastdatamasker\MyMask.csv" "C:\Users
\<username>\AppData\Roaming\Grid-Tools\Fastdatamasker\MyMask_options.txt"
```

In this snippet, you can find the following example locations:

- C:/Program Files/Grid-Tools/FastDataMasker/Fastdatamasker.jar shows the location of the Fast Data Masker .jar file.
- C:/Program Files/Grid-Tools/FastDataMasker/doc/connectSQLSERVER.txt shows the location of the Microsoft SQL Server connection file.
- C:\Users\<username>\AppData\Roaming\Grid-Tools\Fastdatamasker\MyMask.csv shows the location of the file that contains masking information.
- C:\Users\<username>\AppData\Roaming\Grid-Tools\Fastdatamasker\MyMask_options.txt shows the location of the file that contains the applied options information.

Mask Stored Data

As a test data engineer, use the Fast Data Masker UI to access, set up, save, and run the masking options for the data stored in different data sources—relational (for example, Oracle and Microsoft SQL Server) and flat files (for example, fixed-width and JSON files).

The process that you follow in Fast Data Masker to mask the data is as follows:

1. **Input:** Connect Fast Data Masker to the data source.
   To get started with the masking process, you must first connect your Fast Data Masker instance to the data source that contains the data you want to mask. You establish this connection with the help of a connection file. This connection file includes all the relevant information about the data source. You create this connection in the Fast Data Masker UI. After you create this file, you can use it to connect to the data source whenever you want.
2. **Rule Definition:** Define masking rules.
   You define all the masking rules by using various available operations (for example, masking functions and options) in the Fast Data Masker UI. After you define the rules, you run the masking job to mask the stored data in the data source.
3. **Output:** Verify the masked data.
   You access the data source and verify the output; that is, the masked data. Ensure that you note the pre-masked data before you run the masking job so that you can verify it with the masked data after the masking job completes.

Note: This process is a generic process that is outlined here for easier understanding of the masking process in the context of Fast Data Masker. You might need to perform additional steps depending on your unique data source.

Mask Data Stored in Relational Databases

As a test data engineer, use the Fast Data Masker UI to access, set up, save, and run the masking options for the data stored in relational databases (for example, Oracle, Microsoft SQL Server). The process that you follow in Fast Data Masker to mask the data stored in a relational database is as follows:

1. **Input:** Connect Fast Data Masker to the data source.
2. **Rule Definition:** Define masking rules and run masking.
3. **Output:** Verify the masked data.

Note: This is a generic process. For some data sources, you might need to perform additional steps. For example, for iSeries (DB2AS400), see Work with Fast Data Masker in iSeries (DB2/400).

Use, Create, and Manage Connection Files

To get started with the Fast Data Masker Mapper, you use existing connection files (or create new connection files) to connect to the Fast Data Masker Mapper. Connection files include information about the type of database to which you want to connect. Fast Data Masker provides existing connection files in the form of text files, which follow the naming convention: connect<*filename*>.txt. You can also create your new connection files using predefined database types.

This article provides information about the following tasks:

Use Existing Connection Files to Connect to the Fast Data Masker Mapper

You can connect to the Fast Data Masker Mapper by using existing connection files.

1. Click **Start, All Programs, FastDataMasker, FastDataMasker** to launch Fast Data Masker.
   The Fast Data Masker connection dialog opens. This dialog contains a list of existing connection files in the left pane and corresponding fields in the right pane.
2. Click a connection file in the left pane.
   The fields in the right pane change depending on the connection file that you select.
   Note: You can define or change the directory you are working in by clicking **Set Directory**.
3. Verify the existing information that is auto-filled and enter or update the configuration details as appropriate.
   Note: The **Default Schema** field is case-sensitive. Additionally, the **Database Encoding** field is displayed when you try to connect to the Microsoft SQL Server or Sybase database. You can specify an appropriate encoding format based on what Microsoft SQL Server or Sybase supports. For example, for Microsoft SQL Server, you can specify the encoding format as UTF-8.
4. Click **Connect** to connect to the Fast Data Masker Mapper.
   The Fast Data Masker Mapper interface opens.

Create a New Connection File

You can create a new connection file if you do not want to use existing files.

1. Click **Start, All Programs, FastDataMasker, FastDataMasker** to launch Fast Data Masker.
2. Click **New**.
   A blank connection file opens in the right pane.
3. Enter a connection name and select a database type from the **DBMS** drop-down list. For more information, see Supported Data Sources.
   Appropriate fields are displayed depending on the database type that you select.
4. Enter the required information in the fields.
   Note: The **Default Schema** field is case-sensitive.
5. Click **Save,** and click **OK** in the **Save Connection File** dialog.
   The new connection file is saved.

Note: If the Microsoft SQL Server database has enabled *force encryption* for a specific database instance, you can use the **Force Encryption** option in Fast Data Masker. This option enables you to connect to such database instances in the encryption mode.

Manage Connection Files

You can perform various actions on the connection files. For example, you can copy a connection, delete a connection, and so on.

1. Click **Start, All Programs, FastDataMasker, FastDataMasker** to launch Fast Data Masker.
2. Perform the following tasks as appropriate:
   - **Copy Connection**
     Copies the connection file configuration details to create a new connection file with the same details:
     a. Click the connection file that you want to copy to create a new connection file.
        The connection file configuration details are displayed in the right pane.
     b. Click the **Copy Connection** button.
        The **Connection Name** field details are removed, allowing you specify a name for the connection file that you want to create after copying the existing file.
     c. Enter a new name.
     d. Review the remaining fields and click **Save** to save the changes.
        You have successfully created a new file by copying an existing file information.
   - **Delete Connection**
     Deletes the existing connection configuration details:
     a. Click the connection file that you want to delete.
        The connection file configuration details are displayed in the right pane.
     b. Review the connection configuration to verify that you want to delete the same.
     c. Click **Delete Connection**.
        The **Delete File** dialog opens.
     d. Click **Yes**.
        A message appears stating that you have successfully deleted a connection file.
     e. Click **OK**.
   - **Cancel Connection**
     Cancels the connection and closes the window:
     a. Click the **Cancel** button.
        Fast Data Masker cancels the connection and closes the interface.

More information:

- Supported Data Sources.

Define Masking Rules

The following tabs help you define masking rules for the data stored in a relational database:

- **Masking**
  This tab defines and creates masking rules for tables and columns that you want to mask.
  Note: For regular flat-file masking, the same column cannot be masked twice in a given masking map.
- **Restartability**
  If a masking job fails, use this tab to allow the process to resume from the point of the failure.

Note: This tab is used only for database masking.

- **Options**
  This tab lets you set important options for the masking process. These options include additional parameters to control the masking run, audit options, cross-referencing options, and seed table options.
- **Summary**
  This tab provides an overview of all of the rules and parameters that are defined for the mask and lets you save and run the mask.

After you access the Fast Data Masker Mapper interface, you use these tabs to complete the data masking process, which includes the following steps:

1. Select a table to mask.
2. Select a column to mask for the selected table.
3. Select the mask type depending on the categories.
4. Specify the option to resume the process at the fail point if the masking job fails.
5. Set options for the masking process.
6. Review the masking definitions, save the mask, or save and run the mask.

Note: Fast Data Masker does not support binary data masking for files. The RANDOMBLOB function provides binary data masking support only for the Oracle, SQL, and DB2 databases.

Select a Table to Mask

The first step in the masking process is to select a table that you want to use for masking.

Note: This step is not applicable for file-based masking. In file-based masking, you directly select a column that you want to mask.

1. Access the Fast Data Masker Mapper interface.
2. Click a table in the **Available Tables** field.
   Note: Use left-click/Shift to select multiple tables.
3. Use the forward arrow to move a table from the **Available Tables** field to the **Masked Tables** field.
   The selected table is moved to the **Masked Tables** field.
   Note: The **Available Tables** field lists tables that are available for masking. The **Masked Tables** field lists tables that you have selected for masking.

You have successfully selected a table that you want to use for masking. You can now select a column to mask.

Select a Column to Mask

After you select a table to mask, select a column to mask for the table.

1. Select the table that you moved to the **Masked Tables** field.
   A tab with the same name as the table name opens in the right pane.
2. Select a column that you want to mask from the **Add column to mask** drop-down list.
   Note: Use the **Where Condition** field to set a SQL condition. If you use this field, only rows for the provided condition are masked. Fast Data Masker currently supports the following operators: <,<=,=, =>, > and LIKE (LIKE is not supported for flat files. It is only used for database masking).
3. Click **Add**.
   Note: To select multiple columns to mask, click the Add Multiple Column icon, select multiple columns, and click **OK**. The column that you selected is displayed as a tab on the top right. The column that you are defining is highlighted in blue.
4. (Optional) To remove a column, right-click the tab for that column and select **Remove Column** from the context menu.

You have successfully selected a column that you want to mask for the selected table.

Note: For regular flat-file masking, the same column cannot be masked twice in a given masking map. You can now select the mask type.

Select the Mask Type

After you select the columns to mask, select the type of masking that you want to use.

1. (File-based masking) Select the data type that is applicable to the masking column from the **Data Type** drop-down list. Available data types are: Character, Numeric, and Date.
   The **Mask Type** drop-down list filters the list of masking functions and displays only those functions that are applicable to the selected data type.
   Note: For database-based masking, the **Data Type** drop-down list is automatically populated depending on the data type of the column in the database. Based on the pre-populated data type, the **Mask Type** drop-down list displays the list of filtered functions. In this case, the **Data Type** drop-down list is in the read-only mode. Therefore, you cannot change the data type.
2. Select the mask type that you want to use from the **Mask Type** drop-down list.
   The mask types are defined by the type of the column; for example, separate masking functions for character, numeric, and date types are available. The mask types on the list fall into the following main categories; each of which provides different options:
   - Hashed or random substitutions from a database seed table or a file containing seed data.
   - All other masking types not involving seed data substitutions.
3. Review the following information to view how you can use some of the masking types:
   a. For random seed data transformations, when you select the mask type that uses seed data, you use one of the following options to get the seed data:
      From File
      a. If you select **From File**, select the seed data category from the **Data Category** drop-down list.
         Fast Data Masker contains a large number of built-in seed data files to select from, or you can create your own custom seed data files. By default, these files are located in the seedtables subdirectory (<drive_name>\Program Files\Grid-Tools\FastDataMasker\seedtables).
         Note: You can also place your seed data files in any user-defined location on your system and access the files from that location. To access the seed data files from a custom location (for example, C:\My_Custom_Directory), provide the location of your custom directory in the **Seed File Directory** field on the **Set Default Save Directories** dialog. All the seed data files present in the custom directory now become available in the **Data Category** drop-down list. You can access the **Set Default Save Directories** dialog by selecting **Settings, Set Default Directories** from the main menu.
      b. Click the Preview Data icon to view the first 20 values contained in the seed data file.
      c. Enter appropriate information in other fields.

      From Database
      If you select **From Database**, define the connection file for your seed data database. Normally, this table is already provided to you and is available in an Oracle or MS SQL server database. The **Connect File** drop-down menu contains a list of available connections.
      a. Pick an item from this list (or create a new connection file by selecting the Create Connect File icon).
      b. After you select the correct connection file, click the forward arrow icon to connect to the database.
      c. Select the table that contains your seed data (normally table gtsrc_reference_data) from the **Seed Table** drop-down list.
         The **Data Category** drop-down list is populated with the appropriate associated values.
      d. Select from this list to substitute values of this data category.
      e. If you select a seed data category that contains multiple data elements, the **Link masking** dialog opens.
      f. To link data in this category to multiple columns in your table, click **Yes** and enter the appropriate data in the subsequent dialog.

This action fills in the correct column numbers for each column you are masking. This column number corresponds to the appropriate column in your seed data table so that you are substituting the correct type of data. For example, a street address or the corresponding city name or zip code.

g. Enter appropriate information in other fields as required.
Note: If your column includes the XML or JSON data, the **XML/JSON Data** field shows the XPATH (for XML) and JSON paths (for JSON). An example of the JSON path is `$[employee][firstname]` and of the XPATH is `//employee/firstname` .

b. For hashed seed data transformations, follow the same steps as for random seed data transformations in addition to the following information.
This type of transformation takes the existing value in the table to mask and hashes it to get a consistent integer value. This value is used to get the value from the seed data table. In this way, you can consistently mask, (for example) names or addresses in different tables within your database or even tables in different RDBMS. In addition to the values you fill in for random substitutions, for hashed substitutions, you also have the option to specify the column in your table you will be hashing on. This column should ideally be one containing unique values, and if masking multiple columns with the same seed data category, you should choose the same Hash Column for each column you are masking.

c. For numeric transformations that use range, you are prompted to typically enter a low and high value. You are also asked whether you want to Keep Nulls. The default setting is Yes.

d. For date transformations, you might be prompted to enter a parameter value. Additionally, you are also required to define the date format; the default is YYYYMMDD.

4. Verify your settings.

You have successfully specified the mask type. You can now specify the restartability option if you want.

Mask Substrings with Different Lengths and Values

For all masks, there is an option to mask only a portion of the original value. This option is typical for those that involve masking character columns.

To mask substrings that have different lengths and values, enter values for **Start at position** and **Number of characters to mask** in the **Extra Options** area. After you specify the start value, you have the flexibility of deciding whether to provide a value for the number of characters to mask; it is not a mandatory option. If your use case requires you to mask specific number of characters, you can enter the value in this field; otherwise, you can keep it empty. If you do not enter a value for the number of characters to mask, Fast Data Masker identifies the starting value and masks all the characters until the last character. If you specify the number of characters to mask, Fast Data Masker starts masking data from the specified start value and goes only up to the specified number of characters. Consider the following points when using these options:

- If the start at position value is greater than the length of the string and the number of characters to mask is not provided, masking is not done.
- Only blank or values greater than zero are allowed in the **Number of characters to mask** field. If you provide 0 or a negative integer as a value in this field, an error message is displayed.
- If the masking function generates fewer characters than the length of the string that you want to mask (length from the start index until the end of the string), blank spaces are used as a masking characters for the remaining length.
- If multiple masking functions are used on a single column, we recommend that you enter a value in the **Number of characters to mask** field. If you do not provide a value in the **Number of characters to mask** field, each masking function would mask the value from the start at position value until the end of the string. Therefore, some substrings might get masked by more than one function.
Consider a string value "Larissa" in a column for which you have defined two masking functions. The start at position values for the two functions are specified as 1 and 5, respectively. Also, the number of characters to mask is not specified for both the functions.
In this case, the first function starts masking from the position 1 (which is "L") until the end of the string (which is "a"), generating a masked value "sdfguyt" (for example). Now, the second function starts masking from the position 5 (which is now "u") until the end of the string (which is now "t"). The second function works on the already masked string "uyt"

instead of the original string "ssa". As a result, the original substring "ssa" is masked twice, which is not the intended masking purpose in this case. Therefore, we recommend that you provide the number of characters to mask if you use multiple functions on the same column.

Some examples of masking substrings with different lengths and values are as follows:

- If the specified substring does not exist in the column that you want to mask (the column value is too short), the column is padded with blanks and then masked.
  Example: If the varchar column contains "abc", and your substring starts from position 5 for 2 characters using function FIXED and value of 12, then the masked value is "abc12".
  Note: If the column is null and keepnulls=y applies, the column is not masked.
- If the masking value is bigger than the substring that is specified, the value is truncated.
  Example: If a column contains "abcdefghi" and you attempt to mask from position 2 for 2 characters using ("FIXED 123"), then the masked value is "a12defghi".
- If the masking value is smaller than the substring specified, the value is padded with blanks.
  Example: If a column contains "abcdefghi" and you mask from position 2 for 4 characters using FIXED ("1"), then the masked value is "a1 ghi".

Specify the Restartability Option

You can set the restartability option for the mask run. If a masking job fails, the restartability option allows the process to start from the same point where it failed. A column in the table (which you want to mask) is used to set a flag to indicate rows that have been masked. If no existing column is specified, Fast Data Masker attempts to add a character column (de_ident_ind) that includes NULL to use as a restart flag. The restart column is helpful in scenarios where you want to use it for audit purposes. When the original value is not replaced with the masked value at the time of masking, the restart column is set to NULL and an entry is made to the log file. You can analyze the log file and look for NULL in the restart column. This information helps you determine the values that were not masked. You can perform the necessary changes and restart masking. Fast Data Masker then automatically chooses these rows for masking during the next run.

Note: This option is used only for database masking. Also, ensure that the column that you want to use to restart the masking job includes the values as NULL, not any other values. You can also review the DROPRESTART masking option.

1. Click the **Restartability** tab.
2. Click in the **Restart Column** cell.
   A drop-down list containing all the columns for the masked table is displayed.
3. Select the appropriate column that you want to use for restart.
4. Select the **Mask Has Restartability** option at the bottom.

You have successfully specified the information to restart the masking job in case of a failure. You can now specify the masking options.

Specify Options for the Masking Process

Click the **Options** tab and specify values for the appropriate options that you want to use for the masking process.

To specify a value for an option, locate the option, double-click the cell under the **VALUE** column for that option, and specify the required values.

For more information about all the masking options, see the Masking Options section.

Note: You can also use the preview mode (simulation mode) to review the *before mask* and *after mask* values before you make the actual updates in the database. When you specify the options to use the preview mode and run the masking job, Fast Data Masker runs the masking process without making any actual database updates. Fast Data Masker saves all the information in a CSV file along with the original and masked values. You can view the CSV file and analyze how your masking is going to affect the data in the database. Based on your analysis, you can then take an informed decision. For

example, you can decide whether you want to proceed with the specified masking information or you want to further refine your masking data.

Save and Run the Mask

The **Summary** tab provides an overview of all of the rules and parameters that are defined for the mask.

1. Click the **Summary** tab.
2. To edit the details of each cell in the **Summary** tab, click the cell and edit as follows:
   - **Table**
     Indicates the table that contains the column you want to mask.
   - **Column**
     Indicates the column you want to mask.
   - **Function**
     Indicates the masking rule you defined.
   - **Param 1**
     Indicates the first parameter for the function.
   - **Param 2**
     Indicates the second parameter for the function.
   - **Param 3**
     Indicates the third parameter for the function.
   - **Param 4**
   - Indicates the fourth parameter for the function.
   - **Keep Nulls**
     Specifies whether to keep null values, or to replace null values with masked values.
     Values: Y (Retain Values), N (Replace Values)
     Default: Y
   - **DateFormat**
     Specifies the date format for the database character field dates.
   - **Cross Reference**
     Specifies the cross-reference details. For more information, see the information about cross-referencing in this documentation.
   - **Override SQL**
     Indicates an extra parameter that is used for the HASHLOV1 function.
   - **Unique Columns**
     Lets you use a comma-separated list of columns that provides uniqueness in a table.
     Note: Use this rule if your table has no unique or primary key.
   - **Xpath Element**
     Indicates the location of the XML or JSON data that you want to mask.
   - **Substr Start**
     Specifies the substring start value. For more information, see the information about masking substrings in this documentation.
   - **Substr Length**
     Specifies the substring length value. For more information, see the information about masking substrings in this documentation.
   - **Notes**
     Indicates notes added to the mask.
   - **Preformat**
     Specifies the date format before masking.
   - **Update**
     Allows you to not update this column if you use masked values.

Values: Y (update), No (do not update)

- **Used Masked Value**
  Uses masked values from previous columns.
  Values: Y (Use previous values), N (Do not use previous values)
- **Restart Column**
  Lets you use the column to set the restart flag.
  Note: Set this parameter to a valid column name where the column is nullable and does not contain data.

3. To save your masking definitions, click the **Save Mask** button.
4. To run the mask directly from this dialog after saving the masking definition, click the **Save & Run Mask** button.

You have successfully saved and run your masking definitions.

Note: If you want to use a saved masking definition, you can open that masking definition by clicking the **File, Open Saved Mask** option in the main menu. Alternatively, you can also open it by clicking the **Open Saved Mask** button on the **Masking** tab.

Verify the Masked Data

After you mask the data stored in a relational database, you must verify whether the output is correct; that is, data is masked correctly and as expected.

Follow these steps:

1. Access the database that stores the masked data.
2. Locate the tables that contain the columns where you masked the data.
3. Verify the masked values in the required columns.
4. Review that the masked values are masked as expected and are not the same as the original values.

Example: Mask Employee First Name, Last Name, and Email ID

This example use case provides information about how you can mask the first name, last name, and email ID of an employee by using Fast Data Masker. This example also helps you understand the overall flow and steps that you perform to complete a masking job.

The following table shows a snippet of the data that is available in the "employee" table in the database. You want to mask the first name, last name, and email ID in this database table:

employee_id	first_name	last_name	email
1001	John	Mathew	John.Mathew@xyz.com
1002	Jim	Parker	Jim.Parker@xyz.com
1003	Sophia	Ran	Sophia.Ran@xyz.com
1004	Wendi	Blake	Wendi.Blake@xyz.com

Note: The "employee" database table also includes other columns that are not shown in this example table.

1. Access the Fast Data Masker Mapper.
2. Use the appropriate connection file to connect to the database that includes the employee data you want to mask.
3. In the **Masking** tab, move the "employee" table from the **Available Tables** field to the **Masked Tables** field.
   The employee table includes data about the first name, last name, and email ID of employees.
4. Double-click the employee table that is moved to the **Masked Tables** field.
   The **employee** tab opens in the right pane.

5. Click the Add Multiple Columns icon (next to the **Add** button) to select the following database table columns that you want to mask:
   - **first_name**
     This column in the "employee" table includes the first names of all the employees.
   - **last_name**
     This column in the "employee" table includes the last names of all the employees.
   - **email**
     This column in the "employee" table includes the email IDs of all the employees.

   All the selected columns are displayed as tabs in the UI.
6. Click the **first_name** tab to specify the mask type that you want to use for this column:
   a. Select the **HASHLOV** masking function from the **Mask Type** drop-down list.
      The HASHLOV masking function hashes the current value to consistently pick a value from the seed list in this case.
   b. Select the **From File** option to get the seed data from a file.
      Values from a seed data file are used to mask the first names of employees.
   c. Select **First Names** from the **Data Category** drop-down list.
      The **Data Category** drop-down list specifies the type of data you want to use, which is first names in this case.
   d. Select **first_name** from the **Hash Column** drop-down list.
7. Click the **last _name** tab to specify the mask type that you want to use for this column:
   a. Select the **HASHLOV** masking function from the **Mask Type** drop-down list.
   b. Select the **From File** option to get the seed data from a file.
   c. Select **Last Names** from the **Data Category** drop-down list.
   d. Select **last_name** from the **Hash Column** drop-down list.
8. Click the **email** tab to specify the mask type that you want to use for this column:
   a. Select the **CONCAT** masking function from the **Mask Type** drop-down list.
      The CONCAT masking function concatenates values that you provide in the **Value or Column** fields.
   b. Enter the following values in the **Value or Column** fields; these fields represent the parameters for the CONCAT function:
      a. **first_name**
      b. **.** (a dot character)
      c. **last_name**
      d. **@testing.com**

      When all the parameters are concatenated, the final string becomes **first_name.last_name@testing.com**, which represents the employee email ID.
   c. Select the **Use Masked Values** option to ensure that the masked values are used to generate the masked email ID.
9. Click the **Summary** tab and perform the following steps:
   a. Review the summary of the complete masking job that you have created. For example, verify that you have used the correct database table, columns, masking functions, and parameter values for the mask operation.
   b. Click **Save & Run Mask**.
      You are prompted to save the masking information in the form of a CSV file and masking options information in the form of a text file.
      Note: All the masking information that is displayed in the **Summary** tab is saved as a CSV file and all the options that are selected in the **Options** tab are stored in a text file.
   c. Click **Save** to the save the information.
      The confirmation message appears.
   d. Click **OK**.
      The Fast Data Masker Mapper runs the masking job to initiate and complete the masking process.
      The **Mask Complete** message appears after the masking is done.

e. Click **OK**.
The data masking is complete.

10. Query your database to verify that the masked data is now available in the database.
For example, the following "employee" table now shows the masked first names, last names, and email IDs for the same employee IDs:

employee_id	first_name	last_name	email
1001	Stephan	Lai	Stephan.Lai@testing.com
1002	Fay	Van Damme	Fay.Van Damme@testing.com
1003	Brevin	Dice	Brevin.Dice@testing.com
1004	Regina	Oleveria	Regina.Oleveria@testing.com

You have successfully masked the first name, last name, and email ID of your employees in the database.

Work with Fast Data Masker in iSeries (DB2/400)

This article includes information about how you can work with Fast Data Masker in iSeries V7R1.

To work with Fast Data Masker in iSeries, follow this high-level process:

Understand Key Files Needed for Fast Data Masker

The following are the important files that you need for Fast Data Masker:

Windows Platform

- C:\Program Files\Grid-Tools\FastDataMasker
- C:\Program Files\Grid-Tools\FastDataMasker\Seedtables\[All the seed list text files]
- C:\Program Files\Grid-Tools\FastDataMasker\GTMapper.exe
- C:\Program Files\Grid-Tools\FastDataMasker\BuildMap.xls
- %AppData%\Roaming\Grid-Tools\FastDataMasker\Connectdb2400.txt
- %AppData%\Roaming\Grid-Tools\FastDataMasker\Map_db2400.csv
- %AppData%\Roaming\Grid-Tools\FastDataMasker\Options.txt

iSeries Platform [In Integrated File System]

- /FastDataMasker
- /FastDataMasker/SEEDTABLES/[All the seed list text files]
- /Fastdatamasker.jar
- /Connectdb2400.txt
- /Map_db2400.csv
- /Options.txt

Further information about these files is as follows:

- **Fastdatamasker.jar**
This is a Java JAR file that contains core masking logic and is useful in non-Windows environments.
- **GTMapper.exe**
This is a Windows executable file for the Fast Data Masker Mapper application. The Fast Data Masker Mapper is a quick way to connect to target database, design transformation maps, set masking options, and perform masking.
- **BuildMap.xls**

This is a spreadsheet that supports GTMapper.exe. The spreadsheet contains rules related to different types of masking functions supported by different data types.

- **Connectdb2400.txt**
  This is a text file that contains information about location, port, schema, license, and credentials of the target database. This file is important in both Windows and non-Windows environments.
- **Map_Db2400.csv**
  This is a CSV file that contains names of tables, columns, masking functions, and function parameters. This file is important in both windows and non-Windows environments. This file can be constructed manually in Notepad by advanced users. Regular users must take help of Windows software like Fast Data Masker Mapper application (GTMapper.exe) to design it. If you have access to the Datamaker UI, you can use the Datamaker UI to design the CSV file.
- **Options.txt**
  This is a text file that contains important masking options such as commit frequency. This file is important in both Windows and non-Windows environments.

Verify the JRE Setup

iSeries is capable of hosting multiple JVM/JREs at the same time. However, Fast Data Masker is certified only against JRE 1.7. iSeries users must not attempt to run Fast Data Masker on lower JRE versions as it may lead to errors.

Follow these steps before running Fast Data Masker for the first time on iSeries:

1. Ensure that JDK 1.7/J2SE 7 is installed on the iSeries computer by running the following command on iSeries computer (V7R1 only):
   `- DSPSFWRSC`
   Check that 5761JV1 is installed with *BASE+ option 14 (Java SE7 - 32 bit) on iSeries OS V7R1.

```
Resource

     ID     Option     Feature     Description

5761JV1     8          5108        J2SE 5.0 32 bit

5761JV1     11         5111        Java SE 6 32 bit

5761JV1     14         5114        Java SE 7 32 bit
```

2. Verify that JRE 1.7 is working fine on iSeries (V7R1) by running the following command on iSeries:
   `- java -version`
   If everything is fine, you can see the following output in QSHELL:

```
java -version

java version "1.7.0"

Java (TM) SE Runtime Environment (build pap3270_27sr2-20141101_01(SR2))

IBM J9 VM (build 2.7, JRE 1.7.0 OS/400 ppc-32 jvmap3270_27sr2-20141101_01 (JIT
 enabled, AOT enabled)
```

3. If JRE 1.7 is not the default JVM of the iSeries system, it should be set for the iSeries profile that is used to run Fast Data Masker, as follows:
   a. Create a directory for the iSeries profile in `/Home/<profile_name>` that is used to run Fast Data Masker. For example, if the profile name is `GRIDTOOLS` , then this directory must be present (or created).

```
Parameters or command

===> wrklnk '/HOME/Tools'
```

   b. In this directory, create the file `SystemDefault.properties` .

```
Opt     Object link             Type     Attribute

5       SystemDefault.prop >    STMF
```

   c. In this file, set the Java version as 1.7 by putting the following content:

```
....+....1....+....2....+....3....+....4....+...

 ***************Begining of Data***********

java.version=1.7

 ***************End of Data***************
```

Set Up the Fast Data Masker Directory and Files

Set up the Fast Data Masker directory and files on iSeries computer.

1. Use System i Navigator to create the following directories in IFS (Integrated File System):
   - `/FastDataMasker`
   - `/FastDataMasker/SEEDTABLES`
   - `/FastDataMasker/lib`
2. If you have sufficient privileges, you can create a read-write share on the `ROOT/FastDataMasker` folder. This share allows users to create mapped network drives to this folder from their Windows computers.
3. Use an FTP client, System i Navigator, or mapped network drive to transfer the following files from your Windows computer to iSeries:
   - `C:\Program Files\Grid-Tools\FastDataMasker\Seedtables\[All Files]` to `/FastDataMasker/SEEDTABLES/[ALL FILES]`
   - `C:\Program Files\Grid-Tools\FastDataMasker\Fastdatamasker.jar` to `/FastDataMasker/Fastdatamasker.jar`
   - `%AppData%\Roaming\Grid-Tools\FastDataMasker\Connectdb2400.txt` to `/FastDataMasker/Connectdb2400.txt`

Note: Ensure that the license key information is already available in the Connectdb2400.txt file.

- `%AppData%\Roaming\Grid-Tools\FastDataMasker\ConnectSCRAMBLE.txt` to `/FastDataMasker/Connect SCRAMBLE.txt`
- `%AppData%\Roaming\Grid-Tools\FastDataMasker\Options.txt` to `/FastDataMasker/Options.txt`
- `%AppData%\Roaming\Grid-Tools\FastDataMasker\<Trasformation_map>.csv` to `/FastDataMasker/<Trasformation_map>.csv`
- `C:\Program Files\Grid-Tools\FastDataMasker\lib\[All Files]` to `/FastDataMasker/lib/[All Files]`

  Note: Ensure that the `lib` folder includes the jt400.jar file.

4. Ensure that iSeries server mentioned in Connectdb2400.txt and ConnectSCRAMBLE.txt are the same as the one on which these files reside. This best practice ensures maximum performance during masking.

Create the License File (lic.dat)

You need to create a license file (lic.dat) to start using Fast Data Masker on iSeries.

1. Start QSHELL by typing the following command:
   `STRQSH`
2. In QSHELL, change the current directory to `/FastDataMasker` :
   `cd /FastDataMasker`
3. List the contents of the directory:
   `ls -l`
   The following is an example output snippet:

```
Mar 4 07:06 connectdb2400.txt

Mar 4 07:10 connectSCRAMBLE.txt



Mar 4 07:12 employee_test.csv

Mar 4 07:09 Fastdatamasker.jar



Mar 4 07:21 lib

Mar 4 07:10 OPTIONS.txt

Mar 4 07:09 seedtables
```

4. Run the following command to verify that `FastDataMasker.jar` is not corrupted and is running in correct Java version
   `java -jar Fastdatamasker.jar`
   The following is an example output snippet:

```
Fastdatamasker version: 4.5.0.7

Fastdatamasker build date - March 3 2016
```

```
PID:528@USI.CA.COM

Java version 1.7.0

OPERATING SYSTEM USER:

Usage:

For Masking:Fastdatamasker <connect file> <map file> <Optional:options file>

To Test Connection and produce candidate license key: Fastdatamasker <connect file>
```

5. Run the following command to generate the license file `lic.dat` in the `$HOME/.CA/TDM/lic.dat` location:
   `java –jar Fastdatamasker.jar connectdb2400.txt`

Run Data Masking Using Fast Data Masker

One of the simplest ways to run Fast Data Masker in iSeries is using QSHELL. In any Linux-like shell environment, the standard syntax for running Fast Data Masker is as follows:

Syntax: `java –jar Fastdatamasker.jar <connection_filename> <Transformation_CSVname> <options_filename>`

Example: `java –jar Fastdatamasker.jar connectdb2400.txt map_db2400.csv options.txt`

Before executing the above command, ensure the following verifications are done:

- Current directory in QSHELL is changed to `/FastDataMasker`
  - QSHELL Command: `cd /FastDataMasker`
- Connection file, for example, `connectdb2400.txt`
  - is present in the `/FastDataMasker` directory.
  - is pointing to the correct target iSeries environment and schema.
  - is using an encrypted password.
- Cross-reference connection file in the options.txt file
  - is present in the `/FastDataMasker` directory.
  - is pointing to the correct iSeries server and schema.
  - is using encrypted password.
- Options file
  - is present in the `/FastDataMasker` directory.
  - has an optimal commit frequency; for example, for a very large table, 10000 is a good commit frequency. For tables smaller than 10000 records, db2 for iSeries users may want to commit small frequencies; that is, between 100 to 1000.
  - has cross-reference connection and table name information correctly populated

After you have checked the above information, you are ready to run your masking process. When Fast Data Masker starts, you are presented with the following information:

```
Valid to: 01-01-2020

**********************

Checking license
```

```
Valid license key found

LICENSE DETAILS

License Type: FULL

Company Name: companyName

Site Id: siteId

Product Id: productId

Valid to: 01-01-2020

**********

url=jdbc:as400://usi:5000:usi;transaction isolation=none;date format=iso

attempting to connect
```

At the beginning of, during, and after the masking process, you can see this example output snippet:

```
starting masking at 2016.03.04 08:06:43.326 EST

Using commit frequency of: 10000

Masking will be as follows

**************************

table EMPLOYEE_VD and column FIRST_NAME will be masked by hashing the current

value and using this to consistently pick from a seed list
```

Mask Data Stored in Flat Files

You can use Fast Data Masker to mask the data that is stored in flat files. This article provides information about how you can mask the data that is stored in the following file types:

- Fixed-Width or Delimited Files
- XML Files
- JSON Files
- Excel Files

Process

The high-level generic process remains the same for flat file masking as for relational data sources. That is, you connect Fast Data Masker to the data source, define masking rules, and verify the output.

Perform the following tasks:

1. Create a Connection File.
2. Select Columns to Mask.
3. Select Applicable Masking Options.
4. Start Masking.
5. Verify the Masked Data.

After you complete the masking, a file that contains the masked data is created. Test data engineers (TDEs) can access this masked file to use the masked data.

Mask Data in Fixed-Width or Delimited Files

Fast Data Masker lets you mask the data in fixed-width and delimited files. The delimited file type includes character-, comma-, and tab-separated files.

Understand Fixed-Width and Delimited File Formats

Fast Data Masker expects these file types to follow a specific format. If a file does not adhere to the required format, Fast Data Masker cannot process it for masking.

Delimited File

An example of a delimited file that contains the sample data is as follows. This file is a character-separated file, which includes the pipe character (`|`) as a separator. This file includes three rows of data.

```
1|James|Lynn|39
2|John|Smith|50
3|Mary|Jane|30
```

This data file follows the structure that is defined in the associated definition file. An example of a definition file that is related to the delimited data file is as follows. The first row of the definition file gives general details about the file. The `DELIM` parameter in the definition file defines that the pipe character (`|`) is used as a separator in the data file. The file also defines the column names: `ID` , `FIRSTNAME` , `LASTNAME` , and `AGE` , which correspond to the data included in the data file. For example, the first row in the data file is related to the data definition file as `ID` =`1` , `FIRSTNAME` =`James` , `LASTNAME` =`Lynn` , `AGE` =`39` . Other rows also follow the same representation. The definition file does not define any header and trailer values.

```
HEADER=N,TRAILER=N,DELIM=|,DATEQUOTED=,CHARQUOTED=,NUMQUOTED=
ID
FIRSTNAME
LASTNAME
AGE
```

The following list includes information about the parameter names, description, and values used in the first row:

- HEADER
  Specifies the header record.
  Values:
  – Y, N, or a number
  – Y==1 and N==0
- TRAILER
  Specifies the trailer record.
  Values:

  - Y, N, or a number
  - Y==1 and N==0
- DELIM
  Specifies whether fields are delimited. If yes, specify the value.
  Values:
  - 'fixed', 'FIXED' – fixed width – no delimiters
  - 'comma', '<COMMA>' – comma delimited
  - 'tab', '<TAB>' – tab delimited
  - Any character (for example, pipe)
- DATEQUOTED
  Specifies whether dates are quoted.
  Values:
  - Y or N
- CHARQUOTED
  Specifies whether strings are quoted.
  Values:
  - Y or N
- NUMQUOTED
  Specifies whether numerics are quoted.
  Values:
  - Y or N

Fixed-Width File

An example of a fixed-width file that contains the data is as follows:

```
This is a test header
ARON                TAMMY
                                  19780513416670249



                                                    4712345J09-335      O412857641MELTON         TAYLOR
 J1581378??19981209
CRAB                MIGEL           6817 HAWTHORNE LN
  JOINT BASE LEWIS MCCWA98433122000198803044128977 65                    AUTOZONE INC DEPT 8070        PO BOX
 2198                                                                 MEMPHIS             TN381012 1980000010101

                                                  47037452011-002992??E756100189ABAD           GISELLE
 N4633141??20090904
```

This data file follows the structure that is defined in the associated definition file. An example of a definition file that is related to the fixed-width data file is as follows. This file defines the value `1` for `HEADER` and `FIXED` for `DELIM` (because it is a fixed-width file). Also, review that the file defines each column name with its specific width (`FNAME,20`).

For example, the first line (`This is a test header`) in the data file is not masked because the value for the `HEADER` parameter is set as `1` . This means that the first row in the data file is excluded from masking. Similarly, the name `ARON` in the data file corresponds to the `FNAME` column, and so on.

```
HEADER=1,TRAILER=N,DELIM=FIXED,DATEQUOTED=,CHARQUOTED=,NUMQUOTED=
FNAME,20
LNAME,15
ADDRESS1,75
CITY,20
STATE_ZIP,13
```

```
ID_1,36
S,2
ADDRESS_TITLE,30
POBOX,25
POBOX_2,50
CITY_2,20
STATE_ID_1,21
TITLE_2,35
ADDRESS2,25
ID_2,50
CITY_3,20
STATE_ZIP_2,13
ID_3,15
ID_4,20
ID_5_NAME,25
LNAME_2,13
ID_6,10
ID_7,8
```

Create a Connection File

Before you start masking your flat file, you must create a connection file. This connection file stores information about the data source type, file that contains the data, file that contains the definition for the data file.

After you create the connection file, the connection file is added to the list of connection files in Fast Data Masker. You can then select this file whenever you want to mask the data that is stored in the associated data file.

Follow these steps:

1. Click **Start, All Programs, FastDataMasker, FastDataMasker** to launch Fast Data Masker.
2. Click **New**.
   All applicable options open in the right pane.
3. Enter an appropriate name for the connection file in the **Connection Name** field.
4. Select **File** from the **DBMS** drop-down list.
5. Select the file (fixed width or delimited) that contains the data you want to mask in the **File Name** field.
   Note: Instead of a single data file if your data is spread across multiple files, specify the folder location in the **File Name** field and select the **All Files In Directory** option. Ensure that all data files are available in the same folder location.
6. Select the associated file that includes the definition for the data file, which you selected in the previous step.
   If you do not have the related definition file, follow these steps to create it:
   a. Click the Create Definition File icon.
   b. Select the file type (**Comma Separated**, **Tab Separated**, **Character Separated**, or **Fixed Width**) from the **File Type** drop-down list.
   c. Enter required values in the following fields:
      - **Separator character**
        (For character separated) Specifies the character that you want to use as a separator (for example, pipe) for the data.
      - **Header Lines**
        Specifies the number of lines (counted from the top) in the data file that you want to mark as header lines and exclude from masking.
      - **Trailer Lines**
        Specifies the number of lines (counted from the bottom) in the data file that you want to mark as trailer lines and exclude from masking.

d. Select whether you want to use quotes for the character, numeric, or date data.
e. For the delimited file, click **Parse File to Mask**.
f. Enter information in**Column Name**, **Width**, **Sample Data**, and **Date Format - add for date columns**, as applicable:
For the fixed-width file, the following snippet shows an example of how you add this information in the UI:

```
Column Name          Width          Date Format - add for date columns
FNAME                20
LNAME                15
ADDRESS1             75
...
...
ID_7                 8
```

For the delimited file, the following snippet shows an example of how you add this information in the UI:

```
Column Name          Date Format - add for date columns          Sample Data
ID
FIRSTNAME
LASTNAME
AGE
```

g. Click **OK**.
A message appears that specifies the location of the saved file.
h. Click **OK**.
The file is saved and is added to the **Definition File Name** field.
Note: If your data definition file is always stored in a fixed location, you can specify the file name in the **Definition File Name** field. You can then enter the folder location where the definition file is available in the **Defn File Directory** field.

7. Click **Save**.
8. Click **OK** in the **Save Connection File** dialog to confirm the save action.
9. Click **Connect** to connect Fast Data Masker to the associated file by using the created connection file.
The Fast Data Masker UI opens. You can now proceed with the remaining steps.

Select Columns to Mask

After you connect Fast Data Masker to your flat file, you can start adding columns that you want to mask.

Follow these steps:

1. In the Fast Data Masker UI, use the connection file to connect to the flat file.
2. Verify that the **File Mask** tab (under **Masking)** is selected.
3. Select a column that you want to mask from the **Add column to mask** drop-down list. This drop-down list is populated with all the columns that you defined in the data definition file.
4. Click **Add**.
The column that you selected is displayed as a tab.
5. Select the data type of the column from the **Data Type** drop-down list. Available data types are character, numeric, and date.
The **Mask Type** drop-down list filters the list of masking functions. This list displays only those functions that are applicable to the selected data type.
6. Select the mask type that you want to use from the **Mask Type** drop-down list.
7. Follow the remaining steps as described in the Select the Mask Type section in the Masking Functions and Parameters.
8. Repeat Step 3 through Step 7 for other columns that you want to mask.
You have successfully added appropriate masking functions to the columns that you want to mask.

Select Applicable Masking Options

After you select your columns for masking, you can select the required masking options in the **Options** tab. For more information, see the Specify Options for the Masking Process section in the Masking Options.

Note: Some of the masking options are not applicable for flat files, though they are displayed in the UI.

Start Masking

Use the **Summary** tab to review the masking information before you actually mask the data. For more information, see the Save and Run the Mask section in the Mask Data Stored in Relational Databases article.

When you run the masking, Fast Data Masker creates and saves the following files:

- **Masked data file:** A scramble file that includes the masked data. This file is created in the same location where your original data file is available.
  For example, if the name of the data file is EmpMask.txt, the masked data file is generated with the name EmpMask.txt.scramble.
- **Mapping file:** A .csv file that includes the complete information that you see in the **Summary** tab.
  For example, masking columns, masking functions, and masking options. For example, C:\Users\abc01\AppData\Roaming\Grid-Tools\FastDataMasker\EmpMask.csv.
- **Options file:** A .txt file that includes information about the masking options that you have used.
  For example, C:\Users\abc01\AppData\Roaming\Grid-Tools\FastDataMasker\EmpMask_options.txt.
- **Masking batch file:** A .bat file that includes information about the mapping file, options file, connection file, memory (start and maximum) allocation, Fast Data Masker JAR file (for a flat file), and log configuration file.
  For example, C:\Users\abc01\AppData\Roaming\Grid-Tools\FastDataMasker\EmpMask.bat.
- **Log file:** A .log file that includes the log information.
  For example, C:\Users\abc01\AppData\Roaming\Grid-Tools\FastDataMasker\EmpMask.log.

You have successfully masked the data in the fixed-width and delimited files.

Verify the Masked Data

After you mask the data stored in a flat file, you must verify whether the masked output is correct; that is, data is masked correctly and as expected. In the case of flat files, Fast Data Masker generates a new data file that contains the masked data.

Follow these steps:

1. Navigate to the location where the masked data file (for example, EmpMask.txt) is saved.
2. Open the file using an editor (for example, Notepad++).
3. Locate the columns that you used for masking.
4. Note the masked data in the identified columns.
5. Verify that the data is masked as expected and is not the same as in the original data file.

Mask Data in XML Files

The high-level steps to mask the data that is stored in XML files is the same as for the fixed-width and delimited files that are discussed in the preceding section. Therefore, this procedure does not include the detailed information and outlines only the required steps.

Follow these steps:

1. Launch Fast Data Masker.
2. Click **New**.
3. Enter the connection file name.

4. Select **XMLFILE** from the **DBMS** drop-down list.
5. Specify the location of the XML file in the **File Name** field.
6. Click **Connect**.
7. Verify that the **XML Mask** tab is selected under the **Masking** tab.
8. Select the XPATH corresponding to the data that you want to mask from the **Add XPATH to mask** drop-down list. This drop-down list is populated with all the XPATHs that are available in the XML file that you are masking. For example, /purchaseOrder/shipTo/name.
9. Click Add.
   The selected XPATH element is added as a tab.
10. Select the data type, mask type, and provide the required information.
11. (Optional) Click the **Options** tab and enter values for the masking options that you want to use.
12. Click the **Summary** tab and review the masking information.
13. Click **Save & Run Mask** to save the masking information and mask the data in the XML file.
    A message appears when the masking completes.
14. Verify that a file containing the masked data is created at the same location where your original data file is available. For example, Client.xml.scramble.xml. Also, verify that the mapping file, options file, batch file, and log file are created.

You have successfully masked the data in the XML file.

Mask Data in JSON Files

The high-level steps to mask the data that is stored in JSON files is the same as for the fixed-width and delimited files that are discussed in the preceding section. Therefore, this procedure does not include the detailed information and outlines only the required steps.

Follow these steps:

1. Launch Fast Data Masker.
2. Click **New**.
3. Enter the connection file name.
4. Select **JSONFILE** from the **DBMS** drop-down list.
5. Specify the location of the JSON file in the **File Name** field.
6. Click **Connect**.
7. Verify that the **JSON Mask** tab is selected under the **Masking** tab.
8. Select the JSON path corresponding to the data that you want to mask from the **Add JSON PATH to mask** drop-down list. This drop-down list is populated with all the JSON paths that are available in the JSON file that you are masking. For example, $['employee']['firstname'].
9. Click **Add**.
   The selected JSON path is added as a tab.
10. Select the data type, mask type, and provide the required information.
11. (Optional) Click the **Options** tab and enter values for the masking options that you want to use.
12. Click the **Summary** tab and review the masking information.
13. Click **Save & Run Mask** to save the masking information and mask the data in the JSON file.
    A message appears when the masking completes.
14. Verify that a file containing the masked data is created at the same location where your original data file is available. For example, employee.json.scramble.json. Also, verify that the mapping file, options file, batch file, and log file are created.

You have masked the data in the JSON file.

Mask Excel Files

The high-level steps to mask the data that is stored in Microsoft Excel spreadsheets is the same as for the fixed-width and delimited files that are discussed in the preceding section. Therefore, this procedure does not include the detailed information and outlines only the required steps.

Follow these steps:

1. Open Fast Data Masker to create a connection profile to the Excel file.
2. Specify the **Connection Name**, for example: PERSONS DATA
3. Specify EXCEL FILE as the **DBMS** type.
4. Define the **File Name** by browsing to the Excel file to connect to.
5. Define the **Definition File Directory** by browsing to the directory where to store the definition files.
6. Define the **Definition File Name** to create definition files for the worksheets.
   The File Definition dialog opens. The dialog shows one tab for each worksheet.
   a. Specify the **Rows with data before column header line**, if applicable. Enter the number of header rows to ignore.
      Note: Empty rows are ignored automatically, subtract them from the number of lines to ignore.
   b. Select the '**include in definition**' checkbox to retrieve the worksheet info.
      The tabs display the work sheet content.
   c. Define the format of any date data in the center column, for each worksheet. Double-click on a cell, enter the date format, then press return to commit the change.
      Example: `dd/mm/yyyy hh:mm`
   d. Click **Save** to save the file definitions.
7. Click **Connect** to connect to the file.
   A window opens. The window shows one tab for each worksheet.
8. Define the columns that you want to mask for each worksheet.
   Note: You do not need to provide a date format for date masking. You have already defined the date format in the definition file.
9. Run the mask to create a masked copy of the original Excel File.

You have masked the data in the Excel file.

Mask Data Stored in Hadoop

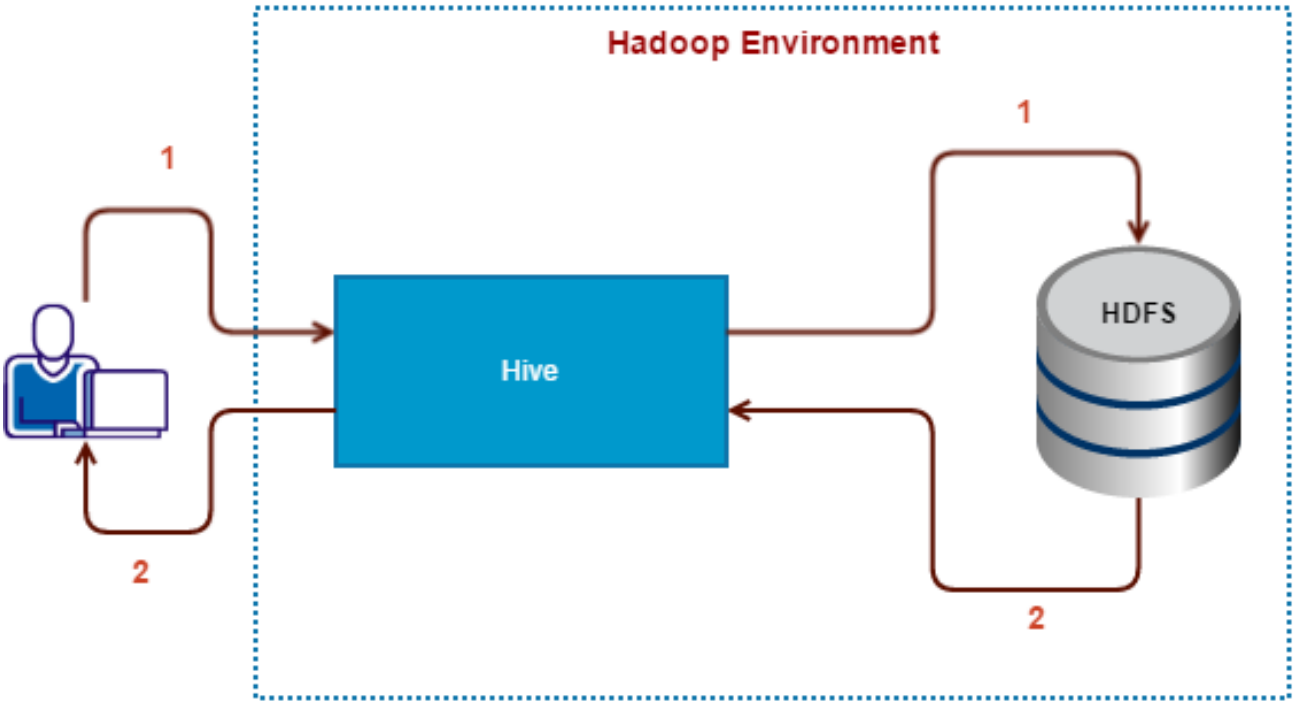
You can use Fast Data Masker masking functions as Hive user-defined functions (UDFs) to mask data stored in Hadoop. The stored data must be structured data and must have a defined schema.

CA TDM provides a JAR file that includes Hive UDFs, which are developed based on a standalone Java masking library. The Java masking library includes Fast Data Masker masking functions. When you execute these Hive UDFs (provided in the JAR file) in your Hadoop environment, they perform the defined masking operations and mask the structured data.

High-Level Architecture

The following illustration shows a simple representation of the interaction among different systems:

Figure 28: Hadoop_Hive_Masking



The details are as follows:

- The digit **1** in the diagram shows a user executing Hive UDFs provided in the JAR file through the Hive query language and accessing the structured data that is stored in Hadoop. These Hive UDFs include Fast Data Masker masking functions, which are provided in a masking library.
- The digit **2** in the diagram shows the updates that are made to the structured data in Hadoop as a result of executing Hive UDFs.

Mask Structured Data

The high-level process to mask structured data stored in Hadoop by using the provided JAR files includes the following steps:

1. Review the files in the masking package.
2. Review the supported masking functions.
3. Deploy the required JAR files and register provided Hive UDFs on the system where Hive is already present.
4. Execute the appropriate Hive UDFs using the Hive query language.

Review the Files in the Masking Package

CA TDM provides the following files for this masking use case. These files are included in a .zip file (MaskingSDK-<*version*>.zip), located in the root directory of your CA TDM installation media:

- **JAR files**
  The following JAR files are available in the package to help you mask the structured data that is stored in Hadoop:
  - **catdm-masker-hive-<*version*>.jar**
    This JAR file includes Hive UDFs. These Hive UDFs include Fast Data Masker masking functions. You can use the Hive UDFs in your Hive queries and can perform the masking operation.
  - **catdm-masker-library-<*version*>.jar**

This JAR file contains a library of all the supported Fast Data Masker masking functions. Hive UDFs in the catdm-masker-hive-<*version*>.jar file reference these masking functions.

– **commons-validator-<version>.jar**
This JAR file contains a library used by the HASHIBAN masking function.

- **The HQL file**
The **catdm-masker-init.hql** utility automates the following tasks in the Hive environment:
a. Adds the JAR files to the Hive session.
b. Adds defined Hive UDFs for all the supported Fast Data Masker masking functions to the Hive session.
- **seedtables-reference folder**
The seedtables-reference folder contains the seedtables which are packaged as part of maskingsdk which can be used in HASHLOV function.
- **ReadMe.txt**
This document explains the usage of seedtables and how to create custom seedtables for use with Hashlov, with examples.

Review the Supported Masking Functions

TIP
For Hive UDFs, the first parameter is the original value to be masked. Use the `desc function <function_name>` statement to get more information about the Hive UDF, as described in this article.

The following table shows Fast Data Masker masking functions and their corresponding Hive UDF equivalents:

Fast Data Masker Masking Function	Corresponding Hive UDF
ADD	tdm_add
ADDDAYS	tdm_adddays
ADDPERCENT	tdm_addpercent
ADDRANDOM	tdm_addrandom
ADDRANDOMDAYS	tdm_addrandomdays
ADDRANDOMHOURS	tdm_addrandomhours
ADDRANDOMMINUTES	tdm_addrandomminutes
ADDRANDOMSECONDS	tdm_addrandomseconds
ADDRANDOMYEARS	tdm_addrandomyears
AMEXCARD	tdm_amexcard
CONCAT	tdm_concatenate
DOB	tdm_dateofbirth
DOD	tdm_dateofdeath
DECRYPT	tdm_decrypt
ENCRYPT	tdm_encrypt
FILL	tdm_fill
FIXED	tdm_fixed
FIXEDDAY	tdm_fixedday
FORMATENCRYPT (for characters)	tdm_formatencrypt
FORMATENCRYPT (for numerics) (PARM7 "Excluded Chars' is not supported)	tdm_nformatencrypt
FORMATENCRYPT1 (for characters)	tdm_formatencrypt1

FORMATENCRYPT1 (for numerics) (PARM7 "Excluded Chars" is not supported)	tdm_nformatencrypt1
FORMATHASH	tdm_formathash
GENCARD	tdm_generatecard
GUID	tdm_guid
HASH	tdm_hash
HASHDOB	tdm_hashdob
HASHIBAN	tdm_hashiban
HASHLOV	tdm_hashlov
INTRANGE	tdm_intrange
LUHN	tdm_luhn
MASTERCARD	tdm_mastercard
NINO	tdm_nino
NUMERICRANGE	tdm_numericrange
PARTMASK	tdm_partialmask
RANDOM	tdm_random
RANDOMDATE	tdm_randomdate
RANDOMDAYS	tdm_randomdays
RANDEIN	tdm_randomein
RANDHIC	tdm_randomhic
RANDSSN	tdm_randomssn
RANDOMTXT	tdm_randomtext
TRANSLATE	tdm_translate
TRANSPOSE	tdm_transpose
TRIM	tdm_trim
VISACARD	tdm_visacard

TIP
For more information about the Fast Data Masker masking functions, see the Masking Functions and Parameters section.

Deploy the JAR Files and Register Hive UDFs

To deploy the JAR files and register defined Hive UDFs in your Hive environment, run the catdm-masker-init.hql utility as follows:

1. Extract the MaskingSDK-<*version*>.zip file from the root directory of your CA TDM installation media, to an appropriate location.
2. Locate and copy the utility (catdm-masker-init.hql) to a computer (where Hive is available) ensuring that no special characters are added to the utility file name.
3. Locate the JAR files (catdm-masker-hive-<*version*>.jar, catdm-masker-library-<*version*>.jar, and commons-validator-<*version*>.jar) and copy them to the same computer.
4. Update the catdm-masker-init.hql utility with the paths of the JAR files.
5. Run the following command in the Hive environment to execute the catdm-masker-init.hql utility:

   ```
   hive -i catdm-masker-init.hql
   ```

   The utility successfully adds the JAR files and defined Hive UDFs to the Hive session.

6. Verify that the JAR files are added successfully by using the following Hive statement:

```
list jars;

Example response is as follows:

/home/hadoopuser/camasking/catdm-masker-hive-<version>.jar
/home/hadoopuser/camasking/catdm-masker-library-<version>.jar
/home/hadoopuser/camasking/commons-validator-<version>.jar
```

7. Verify that all the Hive UDFs present in the catdm-masker-init.hql utility are added to your Hive environment by using the following Hive statement:

```
show functions;

Example response is as follows; the list includes all the Hive UDFs that you have added:

acos
array
tdm_add
tdm_adddays
tdm_addpercent
....
....
tdm_trim
tdm_visacard
 Note: In addition to the Hive UDFs that you have added, the list also displays other UDFs if they are
 already present in the environment. For example, acos and array are the two UDFs that are already present
 in the Hive environment.
```

In secured Hadoop clusters, adding JARs may result in an error message similar to

```
insufficient privileges to execute add (state=42000, code=0)
```

One solution is to ask your Hadoop Cluster Admin to update the Hadoop cluster Hive server configuration hive-site.xml to add the JAR files. Define the full path of the jar files in the the property hive.aux.jars.path.

1. Log on with Hadoop cluster admin privileges.
2. Copy the jar files catdm-masker-library-<*version*>.jar, catdm-masker-hive-*version*.jar and commons-validator-<*version*>.jar to any directory on the Hive Server nodes.
   Example: /usr/catdm/maskingsdk/
3. Edit the hive-site.xml file and define the hive.aux.jars.path property, and save the file.
   Example:

```
hive-site.xml
<property>
    <name>hive.aux.jars.path</name>
    <value>file:///usr/catdm/maskingsdk/catdm-masker-library-<version>.jar,
           file:///usr/catdm/maskingsdk/catdm-masker-hive-<version>.jar,
           file:///usr/catdm/maskingsdk/commons-validator-<version>.jar
    </value>
</property>
```

4. Remove the add jars statements from the init hql file.
5. Restart the Hive server for the configuration changes to take effect.

Execute Hive UDFs

Use Hive UDFs that the deployed JAR file includes to perform the supported masking operations.

1. View all Hive UDFs that are available in your Hive environment by using the following Hive statement:
   ```
   show functions;
   ```
   All Hive UDFs in the Hive environment are displayed.
2. Note the Hive UDF name that you want to use for masking.
3. Use the following Hive statements to know more about the Hive UDF:
   ```
   desc function <function_name>;
   This statement provides information about the Hive UDF.

   desc function extended <function_name>;
   This statement provides an example about the Hive UDF usage.
   ```
   Appropriate description about the Hive UDF is displayed.
4. View the schema of the table that you want to mask by using the following Hive statement:
   ```
   desc <table_name>;
   ```
   The table schema is displayed on the screen.
5. View the data in the table that you want to mask by using the following Hive statement:
   ```
   select * from <table_name>;
   ```
   The existing data is displayed on the screen.
6. Use the appropriate Hive UDF in a Hive "select" statement to preview how the structured data is masked in the database table:
   ```
   select <UDF_name_with_parameters> from <table_name>;
   ```
   The result of the Hive "select" statement shows how the structured data would be masked in the database if you use the Hive UDF in a Hive "insert" statement.

   NOTE
   The MaskingSDK only provides Hive UDFs to mask the data stored in Hadoop. In order to save the masked data into Hadoop, depending on the table configuration in Hive, use Insert statement variants, like `insert overwrite` or `insert into`.

Example: Use the following insert statement to mask the data in the database:

```
insert overwrite table <table_name> select <Hive_UDF>(<table_column_1>),<table_column_2>,....,<table_column_n>
 from <table_name>
```

The data is updated and is masked in the table.

Run a Masking Job in the Simulation Mode

You can run your masking job in the simulation mode (preview mode). The simulation mode lets you review the *before mask* and *after mask* values before you make the actual updates in the database. When you specify options to run the masking job in the simulation mode, Fast Data Masker runs the masking process without updating the database. Fast Data Masker saves all the information in an audit file (CSV file) with the original and masked values. You can view the CSV file and can analyze how your masking affects the data in the database. Based on your analysis, you can take an informed decision. For example, you can decide whether you want to proceed with the specified masking job or you want to further refine the options.

You configure appropriate options in the **Options** tab to enable the simulation mode.

1. Access the Fast Data Masker Mapper interface.

2. Click the **Masking** tab and perform the following tasks:
   a. Select the table that you want to use for masking.
   b. Select the column that you want to mask for the selected table.
   c. Select the appropriate mask type.
3. Click the **Options** tab and provide information for the appropriate options to use the simulation mode based on your requirements:
   – DBUPDATES
     Specify the value as N to run the masking job in the simulation mode.
   – AUDIT
     Specify whether you want to audit all the rows in the column (ALL), first n rows in the column (ROWnnn), or every nth row in the column (SAMPLEnnn).
   – AUDITFILE
     Specify the name of the audit file in which you want to store the audit information. The default name is myaudit.csv.
   – AUDITDIR
     Specify the location where you want to save the audit file.
   – AUDITZIP
     Specify whether you want to zip the audit CSV file. Values are winzip or jzip for the program to use for the zip.
     Note: Ensure that the WinZip command-line utility is already installed on the same system. This utility is required for all the options that involve zipped file (AUDITZIP and AUDITPASSWORD).
   – AUDITPASSWORD
     Specify the password for the audit ZIP file. This password is required when you try to open the saved audit ZIP file.
   – AUDITVALUESLeave this option empty to show the old and new values in the audit file.
   – AUDITONLYCOLUMNS
     Specify the specific list of columns that you want to audit in the format—table1.column1, table2.column2, table3.column3. This option is helpful in scenarios where you try to mask more columns but want to audit only a few of them. For example, you try to mask five columns and want to audit only two of them.

   Note: For more information about all the masking options, see the Understand Masking Options section.
4. Click the **Summary** tab and perform the following tasks:
   a. Click the **Save & Run Mask** button.
   b. Save the CSV file that contains all the masking information that is displayed in the **Summary** tab.
   c. Save the TXT file that contains all the options information that is displayed in the **Options** tab.
   d. Click **OK** on the confirmation dialog that states that both the files are saved to the specified locations.
      The Fast Data Masker Mapper runs the masking job to initiate and complete the masking process. The **Mask Complete** message appears after the masking is done.
   e. Click OK.
      The data masking in the simulation mode is complete.
5. Navigate to the location where you saved the audit file.
6. Open the saved file.
7. Compare the old data (that you want to mask) against the new data (with which you want to mask) for the columns that you selected for masking.
   This information helps you analyze the impact of your masking.
8. Query your database to verify that the original data remains unchanged in the database because the masking was done in the simulation mode.

Example 1: Mask All Rows in the first_name Column (ALL)

Consider a scenario where you want to mask the first name of all the employees. Additionally, you want to use the simulation mode so that you can view the masked values before database updates are done. In this case, after you specify appropriate values in the **Masking** tab, specify the following values in the **Options** tab:

- AUDIT=ALL
- AUDITDIR=c:\Audit
- AUDITFILE=firstName.csv
- DBUPDATES=N

Now, when you run the masking job (**Summary** tab), the Fast Data Masker Mapper runs the masking in the simulation mode and saves all the information about old and new data in this CSV audit file. The Fast Data Masker Mapper does not update the database with the masking data. The following snippet shows an example of the audit file firstName.csv that includes the old and new first names:

```
MASKING STARTED AT: 2016.01.08 12:08:12.167 IST

MASKING DATABASE USER: sa

OPERATING SYSTEM USER: Administrator

MAPPING FILE ARCHIVED AS: C:/Users/Administrator/AppData/Roaming/Grid-Tools/
FastDataMasker/backups/PreviewMask.20160118115152.csv

OPTIONS:

AUDIT=ALL

AUDITDIR=c:\Audit

AUDITFILE=firstName.csv

DBUPDATES=N

SEEDFILEDIR=C:\Program Files\Grid-Tools\FastDataMasker\seedtables

TABLE,UNIQUE COLUMNS, UNIQUE COLUMN VALUES, MASK COLUMN, FUNCTION, OLDVALUE,NEWVALUE

employee,"employee_id","1001","first name","HASHLOV","Keyla","Andrea"

employee,"employee_id","1002","first name","HASHLOV","Kiara","Jaylene"

employee,"employee_id","1003","first name","HASHLOV","Cassandra","Lina"

employee,"employee_id","1004","first name","HASHLOV","Alena","Carissa"

employee,"employee_id","1005","first name","HASHLOV","Janice","Genevieve"

employee,"employee_id","1006","first name","HASHLOV","Iliana","Mayra"

employee,"employee_id","1007","first name","HASHLOV","Raina","Kennedy"

employee,"employee_id","1008","first name","HASHLOV","Christa","Cassie"
```

```
employee,"employee_id","1009","first name","HASHLOV","Vanesa","Adeline"

employee,"employee_id","1010","first name","HASHLOV","Jamya","Cassidy"

employee,"employee_id","1011","first name","HASHLOV","John","William"

employee,"employee_id","1012","first name","HASHLOV","Stephen","Peter"

employee,"employee_id","1013","first name","HASHLOV","Shawn","Derek"

employee,"employee_id","1014","first name","HASHLOV","Lauri","Pamela"

employee,"employee_id","1015","first name","HASHLOV","Michael","Gregg"
```

This preview information helps you understand how your masking changes the data in the database.

The following table shows a snippet of the "employee" table in the database after you run the masking job in the simulation mode. You can verify that no changes are made to the first_name column (as expected in this mode). The first_name column still shows the original name (before masking):

employee_id	first_name	last_name
1001	Keyla	Strogoff
1002	Kiara	Garine
.....		
1009	Vanesa	Noot
1010	Jamya	Bach
.....		
1015	Michael	Witt

Example 2: Mask First 10 Rows in the first_name Column (ROW010)

Consider a scenario where you want to mask the first name of the first ten employees in the employee table. In this case, specify the following values in the **Options** tab:

- AUDIT=ROW010
- AUDITDIR=c:\Audit
- AUDITFILE=tenFirstName.csv
- DBUPDATES=N

The following snippet shows an example of the audit file tenFirstName.csv. This file includes the old and new first names of the first ten employees:

```
MASKING STARTED AT: 2016.01.08 11:51:52.913 IST

MASKING DATABASE USER: sa

OPERATING SYSTEM USER: Administrator
```

```
MAPPING FILE ARCHIVED AS: C:/Users/Administrator/AppData/Roaming/Grid-Tools/
FastDataMasker/backups/PreviewROW.20160118120812.csv

OPTIONS:

AUDIT=ROW010

AUDITDIR=c:\Audit

AUDITFILE=tenFirstName.csv

DBUPDATES=N

SEEDFILEDIR=C:\Program Files\Grid-Tools\FastDataMasker\seedtables

TABLE,UNIQUE COLUMNS, UNIQUE COLUMN VALUES, MASK COLUMN, FUNCTION, OLDVALUE,NEWVALUE

employee,"employee_id","1001","first name","HASHLOV","Keyla","Andrea"

employee,"employee_id","1002","first name","HASHLOV","Kiara","Jaylene"

employee,"employee_id","1003","first name","HASHLOV","Cassandra","Lina"

employee,"employee_id","1004","first name","HASHLOV","Alena","Carissa"

employee,"employee_id","1005","first name","HASHLOV","Janice","Genevieve"

employee,"employee_id","1006","first name","HASHLOV","Iliana","Mayra"

employee,"employee_id","1007","first name","HASHLOV","Raina","Kennedy"

employee,"employee_id","1008","first name","HASHLOV","Christa","Cassie"

employee,"employee_id","1009","first name","HASHLOV","Vanesa","Adeline"

employee,"employee_id","1010","first name","HASHLOV","Jamya","Cassidy"
```

The following table shows a snippet of the "employee" table in the database after you run the masking job in the simulation mode. You can verify that no changes are made to the first_name column (as expected in this mode). The first_name column still shows the original name (before masking):

employee_id	first_name	last_name
1001	Keyla	Strogoff
1002	Kiara	Garine
.....		
1008	Christa	Chastain
1009	Vanesa	Noot
1010	Jamya	Bach

Example 3: Mask Every 10th Row in the first_name Column (SAMPLE10)

Consider a scenario where you want to mask the first name of every tenth employee in the employee table. In this case, specify the following values in the **Options** tab:

- AUDIT=SAMPLE010
- AUDITDIR=c:\Audit
- AUDITFILE=firstNameTenth.csv
- DBUPDATES=N

The following snippet shows an example of the audit file firstNameTenth.csv. This file includes the old and new first names of every tenth employee in the employee table in the database:

```
MASKING STARTED AT: 2016.01.08 12:03:17.402 IST

MASKING DATABASE USER: sa

OPERATING SYSTEM USER: Administrator

MAPPING FILE ARCHIVED AS: C:/Users/Administrator/AppData/Roaming/Grid-Tools/
FastDataMasker/backups/PreviewSample.20160118120317.csv

OPTIONS:

AUDIT=SAMPLE010

AUDITDIR=c:\Audit

AUDITFILE=firstNameTenth.csv

DBUPDATES=N

SEEDFILEDIR=C:\Program Files\Grid-Tools\FastDataMasker\seedtables

TABLE,UNIQUE COLUMNS, UNIQUE COLUMN VALUES, MASK COLUMN, FUNCTION, OLDVALUE,NEWVALUE

employee,"employee_id","1010","first name","HASHLOV","Jamya","Cassidy"

employee,"employee_id","1020","first name","HASHLOV","Nicole","Kylie"

employee,"employee_id","1030","first name","HASHLOV","Adrian","Lisa"

employee,"employee_id","1040","first name","HASHLOV","Robin","Anya"

employee,"employee_id","1050","first name","HASHLOV","Kristina","Madeline"

employee,"employee_id","1060","first name","HASHLOV","Paul","James"

employee,"employee_id","1070","first name","HASHLOV","Terrence","Richard"

employee,"employee_id","1080","first name","HASHLOV","Maria","Elisa"
```

```
employee,"employee_id","1090","first name","HASHLOV","Jina","Jennifer"
```

The following table shows a snippet of the "employee" table in the database after you run the masking job in the simulation mode. You can verify that no changes are made to every tenth row in the first_name column (as expected in this mode). Every tenth row in the first_name column still shows the original name (before masking):

employee_id	first_name	last_name
1001	Keyla	Strogoff
1002	Kiara	Garine
.....		
1009	Vanesa	Noot
1010	**Jamya**	**Bach**
1011	Vladimir	Yagolnister
.....		
1020	**Nicole**	**Appling**

Additional Examples

The following are a few more examples that explain other use cases in the simulation mode:

- **Zip Your Audit File**
  If you want to zip your firstName.csv audit file and save the zipped file in the audit directory, you can specify the following information in the **Options** tab:
  – AUDIT=ALL
  – AUDITDIR=c:\Audit
  – AUDITFILE=firstName.csv
  – AUDITZIP=winzip
  – DBUPDATES=N
  After you run the masking job, the audit file is zipped and is saved as firstName.zip in the c:\Audit folder. To access the firstName.csv file, extract the zipped file (firstName.zip).
- **Use Password to Protect Your Audit File**
  If you want to zip your firstName.csv audit file, protect your audit file with the help of a password, and save the zipped file in the audit directory, you can specify the following information in the **Options** tab:
  – AUDIT=ALL
  – AUDITDIR=c:\Audit
  – AUDITFILE=firstName.csv
  – AUDITPASSWORD=MyPassword01
  – AUDITZIP=winzip
  – DBUPDATES=N
  After you run the masking job, the audit file is zipped, protected with a password, and is saved as firstName.zip in the c:\Audit folder. To access the firstName.csv file, extract the zipped file (firstName.zip) and enter the password.
- **Audit Only Specific Columns**
  Consider a scenario where you want to mask the first_name and last_name columns in the employee table but want to audit only the first_name column. In this case, you specify the following information in the Options tab:

- AUDIT=ALL
- AUDITDIR=c:\Audit
- AUDITFILE=firstName.csv
- AUDITONLYCOLUMNS=employee.first_name
- DBUPDATES=N

The Fast Data Masker Mapper audits only the first_name column in the employee table. The Fast Data Masker Mapper ignores the last_name column for audit.

Fast Data Masker Troubleshooting

This article includes information that can help you troubleshoot Fast Data Masker issues.

Chinese Characters Are Not Rendered Correctly

Problem:

I am trying to mask chinese characters using FDM functions. When I enter Chinese characters in parameter fields, the user interface does not display proper characters, but empty boxes.

Solution:

For the GTMapper's fields to support Chinese characters, configure the locale details in the config.xml file. FDM can mask Chinese data, but only the following functions have been tested: CONCAT, SQLFUNCTION, FIXED and HASHLOV. Note that this configuration does not enable Chinese localization of the user interface.

Default locale values:

```
<locale>
  <language>en</language>
  <country>US</country>
  <variant></variant>
  <font>Arial</font>
</locale>
```

For example, configure the following options to be able to display Hong Kong Chinese characters. For more information on subtags, see the language subtags registry: https://www.iana.org/assignments/language-subtag-registry/language-subtag-registry

```
<locale>
  <language>zh</language>
  <country>HK</country>
  <variant></variant>
  <font>MS Song</font>
</locale>
```

Connection Error (Microsoft SQL Server) in Fast Data Masker

Symptom: After I launch Fast Data Masker and try to connect to a Microsoft SQL Server database profile, I receive the following error:

```
Connection Error: This driver is not configured for integrated
authentication.authentication.
```

The steps that caused this error in my case are as follows:

1. Launch Fast Data Masker by clicking **Start, All Programs, FastDataMasker, FastDataMasker**.

2. Copy the default connection connectSQLSERVER.txt by selecting it in the **Connection Files** list in the left pane and then clicking the **Copy Connection** button at the bottom of the dialog.
3. Enter the database connection information as required.
   Note: The name that you enter in the **Connection Name** field shows up in the **Connection Files** list after it is saved. You cannot see any underscores that you type inside this connection dialog. Instead, all underscores look like spaces. For example, `connect_test` appears as `connect test` . Additionally, if you get an error about the connection name, ensure that the name starts with the word connect.
4. Click **Save** to save the connection information.
5. Click **Connect** to connect to this profile.
   The following error is displayed:
   `Connection Error: This driver is not configured for integrated authentication.Authentication.`

Solution: If you get this error, it means that a Microsoft SQL Server authentication DLL file is missing from your Fast Data Masker folder. To address this issue, follow these steps:

1. Close the Fast Data Masker application.
2. Navigate to the location where you installed Fast Data Masker. For example, the default location is C:\Program Files \Grid-Tools\FastDataMasker.
3. Open the SQLSERVER_DLLs folder.
4. Copy the sqljdbc_auth.dll file.
5. Paste this sqljdbc_auth.dll file into the C:\Program Files\Grid-Tools\FastDataMasker folder.
6. Launch the Fast Data Masker application.
7. Enter your Microsoft SQL Server database credentials.
8. Click **Save**.
9. Click **Connect**.
   You can now connect to and use Fast Data Masker without any issue.

"no dbjdbc12 in java.library.path" Error in Fast Data Masker

Symptom: When I try to connect to Fast Data Masker, I receive the following error message

`no dbjdbc12 in java.library.path`

I followed these steps when I received the error:

1. Launch Fast Data Masker.
2. In the **Connection Files** pane, click the **New** button.
3. Select a data source from the **DBMS** drop-down list.
4. Enter all the required connection details.
5. Click the **Connect** button.
   The following error is displayed:
   `no dbjdbc12 in java.library.path`

Solution: This issue pertains to having the SQLANYWHERE drivers (sajdbc.jar and sajdbc4.jar) in the Fast Data Masker `lib` folder. These drivers require an underlying DLL to work, even if you are not connecting to SQLANYWHERE. Fast Data Masker throws an error irrespective of the database you are connecting to.

Since CA TDM 3.5, certain jar files are no longer shipped with the product. If you are upgrading to CA TDM 3.5 (or later) and are seeing this error message, it is because the Fast Data Masker `lib` folder now contains these jar files. You must remove these files.

To address this issue, you can follow one of the following methods:

- **Method 1: Remove the two JARs**
  a. Close the Fast Data Masker interface.

b. Remove the two JAR files (sajdbc.jar and sajdbc4.jar) from the Fast Data Masker `lib` folder.
The default location is `C:\Program Files\Grid-Tools\FastDataMasker\lib`.
c. Launch Fast Data Masker because many of these files are loaded upon launch.
You must be able to connect to Fast Data Masker.

- **Method 2: Ignore the Error**
  a. If you are using any other database connection other than SQLANYWHERE (for example, Oracle), you can click the **OK** button on the error message to ignore it.
- **Method 3: Copy the old contents**
  If you have recently upgraded and have the older version backed up or installed in another location, you can copy and paste the contents (not the actual folder) of your old Fast Data Masker `lib` folder over to your current Fast Data Masker `lib` folder. This way you can have all the JAR files that you need.
  a. Close the Fast Data Masker interface.
  b. Copy all the contents from your old `lib` folder into the new `lib` folder. All new drivers remain the same.
  c. Launch Fast Data Masker because many of these files are loaded upon launch.
  You must be able to connect to Fast Data Masker.

Connect to Microsoft SQL Server Using Windows Authentication in Fast Data Masker

Symptom

The data that I want to mask is available in a Microsoft SQL Server database that allows only Windows (Integrated authentication. How can I connect to this database from Fast Data Masker so that I can mask the required data?

Solution

You might need to mask the data that resides in a Microsoft SQL Server database that has been configured only for Windows authentication, not for the Microsoft SQL Server authentication. In such cases, follow these steps to establish the connection:

1. Close the Fast Data Masker application.
2. Navigate to the SQLSERVER_DLLs folder under the FastDataMasker installation folder (by default, C:\Program Files \Grid-Tools\FastDataMasker).
3. Copy the file named sqljdbc_auth.dll from the x64 subfolder. This is the 64-bit version of the DLL and is the one you need, assuming your system runs on a 64-bit processor.
4. Place the sqljdbc_auth.dll file into the FastDataMasker folder (by default, C:\Program Files\Grid-Tools \FastDataMasker).
5. Launch the Fast Data Masker UI.
6. Enter the Microsoft SQL Server database connection details.
7. Make sure the username and password fields are left blank.
8. Save the information.
9. Click the **Connect** button.
   You can now connect to the Microsoft SQL Server database that allows connections using only Windows authentication.

Note: After copying the sqljdbc_auth.dll file, if you get the error message like `This driver is not configured for windows authentication`, it is most likely that the incorrect version of sqljdbc_auth.dll has been copied. Repeat the process with a different version of the DLL.

Disable Fast Data Masker's connection to TDM Portal

Symptom

I have upgraded FDM to version 4.9, but my installation of TDM Portal is still an earlier version, or the TDM Portal service is not active. I experience a delay at startup of FDM, while it attempts to connect with the TDM Portal service.

Solution

To avoid this delay, and to prevent FDM from attempting to connect with TDM Portal, you can change the value of the Windows Environment Variable **TDM_PORTAL_URL** to `N` or `n` . **TDM_PORTAL_URL** defines the address where the TDM Portal service is active.

> **WARNING**
>
> If you later upgrade TDM Portal to the same version as FDM, you need to amend this parameter again, to the URL where the TDM Portal service runs.

Mask Data Stored in Teradata

You can use Fast Data Masker masking functions as Hive user-defined functions (UDFs) to mask data stored in Teradata. The stored data must be structured data with a defined schema.

Create Teradata Type Transformation Map in Datamaker

1. Connect Datamaker to a Teradata data source
2. Create a project of type Teradata in Datamaker
   - Project name: ORDERS TERADATA
   - File Publish DBMS: Teradata
   - Publish To: Data Target
   - Inherit Tables: Yes
   - Type: DB
3. Register the tables in the project.
4. Click **Project,Transformation Maps** to create a transformation map.
   Important: Set the DBMS for the map to TERADATA.
5. Use the drop-down menus under **Transformation** to set the masking function for each column that you want to mask.
6. Save the transformation map.

Teradata supports the following masking functions:

ADDPERCENT,*n*

Modifies an existing value by adding *n* percent of the original value.

Applies to: Numeric data types

Example: ADDPERCENT,40
We add 40% to the value in the APPROXIMATE_INCOME column in the table PERSONS. The original value 18000 is masked as 25200.

ADDRANDDAYS,*min*,*max*

Adds a random number of days between *min* and *max* to an existing date value.

Applies to: DATE data types

Example: ADDRANDDAYS,1,30
We want to add a random number of days between 1 and 30 to the existing value in the EXPIRATION_DATE column in the table CREDIT_CARDS. The original value 2018-01-07 is masked as 2018-01-14.

ADDRANDMONTHS,*min*,*max*

Adds a random number of months between *min* and *max* to an existing date value.

Applies to: DATE data types

Example: ADDRANDMONTHS,1,12
The EXPIRATION_DATE column in the table CREDIT_CARDS adds a random number of months between 1 and 12 to the existing value. The original value 2018-01-07 is masked as 2018-11-07.

ADDRANDOMNUM,*min*,*max*

Adds a random number between *min* and *max* to the existing numeric value.

Applies to: Numerics

Example: ADDRANDOMNUM,10,90
You want to add a random number between 10 and 90 to the existing value in the ORDER_TOTAL column in the table ORDERS. The original value 1721 is masked as 1788.

AMEXCARD

Generates a random American Express credit card number. The generated number has a valid number format and check digit.

Applies to: Character columns

Example: AMEXCARD
You want random valid Amex numbers in the CARD_NUMBER column in the table CREDIT_CARDS. The original value 371449635398431 is masked as 375548058820296.

CHARHASH

Hashes a character string to consistently change to a new value. This function does not keep the original format of the string, nor does it guarantee uniqueness.

Applies to: Character columns

Example: CHARHASH
You want to replace the text in the EMAIL column in the table PERSONS by random text. The original value `MARK4784@yahoo.co.uk` is masked as `iBNr35^4\]}CU;<@/Q/qAGm=` .

CHARHASHPAT

Hashes a character string to consistently change to a new value. The function keeps the original format of the string, but does not guarantee uniqueness.

Applies to: Character columns

Example: CHARHASHPAT
You want to replace the text in the EMAIL column in the table PERSONS with similar values. The original value `MARK4784@yahoo.co.uk` is masked as DBBO3340@EATLF.OQ.AO.

CREDITCARDKEEPTYPE

Generates a random credit card number while retaining the card type. For example, an existing Visa card number will be masked as a new Visa card number, Amex as a new Amex number etc. The generated number has a valid number format and check digit.

Applies to: Character columns

HASH[,*n*]

Hashes a number to consistently change to another number. This function does not guarantee unique values. The optional numeric parameter, for example "`HASH,123`", provides the same function as HASH but with a hash key of 123.

Applies to: Numeric columns

Example: HASH
You want to change the PERSON_ID column in the table PERSONS consistently to a new number. The original value 202 is masked as 682671856

HASHLOV,*SEED DATA CATEGORY NAME*

Hashes the current value to get an integer value. This integer value is used to consistently look up a value from a seed list.

Applies to: character columns

Example: HASHLOV,NAME - FEMALE INDIAN FIRST
You want to change the values in the FIRST_NAME column in the table PERSONS consistently to female Indian first names. The original value Donnachadh is masked as Poonam.

MASTERCARD

Generates a random Mastercard credit card number. The number has a valid number format and check digit.

Applies to: Character columns

Example: MASTERCARD
You want to mask the values in the CARD_NUMBER column in the table CREDIT_CARDS by random Mastercard numbers. The original value 4547357219782851 is masked as 5131725103379163.

RANDLOV,*SEED DATA CATEGORY NAME*

Picks a random value from a seed list for the given category name.

Applies to: character columns

Example: RANDLOV,NAME - FEMALE INDIAN FIRST
You want to change the value in the FIRST_NAME column in the table PERSONS randomly to a female Indian first name. The original value Lennox is masked as Sumita.

RANDOMNUM,*min*,*max*

Generates a random number between the supplied minimum and maximum values.

Applies to: numeric columns

Example: RANDOMNUM,10,70
You want to set the value in the QUANTITY column in the table ORDER_ITEMS to a random number between 10 and 70. The original value 40 is masked as 58.

RANDOMTXT,*min*,*max*

Generates random text within the specified minimum and maximum length.

Applies to: character columns

Example: RANDOMTXT,10,20
You want to replace the values in the OBJECT_NOTES column in the table ORDER_ITEMS by random text of length between 10 and 20. The original value "Test notes" is masked as "wwohultvirbaqmg".

REPLACE,*abc*,*xyz*

Replaces all instances of the string *abc* in the original value with the string *xyz*. The replacement is case sensitive.

Applies to: character columns

Example: REPLACE,provider.com,post.com
You want to replace all occurrences of yahoo.com in the EMAIL column in the table PERSONS with post.com. The original value MARK4083@provider.com is masked as MARK4083@post.com, while JOHN4003@abc.com remains the same.

SHUFFLE

Shuffles the columns values in a table.

Applies to: numeric, date, and character columns

Example: SHUFFLE
You want to shuffle the values in the EMAIL column in the table PERSONS relative to the other column values. The original table (left) is masked as follows (right):

First Name	Last Name	Email
Tevin	Carr	Tevin.Carr@yahoo.com
Clennan	Riddle	Clennan.Riddle@yahoo.com

First Name	Last Name	Email
Tevin	Carr	Clennan.Riddle@yahoo.com
Clennan	Riddle	Tevin.Carr@yahoo.com

SSNKEEPTYPE[,*separator*]

Masks a US social security number, retaining the first 3 digits, which determine the area. Optionally, specify a *separator* character.

Example: SSNKEEPTYPE,-
You want to generate new values for the SSN column in the table PERSONS, keeping the first 3 digits, and using a hyphen as a separator. The original value 473925722 is masked as 473-81-6680.

TAKERANDDAYS,*min*,*max*

Subtracts a random number of days between *min* and *max* from an existing date value.

Applies to: DATE data types

Example: TAKERANDDAYS,1,30
You want to mask the EXPIRATION_DATE column in the table CREDIT_CARDS by subtracting a random number of days between 1 and 30 off the existing value. The original value 2018-01-07 is masked as 2017-12-16

TAKERANDMONTHS,*min*,*max*

Subtracts a random number of months between *min* and *max* from an existing date value.

Applies to: DATE data types

Example: TAKERANDMONTHS,1,12
You want to mask the EXPIRATION_DATE column in the table CREDIT_CARDS by subtracting a random number of months between 1 and 12 from the existing value. The original value 2018-01-07 is masked as 2017-09-07

TRANSLATE,*abc*,*xyz*

Replaces all occurrences of each character in *abc* by its corresponding character in *xyz*. That is, you want to replace a by x, b by y, and c by z. Characters in the original value that are not in ABC are not replaced. The replacement is case sensitive.

Applies to: Character data types

Example: TRANSLATE,abcdef,hrtsgj
You want to replace all occurrences of the characters "abcdef" with the corresponding character in "hrtsgj", that is, replace a with h, replace b with r, etc. The original value Comyn.Barry@yahoo.com is masked as Comyn.Bhrry@yhhoo.tom.

VISACARD

Generate a random Visa card credit card number. The number will have a valid number format and check digit.

Applies to: Character columns

Example: VISACARD
You want to replace the value in the CARD_NUMBER column in the table CREDIT_CARDS by a random Visa card number. The original 5131725103379163 is masked as 4547357219782851.

XREFLOOKUP

Looks up a masked value from the gtscr_xref table which is in the database where the scramble functions are installed. A prerequisite for this to work is that the gtscr_xref table is pre-populated with old and masked values for the identifier. If the original value is not found in the cross reference table (column rx_old_value) then it is ignored and the original value is retained.
Note: You must supply the **Cross Ref Ident** value.

Applies to: Numeric and Character columns

Example: XREFLOOKUP
You want to mask values in the column SSN using the lookup identifier SSN in the cross-reference table. The original value 111111112 is masked as 473925722.

Rx Ref Id	Rx Old Value	Rx New Value
SSN	111111112	473925722
SSN	111111115	323632875

Generate Masking Scripts in TDM Subset

Generate scripts to move data.

1. Open TDM Subset.
2. Select **File, Build Database Actions**. The Database Action - Build Teradata Window opens.
3. Go to the **Extra Details** tab and fill in the following Teradata-specific fields.
   a. **Action name**
   b. **Functions Database** – if generating scramble scripts, this is the name of the database where the scramble C UDF functions have been installed – choose from drop down list or type database name
   c. **Export Log Database** – the database where the export log tables will be created. Typically this will be the database where the data is extracted from – choose from drop down list or type database name.
   d. **Load Log Database** – the database there the load log tables are created. Typically this will be the target database where the data will be loaded into – choose from drop down list or type database name.
   e. **Source Database** – the database the data will be extracted from– choose from drop down list or type database name, or choose environment variable (name of the variable enclosed in %)

f. **Target Database** – the database the data will be loaded into – choose from drop down list or type database name, or choose environment variable (name of the variable enclosed in %)
g. **Stage Source Database** – for stage insert scripts, the database the data will be selected from – choose from drop down list or type database name.
h. **Stage Target Database** – for stage insert scripts, the database the data will be inserted into - choose from drop down list or type database name, or choose environment variable (name of the variable enclosed in %)
i. **Log Directory** – (Windows only) the directory the log files will go to. This can be a directory path or a Windows environment variable that points to a directory or choose environment variable (name of the variable enclosed in %)
j. **Extract file directory** — (Windows only) the directory the fast export data files will go to, and where the load scripts will load from – this can be a directory path or a windows environment variable that points to a directory or choose environment variable (name of the variable enclosed in %)
k. **Get Metadata from repository connection** — (Optional) Extract the details directly from the Datamaker repository to speed up the response when building masking scripts.
l. Click **Generate** to extract the details as normal.

4. Go to the **Extract Tables** tab
   a. Click **Open Extract Directory** and browse to the directory where you saved extracts definitions.
   b. Select check boxes to add extract tables to the list at the bottom of the screen. If you wish to choose from ALL the schema tables rather than those defined in a Subset extract definition, then check the ‘Use Connection Tables Only’ checkbox. This leaves the ‘Subset Tables’ list box empty and fills the ‘All Data Tables’ list with the tables for the current connection. You can then move tables between this list and the ‘No Data Tables’ list to define which tables have their data extracted.
   c. Click Generate to create the appropriate export and import scripts to populate the subset schema
5. (Optional) Go to the **Extra Scripts** tab.
   a. Drop / Create Indexes – Creates SQL scripts to drop and recreate indexes and primary keys. These are called by the relevant load scripts since Teradata does not support loads into tables with secondary indexes. This is the default.
   b. Truncate / Delete From Load Tables – Clear down data from the target database prior to loading data
   c. Disable / Enable or Drop / Create FKs – For Teradata, this drops foreign key constraints prior to the data load, then recreates them after the load has completed

      TIP

      Use bind variables in Teradata Windows exports when defining extracts against Teradata. For example:

      ```
      select * from orders.PERSONS
      where person_id < :id
      AND FIRST_NAME  = :NAME
      ```

      The binds are added to the driving table SQL. Create environment variables with the same name in the Windows system to resolve the variables at run time. Enclose non-numeric values in single quotes.

Use Transformation Map Files

To use transformation map files in Fast Data Masker, ensure that you have already exported your transformation maps in the form of CSV files by using the Datamaker UI. Also, ensure that the table and the column information included in your exported transformation map files exist in the connection profile database that you are using in Fast Data Masker.

1. Click **Start, All Programs, FastDataMasker, FastDataMasker** to launch Fast Data Masker.
   The Fast Data Masker connection dialog opens. This dialog contains a list of existing connection files in the left pane and corresponding fields in the right pane.
2. Click a connection file in the left pane.
3. Verify the existing information that is auto-filled and enter configuration details as appropriate.

4. Click **Connect** to connect to the Fast Data Masker Mapper.
   The Fast data Masker Mapper interface opens.
5. Click the **Open Saved Mask** button in the **Masking** tab.
6. Browse to the location where you exported the transformation map CSV file.
7. Select the transformation map CSV file and click **Open**.
   The complete masking information that is included in the transformation map file is displayed in the right pane.
8. Review masking details and update the values, if required.
9. Use the **Restartability** and **Options** tabs to provide appropriate information, if required.
10. Use the **Summary** tab to view the overall information and click **Save & Run Mask** to save and run the masking.

You have successfully used a transformation map to mask the data in Fast Data Masker.

Data Scrambling

The data scrambling process can be split into two discrete steps.

1. In Datamaker, identify the table columns that you want to scramble, and select the appropriate scramble function.
2. In Data Subset, design the extract, then connect to the Datamaker repository to select the scramble project.
   a. Choose the table columns that you want to scramble, and specify the scramble functions to apply to these columns.
   b. If necessary, also select the table order project from the Datamaker repository.

Familiarize yourself with the following concepts:

Add Scrambled Data

Datamaker has several different techniques to scramble (mask or obfuscate) data. The method that you use depends on the particular task you are perform. You can use multiple techniques to ensure that the sensitive records are secure.

Method 1

1. Copy data to the central test data repository, and apply randomize functions to store test data.
2. Copy data into your project, and edit the data for publishing later. These edits include:

- – Converting column values to variables
- – Converting column values to functions, which could include a randomize function. For example, *randrange(min,max)*
- – Selecting one of the right-hand click edit functions, such as randomize by range

Method 2

1. Define transformation functions that are applied to the data as it is extracted, or to update the data in-situ
2. To identify the columns to scramble and the masking function to apply, in Datamaker, select Projects, and Transformation Maps.
3. Select a Transformation Map or create a new one. For more information, see transformation Maps.
   The function is labeled as part of the extract process on the server, or as part of an in-situ data masking process. The list of available masking functions can be expanded to include existing transformation, or customized by request.

Add More Scramble Functions (Oracle only)

To include more functions to scramble the date, do the following in Datamaker:

1. Click Tools, and Maintain Data Functions.
2. Click the Plus icon, and enter the function name.

3. Amend the master functions to call your section of code. For example, in Oracle edit the master function F_SCRAMBLE2.PLS.
   The function contains a standard set of input parameters.
4. Create or replace function `gtsrc_scramble2 (in_table_column in varchar,`

```
                                        in_data_precision in number,
                                        in_data_length in number,
                                        in_nullable in char,
                                        in_rownum in number,
                                        in_xref in varchar,
                                        in_keepnull in varchar,
                                        in_list_colno in number,
                                        in_value in varchar,
                                        in_type in varchar,
                                        in_parm1 in varchar,
                                        in_parm2 in varchar)
```

The parameters are available for you to use as parameters to your function. Find the section of code and add in the call to your function. If the function is written in Java, create an Oracle Java function first.

```
  elsif in_type = 'REPLACE' then

     if wk_text is not null then

       wk_text := replace(wk_text,in_parm1,in_parm2);

     end if;

   elsif in_type = 'MYFUNCTION' then

     if wk_text is not null then

       wk_text := myfunction(wk_text);

     end if;
```

Compile the function in the Scramble user and test. The three master functions for Oracle are:

- **F_SCRAMBLE2.PLS**
  Handles Characters
- **F_SCRAMBLED2.PLS**
  Handles Dates
- **F_SCRAMBLEN.PLS**
  Handles Numerics

Add More Seed Tables

1. Add seed rows to the table GTSRC_REFERENCE_DATA.
2. Set the column RD_REF_ID to the name of the seed table.
3. Click Tools, Maintain Data Functions, and add a LIST function to the available functions.
   The function is available in the scramble function drop-down list.

Check Columns for Quoted Data (Microsoft SQL Server only)

When you generate scripts for scrambled Microsoft SQL Server extracts in Data Subset, first verify if the tables in your extract schema have data that contains quotes. Data Subset generates insert statements where character fields are delimited by single quotes. If the data contains quotes, Data Subset issues a replace statement to replace each single quote with two quotes. The first quote acts as an escape character so SQL Server knows that the end of a data field is not reached.

To ascertain which tables and columns in your schema contain quoted data, follow these steps:

1. Register your source tables in Datamaker
2. Click **Tools** and **Actions for Registered Tables**. From the drop-down actions list, select **Check for Quotes in data**.
3. Click the GO button next to the actions list to verify table columns for quoted data.

Datamaker creates a table tag in the right-hand list that is named GT Contains Quotes.

1. Click the GT Contains Quotes tag.
   Datamaker highlights those tables that contain quoted data.
2. Double-click one of these highlighted tables in the center list.
   Datamaker shows the columns for this table that contain quoted data.
3. Click the ‘tables’ node in the tree view on the left of the screen to return to the tables view for the center list.

Datamaker stores the results of these searches in its repository.

If you select the same project as the scrambled project in Data Subset, it interrogates the Datamaker repository for quoted columns and automatically wraps those columns with replace statements when it generates its scramble scripts.

Create a Cross Reference Table for Character Columns (Microsoft SQL Server only)

Creating a cross-reference table for character columns lets you create consistent from-and-to scrambling. You use this method if you want, for example, ACME CORP to be changed to CUSTOMER 1, and MICROSOFT INC to be changed to CUSTOMER 2, and you want this change to occur for several columns across different databases or schemas.

The steps below describe how to scramble the table PRODUCT and the column DESCRIP across databases:

1. Create user-defined function gtsrc_scramble using f_scrambleMSSS.sql in the \scrambleinstall \MSSQLSERVER folder.
2. Create user defined stored procedure gtexplode_ref using gtexplode_ref.sql
3. Add some values to gtsrc_reference_data as follows (this only needs to be done once).
4. Explode data in gtsrc_reference_data by issuing the following command (this only needs to be done once).

```
EXECUTE gtexplode_ref  '[reference id of existing data to use]' ,  '[newid to insert]', '[tablename you are
 inserting into]' , '[column name to scramble]'
```

The Data Subset Steps

1. Open the Database Actions Screen.
2. Select the action type:
   a. (SQL Server only) Scrambled SQL Server Extract
   b. (Oracle only) Scrambled Windows Extract / Loader Import, or UNIX Extract / Loader Import
3. (If you have chosen Scrambled SQL Server Extract only) Enter the functions database name where you have installed the scramble functions ‘md5hash’ and ‘selectrand’.
4. Choose the Data Scrambling tab.
5. Connect to a Datamaker repository if prompted.
6. Choose your GT Datamaker project from the drop-down list where you defined your sensitive columns.
7. Click the Table Order tab if your target database (where you extract to) is constrained.

8. Choose the project where you calculated your insert orders.
9. Click the Generate button to create your scramble scripts.

Install DB2 Scramble Components

To scramble DB2 data, install the following required components.

The DB2 scramble components work with DB2 for zOS only.

1. Browse to the *scrambleinstall/db2* directory.
2. Locate the file *gtsrc_reference_data.ddl.*
3. Run the DDL script in your source database.
   The seed data table gtsrc_reference_data is created, and populated with the seed values for scrambling.

Install Oracle Scramble Components

The components required for Oracle are found in the CA Test Data Manager Repository Kit package at `<installkit>/Oracle_Install_Kit/Databases/Scramble`.

Note: Oracle 11g XE does not support Java. If you want to use the scramble functionality of Data Subset, Oracle 11g standard version or higher is required.

To install the Oracle scramble components:

Follow these steps:

1. Create a new user called SCRAMBLE in your production or copy of production instance.
2. (Optional) Use an existing administrative processes schema.
3. To create a schema, issue the commands:
   ```
   CREATE USER SCRAMBLE IDENTIFIED BY [enter a password here] DEFAULT TABLESPACE USERS;
   GRANT CONNECT, RESOURCE TO SCRAMBLE;
   ```
4. Import the scramble.dmp file user into the schema.
   - **On Windows:**
     a. Browse to the Datamaker directory:
        `install_kit\Oracle_Install_Kit\Database\Scramble`
     b. Edit the
        `scramble.bat`
        file and update the version number in the impdp command to reflect your Oracle database version (11.1, 12.0, or 12.1).
        Example for Oracle database version 12.1:
        ```
        impdp '"%GT_SYS_USER%/%GT_SYS_PASSWORD%%GT_TNS%"' remap_schema=scramble:%DB_USER%
        directory=SCRMBLPUMP dumpfile=scramble.dmp logfile=import.log version=12.1
        ```
     c. Run the scramble.bat file.
        You have imported the `scramble.dmp` file user into the schema.
   - **On *UNIX:**
     a. Copy or ftp the `scramble.dmp` file to, for example, the `/usr/temp/oracle` directory on your Unix box.
     b. Log in as your DBA user.
     c. Create the "scramble" directory.
     d. Grant the SCRAMBLE_USER create privileges for the "scramble" directory and any child directory.
     e. Log in as your scramble user.
     f. Create or replace the "scramble" directory as

```
/usr/temp/oracle
```

' in Oracle SQL PLUS:

```
CREATE OR REPLACE DIRECTORY scramble AS '/usr/temp/oracle';
```

g. Run the following command in the Terminal. Update the version number in the `impdp` command to reflect your Oracle database version (11.1, 12.0, or 12.1).

```
impdp username/password@tnsname remap_schema=scramble:%DB_USER% directory=scramble
dumpfile=scramble.dmp logfile=import.log version=12.1
```

You have imported the `scramble.dmp` file user into the schema.

5. Create public synonyms as follows:

```
CREATE PUBLIC SYNONYM GTSRC_XREF FOR SCRAMBLE.GTSRC_XREF;
CREATE PUBLIC SYNONYM GTSRC_SHUFFLE FOR SCRAMBLE.GTSRC_SHUFFLE;
CREATE PUBLIC SYNONYM GTSRC_SHUFFLEID FOR SCRAMBLE.GTSRC_SHUFFLEID;
CREATE PUBLIC SYNONYM GTSRC_REPLACE FOR SCRAMBLE.GTSRC_REPLACE;
CREATE PUBLIC SYNONYM GTSRC_REFERENCE_DATA FOR SCRAMBLE.GTSRC_REFERENCE_DATA;
CREATE PUBLIC SYNONYM GTSRC_CHECKSUM FOR SCRAMBLE.GTSRC_CHECKSUM;
CREATE PUBLIC SYNONYM GTSRC_SETCOUNT FOR SCRAMBLE.GTSRC_SETCOUNT;
CREATE PUBLIC SYNONYM GTSRC_SCRAMBLE2 FOR SCRAMBLE.GTSRC_SCRAMBLE2;
CREATE PUBLIC SYNONYM GTSRC_SCRAMBLED2 FOR SCRAMBLE.GTSRC_SCRAMBLED2;
CREATE PUBLIC SYNONYM GTSRC_SCRAMBLED3 FOR SCRAMBLE.GTSRC_SCRAMBLED3;
CREATE PUBLIC SYNONYM GTSRC_SCRAMBLEN2 FOR SCRAMBLE.GTSRC_SCRAMBLEN2;
```

6. Issue the following Grants on:

```
GRANT SELECT,UPDATE,DELETE,INSERT ON  SCRAMBLE.GTSRC_XREF TO PUBLIC;
GRANT SELECT,UPDATE,DELETE,INSERT ON  SCRAMBLE.GTSRC_SHUFFLE TO PUBLIC;
GRANT SELECT   ON  SCRAMBLE.GTSRC_REFERENCE_DATA TO PUBLIC;
GRANT EXECUTE  ON  SCRAMBLE.GTSRC_SCRAMBLE2      TO PUBLIC;
GRANT EXECUTE  ON  SCRAMBLE.GTSRC_SCRAMBLEN2     TO PUBLIC;
GRANT EXECUTE  ON  SCRAMBLE.GTSRC_SCRAMBLED2     TO PUBLIC;
GRANT EXECUTE  ON  SCRAMBLE.GTSRC_SCRAMBLED3     TO PUBLIC;
GRANT EXECUTE  ON  SCRAMBLE.GTSRC_CHECKSUM       TO PUBLIC;
GRANT EXECUTE  ON  SCRAMBLE.GTSRC_SETCOUNT       TO PUBLIC;
GRANT EXECUTE  ON  SCRAMBLE.GTSRC_REPLACE        TO PUBLIC;
GRANT EXECUTE  ON  SCRAMBLE.GTSRC_SHUFFLEID      TO PUBLIC;
```

7. (Optional) You can add extra seed data to the GTSRC_REFERENCE_DATA. Manually enter the data, or build insert statements as follows:

```
INSERT INTO gtsrc_reference_data (rd_ref_id, rd_ref_value )
VALUES ('MY CUSTOMERS','FREDS AUTOREPAIRS');
INSERT INTO gtsrc_reference_data (rd_ref_id, rd_ref_value )
VALUES ('MY CUSTOMERS','JOES TYRES');
```

Install SQL Server Scramble Components

The components required for SQL Server are found in the CA Test Data Manager Repository Kit package at `<installkit>/SQL_Server_Install_Kit/Databases/Scramble`.

To use the scramble functionality of Data Subset for SQL Server, set up the following database functions and tables.

Prerequisites to Install Scrambled Components

To install scramble components, verify the following prerequisites:

1. Verify that an SQL Server is installed.
2. Verify users have privileges to create a database and tables, views, functions, procedures, indexes, primary keys, foreign keys, and constraints.
3. Verify you have access to Microsoft SQL Server Management Studio or Enterprise Manager.

Use one of the following methods to create your starter SQL Server database:

- Method 1: Use a simple database attach command.
- Method 2: Use standard utilities to perform a full import of data.

Method 1 Installation

1. Copy the file scramble.mdf to your sql server standard database file system.
2. From SQLServer Management Studio, right-click the Databases icon and select Attach.
3. Enter the database name (gtrep) and add in the file gtrep_test.mdf
   A log file (suffix .ldf) is added.
4. Click OK.

   NOTE
   If there is a log file error, delete the log file entry, and click **OK**.

   When you attach the MDF file and you get an access denied error, place the MDF file in the default BACKUP directory of the MS SQL Server instance and retry from there.

   Example of the BACKUP default dir location:

   C:\Program Files\Microsoft SQL Server\MSSQL11.SQLEXPRESS64BIT\MSSQL\Backup

   The database is attached and is ready to be used as a starter repository.

Method 2 Installation

Note: The SQL Server administrator is required to create the GTREP database and run the batch file to create the necessary tables.

1. Start **Management Studio/Enterprise Manager**, and log in to the **Database Engine**.
2. Click **Databases**, right-click to show the menu with the option **Create Database**.
3. Enter a Database name. For example, "Scramble".
4. Set all options to default, except for **Initial Size** and **Autogrowth.**
   a. For the database file in the **PRIMARY Filegroup**, set **Initial Size** (488MB) and **Autogrowth** (1MB).
   b. For the log file, set **Initial Size** (1744MB) and **Autogrowth** (100%)
5. Click **OK**

Add Tables, Views, Functions, Procedures, Indexes, Primary Keys, Foreign Keys, and Constraints

1. Open the file GTFUNCTIONS.sql in a query window.

   NOTE
   The file is in the directory: C:\.....\datamaker_scrambleinstall\scrambleinstall\MSSQLSERVER \2005-2008-2012 directory, or 2000 for SQL Server 2000.
2. Choose **Scramble** from the database drop-down.
3. Click the **tick** icon on the toolbar to parse the SQL.

4. Verify that you get the message **Command(s) completed successfully** in the **Results**.
5. Click **Execute** to execute the SQL.
   There should be no errors in results. Verify any warning messages that appear.

Add the Data

1. In the Command Prompt window, enter the following command, and click **Entry Key**.
   REFDATA.bat
2. Use the one of the following parameters:
   - `REFDATA username password server target_database [owner] > refdata_log.txt 2> refdata_errors.txt`
   - `REFDATA TRUSTED server target_database [owner] > refdata_log.txt 2> refdata_errors.txt`
     The bcp commands runs for all tables in the Scramble database
3. Perform a **Find on Error** to verify that no bcp commands failed in the refdata_log.txt file.
4. The file refdata_errors.txt captures any errors from the DOS Commands.

Install Teradata Scramble Components

The scrambling functionality for Teradata requires the existence of a C Compiler on the client computer. The following procedure assume an installation of a Microsoft C compiler in a Windows environment.

1. Locate the path to the compiler executables CL.exe and LINK.exe
2. Create a configuration file compiler.cfg with paths, similar to the following example:
   ```
   CompilerPath: C:\Program Files\Microsoft Visual Studio\VC98\Bin\CL.EXE
   LinkerPath: C:\Program Files\Microsoft Visual Studio\VC98\Bin\LINK.EXE
   ```
3. Run the Teradata configuration utility and supply this configuration file as an argument. **Note:** Locate the cufconfig file in your Teradata install if the directory is not in your path.
   ```
   cufconfig -f compiler.cfg
   ```
4. (for Visual Studio 6 only) Set up the following registry entries:
   ```
   HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\VisualStudio\6.0
   "InstallDir"="C:\Program Files\Microsoft Visual Studio\Common\IDE\IDE98"

   HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\VisualStudio\6.0\Setup\Microsoft Visual C++
   "ProductDir"="C:\Program Files\Microsoft Visual Studio\VC98"
   ```
5. Run the functions.bat file from the `scrambleinstall/Teradata` directory.
   `Functions` *`TDPID username password Databasename`*
6. View the [functionName].log files for compiler errors and identify to incorrect environment settings.

Masking DB2 Cross Reference Columns

In some circumstances, columns join between tables on data types other than simple numerical IDs. In this case, you want to consistently mask the data in these table columns, and maintain the context and data integrity. For example, mask a credit card number so that the number is still the correct format for this credit card type.

Prerequisite Steps

1. Edit the script gtsrc_cross_reference.sql in your `Data Subset install\scramble` directory. Replace <DBNAME> with your DB2 schema name.

2. Run the script gtsrc_cross_reference.sql in your DB2 source connection. The source is the DB2 database that you are to extract and mask.

Set Scramble Functions

Mask column data consistently across tables in your schema;

Do one of the following:

- Select hash functions for basic numeric data
- Select functions that begin with CROSSREF

The Subset Steps

1. Select Run DB2 Windows Scramble as the action type.
2. (Optional) Run DB2 UNIX Scramble as the action type.
3. On the Data Scrambling tab, select your Datamaker project from the drop-down list.
4. Select your extract, and click Generate.

Generate Masking Scripts for SQL Server

As a testdata engineer, you follow the requirement that copies of your testdata must be masked. You want to use Datamaker to generate masked BCP scripts for SQL Server. Datamaker cannot directly generate masked BCP scripts, because BCP does not support function calls in queries. Set up your project to first create views containing the masking calls, and then produce BCP scripts against the views.

Define Transformations

1. Open Datamaker.
2. Create a project and register SQL Server tables.
3. Click **Projects, Transformation Maps**. The Transformation Maps window opens.
4. Click the Plus-sign to open the list of Transformation Maps.
   a. Click the Plus-sign to create a Transformation Map.
   b. Name the Transformation Map in the first column.
   c. Specify **SQL Server** in the DBMS column.
   d. Click the Checkbox to save your changes.
5. Select the Transformation Map that you created from the drop-down.
6. Click a table in the left pane and select appropriate transformations for columns.
7. Click Save.

Generate Extract Scripts

1. Open CA Data Subset and connect.
2. Select **Build MS SQL Server Scrambled Extract** from the Database Actions dropdown.
3. Specify the project and version, and log in to the repository.
4. Press the **Run** button.
   The Database Actions window opens.
5. Open the Extract Details tab and fill in the following fields:
   a. Define an **Action Name**, for example, MYACTION.
   b. Select your **Functions Database**.
   c. Verify that the option **Persist views in functions database** is enabled.
   Note: If you disable this option, the necessary temporary views are created in your production database.

d. Enter your source and target databases.
6. Open the Data Scrambling tab and select the Transformation Map.
7. Click Generate.

CA Data Subset creates several sql and batch script files.

Export Data to a Dump File

To export data, run the following batch script in your export directory.

`EXPORT\SQLSERVER\MYACTION\extractandload\MYACTION_export.bat`

The batch scripts expect the following parameters:

- If you do not have a trusted connection configured:
  *MYACTION*_export username password server database
  Note: The user account must have "create view/function/table" privileges for the scramble database.
- Alternatively, if you have a trusted connection configured:
  *MYACTION*_export TRUSTED *server database*

The script outputs dump files into the `extractandload` directory.

Import Data From a Dump File

To import a dump file, verify that the target table is empty, and run the following batch script in your export directory.

`EXPORT\SQLSERVER\MYACTION\extractandload\MYACTION_import.bat`

The batch scripts expect the following parameters:

- If you do not have a trusted connection configured:
  *MYACTION*_import *username password* server *target*_database [*target_schema]*
- Alternatively, if you have a trusted connection configured:
  *MYACTION*_import TRUSTED server *target*_database [*target_schema]*

The data is imported into the target database.

Mask XML in a Database Using CONCAT

As a Test Data Engineer, you want to mask XML data stored in a database column. These example procedures use the mask type CONCAT to concatenate values to mask an XML element.

CONCAT Syntax

When you define **Values or Columns**, there are four options for each parameter used in the concatenation. Options within the CONCAT parameter are separated by two tilde characters (~~).

CONCAT Syntax: `COLUMN_NAME~~XPATH~~substring_start,substring_length`

- **COLUMN_NAME**
  Defines the column from where to get values to mask.
- **XPATH**
  (Optional) Defines the XPATH value that identifies the XML element inside the column that you want to mask.
- ***substring_start***
  (Optional) Specifies the start position of the substring.
  – **,** ***substring_length***

Specifies the length of the substring. If you specify a value greater than the length of the string, *substring_length* defaults to the length of the string.

Syntax example 1: `NAME`

We mask just the column NAME or a fixed value. Default.

Syntax example 2: `NAME~~1,2`

We want to mask any XML element in the `NAME` column. We mask the substring from character position 1 to 2. If the length of the string is 1, then we mask only position 1.

Syntax example 3: `XML_DATA~~/Entry/entity-Person/LicenseNumber`

The XPATH value `/Entry/entity-Person/LicenseNumber` specifies the XML element that we want to mask in the `XML_DATA` column. We mask the whole string.

Syntax example 4: `XML_DATA~~/Entry/entity-Person/LicenseNumber~~2,20`

The XPATH value `/Entry/entity-Person/LicenseNumber` specifies the XML element that we want to mask in the `XML_DATA` column. We mask the substring from character position 2 to 20. If the length of the string is less than 20, then we mask from position 2 to the end of the string.

Example Masks

Fixed value

The string literal MYFIXEDVALUE is concatenated with the masked value for the NAME column, in this example, "Kiela".

Add column to mask: XML_DATA

- **Data Type:** Character
- **Mask Type:** CONCAT
- Enable **Use Masked Values**.
- **Value or Column:**
  - MYFIXEDVALUE
  - NAME
- **XML/JSON Data XPATH:** `/Entry/entity-Person/LicenseNumber`

Add column to mask: NAME

- **Data Type:** Character
- **Mask Type:** HASHLOV
- **Get Seed data:** from File
- **Data Category:** English Last Names

Data before:

```
<Entry>

  <entity-person>

    <FirstName>Amphitryon</FirstName>

    <LastName>Tiryns</LastName>

    <LicenseNumber>X123456</LicenseNumber>
```

```
  </entity-person>
```

Data after:

```
<Entry>
  <entity-person>
    <FirstName>Amphitryon</FirstName>
    <LastName>Tiryns</LastName>
    <LicenseNumber>MYFIXEDVALUEKiela</LicenseNumber>
  </entity-person>
```

Example: Substrings 1

In this example, we replace the first letter of the value of the XML element `/Entry/entity-Person/LicenseNumber` (X) by the first letter of the masked value of the XML element `/Entry/entity-Person/LastName` (K). The masked value in NAME is Keila.

Add column to mask: XML_DATA

- **Data Type:** Character
- **Mask Type:** CONCAT
- Enable **Use Masked Values**.
- **Value or Column:**
  - `XML_DATA~~/Entry/entity-Person/LastName~~1,1`
  - `XML_DATA~~/Entry/entity-Person/LicenseNumber~~2,20`
- **XML/JSON Data XPATH:** `/Entry/entity-Person/LicenseNumber`

Add column to mask: XML_DATA

- **Data Type:** Character
- **Mask Type:** HASHLOV
- **Get Seed data:** from File
- **Data Category:** American Female First Name

Data before:

```
<Entry>
  <entity-person>
    <FirstName>Amphitryon</FirstName>
    <LastName>Tiryns</LastName>
```

```
    <LicenseNumber>X123456</LicenseNumber>
  </entity-person>
```

Data after:

```
<Entry>
  <entity-person>
    <FirstName>Amphitryon</FirstName>
    <LastName>Tiryns</LastName>
    <LicenseNumber>K123456</LicenseNumber>
  </entity-person>
```

Example: Substrings 2

In this example, we concatenate the substring of the masked value in the NAME column from position 1 with length 3, with the substring from the value in `/Entry/entity-Person/LicenseNumber` from position 2 with length up to 20. The masked value in NAME is Keila.

Add column to mask: XML_DATA

- **Data Type:** Character
- **Mask Type:** CONCAT
- Enable **Use Masked Values**.
- **Value or Column:**
  - `XML_DATA~~/Entry/entity-Person/LastName~~1,3`
  - `XML_DATA~~/Entry/entity-Person/LicenseNumber~~2,20`
- **XML/JSON Data XPATH:** `/Entry/entity-Person/LicenseNumber`

Add column to mask: NAME

- **Data Type:** Character
- **Mask Type:** HASHLOV
- **Get Seed data:** from File
- **Data Category:** American Female First Name

Data before:

```
<Entry>
  <entity-person>
    <FirstName>Amphitryon</FirstName>
    <LastName>Tiryns</LastName>
```

```
    <LicenseNumber>X123456</LicenseNumber>
  </entity-person>
```

Data after:

```
<Entry>
  <entity-person>
    <FirstName>Amphitryon</FirstName>
    <LastName>Tiryns</LastName>
    <LicenseNumber>Kei123456</LicenseNumber>
  </entity-person>
```

Visualize Test Data Coverage

You do not want to rely on testing (copies of) production data: Apart from legal and privacy objections, production data has high volume (that means, it takes long to test) and low variance (that means, it contains few edge cases). You want to avoid overtesting common cases, and undertesting edge cases. There are more combinations than you can possibly test. Your goal is to identify which columns are relevant which cases you want your tests to cover, and what you consider 100 percent test coverage.

The Test Data Visualizer is a business intelligence-like utility that is designed to help you build better test data. Use it to identify missing combinations of data, analyze invalid sets of combinations, and compare the coverage of different environments (production data compared to QA data, compared to DEV data). The utility lets you measure coverage accurately and track progress. Data visualization is an essential tool for understanding where you want to generate synthetic data to ensure that you have sufficient data coverage across your test repository.

TIP

Tip: Ideal candidates for coverage visualization are columns that have only few, unique values (such as languages, currencies, user roles, or account types). You do not use it to visualize columns that contain large sets of unique values (such as names, IDs, addresses, phone numbers) because the result would be meaningless. Look at your test plan to identify the relevant test data attributes, and create a flattened abstraction of relevant columns to run the visualization on.

Open the Test Data Visualizer

To visualize your data coverage, use one of the following methods:

To visualize a CSV file:

1. Launch C:\Program Files\Grid-Tools\TestData\Visualizer\TestDataVisualizer.exe
   Test Data Visualizer opens.
2. Click CSV in the toolbar to load a CSV file.

To visualize data from a table in TDM:

1. Open TDM, and open any data source or data target.
2. Open a Tables node, and write, and execute any SQL statement.
3. Click the Visualize Data button (top right of the results table).
   Test Data Visualizer opens and loads the table.

Analyze Coverage of Data Attributes

Define two or more attributes in whose test data coverage you are interested.

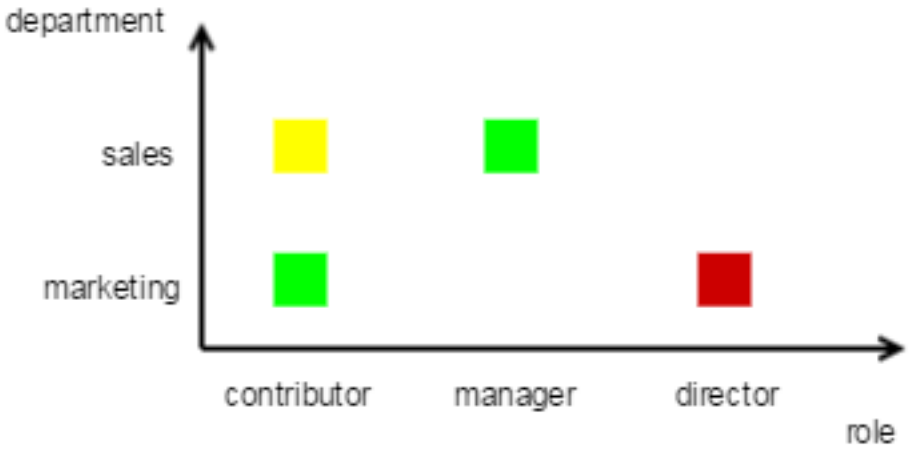
TIP

Tip: Use the Values column to identify attributes that have few unique values.

In this example, you have decided to analyze the coverage for the attributes role and department. This example refers to the default intervals and spot color settings.

1. Load a table into the Spot Graph tab and open the Data Attributes section.
2. Click the X-axis check box for the first attribute.
3. Click the Y-axis check box for the second attribute.
4. (Optional) If you inspect more than two attributes, assign related attributes to the same axis.
   Example: Assign `credit_card_name` to the same axis as `account_type`.
5. (Optional) Click **File, Settings** to configure colors and thresholds.
   - Define spot colors for your low, medium, and good coverage thresholds.
   - Add rows and conditions to visualize more fine-grained coverage thresholds.
6. Interpret the visualization:
   - The visualizer displays the coverage as a percentage in top of the diagram.
   - A green spot means good data coverage for testing these two attributes.
     Example: The table contains many rows for testing `sales manager` and `marketing contributor`.
     Result: Do not generate more test data of this type. You could test less data of this type, and could get equally good test results faster.
   - A yellow spot means medium data coverage for testing these two attributes.
     Example: The table contains enough rows for `sales contributor`.
   - A red spot means low data coverage for testing these two attributes.
     Example: The table does not contain enough rows for `marketing director`.
     Result: You can improve test coverage by generating more test data of this type.
   - No spot means that no data is available for testing these two attributes together.
     Example: The table does not contain any rows for testing `marketing manager` and `sales director`.
     Result: Focus on adding test data of this type to improve coverage considerably.

Figure 29: visualizing test data coverage in a 2D diagram



TIP

If you compare a large dataset, the diagram becomes harder to read. Drag the mouse to select a rectangular group of spots, then right-click and select Zoom to Region to change the display scale.

Reserve or Export Data

Drag the mouse to select a rectangular group of spots, or double-click an individual colored spot to inspect the rows that fulfill these criteria. You can perform the following operations against the selected data:

- Reserve rows. For more information, see Configure Test Data Reservation Service.
- Export to Agile Designer For more information, see CA Agile Requirements Designer.
- Export to CSV. Use this format to save data as spreadsheet.
- Export to Rally. For more information, see Test Matching Rally Integration.
  a. Enter your credentials and Rally instance and click Login.
  b. Browse a workspace or environment, select a story, and click **Export**.
  The test data is attached as .csv file to the story.

Filter and Compare

Switch to Test Factory to filter and compare data, and highlight relevant differences.

- **Filter Data**
  Refine the data that is visualized in the diagram. The Data Filter window lets you select and unselect individual values for each non-numeric data attribute, and define an interval for numeric values. Click Reset to remove all custom filters.
  – **Filter Axes**
  Enable this option to hide the column label from the visualization if no data exists for that column. If you disable this option, a column label remains in the visualization even if no data exists for the column. You typically use this option when you visualize missing values or export.
  – **Filter Inspector**
  Disable this option to stop applying the same filter to the data inspector window.
  – **Auto Update**
  Update the visualization while you edit the data filters. Disable this option to speed up performance while you define filters for a large set of data.
- **Load Override List of Variables**

Load more values for attributes that you know are missing from the test data. Loading these missing values adds them to the axes, so they are considered in the coverage calculation.

TIP

Tip: Use the Export Values operation to create a base CSV file as input template, edit it, and fill columns with values to add.

- **Load Invalid or Must Have Combinations**
  Highlight variable combinations between a source and a target set of data. A spot that is surrounded by a gray box means that the combination exists in both data sets. A spot without gray box means that the source data has better coverage than the target data. An empty gray box means that this combination in your target data is missing in the source data.
  Example use cases:
  – Compare production data (source) with QA data (target) to identify missing must-have combinations. First, load the target table and use the Export Values operation to create a CSV file. Then, load the source table and use the CSV file as input for the comparison.
  – Highlight invalid combinations that you do not want to test. Load the source table and use the Export Values operation to create a CSV file as template. Manually edit the CSV so that only invalid combinations remain. Then load the edited CSV as input for the comparison.
- **All Pairs Combination**
  Overlay an All Pairs Combination test suite over the display. Use this display to identify the coverage of all attribute pairs. Most bugs can be detected by testing interactions between all attribute pairs. It is not possible to test *all* attribute combinations, while testing *all* pairs is a realistic goal.
- **Visualize Missing Values / Restore Inverse**
  Display the inverse of the current data. The display now marks attributes that lack coverage with a green spot. Areas that have at least minimum coverage are not marked. Use this visualization to concentrate on identifying missing data.
- **Export Values**
  Export the data values shown on the display. Edit the exported CSV file, and use it as comparison input for the 'Override List of Variables', or 'Invalid and Must Have Combinations' displays.

Switch Graph Types

Click Add Graph in the Toolbar to display a different visualization that represents your data optimally.

- P-Coords (Parallel Coordinates)
- Spot Graph
- Bar Chart
- Pie Chart
- Radial Coverage
- Rect Coverage
- Data Table

Visualize Data Flow

Open the P-Coords Tab to use a Parallel Coordinates visualization to show how multi-dimensional data flows through the system. The goal of using parallel coordinates is to identify relationships between data dimensions. The thickness of a connector line represents the data coverage: A thick line stands for high coverage, and a thin line for low coverage of this attribute pair. No line means that an attribute pair is not covered.

Figure 30: visualization of values occuring together

Customize Settings

Click **File, Settings** to configure colors and thresholds for all graphs.

- Define custom spot colors to represent low, medium, and good coverage thresholds.
- Add rows and conditions to visualize more fine-grained coverage thresholds.

You can configure **Graph Options** for the SpotGraph.

- Color by property—By default, the visualizer uses colors to represent the Test Occurrence attribute; it uses three colors to highlight low, adequate, too much coverage relative to a given threshold.
  If you select any other data attribute, the visualizer uses unique colors to distinguish property values.
- Size by property—Disabled by default. Use this setting to represent different values by different dot sizes.
- Shape by property—Disabled by default. Use this setting to represent different values by different dot shapes.
- Horizontal zoom
- Vertical zoom

Generate Synthetic Test Data

Synthetic test data is data that contains all the characteristics of production, but with none of the sensitive content. CA TDM uses data profiling techniques to take an accurate picture of your data model. CA TDM uses this information to generate smaller, richer, more sophisticated sets of test data.

Using synthetic test data generation to provision data for testing helps you in the following ways:

- Eliminate the risk of data breach by creating production-like data without sensitive content.
- Reduce infrastructure by covering all combinations in the optimal minimum set of test data.
- Enhance existing subsets of production data with rich, sophisticated sets of synthetic data.
- Create large volumes of data for performance and load tests.
- Improve the efficiency of your testing.

Data generation functionality is available in Datamaker and the CA TDM Portal.

Generate Synthetic Data Using Datamaker

This section describes how to use the data generation functionality in Datamaker. Datamaker provides full data generation functionality, including:

- Data painter dialog for creating data generation rules
- Intelligence to derive rules based on data types
- Variables that can be set before publishing
- A variety of pre and post publish actions

Define Synthetic Test Data

You can use the Datamaker UI to define data. Defining data includes (for example) creating data, editing data, creating data pools, including generic test cases, and so forth.

Create and Edit Data

You can use the Datamaker UI to create and edit data. The data is held within the projects that you have setup. For example, you can browse through the test sets available in the project. These test sets include various test cases that contain the data.

Create Data

1. Open the Datamaker UI and select **Projects, Project Manager** from the main menu.
2. Navigate the project hierarchy in the left pane until you find the required data pool.
3. Double-click the selected data pool.
   The **Data Definition** dialog opens.
   The left pane shows a tree hierarchy that includes the **Used Tables** node. This node contains all the tables that are used in this particular test case. The **Unused Tables** node includes other registered tables that are not used in the selected test case.
4. To add data to a new table, click the table name under **Unused Tables** in the left pane.
   The table opens a new tab in the data editing dialog. The tab displays a white block next to its name. Tables without data are shown as white.
5. Click the plus icon to add a row to the table.
   Note: If you want to use the **Use All Pairs to create rows** option, see the Use All Pairs to Create Test Data section.
6. Enter the number of rows, press Enter, and confirm the number.
7. Double-click each cell in the table and add the appropriate information.
   You can type data into the column values. Data must conform to the data type of the defined table. For example, date type columns must contain dates, numeric types must contain number characters, and char type data can contain any characters up to the length of the column.
8. Save your changes.

Edit Data

1. In the Datamaker UI, to edit the data that already exists, click the table name under **Used Tables** in the left pane.
   The table opens a new tab in the data editing dialog. The tab displays a green block next to its name and the number of rows are shown in brackets; for example, (1000). Also, the data that is displayed in the table includes various substitution variables (for example, ~NEXT~) instead of the actual data. These variables are populated at the time of data publishing.
2. Click the drop-down list icon in the appropriate cell.

A list of values you can select from is displayed.

3. You can control the source of this list by clicking the Lov icon.
   The list of available values change depending on the options you select:
   - **Values from DDL**
     Specifies the check constraint values that are extracted from the schema.
   - **Values from Production**
     Specifies that the values are extracted data characteristics from production.
   - **Values from Development**
     Specifies that the values are extracted data characteristics from development.
   - **Invalid Values**
     Enters values that are defined as invalid.
   - **Used values from Test Data Repository**
     Specifies any occurrences of the column data already existing in other test cases.
   - **Used values from Data Target**
     Specifies any values that exist for this table column in the data target.
   - **Used values from Data Source**
     Specifies any values that exist for this table column in the data source.
   - **Related keys from Test Data Repository**
     Specifies the foreign keys or application relationships (if defined) that are used to display the related tables in the test case repository.
   - **Related keys from Data Target**
     Specifies the foreign keys or application relationships (if defined) that are used to display the related tables in the data target.
   - **Related keys from Data Source**
     Specifies the foreign keys or application relationships (if defined) that are used to display the related tables in the data source.

   Note: You can maintain this list of values by right-clicking in a column in **Projects, Actions for Registered Objects**.
4. Select the value and click the tick mark icon.
5. Save your changes.

In the UI, the color of the table row column indicates whether the data is inherited from the tables present in the **Includes** section:

- The gray color in the **Row** column indicates that the data is not inherited and has been entered at this level.
- The blue color in the **Row** column indicates that the data is inherited from an included table, but it has been changed locally.
- The light gray color in the **Row** column indicates that the data has been inherited from an included table.

Similarly, the color of the block next to the name of the tab represents the following information:

- Green block indicates that the table already contains data.
- White block indicates that the table does not include any data.
- Yellow block indicates that the table belongs to the Includes category.

Use All Pairs to Create Test Data

All Pairs is a mathematical technique where all combinations of values are included in the set with the minimum number of rows of data. Another term that is used is *pairwise testing*, which is a combinatorial software testing method. For each pair of input parameters to a system (typically a software algorithm), test all possible discrete combinations of those parameters.

1. Open the Datamaker UI and select **Projects, Project Manager** from the main menu.

2. Navigate the project hierarchy in the left pane until you find the required data pool.
3. Double-click the selected data pool.
   The **Data Definition** dialog opens.
4. To add data to a new table, click the table name under **Unused Tables**.
   The table opens a new tab in the data editing dialog. The tab displays a white block next to its name. Tables without data are shown as white.
5. Click the plus icon to add a row to the table.
6. Select the pair icon.
   A dialog displaying the columns in the current table with a count of potential values is displayed. The list of # values depends on which list of value options you have set.
7. Click the LoV icon to view the used list of values.
   Note: The **Used Values from Test Data Repository** drop-down list contains the **Project Version**, **Set**, **Test Case**, and **Specific Test Case** options.
8. Adjust the chosen sources for your lists of values and close the dialog.
9. Select the columns for which you want to build the **All Paired** combinations (or **All Combinations**) and click the tick mark icon. You can select from the following options:
   - **Use a standard All Pairs algorithm**
   - **Add all combinations**.
     If you select this option, it can quickly build a large list.

   The combinations are included in the data dialog.
10. Enter any other required columns in the missing columns and save the data.

Include Generic Test Cases

You can include existing data layers into the current data layer. This inclusion pre-populates your test case data with any data already contained in the generic test case. You can amend or delete the inserted data without affecting the original generic test case. If, however, the original generic test case is changed, test cases using the generic test case change unless a local edit is made to the test case.

1. Open the Datamaker UI and select **Projects, Project Manager** from the main menu.
2. Navigate the hierarchy in the left pane until you find the required data pool.
3. Double-click the selected data pool.
   The **Data Definition** dialog opens.
4. Select **Include Test Case** from the drop-down list and click the forward arrow icon.
   The **Select required Test Case** dialog opens.
5. Navigate the hierarchy and select the required test case.
6. Click the tick mark icon.
   The selected test case is included under the **Includes** item in the left pane.

Create Copies of Data Pools

You can create copies of data pools using the following types:

- Inherit data
- Copy data and inheritance
- Copy data without inheritance

At the end of the procedure an example is provided. This example explains the usage of all the three copy options.

1. Open the Datamaker UI and select **Projects, Project Manager** from the main menu.
2. Navigate the hierarchy in the left pane until you find the required data pool.

3. Right-click the data pool and select **Copy Data Pool** from the drop-down list.
   The **Copy Data Pool** dialog opens.
4. Enter the appropriate information in the **Name**, **Description**, and **Copy to** fields.
5. Select the required copy option:
   – **Inherit data from Data Pool**
   – **Copy data (and inheritance) from Data Pool**
   – **Copy data only (without inheritance) from Data Pool**
6. Click the **Run** button.

The following example explains all the use cases for the three copy options:

In this example, a data pool (DP1) contains one row in a table DEPT with three columns as shown in the following table:

Row	Deptno	Dname	Loc
1	1	Support	Office

- **Inherit Data**
  When you use the **Inherit data from Data Pool** option and create a second data pool (DP2), DP2 includes a single row but no column values. These values are inherited from its parent DP1. In this case, the row number shows in green in the **Data Definition** dialog.

Row	Deptno	Dname	Loc
1	1	Support	Office

  If you change the data in any column in DP2 (Loc value is changed to New Office), it is stored against DP2 and the row number now shows in blue to indicate that a change has been made:

Row	Deptno	Dname	Loc
1	1	Support	New Office

- **Copy Data with Inheritance**
  Now copy DP2 to a third data pool (DP3) with inheritance by using the **Copy data (and inheritance) from Data Pool** option. DP3 is now an exact copy of DP2, both inherited from DP1.

Row	Deptno	Dname	Loc
1	1	Support	New Office

- **Copy Data Without Inheritance**
  Now copy DP3 to a fourth data pool (DP4) without inheritance by using the **Copy data only (without inheritance) from Data Pool** option. The row number background is gray as there is no inheritance:

Row	Deptno	Dname	Loc
1	1	Support	New Office

  There is no change to the display of DP1 in the **Project Manager** dialog as it has no link to DP4.

Use Case: Change the Data in Data Pool 1 (DP1)

If you change the data in DP1 as described in the following table:

Row	Deptno	Dname	Loc
1	1	SUPPORT	Basement

2	2	Sales	Penthouse

DP2 looks like the following table with the change in Row 1 of Dname, but not the change to Loc, as the change made in DP2 takes precedence:

Row	Deptno	Dname	Loc
1	1	SUPPORT	New Office
2	2	Sales	Penthouse

DP3 still shows as an identical copy of DP2; the following table shows this information:

Row	Deptno	Dname	Loc
1	1	SUPPORT	New Office
2	2	Sales	Penthouse

And, DP4 is exactly as it was before because it is not inherited. The changes to DP1 have no effect; the following table shows this information:

Row	Deptno	Dname	Loc
1	1	Support	New Office

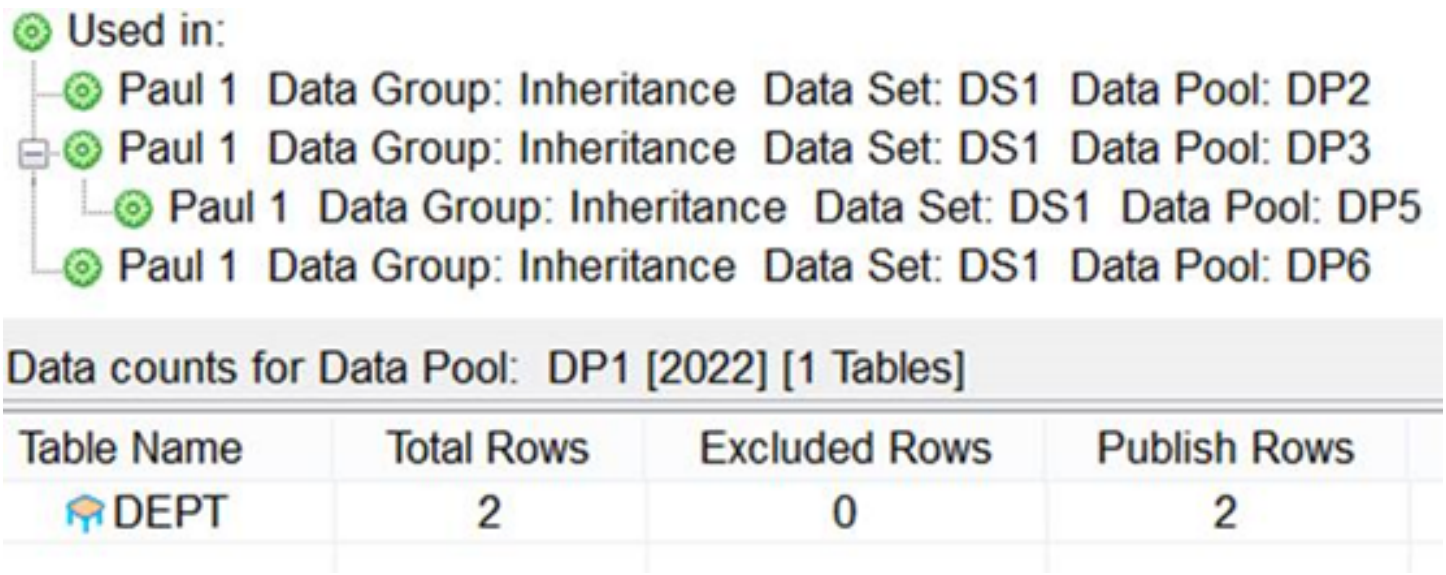
Use Case: Copy Data and Inherit

Create a fifth data pool (DP5) as a copy (Inherit Data) of DP3. Both rows are green—unchanged from their immediate parent data pool (DP3).

Row	Deptno	Dname	Loc
1	1	Support	New Office
2	2	Sales	Penthouse

Whereas, if you create data pool 6 (DP6) by copying data with inheritance, row 1 shows in blue as it differs from its immediate parent.

If you view DP1 in the **Project Manager** dialog, it shows all the inheritance as shown in the following screen shot:



Edit Data Creation Functions

CA TDM provides a series of data creation functions that you can edit and test.

1. Open the Datamaker UI and select **Projects, Project Manager** from the main menu.
2. Navigate the hierarchy in the left pane until you find the required data pool.
3. Double-click the selected data pool.
   The **Data Definition** dialog opens.
4. Double-click the table in the left pane.
   The table opens in a tab in the right pane.
5. Double-click in a column.
   The **View/Edit data in column** dialog opens.
6. Use this dialog to edit and test the data functions. The dialog allows you to click on objects in each of the three panes. These objects transfer to the edit section where they are manipulated. The three sections are as follows:
   Note: Expressions in this window have a limit of 16000 characters.
   - Functions
     Functions can use hard-coded values, columns, or variables as parameters. Functions can also use other functions as parameters. For example, you can use a function as a result from a Boolean operator in the IF function.
   - Columns
     The columns list contains any other columns in the table or other tables. Click the **Other Tables** option to expand the list of columns from other tables.
   - Variables
     The variables column contains a mixture of system operators (prefixed with a star) and any substitution variables you have created.
7. Click the help on functions icon to find more information about the list of available data editing functions. Some common features to these functions are as follows:
   - Percnull
     This function allows you to identify the percentage of rows that are null. Selecting 20 means 20 percent of the values are designated null.
   - Sources

The valid values are R = Repository, S = Source, T = Target. When this function is run, it accesses the database connection to run the appropriate SQL. For example, seqlov(0,S) looks for a sequential list of values based on your SOURCE connection.

– Sources (LoV)
In the **List of Values options** dialog, you see that each source of value has an identifier, marked from A to J. You can include these values in any of the data sources. Therefore, randlov(0,DG) produces a random list of values using the invalid values and values used in the data source.

8. Review and save your changes.

Note: You can select the first row value for the column by including (1) after the column name. For example, ASSIGNED_SEAT(1) provides the value of the first row. You can use the value of a previous row by adding (-1). This approach is useful if you want to sum a value through the rows. For example, in the column CUR_AMT, set the function to @SUM(cur_amt,transaction_amt(-1))@. This function adds the value of transaction_amt from the previous row to the cur_val column.

Create Substitution Variables

When you edit test data in the repository, you can use variables that are substituted when you publish the test data. Variables have the format '~variablename~'. The two types of substitution variables are *Standard* and *User Defined*.

- *Standard* variables are standard functions that you use to manipulate data when you publish.
  For example, '~SDATE~' resolves to the current system date, and '~CDATE~' resolves to the current date as defined in the project settings.
- *User Defined* variables are created by the user and allow you to substitute specific application variables.

You can change the default value of a substitution variable when you publish the data.

Variable Scope

A variable is available only on the level where it was defined, and below. You add variables with datagroup, dataset, or datapool scope on the datagroup, dataset, or datapool level, respectively. Additionally, you can add variables with project or even repository scope from any level. A variable on repository level has the widest scope and it is accessible in all projects.

Example: You create a variable `abc` at project level and set its value to `~CD~` . You create another variable `abc` at dataset level and set its value to `123` . At datapool and dataset level, `abc` now resolves to `123` . At datagroup and project level, `abc` resolves to `~CD~` . The variable is only accessible in the one project where it was defined.

Variable Resolution

A variable value can be dynamic, for example, if it is a function such as `@randrange(1,20)@` . If a variable is used multiple times within a row, and the table repeat count is larger than 1, and the variable value is dynamic, you must define how and when Datamaker resolves the repeated variable. In the New Variable dialog, choose *one* of the following behaviors:

- No selection (default) — The repeated variable is resolved each time it is used.
- Resolve Prior to Publish — The repeated variable is resolved once per publish and the value is the same in all rows.
- Resolve Per Published Row — The variable is resolved once per row and the value is the same within one published row. Values can be different in other rows.

Create Substitution Variables

1. Open the Datamaker UI and choose **Projects, Project Manager** from the main menu.
2. Navigate the hierarchy in the left pane and drill down to either project, datagroup, dataset, or datapool level.
3. Click **Variables**.

All the variables available on this level are displayed in the right pane.

4. Right-click in the right pane and select **New Variable** from the context menu.
5. Specify variable details in the dialog.
6. Validate the variable by clicking the tick mark icon in the **Validation** area.
   The available validation functions are as follows:
   - IN(…,…)
     Specifies that the only valid values are in this list.
   - MIN(…)
     Specifies that the value cannot be less than this value.
   - MAX(…)
     Specifies that the value cannot be more than this value.
   - RANGE(…,…)
     Specifies that the value must be within this range.
7. Save your changes.

Create Substitution Variables Dynamically

If you enter a substitution variable into your data as it is being edited, Test Data Manager verifies whether it already exists. If it does not exist, it prompts you to create a new variable. You can perform one of the following actions:

- **Ignore**
  If you ignore the message, you can return later to create a substitution, or enter a value during data publishing.
- **Edit Data**
  If you edit the data, you can correct or remove a variable.
- **Create Substitution**
  If you click **Create Substitution**, you are prompted to add the substitution to the local level (Set Level) or to the global (Project) level. If no local level exists, the substitution is created for you.

Available standard substitution variables are as follows:

- **~CD~**
  Specifies the day of the current date.
- **~CDATE~**
  Specifies the user-specified current date as defined in the project settings.
- **~CM~**
  Specifies the current month.
- **~COLNUM~**
  Specifies the ordinal position of the column in the table.
- **~COLUMN_NAME~**
  Specifies the name of the column in the table that you are editing.
- **~CY~**
  Specifies the year of the current date.
- **~EMPTY~**
  Shows an empty string.
- **~ITERATION~**
  Specifies when you publish data multiple times using either a control file or by entering the number of iterations in the publish screen.
- **~LD_DESC~**
  Specifies the description of the publish details.
- **~LD_ID~**
  Specifies the numeric ID of the publish level.
- **~LD_NAME~**

Specifies the name of the publish level.

- **~MAX~**
  Changes the maximum value in a cell.
- **~NEXT~**
  Specifies that the next highest value of the column is found when you publish the data.
  Note: ~NEXT~ can only be used in one column per table.
- **~NEXTSUB~**
  Specifies that the function is used with ~NEXT~ if a multi-part key is used. When the ~NEXT~ value is set, the next part of the key is set to 1 and incremented until the ~NEXT~ value is updated.
- **~PUBROW~**
  Specifies the published row number. The published row number is a combination of the rownum plus the iteration minus one times the total number of rows to be published.
- **~ROWNUM~**
  Specifies the number of the row for the table.
- **~SDATE~**
  Specifies the date as defined by the system.
- **~SDATETIME~**
  Specifies the date and time as defined by the system.
- **~SPACE~**
  Enters a space within a string.
- **~STIME~**
  Specifies the time as defined by the system.
- **~STIMESTAMP~**
  Specifies the time as defined by the system, to fractions of a second.
- **~TABLE_NAME~**
  Specifies the name of the table you are editing.
- **~USER~**
  Specifies the CA TDM user for the Datamaker UI.
- **~WINUSER~**
  Specifies the Windows user that is logged in.

Publish Data Using Datamaker

Publishing data implies creating data; that is, you create data based on the criteria defined for the data. You can create data directly into a database, flat file, API call, SOAP Harness, or REST protocol.

- The two **Enterprise** submission methods support all file formats that CA TDM Portal publish supports; for a list of supported file formats, see Publish Data Using the CA TDM Portal.
- You can use the **Immediate** publish engine for quick, small, immediate publishes. The Immediate submission method supports all file formats listed below.
- If you execute a .bat file that you have created in a previous version of Datamaker, Datamaker defaults to the non-enterprise publishing mode, and the publish job is executed by the Datamaker publish engine rather than the by the CA TDM Portal publish engine. For more information see Publish in Batch Mode.
  Do one of the following to enable Enterprise publishing mode for old .bat files:
  – Update the old publish job XML file from Datamaker by choosing **Batch** mode when publishing.
  The log will include the line "`Changes to handle non supported Enterprise Publish Option`".
  – Edit the old publish job XML file manually and add the following two lines between the <GTDatamaker> and </GTDatamaker> tags:

```
<use_enterprise_mode>true</use_enterprise_mode>
```

```
<tdm_portal_url>
https://
YOUR_TDM_PORTAL_URL
</tdm_portal_url>
```

Follow these steps:

1. Open the Datamaker UI and select **Projects, Projects Manager** from the main menu.
2. Navigate the hierarchy in the left pane and select the appropriate data pool in the left pane.
3. Select **Test Data, Publish Data** from the main menu.
   The publish dialog opens. Select a **Submission Method** for this publish job:
   - **Enterprise Mode**—Use this publish option to create large scale data. This option is the default.
   - **Immediate**—Use the publish engine built into Datamaker for quick, small publishes.
   - **Batch**—Use this publish option to create scheduled data.
   - **Remote**—Use this publish option to create scheduled large scale data.
4. Perform the following actions as appropriate:
   - Publish Data Multiple Times
   - Repeat Table Data Multiple Times
   - Publish Data with Substitution Variables
   - Create a Starter CSV file using the Datamaker UI
   - Publish to Appropriate Formats
   - Publish to Multiple Schemas
5. Click the forward arrow icon in the top-right corner of the dialog to complete the publishing.

Note: If you try to publish data to your Hadoop/Hive target database, Hadoop/Hive inserts NULL in a column if the generated value is not valid for the column. For example, for Hadoop/Hive, the valid value for the TINYINT data type is a 1-byte signed integer ranging from -128 to 127. Therefore, for any generated value outside of this range, Hadoop/Hive inserts NULL in the column.

Publish Data Multiple Times

Enter the number of times you want to repeat publish in the **Repeat** field.

Note: If you are not using variables such as ~ITERATION~ or ~NEXT~, you can get duplicate values in your data output.

Repeat Table Data Multiple Times

If you want to repeat individual tables in a set of data:

1. Right-click the table name and select **Table repeat count** from the context menu.
2. Enter the number of times you want to repeat this table. You can also use functions, from the Functions pane, to vary the number of times a particular table is repeated. For example, @randrange(1,5)@ and @randnorm(1,100,70,10)@ If you have created large volumes of data within a table, it is also possible to publish a small selection of rows. For example, enter -3 in the field publishes up to three rows from the selected table.

Publish Data with Substitution Variables

When you publish data, you can control the publish by importing lists of substitution variables from the sources that are mentioned in the **Values for Variables** section—default values, from a CSV file, from a test case, or from SQL. Different ways to use the substitution variables for publishing are as follows:

- **Importing Variables from CSV files**

Select the **from file** option and browse for the CSV file you want. This file performs one publish per row. If you have a repeat set, each row in the CSV iterates multiple times. Ensure that the CSV file follows the following format:

```
TRAVELDATE,CURRENCY,CREDITCARD,FLIGHTCOST,FLIGHTPRICE,TICKETTYPE

2008-03-01,GBP,AX,100,110,900

2008-03-02,GBP,VI,110,110,900

2008-03-03,GBP,VI,120,110,900

2008-03-04,GBP,AX,130,110,900

2008-03-05,GBP,VI,140,110,900

2008-03-01,USD,AX,150,110,900
```

Each substitution variable that you want to modify must be in its own column. If you do not include a substitution variable, the default value is used.

- **Import Variables from a Test Case**
  Select the **from Test Case** option to browse through all available test cases in the project to source your variables.
- **Import Variables from SQL**
  Select the **from SQL** option, then select whether to import from the target, source, or repository. A drop-down list of available SQL from which you can select your variables displays.
- **Use Repeat in CSV Control Files**
  You can add a special reserved column that is named REPEAT to a CSV file. This column controls the number of iterations. Each row in the CSV is considered as a separate publish.
  An example CSV file that includes REPEAT is as follows:

```
REPEAT,CREDITCARD,CURRENCY,FLIGHTCOST,FLIGHTPRICE,TAXRATE

"10","AX","GBP","75","100","0.10"

"5","VI","GBP","75","100","0.10"

"50","MC","USD","80","100","0.10"
```

- **Using Excel Spread sheets to control publishes**
  You can convert your CSV file to a spreadsheet and can use this file to publish the data. However, set the name of the worksheet and the location of the row header and starting data file. To do so, go to **Settings > Excel Parameter Settings** from the main menu. Enter the name of the worksheet. You also need to specify the row with the column headers and the row number where your data starts. The following segment is an example of a spreadsheet that is used to control a publish:

```
Please enter the test cases you want starting at row 5


Name:                                Email:
```

MyTestName	Loyality	Location	Sex	Shipping Class
Huw Invalid1	1	FL	F	Class3
Huw Valid1	2	CA	F	Class3
Huw Valid11	2	CA	F	Class2
Huw Valid12	1	FL	F	Class1

The column headers are in row 4 and have been hidden from the user.

Please enter the test cases you want starting at row 5

Name: **Email:**

MyTestName	Loyality	Location	Sex	Shipping Class
TEST_NAME	Loyality	Location	Gender	Shipping
Huw Invalid1	1	FL	F	Class3
Huw Valid1	2	CA	F	Class3
Huw Valid11	2	CA	F	Class2
Huw Valid12	1	FL	F	Class1

- **Storing CSV Control Files within CA TDM**
  You can maintain the CSV file using Microsoft Excel as long as you save it back in the .csv format. You can also store the CSV file in CA TDM. You can then publish the CSV file and use it to control another publish.

Publish Data-Only Generators Without Resolving Metafunctions

You as Test Data Engineer may want to create data-only generators, so you can use them as an intermediate publish job to execute further activities. You expect to be able to put any text into the painter and it will not be resolved as an expression. For example, you want expressions in the table counter to be resolved, but data inside the data pool itself not.

In Datamaker, such a publish fails if the data-only generator contains expressions in columns which are not of string type, for example the expression ~NEXT~ in a numeric column.

In CA TDM Portal, such a publish works when the target is not a database. For example, publishing to a CSV file, or to another Data Pool. When publishing to a target database table, however, the publish displays an error, because an unresolved string expression cannot be inserted in a non-string table column.

Create a Starter CSV File Using the Datamaker UI

If you click the save icon in the bottom-right corner of the publish dialog, the **CSV Variables** dialog containing a list of the current substitution variables with their default values is displayed. If you click the save icon on the **CSV Variables** dialog, you can save a starter CSV file. You can then open the CSV with Microsoft Excel and create extra rows (each row being an individual publish) as you require.

Publish to Appropriate Formats

Select the appropriate format from the drop-down list in the top-right corner of the data publish dialog. Some of the options are as follows:

- **Publish to SQL**
  SQL insert statements are built and various control statements are added depending on the database you are connected to. For example, if you are connected to Oracle, the file includes SQLPlus spool statements.
- **Publish to Excel (or XLSX)**
  This option takes the multiple tables and creates a single Microsoft Excel/Open Format XML spreadsheet with multiple workbooks.
- **Publish to XML**
  This option creates each row as an XML object.
- **Publish to HTML**
  This option creates a basic HTML page.
- **Publish to CSV**
  This option creates a comma-separated file with headings.
- **Publish to TXT**
  This option creates a text file with columns separated by tabs.
- **DDL for Used Tables**
  This option builds a SQL script containing all the table definitions to be published.
- **Publish to Target**
  This option inserts the data directly into your target connection. If the data publishes correctly, the table name is prefixed with a tick mark icon in the data publish dialog. This icon represents that the columns match and that data publishes correctly. If mismatched tables exist, you can identify them by clicking the table name.
- **Publish to Test Case**
  This option allows you to publish data from one test case to another. When you select **Publish to Test Case** and publish, a dialog opens to let you select the required test case. This feature lets you generate multiple rows of static data that a SoapUI or QTP can use to store actual and expected values. For example, you can generate a test case with all Zip Codes in the US. You can use this static test data to feed a SoapUI data harness issuing requests and storing responses for each static row of data.

Publish to Multiple Schemas

You can also publish individual tables to separate schemas in a single publish. In the publish dialog, right-click the table that you want to publish and select the **Specify table location** option from the context menu. A dialog opens and displays a choice of other connection profiles (containing connection properties to your alternative schema).

Understand Data Multiplier and Data Bulking

The Datamaker UI allows you to generate rich sets of data directly into your database. After you create the *data shapes* you need, you can add more data (data bulking) using the Data Multiplier script. This method is useful when you want to increase transactional data for performance testing. You would not use this method for reference tables, such as customer or products, though. This method is, however, useful for building a 100 million orders and items, for instance.

The Data Multiplier script contains SQL statements that you can run directly against the database to double the amount of data each time you run it. So, if you have generated a million rows, you can double this number to 2 million, then to 4 million, and so forth. Therefore, by running the script 5 times (for example), you can create an extra 63 million rows. This method allows you to create data quickly as there is no external file activity to slow the creation.

Save and run the generated script against the database. The script handles assigning new IDs and creating rows with referentially intact relationships. You can also add defined Transformation Maps to the generated scripts. These function maps allow you to add extra randomization or data conditioning to the multiplied data. You can, for example, quickly create many customers based on your existing customers, but assign new random names and addresses to the new customers.

To build the script, follow these steps:

1. Select or create a test case with the transactional tables you need. Ensure that the table relationships are defined and that one row of data containing ~NEXT~ and ~PARENT(1)~ exists.
2. On the publish dialog, select the **Data Multiplier** option from the drop-down list in the top-right corner.
3. Click the forward arrow icon next to the drop-down list.
   A script is created.

An example of a script that is generated is as follows:

```
-- SQL Data Muliplier Script - 2009/09/02 - 11:43:00
spool data_multiplier



whenever sqlerror exit



COLUMN ORDER_ID_ORDERS__RANGE NEW_VALUE ORDER_ID_ORDERS__MOD
BREAK ON ORDER_ID_ORDERS__RANGE
SELECT MAX(ORDER_ID) - MIN(ORDER_ID) + 1 ORDER_ID_ORDERS__RANGE FROM ORDERS;



INSERT INTO ORDERS (
         ORDER_ID,
         ORDER_DATE,
         ORDER_SHIPPED_DATE,
         ORDER_STATUS_CODE,
…
         OBJECT_VERSION_ID,
         OBJECT_NOTES )
SELECT ORDER_ID + &ORDER_ID_ORDERS__MOD,
```

```
          ORDER_DATE,
          ORDER_SHIPPED_DATE,
          ORDER_STATUS_CODE,
…
          ORDER_TOTAL,
          OBJECT_VERSION_ID,
          OBJECT_NOTES
FROM ORDERS;
COMMIT;



SELECT to_char (COUNT(*)) || ' ORDERS rows' ORDERS_COUNT FROM ORDERS;
COLUMN LINE_ITEM_ID_ORDER_ITEM__RANGE NEW_VALUE LINE_ITEM_ID_ORDER_ITEMS__MOD
BREAK ON LINE_ITEM_ID_ORDER_ITEM__RANGE



SELECT MAX(LINE_ITEM_ID) - MIN(LINE_ITEM_ID) + 1 LINE_ITEM_ID_ORDER_ITEM__RANGE
FROM ORDER_ITEMS;



INSERT INTO ORDER_ITEMS (
           ORDER_ID,
           LINE_ITEM_ID,
           PRODUCT_ID,
           QUANTITY,

…
           LINE_TOTAL,
           OBJECT_NOTES )
SELECT ORDER_ID + &ORDER_ID_ORDERS__MOD,
LINE_ITEM_ID + &LINE_ITEM_ID_ORDER_ITEMS__MOD,
           PRODUCT_ID,
           QUANTITY,

…
           LINE_TOTAL,
           OBJECT_NOTES
FROM ORDER_ITEMS;
COMMIT;



SELECT to_char (COUNT(*)) || ' ORDER_ITEMS rows' ORDER_ITEMS_COUNT FROM
ORDER_ITEMS;
EXIT
```

The script varies from database type to database type. This example is an Oracle script that you must run using SQLPLUS. Other database types create temporary tables to identify the increment to any primary key columns.

The generated script calculates the difference between the minimum and maximum values of a key column. For example, if ORDER_ID has a minimum value of 10 and a maximum of a 100, the difference is 90. The INSERT clause selects the existing Orders, but adds 91 to the ORDER_ID, so you get values of 101 to 191 added. The ORDER_ITEMS similarly has the LINE_ITEM_ID incremented; however, the ORDER_ID is incremented by 91.

The result of this is that the new orders are created with a new set of Order_Lines. The new Orders and Order_Lines refer to the original products and have the exact same data apart from the primary keys and any key values linking the data.

If you are running a query on products, you now have twice as many orders for the product. So if you have created a rich set of order types for each product, you can then bulk up the number of orders using this technique.

The following segment includes an example of a script output:

```
data_multiplier.LST

ORDER_ID_ORDERS__RANGE

----------------------

482

old 26: SELECT ORDER_ID + &ORDER_ID_ORDERS__MOD,

new 26: SELECT ORDER_ID + 482,

423 rows created.

Commit complete.

ORDERS_COUNT

----------------------------------------

846 ORDERS rows

LINE_ITEM_ID_ORDER_ITEM__RANGE

------------------------------

670

old 14: SELECT ORDER_ID + &ORDER_ID_ORDERS__MOD,

new 14: SELECT ORDER_ID + 482,

old 15: LINE_ITEM_ID + &LINE_ITEM_ID_ORDER_ITEMS__MOD,
```

```
new 15: LINE_ITEM_ID + 670,

646 rows created.

Commit complete.

ORDER_ITEMS_COUNT

-------------------------------------------

1292 ORDER_ITEMS rows
```

Exclude Columns from Publishing

You can exclude columns from the publish process. This ability lets you create extra columns for deriving data that do not appear in the data output. The Datamaker UI creates excluded columns (ID and PARENT_ID) when registering certain file types (such as XML) to hold the links between the record types.

The following list shows the allowed values—one for each publish type and ALL for global exclusion:

- ALL
  Represents all publish types.
- CSV
  Represents comma-separated variable files.
- DB
  Represents the target database.
- DBU
  Represents the target database updates.
- FD
  Represents the file definition.
- HIPAA
  Represents Health Insurance Portability and Accountability Act (HIPPA).
- HTML
  Represents Hyper Text Markup Language (HTML).
- JSON
  Represents JavaScript Object Notation (JSON).
- LOC
  Represents local publish.
- REP
  Represents repository.
- REST
  Represents Representational State Transfer (REST).
- SQD
  Represents SQL deletes.
- SQL
  Represents SQL inserts.
- SQU
  Represents SQL updates.
- STF

Represents source to file.
- TXT
  Represents tab-separated text.
- XLS
  Represents Microsoft Excel (older versions).
- XLSX
  Represents Microsoft Excel.
- XML
  Represents Extensible Markup Language (XML).

You can concatenate multiple types using a colon. For example, JSON:XML excludes a column from publish to JSON or XML, but not from other publish types.

Note: The exclusion is set in the **Maintain Table** dialog. You can access this dialog by right-clicking on a table in the left hierarchy of the **Actions for Registered Objects** dialog.

Manage Publish and Ad hoc Actions

You can add various actions to the project items available in the project hierarchy:

- The **Ad hoc action** type can be executed whenever you want.
- The **Pre-publish action** type is executed before the publish. Use it, for example, to clear specified tables.
- The **Post-publish action** type is executed after the publish. Use it, for example, to update some columns using SQL.

You can create actions for the following **Code Types**:

- **DDL** — Executes valid SQL and DDL (Data Definition Language) against a specific profile.
- **SQL** — Executes valid SQL against a specific profile.
- **Export** — Stores an Oracle export in the repository and you can import it as required.
- **Host** — Calls a specific external program. Can run synchronously or asynchronously.
- **Javelin** — Runs an external Javelin script. Can run synchronously or asynchronously.

Add an Action

1. Open the Datamaker UI and click **Projects, Project Manager** in the main menu.
2. Right-click the project manager item to which you want to add actions, and click **Maintain actions**.
   Note: Actions that you define become available to other items lower down the tree structure. For example, if you create a Post-Publish action at the **Version** level, all the test cases for that version execute the action after each publish.
   The **Actions** dialog opens.
3. Click the plus icon and then click **Yes**.
4. Select the code type from the **Code Type** drop-down list, and click the tick mark icon.
5. Enter a name and description for the action.
   Note: We strongly recommend not to create any actions with duplicate names in Datamaker after the TDM Portal installation. TDM Portal installation renames the duplicate action names to avoid any possible errors. TDM Portal does not allow duplicate action names.
6. Enter other appropriate information in the actions dialog and save your changes.
7. The action is added under the selected item in the tree structure.
8. To access the action for any item, right-click the selected item, and click **All available Actions** from the context menu. All actions applicable to the selected item are displayed.
9. Click the required action to run the action.

Run External Processes Synchronously (Host, Javelin)

By default, DataMaker runs actions asynchronously and does not wait. When running external action types (such as Host or Javelin) asynchronously this means that actions do not return results back to DataMaker. It is therefore possible that the job finishes before the action completes.

You have the option to run actions of type Host and Javelin synchronously.

1. Right-click the action and choose **Edit *name* Action**.
2. Enable the **Synchronous Call** option.
3. Specify the **Time to Wait** before the action reports back its result and DataMaker starts the next action.
   - ***n*** — Waits n seconds before returning the result.
   - **-1** — Runs the first action until completion before starting the next action.
   - **0** — Runs actions asynchronously. Datamaker does not wait for a result (default).
4. Specify whether to **Terminate on Timeout**, that is, what to do if the action runs longer than the Time to Wait.
   - **Enabled** — Datamaker terminates the timed-out process and returns a failure.
   - **Disabled** — Datamaker carries on and considers the action successful even if it has not completed.

Examples

- **Update email address**
  You can define a post-publish action of code type SQL that updates email addresses. You can use the following code to update the email address:
  ```
  update people set email=first_name||'.'||last_name||'@company.com'
  ```
- **Invoke a program**
  You can define an ad hoc action of code type Host that invokes Internet Explorer and displays a website. You can use the following program call:
  ```
  "c:\program files\internet explorer\iexplorer.exe"http://www.broadcom.com
  ```
- **Include substitution variables (Post-publish Actions only)**
  You can also include any predefined substitution variables in the body of Post-publish Actions; for example,
  ```
  C:\gts\creord\double\loadup.bat\ ~OracleSID~ ~LoadSchema~
  ```
  When you execute the Post-publish Action, you are prompted for the variable values.

Publish in Batch Mode

CA TDM lets you publish data in a batch mode. You can publish data without running the user interface. This ability helps you build scripts that are controlled by dynamic run parameters and also combine multiple publishes so that you can quickly prepare a test environment. This feature also allows you to interface the tool with products such as Quality Center from HP. You can pre-populate data for matching test scripts being run under the control of Quality Center.

Prepare and Generate the XML Control File

CA TDM uses a control file in XML format to publish in batch mode. Use the publish dialog to create this XML file. After you click the forward arrow icon on the publish dialog, select the **Batch** option, and click the forward arrow icon to generate the batch files.

Two files are created; a windows command file and .xml control file. You can edit these files and can move them to a server for submission. You can also use them with the batch .csv submission control file.

An example format of the created .xml file is as follows:

```
<GTDatamaker™>
```

```
<rep>GTREP</rep>

<tgt>Travel_e</tgt>

<src>Travel</src>

<ld_id>1260</ld_id>

<publish>true</publish>

<connect_rep>true</connect_rep>

<connect_tgt>true</connect_tgt>

<connect_src>true</connect_src>

<username>Administrator</username>

<password>marmite</password>

<onduplicate>continue</onduplicate>

<publishto>TGT</publishto>

<outfile>\1260</outfile>

<outdir></outdir>

<infile></infile>

</GTDatamaker™>
```

If you want to force a specific CDATE value, include the following XML:

```
<cy>2010</cy>

<cm>08</cm>

<cd>09</cd>
```

The description of various parameters is as follows:

- Rep
  Specifies the exact name of profile that is used to connect to the repository.
- Tgt
  Specifies the exact name of profile that is used to connect to the target.
- Src
  Specifies the exact name of profile that is used to connect to the source.
- infile

(Optional) Specifies the name of a .csv control file containing substitution variables.

- outfile
  Specifies the name of the log file and any output files to be created.
- ld_id
  Specifies the test case ID. Right-click on the context line of the GUI to obtain this ID
- publish
  Specifies whether to publish. True publishes the data and false allows you to test whether the script is correct up to the point of publish.
- connect_rep
  Identifies whether to connect to the repository or not.
- connect_tgt
  Identifies whether to connect to the target or not.
- connect_src
  Identifies whether to connect to the source or not.
- Cy
  (Optional) Specifies the current year.
- Cm
  (Optional) Specifies the current month.
- Cd
  (Optional) Specifies the current day.
- Publishschema
  (Optional) Specifies the schema for publishing.
- username
  Specifies the Datamaker UI user to log in.
- password
  Specifies the Datamaker UI user password.
- onduplicate
  Specifies whether to exit or continue when duplicate is detected. Exit fails immediately at the first duplicate; continue carries on to the end of the publish where duplicates are logged.
- publishto
  Specifies the destination of your published data. Values are TGT or SRC or FILE. If the option FILE is selected, the parameter OUTFILE must be also set.
- outdir
  Specifies the directory to which all output files are directed.
- max_lov_items
  Specifies the maximum number of items that are allowed in each list that the LOV processing creates.
- iterations
  Specifies the number of times you want to repeat the publish.

Move the Control Files to Another Computer

You can move the .xml, .csv (if used), and .bat file to another computer and can run it from there. Verify and amend the following items:

- Verify the exact location and driver of any hard-coded paths in the .bat and .xml file.
- Ensure the <tgt>, <src>, and <gtrep> connection profiles exist, spelt the same way (including case), and connect to the correct schema.

Propagate Seed List Data Across Masking Engines

As a test data engineer, review this article to know how you can propagate seed list data across various masking engines for different databases.

This article expects the input in the form of a structured CSV file and provides appropriate SQL scripts for all the different databases. The rationale behind adopting this approach is that the schema is different across different databases; therefore, exporting from one table and importing into another table may or may not work. Also, the input data and the SQL commands vary.

The information in this article represents a generic use case. The objective is to help users understand this generic information and follow a similar approach for their specific requirements.

Understand the Current Seed List Repository

Currently, the seed list that is shipped with CA TDM is available at the following places:

- Repository database (GTREP)
  The Datamaker component of CA TDM uses the seed list present in this database for synthetic data generation. The Fast Data Masker component can also use this seed list.
- Scramble database
  The Fast Data Masker component uses the seed list present in this database.
- File system

The table schema of the seed list tables in the GTREP and Scramble databases is different. Also, the schema of the seed list table in the Scramble database is not the same across different databases (Oracle, MS SQL, and Teradata).

For any record in the seed list table, the following fields are relevant to the user:

- Category Name
- Value
  The value can include a single value or multiple values. In the GTREP database table, each value is further associated with a name.

The other fields in the seed list table are calculated values that Datamaker or Fast Data Masker uses internally.

Considerations

Review the following considerations:

- Ensure that the seed data is present in a CSV (comma-separated value) file. Each row in the CSV file represents a record in the database. The first value in the row represents the category name and the remaining items in the row represent the name-value pairs. For example, consider a row in the CSV file:

  ```
  Address-US,City,Plano,State,Texas
  ```

  In this example, `Address-US` represents the category name and `City,Plano` and `State,Texas` represent the name-value pairs.
- Remove the column headers from the CSV file.
- If the CSV file contains non-ASCII characters, use UTF-8 encoding.
- The minimum number of name-value pairs is one and the maximum is 30.
- The SQL scripts that are used in this article are generic and are written considering the maximum allowed values. If you have fewer values in a row of a CSV file, change the SQL scripts accordingly.
- If you need to insert data only in the Scramble database where names are not required for the values, you can simplify the CSV file by providing only the category names and values. Also, you must modify the SQL query that is used for loading the CSV file into the table as appropriate.
- Some databases (for example, MS SQL and Teradata) expect that the number of values in each row of a CSV file must match the number of columns present in the database table. Therefore, if you have a CSV file where the number

of values is not the same for each row or the number of values is fewer than the number of table columns, you can use the provided PowerShell script. This script helps you transform the CSV file into the format that the SQL scripts included in this article require. For more information about how to run this script see, the Run the Transform Script section.

Repository Database

The repository database supports the following flavors of databases:

- Oracle
- MS SQL

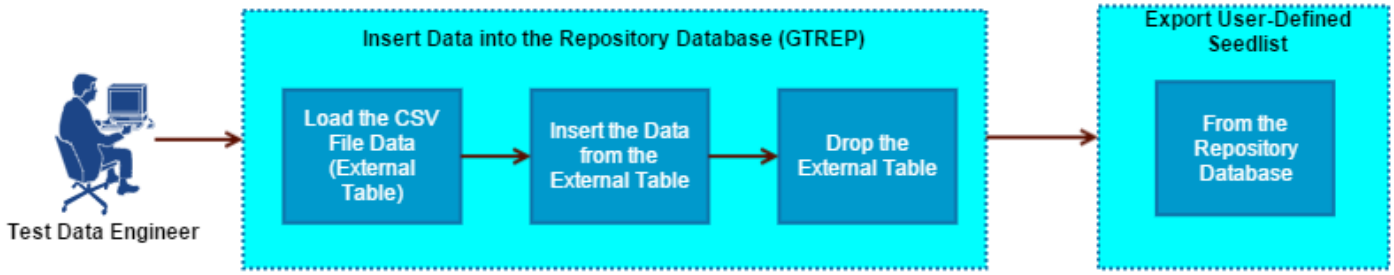
You can use the appropriate SQL scripts provided in each section to import/export the seed list data:

Seed List Propagation in the Repository Database for Oracle

This section includes information about how you can propagate seed list data when Oracle is used as a database.

The following illustration outlines the high-level process:

Figure 31: Seedlist_Oracle



Insert Data into the Repository Database (GTREP)

The seed list in the repository database is stored in the `gtrep_reference_data` table. CA TDM uses this data for synthetic data generation.

To insert the seed data into this table, perform the following steps:

1. Load the CSV file data as an external table in the Oracle database as follows:
   a. Create a directory for the Oracle user and map it to the hard disk location on the Oracle server host system by running the following SQL commands:

```
create or replace directory CSVDIR as 'C:/mycsv';
grant read, write on directory CSVDIR to GTREP;
```

   b. Copy the CSV file to the created directory (in this case, `C:/mycsv`).
   c. Run the following SQL command to create the external table:

```
ALTER SESSION SET CURRENT_SCHEMA = GTREP;
CREATE TABLE csv_ext_table
 (
 GROUPNAME VARCHAR2(254),
 NAME1 VARCHAR2(254),
 VALUE1 VARCHAR2(254),
 NAME2 VARCHAR2(254),
```

```
VALUE2 VARCHAR2(254),
NAME3 VARCHAR2(254),
VALUE3 VARCHAR2(254),
NAME4 VARCHAR2(254),
VALUE4 VARCHAR2(254),
NAME5 VARCHAR2(254),
VALUE5 VARCHAR2(254),
NAME6 VARCHAR2(254),
VALUE6 VARCHAR2(254),
NAME7 VARCHAR2(254),
VALUE7 VARCHAR2(254),
NAME8 VARCHAR2(254),
VALUE8 VARCHAR2(254),
NAME9 VARCHAR2(254),
VALUE9 VARCHAR2(254),
NAME10 VARCHAR2(254),
VALUE10 VARCHAR2(254),
NAME11 VARCHAR2(254),
VALUE11 VARCHAR2(254),
NAME12 VARCHAR2(254),
VALUE12 VARCHAR2(254),
NAME13 VARCHAR2(254),
VALUE13 VARCHAR2(254),
NAME14 VARCHAR2(254),
VALUE14 VARCHAR2(254),
NAME15 VARCHAR2(254),
VALUE15 VARCHAR2(254),
NAME16 VARCHAR2(254),
VALUE16 VARCHAR2(254),
NAME17 VARCHAR2(254),
VALUE17 VARCHAR2(254),
NAME18 VARCHAR2(254),
VALUE18 VARCHAR2(254),
NAME19 VARCHAR2(254),
VALUE19 VARCHAR2(254),
NAME20 VARCHAR2(254),
VALUE20 VARCHAR2(254),
NAME21 VARCHAR2(254),
VALUE21 VARCHAR2(254),
NAME22 VARCHAR2(254),
VALUE22 VARCHAR2(254),
NAME23 VARCHAR2(254),
VALUE23 VARCHAR2(254),
NAME24 VARCHAR2(254),
VALUE24 VARCHAR2(254),
NAME25 VARCHAR2(254),
VALUE25 VARCHAR2(254),
NAME26 VARCHAR2(254),
VALUE26 VARCHAR2(254),
NAME27 VARCHAR2(254),
VALUE27 VARCHAR2(254),
NAME28 VARCHAR2(254),
VALUE28 VARCHAR2(254),
```

```
NAME29 VARCHAR2(254),
VALUE29 VARCHAR2(254),
NAME30 VARCHAR2(254),
VALUE30 VARCHAR2(254)
)
organization external
(
type ORACLE_LOADER
default directory CSVDIR
access parameters
(
RECORDS DELIMITED by NEWLINE
FIELDS TERMINATED by ","
MISSING FIELD VALUES ARE NULL
REJECT ROWS WITH ALL NULL FIELDS
)
location ('SAMPLE_SEED.csv') -- provide your csv file name here
)
reject limit unlimited;
```

2. Insert the data from the external table `csv_external_table` into the `gtrep_reference_data` table by running the following SQL command:

```
ALTER SESSION SET CURRENT_SCHEMA = GTREP;
INSERT
INTO gtrep_reference_data(rd_ref_id, rd_ref_type, rd_col_cnt, rd_ref_name_1, rd_ref_value_1,
 rd_ref_name_2, rd_ref_value_2, rd_ref_name_3, rd_ref_value_3,
rd_ref_name_4, rd_ref_value_4, rd_ref_name_5, rd_ref_value_5, rd_ref_name_6, rd_ref_value_6,
 rd_ref_name_7, rd_ref_value_7, rd_ref_name_8, rd_ref_value_8,
rd_ref_name_9, rd_ref_value_9, rd_ref_name_10, rd_ref_value_10, rd_ref_name_11, rd_ref_value_11,
 rd_ref_name_12, rd_ref_value_12, rd_ref_name_13, rd_ref_value_13,
rd_ref_name_14, rd_ref_value_14, rd_ref_name_15, rd_ref_value_15, rd_ref_name_16, rd_ref_value_16,
 rd_ref_name_17, rd_ref_value_17, rd_ref_name_18, rd_ref_value_18,
rd_ref_name_19, rd_ref_value_19, rd_ref_name_20, rd_ref_value_20, rd_ref_name_21, rd_ref_value_21,
 rd_ref_name_22, rd_ref_value_22, rd_ref_name_23, rd_ref_value_23,
rd_ref_name_24, rd_ref_value_24, rd_ref_name_25, rd_ref_value_25, rd_ref_name_26, rd_ref_value_26,
 rd_ref_name_27, rd_ref_value_27, rd_ref_name_28, rd_ref_value_28,
rd_ref_name_29, rd_ref_value_29, rd_ref_name_30, rd_ref_value_30, rd_proj_id)

SELECT GROUPNAME, 'SAM' REF_TYPE,
 ((CASE WHEN NAME1 IS NULL THEN 0 ELSE 1 END)
 + (CASE WHEN NAME2 IS NULL THEN 0 ELSE 1 END)
 + (CASE WHEN NAME3 IS NULL THEN 0 ELSE 1 END)
 + (CASE WHEN NAME4 IS NULL THEN 0 ELSE 1 END)
 + (CASE WHEN NAME5 IS NULL THEN 0 ELSE 1 END)
 + (CASE WHEN NAME6 IS NULL THEN 0 ELSE 1 END)
 + (CASE WHEN NAME7 IS NULL THEN 0 ELSE 1 END)
 + (CASE WHEN NAME8 IS NULL THEN 0 ELSE 1 END)
 + (CASE WHEN NAME9 IS NULL THEN 0 ELSE 1 END)
 + (CASE WHEN NAME10 IS NULL THEN 0 ELSE 1 END)
 + (CASE WHEN NAME11 IS NULL THEN 0 ELSE 1 END)
 + (CASE WHEN NAME12 IS NULL THEN 0 ELSE 1 END)
 + (CASE WHEN NAME13 IS NULL THEN 0 ELSE 1 END)
```

```
    + (CASE WHEN NAME14 IS NULL THEN 0 ELSE 1 END)
    + (CASE WHEN NAME15 IS NULL THEN 0 ELSE 1 END)
    + (CASE WHEN NAME16 IS NULL THEN 0 ELSE 1 END)
    + (CASE WHEN NAME17 IS NULL THEN 0 ELSE 1 END)
    + (CASE WHEN NAME18 IS NULL THEN 0 ELSE 1 END)
    + (CASE WHEN NAME19 IS NULL THEN 0 ELSE 1 END)
    + (CASE WHEN NAME20 IS NULL THEN 0 ELSE 1 END)
    + (CASE WHEN NAME21 IS NULL THEN 0 ELSE 1 END)
    + (CASE WHEN NAME22 IS NULL THEN 0 ELSE 1 END)
    + (CASE WHEN NAME23 IS NULL THEN 0 ELSE 1 END)
    + (CASE WHEN NAME24 IS NULL THEN 0 ELSE 1 END)
    + (CASE WHEN NAME25 IS NULL THEN 0 ELSE 1 END)
    + (CASE WHEN NAME26 IS NULL THEN 0 ELSE 1 END)
    + (CASE WHEN NAME27 IS NULL THEN 0 ELSE 1 END)
    + (CASE WHEN NAME28 IS NULL THEN 0 ELSE 1 END)
    + (CASE WHEN NAME29 IS NULL THEN 0 ELSE 1 END)
    + (CASE WHEN NAME30 IS NULL THEN 0 ELSE 1 END)) COLM_COUNT,
    NAME1,VALUE1,NAME2,VALUE2,NAME3,VALUE3,NAME4,VALUE4,
    NAME5,VALUE5,NAME6,VALUE6,NAME7,VALUE7,NAME8,VALUE8,
    NAME9,VALUE9,NAME10,VALUE10,NAME11,VALUE11,NAME12,VALUE12,
    NAME13,VALUE13,NAME14,VALUE14,NAME15,VALUE15,NAME16,VALUE16,
    NAME17,VALUE17,NAME18,VALUE18,NAME19,VALUE19,NAME20,VALUE20,
    NAME21,VALUE21,NAME22,VALUE22,NAME23,VALUE23,NAME24,VALUE24,
    NAME25,VALUE25,NAME26,VALUE26,NAME27,VALUE27,NAME28,VALUE28,
    NAME29,VALUE29,NAME30,VALUE30,0

   FROM CSV_EXT_TABLE;
```

3. Drop the external table as follows:

```
   ALTER SESSION SET CURRENT_SCHEMA=GTREP;
   drop table CSV_EXT_TABLE;
```

Export User-Defined Seed List

You can export the user-defined seed list from the repository database.

If a user-defined seed list is present in the `gtrep_reference_data` table in the repository database (GTREP), you can export it to a CSV file. The user-defined data in seed list must have been added with the `rd_ref_type` column value as `SAM` to identify that this is a user-defined data.

Run the following SQL script to export user-defined seed data from the `gtrep_reference_data` table into the GTREP database:

```
ALTER SESSION SET CURRENT_SCHEMA=GTREP;
DECLARE file_handle utl_file.file_type;
delimiter CHAR := ',';
BEGIN
 --
 -- open file for writing
 file_handle := utl_file.fopen('CSVDIR', 'GTREP_SAMPLE_SEED.csv', 'w');
 --
 -- loop through the records
 FOR seed_rec IN
```

```
 (SELECT *
 FROM gtrep_reference_data
 WHERE rd_ref_type = 'SAM') -- add more clauses to filter the records
 LOOP
 utl_file.PUT_LINE(file_handle, seed_rec.rd_ref_id || delimiter || seed_rec.rd_ref_name_1 || delimiter ||
 seed_rec.rd_ref_value_1 || delimiter || seed_rec.rd_ref_name_2 || delimiter || seed_rec.rd_ref_value_2
 || delimiter || seed_rec.rd_ref_name_3 || delimiter || seed_rec.rd_ref_value_3 || delimiter ||
 seed_rec.rd_ref_name_4 || delimiter || seed_rec.rd_ref_value_4 || delimiter || seed_rec.rd_ref_name_5
 || delimiter || seed_rec.rd_ref_value_5 || delimiter || seed_rec.rd_ref_name_6 || delimiter ||
 seed_rec.rd_ref_value_6 || delimiter || seed_rec.rd_ref_name_7 || delimiter || seed_rec.rd_ref_value_7
 || delimiter || seed_rec.rd_ref_name_8 || delimiter || seed_rec.rd_ref_value_8 || delimiter ||
 seed_rec.rd_ref_name_9 || delimiter || seed_rec.rd_ref_value_9 || delimiter || seed_rec.rd_ref_name_10
 || delimiter || seed_rec.rd_ref_value_10 || delimiter || seed_rec.rd_ref_name_11 || delimiter ||
 seed_rec.rd_ref_value_11 || delimiter || seed_rec.rd_ref_name_12 || delimiter || seed_rec.rd_ref_value_12
 || delimiter || seed_rec.rd_ref_name_13 || delimiter || seed_rec.rd_ref_value_13 || delimiter ||
 seed_rec.rd_ref_name_14 || delimiter || seed_rec.rd_ref_value_14 || delimiter || seed_rec.rd_ref_name_15
 || delimiter || seed_rec.rd_ref_value_15 || delimiter || seed_rec.rd_ref_name_16 || delimiter ||
 seed_rec.rd_ref_value_16 || delimiter || seed_rec.rd_ref_name_17 || delimiter || seed_rec.rd_ref_value_17
 || delimiter || seed_rec.rd_ref_name_18 || delimiter || seed_rec.rd_ref_value_18 || delimiter ||
 seed_rec.rd_ref_name_19 || delimiter || seed_rec.rd_ref_value_19 || delimiter || seed_rec.rd_ref_name_20
 || delimiter || seed_rec.rd_ref_value_20 || delimiter || seed_rec.rd_ref_name_21 || delimiter ||
 seed_rec.rd_ref_value_21 || delimiter || seed_rec.rd_ref_name_22 || delimiter || seed_rec.rd_ref_value_22
 || delimiter || seed_rec.rd_ref_name_23 || delimiter || seed_rec.rd_ref_value_23 || delimiter ||
 seed_rec.rd_ref_name_24 || delimiter || seed_rec.rd_ref_value_24 || delimiter || seed_rec.rd_ref_name_25
 || delimiter || seed_rec.rd_ref_value_25 || delimiter || seed_rec.rd_ref_name_26 || delimiter ||
 seed_rec.rd_ref_value_26 || delimiter || seed_rec.rd_ref_name_27 || delimiter || seed_rec.rd_ref_value_27
 || delimiter || seed_rec.rd_ref_name_28 || delimiter || seed_rec.rd_ref_value_28 || delimiter ||
 seed_rec.rd_ref_name_29 || delimiter || seed_rec.rd_ref_value_29 || delimiter || seed_rec.rd_ref_name_30
 || delimiter || seed_rec.rd_ref_value_30);
 END LOOP;
 --
 -- close the file
 utl_file.fclose(file_handle);
EXCEPTION
WHEN others THEN
 IF utl_file.is_open(file_handle) THEN
 utl_file.fclose(file_handle);
 END IF;
END;
```

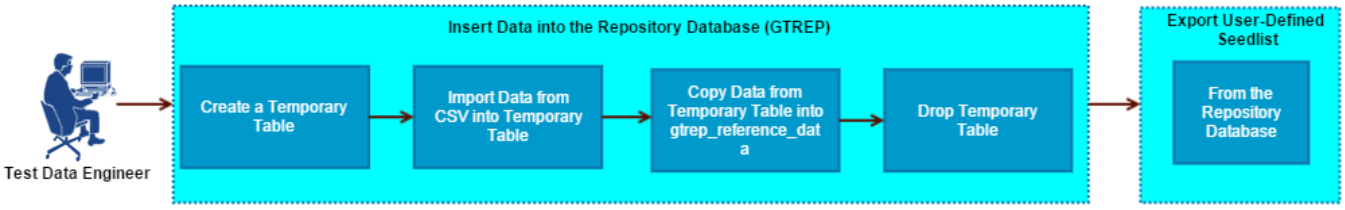
The file is exported to the directory mapped to the CSVDIR directory on the Oracle server.

Seed List Propagation in the Repository Database for MS SQL

This section includes information about how you can propagate seed list data when MS SQL is used as a database.

The following illustration outlines the high-level process:

Figure 32: MSSQL_Seedlist



Insert Data into the Repository Database (GTREP)

Note: The scripts in this section use the GTREP as the repository database.

1. Create a temporary table in the MS SQL database as follows:

```
USE [GTREP]
CREATE TABLE csv_ext_table
(
GROUPNAME VARCHAR(254),
NAME1 VARCHAR(254) ,
VALUE1 VARCHAR(254),
NAME2 VARCHAR(254) DEFAULT NULL,
VALUE2 VARCHAR(254) DEFAULT NULL,
NAME3 VARCHAR(254) DEFAULT NULL,
VALUE3 VARCHAR(254) DEFAULT NULL,
NAME4 VARCHAR(254) DEFAULT NULL,
VALUE4 VARCHAR(254) DEFAULT NULL,
NAME5 VARCHAR(254) DEFAULT NULL,
VALUE5 VARCHAR(254) DEFAULT NULL,
NAME6 VARCHAR(254) DEFAULT NULL,
VALUE6 VARCHAR(254) DEFAULT NULL,
NAME7 VARCHAR(254) DEFAULT NULL,
VALUE7 VARCHAR(254) DEFAULT NULL,
NAME8 VARCHAR(254) DEFAULT NULL,
VALUE8 VARCHAR(254) DEFAULT NULL,
NAME9 VARCHAR(254) DEFAULT NULL,
VALUE9 VARCHAR(254) DEFAULT NULL,
NAME10 VARCHAR(254) DEFAULT NULL,
VALUE10 VARCHAR(254) DEFAULT NULL,
NAME11 VARCHAR(254) DEFAULT NULL,
VALUE11 VARCHAR(254) DEFAULT NULL,
NAME12 VARCHAR(254) DEFAULT NULL,
VALUE12 VARCHAR(254) DEFAULT NULL,
NAME13 VARCHAR(254) DEFAULT NULL,
VALUE13 VARCHAR(254) DEFAULT NULL,
NAME14 VARCHAR(254) DEFAULT NULL,
VALUE14 VARCHAR(254) DEFAULT NULL,
NAME15 VARCHAR(254) DEFAULT NULL,
VALUE15 VARCHAR(254) DEFAULT NULL,
NAME16 VARCHAR(254) DEFAULT NULL,
VALUE16 VARCHAR(254) DEFAULT NULL,
NAME17 VARCHAR(254) DEFAULT NULL,
VALUE17 VARCHAR(254) DEFAULT NULL,
```

```
NAME18 VARCHAR(254) DEFAULT NULL,
VALUE18 VARCHAR(254) DEFAULT NULL,
NAME19 VARCHAR(254) DEFAULT NULL,
VALUE19 VARCHAR(254) DEFAULT NULL,
NAME20 VARCHAR(254) DEFAULT NULL,
VALUE20 VARCHAR(254) DEFAULT NULL,
NAME21 VARCHAR(254) DEFAULT NULL,
VALUE21 VARCHAR(254) DEFAULT NULL,
NAME22 VARCHAR(254) DEFAULT NULL,
VALUE22 VARCHAR(254) DEFAULT NULL,
NAME23 VARCHAR(254) DEFAULT NULL,
VALUE23 VARCHAR(254) DEFAULT NULL,
NAME24 VARCHAR(254) DEFAULT NULL,
VALUE24 VARCHAR(254) DEFAULT NULL,
NAME25 VARCHAR(254) DEFAULT NULL,
VALUE25 VARCHAR(254) DEFAULT NULL,
NAME26 VARCHAR(254) DEFAULT NULL,
VALUE26 VARCHAR(254) DEFAULT NULL,
NAME27 VARCHAR(254) DEFAULT NULL,
VALUE27 VARCHAR(254) DEFAULT NULL,
NAME28 VARCHAR(254) DEFAULT NULL,
VALUE28 VARCHAR(254) DEFAULT NULL,
NAME29 VARCHAR(254) DEFAULT NULL,
VALUE29 VARCHAR(254) DEFAULT NULL,
NAME30 VARCHAR(254) DEFAULT NULL,
VALUE30 VARCHAR(254) DEFAULT NULL
)
```

2. Import data from the CSV file into the temporary table as follows:

```
use [GTREP]
bulk insert [dbo].[csv_ext_table]
from 'C:\TEST_GROUP.csv' --provide your csv file path here
with (fieldterminator = ',', rowterminator = '\n', keepnulls)
go
```

3. Copy the data from the temporary table to the `gtrep_reference_data` table in the repository database as follows:

```
USE [GTREP]
INSERT
INTO [DBO].[GTREP_REFERENCE_DATA](rd_ref_id, rd_ref_type, rd_col_cnt, rd_ref_name_1, rd_ref_value_1,
 rd_ref_name_2, rd_ref_value_2, rd_ref_name_3, rd_ref_value_3,
rd_ref_name_4, rd_ref_value_4, rd_ref_name_5, rd_ref_value_5, rd_ref_name_6, rd_ref_value_6,
 rd_ref_name_7, rd_ref_value_7, rd_ref_name_8, rd_ref_value_8,
rd_ref_name_9, rd_ref_value_9, rd_ref_name_10, rd_ref_value_10, rd_ref_name_11, rd_ref_value_11,
 rd_ref_name_12, rd_ref_value_12, rd_ref_name_13, rd_ref_value_13,
rd_ref_name_14, rd_ref_value_14, rd_ref_name_15, rd_ref_value_15, rd_ref_name_16, rd_ref_value_16,
 rd_ref_name_17, rd_ref_value_17, rd_ref_name_18, rd_ref_value_18,
rd_ref_name_19, rd_ref_value_19, rd_ref_name_20, rd_ref_value_20, rd_ref_name_21, rd_ref_value_21,
 rd_ref_name_22, rd_ref_value_22, rd_ref_name_23, rd_ref_value_23,
rd_ref_name_24, rd_ref_value_24, rd_ref_name_25, rd_ref_value_25, rd_ref_name_26, rd_ref_value_26,
 rd_ref_name_27, rd_ref_value_27, rd_ref_name_28, rd_ref_value_28,
rd_ref_name_29, rd_ref_value_29, rd_ref_name_30, rd_ref_value_30, rd_proj_id)
```

```
SELECT GROUPNAME, 'SAM' REF_TYPE,
((CASE WHEN NAME1 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME2 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME3 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME4 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME5 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME6 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME7 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME8 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME9 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME10 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME11 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME12 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME13 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME14 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME15 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME16 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME17 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME18 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME19 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME20 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME21 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME22 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME23 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME24 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME25 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME26 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME27 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME28 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME29 IS NULL THEN 0 ELSE 1 END)
+ (CASE WHEN NAME30 IS NULL THEN 0 ELSE 1 END)) COLM_COUNT,
NAME1,VALUE1,NAME2,VALUE2,NAME3,VALUE3,NAME4,VALUE4,
NAME5,VALUE5,NAME6,VALUE6,NAME7,VALUE7,NAME8,VALUE8,
NAME9,VALUE9,NAME10,VALUE10,NAME11,VALUE11,NAME12,VALUE12,
NAME13,VALUE13,NAME14,VALUE14,NAME15,VALUE15,NAME16,VALUE16,
NAME17,VALUE17,NAME18,VALUE18,NAME19,VALUE19,NAME20,VALUE20,
NAME21,VALUE21,NAME22,VALUE22,NAME23,VALUE23,NAME24,VALUE24,
NAME25,VALUE25,NAME26,VALUE26,NAME27,VALUE27,NAME28,VALUE28,
NAME29,VALUE29,NAME30,VALUE30,0

FROM [DBO].[CSV_EXT_TABLE];
```

4. Drop the temporary table as follows:

```
USE [GTREP]
DROP TABLE [dbo].[csv_ext_table]
```

Export the User-Defined Seed List

You can export the user-defined seed list from the `gtrep_reference_data table` into the GTREP database.

Use the **Results to File** option in the SQL server query browser to export the data. Set the output format to comma delimited and exclude headers in **Query Options**.

Run the following script to export:

```
SELECT [rd_ref_id]
,[rd_ref_name_1]
,[rd_ref_value_1]
,[rd_ref_name_2]
,[rd_ref_value_2]
,[rd_ref_name_3]
,[rd_ref_value_3]
,[rd_ref_name_4]
,[rd_ref_value_4]
,[rd_ref_name_5]
,[rd_ref_value_5]
,[rd_ref_name_6]
,[rd_ref_value_6]
,[rd_ref_name_7]
,[rd_ref_value_7]
,[rd_ref_name_8]
,[rd_ref_value_8]
,[rd_ref_name_9]
,[rd_ref_value_9]
,[rd_ref_name_10]
,[rd_ref_value_10]
,[rd_ref_name_11]
,[rd_ref_value_11]
,[rd_ref_name_12]
,[rd_ref_value_12]
,[rd_ref_name_13]
,[rd_ref_value_13]
,[rd_ref_name_14]
,[rd_ref_value_14]
,[rd_ref_name_15]
,[rd_ref_value_15]
,[rd_ref_name_16]
,[rd_ref_value_16]
,[rd_ref_name_17]
,[rd_ref_value_17]
,[rd_ref_name_18]
,[rd_ref_value_18]
,[rd_ref_name_19]
,[rd_ref_value_19]
,[rd_ref_name_20]
,[rd_ref_value_20]
,[rd_ref_name_21]
,[rd_ref_value_21]
,[rd_ref_name_22]
,[rd_ref_value_22]
,[rd_ref_name_23]
,[rd_ref_value_23]
,[rd_ref_name_24]
,[rd_ref_value_24]
,[rd_ref_name_25]
,[rd_ref_value_25]
,[rd_ref_name_26]
```

```
,[rd_ref_value_26]
,[rd_ref_name_27]
,[rd_ref_value_27]
,[rd_ref_name_28]
,[rd_ref_value_28]
,[rd_ref_name_29]
,[rd_ref_value_29]
,[rd_ref_name_30]
,[rd_ref_value_30]
FROM [GTREP].[dbo].[gtrep_reference_data]
WHERE [rd_ref_type]='SAM' -- add more clauses to filter the records further
```

Scramble Database

The Scramble database supports the following flavors of databases:

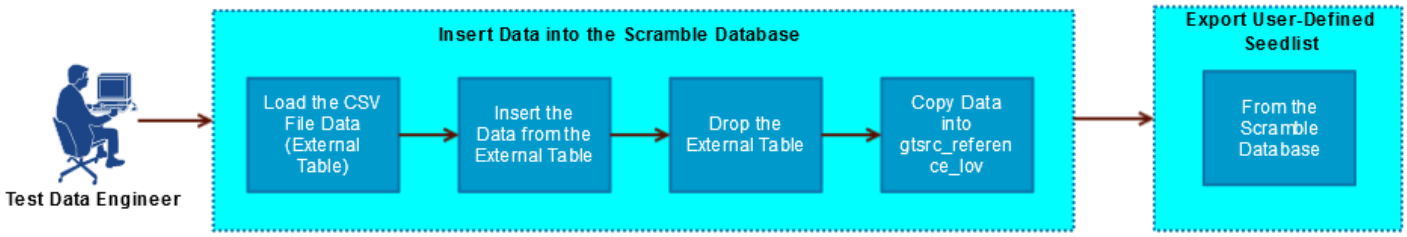
- Oracle
- MS SQL
- Teradata
- File System

Seed List Propagation in the Scramble Database for Oracle

This section includes information about how you can propagate seed list data when Oracle is used as a database.

The following illustration outlines the high-level process:

Figure 33: Oracle_Scramble



Insert Data into the Scramble Database

The Fast Data Masker component uses seed list from the Scramble database to mask data. The seed list is stored in the `gtsrc_reference_lov` table of the Scramble database.

To insert the new seed data, the data is first inserted into the `gtsrc_reference_data` table. The number of rows available for each seed category is calculated and the data is copied into the `gtsrc_reference_lov` table accordingly.

To insert the seed data into the `gtsrc_reference_lov` table, perform the following steps:

1. Load the CSV data as an external table in the Scramble database as mentioned in the Insert Data into the Repository Database (GTREP) section.
2. Import data from the external table into the `gtsrc_reference_data` table as follows:

```
ALTER SESSION SET CURRENT_SCHEMA=SCRAMBLE;
INSERT
INTO gtsrc_reference_data(rd_ref_id, rd_ref_value, rd_ref_value2, rd_ref_value3, rd_ref_value4,
 rd_ref_value5, rd_ref_value6, rd_ref_value7,
```

```
rd_ref_value8, rd_ref_value9, rd_ref_value10, rd_ref_value11, rd_ref_value12, rd_ref_value13,
 rd_ref_value14, rd_ref_value15, rd_ref_value16,
rd_ref_value17, rd_ref_value18, rd_ref_value19, rd_ref_value20, rd_ref_value21, rd_ref_value22,
 rd_ref_value23, rd_ref_value24, rd_ref_value25,
rd_ref_value26, rd_ref_value27, rd_ref_value28, rd_ref_value29, rd_ref_value30)

SELECT groupname, value1, value2, value3, value4, value5, value6, value7, value8, value9, value10,
 value11, value12, value13, value14, value15, value16, value17, value18, value19, value20, value21,
 value22, value23, value24, value25, value26, value27, value28, value29, value30
FROM csv_ext_table;
```

3. Drop the external table as follows:

```
ALTER SESSION SET CURRENT_SCHEMA=SCRAMBLE;
drop table csv_ext_table;
```

4. Copy the data from the `gtsrc_reference_data` table to the `gtsrc_reference_lov` table as follows:

```
ALTER SESSION SET CURRENT_SCHEMA=SCRAMBLE;
exec gtsrc_setcount.setcount();
```

Export User-Defined Seed List

You can export the user-defined seed list from the Scramble database.

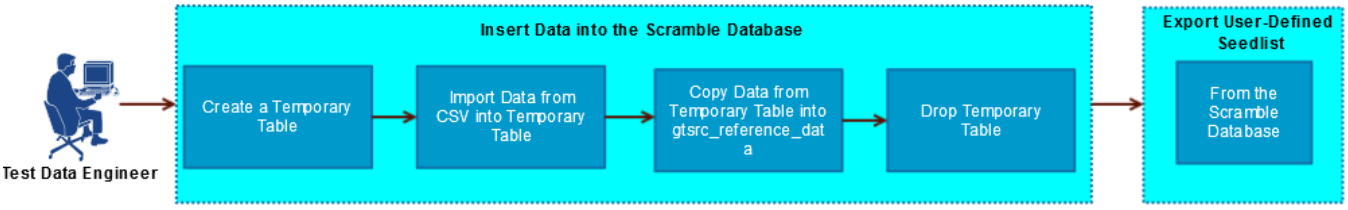
In the existing schema of the `gtsrc_reference_lov` table in the Scramble database, no information is available to identify the records added by the user. However, if users can use a WHERE clause to identify the records, they can then export the records using the similar export script as provided in the repository section.

Seed List Propagation in the Scramble Database for MS SQL

This section includes information about how you can propagate seed list data when MS SQL is used as a database.

The following illustration outlines the high-level process:

Figure 34: MSSQL_Scramble



Insert Data into the Scramble Database

Note: The scripts in this section use the Scramble database.

1. Create a temporary table in the MS SQL database as follows:

```
USE [SCRAMBLE]
CREATE TABLE csv_ext_table
 (
 GROUPNAME VARCHAR(254),
 NAME1 VARCHAR(254) ,
 VALUE1 VARCHAR(254),
```

```
NAME2 VARCHAR(254) DEFAULT NULL,
VALUE2 VARCHAR(254) DEFAULT NULL,
NAME3 VARCHAR(254) DEFAULT NULL,
VALUE3 VARCHAR(254) DEFAULT NULL,
NAME4 VARCHAR(254) DEFAULT NULL,
VALUE4 VARCHAR(254) DEFAULT NULL,
NAME5 VARCHAR(254) DEFAULT NULL,
VALUE5 VARCHAR(254) DEFAULT NULL,
NAME6 VARCHAR(254) DEFAULT NULL,
VALUE6 VARCHAR(254) DEFAULT NULL,
NAME7 VARCHAR(254) DEFAULT NULL,
VALUE7 VARCHAR(254) DEFAULT NULL,
NAME8 VARCHAR(254) DEFAULT NULL,
VALUE8 VARCHAR(254) DEFAULT NULL,
NAME9 VARCHAR(254) DEFAULT NULL,
VALUE9 VARCHAR(254) DEFAULT NULL,
NAME10 VARCHAR(254) DEFAULT NULL,
VALUE10 VARCHAR(254) DEFAULT NULL,
NAME11 VARCHAR(254) DEFAULT NULL,
VALUE11 VARCHAR(254) DEFAULT NULL,
NAME12 VARCHAR(254) DEFAULT NULL,
VALUE12 VARCHAR(254) DEFAULT NULL,
NAME13 VARCHAR(254) DEFAULT NULL,
VALUE13 VARCHAR(254) DEFAULT NULL,
NAME14 VARCHAR(254) DEFAULT NULL,
VALUE14 VARCHAR(254) DEFAULT NULL,
NAME15 VARCHAR(254) DEFAULT NULL,
VALUE15 VARCHAR(254) DEFAULT NULL,
NAME16 VARCHAR(254) DEFAULT NULL,
VALUE16 VARCHAR(254) DEFAULT NULL,
NAME17 VARCHAR(254) DEFAULT NULL,
VALUE17 VARCHAR(254) DEFAULT NULL,
NAME18 VARCHAR(254) DEFAULT NULL,
VALUE18 VARCHAR(254) DEFAULT NULL,
NAME19 VARCHAR(254) DEFAULT NULL,
VALUE19 VARCHAR(254) DEFAULT NULL,
NAME20 VARCHAR(254) DEFAULT NULL,
VALUE20 VARCHAR(254) DEFAULT NULL,
NAME21 VARCHAR(254) DEFAULT NULL,
VALUE21 VARCHAR(254) DEFAULT NULL,
NAME22 VARCHAR(254) DEFAULT NULL,
VALUE22 VARCHAR(254) DEFAULT NULL,
NAME23 VARCHAR(254) DEFAULT NULL,
VALUE23 VARCHAR(254) DEFAULT NULL,
NAME24 VARCHAR(254) DEFAULT NULL,
VALUE24 VARCHAR(254) DEFAULT NULL,
NAME25 VARCHAR(254) DEFAULT NULL,
VALUE25 VARCHAR(254) DEFAULT NULL,
NAME26 VARCHAR(254) DEFAULT NULL,
VALUE26 VARCHAR(254) DEFAULT NULL,
NAME27 VARCHAR(254) DEFAULT NULL,
VALUE27 VARCHAR(254) DEFAULT NULL,
NAME28 VARCHAR(254) DEFAULT NULL,
```

```
 VALUE28 VARCHAR(254) DEFAULT NULL,
 NAME29 VARCHAR(254) DEFAULT NULL,
 VALUE29 VARCHAR(254) DEFAULT NULL,
 NAME30 VARCHAR(254) DEFAULT NULL,
 VALUE30 VARCHAR(254) DEFAULT NULL
 )
```

2. Import data from the CSV file into the temporary table as follows:

```
use [SCRAMBLE]
bulk insert [dbo].[csv_ext_table]
from 'C:\TEST_GROUP.csv' --provide your csv file path here
with (fieldterminator = ',', rowterminator = '\n', keepnulls)
go
```

3. Copy the data from the temporary table to the `gtsrc_reference_data` table in the Scramble database:

```
use [SCRAMBLE]
INSERT
INTO [dbo].[gtsrc_reference_data](rd_ref_id, rd_ref_value, rd_ref_value2, rd_ref_value3, rd_ref_value4,
 rd_ref_value5, rd_ref_value6, rd_ref_value7,
rd_ref_value8, rd_ref_value9)

SELECT groupname, value1, value2, value3, value4, value5, value6, value7, value8, value9
FROM [dbo].[csv_ext_table];
```

4. Drop the temporary table as follows:

```
USE [SCRAMBLE]
DROP TABLE [dbo].[csv_ext_table]
```

Export the User-Defined Seed List

You can export the user-defined seed list from the Scramble database.

In the `gtsrc_reference_data` table in the Scramble database, no information is available to identify the user-defined rows. If users can create an appropriate WHERE clause, they can then use the following query and can export the results to a file:

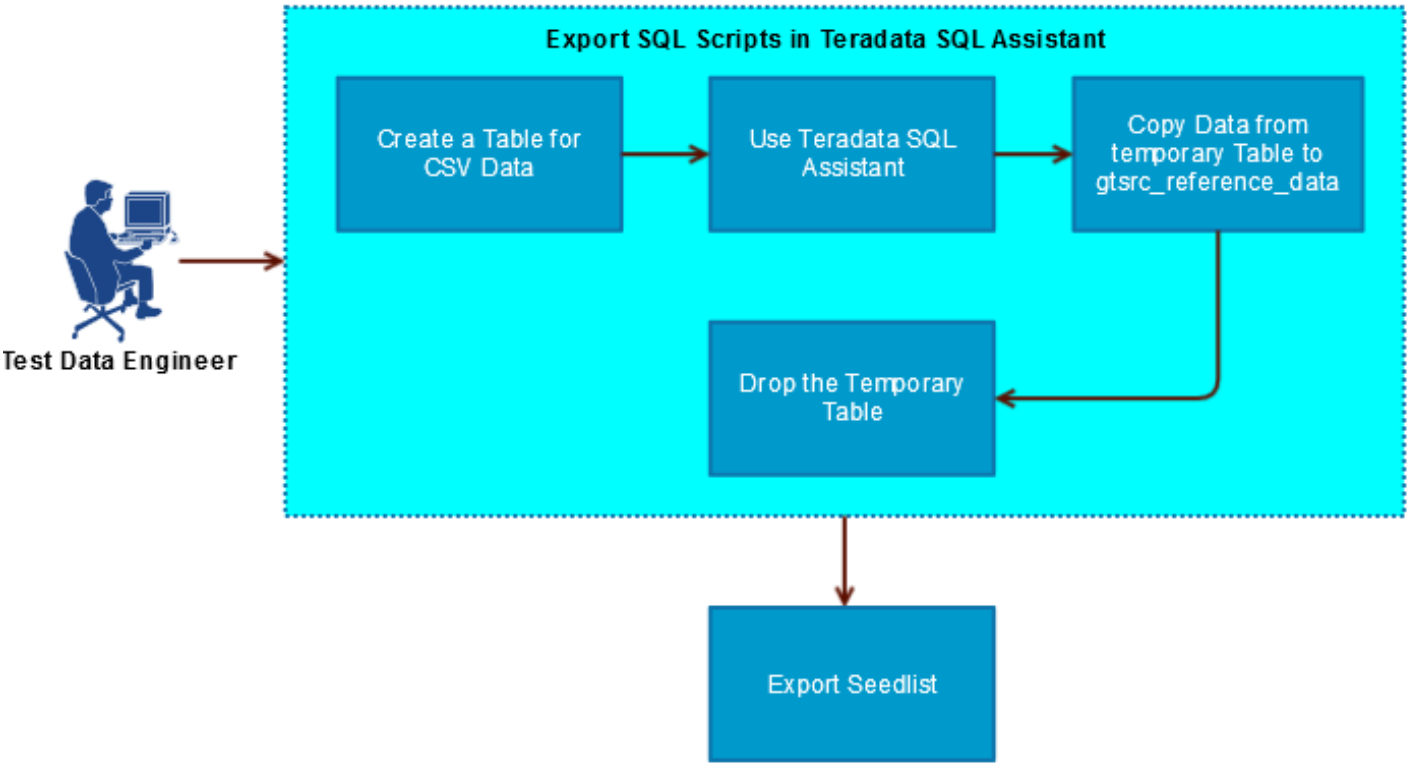
```
SELECT [rd_ref_id],
 [rd_ref_value]
 ,[rd_ref_value2]
 ,[rd_ref_value3]
 ,[rd_ref_value4]
 ,[rd_ref_value5]
 ,[rd_ref_value6]
 ,[rd_ref_value7]
 ,[rd_ref_value8]
 ,[rd_ref_value9]
 FROM [SCRAMBLE].[dbo].[gtsrc_reference_data] -- add your WHERE clause here
```

Seed List Propagation in the Scramble Database for Teradata

In Teradata, the seed list is present only in the `gtsrc_reference_data` table.

The following illustration outlines the high-level process:

Figure 35: Teradata_Seedlist



Consider the following points for Teradata:

- The database name that is used in the SQL scripts (used in this section) is *functions*.
- The scripts that are used in this section assume the CSV file to be in the same format as mentioned earlier in this article, with the exception that it contains only 19 values per row (one category name and nine name-value pairs) because the seed list for Teradata contains only nine values.
  Also, no names are required for the Teradata seed list. However, to keep the CSV format consistent, it is recommended to include them in the CSV file. They are ignored at the time of import. If you do not want to add names with values in the CSV file, you must modify the scripts accordingly.
- The CSV file must have exactly 19 values per row for the scripts that are used in this section to work. If fewer values are present, review the main Considerations section in this article.

Execute the SQL Scripts in Teradata SQL Assistant

1. Create a table to hold the values included in the CSV file:

```
CREATE TABLE csv_ext_table
 (
 groupname VARCHAR(254),
 name1 VARCHAR(254),
 value1 VARCHAR(254),
 name2 VARCHAR(254) DEFAULT NULL,
 value2 VARCHAR(254) DEFAULT NULL,
 name3 VARCHAR(254) DEFAULT NULL,
 value3 VARCHAR(254) DEFAULT NULL,
 name4 VARCHAR(254) DEFAULT NULL,
```

```
 value4 VARCHAR(254) DEFAULT NULL,
 name5 VARCHAR(254) DEFAULT NULL,
 value5 VARCHAR(254) DEFAULT NULL,
 name6 VARCHAR(254) DEFAULT NULL,
 value6 VARCHAR(254) DEFAULT NULL,
 name7 VARCHAR(254) DEFAULT NULL,
 value7 VARCHAR(254) DEFAULT NULL,
 name8 VARCHAR(254) DEFAULT NULL,
 value8 VARCHAR(254) DEFAULT NULL,
 name9 VARCHAR(254) DEFAULT NULL,
 value9 VARCHAR(254) DEFAULT NULL,
 row_index INTEGER
);
```

2. Open the Teradata SQL Assistant interface and perform the following steps:
   a. Verify that the import file accepts comma as a delimiter. If not, change it from the **Tools -> Options -> Import/ Export** menu.
   b. Enable the import operation by selecting **File -> Import Data** from the menu.
   c. Execute the following SQL statement:

```
INSERT INTO functions.csv_ext_table
 (groupname, name1, value1, name2, value2, name3, value3, name4,
 value4, name5, value5, name6, value6, name7, value7, name8, value8,
 name9, value9, row_index)
VALUES
(?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?,-1);
```

   A file browser dialog opens.
   d. Change the select file type in the file browser to All files(*.*), browse and select the CSV file, and click **OK**.
   e. After the file is imported, disable the import operation by selecting **File -> Import Data** from the menu.
3. Copy the data from temporary table to `gtsrc_reference_data` :

```
INSERT INTO functions.csv_ext_table
 (row_index, groupname, value1, value2, value3, value4, value5, value6, value7, value8, value9)
SELECT ROW_NUMBER() OVER (PARTITION BY groupname ORDER BY groupname)-1 AS RNUM, tab.groupname,tab.value1,
 tab.value2, tab.value3, tab.value4, tab.value5, tab.value6, tab.value7, tab.value8, tab.value9
FROM functions.csv_ext_table tab;
DELETE FROM functions.csv_ext_table WHERE row_index=-1;
UPDATE functions.csv_ext_table tempTab
SET row_index=COALESCE(row_index+(SELECT MAX(rd_index) FROM functions.gtsrc_reference_data refTab GROUP BY
 refTab.rd_ref_id where refTab.rd_ref_id=tempTab.groupname)+1,row_index);
INSERT INTO functions.gtsrc_reference_data
 (rd_ref_id, rd_ref_value,rd_ref_value2, rd_ref_value3,
 rd_ref_value4, rd_ref_value5, rd_ref_value6, rd_ref_value7, rd_ref_value8,
 rd_ref_value9, rd_index)
SELECT groupname,value1,value2,value3,
 value4,value5,value6,value7,
 value8,value9, row_index
FROM functions.csv_ext_table;
```

4. Drop the temporary table:

```
DROP TABLE csv_ext_table;
```

Export Seed List

You cannot identify user-defined rows in this table. If you can provide a WHERE clause to identify the data, use the following query to export the data. Set the Teradata SQL Assistant in export results to the file mode by selecting **File -> Export Results** before executing the query:

```
SELECT rd_ref_id, rd_ref_value, rd_ref_value2, rd_ref_value3,
 rd_ref_value4, rd_ref_value5, rd_ref_value6, rd_ref_value7, rd_ref_value8,
 rd_ref_value9
FROM functions.gtsrc_reference_data ---provide your where clause here
```

Note: On exporting from the Scramble database, the data is in the format CategoryName,Value1,Value2. That is, names are not associated with the values.

Seed List Propagation in the File System

Seed list present in the file system is shipped with the Fast Data Masker component. The seed list is present in the directory C:\Program Files\Grid-Tools\FastDataMasker\seedtables (if you installed the Fast Data Masker in the C:\ drive).

Each file or group of files present in the directory represents a category. If a category has single data in a row, it is represented by a single file and the file name represents the category name. If a category has multiple values in a row, the same number of files are present for that category. For example, category `australianpostalcodes` has two values, so two files `australianpostalcodes.1.txt` (contains first value for the row) and `australianpostalcodes.2.txt` (contains second value for the row) are present. The contents in a file are the values that are separated by a new line.

If multiple files for a category are present, each file must have an equal number of lines. In case of multiple files, it is possible for some rows to have only one value. In such cases, the second value in the second file must be blank.

To add more data to the existing seed category, add the values to the existing files. Each category file name is mapped with a logical name to be displayed in the Fast Data Masker UI.This mapping is done in the file BuildMap.xls located in the folder C:\Program Files\Grid-Tools\FastDataMasker. The mapping is present in the File Descriptions sheet of the BuildMap.xls file. If the file description is available for a file name, it is shown to the user in the UI. Otherwise, only the file name is displayed. Ideally, all files must have entries in this sheet.

To add a new category, create a new file (or files) based on the description that is mentioned in this section.

Limitations

The `gtsrc_reference_data` table in the Scramble database and the `gtrep_reference_data` table in GTREP do not have primary columns. So, you cannot identify a row uniquely. You must inspect the data to identify the appropriate record that you want to update or delete.

Run the Transform Script

The following scripts help you transform the CSV file into the required format that the scripts in this article can use: transformcsv.ps1 (For MS SQL) and transformcsv_teradata.ps1 (For Teradata).

1. Copy the files on your Windows computer where PowerShell is enabled.
2. Copy the CSV file on the same computer.
3. Open the appropriate transform file in a text editor and edit the first and the second lines to provide the path of your input CSV file and output CSV file.
4. Save your changes.
5. Right-click on the transform file and select **Run with PowerShell** from the context menu (or run it using the PowerShell command shell).

Script-Based Masking

Script-based masking uses the list of functions provided in the GTREP_FUNCTIONS table in the repository database (GTREP). If you add any new categories to the seed list table, ensure that you make them available for the HASHLOV, RANDLOV, and SEQLOV functions in the GTREP_FUNCTIONS table in the repository database (GTREP).

The scripts used in this section show how you can insert entries into the Oracle repository database (GTREP) when new categories are added to the scramble seed list (for different flavors of the databases).

Note: You must explicitly run the SQL statement for each new category; the SQL statement does not insert all the entries at the same time.

Update Oracle Repository Database (GTREP) Based on Oracle Scramble Seed List

If you add a new category to the Oracle seed list (scramble), add a corresponding category for the HASHLOV function in the format 'HASHLOV','CUSTOM_USER_GROUP' to the Oracle repository database table (GTREP). The function name and the category name are enclosed within quotes and are separated by a comma. Similarly, provide entries for the RANDLOV and SEQLOV functions.

Use the following SQL statement to insert the new category into the Oracle repository database (GTREP):

Note: In this case, the seed list that already contains the new category belongs to the Oracle scramble database.

```
INSERT INTO gtrep_functions(
                        fun_rdbms, --rdbms type ORACLE
                        fun_datatype,--the datatype of the category values.Options are CHAR,DATE & NUMERIC
                        fun_function, --function name
                        date_created,
                        date_updated,
                        who_created,
                        fun_description --description of what the function does
                        )
values
 ('ORACLE','CHAR','''HASHLOV'',''CUSTOM_USER_GROUP'''',to_char(sysdate),to_char(sysdate),'USER_NAME','custom_description
```

Apart from the mentioned columns, the table also has two more columns named `program_created` and `program_updated` , which you can leave blank.

Update Oracle Repository Database (GTREP) Based on MS SQL Scramble Seed List

If you add a new category to the MSSQL seed list (scramble), add a corresponding entry for RANDLOV in the format RANDLOV,CUSTOM_USER_GROUP to the Oracle repository database table (GTREP). The function name and the category name are not enclosed within quotes and are separated by a comma.

Use the following SQL statement to insert the new category into the Oracle repository database (GTREP).

Note: In this case, the seed list that already contains the new category belongs to the MS SQL scramble database.

```
INSERT INTO gtrep_functions(
                        fun_rdbms, --rdbms type SQLSERVER
                        fun_datatype,--the datatype of the category values.Options are CHAR,DATE & NUMERIC
                        fun_function, --function name
                        date_created,
                        date_updated,
                        who_created,
                        fun_description --description of what the function does
                        )
```

```
values
 ('SQLSERVER','CHAR','RANDLOV,CUSTOM_USER_GROUP',to_char(sysdate),to_char(sysdate),'USER_NAME','custom_description')
```

Update Oracle Repository Database (GTREP) Based on Teradata Scramble Seed List

For the Teradata seed list, the function name syntax is the same as for MS SQL. Provide the value for the `fun_rdbms` column as 'TERADATA'. The function name and the category name are NOT enclosed within quotes and are separated by a comma.

Update Oracle Repository Database (GTREP) Based on File System Scramble Seed List

For the file system seed list, the function name syntax is RANDLOV,FileName.txt. Provide the value for the `fun_rdbms` column as 'SDM'. The function name and the category file name are NOT enclosed within quotes and are separated by a comma.

Cloning in Datamaker

You as test engineer need adhoc or regular copies of customer data. You need only the most pertinent subset of the data, because testing the full, redundant data set wastes time. When copying data, you encounter typical data integrity issues: The additional environment planning to accommodate for data copying interferes with development and production work schedules. Also, all personally identifiable data must be masked in copied data, and it must be masked consistently.

For example: if you replace country names in one table with random letters, and you replace phone prefixes in another table with random numbers, then your phone number validation tests fail unnecessarily, and the tests become meaningless. Therefore, you need to ensure that complex test data is copied and masked in a coordinated manner.

Cloning in Datamaker retains core characteristics of the data, and maintains cross-application integrity:

- Datamaker extracts data from multiple systems.
- Datamaker masks data in transit.
- Datamaker assigns new keys every time when data is loaded.
- Testers can request only the data subset that suits a specific problem (TDM Portal).
- Testers receive test data adhoc, after a short time (TDM Portal).
- Testers can receive consistent copies of the same data and can run tests in parallel. (TDM Portal)
- Testers can pass on test data among each other in a controlled way (TDM Portal)

Video: Data Cloning in Datamaker -- Concepts

Video: Data Cloning in Datamaker -- How To

Store a Custom Cross Reference ID List in the Repository

When cloning and subsetting in Datamaker, you have the option to store a custom generated cross reference list in the repository.

1. Click the **LoV Options** button in the toolbar.
2. Enable **Store Xref Values in Test Data Repository** to activate this functionality.

For example, you want to publish a table in a datapool to clone data between a source and target. You use an xref function in the table's data definition:

1. Read an identifying ID from a column in the source, and generate a new ID to be used in the target.
2. Choose a list name in which to store the mapped values.
   The xref maps source and target ID values, and stores them in the given cross reference list.

3. Publish the table.
4. CA Datamaker publishes and inserts the row that you defined in the xref list.
5. Click **Tools**, **View and Authorize Jobs**.
   The **View Publish Logs** window opens.
6. Click the **Stored Values** tab for the last job and verify that the xref mapping that you created contains the values from target and source table.
7. Use the xlookup function to retrieve the mapping of the two values in a later expression.

NOTE

Xrefpersist performs the same basic function as xref, but additionally stores the specified old and new values in the repository. CA TDM portal stores only the table-column combinations that use xrefpersist in the repository. In CA Datamaker, using @xrefpersist once causes all values used in both @xref and @xrefpersist to be stored to the repository.

More Information:

- Data Generation Functions and Parameters: XREF, XREFPERSIST, and XLOOKUP

Generate Data Using the CA TDM Portal

This section includes information about how you can perform synthetic data generation using the CA TDM Portal. You can use the portal to perform various operations; for example, create and edit data generators, register tables, create and register derived objects, and perform actions on objects.

Prerequisites:

1. Create and Edit Connection Profiles.
2. Create a project.
3. Register file objects.
   Note: For the file objects that you register, you must perform actions on the data.

Complete the following process to generate synthetic data in the CA TDM Portal:

1. Create a data generator.
2. Create data generation rules.
3. Publish data.

Create Data Generator

CA TDM Portal lets you create Data Generators for the projects created in CA TDM Portal (recommended) or Datamaker. The Data Generator includes the functionality to create data generation rules and publish synthetic data. You can copy generators, but only within the same project version.

If you want to create Data Generators for the projects created in Datamaker, ensure that the respective projects are configured with the default project levels. For example, Data Group, Data Set, Data Pool.

Generators require registered objects to generate data for in the project version that you associate with the generator. Before creating a generator, register file objects and create derived tables from them.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click the **Generators** options in the left pane.
   The **Data Generators** page opens.

4. Click **New Generator**.
   The **Create New Generator** page opens.
5. Specify the following and click **Save**.
   - **Name**
     Specifies the data generator name.
     Note: Ensure that you do not use spaces or slashes (/) in your generator name.
   - **Description**
     Specifies a brief description of the data generator.

   The data generator is created and added to the list.
   Use the Refresh button to reload the page with an updated list.
   Use the Search feature to find a specific Data Generator from the list. The text you enter to search, yields the full and partial matches with the data generator name and data generator description.
   You can see the Delete Generator icon against each Generator Name. Click the Delete Generator icon to delete the generator.

Note: For more information about creating variables at a generator level, see Create and Manage Variables.

Create Data Generation Rules

Use the Generation functionality in the CA TDM Portal to create data generation rules. These rules include data functions that help you generate synthetic data when you perform the data publish operation. At the time of data publishing, the CA TDM Portal evaluates each column and resolves the rule that is applied to each column. The application then populates the column with the value that is generated based on the defined rules.

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Generators** in the left pane.
   The **Generators** page opens.
   Note: If the left pane is not visible, click the icon (represented by three horizontal bars) in the top left corner.
4. Click the data generator that you want to use to publish data.
   The **<Data Generator Name>** page opens. This page includes information (for example, associated project and project version) about the data generator.
5. Click the **Select Tables** button to open the **Registered Tables** dialog.
   This dialog lists the registered tables (*used* and *unused*) based on the project that is associated to the selected generator. A *used* table is a table that includes data (or data generation rules). *Used* tables are represented by a Yes in the **Used** column.
   Note: You can also view the table relationships. For more information, see View Table Relationships.
6. (Optional) Click the **Relational Edit** button and select the following actions as required. These actions are applicable for the *used* tables in the selected data generator. These actions help you automate the data generation process to some extent:
   - **Make All Children References**
     Lets the foreign key column in the child table refer to the primary key column in the parent table using data generation functions. This option helps you automatically generate data generation rules for the foreign key column in the child table, which as a manual procedure is prone to error. This reference, therefore, enables you to establish a relationship between the parent and the corresponding child tables. This approach also maintains the referential integrity of the data at the time of publishing.
     For example, consider a scenario where the value in the foreign key column `purchaseOrder_tdm_root_SHRED_ID` of the child table `items` references the primary key column `SHRED_ID` in the parent table `purchaseOrder_tdm_root`. When you implement this option, the value in the foreign key column of the child table is replaced with an

expression `^purchaseOrder_tdm_root.SHRED_ID(1)^`. This expression is a data generation function that is generated automatically without any manual effort. And, it establishes a parent-child relationship.

– **Make All Parents default**
Replaces all the primary keys in the parent tables with the valid next sequence using data generation functions. For example, `SHRED_ID` is a primary key column that includes some value after the sample data is imported into the table `purchaseOrder_tdm_root`. Therefore, when you implement this option, the sample value in this column is automatically replaced with the expression `@nextval(SHRED_ID_SEQ)@`. This data generation function is automatically generated without any manual effort. This function establishes a sequence that generates unique, sequential values at the time of publishing.

Click **OK**, review the summary of the affected tables in the **Relational Editor Results** dialog, and click **OK**.
All *used* tables are updated based on the selected options.

7. Click the row corresponding to the table for which you want to create data generation rules.
All the table rows are added to the **<Data Generator Name>** page, including columns. You can perform the following actions as required:
   – To add another table to the **<Data Generator Name>** page, repeat this step.
   – Click the **+r** (Add Row) icon if no row is available in the table or you want to add another row to the table. To delete a row, click on the row number and then the **X** (Delete Row) icon.
   – To view the constraints applicable to a column belonging to a derived object, move the mouse pointer to the column name. A tool tip appears that shows the constraints. When you move out of a table cell, Portal validates the value based on the constraints applied to the column.

     NOTE

     If your Data Generation rule includes complex constraints that cause validation to take a long time, you can suppress Portal's automatic validation. To do so, add this line to your `application.properties` file:

     ```
     tdmweb.TDMGeneratorService.disableValidation=true
     ```

   – Click the First Row Display button to view the values of first row of a used table.
   Used table first row details help you understand the columns used in the respective table. You can use this information to select appropriate tables for which you can write necessary data generation rules. You can see the following details of the first row of each table:
     - Column Name
     - Definition
     - Default
     - Global Default
     - Data Type
8. Click the cells in each table row where you want to add the rule. You can also enter variables in the cell. You can type in the values in each cell, or use the pop-up menu (recommended) that appears when you click a cell. If you type in the values in the cells, the values entered in each row are saved only when all the mandatory cells in that row are filled in. If you choose to work with the pop-up menu, follow these steps:
   – Use the Cut and Paste icons to move the values from one cell to another.
   – Use Copy icon to copy the values from one cell to another cell/row/column. Following are the available copy options:
     - **Copy Row Down**
     All Rows - all cells
     Copies the values of the cells in the focused row to the corresponding cells in the rows below the current row of the same table.
     All Rows - empty cells
     Copies the values of the cells in the focused row to the corresponding empty cells in the rows below the current row of the same table.

New Rows - all cells
Copies the value of the cells in the focused row to the corresponding cells in a new row added below the current row in the same table.
New Rows - empty cells
Copies the value of the cells in the focused row to the corresponding empty cells in a new row added below the current row in the same table.

- **Copy Column Down**
  Copy Down All
  Copies the value of the focused cell to all the cells below in the same column in the same table.
  Copy Down Null (Empty)
  Copies the value of the focused cell to the empty cells below in the same column in the same table.
  Copy Down All with Increment
  Copies the value of the focused cell to all the cells below in the same column in the same table with the specified increment value.
  Copy Down Null with Increment
  Copies the value of the focused cell to the empty cells below in the same column in the same table with the specified increment value.
- **Copy Column Right**
  Copy Right All
  Copies the value of the focused cell to all the cells on the right side in the same row.
  Copy Right Null (Empty)
  Copies the value of the focused cell to the empty cells on the right side in the same row.
  Copy Right All with Increment
  Copies the value of the focused cell to all the cells on the right side in the same row with the specified increment value.
  Copy Right Null with Increment
  Copies the value of the focused cell to the empty cells on the right side in the same row with the specified increment value.
- **Copy Column Left**
  Copy Left All
  Copies the value of the focused cell to all the cells on the left side in the same row.
  Copy Left Null (Empty)
  Copies the value of the focused cell to the empty cells on the left side in the same row.
  Copy Left All with Increment
  Copies the value of the focused cell to all the cells on the left side with the specified increment value in the same row.
  Copy Left Null with Increment
  Copies the value of the focused cell to the empty cells on the left side with the specified increment value in the same row.

– Use the icons F, C and V to insert Functions, Columns and Variables respectively in the cells.
– Use the Data Painter icon (brush icon) to open the **Data Painter: <Table_Name>.<Column_Name>** dialog and create data generation rules.

9. Use the **Data Painter: <Table_Name>.<Column_Name>** dialog that opens when you click the Data Painter icon to edit and test the data functions. The dialog lets you click on objects in each of the three sections—Functions, Columns, and Variables. These objects transfer to the edit section (left side) where you manipulate them. The details of the three sections are as follows:
   Note: There is no character limit for data painter expressions in the CA TDM Portal. However, Datamaker has a limit of 16000 characters. If you expect parallel usage of this data generation rule in Datamaker, adhere to the Datamaker character limit.

- **Functions**
  Functions can use hard-coded values, columns, or variables as parameters. Functions can also use other functions as parameters. For example, you can use a function as a result from a Boolean operator in the IF function.
- **Columns**
  The Columns list contains any other columns in the table. This list also shows the columns in the other *used* tables.
- **Variables**
  The Variables section contains a combination of system operators and any substitution variables you have created. For more information about creating and managing variables, see Create and Manage Variables.

Note: The dialog also displays information about the constraints applicable to the columns belonging to derived objects. When you click the **Validate** button, the Portal validates the value based on the constraints applied to the column.

10. Expand the appropriate section (**Functions**, **Columns**, **Variables**) and click the required object.
    The object is added to the edit section.
    In case of any priorpublishkeylist function, you must replace the **"Id_id"** value with the respective generator ID. Select the applicable generator from the auto populated list when you add a priorpublishkeylist function to a column.
    In case of any sqllist function, you must replace the **"connection"** value with the respective connection profile. Select the applicable connection profile from the auto populated list when you add a sqllist function to a column. The parameter connection is replaced with the name of connection profile adding the character "P" as a prefix.In case of any seedlist function, you must replace the **"seedname"** value with the respective seed data type. Select the applicable seed data type from the auto populated list when you add a seedlist function to a column. For more information, see Data Generation Functions and Parameters.
    Note: You can also use the search field at the top of the **Functions** section to find an object.
11. Edit the rule and click **Validate** to verify that the data generation rule is working correctly.
    If the validation is unsuccessful, a proper error message is displayed. For more information about Functions, Parameters, and Return Values see Data Generation Functions and Parameters.
12. Click the **Insert** button to insert the validated rule into the cell.
13. Follow the same steps for other cells in the table.

You have successfully added data generation rules to the table columns. After you add data generation rules to a table, the table becomes a *used* table. You can verify this by reviewing the presence of a tick mark in the **Used** column. You can now publish the data into the target database schema.

Example: Create Data Generation Rules for the Employee Table

In this example, you create data generation rules for the Employee table so that you can generate and insert synthetic data into the table cells. This synthetic data does not include real data; it includes random data that is as close to the real data as possible. This random data is generated based on the data generation rules that you specify.

The Employee table includes the following columns:

- employee_id
- first_name
- last_name
- email
- phone
- birth_date
- salary

You add data generation rules to the cells in the Employee table as follows:

Note: For more information about specific data functions and their usage, see Data Generation Functions and Parameters.

employee_id

1. Select the **NEXT [Next value for column]** variable from the **Variables** list. This variable adds the next value in the sequence to this cell whenever you publish the table.
   The variable is added as follows to the edit section in the left: `~NEXT~`
2. Click **Validate**.
   A random number is generated and is displayed in the field next to the **Validate** button.
3. Click **Insert**.
   The variable is added to the **employee_id** cell in the table.

first_name

1. Select the **randlov(percnull,@seedlist(seedname)@)** function from the **Functions, List of Values** list. The data function is added as follows: `@randlov(percnull,@seedlist(seedname)@)@`
2. Search for the **seedname** value (**FirstName** in this case) in the pop-up menu and click the seedlist name to add it to the expression.
   The data function is updated as follows: `@randlov(percnull,@seedlist(FirstName)@)@`
3. Enter `0` as a value for percnull. This function allows you to identify the percentage of rows that are null. Selecting 20 means that 20 percent of the values are designated null.
   The data function is updated as follows: `@randlov(0,@seedlist(FirstName)@)@`
4. Click **Validate**.
   A random first name is generated based on the provided parameters and is displayed in the field next to the **Validate** button.
5. Click **Insert**.
   The data function is added to the **first_name** cell in the table.

last_name

- Follow the same steps as for the first_name column. The only difference is that you must select the seedlist name as **LastName**.

email

Note: The format of the email is `first_name.last_name@xyz.com`.

1. Select the **first_name** column from the **Column, employee** list.
   The data function is added as follows: `^first_name^`
2. Add a dot (.) after the column name and select the **last_name** column from the **Column, employee** list.
   The data function is updated as follows: `^first_name^.^last_name^`
3. Select the **atsign()** function from the **Functions, String** list.
   The data function is updated as follows: `^first_name^.^last_name^@atsign()@`
4. Add `xyz.com` after the last `@` symbol.
   The data function is updated as follows: `^first_name^.^last_name^@atsign()@xyz.com`
5. Click **Validate**.
   A random email ID is generated based on the provided parameters and is displayed in field next to the **Validate** button.
6. Click **Insert**.
   The data function is added to the **email** cell in the table.

phone

1. Select the **randlov(percnull,@seedlist(seedname)@)** function from the **Functions, List of Values** list.
   The data function is added as follows: `@randlov(percnull,@seedlist(seedname)@)@`
2. Search for the **seedname** value (**US Phone no** in this case) in the pop-up menu and click the seedlist name to add it to the expression.
   The data function is updated as follows: `@randlov(percnull,@seedlist(US Phone no)@)@`

3. Enter `0` as a value for percnull.
   The data function is updated as follows: `@randlov(0,@seedlist(US Phone no)@)@`
4. Click **Validate**.
   A random phone number is generated based on the provided parameters and is displayed in the field next to the **Validate** button.
5. Click **Insert**.
   The data function is added to the **phone** cell in the table.

birth_date

1. Select the **addranddays(date,min,max)** function from the **Functions, Date &/or Time** list.
   The data function is added as follows: `@addranddays(date,min,max)@`
2. For the **date** parameter, select **SDATE [System Date]** from the **Variables, System Variables** list.
   The data function is updated as follows: `@addranddays(~SDATE~,min,max)@`
3. For the **min** parameter, enter -20000 as the minimum value and for the **max** parameter, enter -8000 as the maximum value.
   The data function is updated as follows: `@addranddays(~SDATE~,-20000,-8000)@`
4. Click **Validate**.
   A random date is generated based on the specified parameters and is displayed in field next to the **Validate** button.
5. Click **Insert**.
   The data function is added to the **birth_date** cell in the table.

salary

1. Select the **addrand(number,min,max)** function from the **Functions, Numeric** list.
   The data function is added as follows: `@addrand(number,min,max)@`
2. Enter 100, 1000, and 100000 as values for the number, min, and max parameters respectively.
   The data function is updated as follows: `@addrand(100,1000,100000)@`
3. Click **Validate**.
   A random value is generated based on the specified parameters and is displayed in field next to the **Validate** button.
4. Click **Insert**.
   The data function is added to the **salary** cell in the table.

You have successfully generated data creation rules for the Employee table columns. Now, when you publish the data for this table, the data is generated based on the defined rules and then the generated synthetic data is inserted into the table.

Create and Manage Variables

The CA TDM Portal lets you use variables while editing the test data. These variables are resolved to appropriate values when you publish the test data. The two variable types are as follows:

- **Standard**
  These variables are standard functions that you use to manipulate data when you publish.
  For example, the standard variable `~CDATE~` resolves to the current date as defined in the project settings. You can use this value to identify the current data that is defined in the connection profile to which you are publishing.
- **User Defined**
  These variables are created by users.
  For example, create a variable named FIRSTNAME with a default value as `@randlov(0,@seedlist(FirstName)@)@` . Add this variable to the data in the form of ~FIRSTNAME~. When the data is published, the variable is resolved to an appropriate value. This variable is a user-defined variable that serves the appropriate business requirement. This approach of using a variable instead of a complete expression also improves the user experience, because you do not have to read the complex expression to understand what it implies. You can simply give a meaningful name to your defined variable to easily understand its purpose.

In more advanced use cases, user defined variables can be used like macros.

Variables follow the *~variablename~* format in the Portal.

This article covers the following procedures:

Understand the Variable Scope

You can define variables with the following scope:

- **Repository**
  A variable that is defined with the repository scope is accessible to all the projects, versions, and generators. This is a global variable.
- **Project**
  A variable that is defined with the project scope is accessible to that project, versions associated to the project, and generators associated to the project.
- **Version**
  A variable that is defined with the version scope is accessible to that version and generator associated to the version. This variable is not accessible to the repository and projects.
- **Generator**
  A variable that is defined with the generator scope is accessible only to that generator. This variable is not accessible to the associated projects and versions.
  When you view the list of variables for a generator, all variables that are defined with a higher scope (repository, associated project, and associated version) are also displayed. These variables are in addition to the ones that are explicitly defined for that generator. However, if a variable with the same name is defined for the repository (project, version, or generator), then only the variable for the generator is listed when you try to view the list of variables for a generator.
  For example, you create a variable `Name` with the project scope and set its value to `John` . You create another variable `Name` with the generator scope and set its value to `Mitchel` . Now, for the generator, the Portal displays the `Name` variable that is defined explicitly for that generator; it does not display the `Name` variable that is defined for the project. Therefore, in this case, the `Name` variable resolves to the value `Mitchel` , which is specified for the generator variable.

Considerations

Review the following considerations:

- Variables that follow a cyclic dependency (loop) are not supported.
  For example, consider a scenario where a variable `Var1` includes another variable `Var2` as its default value. The variable `Var2` in turn includes the variable `Var1` as its default value. In this case, the Portal displays an error when you try to add such variables to the table cell.
- Variables can resolve the hierarchy, if used.
  For example, a variable `Var1` includes `HR` as its value. Another variable `Var2` includes the variable `Var1` as its default value. So, the value of the variable `Var2` becomes `HR` . Now, you create another variable `Var3` , which includes the variable `Var2` as its default value. In this case, the value of the variable `Var3` also resolves to `HR` .
- For a generator, you cannot edit or delete a variable if it is an inherited variable.
  For example, you define a variable `Var1` with the project scope. This project variable (`Var1`) automatically becomes accessible to the generator (for example, `Gen 1`) that is associated to that project. Now, if you view the variables for `Gen 1` , you can find that the variable `Var1` is also listed as one of the variables in the list. If you try to edit or delete

the version `Var1` , you cannot do so, because it is an inherited variable. You can identify whether a variable is an inherited variable by reviewing the value in the **Scope** column.

- The display type GENERAL in Datamaker is represented as Text Box in the Portal.
- In Datamaker, if you have created variables at the Data Group, Data Set, or Data Pool level for a project, all these variables are listed under the associated generator in the Portal. Also, the **Scope** column value is blank for them in the Portal. That is, the actual inheritance is not shown in this case.
- In Datamaker, if you have created variables with the same name at the Data Group, Data Set, or Data Pool level, only the variable belonging to the lowest level is shown in the Portal. If you delete this variable from the Portal, the variable is deleted. But, the variables list still shows a variable with the same name to be present in the list, because another variable with the same name is already present at a higher level in the hierarchy. In this case, you delete the variable that is at the lowest level, and then the variable that is defined at the next level is shown in the list.
  For example, you create variables with the same name (`Var1`) in Datamaker at these levels—Data Group, Data Set, and Data Pool. Now, the variable `Var1` that is defined at the lowest level (Data Pool) is shown in the Portal, not the `Var1` variables defined at the other two levels. You now delete the displayed `Var1` variable from the Portal, the list still shows the variable `Var1` to be present. The `Var1` variable that the list now shows is the one that is defined at the next higher level in Datamaker, which is Data Set.
- The variable type GENERAL (which is present in Datamaker) is not available in the Portal. However, for the backward compatibility, the Portal supports the editing of the variables of type GENERAL that are coming from Datamaker.

Variable Access Permissions

To perform actions in the Portal, you must have access to specific privileges. These privileges come from the security functions that are associated to groups. And, users are part of these groups. Therefore, by being part of a group, users have access to the associated security functions, which allow them to perform appropriate tasks in the Portal.

In the context of variables, users who are part of a group that has access to the below-mentioned security functions can only create, update, delete, and view variables in the Portal. If users are not part of such user groups, they cannot view and use the variable-related functionality in the Portal. The following table shows the specific security functions that are required to work with variables:

Variable Scope	Tasks	Security Functions
Project, Version, and Generator	Create, Update, and Delete variables	Data Definition (for a specific project)
	View variables	Publish Data or Data Definition (for any project)
Repository	Create, Update, and Delete variables	Data Definition (for All Projects)
	View variables	Publish Data or Data Definition (for any project)

For example, if you want to create a variable with the project scope, you must be part of a user group that has access to the Data Definition security function for a project for which you are creating the variable. In this case, you can access the **Variables** option under **Modeling** and can proceed with the variable creation process.

> **NOTE**
> To create a data generator in the Portal, you must have access to the Edit Object security function. Similarly, to create projects and versions, you must have access to the Maintain Project security function.

Create a Variable

You can create variables with the repository, project, version, or generator scope.

Create a Variable with the Repository, Project, or Version Scope

You can create variables with the repository, project, or version scope based on your business requirements.

This procedure shows the most common way of creating a variable with the repository, project, or version scope. You can also create variable from other views. For more information, see Other Views from Which You Can Create Variables.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Modeling** in the left pane.
   The **Modeling** option expands and displays two options: **Objects** and **Variables**.
4. Click **Variables**.
   The **Variables** page opens.
5. Click the **New Variable** button.
   The **Create Variable** page opens.
6. Enter information in the following fields:
   - **Name**
     Specifies an appropriate name for the variable.
   - **Description**
     Specifies an appropriate description for the variable.
   - **Scope**
     Specifies the scope of the variable. Select the required scope from this drop-down list:
     - Repository
     - Project
     - Version

     NOTE
     To create a variable with the generator scope, see Create a Variable with the Generator Scope in this article.
   - **Type**
     Specifies the type of the variable that you are creating. You can select one of the following types:
     - String
     - Number
     - Date
     - Boolean
7. Select a value from the **Display Type** drop-down list to specify how you want to display the variable in the Portal. The type of display depends on the variable type that you select.

   NOTE
   The values that you provide for validation are case-sensitive.

   Variable Type: String
   The supported display types are Text Box, Drop Down List, Multi Select List, and Radio Button.
   - For Text Box, use the following field and then proceed with Step 8:
     - **Default Value**
       Specifies a default value for the variable. You can use one variable as a default value in another variable. You can also use a data generator expression as a default value. Use the **Data Painter** dialog to create and validate the data generator expression. To access the **Data Painter** dialog, click the data painter icon (next to the field). For more information about how to work with Data Painter, see Create Data Generation Rules.
   - For Drop Down List, Multi Select List, and Radio Button, use the following fields and then proceed with Step 8:
     - **List Definition**
       Provides a list of values that you want to use as options. Use the aslist() and getsql() functions in the **Data Painter** dialog. These functions are displayed in the **Data Painter** dialog only in the context of these display

types. Otherwise, they are not displayed. To access the **Data Painter** dialog, click the data painter icon (next to the field). For more information about how to work with Data Painter, see Create Data Generation Rules.

- **Default Value**
  Specifies the default value.
- **Validate**
  Lets you verify whether the default value is part of the options defined in the **List Definition** field. Click the **Validate** button for the verification.

Variable Type: Number

The supported display types are Text Box, Drop Down List, Multi Select List, and Radio Button.

– For Text Box, use the following fields and then proceed with Step 8:
  - **Include Validation**
    Enables the rule that you define in the **Rule Definition** field. The default value that you provide in the **Default Value** field is then validated against the defined rule. When you enable the **Include Validation** option, the **Rule Definition** field is also enabled.
  - **Rule Definition**
    Lets you define the rule that you want to use for the variable. You can select from the following options:
    - **In Values**
      Specifies that the only valid values are those in the comma-separated list of values you provide in the corresponding field.
      For example, if you provide the list of values as `1,2,3,4,5` , then the default value is validated against this list. If you specify the default value as `6` (which is not part of the list) and click the **Validate** button, an error message appears in the **Validate** field. This message states that the resolved default value is not present in the defined rule. However, if you specify the value as `3` (which is part of the defined list) and click **Validate**, the value `3` is validated and is displayed in the **Validate** field with a tick mark, indicating that the value complies with the defined rule.
    - **Greater Than**
      Specifies that the value must be larger than this value. Select Greater Than and provide the appropriate value in the field.
      For example, if you specify the value as `100` , then the default value must be greater than `100` .
    - **Less Than**
      Specifies that the value must be smaller than this value. Select Less Than and provide the appropriate value in the field.
      For example, if you specify the value as `100` , then the default value must be smaller than `100` .
    - **Between**
      Specifies that the value must be within this range. Select Between and provide the appropriate range in the fields.
      For example, if you specify the range values as `100` in the first field and `109` in the second field, then the default value must be within this range.
  - **Default Value**
    Specifies the default value.
  - **Validate**
    Lets you verify whether the default value is part of the defined rule. Click the **Validate** button for the verification.

– For Drop Down List, Multi Select List, and Radio Button, use the following fields and then proceed with Step 8:
  - **List Definition**
    Provides a list of values that you want to use as options.
  - **Default Value**
    Specifies the default value.
  - **Validate**
    Lets you verify whether the default value is part of the options defined in the **List Definition** field. Click the **Validate** button for the verification.

Variable Type: Date

The supported display types are Date Picker, Drop Down List, Multi Select List, and Radio Button. The supported date formats are `yyyy/MM/dd`, `yyyy.MM.dd`, `yyyy-MM-dd`, `MM/dd/yyyy`, `MM.dd.yyyy`, `MM-dd-yyyy`, `dd/MM/yyyy`, `dd.MM.yyyy`, `dd-MM-yyyy`, `yyyy-dd-MM`, `yyyy/dd/MM`, and `yyyy.dd.MM`.

- For Date Picker, use the following fields:
  - **Include Validation**
    Lets you enable the rule that you define in the **Rule Definition** field.
  - **Rule Definition**
    Lets you define the rule that you want to use for the variable. You can select from the following options:
    - **In Values**
    - **Greater Than**
    - **Less Than**
    - **Between**
  - **Default Value**
    Specifies the default value.
  - **Validate**
    Lets you verify whether the default value is part of the defined rule. Click the **Validate** button for the verification.
- For Drop Down List, Multi Select List, and Radio Button, use the following fields and then proceed with Step 8:
  - **List Definition**
    Provides a list of values that you want to use as options.
  - **Default Value**
    Specifies the default value.
  - **Validate**
    Lets you verify whether the default value is part of the options defined in the **List Definition** field. Click the **Validate** button for the verification.

Variable Type: Boolean

The supported display type is Check Box. Use the following fields for this display type:

- **Checked Value**
  Specifies the value to use when the variable is selected.
- **Unchecked Value**
  Specifies the value to use when the variable is not selected.
- **Default Value**
  Specifies the default value.
- **Validate**
  Lets you verify whether the default value is part of the options defined in the **Checked Value and Unchecked Value** fields. Click the **Validate** button for the verification.

8. Enter information in the following fields:

   NOTE
   These fields are applicable only for the Data Catalog form.

   - **Help Message**
     Specifies the appropriate text that is displayed as a tooltip for the variable in the Data Catalog form.
   - **Optional**
     Specifies whether the variable that you are creating is optional in the Data catalog form.
   - **Display Only**
     Specifies whether the variable is displayed in the read-only mode in the Data Catalog form. That is, only for the information purpose.
9. Select the **Resolve Prior to Publish** option if you want to resolve the variable before you start the publishing process. If you select this option, the variable value is resolved before the publishing and the same is used at all the places where the variable is referred. If you do not select this option, the variable is resolved during the publishing and the value is inserted at the time of publishing.

For example, consider a variable value that contains an expression (for example, `@addrand(100,1,10)@`). This expression adds a random number between `1` and `10` , starting with `100` as the base (for example, `101` , `102` , `103` ...`110`). So, when you select **Resolve Prior to Publish** and publish the data, the expression gets resolved to a random number; for example, `105` . Now, this value `105` is used at all the places where the variable is being referenced. However, if you do not select **Resolve Prior to Publish** and publish the data, all the variable references get different random values (for example, `103` , `104` , `105` , `110` , `107` , and so on) during the publishing.

10. Click **Save**.
    The variable is added to the **Variables** page.

Create a Variable with the Generator Scope

When you create a variable with the generator scope, that variable is accessible only to that generator.

This procedure shows the most common way of creating a variable with the generator scope. You can also create variable from other views. For more information, see Other Views from Which You Can Create Variables.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Generators** in the left pane.
   The **Data Generators** page opens. This page lists all the generators that are available for the project and version that you selected in Step 2.
4. Click the generator for which you want to create a variable.
   The **<Generator_Name>** page opens.
5. Click the **Variables** button.
   The **Variables** page opens.
6. Click the **New Variable** button.
   The **Create Variable** page opens.
7. Enter the required information as explained in Create a Variable with the Repository, Project, or Version Scope.
   The variable is added to the **Variables** page. This section also includes those variables that are defined for the associated repository, project, and version.

Create and Manage Macros

As a Test Data Engineer or Tester, you sometimes want to be able define macros, so that you can reuse complex queries in various locations. You can use macros like custom functions that use parameters as arguments, and return a value. The value associated with a macro is an expression that can contain functions, variables, column references, and constants. In expressions, you refer to parameters in the form `#param#` . If your expression contains literal hash characters, represent them using the `@hashsign()@` function, otherwise they will be misinterpreted as parameters. Create a macro using the following format:

- **Variable name:** `MyMacroName(param1, param2, param3, ...)`
- **Variable value:** `@function(#param1#, #param2#, #param3#, …)@`

Test Data Manager checks for existing variables with the same name. Duplicates are not possible. If you already have defined a macro that is named `MyMacroName(param1)` , you cannot create another macro or variable `MyMacroName(param1,param2)` nor `MyMacroName` .

1. Open the Portal and click **Modelling, Variables**.
   Note: You can create macros from all UIs where variables are created, at any level.
2. Define the variable name, for example:
   `MyAddFunc(num1, num2)`
3. Define the variable value, for example, this macro function adds two values:

```
@add(#num1#, #num2#)@
```

4. Use the macro in a Generator cell by plugging in specific values, for example:
   `~MyAddFunc(123, 456)~`
   This macro returns the value 579.

Arguments to a macro can be any expression, including other variables, functions, strings, numbers, column references, or macros. Variables etc. will be resolved, but note that Boolean expressions are read literally. If you really want a Boolean expression to be evaluated, use the `@and` function

Macro	Values	Argument interpretation
`~mymacro(~v1~=~v2~)~`	v1=1 and v2=2	1=2
`~mymacro(@and(~v1~=~v2~)@)~`	v1=1 and v2=2	false

Macro use case examples:

- If the argument is more than 5, then output the string "yes", otherwise out "no". For example, ifmore5(6) returns yes.
  ```
  ifmore5(a) = @if(#a# > 5, yes, no)@
  ```
- The Factorial function. For example, factorial(10) returns 3628800.
  ```
  Factorial(a) =
            @case(#a# < 0, error, #a# = 0, 1, #a#=1, 1, @multiply(#a#,~factorial(@subtract(#a#,1)@)~)@)@
  ```
- Add three numbers and the value of variable OFFSET. If the result is more than 100, then output the string "std", otherwise "non-std".
  ```
  Addcheck(a,b,c) = @if(@sum(#a#,#b#,#c#,~OFFSET~)@ > 1000, std, non-std)@
  ```

Other Views from Which You Can Create Variables

In addition to the main procedures that describe how to create variables, the Portal also allows you to create variables from other views. These views give you the flexibility of creating variables *inline*; that is, without loosing your current context.

Create Variables from the <Data_Generator_Name> Page

You can also create variables from the data generator view while defining data generation rules. This approach is helpful in situations where you are in the process of defining data generation rules. And, you want to add a variable to a table cell; however, the variable does not exist in the Portal. In this case, you want to create a variable, but do not want to move out of your current view. Your current view, the data generator page, provides an option that lets you access the **Create Variable** dialog. You can then define the variable and can proceed with your data generation rules.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Generators** in the left pane.
   The **Data Generators** page opens. This page lists all the generators that are available for the project and version that you selected in Step 2.
4. Click the generator for which you want to write data generation rules.
   The **<Generator_Name>** page opens
5. Click the **Select Tables** button.
6. Click the table where you want to add data generation rules.
   The table is added to the **<Generator_Name>** page.
7. Identify the table cell where you want to add the variable.

NOTE
The same steps are applicable to create a variable if you use the First Row Display view. For more information about the first row display, see Create Data Generation Rules.

8. Add the required variable (for example, `~MyVar~`) to the table cell.
   A red error icon with an error message is displayed next to the variable name. The red icon is displayed when you move the pointer out of that cell. The message states that the variable that you are trying to add to the table cell does not exist. The message also provides an option (**Create Variable**) to create the variable.
9. Click the **Create Variable** button in the error message.
   The **Create Variable** dialog opens, displaying the basic view. The basic view expects the least amount of information that is required to create a variable.
10. (Optional) Click the **Advanced View** button to access the advanced view, which includes more fields.
    The **Create Variable** page opens. This page includes all the fields. For more information, see Create a Variable with the Repository, Project, or Version Scope.
11. Enter information in the fields as required and save the information.
    The **<Generator_Name>** page opens. Review that the table cell now displays the variable with a green tick mark, which implies that the variable that you are trying to use is available in the Portal. You can now use this variable and proceed with your data generation rules.

Create Variables from the Data Painter Dialog

If you are in the **Data Painter** dialog and want to create a variable that does not exist, you can do so. This ability is helpful if you are creating an expression in the Data Painter dialog and you want to add a variable that does not exist. In this case, you do not need to move out of the current context. You can access the variable creation dialog from the same view.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Generators** in the left pane.
   The **Data Generators** page opens. This page lists all the generators that are available for the project and version that you selected in Step 2.
4. Click the generator for which you want to write data generation rules.
   The **<Generator_Name>** page opens
5. Click the **Select Tables** button.
6. Click the table where you want to add data generation rules.
   The table is added to the **<Generator_Name>** page.
7. Click in the table cell where you want to add the rule and select the Data Painter icon (paint brush icon) from the pop-up menu.
   The **Data Painter** dialog opens.
8. Perform the following tasks based on your requirements:
   Create a Variable by Converting the Complete Expression into a Variable
   a. Enter the expression in the text field; for example, `@randlov(0,@seedlist(UK Address)@)@` .
   b. Select the complete expression, because you want to convert the complete expression into a variable. Whatever you select in the text field becomes the default value of the variable.
   c. Click the **Create Variable** button.
      The **Create Variable** dialog opens, displaying the basic view. The basic view expects the least amount of information that is required to create a variable. Click the **Advanced View** button to access the advanced view, which includes more fields. The **Create Variable** page opens. This page includes all the fields. For more information, see Create a Variable with the Repository, Project, or Version Scope.
   d. Review that the default value is displayed as `@randlov(0,@seedlist(UK Address)@)@` in the dialog, which is the same expression that you selected in the text field.

e. Enter information in the fields as required and click **Save**.
The **Data Painter** dialog appears. Review that the text field includes the variable name.
f. Click **Validate** to validate the variable.
An appropriate value (for example, `Orchard Road`) is displayed after validating the variable expression.
g. Click the **Insert** button.
The **<Generator_Name>** page opens. Review that the table cell now includes the variable name; for example, `~Address~` . The default value of this variable corresponds to the complete expression.
h. Move the pointer out of the table cell.
A tick mark icon appears after the variable name, which suggests that the variable is successfully added to the table cell. You have successfully converted an expression into a variable.

Create a Variable by Converting a Part of the Expression into a Variable

a. Enter the expression in the text field; for example, `@randlov(0,@seedlist(UK Address)@)@` .
b. Select a part of the expression; for example, `@seedlist(UK Address)@` , because you want to convert only a part of the expression into a variable.
c. Click the **Create Variable** button.
The **Create Variable** dialog opens. Review that the default value is displayed as `@seedlist(UK Address)@` , which is the same expression segment that you selected in the text field.
d. Enter information in the fields as required and click **Save**.
The **Data Painter** dialog appears. Review that the text field includes the variable name in the expression; for example, `@randlov(0,~Address~)@` .
e. Click **Validate** to validate the variable.
An appropriate value (for example, `Hayward Road`) is displayed after validating the variable expression.
f. Click the **Insert** button.
The **<Generator_Name>** page opens. Review that the table cell now includes the variable name (`~Address~`) as part of the expression; for example, `@randlov(0,~Address~)@` .
g. Move the pointer out of the table cell.
A tick mark icon appears after the variable name, which suggests that the variable is successfully added to the table cell. You have successfully converted a part of the expression into a variable.

Create a Variable Without Converting Any Expression

a. Enter the variable name (for example, `~FirstName~`) in the text field and click **Validate**.
An error message appears. The message states that the variable does not exist and gives you the option to create it.
b. Click the **Create Variable** button in the message.
The **Create Variable** dialog opens. Review that the dialog does not include any default value, because you did not select anything in the text field.
c. Enter information in the fields as required and click **Save**.
The **Data Painter** dialog appears. Review that the text field displays the variable name, which is `~FirstName~` .
d. Click **Validate**.
The Portal validates and resolves the variable to an appropriate value based on what you specify in the default value.
e. Click **Insert**.
The **<Generator_Name>** page opens. Review that the table cell now includes the variable name.
f. Move the pointer out of the table cell.
A tick mark icon appears after the variable name, which suggests that the variable is successfully added to the table cell.

You have successfully created variables from other views.

Edit a Variable

If you want to update the variable information to reflect changed requirements or use cases, you can edit the defined variable.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. For the repository, project, and version variables, follow these steps:
   a. Click **Modeling** in the left pane.
      The **Modeling** option expands and displays two options: **Objects** and **Variables**.
   b. Click **Variables**.
      The **Variables** page opens.
4. For the generator variables, follow these steps:
   a. Click **Generators** in the left pane.
      The **Data Generators** page opens.
   b. Click the generator that includes the variable that you want to update.
      The **<Generator_Name>** page opens.
   c. Click the **Variables** button.
      The **Variables** page opens.

      NOTE
      For generators, you can edit only those variables that are created with the generator scope.
5. Locate the variable in the table.
6. Click the row corresponding to the identified variable.
   The **Edit Variable** page opens.
7. Update the information in the following fields:
   – **Description**
   – **Type**
   – **Display Type**

     NOTE
     Based on the display type and the corresponding variable type, the **List Definition**, **Include Validation, Rule Definition, Checked Value**, and **Unchecked Value** fields are displayed. For example, **List Definition** is displayed if you select the String variable type and Drop Down List display type.
   – **Default Value**
   – **Help Message**
   – **Optional**
   – **Display Only**
   – **Resolve Prior To Publish**
8. Click **Save**.
   The changes are saved and the updated variable information becomes available.

Delete a Variable

If you do not need a variable in your environment, you can delete it from the Portal. You must have appropriate privileges to delete a variable.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. For the repository, project, and version variables, follow these steps:
   a. Click **Modeling** in the left pane.
      The **Modeling** option expands and displays two options: **Objects** and **Variables**.

b. Click **Variables**.
The **Variables** page opens.

4. For the generator variables, follow these steps:
   a. Click **Generators** in the left pane.
   The **Data Generators** page opens.
   b. Click the generator that includes the variable that you want to delete.
   The **<Generator_Name>** page opens.
   c. Click the **Variables** button.
   The **Variables** page opens.

   NOTE
   For generators, you can delete only those variables that are created with the generator scope.
5. Locate the variable in the table.
6. Click the Delete Variable icon (X icon) in the row corresponding to the identified variable.
   A confirmation dialog opens.
7. Confirm the delete action.
   The variable is deleted from the list.

Example: Create and Use the `FIRSTNAME` Variable in the `First_Name` Column of the `Employee` Table

In this example, you create a variable `FIRSTNAME` that you want to use in the `First_Name` column of the `Employee` table. The variable is created with the project scope. The default value of the variable is set as `@randlov(0,@seedlist(FirstName)@)@` . After you create this variable, you use it in the `First_Name` column as `~FIRSTNAME~` . You decide to resolve this variable at the time of publishing so that the random first name values are generated during publishing.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project (for this example, `Employee)` and its corresponding version (for this example, `1.0`) from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Modeling** in the left pane.
   The **Modeling** option expands and displays two options: **Objects** and **Variables**.
4. Click **Variables**.
   The **Variables** page opens.
5. Click the **New Variable** button.
   The **Create Variable** page opens.
6. Enter information in the following fields:
   – **Name**
   Enter the name of the variable as `FIRSTNAME` .
   – **Description**
   Enter the description as `This variable adds first names` .
   – **Scope**
   Specify the scope of the variable as `Project` .
   – **Type**
   Specify the type of the variable as `String` .
   – **Display Type**
   Specify the display type as `Text Box` .
   – **Default Value**

Enter the default value as `@randlov(0,@seedlist(FirstName)@)@` and follow these steps to validate the expression:

a. Click the data painter icon (next to the field).
   The **Data Painter** dialog opens. The expression is automatically populated in the text field above the **Validate** button.
b. Click the **Validate** button.
   The expression is validated and a random value is displayed in the text field.
c. Verify the value.
   If the value is not correct, review your expression and fix the issue.
d. Close the **Data Painter** dialog.
   The entered expression is validated.

NOTE
For more information about how to use Data Painter, see Create Data Generation Rules.

- **Help Message**
  Enter the tooltip that you want to display for this variable as `This variable lets you add employee first names .`
- **Optional**
  Do not select this option for this example, because this variable is not an optional variable.
- **Display Only**
  Do not select this option for this example, because this variable is not a read-only variable and is not for the information purpose.
- **Resolve Prior to Publish**
  Do not select this option for this example, because you want to resolve this variable at the time of publishing to generate random data (first names).

7. Click **Save**.
   The variable is added to the **Variables** section.
8. Navigate to the page that lists all data generators.
9. Click the data generator applicable (for example, `Employee_Generator`) for this project.
10. Click the **Select Tables** button and then select the `Employee` table.
    The `Employee` table is added to the data generator page.
11. Click in the **First_Name** cell.
    A menu with different icons appears.
12. Click the Variables icon (icon V) in the icons menu and select the `FIRSTNAME` variable from the list of variables that is displayed under **Project**.
    The variable is added as `~FIRSTNAME~` in the **First_Name** table cell. You have successfully added a variable to a table.
13. Click the **Publish** button to display the publishing dialog.
14. Enter information that is required for this example as follows:
    - Enter `1` as a table count value.
    - Ensure that the **Include Page** option is selected.
    - Select the connection profile as `Employee_Connection` .
    - Select the schema as `dbo` .
    - Enter the repeat count as `5` .
    - Expand the **Variables** section and verify that the `FIRSTNAME` variable is present. Also, ensure that **Default Values** is selected in the **Variables** drop-down list.
    - Enter the email `abc01yz@xyz.com` (for this example) where you want to send the publishing notifications.
    - Ensure **Now** is selected as the publishing schedule.
15. Click **Publish** to start the publishing process.

When the publishing job is completed, verify the target schema database to review that the `First_Name` column in the published data includes random first names. This verification ensures that the `FIRSTNAME` variable is resolved correctly during the publishing process.

Key Board Support for Edit Generator Table

In CA TDM Portal when you create data generation rules, the following key board short cuts are supported to edit the generator tables.

- **Tab**
  Moves the focus to the next UI field.
- **Shift + Tab**
  Moves the focus to the previous UI field.
- **Ctrl + Space Bar**
  Opens Data Painter window for a focused cell in a table.
- **Escape**
  Exits from the current focused UI field.
- **Up/Down Arrow**
  Toggles between the options in an expanded list items or the toolbar options of a cell.
- **Left/Right Arrow**
  Toggles between the options in expanded list items or the toolbar options of a cell.
- **Enter**
  Performs the action relevant to focused UI field.

View Table Relationships

You can view table relationships in the CA TDM Portal while creating data generation rules. Table relationships help you understand how tables are related to each other. You can use this information to select appropriate tables for which you can write necessary data generation rules.

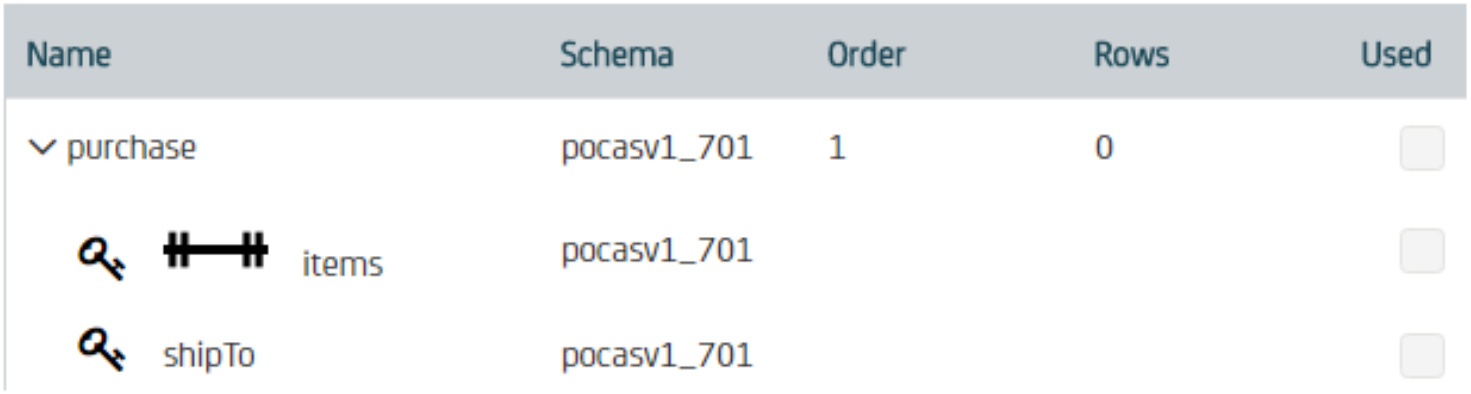
You can view the following information that is related to table relationships:

- List of tables that are related to a specific table.
- Foreign key relationships for a table.
- Parent-child relationships for a table.

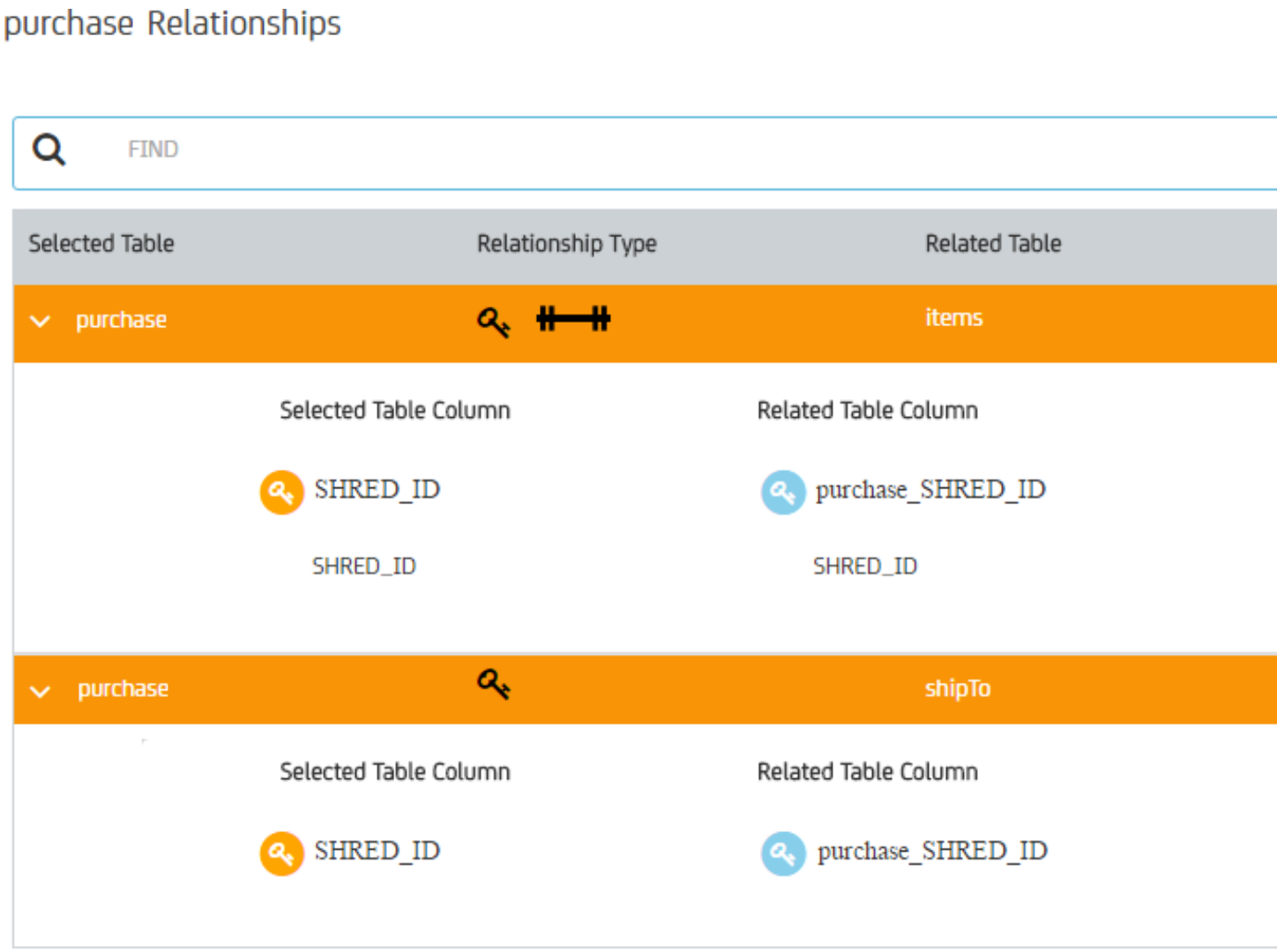
Table relationships information is available in the **Registered Tables** and **<Table_Name> Relationships** dialogs. You can access these dialogs when defining data generation rules. To view the table relationships information, follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click the **Generators** option in the left pane.
   The **Generators** page opens. This page lists the available generators.
   Note: If the left pane is not visible, click the icon (represented by three horizontal bars) in the top left corner.
4. Click the appropriate data generator for which you want to view table relationships.
5. Click the **Select Tables** button.
   The **Registered Tables** dialog opens.
6. Click the arrow (**>**) that is present before the table name to expand the table row.
   Note: Availability of an arrow before the table name indicates that the table is related to other tables.
   A list of all related tables appears.

7. Review the displayed relationships. The relationships are represented with the help of a key symbol and Crow's Foot notation.
   Note: The information that the Crow's Foot notation depicts comes from Datamaker (if it exists in Datamaker for the specific tables).
   The following example screen shot helps you understand how to decipher the relationships:



   In this example, the selected table "purchase" is related to two tables—items and shipTo—as follows:
   – A foreign key with *One to One* relationship exists between the "purchase" and "items" tables. This information is represented in the screen shot with the help of a key symbol and Crow's Foot notation.
   – A foreign key relationship exists between the "purchase" and "shipTo" tables.
8. To find detailed information about the relationships for the selected table (for example, purchase), follow these steps:
   a. Click the table in the **Registered Tables** dialog to display it in the **<Data Generator Name>** page (which is displayed in the background).
   b. Close the **Registered Tables** dialog.
   c. Click the Related tables icon for the table (for example, purchase).
      The **<Table_Name> Relationships** dialog opens.
   d. Review the relationships information in the dialog.
      The following example screen shot helps you understand the details of the relationships:



In this example, when you expand the relationship information between the "purchase" table and the "items" table, the following details are shown:

- In the first row, the "SHRED_ID" column is a primary key in the "purchase" table. This is represented with the help of a key symbol in orange color. This column is referenced by the "purchase_SHRED_ID" column in the related "items" table. Therefore, the "purchase_SHRED_ID" column is a foreign key in the "items" table. The key symbol in this case is shown in blue color.
- The second row shows the One to One relationship information between the columns; in this case, the columns are "SHRED_ID" for both the tables.

Similarly, when you expand the relationship between the "purchase" table and the "shipTo" table, the following details are shown:

- The "SHRED_ID" column is a primary key in the "purchase" table. This is represented with the help of a key symbol in orange color. This column is referenced by the "purchase_SHRED_ID" column in the related "shipTo" table. Therefore, the "purchase_SHRED_ID" column is a foreign key in the "shipTo" table, too. The key symbol in this case is shown in blue color.

This information enables you to identify the parent-child relationships between the selected and the related tables.

e. Click a table row in the **<Table_Name> Relationships** dialog to add that table to the background page (**<Data Generator Name>**).

9. Follow the usual steps as mentioned in Create Data Generation Rules to generate data generation rules.

Publish Data Using the CA TDM Portal

Publishing data means creating data in the target database schema or files. When you publish data, the data is generated based on the data generation rules that you define, and the data is added to the tables in the target database schema or files. The target database schema is the connection profile that you select at the time of publishing. The target table must already exist in the schema for the publish to work.

Select Project and Data Generator

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the **Project** drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Generators** in the left pane.
   The **Generators** page opens.
   Note: If the left pane is not visible, click the icon (represented by three horizontal bars) in the top left corner.
4. Click the data generator that you want to use to publish data.
   The **<Data Generator Name>** pages opens. This page includes the details about the data generator; for example, associated project and project version.
5. Click the **Select Tables** button to open the **Registered Tables** dialog.
   The dialog lists the registered tables (*used* and *unused* tables) based on the project version that is associated to the selected data generator.
   Note: Publish from a Teradata database with special characters in table or column names fails with an "Inconsistent Table definitions" error. Ensure that all characters are UTF-8 encoded.
6. Click the row corresponding to the *used* table that you want to use for data publishing.
   Note: To identify a *used* table, verify the presence of `Yes` in the **Used** column of the table. A *used* table is a table that includes data or for which you have already defined data generation rules.
   All the table rows are listed in the **<Data Generator Name>** page, including columns.
7. Review the data generation rules added to the table.
8. Click the **Publish** button.
   A dialog with the publishing options opens.

Configure Publish Options and Publish

Expressions in tables that use metafunctions such as @execsql with the "T" argument use the selected *target* profile when publishing. Expressions with the "S" argument use the selected *source* profile when publishing.

1. Select the appropriate schema from the **Publish From** drop-down list. This drop-down list includes all the schemas that are available for the selected data generator.
2. Specify the following for each table:
   - **Include In Publish**
     Specifies whether to include the table while publishing data or not. Only the tables that you checked are included in publish and the tables unchecked are excluded. You can generate and publish the data only for the included tables.
   - **Table Count**
     Specifies the number of times that you want to repeat each table while generating data. Enter a number to specify the table repeat. You can also use functions and variables to specify the Repeat count. Click the Data Painter icon to open the Data Painter dialog. The dialog lets you click on objects in the Functions and Variables sections. These objects transfer to the edit section where you manipulate them.
     Currently negative values are not supported in this field. In TDM Datamaker, you specify negative value in table repeat count field to publish a limited number of rows from large volume of data created within a table.
   - **Stored Columns**

(Optional) Specifies a comma-separated list of stored columns from registered tables and publish data. These tables are saved to the repository for later reuse in another publish job.

- **Publish Options**(Optional) Specifies what CA TDM Portal does if it encounters a record with the same key in the target database. Select one of the following options:
  - **Use publish level setting**
    Specifies that the publish job should use global setting for the selected data generator. This is the default behavior.
  - **Continue**
    Specifies the publish job skips duplicate records.
  - **Update**
    Specifies the publish job updates the duplicate record in the target database.
  - **Exit**
    Specifies that the publish job exits with an error.
- **Publish to Location**
  (Optional) Specifies different target locations for individual tables. If you want to publish to a different location than the registered location, you do not need to create a new connection profile. If the job is set to publish to a file then the location column is not shown.
  Default: The Connection Profile and Schema drop-downs specify the default target location for tables that do not have a Publish to Location set.

3. Review the information in the table.
   This table shows a preview of the registered *used* tables in the database schema that you selected in the previous step. This information lets you decide whether your publish adheres to the schema (in the target database) that you select in the next two steps.
4. Select the format to which you want to publish the data from the **Publish To** drop-down list. The available options are:
   - **Connection Profile**
     Lets you publish the data to a **Target Connection Profile**. If you use meta functions which specify the "S" argument, also define a **Source Connection Profile**.
     Select a Schema that is available in the target database from the **Select Schema** drop-down list. The schema for the source connection profile is set in the query specified in calls to @execsql, if applicable.

     WARNING

     1. If the Target is a **Sybase IQ database:**

     Ensure that you have modified the corresponding database configuration file (.cfg) in the Sybase IQ installed server for the following parameters:

     *- Modify the parameter **-c 48m** to **-c 64m***

     *- Modify the parameter **-gm 10** to **-gm 30***

     *- Add the parameter **-gn 45** at the end of the file*

     After modifying the configuration file (.cfg) of the corresponding database, restart the Sybase IQ database.

     2. If the target is a **TestMatch data pool** with **'Clear Down Existing Data'** option checked, in **Enterprise mode**:

     Ensure that the following line is present in your **application.properties** file:

     ```
     tdmweb.publish.enableTestMatchDataPoolCleardown=true
     ```

   For more information about connection profiles (for example, whether you can access a specific connection profile), see Create and edit Connection Profiles.

NOTE

You can set advanced connection pool settings for your Target Connection Profile. See Set Advanced Target Connection Settings below. These options are intended for Advanced/Network Admin users only.

– **File**
Lets you publish the data to a file. CSV, TXT, SQL, and FD text files are saved in Windows format beginning with a 2-byte BOM mark to indicate UTF-8 encoding, and terminated by \r\n. The file name is the same name as the generator but normalized, so that it makes an acceptable filename, for example, punctuation characters are converted to underscores.
The following file types are supported:
- **CSV** — Comma delimited table
- **TXT** — Tab delimited table
- **SQL** — SQL Statements File
- **FD** — Formatted Text File
  Note: Publishing to the FD file works only for the tables that are created by the registration of the G-T Excel or CSV file object. Therefore, ensure that you select only these tables while publishing the data to the FD file. If you try to publish other tables, the publishing fails.
- **XLS** — Excel 97-2003 Workbook with one worksheet per table
- **XLSX** — Excel Workbook with one worksheet per table
- **XML** — Well-forced XML file(s)

5. Specify the number of rows that you want to publish in the **Repeat** field. You can also use functions and variables to specify the Repeat count. Click the Data Painter icon to open the Data Painter dialog. The dialog lets you click on objects in the Functions and Variables sections. These objects transfer to the edit section where you manipulate them.
6. (Optional) Click Variables to expand the list of variables used in the selected generator. Currently variables table always shows default variables. if we select any other option than "Default", the UI table will not get refreshed with variables values. To override the values used for variables in publish job, follow one of the below methods:
   – **Manually override the default values**
   Select Default Values from the Variables drop-down. Identify the variable you want to override from the list and click the variable. In the Edit Variable Value dialog, modify the default value and click OK. Based on the Variable Type and Display Type you specified to create the variable, the Validate option is enabled. Click Validate to verify whether the modified value is part of the defined rule.
   When you modify the value of any variable, the dependent variable will resolve the hierarchy, if used. For example, a variable `Var1` includes `HR` as its value. Another variable `Var2` includes the variable `Var1` as its default value. So, the value of the variable `Var2` becomes `HR` . Now, you modify the value of `Var1` from `HR` to `Finance` . In this case, the value of the variable `Var2` also resolves to `Finance` .
   – **Override the default values using a CSV file**
   Select Values from a File from the Variables drop-down. Click the Save to CSV button to download a CSV file with the default variables. Modify the Variable values in the CSV file as necessary and save the file. Click the folder icon to open the Load Variable Values dialog. Drag the saved CSV file to the dialog and click the Load button. Uploads the CSV file and uses the variable values from the CSV file while publishing.
   If the uploaded CSV file includes only some of the used variables, then default values of the respective variables are only modified. For remaining used variables default values are applied. Any irrelevant variable information found in file is ignored.
   – **Override default values using Generator**
   Select Values from Generator from the Variables drop-down. Click the Generators button next to Variable drop-down. In the Select Generator dialog expand the project version and select an applicable Generator. Click OK.
   Notes:
   - The variables present in the selected Generator are not considered for evaluation. Only the variable names which exists as columns in used tables are considered as valid variables.
   - The selected generator is a dedicated variable container and can have only one used table. Any generator that has more than one used table will fail the validation.

7. Specify the correct email ID in the **Email** field.
   The CA TDM Portal sends an email that includes log files (related to the publishing operation) to the email ID that you provide in this field.
8. Select **Now** to publish the data immediately. Otherwise, Select **Schedule** and specify the applicable date and time to schedule the publishing.
9. Click **Options** to specify the action to perform, if there are duplicate values in the generated data. Following are the available options:
   - **On Duplicate In Data Target**
     Specifies what CA TDM Portal does if it encounters a record with the same key in the target database. Select one of the following options:
     - **Exit**
       Specifies that the publish job exits with an error. This is the default behavior.
     - **Continue**
       Specifies the publish job skips duplicate records.
     - **Update**
       Specifies the publish job updates the duplicate record in the target database.
   - **On Generated Duplicate**
     Specifies what CA TDM Portal does if it encounters duplicate values in the generated data. That means same values are generated for more than one row. Select one of the following options:
     - **Exit**
       Specifies that the publish fails when the first duplicate is identified in the generated data.
     - **Remove**
       Specifies that the publish carries on and removes the row that corresponds to the duplicate value.
10. Click **Publish**.
    A message with a Request ID for the publish operation is displayed. You also have the option to **Cancel the Request**, then the publish is created in a ‘Cancelled’ status. If you don't cancel, the publish job is added to the jobs queue in the **Submitted Requests** page.

Review, Cancel, or Re-submit Scheduled Jobs

Click **Submitted Requests** in the navigation pane, or click a **Request ID** link in the submit message to open the **Submitted Requests** table. This table includes all submitted jobs and additional details.

- Click the row in the **Submitted Requests** window that includes the Request ID.
  The **Additional Information** dialog opens. This dialog includes comprehensive information about the publish job.
- Press the **Cancel Request** button to unschedule a job that has not started yet.
  A job can only be canceled by its owner, or by an Administrator. You cannot cancel jobs that are already running.
- Press the **Re-submit Request** button to resume canceled jobs.
  A job can only be re-submitted by its owner, or by an Administrator.
  The re-submitted job goes into "Not Started" state and runs on the previously scheduled time. If the scheduled time is in the past, it starts immediately.

When the status of the job changes to **Completed**, the data has been published into the target database successfully.

Set Advanced Target Connection Settings

You can define parameters that control how Test Data Manager manages database connections. These settings override Apache Tomcat's default settings.

> **WARNING**
>
> Users should only make changes to these settings if they are confident that their database connection is problematic.

To set advanced connection pool settings for your Target Connection Profile, it is necessary to add the following lines to your `application.properties` file, located at `C:\Program Files\CA\CA Test Data Manager Portal\conf` in a standard installation:

```
tdmweb.TDMPublishService.db.spring.datasource.tomcat.initialSize=10
tdmweb.TDMPublishService.db.spring.datasource.tomcat.minIdle=10
tdmweb.TDMPublishService.db.spring.datasource.tomcat.maxIdle=100
tdmweb.TDMPublishService.db.spring.datasource.tomcat.maxActive=100
tdmweb.TDMPublishService.db.spring.datasource.tomcat.maxWait=30000
tdmweb.TDMPublishService.db.spring.datasource.tomcat.testWhileIdle=false
tdmweb.TDMPublishService.db.spring.datasource.tomcat.timeBetweenEvictionRunsMillis=5000
tdmweb.TDMPublishService.db.spring.datasource.tomcat.minEvictableIdleTimeMillis=600000
tdmweb.TDMPublishService.db.spring.datasource.tomcat.removeAbandoned=false
tdmweb.TDMPublishService.db.spring.datasource.tomcat.logAbandoned=false
tdmweb.TDMPublishService.db.spring.datasource.tomcat.removeAbandonedTimeout=60
tdmweb.TDMPublishService.db.spring.datasource.tomcat.abandonWhenPercentageFull=0
tdmweb.TDMPublishService.db.spring.datasource.tomcat.maxAge=2200
tdmweb.TDMPublishService.db.spring.datasource.tomcat.suspectTimeout=0
tdmweb.TDMPublishService.db.spring.datasource.tomcat.defaultTransactionIsolation=1
```

All values above are the parameters' default values.

For more information, see https://tomcat.apache.org/tomcat-7.0-doc/jdbc-pool.html.

NOTE

These lines are not present in the standard application.properties file, as supplied with CA TDM.

Create and Manage Generator Configurations

Publishing data implies creating data in the target database schema. When you publish data, the data is generated based on the data generation rules that you define and is added to the tables in the target database schema. You can create Configurations defining the used tables and other criteria to repeatedly use the saved configuration for publishing data.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version from the Project drop-down list in the top blue bar.
   This selection sets the required project and version context for all the related operations that you perform in the Portal.
3. Click **Generators** in the left pane.
   The **Data Generators** page opens.
   Note: If the left pane is not visible, click the icon (represented by three horizontal bars) in the top left corner.
4. Click the data generator that you want to use to publish data.
   The **<Data Generator Name>** page opens. This page includes the details about the data generator; for example, associated project and project version.
5. Click the **Configurations** button.
   The Configurations page opens that includes the list of existing configurations associated to the selected Generator.
6. Click the **New Configuration** button.
   The **New Configurations** page opens.
7. Enter information in the following fields:
   - **Name**
     Specifies the name of the configuration.
   - **Description**
     Specifies the brief description for the configuration.

8. Review the information in the following fields and modify the values as necessary.
   a. **Publish From**
   b. **Publish To**
   c. **Repeat**
   d. **Options**
   For more information about what each of these fields specify, see Publish Data Using the CA TDM Portal.
   In TDM Portal when you publish from a generator directly without using the configuration, the publish screen includes some fields such as Stored Columns, Publish to Location, Source Connection Profile, and Variables in addition to the ones listed in this article. Those fields are not applicable when you are publishing using a configuration and do not affect the data publish.

   Notes:
   – You cannot remove a table from the configuration though the data generation rules for the respective table are deleted. This does not affect the data publishing.
   – You cannot add the tables to the configuration, that are registered to the corresponding project version after creating the configuration. Any table that is registered to a project version prior to creating the configuration can only be added to the respective configuration.
   – When you publish data using the configuration, you cannot modify the default values of the used variables.
9. Click **Save**.
   Configuration is successfully saved and added to the list on the Configurations page.
   Verify that the saved configuration is available in the Configurations page. Click **Cancel** to go back to Configurations page.
10. (Optional) After saving the configuration you can enable the configuration for tester self service or you can publish the data from the configuration. For more information, see the following topics.

Enable Generator Configuration for Tester Self Service

The tester self service flows designed with publish block includes the option to select a configuration. You can execute the publish job based on the criteria specified in selected configuration. After creating the configurations, you can enable them to use in tester self service flows.

Follow these steps:

1. Go to application.properties file and ensure that Tester Self Service is configured to use the new Publish Engine introduced in CA TDM release 3.8. Verify the following settings flag to ***false***:
   – ***tdmweb.tdmJobEngineService.useDatamakerToPublish=false***
   – ***tdmweb.testerSelfService.useDatamakerToPublish=false***

   WARNING

   If you modify the values of these parameters in *application.properties* file, you must restart the TDM Portal service.
2. Access the CA TDM Portal.
3. Select an appropriate project and its corresponding version.
   This selection sets the required project and version context.
4. Click **Generators** in the left pane.
   The Data Generators page opens.
5. Click the data generator that you want to use to publish data.
   The **<Data Generator Name>** page opens.
6. Click the **Configurations** button.
   The Configurations page opens that includes the list of existing configurations associated to the selected Generator.
7. Identify and click the configuration that you want to enable for tester self service.
   The **Edit Configurations** page opens.
8. Select the **Configuration Active** check box and click **Save**.

The configuration is successfully enabled for tester self service. The testers now can see the configuration in the Select Configuration drop-down list when they execute the self service flow designed with publish block.

Publish Data using Generator Configuration

You can repeatedly use the saved configuration for publishing data. The data generation rules and criteria specified in the saved configuration.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version
3. This selection sets the required project and version context
4. Click **Generators** in the left pane.
   The Data Generators page opens.
5. Click the data generator that you want to use to publish data.
   The <Data Generator Name> page opens.
6. Click the **Configurations** button.
   The Configurations page opens that includes the list of existing configurations associated to the selected Generator.
7. Identify and click the configuration that you want to use for publishing data.
   The **Edit Configurations** page opens.
8. Review all the fields and modify as necessary and click **Save**.
9. Enter the relevant information in the following fields.
   - **Email**
   - **Schedule**

   For more information about what each of these fields specify, see Publish Data Using the CA TDM Portal.
10. Click **Publish**.
    A message with a job ID for the publish operation is displayed. You can review the job status from Requests table. For more information, see Publish Data Using the CA TDM Portal.

Edit Generator Configuration

If you want to update the configuration information to reflect changed requirements or use cases, you can edit the saved configuration.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version.
   This selection sets the required project and version context.
3. Click **Generators** in the left pane.
   The Data Generators page opens.
4. Click the data generator that you want to use to publish data.
   The **<Data Generator Name>** page opens.
5. Click the **Configurations** button.
   The Configurations page opens that includes the list of existing configurations associated to the selected Generator.
6. Identify and click the configuration that you want to edit.
   Edit Configuration page opens.
7. Update the information as necessary and click **Save**.
   The changes are saved and the updated configuration becomes available.

Delete Generator Configuration

If you do not need a configuration in your environment, you can delete it from the Portal.

Follow these steps:

1. Access the CA TDM Portal.
2. Select an appropriate project and its corresponding version
3. This selection sets the required project and version context
4. Click **Generators** in the left pane.
   The Data Generators page opens.
5. Click the data generator that you want to use to publish data.
   The <Data Generator Name> page opens.
6. Click the **Configurations** button.
   The Configurations page opens that includes the list of existing configurations associated to the selected Generator.
7. Locate the configuration that you want to delete.
8. Click the **Delete Configuration** icon (X icon) in the row corresponding to the identified configuration.
   A confirmation dialog opens.
9. Confirm the delete action.
   The configuration is deleted from the list.

Create and Manage Publish and Table Actions

The CA TDM Portal lets you create **Publish** and **Table** actions, that you can execute before or after data generation.

- **Pre-publish actions** are actions that you want to execute before you perform the publish. For example, by creating a pre-publish action, you can clear down some specified columns.
- **Post-publish actions** are actions that you want to execute after you perform the publish. For example, by creating a post-publish action you can update some columns using SQL.

> **NOTE**
>
> You cannot create, edit or execute Actions of type **HOST** or **WORKFLOW** from within TDM Portal in Docker. For more information, see TDM Portal REST ActionService container.

Enable HOST Actions

HOST actions can manipulate files and folders and execute commands with system level privileges on the CA TDM server. When users try to run a job that contains a HOST action, the back-end returns the following error by default:

```
HOST actions are not enabled. Contact your admin.
```

To allow authorized CA TDM users to run HOST actions, the CA TDM administrator must enable this feature.

1. Navigate to the directory where you installed the CA TDM Portal, and open the "conf/" sub-directory.
2. Locate and open the `application.properties` file in a text editor.
3. Enable the execution of host actions by setting the following flag to true.
   ```
   EnableHostActions={true|false}
   ```
   Default: false
   - **false** — specifies that users cannot run HOST actions. Users can still create HOST actions from the CA TDM Portal, and are still able to see HOST actions created for the job in the list.
   - **true** — specifies that users can run HOST actions with system level privileges.
4. Restart the CA TDM service.

Create an Action

1. Access the CA TDM Portal.
2. Select a project and version from the Project drop-down list available in the portal header. Optionally, click the gear icon (next to Project drop-down in the portal header) to search for a specific project.
   If you want to create a new project, see Create and Edit Projects.
3. Click **Generators** in the left pane.
   The **Generators** page opens and lists the existing generators created for the selected project and version.
   If you want to create a new generator, see Create Data Generator.
4. Click the generator for which you want to create publish actions.
   The **Generator Details** page opens.
5. Click the **Actions** button.
   The **Actions** page opens.
6. Select an action type from the **Actions** drop-down. The following options are available:
   - **Publish**
     Executes the action every time you perform publish data generation.
   - **Table**
     Executes the action only when the specified used table is associated in publish data generation.
7. Click the **Create Publish Action** or **Create Table Action** button based on the action type you selected.
   The **New Publish Action** or **New Table Action** page opens.
8. Enter the following information:
   - **Name**
     Defines a name for the action that you want to create.
   - **Description**
     Defines a brief description of the action that you want to create.
9. Select the **Code Type**. The following options are available:
   - **Host**
     To run an actual program on the TDM server.
   - **Javelin**
     To create and execute Javelin actions for a generator in the CA TDM Portal. For more information, see Using Workflows in CA TDM Portal.
   - **SQL**
     To execute an SQL statement in the environment into which you publish data.
   - **REST**
     To perform a call to a REST API endpoint.
10. (**Table** Action only) From the Table drop-down list, select the table on which you want to perform the Action.
11. (Code Type **Host** only) Specify the following fields:
    - **Command**
      Defines the command to execute on the host machine as system user.
      Example: This command shuts down the computer after publishing:
      ```
      cmd /c shutdown -s
      ```
    - **Wait for Completion**
      Specifies whether you want to run the action of type Host synchronously. If so, enable this option and specify a value in the **Timeout** field.
    - **Execution Timeout**
      Specifies the time (in seconds) for which the Portal waits for the completion of the action. If the action does not complete before the timeout limit, it is terminated and a failure is returned for the action. If the value of timeout field is greater than 0, the action is considered as synchronous. For all other cases (including not providing the value), the action is considered as asynchronous.
    - **Success Required**

Specifies whether invoking the program is required or optional. Select this check-box, if invoking the specified program is mandatory. If invoking the program fails, the publish will not be completed.

12. (Code Type **SQL** only) Specify the following fields:
    - **DB Connection**
      Specifies the Database on which you want to execute the SQL. Select a database from the drop-down list.
    - **Use**
      You can use either Stored SQL or Direct Code to execute the publish action. Specify one of the following as required:
      - **Stored SQL**
        Select the stored SQL from the drop-down list.
      - **Direct Code**
        Enter the code required to execute the SQL. Separate multiple queries with a semicolon.
        Example:

```
delete from [dbo].[BLOB_TABLE_9];
delete from [dbo].[BLOB_TABLE_20];
delete from [dbo].[BLOB_TABLE_8];
```

    - **Success Required**
      Specifies whether successful execution of the SQL is required. If you select this check-box, and execution of the SQL fails, the publish is not completed.
    - **Enable Concurrency**
      Specifies whether each query runs in its own thread when more than one query is inserted in the publish action. All queries run concurrently up to a maximum of 10 threads. When the number of queries to run is larger than the maximum number of threads, the remaining queries are put in a queue waiting for a thread to be available.
      Note: The Test Data Engineer can configure the maximum number of threads through the `tdmweb.publish.action.query.maxthreads` entry in the application.properties file.
    - **Success Criteria**
      Specifies the criteria to meet for successful execution of SQL. Select one of the following two options and specify the criteria as required:
      - Results
      - Row Count
    - **Execution: Wait for Completion**
      Specifies whether to terminate run-away queries if the query exceeds the timeout value. If more than one query is inserted in the publish action, the behavior depends on the concurrency setting:
      - If Concurrency is enabled, any query is terminated when it takes longer to run than the timeout value.
      - If Concurrency is disabled, the timeout value applies to the cumulative execution time of all queries. When the timeout is exceeded, the currently executing query is terminated; all remaining queries that have not yet started executing do not start.
      - **Timeout**
        Specifies the time (in seconds) for which the Portal waits for the completion of the action before it terminates the queries.
13. (Code Type **REST** only) Specify the following:
    - **REST Action URL**
      Full REST API URL. For example: http://my.action.com
    - **Action Secret**
      This value acts as a security measure for the target API URL, and its contents are concealed as you type. Check what security protection is in place on the target API; if there is no such security protection on the target API, this value is not used.

NOTE

This value appears in the REST API body that TDM generates as a *hashed* value using the hashing algorithm sha-256.

For example, if you enter 'marmite' in the **Action Secret** field, the value of parameter `secret` in the API body is `7362DEDB2123ABABBD1446A395A7235DE3DFADF91F173961DABDE8FECFDBFC1C`.

- **Variables for the Action**
  Enter any substitution variables you want to use in your TDM commands, separated with commas. You can also define (or redefine) variables with the syntax varName="value".
  For example: `CDATE, my_variable="red", client_name, another_variable`

  NOTE

  Enter all variables **without** the tilda (~) symbol before and after.

  These variables appear in the parameters section of the API body that TDM generates, with their associated default values. For more information, see Create and Manage Variables.
- **Identity Delegation**
  Choose whether the user that performs the Action is you, or a **Custom User**. For a Custom User, enter User Name and Password fields.

14. Select the Execute Action to determine whether the action is a Pre-Publish or Post-Publish action. Following are the available options:
    - **Before (Pre-Publish)**
      Specifies the action to complete before the publishing starts.
    - **After (Post-Publish)**
      Specifies the action to perform after the publishing completes.
15. Click **Save**.
    The publish action is successfully created. The created publish actions for executing Before and After are added to Pre Publish Actions and Post Publish Actions respectively on the Actions page.
    Repeat the above steps to create more publish actions.

Re-order Actions

1. Access the CA TDM Portal.
2. Select a project and version from the Project drop-down list available in the portal header. Optionally, click the gear icon (next to Project drop-down in the portal header) to search for a specific project.
3. Click Generators in the left pane.
   The Generators page opens and lists the existing generators created for the selected project and version.
4. Click the generator to see the existing actions.
   The generator details page opens.
5. Select Publish or Table from Actions drop-down to see the respective type of existing actions.
   The Actions page opens and lists the Publish Actions or Table Actions based on the action type you selected.
6. Identify the actions that you want to re-order and click the upward arrow or the downward arrow to move the respective action up or down. The actions are executed in the order they appear in the list.

Execute an Action

1. Access the CA TDM Portal.
2. Select a project and version from the **Project** drop-down list available in the portal header. Optionally, click the gear icon (next to Project drop-down in the portal header) to search for a specific project.
3. Click **Generators** in the left pane.
   The **Generators** page opens and lists the existing generators created for the selected project and version.
4. Click the generator to see the existing publish actions.

The **Generator Details** page opens.

5. Select **Publish** or **Table** from the **Actions** drop-down to see the respective type of existing actions.
   The Actions page opens and lists the Publish Actions or Table Actions based on the action type you selected.

   NOTE
   For the Publish actions of the type Host and SQL, the **Wait for Completion** and **Timeout** options also apply. Additionally, you can specify a different value in the **Timeout** field and execute your action using that value. This way you can execute your publish actions with different timeout values. You can then edit your action if you want to change the execution timeout value based on your testing. The new timeout value is not persistent; that is, you cannot save this value to override the original value, which you specified at the time of creating your action.
6. Identify and click the forward arrow (>) in the row that corresponds to the action that you want to execute.
   The Portal executes the respective Action and shows a message with the success or failure information.

Edit an Action

1. Access the CA TDM Portal.
2. Select a project and version from the Project drop-down list available in the portal header. Optionally, click the gear icon (next to Project drop-down in the portal header) to search for a specific project.
3. Click **Generators** in the left pane.
   The **Generators** page opens and lists the existing generators created for the selected project and version.
4. Click the generator to see the existing publish actions.
   The **Generator Details** page opens.
5. Select **Publish** or **Table** from the Actions drop-down to see the respective type of existing actions.
   The **Actions** page opens and lists the existing Publish Actions or Table Actions based on the action type you selected.
6. Identify and click the action that you want to edit.
   The Edit Publish Action or Table Action page opens based on the action type.
7. Modify the details as necessary and click **Save**. You can edit all the details except the Code Type.

Delete an Action

1. Access the CA TDM Portal.
2. Select a project and version from the **Project** drop-down list available in the portal header. Optionally, click the gear icon (next to Project drop-down in the portal header) to search for a specific project.
3. Click **Generators** in the left pane.
   The **Generators** page opens and lists the existing generators created for the selected project and version.
4. Click the generator to see the existing publish actions.
   The **Generator Details** page opens.
5. Select **Publish** or **Table** from the Actions drop-down to see the respective type of existing actions.
   The **Actions** page opens and lists the existing Publish Actions or Table Actions based on the action type you selected.
6. Identify and click the cross icon (X) in the row that corresponds to the action that you want to delete.
   A confirmation dialog opens.
7. Click **Delete**.
   A message confirms the successful deletion of the action.
8. Review the actions list to verify that the deleted action is no longer available in the table.

Publish and Export Non-Relational Data using Self Service Catalog

TDM Portal Self Service Catalog (Tester Self Service) now supports publishing the XML, XSD, WSDL and JSON files and exporting the request results. Based on the object type you supplied as input, you can export and download the request results in the same file type from Requests page.

As a Test Data Engineer you can create a self service flow that performs both publish and export jobs for non-relational data sources.

Note: The test engineer can publish the data using the same connection profile that is used to export the non-relational data sources. The test engineer must have permissions to the respective connection profile. For more information about adding users and assigning permissions, see Groups and Users.

Follow these steps:

1. Launch Datamaker and go to data pool that you created for non-relational data source.
2. Create a variable with the name " ***~SHRED_GROUP_ID~*** " for the respective generator (data pool). The variable name is case sensitive and to be entered in upper case only.
3. Go to shredded tables and specify the variable in "shred_group_id" column of each row.
4. Go to CA Agile Requirements Designer and create a visual flow and add a process block.
5. Double click on the process block added to the flow and do the following:
   a. Go to the Make System Data tab, click Set Publish.
   b. Select the respective data pool and do the following:
      - Select the variable " ***~SHRED_GROUP_ID~*** ".
      - Choose the option "**Use in TDoD**".
   c. Select other variables as required.
   d. Click OK.
   e. Go to the Test Data tab and do the following:
      a. If you are using CA Agile Requirements Designer 1.9.3 or earlier:
         a. Click the Add Variable/Value Pair button.
         b. Enter the variable name as **SHRED_OBJECT_ID** in the first text box.
         c. Enter the variable value in the second box. The variable value is the **Object ID**. Go to Registered Objects list of the respective project, version, find the request and enter the respective Object ID in this field.
         d. Click Save.
      b. If you are using CA Agile Requirements Designer 1.9.5 or later:
         a. Click the Add Variable button and then click New Variable.
         b. In the Add Variable dialog specify the following:
            - **Name**
              Specifies the name for the variable. Enter the variable name as **SHRED_OBJECT_ID** in this field. The variable name is case sensitive and to be entered in upper case only.
            - **Description**
              Specifies a brief description of the variable.
            - **Type**
              Specifies the type of variable. Select **Integer** from the drop-down list.
            - **Value**
              Specifies the Object ID of the request created for object registration. Go to Registered Objects list of the respective project, version, find the request and enter the respective Object ID in this field.
         c. Click OK. Then click Save.
6. Repeat step 5 for each process block that you want to export in the current flow.
7. Expose the flow to Test Data on Demand.
   Now the testers can see the flow in TDM Portal Self Service Catalog based on their user privileges. When the flow is executed, based on the conditions specified, publish and export jobs can be performed directly from the flow.

Configure Publishing Behavior

For publish jobs, you can define how CA TDM Portal processes batches, and what to do with already processed batches if an error occurs. The settings in the application.properties file are global and apply to all publish jobs. The settings in the CA TDM Portal are per publish job.

Follow these steps:

1. Navigate to the directory where you installed the CA TDM Portal, and open the `conf` subdirectory.
2. Locate and open the `application.properties` file in a text editor.
3. Define what CA TDM Portal does if an error occurs during publishing.
   `tdmweb.publish.actionOnError=exit|rollback`
   - **exit** — Specifies that any records already published to the target database remain in the target database.
   - **rollback** — Specifies that any records already published to the target database are rolled back.
     Default: exit
4. Specify how CA TDM Portal inserts records into the target database.Note: This parameter is effective only if the target database supports batch inserts, and if the `actionOnDuplicate` parameter for the publish job (See Publish Data Using the CA TDM Portal) is set to the default value of "exit".
   `tdmweb.publish.batchCommit=false|true`
   - **true** — Specifies that records are inserted in batches in the target database.
   - **false** — Specifies that records are inserted individually to the target database.
     Default: false
5. Define the batch size.
   Note: This parameter is only effective if the parameter tdmweb.publish.batchCommit is set to true.
   `tdmweb.publish.iterationsBeforeCommit=`*n*
   - **n** — Specifies the number of iterations before a batch is committed to the target database.
     Default: 1. This means, commit after each iteration.
     Limits: If you specify a value greater than the repeat count, it defaults to a value equal to the repeat count, and one commit will be done after it completed all iterations. If you specify a value less or equal to 0, it defaults to 1.
6. Save the application.properties file.
7. Restart the CA TDM Portal service.

Configure Test Data Reservation Service

Test Data Manager lets you reserve data for exclusive use by testers to avoid conflicts. You can manage this process in one of the following ways:

- Dynamic Test Data Reservation Service
  In this case, you use the CA TDM Portal to create test data models that facilitate the data reservation process for testers. Defined test data models are shared with testers as forms in the Self-Service Catalog section of the Portal. These forms use the business language that testers understand, which helps testers find the right test data for their testing environments. Testers can access the applicable forms and can perform test data operations - find and reserve. Through these forms, testers get on-demand access to the exact data they need. That is, they can easily request the data they need, analyze it, and reserve it on demand.
- Form Based Test Data Reservation Service
  In this case, you use the Datamaker UI to create a test data mart and configure test matching. After you configure test matching rules, you design the visual flow using CA ARD and publish the forms for the testers to get on-demand access. The testers can access these forms, from CA TDM Portal Self-Service Catalog interface that lets testers dynamically request and receive data to execute their test cases.

Configure Dynamic Test Data Reservation Service

To comprehensively test any application, testers need the right test data to be available *on demand* to conduct varied testing scenarios. Typically, in an enterprise, this data is spread across multiple data sources. This further compounds the challenge of creating a test data mart and finding the data that replicates the production environment and is available on demand. Manually creating the test data mart is not an efficient option and can result in data that has insufficient test coverage, leading to defects in production. Additionally, testers need the capability to easily find and reserve the test data, whenever they want, based on their enterprise environment requirements.

The CA TDM Portal helps organizations address these challenges by managing the full life-cycle of test data reservation. The Portal simplifies the overall data reservation process by encapsulating the automatic creation and management of test data marts. This ability eases the management and maintenance of test data reservation services. The Portal also helps ensure that the data becomes available to testers in minutes, eliminating the time that is otherwise wasted looking for or preparing data, or creating it where none exists.

Test data engineers (TDEs) use the Portal to create test data models that facilitate the data reservation process for testers. Defined test data models are shared with testers as dynamic forms in the Self-Service Catalog interface. These forms use the business language that testers understand, which helps testers find the right test data for their testing environments without any issue. Testers can access the applicable forms and can perform test data operations—find and reserve. Through these forms, testers get on-demand access to the exact data they need, allowing them to easily find, view, analyze, and reserve the data.

The following topics cover all the information:

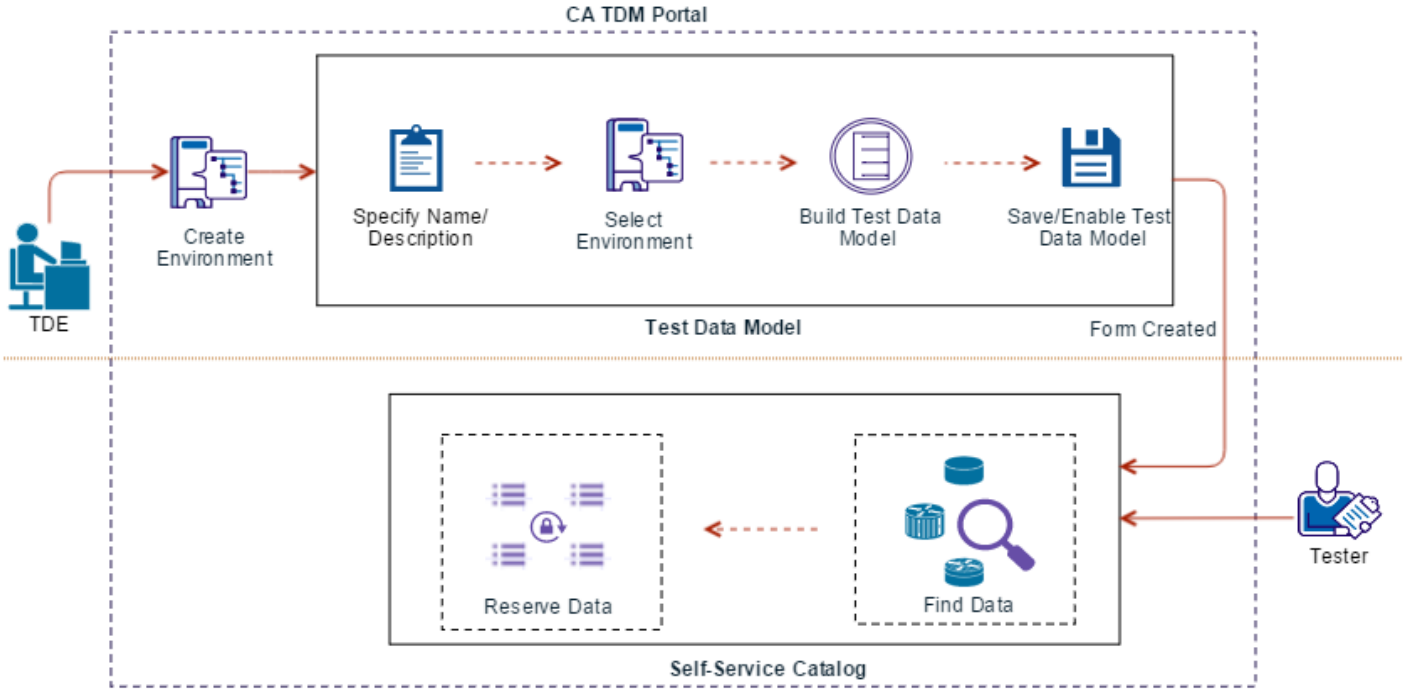
Tutorial Video

Watch the following video for a visual walk-through of a use case of using CA TDM Portal to create a test data model:

High-Level Process

The following illustration shows the complete data reservation process, which involves the TDE and tester personas:

Figure 36: Data Reservation Using the Portal



Understand the Terminology

Review the following terms:

Environment

An environment is a collection of data sources that are associated with an application. Each data source maps to a connection profile that is related to a specific project version in the CA TDM Portal. An environment is used in defining a test data model. Test data search and reservation are also performed in the context of the selected environment.

Review the following points:

- Each environment is specific to a project and version.
- You can create multiple environments.
- Do not use the same connection profile in different environments.
- Reservations performed in an environment are not migrated when the environment is moved to the latest version.
- Only relational data sources are supported currently.
- A connection profile is required for any environment's data source to work. For more information about connection profiles, see Create and edit Connection Profiles. Typically, a schema name is optional for the Microsoft SQL Server or DB2 connection profile that you create. But, if you use the same connection profile in an environment, ensure that you provide a valid schema name for that connection profile.
  - For Microsoft SQL Server, use **SQL Server Schema Name** to provide the schema name in the connection profiles page.
  - For DB2, use **Additional Connection Properties** to provide the schema name in the connection profiles page. **Note:** For model-based test data reservation service, the Portal takes only the first entry in **Additional Connection Properties**; it does not consider the remaining entries (if added). For example, if you use

`libraries=Schema1,Schema2,Schema3` , the Portal considers only the first entry `Schema1` in this case; it ignores the remaining values.

- TDEs must share the required connection profiles with testers.
- Mapping between entities and data sources across environments must be the same within a project version.
- Once you define a data source name for an environment, you cannot change the name across multiple environments available for the same project version. However, you can assign a different connection profile to the data source.

Test Data Model

A test data model lets test data engineers combine related data elements from multiple data sources, which reside on different servers, into a single container. Testers can then perform test data operations (for example, find or reserve) on the defined model. Test data models abstract the technical aspects of the test data and represents those aspects in a domain-specific language to testers. Test data models, therefore, enable testers to easily understand the data model and define their data requirements.

Review the following points:

- You can use a single test data model (form) against multiple environments for a specific project and version.
- TDEs can mark the test data models as visible or not visible to testers after they create them. Only those test data models that are marked as visible are accessible to testers in the Self-Service Catalog section of the Portal as forms. TDEs, however, can access the test data models irrespective of the visible state.
- When a TDE deletes a test data model, all the data reservations associated with that test data model are invalidated. The same test data then becomes available to testers, which they can reserve through other test data model forms.
- The CA TDM Portal supports only one test data model per project version, though it allows you to define multiple test data models for a version. Therefore, do not create more than one test data model for a version.
  The same behavior is applicable for APIs too.

Model Key

A model key is a data element (or set of data elements) that uniquely identifies the resource that needs to be reserved. For example, in an Order Management System, `orderid` uniquely identifies orders in the system. Similarly, `policyid` in an insurance system can act as a model key. A model key drives the reservation of records across the test data model. The entity that contains the model key is called the root entity of the test data model.

Review the following points:

- A model key can be composed of one or more data elements of an entity (for example, table), but both data elements must belong to the same entity.
- You cannot modify a model key after you define it for a test data model.

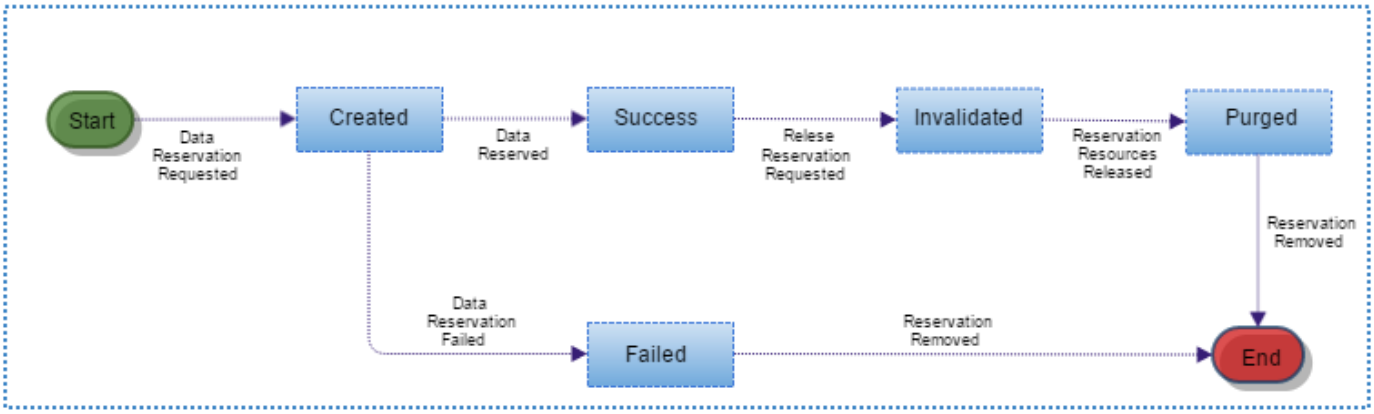
Data Element

A data element provides the ability to filter the test data.

Data Reservation

Data Reservation is the process of finding relevant test data and reserving the model keys to exclusively use in the execution of specific test cases. In the CA TDM Portal, when a data reservation request is submitted, the request is processed through various states. The following diagram illustrates the various states that a data reservation request passes through:

Figure 37: Data Reservation State Diagram



The reservations are permanently deleted after 30 days from the time it reaches the "Purged" or "Failed" state. By default, the delete process runs once in every 12 hours to identify whether any purged reservations are older than 30 days. You can configure the default values. For more information, see Configure CA TDM Portal for Deleting the Purged Reservations.

Note: If any model key value is NULL in a reservation, the reservation is not allowed.

Tasks Based on Personas (TDE and Tester)

As illustrated in the diagram, the complete process involves two personas: TDE and tester.

Test Data Engineer Tasks

A TDE performs the following tasks:

Note: A tester cannot perform these tasks.

1. Create an environment.
   a. Specify name, description, and connection profile.
2. Create a test data model.
   a. Specify the test data model name and description.
   b. Select an environment.
   c. Build a test data model using the data explorer.
      a. Create a model key.
      b. Add data elements to build the test data model.
3. Enable a test data model for testers.

Tester Tasks

A tester performs the following tasks:

Note: A TDE can also perform these tasks, but the primary audience for these tasks is a tester. For more information about how to perform these tasks, see Tester Self-Service.

1. Find test data.
   a. Select the appropriate form from the Self-Service Catalog.
   b. Select the environment.

c. Define data filters.
d. Find the data.
e. Review the data.

2. Reserve test data.
   a. Select the reviewed data.
   b. Submit the reservation request.
   c. Review the reservation.

Considerations

Review the following considerations:

- In the current version of the Find and Reserve capability, there is a limitation with the usage of the connection profiles with the Oracle database. The Oracle connection profiles created in the CA TDM Portal need users to have schema ownership. If a user does not have permissions on all the schemas used for the Find and Reserve, then the functionality will not work.
- The current version of Find and Reserve capability supports the following data sources:
  – Microsoft SQL Server
  – Oracle
  – DB2
- TDEs must have knowledge of the commonly used application objects that testers use, relationships between those objects, and the way they map in the underlying database.
- TDEs must ensure that the required entities are already registered in the Portal.
- TDEs must define relationships between entities in a test data model.
- The following data types are not supported in a test data model:

- CHAR
- NCHAR
- BFILE
- BINARY
- BLOB
- CLOB
- GEOGRAPHY
- GEOMETRY
- HIERARCHYID
- IMAGE
- INTERVAL DAY TO SECOND
- INTERVAL YEAR TO MONTH
- LONG
- LONG RAW
- LONGVARBINARY
- NCLOB
- NTEXT
- RAW
- ROWID
- SQL_VARIANT
- TEXT
- TIMESTAMP WITH LOCAL TIME ZONE
- TIMESTAMP WITH TIME ZONE
- UROWID
- VARBINARY
- XML

Example Scenarios

The Example: Order Management System article includes an example scenario that explains the complete end-to-end process of designing and consuming dynamic test data services, covering all the TDE and tester tasks.

APIs for Designing and Consuming Automated Test Data Services

You can use the exposed APIs to design and consume automated test data services.

NOTE

For more information about how to use the related APIs, see the following articles:

- Use APIs to Design and Consume Automated Test Data Services
- Use APIs to Manage a Test Data Model
- Use APIs to Manage Associations in a Test Data Model
- Use APIs to Manage Fields (Data Elements) in a Test Data Model

Create and Edit a Find & Reserve Model

A test data model consists of one or more data entities organised in a hierarchy. This starts with one root entity, which has associations to other entities. Each data entity is identified by its physical name (for example, table name in the case of the relational model) and the data source in which it is available. As a Test Data Engineer (TDE), you can identify and add data elements to the entity - these elements are identified by their physical names (for example, column names in a table

in the case of the relational model) and by logical names. An entity can contain other child entities through associations. An association defines the relationship between those entities.

This page contains the following sections:

You can follow a worked example of the Test Data Model creation process in the context of a test scenario here: Example: Order Management System.

When a test data model becomes accessible to testers, they can use it to define their data requirements.

Types of Find & Reserve Models

Standard Find & Reserve Models

From version 4.6, CA TDM can create Find & Reserve models from the data model that you create in the Data Model section of the CA TDM Portal. This new process removes your requirement to register tables directly from their original environments.

> **WARNING**
>
> Before you create a Find & Reserve Model, ensure that there is no sensitive data in the Data Model that you use to create the Find & Reserve Model. You can do this with the PII Scan feature within the CA TDM Portal. For more information, see Scan Data Model for PII.

Legacy Find & Reserve Models

You can still create Legacy Find & Reserve data models as in previous versions. Legacy models require you to register tables directly from their original environments.

Create a Find & Reserve Model

To allow testers to find data and reserve it for their use, you can create a Find & Reserve Model. There are 2 kinds of model that you can create, as follows:

- **Create Standard Model**
  A Standard Data Model includes the following features not present in the Legacy Model:
  - This model uses the data model that you create in the Data Model section of the CA TDM Portal. You no longer need to register tables in the Objects section of the Portal.
  - Support for DB2/zOS data sources. See a comprehensive list of data types and database types that CA TDM supports at Data Types Supported by Find & Reserve.
  - Support for fixed-length CHAR data types.
  - Either use Data Prefetch or create a Reservation Table manually.
- **Legacy Find & Reserve Model**
  - You should use this model if you require support for compound relationships. New F&R Models do not currently support compound relationships.
  - Legacy models do not support DB2/zOS data sources.

Create a standard Find & Reserve Model

This is the process to create a standard Find & Reserve Model.

Follow these steps:

1. Access the CA TDM Portal.
2. Select the required project and version from the **Project** drop-down list.
3. Expand **Modeling** in the left pane.

4. Click **Find & Reserve**.
   The **Models for Find & Reserve** page opens.

   NOTE

   All test data models that you create for the selected project and version are listed in this page.
5. Click **Create Model**.
   The **Test Data Model** page opens.
6. Enter the following information:
   – **Name**
     Specifies an appropriate name for the test data model.
   – **(Optional) Description**
     Specifies an appropriate description for the test data model.
   – Choose a Data Prefetch option.
7. Click **Next**.
   The **Find & Reserve Model** page opens.
   On this page, the left pane displays a tree structure of the data model that you created in the **Data Model** section. If you do not have a Data Model defined, the left pane displays 'You currently have no Data Sources. Please run Data Discovery'.
   The right pane displays the contents of the Find & Reserve Data Model that you will create. At the start of the process, the right pane is empty.
8. **Expand tables**
   Click the chevron on the left of the Data Model, Database, Schema and Table names, to expand that element.
   If you type a text string into the **Search** field above the left-hand pane, only tables whose names match that string, within the expanded schema, are displayed.
   When you expand a Table element, columns from that table display underneath it.

   NOTE

   The Find & Reserve feature does not support all data types on all databases. The names of columns whose data types are not supported appear in grey. See Data Types Supported by Find & Reserve for a full list of confirmed supported and unsupported data types.
9. **Add Columns to the Find & Reserve model**
   Click a column name and click the right arrow to put them in your Find & Reserve Data Model. Review the following considerations:
   – If a column's name is grey, this can mean one of the following:
     a. The column's data source is not supported
     b. The column's table has no primary key relationships to other tables. In this case, the names of all the columns in this table are grey.

     If you try to add a column name in grey, the operation fails with an appropriate error message.
   – When you add a column from a table that has one or more relationships to other tables in your Find & Reserve model, the **Relationships** dialog opens. Here you can choose which foreign key relationship to use to connect this table to other tables in your Find & Reserve model.
     • Click **Edit Relationships** to add relationships to columns in other tables in your Find & Reserve model.

       NOTE

       Only columns of the same data type as the target column are shown in the dropdown list of columns.
     • When you click **Save Relationships**, the new relationship appears in the list of relationships from which you can choose this table's foreign key relationship to the Find & Reserve model.
     • Click **Done** to close the **Relationships** dialog.
   – When you add a column from a table that has no relationships to other tables in your Find & Reserve model, an error dialog opens with the text 'No Relationships found to connect Table <table_name> with Find & Reserve

model'. Click **Define Relationships** to open the **Relationships** dialog and add relationship(s) for that table, from which you can choose when you click **Save Relationships**.
Click **Done** to close the **Relationships** dialog.

- If you have chosen Data Prefetch:Off, follow the prompt to create a Reservation Table.

10. **Choose Model Key(s).**
    You must have at least one Model Key column in your Find & Reserve model.
    Click the checkbox in the **Model Key** column to make a column a Model Key. All Model Keys must come from the same table. If you try to add a Model Key from a column other than the column that currently contains Model Key(s), the **Change Model Key(s)** dialog opens. Click **Remove Current Key(s)** to change to the new Model Key.

    WARNING

    You cannot make changes to the Model Keys after you create the Find & Reserve Model.
11. (Optional) **Edit table relationships in your Find & Reserve model.**
    You can click the pencil icon for any table in the Find & Reserve model that does not contain Model Key(s). The **Relationships** dialog opens. Here you can choose which foreign key relationship to use to connect this table to other tables in your Find & Reserve model.
    - Click **Edit Relationships** to add relationships to columns in other tables in your Find & Reserve model.
    - When you click **Save Relationships**, the new relationship appears in the list of relationships from which you can choose this table's foreign key relationship to the Find & Reserve model.
    - Click **Done** to close the **Relationships** dialog.
12. (Optional) **Edit columns in your Find & Reserve model.**
    If you click the pencil icon for any column in the Find & Reserve model, the **Edit F&R Model Column** dialog opens. Here you can:
    - Change the column alias (the column's name in the Data Model does not change).
    - Make the column **Available as Search Criteria**. This option is checked by default.
    - Select whether values in this column **Display as Dropdown Menu**. This option is checked by default.
13. After you add all the columns you want in your Find & Reserve Model, click **Finish**.
    A progress wheel appears. When the process is complete, the **Models for Find & Reserve** page opens. Your new Find & Reserve Model is visible on this list, with the Data Prefetch option **On Demand**.
    The message "Success! Test Data Model '<model_name>' created successfully" appears at the top of the page.

Create a Legacy Find & Reserve Model

This is the process to create a standard Find & Reserve Model.

Follow these steps:

1. Access the CA TDM Portal.
2. Select the required project and version from the **Project** drop-down list.
3. Expand **Modeling** in the left pane.
4. Click **Find & Reserve**.
   The **Models for Find & Reserve** page opens.
   Note: All test data models that you create for the selected project and version are listed in this page.
5. Click **Create Legacy Model**.
6. Enter the following information:
   - **Name**
     Specifies an appropriate name for the test data model.
   - **(Optional) Description**
     Specifies an appropriate description for the test data model.
   - Include Reserved Data
     If checked, testers can decide to include data that is already reserved, in their search results.

WARNING

Testers should be aware, that data previously reserved by other testers must not be used for testing.

7. Click **Next**.
   The **Environment** page opens. This page lists the existing environments that are created for the selected project version. If no environment is defined, use the **New Environment** button to open the environment creation dialog and specify the appropriate information. You can also edit or delete an existing environment from the list.
   For more information about how to create an environment, see Create an Environment.
8. Select the applicable environment.
9. You can activate the **Data Prefetch** toggle. Query time can be reduced by prefetching the data to a TDM database. When Data Prefetch is active, data from all your data sources across all your environments will be cached in a TDM database. Once this data is cached, the query will run on the cached data. When Data Prefetch is disabled, all cached data is deleted.

   TIP

   For smaller data sets with many entities and relationships between them, we recommend the use of Data Prefetch.

   For more information, see Data Prefetch.
10. Click **Next**.
    The **Build Test Data Model: Select Model Key** page opens. You use this page to define a model key for the test data model you are creating. A model key acts like a primary key for a test data model. The left panel in this page shows all the entities (for example, tables) and data elements (for example, columns) that are already registered with the selected project and version.

    NOTE

    While creating a test data model if you click Cancel after defining the model key, the Portal displays a message that prompts you to specify whether you want to save the test data model or delete it. Select the appropriate action.
11. Locate and expand the appropriate entity. Use **Search** to search for a specific entity; you cannot search for a data element of an entity. You can also navigate through the paginated list to find the appropriate entity or data element under the entity.
12. Select the data element that you want to use as a model key for this test data model.
13. Click the forward arrow (>).
    The **Add <*EntityName*>** dialog appears.
    The Portal prompts you to first add the parent entity of the data element (model key). After you add the parent entity details, the Portal allows you to add the data element (model key) information. The added entity acts as a root entity for the test data model; similarly, the added data element acts as a model key for the test data model.
14. Enter the entity information as follows:
    - **Name**
      Lets you enter an appropriate logical name for the entity that contains the data element (model key).
    - **Data Source**
      Displays the data source that contains the entity. The Portal automatically searches all the data sources (connection profiles) for the selected environment and populates the field with the appropriate data source that includes the entity. If the search does not find any data source that includes the entity in the given environment, the Portal displays an error stating that no related data source is available. If the entity is found in multiple data sources, the Portal automatically displays the first data source in the list; however, you must review and then select the data source for which the entity has to be associated.

      TIP

      We recommend that you verify your environment, add the appropriate data sources, and map them to the related connection profiles.

15. Click **Next**.
    The **Add <*EntityName.DataElementName*>** dialog appears.
16. Enter the data element (model key) information as follows:
    - **Name**
      Lets you enter an appropriate logical name for the data element that you are using as a model key.
    - **Display in Tester Self Service**
      Lets you specify whether you want to display this data element to testers in the Self-Service Catalog. Select this option to display the data element. The logical name that you specify in **Name** is displayed to the tester.
    - **Use drop-down filter**
      Lets you choose whether or not testers can select values for this element from a drop-down list of all available values. The field auto-suggests values based on the tester's input.

      NOTE

      Drop-down filter functionality is only available for character string values (not Date, Time or Numeric).
17. Click **Save**.
    The model key (data element) and its related parent entity are added to the right pane.
    After you add the model key, you can proceed to add other data elements to the test data model. If the parent entity of the data element that you are adding is not already present in the test data model (right pane), the Portal prompts you to first add the parent entity details. The Portal also expects you to define the required associations (relationships) while adding the entity to the test data model. Each entity that you add to the test data model must have a direct or indirect (through intermediate entities) association with the root entity already added to the test data model. If the parent entity is already added to the test data model, the Portal skips the entity details dialog and directly displays the data element details dialog.
18. Click **Next**.
    The **Build Test Data Model: Select Data Elements** page opens.
19. Select the required data element in the left pane and click the forward arrow.
    The **Add <*EntityName*>** dialog appears if the parent entity is not already added to the test data model. Otherwise, the **Add <*EntityName.DataElementName*>** dialog appears.
20. Enter a logical name for the entity in **Name**, select the related data source from the **Data Source** drop-down list, and specify the association information as follows:
    - **Association From**
      Lets you select the appropriate entity (source) with which you want to establish the association. This source entity could be a root entity or any other entity that is related to the root entity and is already added to the test data model (right pane). This drop-down list displays the same logical names that you specify while entering the entity information in the previous steps.
    - **Association Type**
      Lets you select the association type based on your requirement. Applicable types are ONE_ONE, ONE_MANY, and MANY_ONE.
      Specifying appropriate association type helps improve the performance of the find operation on the test data model. The search of the data is optimized based on the association types among the entities. Therefore, ensure that you take appropriate care while specifying the association type.
    - **Join Fields**
      Lets you select a data element from the entity (source) already added to the test data model and join it with a data element from the entity (target) that you are adding to the test data model. These data elements help establish the association between the two entities. The data type of both the data elements must be the same. If the two entities are from the same data source and an association exists between the two entities, the Portal automatically displays the related data elements in the **Join Fields** area. If the entities are from different data sources, you need to manually select the appropriate data elements to establish the association.
      For composite keys, click the plus icon (+) to add all the required data elements to establish the complete association.
21. Click **Next**.

The **Add <*EntityName.DataElementName*>** dialog appears.

NOTE

If the source entity is part of multiple associations, you must select the association that you want to use for the data element.

22. Enter the data element information as explained in Step 15 and click **Save**.
    The intended data element and its related entity (if not already added) are added to the test data model.
23. Repeat Steps 17 through Step 20 to add other entities to the test data model.
    All the items are added to the right pane. You can browse through the paginated list to review the entities and data elements added to the test data model.
24. Click **Finish**.
    The test data model is created and is added to the **Models for Find & Reserve** page.

NOTE

To delete a test data model, identify the row that contains the test data model, click the Delete icon (cross), and confirm the deletion.

You have successfully created a test data model for testers.

You can find an example of this process here: Example: Order Management System.

Edit a Find & Reserve Model

If you want to update your test data model after you create it, the Portal allows you to do so. The procedure differs slightly for:

- Edit standard Find & Reserve model
- Edit Legacy Find & Reserve model.

You can update the following information about a test data model:

- Name and description of the test data model.

  NOTE

  We recommend that you always base your automation scripts (if any) on the test data model ID instead of the test data model name. The test data model ID remains unique across all the projects, and it cannot be changed once it is created. This helps ensure that your scripts remain in the working condition even when the name of the test data model is changed.
- Add a data element and entity to the test data model.

  NOTE

  You cannot add a new root entity or model key. You cannot add a new root entity or model key.
- Remove a data element and entity from the test data model.

  NOTE

  You cannot remove an already added root entity or model key.

The process to edit a Find & Reserve Model is different for standard models and Legacy models.

Edit a standard Find & Reserve Model

This describes the process to edit a Find & Reserve Model.

Follow these steps:

1. Access the CA TDM Portal.

2. Select the required project and version from the **Project** drop-down list.
3. Expand **Modeling** in the left pane.
4. Click **Find & Reserve**.
   The **Models for Find & Reserve** page opens. All test data models that you create for the selected project and version are listed in this page. On this page you have the following options:
   - **Display to Testers**
   - **Show Reserved Rows**
   - **Data Prefetch** dropdown
     From this menu you can choose the Data Prefetch behaviour for this Find & Reserve Model. The options are:
     - **Periodic**
       TDM prefetches data in accordance with Synchronization Triggers.
     - **On Demand**
       TDM only prefetches data when you click **Fetch Data** from the **Actions** menu
     - **Off**
       TDM queries your source database directly, without prefetching a copy of the data into the TDM database. You need to create a Reservation Table.
   - **Actions**
     - **Remove model**
     - **Fetch data**
       Only available if Data Prefetch option is not **Off.**
5. Click the test data model that you want to update.
   The **Test Data Model** page opens.
6. Update the following test data model information, if required:
   - Name of the test data model.
   - Description of the test data model.
7. Click **Next**.
   The **Find & Reserve Model** page opens. On this page, you can add more columns to your **Find & Reserve Test Data Model** (right pane) from your **Data Model** (left pane), and remove columns from your Data Model.

   WARNING

   After you create a model, you cannot change which column(s) is/are the model keys, and you cannot remove the model key column.

   If you type a text string into the **Search** field above the left-hand pane, only tables whose names match that string are displayed.
8. **Add Columns to the Find & Reserve model**
   Click a column name and click the right arrow to put them in your Find & Reserve Data Model.
   - If a column's name is grey, this can mean one of the following:
     a. The column's data source is not supported
     b. The column's table has no primary key relationships to other tables. In this case, the names of all the columns in this table are grey.

     If you try to add a column name in grey, the operation fails with an appropriate error message.
   - When you add a column from a table that has one or more relationships to other tables in your Find & Reserve model, the **Relationships** dialog opens. Here you can choose which foreign key relationship to use to connect this table to other tables in your Find & Reserve model.
     - Click **Edit Relationships** to add relationships to columns in other tables in your Find & Reserve model.

NOTE

Only columns of the same data type as the target column are shown in the dropdown list of columns.

- When you click **Save Relationships**, the new relationship appears in the list of relationships from which you can choose this table's foreign key relationship to the Find & Reserve model.
- Click **Done** to close the **Relationships** dialog.

– When you add a column from a table that has no relationships to other tables in your Find & Reserve model, an error dialog opens with the text 'No Relationships found to connect Table <table_name> with Find & Reserve model'. Click **Define Relationships** to open the **Relationships** dialog and add relationship(s) for that table, from which you can choose when you click **Save Relationships**.
Click **Done** to close the **Relationships** dialog.

9. (Optional) **Edit table relationships in your Find & Reserve model.**
You can click the pencil icon for any table in the Find & Reserve model that does not contain Model Key(s). The **Relationships** dialog opens. Here you can choose which foreign key relationship to use to connect this table to other tables in your Find & Reserve model.
   – Click **Edit Relationships** to add relationships to columns in other tables in your Find & Reserve model.
   – When you click **Save Relationships**, the new relationship appears in the list of relationships from which you can choose this table's foreign key relationship to the Find & Reserve model.
   – Click **Done** to close the **Relationships** dialog.
10. (Optional) **Edit columns in your Find & Reserve model.**
If you click the pencil icon for any column in the Find & Reserve model, the **Edit F&R Model Column** dialog opens. Here you can:
   – Change the column alias (the column's name in the Data Model does not change).
   – Make the column **Available as Search Criteria**. This option is checked by default.
   – Select whether values in this column **Display as Dropdown Menu**. This option is checked by default.
11. Click **Finish** to save your changes and return to the list of Find & Reserve Models.
12. Click **Cancel** to discard your changes and return to the list of Find & Reserve Models. You need to confirm this decision.
13. The **Models for Find & Reserve** page opens. This page now includes the updated Find & Reserve Model.

You have successfully updated a Find & Reserve Model.

Edit a Legacy Find & Reserve Model

This describes the process to edit a Legacy Find & Reserve Model.

Follow these steps:

1. Access the CA TDM Portal.
2. Select the required project and version from the **Project** drop-down list.
3. Expand **Modeling** in the left pane.
4. Click **Find & Reserve**.
The **Models for Find & Reserve** page opens. All test data models that you create for the selected project and version are listed in this page. On this page you have the following options:
   – **Display to Testers**
   – **Show Reserved Rows**
   – **Data Prefetch** dropdown
   From this menu you can choose the Data Prefetch behaviour for this Find & Reserve Model. The options are:
   - **Off**
   CA TDM does not Prefetch data for the model.
   - **Periodic**

CA TDM prefetches data in accordance with Synchronization Triggers.
    - **On Demand**
      CA TDM only prefetches data when you click **Fetch Data** from the **Actions** menu.
  - **Actions**
    - **Remove model**
    - **Fetch data**
      Only available if Data Prefetch option is not **Off**.
5. Click the test data model that you want to update.
   The **Test Data Model** page opens.
6. You can update the following test data model information:
   - Name of the test data model.
   - Description of the test data model.
7. Click **Next**.
8. Select a different environment, if applicable. Ensure that you have considered all the relevant factors while selecting a different environment.
9. Click **Next**.
   The **Build Test Data Model: Select Model Key** dialog opens. You cannot change the model key of a test data model.
10. Click **Next**.
    The **Build Test Data Model: Select Data Elements** dialog opens.
11. Add or remove appropriate data elements and entities from the test data model based on your requirements.
12. (Optional) Edit data elements by clicking the Pencil icon, in the Actions column.
    This lets you change the **Name**, **Display in Tester Self-Service** status, and **Use Drop-down filter** status. Click **Save** if you make any changes.
13. Review the changes.
14. Click **Finish**.
    The **Models for Find & Reserve** page opens. This page now includes the updated Find & Reserve Model.

You have successfully updated a Legacy Find & Reserve Model.

NOTE

More information:

- The Data Model in CA TDM Portal
- End-to-End Scenario for Data Discovery
- Scan Data Model for PII

Data Types Supported by Find & Reserve

This page contains notes and details of specific data types that CA Test Data Manager supports.

Support for Data Types in CA TDM

The following table details which data types we confirm that the Find & Reserve feature supports and does not support, for each database type.

WARNING

These data types do not include all possible data types for each database. Other data types may or may not function with the Find & Reserve feature. You can add these other data types to your Find & Reserve model, but we cannot guarantee their functionality.

Database Type	Supported Data Types	Unsupported Data Types
Oracle	VARCHAR2 NVARCHAR2 CHAR NCHAR DATE TIMESTAMP NUMBER FLOAT DOUBLE PRECISION REAL BINARY_FLOAT BINARY_DOUBLE SMALLINT INTEGER INT NUMERIC DECIMAL RAW	TIMESTAMP WITH TIME ZONE TIMESTAMP WITH LOCAL TIME ZONE LONG RAW BLOB CLOB NCLOB BFILE INTERVAL YEAR TO MONTH INTERVAL DAY TO SECOND ROWID UROWID
MS SQL	UNIQUEIDENTIFIER VARCHAR NVARCHAR CHAR NCHAR TIME DATE SMALLDATETIME DATETIME DATETIME2 MONEY SMALLMONEY INT TINYINT SMALLINT BIGINT FLOAT REAL NUMERIC DECIMAL BIT BINARY VARBINARY	DATETIMEOFFSET VARBINARY(MAX) TEXT NTEXT

DB2	VARCHAR VARGRAPHIC CHARACTER GRAPHIC TIME DATE INTEGER SMALLINT BIGINT DECFLOAT DOUBLE FLOAT REAL NUMERIC DECIMAL CHARACTER FOR BIT DATA VARCHAR FOR BIT DATA BINARY VARBINARY	TIMESTAMP WITHOUT TIME ZONE TIMESTAMP WITH TIME ZONE CLOB DBCLOB BLOB ROWID

Notes on Data Type Support

Different source and target database types - dates

MS SQL and DB2 databases support dates in the range 0001 to 9999. Oracle databases support dates in the range -4712 to 9999.

Therefore, if a source database is Oracle and the target database is MS SQL, CA TDM does not copy DATE and TIMESTAMP values with a Year value before 1AD (i.e. <0001).

Notes on DB2 databases

- **z/OS**
  CA TDM Find & Reserve only supports IBM DB2 11 databases and later, on z/OS.
- **Data type DECFLOAT**
  DB2-only data type DECFLOAT supports numbers in the range -10E+6144 to 10E+6144. Oracle / MS SQL datatypes support numbers in the range -1.79E+308 to 1.79E+308.
  Therefore, CA TDM cannot copy DECFLOAT values from the source (DB2) table outside of the target (Oracle / MS SQL) range.

Enable a Test Data Model for Testers

Testers can access only those test data models that are enabled for them. TDEs can decide whether to display a test data model to testers. If enabled for display, a test data model is displayed as a self-service form to testers in the Self-Service Catalog section of the Portal. Testers can start consuming the test data model (as a form) to find and reserve the data. They prepare the filter criteria by using all the required attributes that a TDE has added to the test data model.

Follow these steps:

1. Access the CA TDM Portal.
2. Select the required project and version from the **Project** drop-down list.
3. Expand **Modeling** in the left pane.
4. Click **Test Data Models**.
   The **Test Data Model** page opens. All created test data models for the selected project and version are listed in this page.

5. Identify the row that includes the test data model you want to display to testers.
6. Toggle the value to **Yes** in the **Display to Tester** column.
   The test data model is enabled and becomes visible to testers as a form in the Self-Service Catalog section of the Portal.

You have successfully enabled a test data model for a tester. For more information about how testers use a test data model (form) to find and reserve the test data, see Tester Self-Service.

Access Data Reservation and Model Details in OrientDB

In the CA TDM Portal, all the data reservation and test data model details are stored in OrientDB. OrientDB is installed when you install the CA TDM Portal. You can query OrientDB in one of the following ways to find the details:

Using the Command Prompt

To use the command prompt to access the data reservation and test data model details in OrientDB, follow these steps:

1. Open the command prompt.
2. Navigate to the following location:
   `C:\Program Files\CA\CA Test Data Manager Portal\orientdb\bin`
   Note: `C:\Program Files\CA\CA Test Data Manager Portal` represents the default location where the CA TDM Portal is installed.
3. Open the console as follows:
   `console`
4. Connect to the host as follows:
   `connect remote:<host> root <password>`
5. List the databases as follows:
   `list databases`
6. Connect to OrientDB as follows:
   `connect remote:<host>/ReservationDB root <password>`
7. List all the classes as follows:
   `classes`
8. Browse the contents of a class as follows:
   `browse class DataModel`
9. Run the select statement for a test data model as follows:
   `select * from DataModel`
10. Run the select statement for a data reservation as follows:
    `select * from Reservation`

For more information about other SQL commands, see http://orientdb.com/docs/2.0/orientdb.wiki/Tutorial-SQL.html.

Using the Web Interface

To use the web interface to access the data reservation and test data model details in OrientDB, follow these steps:

1. Access the following URL:
   `http://<hostname>:2480/studio/index.html`
   <hostname> represents the computer where the CA TDM Portal is installed.
2. Click **Schema**.
3. Click **DataModel** or **Reservation** as required.
4. Click **Query All**.
   All relevant details are listed.

Example: Order Management System

This article includes an example scenario that helps you achieve the following objectives:

- How Test Data Engineers (TDEs) can create test data models and can share them with testers as forms.
- How testers can access and use the applicable forms that are shared with them to find and reserve the test data.

> **NOTE**
>
> The example in this article uses the Northwind sample database that is available for Microsoft SQL Server. Refer to the Microsoft website to download the Northwind sample database.

You can find a more detailed guide to the Find & Reserve Test Data Model creation process here: Create and Edit a Find & Reserve Test Data Model.

The page covers the following topics:

Scenario

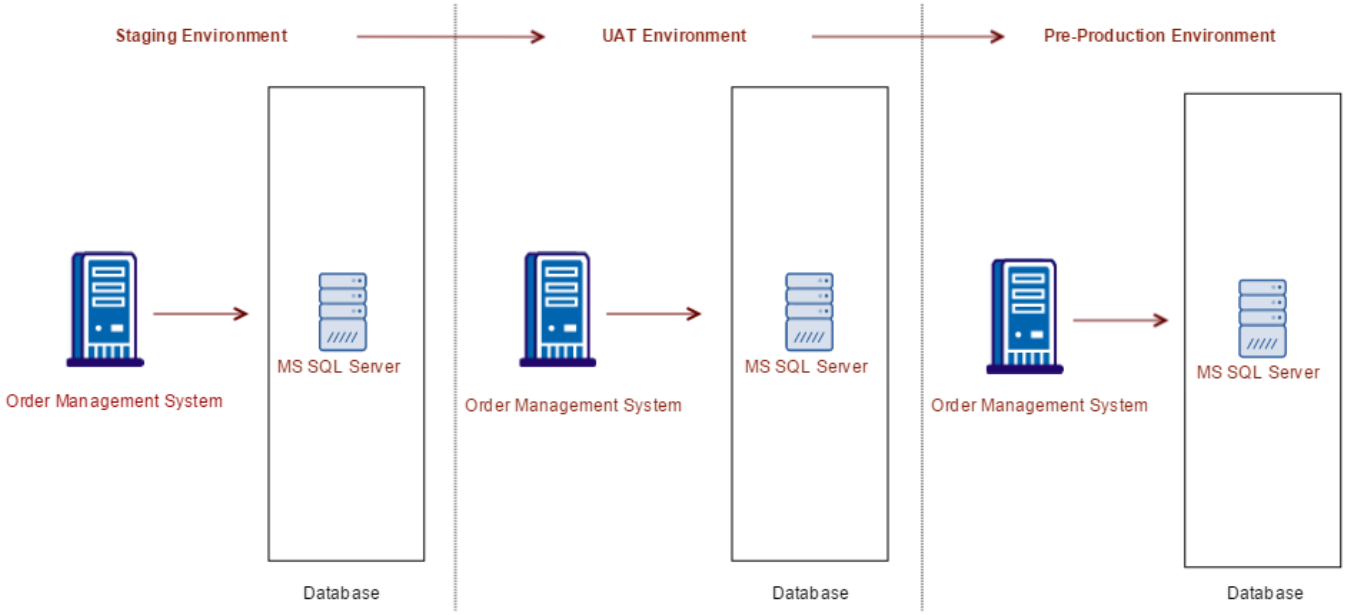
Peter and Joe work at the same company. Peter is a **TDE**, and Joe is a **tester**.

Joe's current assignment is to test an Order Management application. The data for the Order Management application is spread across the following tables in the Microsoft SQL Server database:

- Orders
- Shippers
- Customers
- Categories
- Order Details
- Region
- Territories
- Suppliers
- Products

The Order Management application uses three environments: Staging, UAT, and Pre-Production. Each environment includes a separate instance of the Microsoft SQL Server database, with different data in each instance. The following illustration shows the high-level architecture:

Figure 38: Order Management System Model-Based Tester Self-Service



Joe (tester) needs on-demand access to the data so that he can find the right test data and reserve it. He wants to have complete control over the filter criteria that he wants to specify to find the data. He also wants to review the data that is retrieved based on his criteria. And, finally, he wants to reserve the data if he is satisfied with the result. For example, Joe wants the fields that give him the flexibility of finding the following type of data:

- Find all the orders based on the shipping city.
- Find all the orders based on the shipping postal code.
- Find all the orders based on the shipping region
- Find all the orders based on the unit price.
- Find all the orders based on the discount.
- Find all the orders based on the quantity.
- Find the order information based on the order ID.

Peter (TDE) helps Joe perform all these tasks. Peter creates a Find & Reserve Test Data Model named Orders. Peter aggregates the required data elements from the database (for the selected environment) into the Orders Find & Reserve Test Data Model. Peter also ensures that he includes and exposes data elements in the Find & Reserve Test Data Model that are relevant to the business requirement of Joe. Joe can then use the exposed data elements to specify the data criteria, find the relevant test data, and reserve it.

Prerequisites

As a TDE, Peter needs to do the following:

- Create the appropriate project (Orders) and version (1.0) in the CA TDM Portal.
  For more information about how to create a project and version, see Create and Edit Projects.
- Create the connection profile (Order_SQLServer) to connect to the Microsoft SQL Server data source.
  For more information about how to create a connection profile, see Create and edit Connection Profiles.
- Share the connection profile (Order_SQLServer) with Joe (tester).

High-Level Process

Peter (TDE) performs the following TDE tasks:

1. Create three environments—Staging, UAT, and Pre-Production.
2. Create the Orders test data model by following these steps:
   a. Select the Staging environment.
   b. Specify the model key for the test data model.
   c. Add different data elements to the test data model.
   d. Specify associations between entities while adding data elements to the test data model.
   e. Save the Orders test data model.
3. Enable the Orders test data model for testers.

Joe (tester) performs the following tester tasks:

1. Access the Self-Service Catalog section.
2. Identify the Orders form.
3. Select the environment.
4. Specify the filter requirements for the data by using the available fields in the form.
5. Review the retrieved test data.
6. Reserve the test data if the test data meets the requirements.

Create Environments

Peter (TDE) creates three environments—Staging, UAT, and Pre-Production. Each environment contains a separate instance of the Microsoft SQL Server database. Each instance is available on a different server with different data.

NOTE

For more information about environments, see Configure Dynamic Test Data Reservation Service.

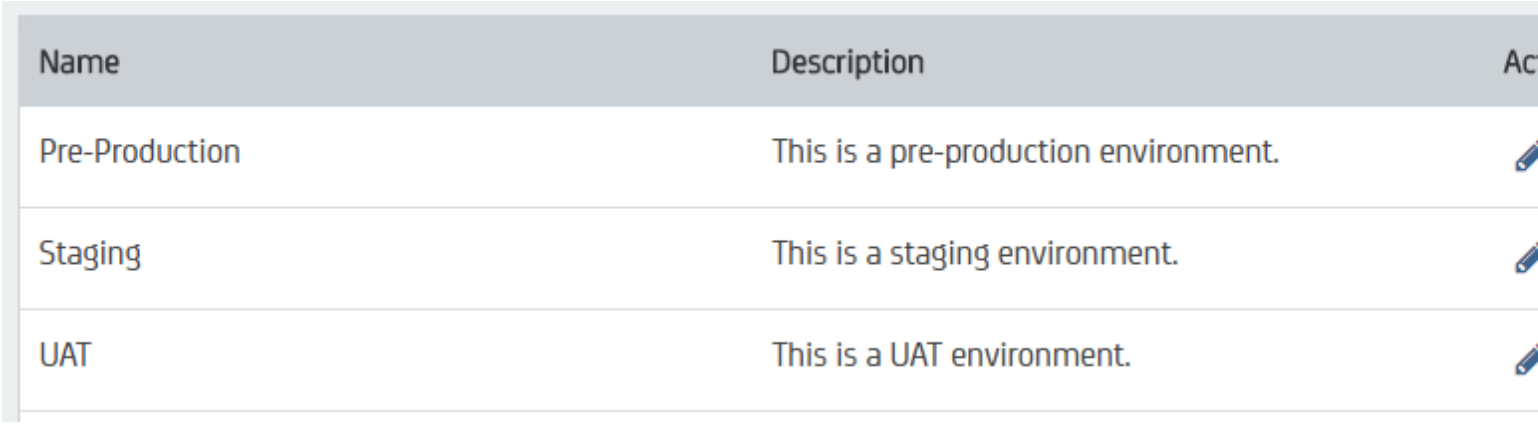
Follow these steps:

1. Access the CA TDM Portal as a TDE.
2. Select the **Order** project and version **1.0** from the **Project** drop-down list.
3. Expand **Modeling** in the left pane.
4. Click **Environments**.
5. Click **New Environments**.
6. Enter the name and description information as follows:
   – **Name:** Staging
   – **Description:** This is a staging environment.
7. Map the data source with the corresponding connection profile as follows:
   – **Data Source Name:** SQLServer, **Connection Profile:** Order_SQLServer

   NOTE

   The data source names remain the same for the other two environments. That is, once you save the data source name for the first environment, you cannot change it for other environments in the same project version.
8. Click **Save**.
   The Staging environment is added to the **Environments** page.

Repeat the same steps to create the UAT and Pre-Production environments. The final list on the **Environments** page looks like the following screen shot:



Name	Description	Ac
Pre-Production	This is a pre-production environment.	
Staging	This is a staging environment.	
UAT	This is a UAT environment.	

Create the Orders Test Data Model

After Peter (TDE) creates the required environments, he proceeds to create the Orders test data model using the Staging environment. He follows the same steps to create test data models using other environments, as required.

To meet the data requirement of Joe, Peter needs to add specific data elements to the test data model. Peter uses three entities—Orders, Order Details, and Products—from the Microsoft SQL Server database to add the appropriate data elements. From the Orders entity, Peter adds the following items:

- OrderID as a model key
- ShipCity
- ShipPortalCode
- ShipRegion

From the Order Details entity, Peter adds the following items:

- Discount
- Quantity

From the Products entity, Peter adds the following item:

- UnitPrice

NOTE

For more information about test data models, see Configure Dynamic Test Data Reservation Service.

Follow these steps:

1. Access the CA TDM Portal as a TDE.
2. Select the **Order** project and version **1.0** from the **Project** drop-down list.
3. Expand **Modeling** in the left pane.
4. Click **Find & Reserve**.
5. Click **New Test Data Model**.
6. Specify the name and description as follows, and click **Next**:
   - **Name:** Orders
   - **Description:** This test data model is for the Order Management application.
7. Select the environment as Staging from the list.
   The **Build Test Data Model: Select Model Key** page opens.
8. Add the model key (OrderID) to the Orders test data model as follows:
   a. Expand the Orders entity, select OrderID as your model key for this test data model, and click the forward arrow to add it to the test data model.

The **Add Orders** dialog opens. This dialog lets you add the parent root entity (Orders) of the OrderID data element, because the parent entity Orders is not already added to the test data model. This dialog does not appear again if you try to add another data element from the already added entity (Orders in this case).

b. Verify the entity name as Orders and the data source value as SQLServer.
c. Click **Next**.
The **Add Orders.OrderID** dialog opens. This dialog lets you add the model key.
d. Verify the model key name as OrderID.
e. Verify that the **Display in Tester Self Service** option is selected, because you want to display this data element to testers.
f. Click **Save**.
The Orders entity and OrderID model key are added to the test data model.
g. Click **Next**.
The **Build Test Data Model: Select Data Elements** page opens.

9. Add other required data elements from the Orders entity to the Orders test data model as follows:
ShipCity
a. Select ShipCity and click the forward arrow.
The **Add Orders.ShipCity** dialog opens, because the related parent entity Orders is already added to the test data model.
b. Enter the name as Ship_City.
c. Select the display type as Text Box.
d. Verify that the display option is selected.
e. Click **Save**.
The ShipCity data element is added under the Orders entity with the display name as Ship_City.
ShipPostalCode
a. Select ShipPostalCode and click the forward arrow.
The **Add Orders.ShipPostalCode** dialog opens, because the related parent entity Orders is already added to the test data model.
b. Enter the name as Ship_Postal_Code.
c. Select the display type as Text Box.
d. Verify that the display option is selected.
e. Click **Save**.
The ShipPostalCode data element is added under the Orders entity with the display name as Ship_Postal_Code.
ShipRegion
a. Select ShipRegion and click the forward arrow.
The **Add Orders.ShipRegion** dialog opens, because the related parent entity Orders is already added to the test data model.
b. Enter the name as Ship_Region.
c. Select the display type as Text Box.
d. Verify that the display option is selected.
e. Click **Save**.
The ShipRegion data element is added under the Orders entity with the display name as Ship_Region.

10. Add other required data elements from the Order Details entity to the Orders test data model as follows:
Quantity
a. Select the Quantity data element from the Order Details entity and click the forward arrow to add.
The **Add Order Details** dialog opens in this case, because the parent entity (Order Details) for the Quantity data element is not already added to the test data model.
b. Verify the entity name as Order Details and the data source value as SQLServer.
c. Select Orders from the **Association From** drop-down list and One_Many from the **Association Type** drop-down list.

d. Select OrderID from the **Orders** drop-down list and OrderID from the **Order Details** drop-down list.
e. Click Next.
The **Add Order Details.Quantity** dialog opens.
f. Verify the data element name as Quantity.
g. Verify that the **Display in Tester Self Service** option is selected.
h. Click **Save**.
The Quantity data element and its related parent entity Order Details are added to the test data model.

Discount

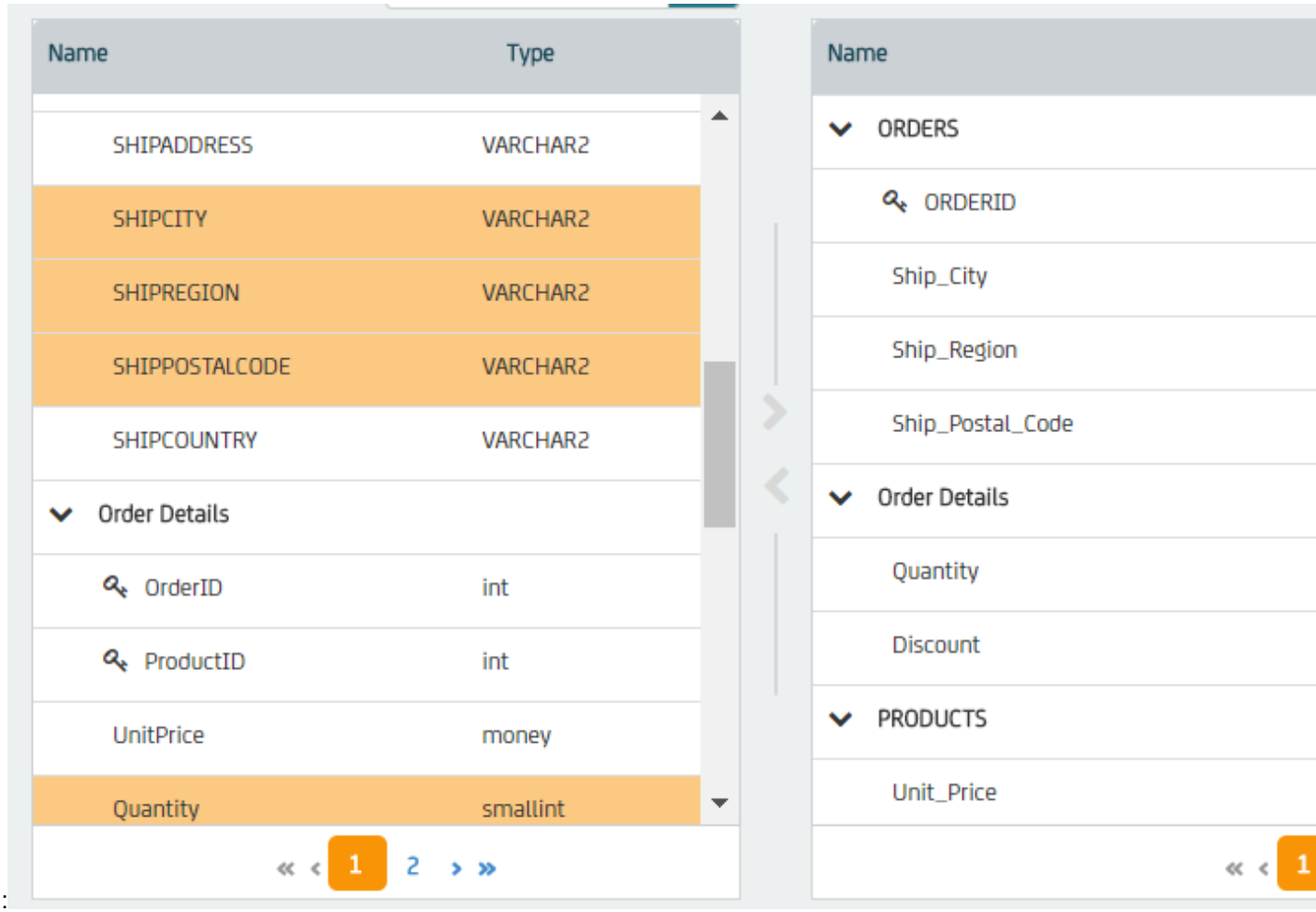
a. Select Discount and click the forward arrow.
The **Add Order Details.Discount** dialog opens, because the related parent entity Order Details is already added to the test data model.
b. Verify the name as Discount.
c. Verify that the display option is selected.
d. Click **Save**.
The Discount data element is added under the Order Details entity with the display name as Discount.

11. Add other required data element from the Products entity to the Orders test data model as follows:

Unit Price

a. Select the UnitPrice data element from the Products entity and click the forward arrow.
The **Add Products** dialog opens in this case, because the parent entity (Products) for the Unit Price data element is not already added to the test data model.
b. Verify the entity name as Products and the data source value as SQLServer.
c. Select Order Details from the **Association From** drop-down list and One_Many from the **Association Type** drop-down list.
d. Select ProductID from the **Order Details** drop-down list and ProductID from the **Products** drop-down list.
e. Click Next.
The **Add Product.UnitPrice** dialog opens.
f. Enter the data element display name as Unit_Price.
g. Verify that the **Display in Tester Self Service** option is selected.
h. Click **Save**.
The Unitprice data element (with the display name as Unit_Price) and its related parent entity Product are added to the test data model.

12. Review the final structure of the Orders test data model. The following screen shot shows the Orders test data model that Peter has created:



13. Click **Finish**.
    The Orders test data model is created and is added to the **Test Data Model** page.

Enable the Orders Test Data Model

Peter (TDE) wants to display the Orders test data model to Joe so that Joe can use it and can get started with the process of finding and reserving the data.

Follow these steps:

1. Access the CA TDM Portal as a TDE.
2. Select the **Order** project and version **1.0** from the **Project** drop-down list.
3. Expand **Modeling** in the left pane.
4. Click **Test Data Models**.
   The **Test Data Model** page opens.
5. Identify the row that includes the Orders test data model.
6. Locate the **Display to Tester** column for the identified row and switch the value to **Yes**.
   The Orders test data model now becomes available in the Self-Service Catalog section.

Find and Reserve the Test Data Using Orders

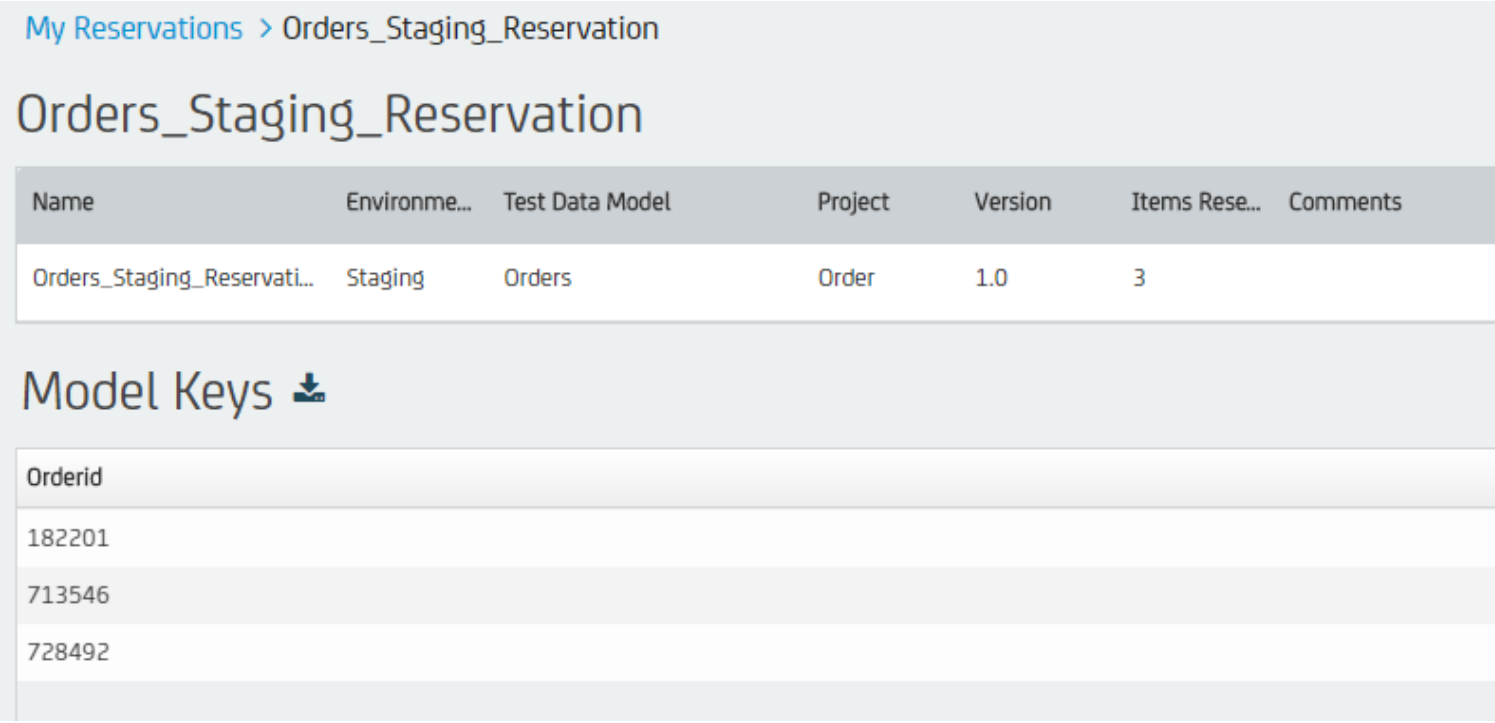
To access the Orders test data model as a form, Joe (tester) accesses the Self-Service Catalog section of the Portal. Joe then uses the exposed data elements to define the test data requirements, reviews the received data, and reserves the data.

NOTE

For more information about finding and reserving the test data, see the tester self-service section.

Follow these steps:

1. Access the CA TDM Portal as a tester.
2. Select the **Order** project and version **1.0** from the **Project** drop-down list.
3. Click **Self Service Catalog** in the left pane.
   The **Self Service Catalog** page opens.
4. Locate the **Orders** catalog.
5. Click the **New Request** button.
   The **Orders** page opens.
6. Select Staging as an environment from the **Environment** drop-down list.
7. Enter the test data filter criteria in the available fields as required. See the find data example screen shots to understand the data filter criteria that Joe has used.
8. Click the **Find Data** button.
   The Portal uses the defined filter criteria, retrieves the data from the applicable data source, and displays the retrieved data.
   Note: Currently, only those data elements from the root entity that are added to the test data model are displayed in the Portal.
9. Review the displayed data and click the tick mark to select the records that you want to reserve. In this case, Joe reserves the three rows that he has received by specifying the filter criteria as Discount (50), Quantity (2), and Ship_Region (TX). The rows with the OrderID values 182201, 713546, and 728492 are selected
   The **Add to Reservation** button is enabled.
10. Click the **Add to Reservation** button.
    All selected records are added to the cart. The cart icon (basket) at the top of the table is enabled and displays the number of records that are added to it.
11. Click the **Complete Reservation** button or the cart icon.The **Items added to Reservation** dialog opens.
12. Enter Orders_Staging_Reservation in **Reservation Name**.
13. Review the records added to the cart and click **Reserve**.A message states that the Orders_Staging_Reservation reservation request is submitted successfully.
14. Click the **Orders_Staging_Reservation** link in the message.
    The **My Reservations** page opens. This page lists all the data reservations.
15. Identify and click the **Orders_Staging_Reservation** reservation.
    The **Orders_Staging_Reservation** page opens. This page displays relevant information about the reservation. For example, environment name, test data model name, project name, version name, number of records reserved, status of the reservation. The page also displays the model keys of the records that have been reserved. The following screen shot shows the **Orders_Staging_Reservation** page:



16. Review the reservation status as **Success**.
17. (Optional) To download the reserved model keys as a CSV file, click the Download Model Keys as CSV icon (down arrow) next to **Model Keys**, specify the required details, and save the file.
18. Open the file to review that all the reserved model keys are available in the file.

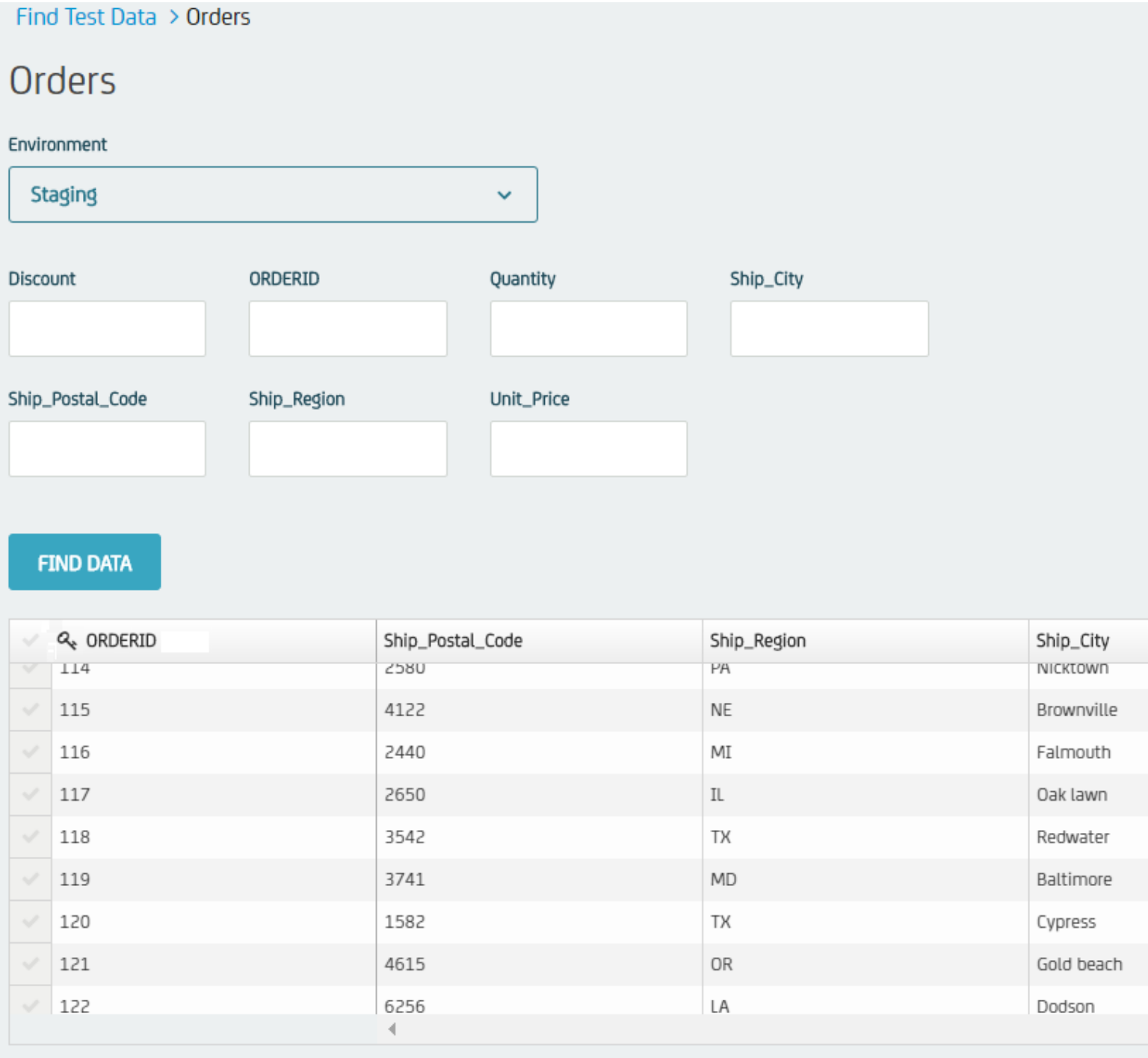
Joe has successfully retrieved, analyzed, and reserved the test data.

Example Screen Shots—Find Data

The screen shots in this section show some examples of the test data that Joe tries to find. Depending on his requirement, he can enter the filter criteria in the available fields and can get the required data.

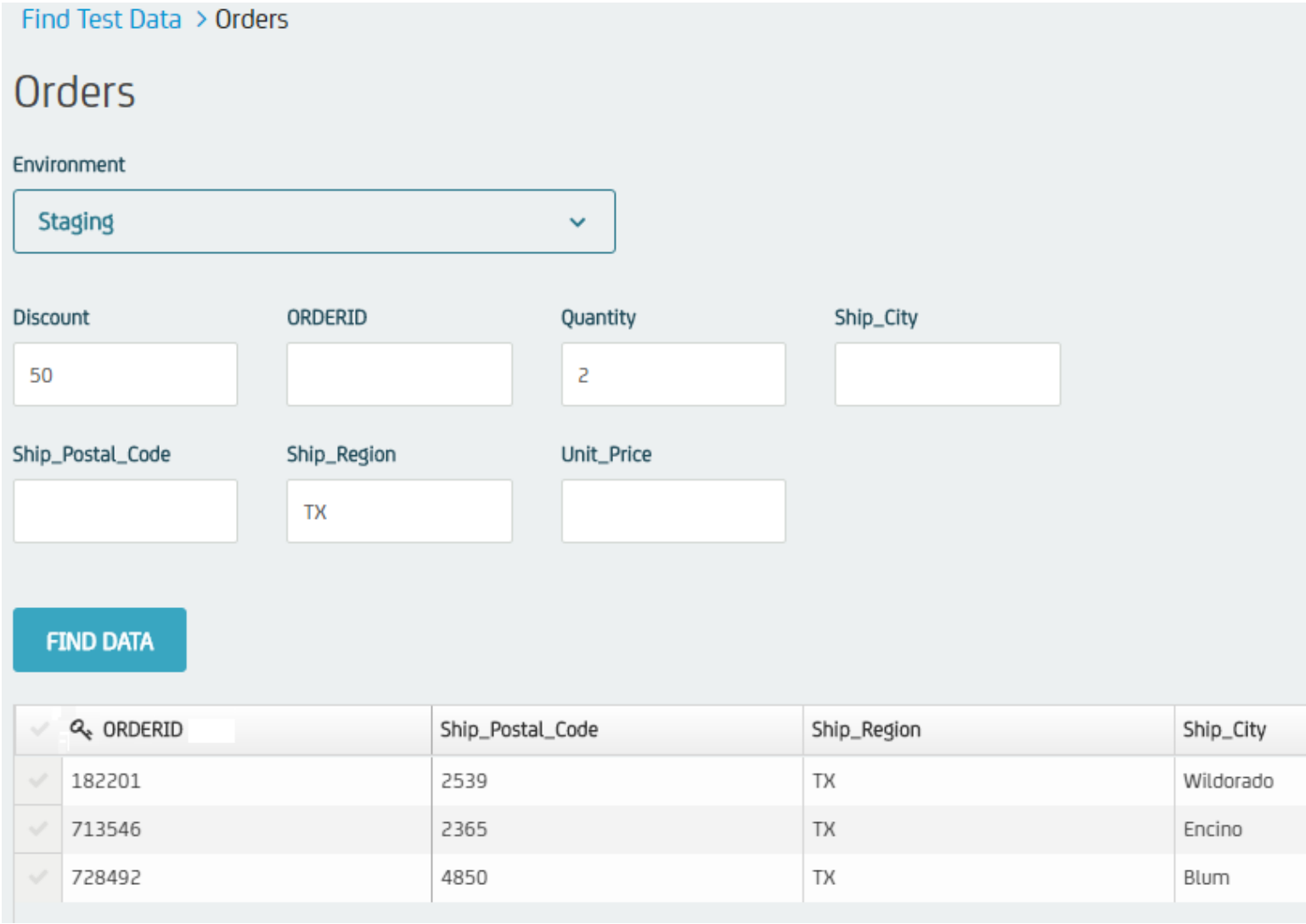
Find all the test data

In this case, Joe does not provide any criteria to filter the data. He leaves all the fields empty. Joe receives all the test data that is available in the data source for the selected environment.

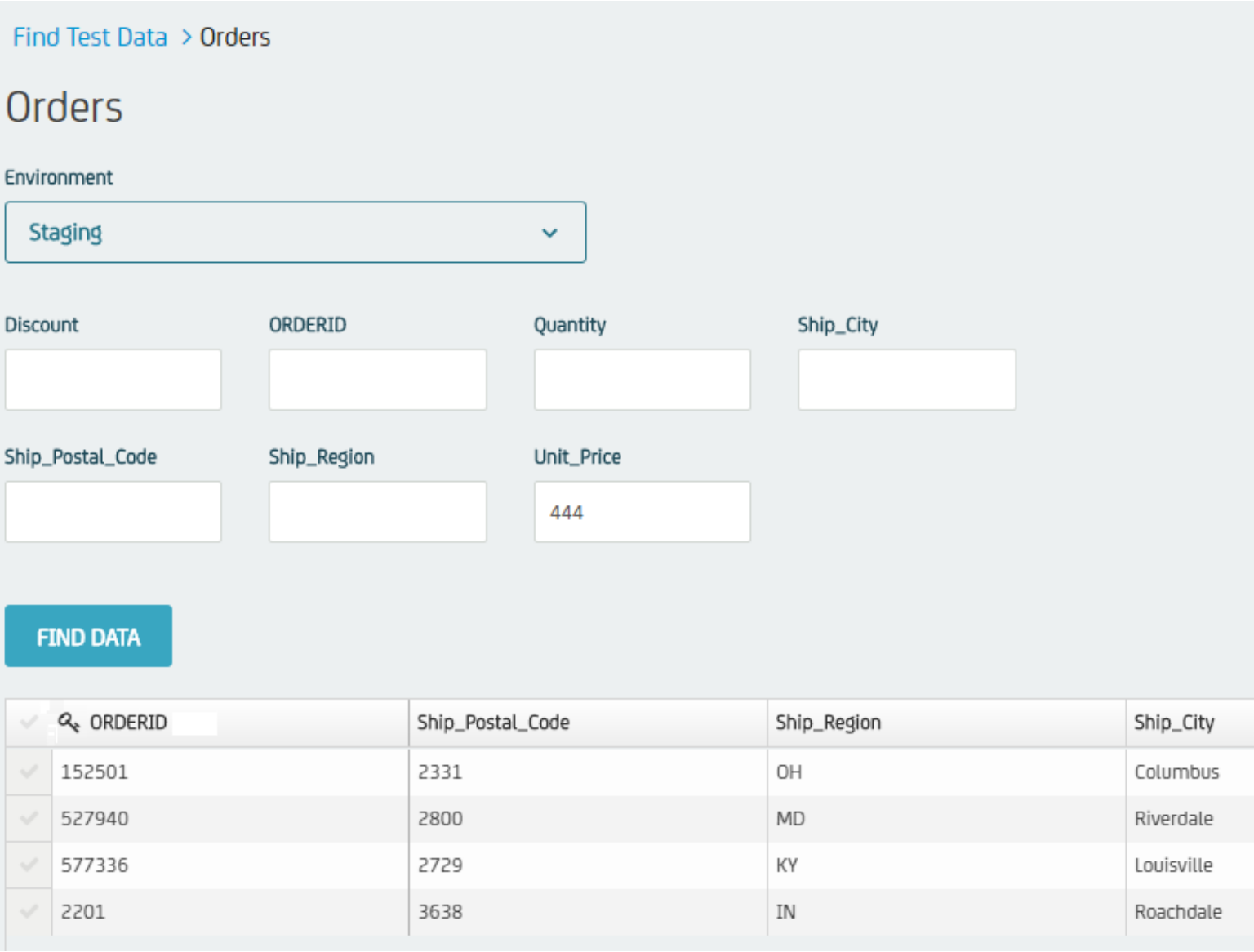


Find the data based on some conditions

In this case, Joe wants to find orders that include the discount value as 50, quantity as 2, and the shipping region as TX. Joe gets three records that meet the filter criteria:



The following screen shot shows the data that is retrieved when Joe specifies the unit price value as 444:



Performance Metrics (Dynamic Test Data Reservation Service)

This section provides performance metrics for Dynamic (Model-Based) Test?Data Reservation Service.

Schema Distribution

The data is spread across three different databases: Oracle, Microsoft SQL Server, and DB2/AS400. The schema distribution is as follows:

Oracle

- Orders (1 million records)
- Shippers
- Customers

Microsoft SQL Server

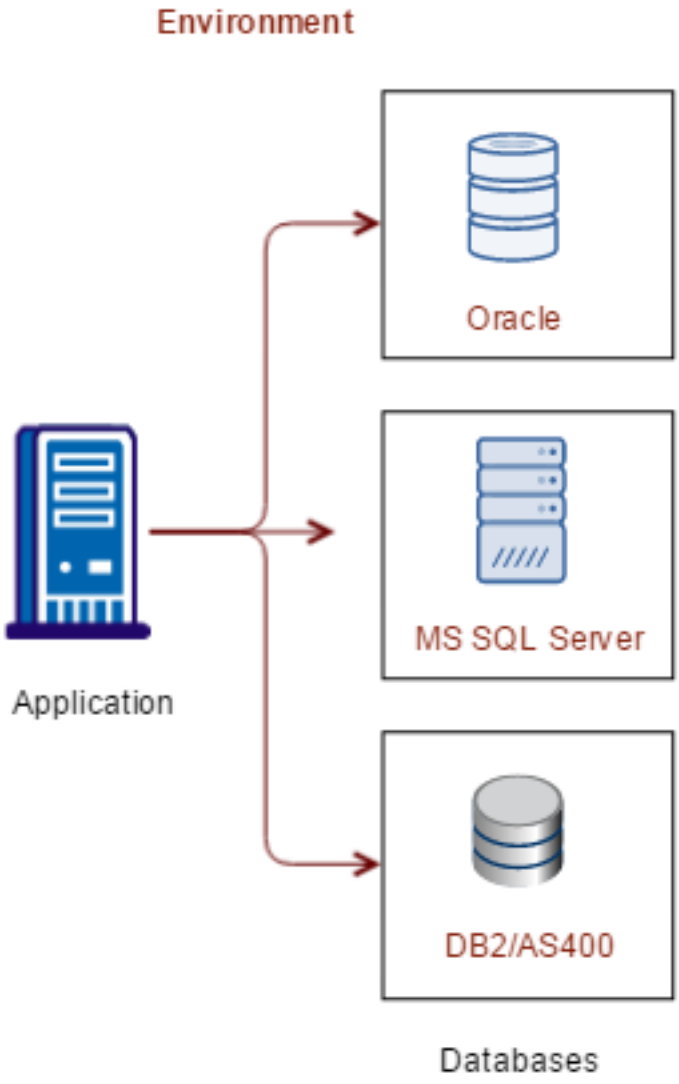
- Categories
- Employees
- EmployeeTerritories
- Order Details (10,000 records)
- Region
- Suppliers (10,000 records)
- Territories

DB2/AS400

- Products (10,000 records)
- CustomerCustomerDemo

The databases are distributed across different geographies with no optimization done on the databases. The following illustration shows how the application interacts with the databases.

Figure 39: Performance_Metrics



NOTE

Performance result values are only indicative. Actual performance is dependent on several factors; for example, server configuration, network configuration, database configuration, and schema size.

Specifications

The following are the specifications:

- Virtual machine (VM) on ESX
- Microsoft Windows Server 2008 R2
- 64-bit system
- 2x Quad Core CPU
- 8 GB RAM
- Google Chrome 56.0.2924.87
- 10 gbps virtual Ethernet network connection

Scenario 1: Test Data Model Including Only Orders Entity

This scenario includes a test data model that contains only the Orders entity. The following table shows the time that is taken to retrieve the specific result:

Requirement	Filter	Time Taken
Single filter on Orders	Get all orders for OrderDate=2016-12-30 00:00:00.0 or EmployeeID=2	Less than 3 seconds
Multiple filters on Orders	Get all orders for OrderDate=2016-12-30 00:00:00.0 , EmployeeID=2, and ShipVia=1	Less than 3 seconds
Return 10,000 orders based on a filter	Get all orders for ShipVia=1	Less than 3 seconds

Scenario 2: Test Data Model Including Orders and Order Details Entities

This scenario includes a test data model that contains Orders and Order Details entities. In this test data model, the Orders entity acts as a root entity and the OrderID field from Orders acts as a model key. The child entity Order Details is associated with the Orders entity. The following table shows the time that is taken to retrieve the specific result:

Requirement	Filter	Time Taken
Filter on Orders	Get all orders for EmployeeID=2	Less than 4 seconds
Filter on Order Details	Get all orders for TotalAmount=197.94	Less than 4 seconds
Filter on both Orders and Order Details	Get all orders for TotalAmount=197.94 and ShipRegion=WV	Less than 4 seconds

Scenario 3: Test Data Model Including Orders, Order Details, Products, and Suppliers Entities

This scenario includes a test data model that contains Orders, Order Details, Products, and Suppliers entities. In this test data model, the Orders entity acts as a root entity and the OrderID field from Orders acts as a model key. All other child entities are directly or indirectly (through associations with other entities) related to the Orders root entity. The following table shows the time that is taken to retrieve the specific result:

Requirement	Filter	Time Taken
Filter on Orders and other child entities	Get all orders for CompanyName=460LTD, Discount=52, EmployeeID=2, ProductID=1, ShipVia=1, SupplierID=2, and UnitPrice=348	Less than 4 seconds
Filter on child entities; however, no filter on Orders	Get all orders for CompanyName=460LTD, Discount=52, ProductID=1, SupplierID=2, and UnitPrice=348	Less than 4 seconds
No filter on Orders, but multiple filters on child entities	Get all orders for CompanyName=460LTD, Discount=52, ProductID=1, SupplierID=2, TotalAmount=197.94, and UnitPrice=348	Less than 4 seconds

Data Prefetch

The Data Prefetch feature allows Test Data Engineers to find test data faster. When Data Prefetch is active, data from data sources used by test data models is cached in a TDM database. Testers' queries are evaluated in the cache, which reduces query time. For more information about Test Data Model creation, see Create and Edit a Find & Reserve Model.

Data Prefetch is supported for MS SQL and Oracle databases. See Supported Data Sources for a complete list of supported data sources.

When creating a new Data Model, you choose between three options:

- data prefetch on demand (data synchronization)
- periodic data prefetch (data synchronization)
- off (no data synchronization)

A Data Model cannot switch betwen Synchronization and non-Synchronization after creation.

Overview

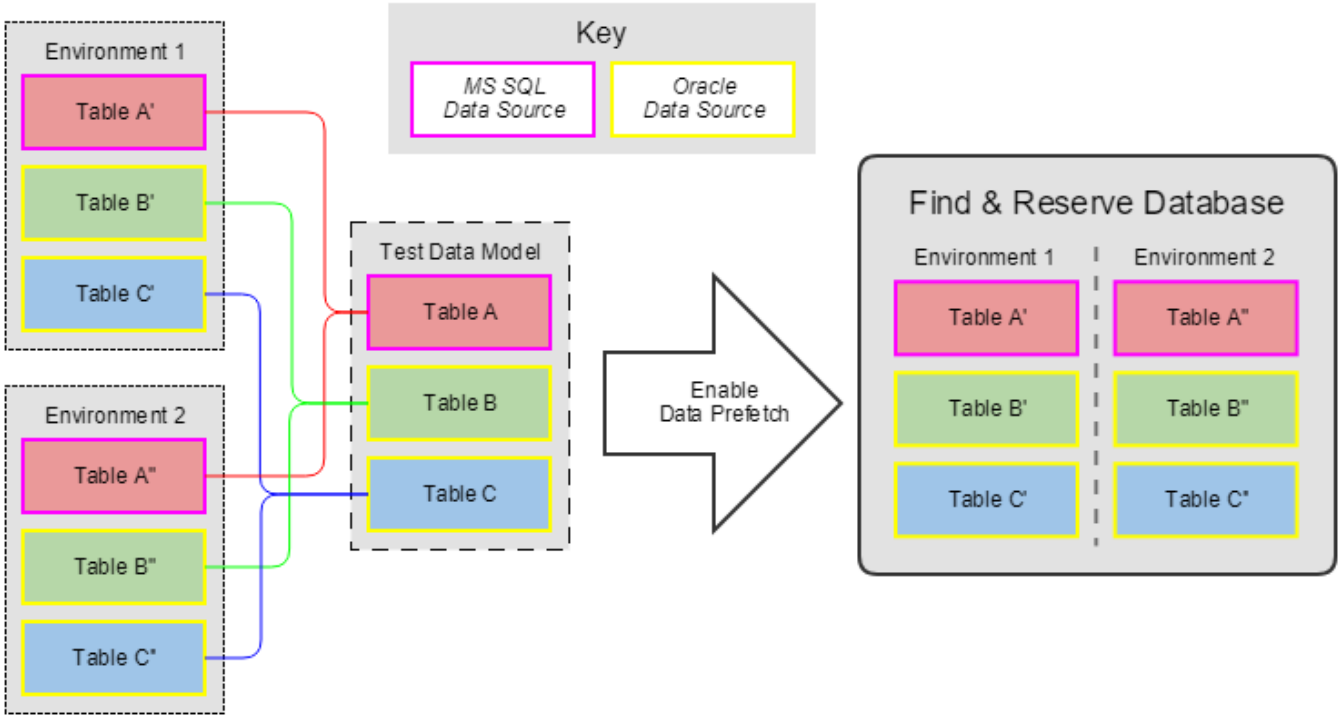
Data Prefetch is recommended for test data models that use multiple data sources, especially for smaller data sets with many entities and relationships between them. When Data Prefetch is active, data from all your data sources across all your environments is cached in a TDM database. When this data is cached, the query runs on the cached data.

> **NOTE**
>
> When you disable Data Prefetch for a data model, all cached data is deleted.

Data Prefetch synchronizes data for all environments and data sources used by test data models.

For example, if your model uses 3 tables in 2 environments, 6 tables are created in the prefetch database (3 for each environment). See diagram below.

Figure 40: FR diagram



No Data Prefetch -- Create a Reservation Table

If you create a data model with **Data Prefetch: Off** (no data synchronization), TDM queries your source database directly, without prefetching a copy of the data into the TDM database. TDM does not change your source data, but the Test Data Engineer must create an additional reservation table in the source database in the second wizard step. TDM uses the Reservation Table to track reserved rows in the root table. The Reservation Table shares its primary key with the root table to identify each reserved row. This functionality supports only a single data source.

To create the Reservation Table:

1. Click **Modelling, Find & Reserve, Create Model** and create a data model with **Data Prefetch: Off**.
2. Select a table and click the right arrow button to add the first table to your data model.
   The **Reservation Table** dialog opens.
3. All models which use the same root table must use the same Reservation Table. Perform *one* of the following options:
   – Click **Add new Reservation Table**, define a new table name, accept the defaults, and click **Save**.
   – Select a previously created text column which serves as reservation ID for this table.
4. Click **Finish**.
   The data model is created. A new icon appears in the Find&Reserve table row with the tooltip "Click to View Reservation Table Details"

Table creation can fail if the Test Data Engineer lacks permissions. In this case, TDM displays additional manual steps in the dialog:

1. Download the provided SQL files, one file per environment.
2. Send these files to your Database Administrator to execute them.

3. In the Portal, click **Data Model, Actions, Rescan**.
   The new table appears.
4. Drill down into the Data Model again where you were previously interrupted due to lacking permissions.
   Now the reservation table exists.
5. Select the $RESERVATION_ID column on the table that contains the relevant ID. Other, grayed-out tables are not reservation tables.
   The **Save** button becomes active.
6. Click **Save** and **Finish**.

If Testers attempt to use an environment that does not have a reservation table, a validation error appears that tells them to contact the TDM Admin or Test Data Engineer to create the table. Testers themselves do not have permissions to create this table.

To create a missing table, follow these steps:

1. Drill down into the Data Model, and look for a yellow warning icon "Reservation table is missing in Your_Database_Name" in the affected table row.
2. Click the **View Reservation Table Details** button for the table.
3. Click **Propagate Table to All Environments**.
   TDM creates the reservation table for this environment.

Configuration file

In a standard installation the Test Data Manager configuration file is located at `C:\Program Files\CA\CA Test Data Manager Portal\conf\application.properties`. Make changes to this file to adjust the following:

- Synchronization triggers
- Location of prefetched databases

Synchronization triggers

CA TDM synchronises fetched data with the original data sources one of two ways: **Periodically**, and **On Demand**. You can choose this behaviour for F&R models on the **Models for Find & Reserve** page. See Create and Edit a Find & Reserve Model.

Periodic synchronization

Data Prefetch Periodic enables F&R model data synchronization into the TDM database. Data is synchronized immediately after model creation. TDM synchronizes data every day automatically, and you also have the option to start it manually. You can include tables from multiple databases in the model.

By default, every 2 hours CA TDM propagates all changes (deletions, updates, inserts) from original data sources to cached data. You can modify the sync period by editing the following line of the configuration file:

```
  # delay between synchronization runs (in minutes)
 tdmweb.findReserveService.sync.delay=120
```

In addition to the periodical updates, the fetch process starts (or restarts) on the following triggers:

- When the toggle is switched on for the first time.
- When models, environments or data sources are changed.
- When you click 'Fetch Data' from the **Options** menu on the **Models for Find & Reserve** page. See Create and Edit a Find & Reserve Model.

On Demand Synchronization

Data Prefetch On Demand enables Find & Reserve model data synchronization into the TDM database. Data is synchronized immediately after model creation. You start further synchronizations manually. You can include tables from multiple databases in the model.

CA TDM only synchronizes Find & Reserve Models with On Demand Data Prefetch Synchronization when you click 'Fetch Data' from the **Options** menu on the **Models for Find & Reserve** page. See Create and Edit a Find & Reserve Model.

Specify location for prefetched data

By default, the Data Prefetch feature caches data in a Repository database (GTREP). The size of this database depends on the volume of data cached. For larger volumes of data, consider whether to use gtrep database as the storage for fetched data. You can specify a different database to store fetched data by changing the configuration file.

For the cleanest switch between prefetched databases, follow the Best Practice Guide below.

> **WARNING**
>
> If you change the configuration file before you disable Data Prefetch on models, data is not removed from the original prefetch database, and it is necessary to perform a manual cleanup.

You can manually specify where CA TDM stores prefetched databases in the configuration file. The relevant values to change are highlighted in red (replace all of '${...}') and the possible replacement values are in bold on the preceding commented line:

```
 # e.g jdbc:oracle:thin:@//tdmserver:1521/tdm.ca.com or jdbc:sqlserver://
tdmserver:1433;database=gtrep
 tdmweb.TDMFindReserveService.db.spring.datasource.url=${spring.datasource.url}
  tdmweb.TDMFindReserveService.db.spring.datasource.username=
${spring.datasource.username}
  tdmweb.TDMFindReserveService.db.spring.datasource.password=
${spring.datasource.password}
  # oracle.jdbc.OracleDriver or com.microsoft.sqlserver.jdbc.SQLServerDriver
 tdmweb.TDMFindReserveService.db.spring.datasource.driver-class-name=
${spring.datasource.driver-class-name}
  # ORACLE or SQL_SERVER
 tdmweb.TDMFindReserveService.db.spring.jpa.database=${spring.jpa.database}
```

For example, to cache the database at **jdbc:oracle:thin:@//tdmserver:1521/tdm.ca.com**, the lines of the configuration file would be:

```
 # e.g jdbc:oracle:thin:@//tdmserver:1521/tdm.ca.com or jdbc:sqlserver://
tdmserver:1433;database=gtrep
 tdmweb.TDMFindReserveService.db.spring.datasource.url=jdbc:oracle:thin:@//
tdmserver:1521/tdm.ca.com
 tdmweb.TDMFindReserveService.db.spring.datasource.username=MyUsername
tdmweb.TDMFindReserveService.db.spring.datasource.password=MyPassword123
 # oracle.jdbc.OracleDriver or com.microsoft.sqlserver.jdbc.SQLServerDriver
 tdmweb.TDMFindReserveService.db.spring.datasource.driver-class-
name=oracle.jdbc.OracleDriver
 # ORACLE or SQL_SERVER
 tdmweb.TDMFindReserveService.db.spring.jpa.database=ORACLE
```

Required user privileges

If you specify an alternative storage location for the cached database, you must ensure that the user with which you access this database has the necessary privileges to access the specified database. The required privileges are as follows:

Oracle

Required privileges:

- CONNECT
- RESOURCE
- CREATE ANY DIRECTORY
- SELECT ANY TABLE
- CREATE VIEW
- UNLIMITED TABLESPACE

Sample command to give user privileges:

```
GRANT CONNECT, RESOURCE, CREATE ANY DIRECTORY, SELECT ANY TABLE, CREATE VIEW, UNLIMITED
TABLESPACE TO <username>;
```

MS SQL Server

On MS SQL Server databases, for a user to access the prefetched data, they must be the database owner or a user with equivalent privileges.

Data Prefetch Troubleshooting

By default, CA TDM Portal's log are located at `C:\ProgramData\CA\CA Test Data Manager Portal\logs \TDMFindReserve.log`.

Best Practice Guide for Data Prefetch

To store prefetched databases somewhere other than in the gtrep database, follow these steps:

1. Turn off prefetch feature for all models. Fetched data is deleted.
2. Stop CA TDM Portal in the **Windows Services** dialog.
3. Change configuration file (application.properties) to store cached data in the new DB.
4. Start CA TDM Portal.
5. Turn on Data Prefetch feature. The database specified in the configuration file is now used.

WARNING

If you change the configuration file before you disable Data Prefetch on models, data is not removed from the original prefetch database, and it is necessary to perform a manual cleanup.

How long does the prefetch take?

These performance statistics were measured with TDM Portal running on a machine with 4-Core CPU @ 2.7 GHz, 16 GB RAM, the target database was on a machine with 2 x virtual 2 GHz CPU, 4GB RAM. These are valid only for the gtrep repository.

Size of database	MS SQL DB to MS SQL DB *or* Oracle DB to Oracle DB	Oracle DB to MS SQL DB
1 table with 100,000 rows	26 seconds	1 min 39 secs
1 table with 1,000,000 rows	4 mins 8 secs	5 mins 49 secs

Data not fetched

If a tester receives empty or incomplete search results from a Test Data Model with prefetched database(s), check **TDMFindReserve.log** for error messages.

Why was data not fetched?

- If the Find&Reserve database was changed (`tdmweb.TDMFindReserveService.db.spring.datasource.*` properties in the configuration file, see Specify where prefetched databases are stored) and this database connection does not work there will likely be a DB-related exception in TDMFindReserve.log immediately after TDM Portal startup. If this is due to the user lacking necessary privileges, see Required user privileges.
- If there is a data type conversion failure (i.e. value cannot be stored in data type), and the source database was Oracle, and the target (i.e. cached) database was another database type, then use an Oracle database as the target database.
  For example, dates from BC are formatted differently in Oracle, and these cannot convert to other database types.
- If there is a database error (due to storage, database configuration etc) then resolve this as the Administrator user.

Remove prefetched data from gtrep database

If you specify a location for the prefetched data after the first prefetch, the gtrep database still contains these tables from the first prefetch. To remove them, run the following scripts on the gtrep schema/database as schema/database owner:

MS SQL Server

```
 DROP TABLE test_data_model;
DROP TABLE data_view_property;
DROP TABLE data_view_instance;
DROP TABLE data_view;
DROP TABLE schema_version_find_reserve;


 DECLARE @cmd varchar(4000)
DECLARE cmds CURSOR FOR
SELECT 'drop table [' + Table_Name + ']'
FROM INFORMATION_SCHEMA.TABLES
WHERE Table_Name LIKE 'dvid_%'


 OPEN cmds
WHILE 1 = 1
BEGIN
```

```
 FETCH cmds INTO @cmd
 IF @@fetch_status != 0 BREAK
 EXEC(@cmd)
 END
 CLOSE cmds;
DEALLOCATE cmds
```

Oracle

```
 DROP TABLE TEST_DATA_MODEL;
DROP TABLE DATA_VIEW_PROPERTY;
DROP TABLE DATA_VIEW_INSTANCE;
DROP TABLE DATA_VIEW;
DROP TABLE "schema_version_find_reserve";


 BEGIN
 FOR c IN ( SELECT table_name FROM user_tables WHERE table_name LIKE 'dvid_%' )
 LOOP
 EXECUTE IMMEDIATE 'DROP TABLE "' || c.table_name || '"';
 END LOOP;
 END;
```

Truncation of Databases

CA TDM truncates prefetched (i.e. target) tables when it is unable to guarantee the same sort order of the source and target databases. This truncation occurs on synchronisation of the prefetched tables with the source tables.

> **NOTE**
>
> CA TDM never truncates source databases.

Oracle and MS SQL

CA TDM truncates target tables only when the following conditions are true for either the source and target databases:

1. The primary key is of one of the following types:
   - **CHAR**
   - **VARCHAR**
   - **NCHAR**
   - **NVARCHAR**

   > **NOTE**
   >
   > In Oracle and MS SQL databases, NCHAR and NVARCHAR data types always have encoding UTF-16.
2. **AND** one of the following conditions is met:
   a. **EITHER** The primary key column character set is **not** one of the following character sets:

- **ASCII**
- **ISO8559-1**
- **UTF-8**
- **UTF-16**

b. **OR** the primary key column's sort order is **not** binary

DB2

CA TDM truncates prefetched tables from DB2 source tables, if the primary key is of one of the following types:

- **CHAR**
- **VARCHAR**

Configure Form Based Test Data Reservation Service

CA Test Data Manager (CA TDM) helps testers link the test cases with relevant data and expected results, as well as eliminate over testing of certain functionality and to prioritize tests based on criticality. CA Agile Requirements Designer (CA ARD) helps to plugging in the CA TDM to existing test management tools, such as HP Application Lifecycle Management (HPALM) and CA Agile Central.

As a Test Data Engineer, you can create a diagram in the CA ARD that represents the set of requirements as a visual flow. In every flow, there is a certain number of possible paths you can take between the Start and the End block and each path represents a test case. You can associate the ARD Flow with CA TDM projects and enable the flow for tester.

Testers can access the ARD Flows that the TDE enabled for them.The ARD Flows are displayed as forms to testers in the Self-Service Catalog section of the Portal. Testers can then consume the ARD Flows (as a form) to find and reserve the data for their specific test cases.

The following procedures explain you how to use CA TDM and CA ARD together to define test data requirements, design the visual flows using ARD, and enabling these ARD flows to tester self service in the TDM Portal.

- Test Matching and Re-Matching
- Test Matching HP ALM Integration
- Test Matching Rally Integration

For more information about how testers use an ARD Flow to find and reserve the test data, see Reserve Data with Self Service Catalog Forms.

Test Matching and Re-Matching

In a software product development environment, testers use product requirements to create their own test cases, which concentrate on a requirement’s outcomes. Test cases probe each way a user may interact with a feature in a software application.

After creating the necessary test cases, testers essentially need the test data which is typically spread across multiple databases. Manually creating test data is not a viable option and rarely results in data that has sufficient test coverage, leading to bugs in production.

Test Case Match uses powerful data mining functionality within Datamaker to quickly identify, mine and link data to automated test cases, from multiple sources. It helps the test engineers to find the right data in their testing and development environments. Test Matching helps you to link the test case with the actual test data.

Note: Test matching supports only Microsoft SQL Server and Oracle databases. You must import data from other data sources into a SQL Server or Oracle database for test matching purposes.

Follow these procedures to match your test cases with the right data:

Construct Test Data Mart

The Test Data Mart is a collection of summary tables built up from the core attributes identified in your test criteria. Construct a Test Mart within a database and register with Datamaker using a connection profile. Ensure that your test mart includes necessary columns to store the following information:

- **Job ID**
  - **Data Type:** Numeric for MS SQL Server; Number for Oracle.
  - **Data Length:** 10
  - **Example:** MS SQL Server - NUMERIC(10); Oracle - NUMBER(10,0)
- **Expected Results**
  - **Data Type:** VARCHAR or NVARCHAR for MS SQL Server; VARCHAR2 or CLOB for Oracle.
  - Data Length (**Recommended**): Minimum - 200; Maximum - as allowed in the database.
  - **Example:** MS SQL Server - VARCHAR(max); Oracle - CLOB(max)
- **Test Name**
  If multiple test cases acquire a read-share lock, the value in this column is proportional to the number of test cases allocated for the shared record.
  - **Data Type:** VARCHAR or NVARCHAR for MS SQL Server; VARCHAR2 or CLOB for Oracle.
  - Data Length (**Recommended**): Minimum - 306; Maximum - as allowed in the database.
  - **Example:** MS SQL Server - VARCHAR(max); Oracle - CLOB(max)
- **Test Locking**
  If multiple test cases acquire a read-share lock, the value in this column is proportional to the number of test cases allocated for the shared record.
  - **Data Type:** VARCHAR or NVARCHAR for MS SQL Server; VARCHAR2 or CLOB for Oracle.
  - Data Length (**Recommended**): Maximum
  - **Example:** MS SQL Server - VARCHAR(max); Oracle - CLOB(max)
- **Test User**
  If the testers are allowed to do a read-share lock, the value in the "Test User" column is proportional to the number of users allocated for the shared record.
  - **Data Type:** VARCHAR or NVARCHAR for MS SQL Server; VARCHAR2 or CLOB for Oracle.
  - Data Length (**Recommended**): Minimum - 200; Maximum - as allowed in the database.
  - **Example:** MS SQL Server - VARCHAR(max); Oracle - CLOB(max)

Review the following points when defining Test Data Mart columns:

- Test Data Mart column names should not match with the SQL keywords.
- Do not use spaces in the column names.
- Ensure that the Test Data Mart columns are defined with appropriate data types including the required maximum length. For example, a test match might result in the number of entries that exceed the limit applied on the Test Data Mart column data type. In this case, the test match fails and an error is displayed. To address such situations, we recommend that you define your column data types based on your business requirements. For example,
  `nvarchar(max)`
  .

Create Project in Datamaker

If you have a simple application, you can work with just one project with one version. For complex applications, you can choose to work with multiple project versions. You can save your data definitions against an initial version of the project and then save (Register) any changes or new tables against a new project version. This method allows you to only identify changes from version to version, rather than having to save the definitions of all the tables. For more information, refer Datamaker User Guide.

Follow these steps:

1. Launch Datamaker and connect to the Profile which is mapped with the Test Data Repository.
2. Provide login credentials in the logon dialog and click the Connect to user button.
3. Select the required Data Target and click the Connect to Databases and Startup button.
4. Go to Projects menu and click Project Manager.
5. Right click the Projects and click New Project and Version.
6. Specify the following in the Create New Project dialog and click the Save Details button:
   - **Project Name**
     Specifies the name of the project.
   - **Project Description**
     Specifies the description of the project. If left empty defaults to Project Name
   - **Version**
     Specifies the version number of the project.
   - **Version Description**
     Specifies the description of the version. If left empty defaults to Version.
   - **Generic**
     Specifies whether the project is Generic or not.
7. Click OK on Project Settings and New Project dialogs.

Register Test Data Mart

You must register the data definitions to Datamaker to manage the data. You can register the data definition just once per version. Thereafter, you can re-register only the table or data definition, if you edit them.

Follow these steps:

1. Expand the newly created project from the Projects tree view and locate the Version folder.
2. Right click the Version folder and click Register.
3. In the Select the type of object to register dialog, select Database Table and click the Next icon.
4. In the Register Object Explorer window, select the test data mart table that you want to register with the project.
5. Select Register Tables from Data Target from the dropdown list, and then click the Go icon.
6. Click the Go icon, in the Reconcile Objects to Register window.
7. Click Yes, in the Calculate Table Order dialog.
8. Close the windows opened during the above steps.

Create a Test Match Data Pool

A data pool is a centralized data object, where the necessary information to perform business transactions is stored in a standardized way. When creating a data pool, you can specify the type based on the format of the data you wish to save. For performing test match, you can create the data pool of Test Match type.

Follow these steps:

1. Expand the newly created project from the Projects tree view and locate the Version folder.
2. Right click on the Version folder, and click New Data Group.

3. Specify the following in the New Data Group dialog and click Save Details:
   - **Name**
     Specifies the name of the Data Group.
   - **Description**
     Specifies the description of the Data Group. If left empty defaults to Name.
   - **On Demand?**
     Specifies the option to access data and visual flows on demand using the Datamaker Service Layer. Select, if you want to run the test match also from Test Data on Demand (TDoD).
   - **Type**
     Specifies the type of Data Group. Following are the available options:
     - **Normal**
       Select this option to specify that the data group is of Normal type.
     - **GTSubsets**
       Select this option to specify that the data group is of GTSubsets type.
     - **CA Agile Requirements Designer**
       Select this option to specify that the data group is of CA Agile Requirements Designer type.
4. Click OK in the confirmation dialog.
5. Right click on the Data Group you created, and then click New Data Set.
6. Specify the required details in the New Data Set window and click Save Details icon. Following are the additional fields to specify in this step:
   - **Make Test Data available for external use**
     Specifies whether the Test Data can be made available for external use.
7. Click OK in the confirmation dialog.
8. Right click on the Data Set you created, and then click New Data Pool.
9. Specify the following in the New Data Pool window and click Save Details icon. Following are the additional fields to specify in this step:
   - **Type**
     Specifies the type of Data Pool. Select Test Match from the dropdown list. Following are the available options:
     - **Data Only**
     - **Normal**
     - **Test Match**
     - **GTSubsets**
     - **CA Agile Requirements Designer**
10. Go to the Test Matching tab and do the following:
    - Click the Tables and Columns tab and specify the following:
      - Profile Name
      - Table Owner
      - Match data in table
      - Report match in column
      - Report user in column
      - Report locking in column
      - Expected Results in column
      - Report Job ID in column
      - Default Summary Table Publish
    - Click the Key Columns tab and specify the following:
      - **Report Key Columns**
        Set of columns that are uniquely identifiable. Locks one record, and blocks all the duplicate records found with same values in reporting key columns.
      - **Primary Key Columns**

These are the surrogate key columns. The clauses of the query are formed based on this column.

- **Data Attribute Columns**
  These are the reporting columns. This identifies the attribute columns to show in the reports.

11. Click the Save Changes and Exit icon.

Connect to a Database Using a Non-EZconnect Connection String

When I run a test match in CA TDM Portal and your `spring.datasource.url` parameter is assigned with a complicated or non-standard jdbc connection string, and you try to connect to a database, you may get an error message similar to the following.

```
Exception occured while updating job start information
ORA-12154: TNS:could not resolve the connect identifier specified
```

In this case, define the connection string in your `application.properties` file in C# format.

1. Navigate to the directory where you installed the CA TDM Portal, and open the conf subdirectory.
2. Locate and open the application.properties file in a text editor.
3. Define the connection string. Add or update the following parameter:
   `tdmweb.TDMLegacyExecuterService.testmatch.connectionString`
   Format: Enter the connection string in C# style.

Edit Test Case Data Criteria

Test Case Data Criteria (TCDC) is a logical table stored inside the Datamaker repository (static data pool) containing one row per test case. Each row defines one or more query criteria that define the test case's data requirements for a given test data entity. Also specifies whether each test case requires an exclusive or a shared lock on the matched test data entity instance (i.e. whether the test case updates the underlying data for that instance, or whether it only reads it).

Follow these steps:

1. Expand the project in which you created the Data Pool with type as Test Match.
2. Double click the Data Pool, and click Edit Data.
3. Expand the Registered Objects tree view, locate Used Tables, and click TESTCASE_DATA_CRITERIA.
4. Add as many test cases as you want to match with the test data. Add a separate row for each test case.

   NOTE

   If TDM is integrated with HP ALM, the test name in ALM and TCDC must be the same. The test name is case sensitive.

5. Go to each test case row and click on each column to provide the test case data criteria for the respective test case. Optionally double click on each column to open the editor window. Enter the values in the editor window for respective column and click validate. Verify that the value you entered is valid and then click Save.
   Following are the mandatory columns for each test case:
   – **Test Repeater**
   Specifies the amount of test data you want to have for the given test condition.
   – **Test Conditions**
   Specifies the test conditions to apply on the primary key columns in the testmart properties. The conditions you enter in these columns form the WHERE clause for matching the data. The TCDC table has 20 columns for specifying the test conditions.

- The test conditions you enter in these columns result in "**AND"** conditions. For example, if you enter "salary < 1000" in "Test Conditions" column and "age > 30" in "Test Conditions2" column, then the final test condition becomes "where salary < 1000 **AND** age > 30".
- If you want to specify "**OR**" conditions among testmart columns, then enter your complete test condition only in the "TEST CONDITIONS" column. Do not use multiple test conditions columns.
- Do the following to enter or edit the values in each test conditions column:
  a. Double-click on the **Test Conditions** column.
  The editor window opens.
  b. Enter the test condition in the text box on the editor window.
  c. Click the **Validate WHERE clause in Test Match Pool** button to verify whether the test condition you entered is valid or not.
  d. Click the **Count number of occurrences** button to see the matched row count for the test condition you entered.
  e. Click **Save**.
  The values you entered are saved in the respective test conditions column for the corresponding test case.

– **Test Override Priority**
  - **Default:** 99
  - **Range:** 1 – 99.
  1 sets the highest priority and 99 sets the lowest priority.

– **Test Locking**
  - **Default:** LK
  - **Available options:** LK and RS
  LK: Specifies Exclusive Lock. Rows that have LK (Exclusive Lock) for a test case cannot be allocated to another test case, even if they meet all the other criteria.
  RS: Specifies Read Share. Rows that have RS (Read Shared Lock) can be additionally allocated to other test cases that only require read shared locks, but not to test cases requiring an exclusive lock.

6. Click the Save icon.

Run Test Match

After having the test cases, test data criteria and the test data ready, now you can match the test cases and get the right test data based on the specified criteria.

Follow these steps:

1. Right click the test match data pool in which you added the test case data criteria, and then click Test Match.
2. Test Match window opens consisting of the Table and Columns, Key Columns and Runtime Parameters tabs.
3. Go to the Runtime Parameters tab and specify the following:
   – Limit to a single Test
   – Simulate Testmatch
   – Allow Partial Matches
   – Clear Existing Matches
4. Click Match.
5. In the Submit Test Match dialog specify the following and click the Go icon.
   – **Immediate**
   Runs the test match right away.
   – **Remote**
   Runs the test match as scheduled. Specify the schedule start date, time, email address, and thread in the respective fields. Follow the test data repository time zone.
6. After successfully completing the test match, the view reports dialog opens with the following options:

- – **Yes**
  Click to open the reports in the browser.
- – **No**
  Click to exit. To access the reports go to **%AppData%\Grid-Tools\Testmatch** using Run Command.

View Test Match Reports

After successfully running the test match, Datamaker provides you two types of reports.

- **Matched Key Report**
  This report shows matched keys of the test cases that have matching data found.
- **Testmatch Summary Report**
  This report shows the summary of test matching for active test cases in Test Case Data Criteria table.

Snapshot of Test Data Mart

You can take a snapshot of the test data mart and manage the necessary information. You can create a history table in the database which holds the history of test matching.

Construct Test Data Mart History Table

Test mart History table is a table used to store the history of test matching. After successfully running the test match, test data mart table gets updated with necessary information. History table is needed to store the updated test mart after the test match. Construct a test mart history table in the database where the test mart is created. Ensure that your test mart history table contains all the columns of test data mart along with the columns to store the following information:

- **History Type**
  - **Data Type:** VARCHAR or nvarchar for MS SQL Server; VARCHAR2 for Oracle.
  - Data Length (**Recommended**): Minimum - 17; Maximum - as allowed in the database.
  - **Example:** MS SQL Server - VARCHAR(17); Oracle – VARCHAR2(17)
- **History Job ID**
  - **Data Type:** Numeric for MS SQL Server; Number for Oracle.
  - **Data Length:** 10
  - **Example:** MS SQL Server - NUMERIC(10); Oracle – NUMBER(10,0)
- **History Date**
  - **Data Type:** DATE for both MS SQL Server and Oracle.
  - **Example:** DATE
- **History Notes**
  - **Data Type:** VARCHAR or NVARCHAR for MS SQL Server; VARCHAR2 for Oracle.
  - Data Length (**Recommended**): As allowed in the database.
  - **Example:** MS SQL Server - VARCHAR(200); Oracle – VARCHAR2(200)

Note: Test Data Mart column names should not match with SQL Keyword. Do not use spaces in the column names.

Register Test Data Mart History

Registering test data mart history follows the same steps that you followed to register test data mart table. For more information refer Registering Test Data Mart.

Once test mart history table is registered, we need to map it’s columns in Test match Data Pool.

Follow these steps:

1. Expand the project in which you created the Data Pool with type as Test Match.
2. Right click the Data Pool, click Data Pool Properties.
3. Go to the Test Matching tab and do the following:
   – Specify the following under the Tables and Columns tab:
     • Store History in table
     • History Date in column
     • History Notes in column
     • History Type in column
     • History Job in ID column
4. Click the Save Details icon.

Take Snapshot

After creating the history table and adding the necessary new columns, you can take a snapshot of the test data mart. The snapshot pushes the locked and blocked rows to test mart history table. It supports three modes of snapshot:

1. **Pre-Match Snapshot**
   Takes snapshot before performing the test match.
2. **Post-Match Snapshot**
   Takes snapshot after performing the test match.
3. **No-Match Snaphot**
   Takes snapshot without performing the test matching (current state).

Follow these steps:

1. Right click the test match data pool in which you added the test case data criteria, and then click Test Match.
2. Test Match window opens consisting of the Table and Columns, Key Columns and Runtime Parameters tabs.
3. Go to the Runtime Parameters tab and select Show Advanced Options.
4. Select one of the following from Advanced Options and click Match:
   – **Snapshot Test Mart before matching**
     Specifies the option to take snapshot before performing test match.
   – **Snapshot Test Mart after matching**
     Specifies the option to take snapshot after performing test match.
   – **Snapshot Test Mart**
     Specifies the option to take snapshot without performing test match.

Snapshot of locked and blocked rows for the test cases is created in Test Mart History table with History Job ID.

Perform Test Re-Match

After running the test match, based on the new requirements when development teams add new features to the software application, it is important to also test old features. Although existing features have not explicitly changed, new features can have an unintentional impact on older features. By testing the test cases from past releases, testers can ensure that the entire product operates as expected.

Testers need to run the test match in multiple iterations to fulfil the test data requirement for the newly added test cases. Datamaker allows you to create a history table, take a snapshot of the test mart and then run a test re-match. This helps the testers to match the newer test cases with right data, retaining the previously allocated reporting keys to older test cases wherever possible.

Refresh Test Mart

As a test data engineer, before each testing cycle, refresh (edit) the test mart to ensure the availability of the current contents of application database(s).

The refreshed test mart contains both the old and new reporting keys. Also the attributes of the old reporting keys can change in the refreshed test mart, due to changes to the respective attributes in the application database(s).

In the new testing cycles, testers add new test cases to TCDC in addition to the old test cases. In such case old test cases are first tried to match with previously allocated reporting keys wherever possible. Where not possible, allocates the old test cases to new reporting keys.

Edit Test Case Data Criteria

Add new test cases as you want to match with the new test data by following the same steps that you followed to edit Test Case Data Criteria for test match.

Run Test Re-Match

Test re-match considers the old test cases with higher priority compared to new test cases. Test Re-Match is performed in two passes.

In Pass 1, matches the old test cases with previously allocated reporting keys from the snapshot. It uses Priority, Rarity and Order concept for matching. The old test cases are matched fully or partially wherever possible.

In Pass 2, processes the partially matched old test cases and allocates the new reporting keys to the fully matched old test cases. Then processes the new test cases using the Priority, Rarity and Order concept. If you allow partial matching, retains the partially matched reporting keys for old and new test cases. If you do not allow partial matching, then rolls back the partially matched reporting keys for test cases. Rolled back old test cases are marked with Skipped, and the new test cases for which the data is not available are marked with No Match Found.

Follow these steps:

1. Right click the test match data pool in which you added the test case data criteria, and then click Test Match.
2. In the Test Match window, go to Runtime Parameters tab and select Show Advanced Options.
3. Specify the following and click Match:
   - **Limit to a single Test**
     Specifies the option to perform test matching on single test case. This is supported only when you perform test match.
   - **Simulate Testmatch**
     Select this option to simulate test matching.
   - **Allow Partial Matches**
     Select to retain the partial matches. If not selected, rolls back the partial matches and marks as Skipped.
   - **Retain existing matches where possible**
     - **Job ID**
       Specifies the History Job ID that contains the snapshot of old keys.
     - **History Type**
       Specifies the type of snapshot.
   - **Retry Prior Partial Matches**
     Select to re-attempt the test cases which are partially matched in prior runs.
   - **No Matching On Clear Down**
     Select to clear the existing locks in test mart without performing test matching when used along with Clear Existing Matches checkbox.
   - **Snapshot Test Mart before matching**

Select to take snapshot before performing the test re-match.

- **Snaptshot Test Mart after matching**
  Select to take snapshot after completing the test re-match.
- **Snapshot Test Mart**
  Select to take snapshot of the test mart in the current state without performing the test re-match.
- **Clear Existing Matches**
  Select this option to clear existing matches in the test mart before performing test matching

4. In the Submit Test Match dialog specify the following and click the Go icon.
   - **Immediate**
     Runs the test match right away.
   - **Remote**
     Runs the test match as scheduled. Specify the schedule start date, time, email address, and thread in the respective fields. Follow the test data repository time zone.
5. After successfully completing the test re-match, the view reports dialog opens with the following options:
   - **Yes**
     Click to open the reports in the browser.
   - **No**
     Click to exit. To access the reports go to **%AppData%\Grid-Tools\Testmatch** using Run Command.

View Test Re-Match Reports

After successfully running the test re-match, Datamaker provides you the following four types of reports:

1. **Matched Key Report**
   This report shows matched keys of both the old and new test cases.
2. **Testmatch Summary Report**
   This report shows the summary of test matching for active test cases in Test Case Data Criteria table.
3. **Retain Key Report-PASS1**
   This report shows the status of old test cases that are matched with old reporting keys in the snapshot specified. Also shows the data attributes comparison results between old and new reporting keys.
4. **Retain Key Report - PASS2**
   This report shows the status of old test cases that are matched with new reporting keys.
5. **JB<id>_RetainKey_Summary - PASS2**
   This report shows the test name and match status for pass 2. This report shows the test cases sorted by test name for each test user. The test cases that do not have any test user assigned are displayed first, followed by test cases that are sorted based on the test user name. You can find these reports under the path, ...\AppData\Roaming\Grid-Tools \Testmatch.
6. **JB<id>_RetainKey_<test user name> - PASS2**
   This report shows the test name and match status for pass 2 for each test user. A separate report is generated for each test user and placed under the path ...\AppData\Roaming\Grid-Tools\Testmatch.

Test Matching HP ALM Integration

Test Matching integration with HP ALM is designed to work with HP ALM 11 and above. This article guides you through various configurations that are essential to work the integration.

Follow these procedures:

Verify Prerequisites for ALM Integration

Before you set up the HP ALM integration in Test Data Manager, perform the following one-time system configuration: Add a firewall exception, configure your Windows group policy, and install the HP ALM Connectivity add-in.

Add Firewall Exception

ALM Service is a self-hosted WCF service and therefore requires you to add a firewall exception for incoming TCP connections for the URL and port where the service runs. The exact steps depend on your firewall software. The following example is for the Windows Firewall.

Follow these steps:

1. Run wf.msc from the command line.
   The Windows Firewall with Advanced Security window opens.
2. Click **Inbound Rules**. Click **Actions**, **New Rule**.
   The New Inbound Rule Wizard opens.
3. Choose rule type **Port** and click Next.
4. Choose **TCP protocol**.
5. Choose **Specific local ports** and enter 8095 in the text box. Click Next.
6. Choose **Allow the Connection,** and click Next.
7. Enable the following options, and click Next:
   - Domain
   - Private
   - Public
8. Enter a Name and Description for the rule.
   Example: "Opened port 8095 for CA Test Data Manager ALM integration."
9. Click Finish.

Install HP Quality Center Connectivity Add-in

Follow these steps:

1. Open HP ALM in a web browser.
2. Click **Add-Ins Page**.
3. Click **HP Quality Center Connectivity** and click **Download**. Save the add-in file as TDConnect.exe.
4. Run TDConnect.exe to install it.
   The add-in is ready to use on the computer that has TDM installed.

Configure Group Policy

The following group policy settings prevent the error message "*Error occurred while creating task. Exception: A specified logon session does not exist. It may already have been terminated. (Exception from HRESULT: 0x80070520)".*

Follow these steps:

1. Run the following command as administrator: **gpedit.msc**The Local Group Policy Editor window opens.
2. Go to Local Computer Policy, Computer Configuration, Administrative Templates, System, User Profiles.
3. Enable "**Do not forcefully unload the user's registry at user logoff**".
4. Go to Local Computer Policy, Computer Configuration, Windows Setting, Security Settings, Local Policies, Security Options.
5. Disable "**Network access: Do not allow storage of passwords and credentials for network authentication**".
6. Click Apply and then click OK.

7. Restart Windows for the changes to take effect.

Configure ALM Integration

You must run several configuration editors to enable the integration with your HP ALM version. Rerun these configuration editors every time you reinstall the service. For more information, see Verify Prerequisites for ALM Integration.

To identify your HP ALM version and patch accurately, click Help, About HP ALM Software in HP ALM. The version and patch numbers are displayed in the Build column.

The following configuration editors are included with your Test Data Manager installation:

Configure HP ALM Service

Follow these steps:

1. Create a local user account that the Task Scheduler can use to run the HPALMService.
   a. Name the account, for example, `ALMService_user` .
   b. Make the account a member of the **Administrators** group.
   c. Provide a strong password.
2. Run the file GTHPALMService\GTHPALMService_ConfigEditor\ALMServiceConfigEditor.exe.
3. Under the **Service Configuration** tab, specify the following, and save the changes:
   – **ALM URL**Define the ALM Instance URL. Example: `http://alm.you.com:8080/qcbin/`
   – **Service Address**Specify the host and port of your ALM instance. Example: `http://*:8095/`
   – **Temp Directory Path**Defines the path where Test Data on Demand Portal drops attachments to be used by the Service.
   Default: `C:\Grid-Tools\TDoD\TDOD_WebUI\file_handler\`
   – **ALM Version**Select the exact HP ALM version and patch that you are using.
4. Under the Connection String and Log Configuration tabs, specify the following and save the changes:
   – **Database**Specify the database type. Available options are SQL Server or Oracle.
   – **Data Source**(SQL Server only) Specify the fully qualified SQL server instance name.
   (Oracle only) Specify TNS_ALIAS that is defined in tnsnames.ora.
   – **Integrated Security**Enable this if the SQL Server is configured to run on Windows Authentication mode.
   – **User ID and Password**Login ID and password of the database user who connects to the repository.
   – Click **Copy to log** (or **Copy to repo**, respectively) to copy the configuration from the Connection String tab to the Log Configuration tab. Click **Verify** to check the connection.
5. Under the Status tab, do the following:
   a. Click **Check group policy status** to verify your group policy settings.
   b. Click **Create task** and enter the user name and password for your ALMService_user.
   c. Click **Enable Task** and click save.
   d. Click **Start Service** to start the HP ALM service.
   e. Go to the ALM Service log to verify that the service has started.
   `C:\Grid-Tools\GTHPALMService\logs`
   This action schedules and runs a "**Start ALM Service**" task in the Windows Task Scheduler. The scheduled task runs using the specified username and password of your ALMService_user.

Problem:

You see the following error message when you create a task:

```
Error occurred while creating task. Exception: A specified logon session does not exist. It may already have
 been terminated. (Exception from HRESULT: 0x80070520)
```

Solution:

Ensure that you have disabled the group policy "Network access: Do not allow storage of passwords and credentials for network authentication". For more information, see Verify Prerequisites for ALM Integration.

Configure the TDoD Service

Follow these steps:

1. Run the file TDoD\TDoD_ConfigEditor\TDoDConfigEditor.exe.
2. Under the **Configure Service, Advanced Settings** tab, specify the following and save:
   - **Service Base Address**
     Define the TDoD service base address and port. Example: http://localhost:8090/
   - Do not enable **Disable Plain Text Auth**.
   - Click **Verify** to check the connection.
3. Under the **Configure Portal, Settings** tab, specify the following and save:
   - **Service URL**
     Define the TDoD service URL. Example: http://localhost:8090/GTService
   - **ALM Service URL**
     Define the HP ALM service URL. Example: http://alm.you.com:8095/ALMService
   - Click **Verify** to check the connection.
4. Under the **Configure Service, Connection string** tab, specify the following and save:
   - **Database**Specify the database type. Available options are SQL Server and Oracle.
   - **Data Source**(SQL Server only) Specify the fully qualified SQL server instance name.
     (Oracle only) Specify TNS_ALIAS that is defined in tnsnames.ora.
   - **Integrated Security**Enable this if the SQL Server is configured to run on Windows Authentication mode.
   - **User ID and Password**Login ID and password of the user who connects to the repository.
   - Start the service.
5. Open the Datamaker web interface, log on, and verify that TDoD works.

   TIP

   Open the command line and use netstat to check which service is running on which port.
   Example: `netstat -ano | find ":8070"`

Configure the Remote Publish Engine

Follow these steps:

1. Run the file RemotePublish\RemotePublish_ConfigEditor\RemotePublishConfiguration.exe.
2. Under the **Configure** tab, specify the following and save:
   - **Email protocol** and **group email address**
     Define the group email address which will receive job notifications.
   - **Thread Name**
     Define one thread name.
3. Under the **Configure, Connection String** tab, specify the following and save.
   - **Database**Specify the database type. Available options are SQL Server and Oracle.
   - **Data Source**(SQL Server only) Specify the fully qualified SQL server instance name.
     (Oracle only) Specify TNS_ALIAS that is defined in tnsnames.ora.
   - **Integrated Security**Enable this if the SQL Server is configured to run on Windows Authentication mode.
   - **User ID and Password**Login ID and password of the user who connects to the repository.

4. Under the **Configure, Job Executors** tab, specify the following, click verify, and save.
   - `name:datamaker, jobName: PUBLISH`
   - `name:testmatch, jobName: TESTMATCH`
   - `name:almworker, jobName: ALM`
   - `name:rallyworker, jobName: RALLY`
   - `name:groupjobexecutor: jobName:GROUP (combines ALM and TESTMATCH)`
5. Under the **Status** tab:
   a. Verify that c:\Grid-Tools\GTDataMaker\rep.xml and tnsnames.ora (for Oracle only), and %APP_DATA%\Grid-Tools \GTDataMaker\rep.xml (?) exist and are configured. The .ora file must point to repository.
   b. Start the service.

Tip: If remote publishing fails, go to c:\Grid-Tools\RemotePublish\ProcessedFiles and look into the zip files for more details.

Configure the Group Job Executor

Follow these steps:

1. Run the file GTGroupJobProcessor\GroupJobExecutor_ConfigEditor\GroupJobExecutor_ConfigEditor.exe.
2. Under the **Configure URL** tab:
   a. Define the **TDoT Service** address and port. Example: *http://localhost:8090/GTService*
   b. Click **Verify** to check the connection.
3. Click Save.

Configure HP ALM Batch

Follow these steps:

1. Run the file ALMBatch\ALMBatch_ConfigEditor\ALMBatchConfigEditor.exe
2. Select the Configuration file using the Browse button.
3. Under the **Configure URL** tab, specify the following:
   - ALM URL
   - TDoD Service Address
   - ALM Version

Configure Rally Batch

Rally provides two authentication methods. Either, Rally users provide their user name and password here. Or they create an API key on their Rally Accounts page and provide the API key here.

Follow these steps:

1. Run the file GTRallyBatch\RallyBatch_ConfigEditor\RallyBatchConfigEditor.exe.
2. Under the **Configure URL** tab, specify the following and save:
   a. **TDoD Service Address**
   Example: http://localhost:8090/GTService
   b. **Rally URL**Example:
   c. **Rally Authentication**
   Provide either a Rally user name and password, or the user's API key.
   d. Click **Verify** to check the connection.
3. Under the **Rally Customization** tab:
   a. Select workspace and project to create requirement links.
   b. Click Update.

Configure CA TDM Portal Application Properties

When you run test match request using CA TDM Portal Self-Service Catalog, the portal performs the test match and attaches the results to the corresponding test case in HP ALM. To run the ALM job from the TDM Portal, you must modify the application.properties file.

Follow these steps:

1. Open the *application.properties* file typically available at *C:\Program Files\CA\CA Test Data Manager Portal\conf*.
2. Identify the following parameter and set the value to "*true*".
   almTesterSelfServiceLegacyIntegration=false|true
   Default: false
3. Restart the CA TDM Portal service.

Configure Test Data Management for ALM

As a Test Data Engineer, configure Test Data Management to complete the integration with HP ALM. Integration lets the testers publish data criteria for each test case and request the data to attach to the test case.

> **NOTE**
>
> We strongly recommend that you configure the test match to use the new Tester Self Service capability in the CA TDM Portal. The Microsoft Silverlight-based Test Data on Demand (TDoD) user interface is deprecated. The TDoD functionality is now available in an improved interface within the CA TDM Portal. TDoD users can use all the same capabilities by using the Self-Service Catalog interface in the CA TDM Portal. Additionally, no license migration is required; the same TDoD licenses are valid with the CA TDM Portal functionality. For more information about how to use the CA TDM Portal as the location for testers to request data, see Configure Test Data Reservation Service. Existing users can still use the TDoD functionality, but we recommend that you start using the Self-Service Catalog functionality in the CA TDM Portal.

Create Data Sets and Data Pools

Create the following data set and data pools. these items are used to edit TCDC table, and to publish and attach the Flow.

- Data Pool of Normal type
- Data Pool of Test Match type
- Data Set of CA Agile Requirements Designer type

Select the **Available on demand** check box when you create the data sets and pools.

Create Test Match Data Pool

Create a data pool of test match type in TDM. Ceate the pool under the project that holds the data that is required for the tests. The test match data pool holds the TCDC table where the data criteria is published. The data criteria is specified by the tester. To perform the test match, associate the test match data pool with appropriate test mart .

- After you create the data pool, run the test match at least once successfully.
- In the **Default Publish** tab, associate the latest successful run of test match data pool properties.

Create Publish Data Pool

Create a normal type data pool in TDM under the same project as the test match data pool. The new data pool is used to publish the data to the TESTCASE_DATA_CRITERIA (TCDC) table of the test match data pool.

Follow these steps:

1. Create a data pool. Ensure the **Available on demand** check box is selected.

2. Right-click and select **Edit Data**.
3. Double-click the TESTCASE_DATA_CRITERIA table.
4. Edit the following mandatory columns with the values that are specified and save the changes:
   Note: The following values are examples for three test conditions. You can edit the number of test conditions as necessary.
   - **Test Name**
     Value: ~testname~
     Note: Case sensitive
   - **Test Active**
     Value: Y
   - **Test Repeater**
     Value: ~repeat~
   - **Test Conditions**
     Value: ~condition1~
   - **Hpalm Path**
     Value: td://~almp~.~almd~.**<alm host>**/qcbin/%5bAnyModule%5d?EntityType=ITest&EntityID=~almoid~
     <alm host> is the HP ALM server address.
5. Create DataSet substitutions for the preceding variables as shown in the following screen capture:

Variable Name	Default Value	Variable Descripti
almd	ALMD	ALM Domain
almoid	1	almoid
almp	ALMP	ALM Project
cond1_val	@execsql(R,select test_conditions from FNV_608_2385 where test_name='~testname~')@	Test condition
condition1	~cond1_val~	Condition for the
repeat	@execsql(R,select test_repeater from FNV_608_2385 where test_name='~testname~')@	Number of data it
testname	Default test case	Test case name

   You can replace the default values as required to your enterprise environment.
6. In the preceding example, the table name **FNV_608_2385** represents the TESTCASE_DATA_CRITERIA of the Test Match data pool. To correct the value on your environment, do the following steps :
   a. Open the TESTCASE_DATA_CRITERIA (TCDC) of the test match data pool. Right click the **TESTCASE_DATA_CRITERIA** tab.
   b. Select **Copy External View Name 'XXXX'.**
   c. Use the copied value in place of FNV_608_2385.
7. Perform publish to test match data pool. Ensure that the successful run is associated in the **Default Publish** tab of publish data pool properties.

Create CA Agile Requirements Designer Data Set

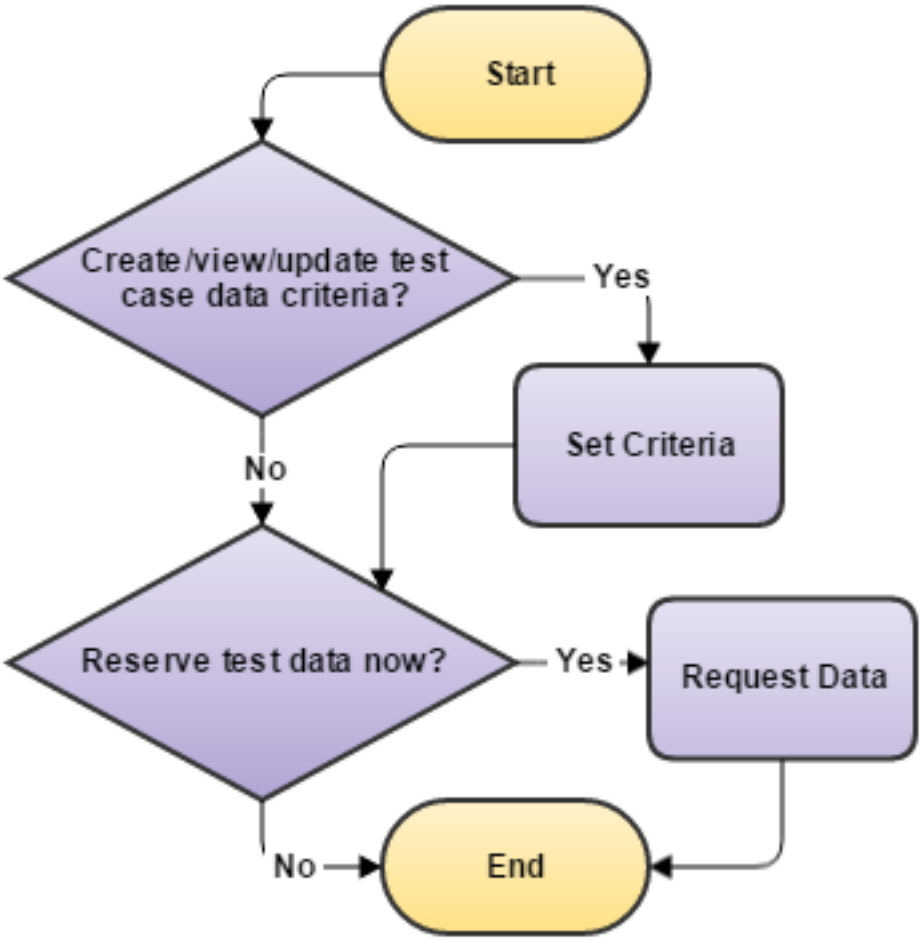
Create a data set of CA Agile Requirements Designer type in TDM under the same project as test match data pool. The new data set is used to associate the Flow.

Create Flow

CA Agile Requirements Designer lets you design the visual and active flow chart and various TDM actions. When you expose the designed flow to TDoD, run the flow to perform the defined actions. For example, you can set criteria and request data for a specific test case. Run the flow that is designed with respective actions to perform in Test Data Management.

The following diagram shows the recommended design of the flow:

Figure 41: ARD Flow



When you create the flow for the defined process block, specify the following values:

Set Criteria

1. Double-click the **Set Criteria** process block, go to the **Make System Data** tab, and click **Set Publish**.
2. Associate to the publish data pool and configure variable options as shown in the following screen capture:

			Variable	Use in TDoD	Fixed	Resolve	Value
1			condition1	☑	☐	☑	~cond1_val~
2			repeat	☑	☐	☑	@execsql(R,select test_repeater from FNV_608_2385 where test_name='~testname~')@
3			testname	☐	☐	☐	Default test case
4			almp	☐	☐	☐	ALMP
5			almd	☐	☐	☐	ALMD
6			almoid	☐	☐	☐	1
7			cond1_val	☐	☐	☐	@execsql(R,select test_conditions from FNV_608_2385 where test_name='~testname~')@

Request Data

1. Double-click the **Request Data** process block, go to the **Find System Data** tab, and click **Set Test Match**.
2. Select the test match data pool to attach, and click **OK**.
3. Go to TDoD **Options** and select the required test match options to expose to TDoD.
4. Modify the following test match options names as required:
   – **Clear Down Prior Matches** (Default)
   Recommended: Release previous data reservations?
   – **Allow Partial Matches** (Default)
   Recommended: Allow partial data reservations?
5. Go to **Details** tab and specify a value in **ALM** field. Following are the valid values:
   – **ATTACH_ALL** or **ATTACH_ALL_TM_RESULTS**
   Performs the synchronization (CSV upload to HP-ALM) of all the tests.
   – **ATTACH_TM_RESULTS**
   Performs the synchonization (CSV upload to HP-ALM) for a specific test.

(Optional) Specify TEST_NAME Variable

You can create a TEST_NAME variable to specify the test case name to execute the test match and get the data only for a specific test case.

1. Go to menu bar and click Tools, Properties.
   The Properties dialog opens.
2. Click the Variables tab.
   The Variables dialog opens.
3. Click New Variable and specify the following:
   – **Name**
   Specifies variable name. Enter TEST_NAME in this field. The variable name is case sensitive and to be entered in Upper Case only.
   – **Description**
   Specifies the variable description. Enter the brief description that explains the variable.
   – **Default Value**
   Specifies the default value to show when the flow is executed in TDoD. Enter a default value of the variable in this field.
4. Click **Save**.
   The global variable TEST_NAME is added to the flow. When the Test Engineer executes this flow from CA TDM Self Service Catalog, the flow presents Test Name field under Global Variables. Tester can enter a specific test case name in this field to execute the test match and get the relevant results, only for the specified test case.

Save to Repository

- Go to the File menu, click **Save to repository**, select **CA Agile Requirements Designer Data Set**, and click **Save**.
  For more information about how to create Flows, see the CA ARD documentation.

Expose Flow to TDoD

Associate the Flow with TDoD. This action lets testers use the flow in the CA TDM Portal Self-Service Catalog to perform Set Criteria and Request Data actions.

Follow these steps:

1. Navigate to the project where Flow is saved.
2. Right-click the Flow and click **Expose to TDoD**.
   The flow is now available in the CA TDM Portal Self-Service Catalog.

Customize HP ALM

You must customize the HP ALM to work the integration with CA Test Data Management. Review the below procedures to customize the HP ALM as per your enterprise requirements. You must have the administrator privileges to HP ALM project instance to perform the below tasks.

Adding ALM Customization Script

Follow these steps:

1. Login to HP ALM and start the project session to customize.
2. Go to Tools menu and click Customize.
3. Go to Workflow and click Script Editor.
4. Under the Script Editor tab, go to Workflow Scripts, Project Scripts, Common Scripts.
5. Download the file VBScriptTDMALMARDIntegration.txt and paste the code snippet into the script editor. Replace the iURL string "demo-win8-vm" with the URL for your TDM Server.
   Note: Ensure that the communication from the HP ALM Server to the TDM Server through your firewalls and networks is possible over port 80. If HP ALM is listening on other port then provide the iURL appropriately.

Configuring HP ALM Site Administration

You must configure the HP ALM Site Administration to add tdod_url and tdod_url_<project> parameters.
The *tdod_url* parameter is used to integrate the HP ALM at application level. If you want to integrate any particular project within HP ALM application, then you must add the *tdod_url_<project>* parameter. If you want to configure multiple projects within one HP ALM application then you must configure the *tdod_url_<project>* parameter for each of the project separately.

TIP

We strongly recommend configuring the test match to use the new Tester Self Service capability in the CA TDM Portal UI.

Follow these steps:

- Go to HP ALM Site Administration.
- Go to Site Configuration tab.
- Click New, and create the following two parameters:

1. **tdod_url**
   Specifies that the configuration is applicable to all the projects in HP ALM. Enter the following values in the New Parameter dialog:

Field	Value
Parameter	tdod_url
Value	To configure Tester Self-Service that you can access using the CA TDM Portal, enter **http://<hostname>:<port>/TestDataManager/main.html#/tdod?** where <hostname> is the CA TDM Portal installed server and <port> is the port number on which the CA TDM Portal service is running. To configure Test Data On-Demand, enter **http://<hostname>/Default.aspx/?** where <hostname> is the TDoD Installed Server.
Description	Enter description for the parameter.

2. **tdod_url_<ProjectName>**

Specifies that the configuration is applicable only to a specific project that is configured in HP ALM. If you want to configure multiple projects specify the parameter for each project separately. Enter the following values in the New Parameter dialog:

Field	Value
Parameter	tdod_url_<ProjectName> Where <ProjectName> is the name of HP ALM project that you want to configure.
Value	To configure Tester Self-Service that you can access using the CA TDM Portal, enter **http://<hostname>:<port>/TestDataManager/main.html#/tdod** where <hostname> is the CA TDM Portal installed server and <port> is the port number on which the CA TDM Portal service is running. To configure Test Data On-Demand, enter **http://<hostname>/Default.aspx** where <hostname> is the TDoD Installed Server.
Description	Enter description for the parameter.

Configuring HP ALM Test Plan

Follow these steps:

1. Go to Tools, Customize, Workflow, and click Script Editor.
2. Under the Toolbar Button Editor, select Test Plan from the Command Bar drop-down list.
3. Select TestPlan_AttachAgDFlow, specify the following and then click Apply:
   Caption
   TestPlan_AttachAgDFlow
   Hint
   Attach to Flow
   Action Name
   TestPlan_AttachAgDFlow
   Image
   Select the image named "124" from the library.
4. Select TestPlan_ExecuteFlow, specify the following and then click Apply:
   Caption
   TestPlan_ExecuteFlow
   Hint
   Execute Flow
   Action Name
   TestPlan_ExecuteFlow
   Image
   Select the image named "268" from the library.
5. Select TestPlan_Resolve, specify the following and then click Apply:
   Caption
   TestPlan_Resolve
   Hint
   Resolve design step description and unresolved test data
   Action Name
   TestPlan_Resolve
   Image
   Select the image named "75" from the library.
6. Save the changes applied.

Extending Test Step Entity

Test Step entity requires to be extended to support a new column ‘Resolved Description’.

Follow these steps:

1. Go to Tools, Customize, Project Entities.
2. Expand Test Step from the Project Entities, select User Fields.
3. Add New Memo Field and specify the following:
   Label
   Enter the text "Resolved Description".
4. Click Save.
5. Click Return.
6. Select Major change and click OK.
   The change is available when you connect to the project next time.

Configure CA TDM Portal with HP ALM Service Account

As a Test Data Engineer you can configure the CA TDM Portal with HP ALM Service Account Credentials, so that the test engineer need not provide their user credentials every time they run the TDoD Flows for Attach, Execute and Resolve variables.

All the test engineers who have access to the CA TDM Portal can submit their Attach, Execute and Resolve variables without the need of providing user credentials.

Follow these steps:

1. Access the CA TDM Portal.
2. Expand the **Configuration** option and click **ALM Credentials** in the left hand menu .
3. Verify that the specified **HP ALM URL** is correct.
   Notes:

- – This is the ALM Instance URL that you specified in the HP ALM Service configuration. For more information, see Configure HP ALM Service.
  – The HP ALM Service must be running on the same server and port that you have specified during CA TDM Portal installation. For more information, see Install CA TDM Portal.

1. Specify the **User Name** and **Password** that you want to use commonly for all the test engineers who have the access to TDM Portal.
2. Click Test. Tests passed successfully message should appear.
3. Click Save.
   The User Name and Password is saved.
4. (Optional) You can edit the User Name and Password anytime you prefer to change the credentials. Repeat the steps 1 to 5 to modify the User Name and Password anytime you prefer to change the credentials.

You can have only one HP ALM instance per repository instance. All users of a repository can have access to the same ALM instance.

If the HP ALM Service Account is not configured, the testers can perform the said operations using their individual HP ALM Credentials.

Test Matching Rally Integration

Tests generally use various input parameter types, such as valid inputs, invalid inputs, normal case, or edge cases. You can run tests under different inputs to see how your system works under different conditions. Testers can also look at the

behavior of the same product for different customers. In that case, execute the same tests multiple times with different sets of inputs.

CA Test Data Manager (TDM) enhances the quality of your production data. TDM fill gaps in your coverage that is based on the optimal minimum set of test cases for requirements. TDM uses innovative Test Matching functionality to match data to the test cases based on the criteria that are specified. The matched data is then exported to a CSV file attached directly to the CA Rally test cases.The Test Matching functionality lets testers can run the tests with appropriate data.

The following list shows the user roles and related procedures that are used to integrate CA TDM with Rally:

Rally Administrator

- Configures custom fields in Rally

Test Data Engineer

- Performs Rally batch configuration
- Creates Data Sets and Pool sets
- Performs post-publish configuration
- Creates TCO flow
- Creates Rally project Requirement links on-demand

Tester

- Tests Requirement and Reservation

Configure Custom Fields in Rally

CA TDM integrates with CA Rally using custom web links that establish a connection between Rally and TDM. Rally administrators with workspace administrator permissions configure custom web links to provide the Rally Test Case page custom fields. The Rally administrator configures the following custom fields for each project that requires TDM integration:

- **Test Data Requirement**
  Lets you generate a Set Criteria link in the Rally Test Case Page. This link helps the testers define the test case data criteria.
- **Test Data Reservation**
  Lets you generate a Request Data link in the Rally Test Case Pag. This link helps the testers fetch test data from CA TDM and attach the data to the respective Rally test case.

  TIP

  We strongly recommend that you configure the test match to use the new Tester Self Service capability in the CA TDM Portal.

Follow these steps:

1. Log in to Rally as Workspace Administrator.
2. Click the Setup icon and then click **WORKSPACES & PROJECTS**.
3. Expand the workspace and click the project name that you want to configure.
4. Click **Fields** in the navigation menu.

5. Select Test Case from the **Type** drop-down list.
6. Click **New Field**, complete the following fields, and save your changes.

- **Name**
  Specifies the name of the custom field. Create the following custom fields one after the other:
  - **TDMTestRequirement**
    Use this name to configure the custom field that allows the testers to set the test case data criteria.
  - **TDMTestDataReservation**
    Use this name to configure the custom field that allows the testers to request the test data
- **Display Name**
  Specifies how the custom field names appear on the Rally UI
  Important: You must specify the display names exactly as provided below. The integration will fail if you deviate from or have typos in the display names.
  - **Test Data Requirement**
    Use this display name for TDMTestRequirement.
  - **Test Data Reservation**
    Use this display name for TDMTestDataReservation.
- **Type**
  Specifies the type of the link.
  Note: Select **Web Link** from this drop-down list
- **Web Link Info**
  Includes the following details for the integration details:
- **URL**
  Specifies the URL of the CA TDM Service:
  Value:
  - To configure Tester Self-Service that you can access using TDM Portal, do the following:
    - For HTTP protocol enter the following value:
      http://<hostname>:<port>/TestDataManager/main.html#/tdod/${id}
      Example: http://tdmwebcomputer:8080/TestDataManager/main.html#tdod/${id}
    - For HTTPS protocol enter the following value:
      https://<hostname>:<port>/TestDataManager/main.html#/tdod/${id}
      Example: https://tdmwebcomputer:8443/TestDataManager/main.html#tdod/${id}
      - **<hostname>**Specifies the TDM Portal installed server.
      - **<port>**Specifies the port number on which TDM Portal service is running.
  - To configure Test Data On-Demand, enter **http://<hostname>/${id}** where <hostname> is the TDoD Installed Server.
    Example: http://tdodcomputer/${id}
- **Required**
  Specifies whether the input to this new field is required to create test case.
  Note: Ensure that the value is **No**.
- **Visible**
  Specifies whether this custom field is visible at the respective project level.
  Note: Select **Yes** from the drop-down list.

The Test Data Reservation and Test Data Requirement fields are successfully created. You can see these fields on the Rally Test Case pages of the configured project.

Configure Test Data Management

As a Test Data Engineer, configure Test Data Management to complete the integration with Rally. Integration lets the testers publish data criteria for each test case and request the data to attach to the test case.

TIP

We strongly recommend that you configure the test match to use the new Tester Self Service capability in the CA TDM Portal. The Microsoft Silverlight-based Test Data on Demand (TDoD) user interface is deprecated. The TDoD functionality is now available in an improved interface within the CA TDM Portal. TDoD users can use all the same capabilities by using the Data Reservation. Existing users can still use the TDoD functionality, but we recommend that you start using the Self-Service Catalog functionality in the CA TDM Portal.

Perform Rally Batch Configuration

Follow these steps:

1. Locate and run the RallyBatchConfigEditor.exe file from the CA TDM installation location. The following is the typical installation path:
   C:\Grid-Tools\GTRallyBatch\RallyBatch_ConfigEditor
2. In the **Configure** tab, **Configure URL,** specify the following details:
   - **TDoD Service Address**
     Specifies the URL to access TDoD.
     - For HTTP protocol enter the following value:
       http://<<machinename>>:8090/GTService
     - For HTTPS protocol enter the following value:
       https://<<machinename>>:8090/GTService
   - **Rally URL**
     Specifies the URL of Rally Service.
   - **Authenticate to Rally**
     Specifies whether to authenticate the user to Rally using User Credentials or API Key. Select one of the following:
     - **Use UserName and Password**
       Specifies that the user should be authenticated using the login credentials. Enter User Name and Password to access Rally.
     - **Use API Key**
       Specifies that the user should be authenticated using API Key. Enter the API Key to access Rally
3. Click Verify to confirm the details, and click Save.

Create Data Sets and Data Pools

Create the following data set and data pools. these items are used to edit Test Case Data Criteria (TCDC) table, and to publish and attach the Flow.

- Data Pool of Normal type
- Data Pool of Test Match type
- Data Set of CA Agile Requirements Designer type

Select the **Available on demand** check box when you create the data sets and pools.

Create Test Match Data Pool

Create a data pool of test match type in TDM. Create the pool under the project that holds the data that is required for the tests. The test match data pool holds the TCDC table where the data criteria is published. The data criteria is specified by the tester. To perform the test match, associate the test match data pool with appropriate test mart .

- After you create the data pool, run the test match at least once successfully.
- In the **Default Publish** tab, associate the latest successful run of test match data pool properties.

For more information about how to create Data Pool, see Create a data group, data set, and data pool.

Create Publish Data Pool

Create a normal type data pool in TDM under the same project as the test match data pool. The new data pool is used to publish the data to the TESTCASE_DATA_CRITERIA (TCDC) table of the test match data pool.

Follow these steps:

1. Create a data pool. Ensure the **Available on demand** check box is selected.
2. Right-click and select **Edit Data**.
3. Double-click the TESTCASE_DATA_CRITERIA table.
4. Edit the following mandatory columns with the values that are specified and save the changes:
   Note: The following values are examples for three test conditions. You can edit the number of test conditions as necessary.
   - **Test Name**
     Value: ~testname~
     Note: Case sensitive
   - **Test Active**
     Value: Y
   - **Test Repeater**
     Value: ~repeat~
   - **Test Conditions**
     Value: ~condition1~
   - **Test Conditions2**
     Value: ~condition2~
   - **Test Conditions3**
     Value: ~condition3~
   - **Rally Path**
     Value:~rallyoid~
     Note: Case sensitive
   - **PJ Id**
     Value: ~PUBJOBID~
     Note: Case sensitive
5. Create DataSet substitutions for the preceding variables as shown in the following screen capture:

Variable Name	Default Value	Variable Description	Validatio	List	Help message	Error message	Display type
cond1_val	@execsql(R,select test_conditions from FNV_601_2358 where test_name='~testname~']@	cond1_val					General
cond2_val	@execsql(R,select test_conditions2 from FNV_601_2358 where test_name='~testname~']@	cond2_val					General
cond3_val	@execsql(R,select test_conditions3 from FNV_601_2358 where test_name='~testname~']@	cond3_val					General
condition1	~cond1_val~	First condition for the data criteria					General
condition2	~cond2_val~	Second condition for the data criteria					General
condition3	~cond3_val~	Third condition for the data criteria					General
rallyoid	1	rallyoid					General
repeat	@execsql(R,select test_repeater from FNV_601_2358 where test_name='~testname~']@	Number of data items to be reserved					General
testname	Test1	test name					General

6. In the preceding example, the table name **FNV_601_2358** represents the TESTCASE_DATA_CRITERIA of the Test Match data pool. To correct the value on your environment, do the following steps :
   a. Open the TESTCASE_DATA_CRITERIA (TCDC) of the test match data pool. Right click the **TESTCASE_DATA_CRITERIA** tab.
   b. Select **Copy External View Name 'XXXX'.**
   c. Use the copied value in place of FNV_601_2358.

7. Perform publish to test match data pool. Ensure that the successful run is associated in the **Default Publish** tab of publish data pool properties.

create CA Agile Requirements Designer Data Set

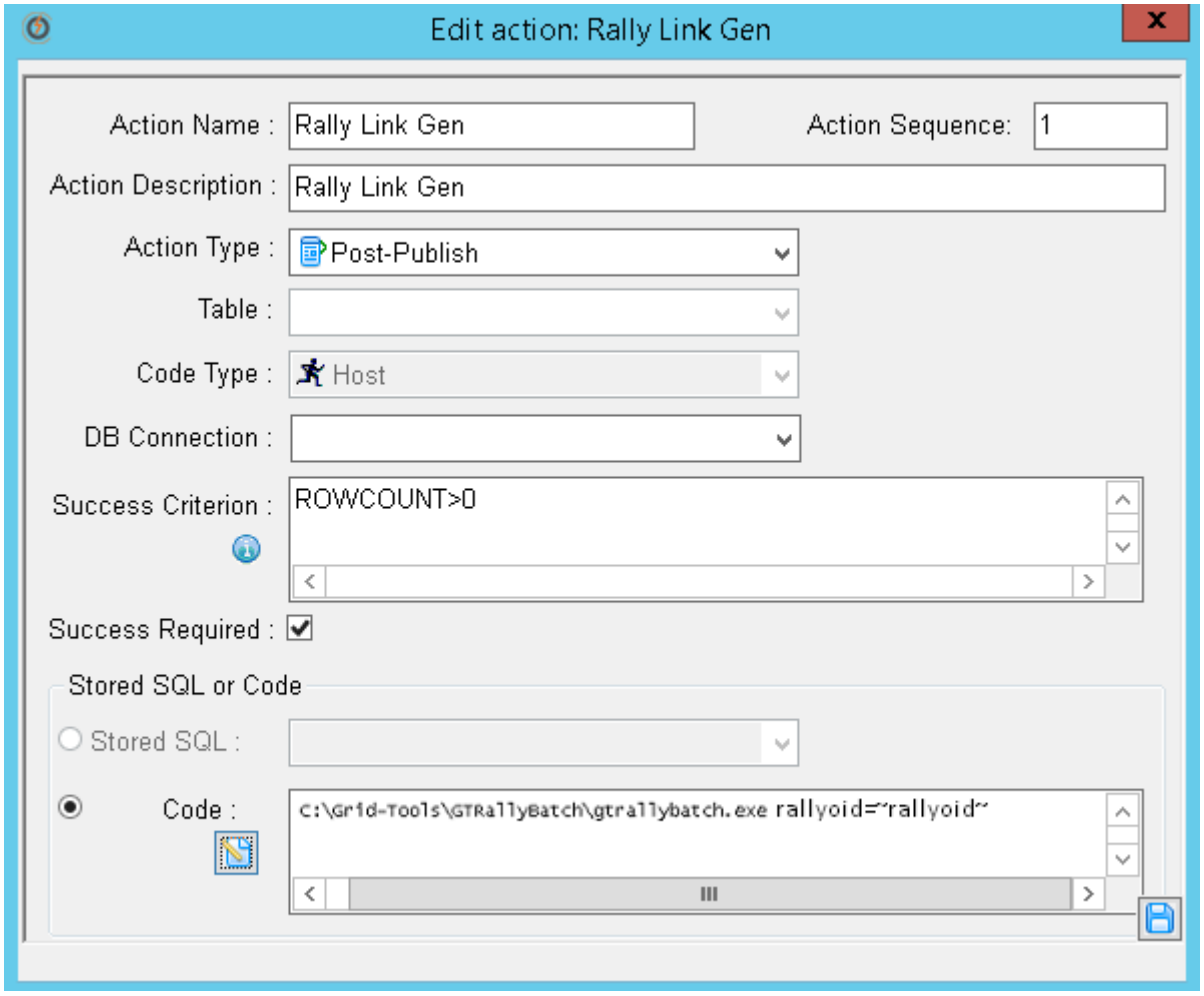
Create a data set of CA Agile Requirements Designer type in TDM under the same project as test match data pool. The new data set is used to associate the Flow.

Perform Post Publish Configuration

Configure post publish action in the publish data pool to generate the Request Data link.

Follow these steps:

1. Enable the execution of host actions.
   To do this, set the **enableHostActions** parameter to `true` in the `application.properties` file (located at `C:\Program Files\CA\CA Test Data Manager Portal\conf\` in a standard installation).
   For more information see, Enable HOST Actions.
2. Go to the **Publish Data Pool** and right click.
3. Click **Maintain Actions**.
4. Create an action as shown in the following screen:



5. Specify the values and save. Confirm that the following fields have the specifies values as specified

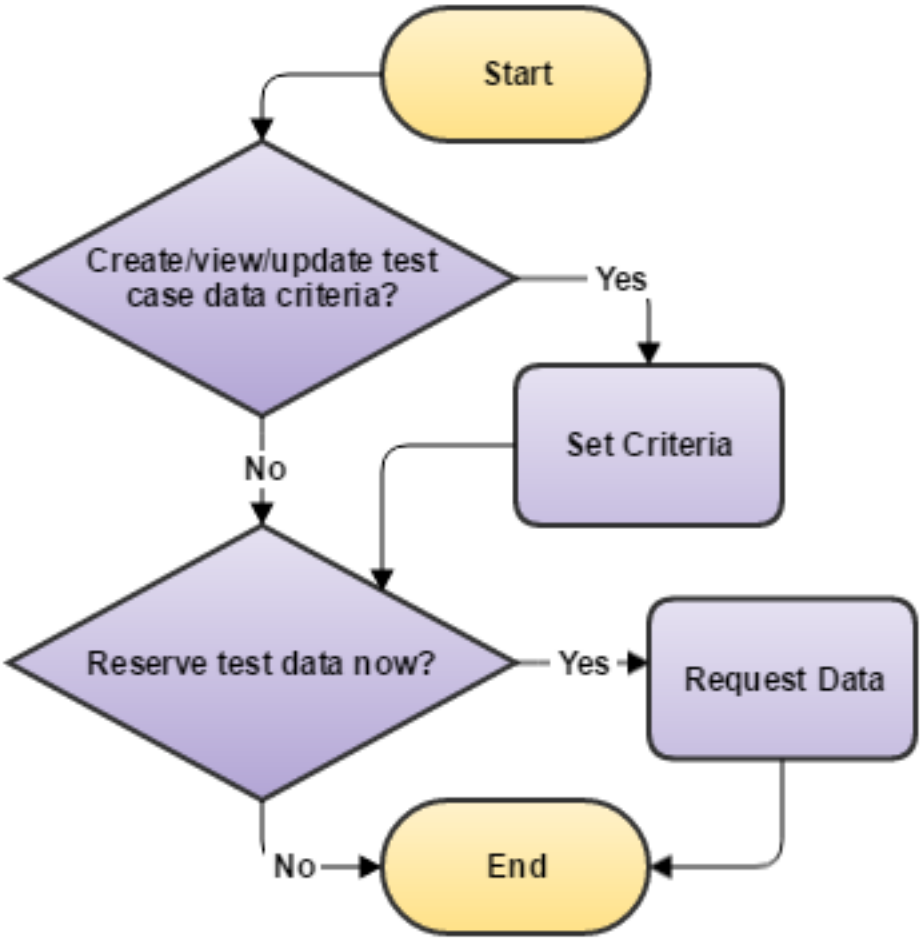
- **Action Type**
  Select **Post-Publish** from the drop-down.
- **Success Criterion**
  Enter the exact case for "ROWCOUNT>0".
- **Code**
  Enter gtrallybatch.exe path as indicated in your installation. Ensure that you use the exact case for "rallyoid=~rallyoid~".

Create Flow

CA Agile Requirements Designer lets you design the visual and active flow chart and various TDM actions. When you expose the designed flow to TDoD, run the flow to perform the defined actions. For example, you can set criteria and request data for a specific test case. Run the flow that is designed with respective actions to perform in Test Data Management.

The following diagram shows the recommended design of the flow:

Figure 42: Recommended Design Flow



When you create the flow for the defined process block, specify the following values:

Set Criteria

1. Double-click the **Set Criteria** process block, go to the **Make System Data** tab, and click **Set Publish**.
2. Associate to the publish data pool and configure variable options as shown in the following screen capture:

			Variable	Use in TDoD	Fixed	Resolve	Value
1			condition1	☑	☐	☑	~cond1_val~
2			condition2	☑	☐	☑	~cond2_val~
3			condition3	☑	☐	☑	~cond3_val~
4			rallyoid	☐	☐	☐	1
5			repeat	☑	☐	☑	@execsql(R,select test_repeater from FNV_10101_7573 where test_name='~testname~')@
6			testname	☐	☐	☐	Test1
7			cond1_val	☐	☐	☐	@execsql(R,select test_conditions from FNV_601_2358 where test_name='~testname~')@
8			cond2_val	☐	☐	☐	@execsql(R,select test_conditions2 from FNV_601_2358 where test_name='~testname~')@
9			cond3_val	☐	☐	☐	@execsql(R,select test_conditions3 from FNV_601_2358 where test_name='~testname~')@

Request Data

1. Double-click the **Request Data** process block, go to the **Find System Data** tab, and click **Set Test Match**.
2. Select the test match data pool to attach, and click **OK**.
3. Go to TDoD **Options** and select the required test match options to expose to TDoD.
4. Modify the following test match options names as required:
   - **Clear Down Prior Matches** (Default)
     Recommended: Release previous data reservations?
   - **Allow Partial Matches** (Default)
     Recommended: Allow partial data reservations?
5. In the **Details** tab, enter the value ATTACH_TO_RALLY in the **ALM** field.

Save to Repository

- Go to the File menu, click **Save to repository**, select **CA Agile Requirements Designer Data Se**t, and click **Save**.
  For more information about how to create Flows, see the CA Agile Requirements Designer documentation.

Expose Flow to TDoD

Associate the Flow with TDoD. This action lets testers use the flow in the CA TDM Portal Self-Service Catalog to perform Set Criteria and Request Data actions.

Follow these steps:

1. Navigate to the project where Flow is saved.
2. Right-click the Flow and click **Expose to TDoD**.
   The flow is now available in the CA TDM Portal Self-Service Catalog.

Create Requirement Links On-Demand for Rally Project

Run the on-demand configuration. This action generates the web links in the Rally test case page.

1. Locate and run the RallyBatchConfigEditor.exe file from the CA TDM installation location. The following path is a typical installation path:
   C:\Grid-Tools\GTRallyBatch\RallyBatch_ConfigEditor
2. Go to the **Rally Customization** tab.

3. To customize Rally TDM fields, select the following items and click **Update**.

- – Select Workspace
  – Select Project

The Set Criteria link in the **Test Data Requirement** field in every Test Case of the Rally project is created. The Request Data link in the Test Data Reservation field is enabled. The field is enabled only after the test case data criteria is published using Set Criteria link.

Test Data Requirement and Reservation

As a tester, you can perform the Set Criteria and can Request Data activities directly from the Rally Test Case. The set Criteria link in the Test Data Requirement field is enabled after Rally is customized to create a requirement link. To create a requirement link, run RallyBatchConfigEditor.exe for the Rally project. You can also manually enable the Set Criteria link.

Request Data is automatically enabled after the criteria is set in CA TDM.

Enable Set Criteria Link Manually

If the Set Criteria link is not enabled in the Test Data Requirement field, you can manually enable the link.

Follow these steps:

1. Select the test case and click to **Edit**
2. Go to Test Data Requirement field and complete the following fields:

- **Link Label Value**: Set Criteria.
- **ID**
  Value: *?mode=attach&rallyoid=<OID of the Test Case>&testname=<Formatted ID of the Test Case>*
  Example: ?mode=attach&rallyoid=**12345123451**&testname=**TC101**

Follow these steps:

Note: The UI labels that are used in the following steps refer to the recommended labels. The labels are shown in the "Create Publish Data Pool" and "Create Flow" sections.

1. Go to Rally Test Case page and click the Set Criteria link in **Test Data Requirement** field.
   Test Data Management login screen opens.
2. Provide User Name, Password, and click **Login**.
   TDM Test Data on Demand launches and displays the available flows.
3. Select an appropriate flow to attach with the test case.
4. Click **Start** and follow the activity flow.
5. Click **Yes** for **Do you want to create/view/update test case data criteria?**.
6. Enter the values for each condition under **Set Criteria** and click **Next**.
7. Click **No** for **Do you want to reserve data now?**.
8. Click **Submit**, and click **Go** in the subsequent dialogs.
   After data is published in TDM, the Request Test Data button is enabled in Rally Test Case page, .

Request Test Data

TDM publishes the Test Case Data Criteria based on the design flow. The product then runs the test match and identifies the right data for the respective test case. Use the **Request Data** link to request data from the Rally test case page.

Follow these steps:

1. Go to Rally Test Case page and click the **Request Dat**a link in the **Test Data Reservation** field.

2. Provide user name and password, and click **Login**.
3. Click **Start** and follow the flow of activities
4. Click **No** for **Do you want to create/view/update test case data criteria?**.
5. Click **Yes** for **Do you want to reserve data now?** and specify the required values.
6. Click **Submit** and click **OK** in the subsequent dialogs.
   The test data is attached to the respective test case in CSV file format.

Verify Test Data

TDM performs the test match according to the test case data criteria. TSM the attaches the required test data for the respective test case in CSV file format. You can use this test data for your testing.

Follow these steps:

1. Go to the Rally Test Case page for which you have requested the test data.
2. Find the CSV file that is attached to the test case.
3. Click the file to find the contents and verify that the data is extracted per the specified criteria.
   Note: The attachment cannot exceed 5 MB in size. If the file exceeds the specified size, system does not attach the file to the test case.

Execute HPALM and Rally Jobs from TDM Portal

In TDM Portal Self Service Catalog (Tester Self Service) when you execute a flow that contains the HP-ALM/Rally integration details, it creates and performs the test match job. Though the flow contains the HP-ALM/Rally integration details, when you execute from TDM Portal it does not create the HP-ALM/Rally batch jobs. You needed to invoke the HP-ALM/Rally integration link only from HP-ALM/Rally UI.

A new property almTesterSelfServiceLegacyIntegration is introduced to enable or disable running HP-ALM/Rally batch jobs from TDM Portal Self Service Catalog. The default value of the option is disabled. To enable the batch jobs creation from Tester Self Service directly, do the following:

1. Go to application.properties file typically available at:
   C:\Program Files\CA\CA Test Data Manager Portal\conf
2. Locate the following property in the file:
   almTesterSelfServiceLegacyIntegration
3. Change the property value to True and save the file.
4. Restart the TDM service.

Now the flow that contains HP-ALM/Rally integration details should create both Test Match and HP-ALM/Rally jobs when you run the flow from TDM Portal Self Service Catalog.

Enable Self Service Catalog Forms for Testers

When you create flows in CA Agile Requirements Designer, you save the ARD flows associating them to the available projects in CA Test Data Manager. These flows are then managed from CA TDM. You can expose the flows to Tester Self Service and execute to perform various actions like Data Generation, Test Match and Data Reservation.

Follow these steps:

1. Access the CA TDM Portal.
2. Select a project and version from the Project drop-down list available in the portal header. Optionally, click the gear icon (next to Project drop-down in the portal header) to search for a specific project. If you want to create a new project, see Create and Edit Projects.
3. Click Self Service Flows in the left .

The Self Service Flows page opens and lists the existing ARD Flows created and saved under the selected project and version.

4. Identify the flow you want to expose and click on the row corresponds the flow. Use the search feature to find a specific flow in the list. You can search for the flows by name or description.
   The Edit Flow page opens.
5. Click the Show in Self Service Catalog check-box and click Save.
   A message appears after successfully exposing the flow to Self Service Catalog.
   Note: If you want to hide the flow, clear the Show in Self Service Catalog check-box and click Save.
6. Review the Self Service Flows page to verify that the Exposed column shows Yes in the row corresponds the respective flow.
   You have successfully exposed the flow to CA TDM Tester Self Service.
   Note: If you want to delete a flow, click the cross (X) icon in the row that corresponds to the flow that you want to delete. Click Delete in the confirmation dialog.

The test engineers can now see the flow in the Self Service Catalog and execute the flow based on their user permissions.

Show Repeat Count in Self Service Catalog Forms

You can show or hide the Repeat Count field in Self Service Catalog. In the Test Data Manager 3.8 release the repeat count field was shown in Self Service Catalog by default and not allowed to hide. When shown, you can specify the Repeat Count for data publish requests executed from Self Service Catalog. When hidden, the repeat count is used based on the selected configuration. In the Test Data Manager 4.0 release, the default is to hide the Repeat Count field.

Follow these steps:

1. Identify the Self Service Flow in which you want to show the Repeat Count field.
2. Open the identified Self Service Flow using CA Agile Requirements Designer.
3. Select the Publish block and double click.
4. In the Publish dialog, go to Test Data tab and click the Add Variable button.
5. Specify the following values in Add Variable dialog:
   Name
   Enter the name of variable. The variable name must be SHOW_REPEAT and is case sensitive.
   Description
   Enter a brief description for the variable.
   Type
   Select the variable type as "Boolean" from the drop-down list.
   True/False
   Select the check box to show the Repeat Count field in Self Service Flow. When the check box is selected, shows the value as "true". When the check box is not selected, shows the value as "false". True indicates that the Self Service Flow shows the Repeat Count field. False indicates that the Self Service Flow hides the Repeat Count field.
6. Click OK and click Save.
7. Go to File menu and Save the flow to Repository.
8. Go to CA TDM Portal and execute the respective Self Service Flow to specify the Repeat Count.
   The Repeat Count field is shown or hidden in the Self Service Catalog as specified for SHOW_REPEAT variable.
   Notes:

- – By default the Repeat count field shows the value specified in the Default Publish configuration of the respective data pool in TDM Datamaker. If the Default Publish includes an expression as the Repeat value, the Repeat count field in Self Service Catalog shows the resolved value of the respective expression.
  – If the Repeat count has dependency on any variables then the repeat count value changes based on the modifications you do to the dependent variables.
  – In TDM Web while running publish flow, if you change the default publish configuration using drop down then repeat count also changes accordingly.
  – When you execute a flow from TDM Portal Self Service Catalog, every process block in the flow that is designed for publish job shows the Repeat Count field.
  – If Iteration variable is specified to any process block, then it shows only the Iterations field.
  – Where there are multiple process blocks in a flow each pointing to a different Generator (data pool), then every process block shows the Repeat Count field, if Iterations variable is not assigned. If assigned, it shows only the Iterations field.
  – Where there are multiple process blocks pointing to same Generator (data pool), only the first process block shows the Repeat Count field, if Iterations variable is not assigned to that Generator. If assigned, it shows only the Iterations field. In the subsequent process blocks the Repeat Count or Iterations field is disabled.

Enabling Iteration Count Variable in Self Service Catalog Forms

As a Test Data Engineer you can create a CA Agile Requirements Designer flow that enables testers to specify the iteration count for data generation. While the testers execute these flows, the iteration count that they specify in the flow supersedes the default publish iteration count specified in the Datamaker.

Follow these steps:

1. Launch Datamaker and create a publish data pool.
2. Create a variable with the name "Iterations" for the respective data pool.
3. Configure a post publish action in the respective publish data pool with the following values:
   – Action Name: Normal
   – Action Type: Post-publish
   – Code Type: File
   – Code: ~Iterations~
4. Save the post publish action and execute.
5. Go to CA Agile Requirements Designer and create a visual flow and add a process block.
   a. Double click on the process block, go to the Make System Data tab, and click Set Publish.
   b. Select the respective publish data pool and do the following:
      - Select the variable "Iterations".
      - Choose the options "Use in TDoD" and "Resolve".
6. Save and expose the flow to Test Data on Demand.

Now the testers can see the flow in TDoD based on their user privileges. Testers can execute the flow and specify the iteration count to publish data as required.

Configuring Decision Blocks in Self Service Catalog Forms

As a Test Data Engineer you can create a CA Agile Requirements Designer flow using variable in a decision block. Based on the conditions specified in the decision block, you can execute the publish or test match job without adding a separate process block.

Follow these steps:

1. Launch Datamaker and create a publish data pool or test match data pool.

2. Create a variable for the respective data pool.
3. Go to CA Agile Requirements Designer and create a visual flow.
   The flow is created with default controls of Start and End.
4. Add a decision block to the visual flow. CA TDM supports the following types of decision blocks:
   – True/False Decision Block
   – Single-Variable Case Decision Block
   – Multi-Variable Case Decision Block

   All these three decision blocks have the same behavior and functionality in general. The only difference is that each of these decision blocks defaults to certain display output values specified for Output Details properties. However, you can edit these display output properties to show-up in the self service catalog forms in the way you want. For more information, see CA Agile Requirements Designer documentation.
5. Double click on the decision block added to the flow and do the following:
   a. Go to the Make System Data tab, click Set Publish, select the respective Data Pool, and select the newly added variable. Choose the options "Use in TDoD" and "Resolve".
   b. Go to the Find System Data tab, click Set Test Match, select the respective Data Pool, and select the newly added variable. Choose the options "Use in TDoD" and "Resolve".
6. Save and expose the flow to Test Data on Demand.

Now the testers can see the flow in TDoD based on their user privileges. When the flow is executed based on the conditions specified, publish or test match jobs can be performed directly from the decision block.

Mask Data with CA TDM Portal

As a Test Data Engineer (TDE), you can mask your data with the CA Test Data Manager Portal. This tool is essential to ensure the security of Personally Identifiable Information (PII) that you handle.

This page contains information on the following topics:

This video contains a summary of the masking process.

You can only mask data that is included in a CA TDM data model. See how to create a data model at Create a Data Model and Audit PII Data.

> **WARNING**
>
> To perform masking tasks on a Data Model, a user must be in a User Group shared with the Connection Profile that created the Data Model. For more information, see User and Group Management.

Masking Process

CA TDM Portal uses Fast Data Masker (FDM) to mask data. The masking process uses masking functions in FDM. These functions are specific to data types (VARCHAR, DATE, NUMBER etc). For a full list of masking functions available in CA TDM, see Masking Functions and Parameters.

Mask Function Groups are configurations of masking functions that CA TDM Portal uses to generate FDM masking jobs. These groups include masking functions, in the order to execute them, with specific seedlists as parameters where appropriate.

For example: the Mask Function Group 'Post Code (UK)' contains one masking function ('HASHLOV'), and specifies the seedlist 'UK Post Codes' as the masking function's parameter.

TDM Portal masks columns with Mask Function Groups, based on tags that you assign to those columns in TDM Portal. You can assign more than one tag to a column - when you do this, TDM Portal defines the first tag as the column's Primary Tag, and assigns a Mask Function Group based on this tag.

You can change the Mask Function Group that CA TDM Portal assigns to a column, and create your own custom groups. For more information, see Configure Data Masking.

Assign tags to columns

You can assign tags to columns in two ways (you can use a combination of both):

- **Via PII Scan**
  Run the PII Scan to assign tags based on analysis of the data model.

  NOTE

  Default Mask Function Groups are assigned to tags by the PII Scan, but you can choose from other compatible Mask Function Groups on the Configure Data Masking page.
- **Manually**Assign tags to columns yourself. See how to do this at Manually Review PII Data.

Masking Procedure - steps

When you have a Data Model, and you have tags assigned to one column or more, you can mask your data. This procedure guides you through the data masking process in CA TDM Portal.

NOTE

The following pages under the Data Masking tab are only available when you have a Data Model.

Follow these steps:

1. Click the **Data Masking** tab from the left-hand panel.
   The Data Masking home page displays. You can click **Get Started** to go directly to the Start Masking page (step 4).
2. (Optional) Click the **Configure** button under the **Data Masking** section in the left-hand panel.
   The Configure Data Masking page opens.
   Here you can adjust masking configurations for individual columns.
3. (Optional) Click the **Masking Settings** button under the **Data Masking** section in the left-hand panel.
   The Masking Settings page opens.
   Here you can adjust settings that affect the entire masking process.
4. Click the **Start Masking** button under the **Data Masking** section in the left-hand panel.
   The Start Masking page opens.
   Here you can begin the masking job, or schedule it to start at a later time.

   NOTE

   If you do not have a data model, the **Data Masking** page only contains a **Create Data Model** button, which directs you to the data model page.
5. (Optional) Click the **Masking Jobs** button under the **Data Masking** section in the left-hand panel.
   The Masking Jobs page displays.
   This page lists all masking jobs (all Scheduled, In Progress or Completed).

The Remote Masking engine in Docker

From Test Data Manager 4.8, data masking is possible in the TDM Portal in Docker.

Additionally, the Masking Engine is available as a Docker container. This means that there are a number of methods available by which you can perform masking with Docker:

- In TDM Portal Docker containerMask data with TDM Portal in Docker, the same way you would with a standard Windows installation.
- **TDM Portal in Docker - masking engine(s) on local Docker network**

Send masking jobs to masking engine(s) in the same Docker network as the TDM Portal container.

- **TDM Portal in Docker - send to external engine(s)**
  Send masking jobs from an instance of TDM Portal in a Docker container, to one or more masking engines on remote hosts.
- **TDM Portal in Windows sends masking jobs to masking engine(s) in Docker container(s)**
  A standard installation of TDM Portal sends masking jobs to masking engine(s) running in Docker on remote host(s). This requires changes to the `application.properties` file.

For more information, see Scalable masking with Docker.

Configure Data Masking

On the Configure page, you can select which Masking Functions CA TDM uses to mask your data. For a full list of masking functions available in Fast Data Masker, see Masking Functions and Parameters.

This video takes you through the masking configuration process:

There are two views available for this page:

- **View by Table**
- **View by Tag**

View by Table

The page consists of a table, with the following columns:

- **Table**
  Name of the table in the Data Model.
- **Columns to be masked**
  Number of columns in this table with PII tags. For more information on how tags are assigned to columns, see Scan Data Model for PII.
- **Where**
  This indicates that at least one of the columns from this table uses a WHERE clause.
- **Edit**
  Make changes to a table's tags and masking functions.

Each row in the table represents a table in the Data Model. Click the plus icon next to the name of a table to show the **Expanded Table View**.

Expanded Table View

The expanded view shows a table underneath the name of the Data Model table you clicked. It contains the following columns:

- **Column**
  Name of the column in the Data Model table
- **Primary Tag**
  The first tag assigned to this column.
- **Mask Function Group**
  The Mask Function Group that CA TDM applies to the column. The Mask Function Groups available from the dropdown menu are those associated with the primary tag. The default Mask Function Group is the one that provided the best match to the classifier seedlist.
  You can assign and create other Mask Function Groups to a column with the **Edit** (pencil) icon. Click the **+** icon to create a copy of the Mask Function Group currently assigned to this column.
- **Where**

This column contains a tick when the Mask Function Group contains at least one masking function that uses a WHERE clause.

- **Edit Mask Function Group**
  Click the pencil icon in this column to open the **Edit Mask Function Group** dialog, to add and make changes to the masking functions CA TDM applies to the column.

  NOTE

  You can only make changes to custom Mask Function Groups. If you try to edit a Classifier-defined Mask Function Group, the dialog's title is 'Mask Function Group - Locked'. You can click **Make Copy & Edit** to make a copy of this Mask Function Group that you can edit.

  From this dialog, you can:
  - Change the **Name** of the Mask Function Group. If you check **Make Function Group Global**, changes you make apply to all instances of this Mask Function Group.
  - For each masking function in the Mask Function Group:
    - Select the function from a drop-down list of available masking functions.
    - Add values to parameters applicable to the selected masking function, or select a seedlist for functions that take a seedlist as their parameter.

      NOTE

      For functions that take a seedlist as their parameter, you can add seedlists from a database table. For more information, see Add Seedlists from a database table.
    - Click **Add a WHERE Clause** for the function, to further customise the masking process.
      This displays a field in which you can add an SQL WHERE clause, which column values must match in order for FDM to apply the masking function.
    - Click the **X** icon to remove the masking function from the Mask Function Group. Click **Yes** to confirm this decision.
  - Add additional masking functions to the Mask Function Group by clicking the plus (**+**) icon inside the dialog. These functions execute sequentially, in the order in which they appear in this dialog. When you have more than one masking function, you can change their execution order with the **Up** and **Down** chevron icons.
- **Add Mask Function Group**
  Click the **+** icon in this column to add a Mask Function Group.
  From this dialog, you can:
  - (Optional) **Copy Function Definition** from a dropdown list of existing Mask Function Groups. When you select one, the Mask Function Group Name field populates with 'Copy of <Mask Function Group name>'.
  - Change the name of the Mask Function Group. If you check **Make Function Group Global**, changes you make apply to all instances of this Mask Function Group.
  - For each masking function, you can:
    - Select the function from a drop-down list of available masking functions.
    - Add values to parameters applicable to the selected function, or select a seedlist for functions that take a seedlist as their parameter.

      NOTE

      For functions that take a seedlist as their parameter, you can add seedlists from a database table. For more information, see Add Seedlists from a database table.
    - Click **Add a WHERE Clause** for the function, to further customise the masking process.

This displays a field in which you can add an SQL WHERE clause, which column values must match in order for FDM to apply the masking function.
    - Click the **X** icon to remove the masking function from the Mask Function Group. Click **Yes** to confirm this decision.
  - Add additional masking functions to the Mask Function Group by clicking the plus (**+**) icon inside the dialog. These functions execute sequentially, in the order in which they appear in this dialog. When you have more than one masking function, you can change their execution order with the **Up** and **Down** chevron icons.

View by Tag

The page consists of a table, with the following columns:

- **Tag**
  Name of the tag in the data model.
- **Columns to be masked**
  Number of columns with this tag assigned, that also have a Mask Function Group assigned. If you assign multiple masking functions to columns with this tag, this value includes the number of different functions assigned.
- **Mask Function Group**
  Mask Function Group that CA TDM applies to columns with this tag. This first displays the tag's default Mask Function Group. If you choose to mask columns with this tag, with different masking functions, this field displays 'Multiple functions'.
- **Where**
  This indicates that at least one of the columns with this tag uses a WHERE clause.
- **Edit**
  Make changes to the Mask Function Group associated with a tag.

Each row in the table represents a tag in the Data Model. Click the plus icon next to the name of a tag to show the **Expanded Tag View**.

Expanded Tag View

The expanded view shows a table underneath the name of the tag you clicked. It contains the following columns:

- **Tag**
  Name of the tag in the data model.
- **Columns to be masked**
  Names of columns with this tag.
- **Mask Function Group**
  The Mask Function Group that CA TDM applies to the column.

  NOTE

  The Mask Function Group that is displayed first is the default Mask Function Group for that tag. You can change this Mask Function Group from other Mask Function Groups associated with the column's primary tag.
- **Where**
  This indicates that this column table uses a WHERE clause.
- **Edit Mask Function Group**
  Click the pencil icon in this column to open the **Edit Mask Function Group** dialog, to add and make changes to the masking functions CA TDM applies to the column.

  NOTE

  You can only make changes to custom Mask Function Groups. If you try to edit a Classifier-defined Mask Function Group, the dialog's title is 'Mask Function Group - Locked'. You can click **Make Copy & Edit** to make a copy of this Mask Function Group that you can edit.

From this dialog, you can:

- Change the **Name** of the Mask Function Group. If you check **Make Function Group Global**, changes you make apply to all instances of this Mask Function Group.
- For each masking function in the Mask Function Group:
  - Select the function from a drop-down list of available masking functions.
  - Add values to parameters applicable to the selected function, or select a seedlist for functions that take a seedlist as their parameter.

    NOTE

    For functions that take a seedlist as their parameter, you can add seedlists from a database table. For more information, see Add Seedlists from a database table.
  - Click **Add a WHERE Clause** for the function, to further customise the masking process.
    This displays a field in which you can add an SQL WHERE clause, which column values must match in order for FDM to apply the masking function.
  - Click the **X** icon to remove the masking function from the Mask Function Group. Click **Yes** to confirm this decision.
- Add additional masking functions to the Mask Function Group by clicking the plus (**+**) icon inside the dialog. These functions execute sequentially, in the order in which they appear in this dialog. When you have more than one masking function, you can change their execution order with the **Up** and **Down** chevron icons.

- **Add Mask Function Group**
  Click the **+** icon in this column to add a Mask Function Group.
  From this dialog, you can:
  - (Optional) **Copy Function Definition** from a dropdown list of existing Mask Function Groups. When you select one, the Mask Function Group Name field populates with 'Copy of <Mask Function Group name>'.
  - Change the name of the Mask Function Group. If you check **Make Function Group Global**, changes you make apply to all instances of this Mask Function Group.
  - For each masking function, you can:
    - Select the function from a drop-down list of available masking functions.
    - Add values to parameters applicable to the selected function, or select a seedlist for functions that take a seedlist as their parameter.

      NOTE

      For functions that take a seedlist as their parameter, you can add seedlists from a database table. For more information, see Add Seedlists from a database table.
    - Click **Add a WHERE Clause** for the function, to further customise the masking process.
      This displays a field in which you can add an SQL WHERE clause, which column values must match in order for FDM to apply the masking function.
    - Click the **X** icon to remove the masking function from the Mask Function Group. Click **Yes** to confirm this decision.
  - Add additional masking functions to the Mask Function Group by clicking the plus (**+**) icon inside the dialog. These functions execute sequentially, in the order in which they appear in this dialog. When you have more than one masking function, you can change their execution order with the **Up** and **Down** chevron icons.

Masking Settings

CA TDM includes predefined rules, that you can apply to the data masking process. To specify and customise these rules, click **Data Masking** in the navigation pane, then click **Masking Settings**.

Change settings on this page

The table on this page has three columns:

- **Option**Name of the rule that you can apply to the data masking process.
- **Description**A description of the rule.
- **Value**The value (parameter) that the rule uses when it applies to the data masking process.

  NOTE

  If you enter a value that is not compatible with that Option, a warning appears to inform you of the correct format.

Before you enter any values in the **Value** column, the **Value** column is empty. This indicates that this rule will not apply to the masking job, or the default value will apply.

When you have the Masking Settings how you want them, click **Next**.

The **Start Masking** page opens.

Additional settings in application.properties

In addition to the settings you can change on the Masking Settings page, there are other parameters that apply to masking in the `application.properties` file (in a default installation, this is at `C:\Program Files\CA\CA Test Data Manager Portal\conf`).

- **tdmweb.TDMMaskingService.pool.size=*4***
  Defines the maximum number of instances of FDM that Portal initiates. Default: 4.

  WARNING

  We recommend not to exceed 4 concurrent instances of FDM, due to the additional load this can place on the TDM Portal host. See Masking Performance Optimization in CA TDM Portal.
- **tdmweb.TDMMaskingService.tableTaskRowThreshold=*1000000***
  Defines the maximum number of rows to assign to an instance of the FDM Masking engine. Default: 1000000.

Start Masking

To begin the data masking process, click **Data Masking** in the left-hand navigation pane. Then click **Start Masking** from the menu below **Data Masking**. From this page you can also download the Fast Data Masker (FDM) Configuration, to allow you to perform the masking yourself in FDM.

You can begin the data masking process when you have the following:

- A Data Model
- Tags on columns in this Data Model to show that they contain PII (either from PII Scan or added manually)

If you do not have a Data Model, the page displays a 'Perform Data Discovery' button. If you click this, the **Data Discovery** page opens.

Create a Masking Configuration

The first Start Masking page contains the following options:

- **Mask all Tagged Columns in:**
  You have the option to mask tagged columns according to one of the following options:
  - **All tables**
    Mask data in all tagged columns from all tables in the selected Environment
  - **Confirmed tables only**

Only mask data in tagged columns, in tables which you manually mark as 'Confirmed' on the Data Model heat map display.

NOTE

If you check the box "Exclude tables marked as 'Not PII'", CA TDM does not mask tables that you manually marked as 'Not PII' in the Data Model section. This option overrides any tags you assign to columns in these tables.

- **Select Environment to Mask**
  You can choose to apply the masking configuration to:
  - **Current Environment**
  - **Other Environment** (select environment from drop-down menu)
- **Masking scope**
  You can choose to apply the masking configuration to:
  - **Whole Environment**
  - **Specific Data Sources** (select data sources from drop-down menu)
- **Mask Data in Preview Mode**
  If you check this option, Fast Data Masker (FDM) does not mask the data. FDM creates an audit file, which contains the results of the masking process on the data you want to be mask.
- **Show pre-masked samples in the Audit Report**
  Includes the unmasked source data in the audit report, and saves it in TDM's repository. This allows you to compare the data before and after masking. You can delete this data at any time from the Masking Jobs page.

Click **Next**.

The following page contains a Summary of your Masking job, and Additional Options.

Additional Options

There are two Additional Options available to you on this page. You should use these if you want to mask the data yourself using Fast Data Masker.

Download FDM Configuration

(Optional) Click on Download a **Configuration File** to download the Fast Data Masker configuration, if you prefer to run the data masking job manually in Fast Data Masker. This file is generated based on your choices from the Masking Settings pages.

CA TDM downloads a zip file with the FDM configurations into your web browser's default download location.

The FDM configuration zip file includes the following files:

- **info.txt file**
  Includes details about the PII data scan job such as:
  - if only confirmed tables included
  - if any not PII tables are excluded
  - for each connection profile, includes connection profile name, database type and the server name.
- **sub-directories:**These are organised by Connection Profile > Server > Database > Schema.
  Each schema directory contains the following files:
  - Text file
    Includes connection details for FDM to connect to a database.
  - CSV file
    Includes the masking rules for tables and columns for a particular database.
  - Batch file

Running the batch file (.bat) automatically uses the Connection Profile details in the text file and the masking rules in the csv file to perform masking in FDM.

  - options.txt
    A text file that contains the masking job's settings, as defined on the Masking Settings page.
- **run_mask_all.txt:**
  It is a batch (.bat) file which is by default saved as a .txt file and performs masking on all your databases. To use the run_mask_all.txt file, rename the extension from .txt to .bat and run it on a Windows machine with FDM installed to perform masking on all your databases.

Generate and Download Database Constraints Scripts

(Optional) Click **Generate and Download Constraints scripts**, to download SQL scripts to disable and enable database constraints on Primary and Foreign keys, which may prevent completion of the mask job. If you need to disable database constraints, you should run the Pre-scripts before you execute the masking job, and run the Post-scripts after you execute the masking job. This applies to masking through CA TDM Portal, and to masking with FDM.

> **NOTE**
>
> If you receive an error message for the masking job (on the Masking Jobs page), that indicates that CA TDM could not mask key columns, you should download and run constraints scripts.

If you choose this option, CA TDM downloads a ZIP file that contains the scripts for each database. The directory structure of this ZIP file is:

- **FDM_Scripts**
  - Connection Profile
    - Server
      - Database
        - Schema.

Start Masking Process

When you are happy with the settings, select a **Schedule** for your masking job. This can be:

- **Now**
  The job runs immediately
- **Schedule**
  The job runs at the time you select

Click **Mask**.

A warning displays, to inform you that the masking job will replace data for the Environments and Databases that you have selected. Click **Confirm** to start this job.

The Masking Jobs page displays. The new masking job begins and is added to the table of masking jobs.

Masking Jobs

The **Masking Jobs** page lists Data Masking Jobs that are Scheduled, Running, Complete or Failed. To find this page, click **Data Masking** from the left-hand navigation pane, then click **Masking Jobs** from the list that appears under **Data Masking**.

Masking Jobs Table

The Masking Jobs table lists all current, future or past masking jobs. The page auto-updates to reflect the most recent status of jobs.

If you click on a job to highlight it, the Additional Information pane appears on the right of the screen, with live status messages about it. This includes:

- **Task state**
  The state of the masking job.
- **Duration**
  Time that the job has been running, or time for which it ran.
- **Progress**
  Status of number of rows masked, compared to total rows to mask.
- **Pre-masked samples**
  Information whether pre-masked samples were stored (option on Masking Settings page). If they were stored, a **Delete** button allows you to delete these samples.
- **Output**
  Response from the Data Masking Engine in the case of a failed job. This includes details of any parts of the job that FDM is unable to execute.

  NOTE

  If your job fails and you receive a response that FDM was unable to overwrite a primary or foreign key, this may be because of database constraints. You can download and run Database Constraints Scripts, to disable the constraint that prevents modification of primary key column values.

From this table, you can cancel jobs, DownloadAudit and delete pre-masked samples.

Cancel a masking job

Click the Stop button, at the right-hand end of the job's row in the table, to cancel the job.

CA TDM starts an instance of Fast Data Masker for each schema in your masking job, up to the maximum you define (see **Tip** below and example of Threads in CA TDM Portal).

When you cancel a masking job, CA TDM allows all instances of FDM that are in progress to complete to avoid database inconsistencies, but no new instances of FDM start. Schemas that are allowed to complete masking after you click Cancel, and schemas that do not start masking, are listed on the **Output** pane for the job that you cancel.

The TDM Portal audit log and masking log provide details of masking jobs that do not complete.

- By default, the portal log file is located at `C:\ProgramData\CA\CA Test Data Manager Portal\logs` . See Manage Portal Log Files for details on how to change the location of log files.
- The audit log is a table within the Test Data Repository (`gtrep.gtrep_audit_log`).

Download an audit report for a masking job

When an audit log ends (either it completes masking, or you cancel it), the **Cancel** icon changes to a **Download** icon. Click this icon to download a ZIP file that contains the following:

- **config.txt** file
  Masking configuration for the job
- **audit.csv** file
  A table in comma-separated value format, that contains a summary of which columns CA TDM masked, and with which masking functions.
- Detailed audit file (name format "**<Connection Profile>_<Server Name>.csv**")
  A table in comma-separated value format, that contains a summary of the values CA TDM masked, and from which table each value came.

NOTE

If you enabled 'Show pre-masked samples in the audit report' on the Masking Settings page, the 'PRE-MASKED SAMPLE' column contains the original values.

Delete pre-masked samples

If you enabled 'Show pre-masked samples in the audit report' on the Masking Settings page, you can delete these samples from the gtrep repository. Click the **Delete** button from the **Additional Information** pane to delete samples.

Add Seedlists From a Database Table

Some masking functions use a seedlist to generate replacement data. These seedlists can come from two sources:

- Text files
- A table in a database.

This page explains how to create seedlists from a table in a database.

WARNING

Users who initiate masking operations must have permission to access the Connection Profile for the Database that contains the Seedlist (otherwise the masking job fails with the error "Configuration did not contain any masking information so FDM cannot be started").

We recommend that you share the Connection Profile with this group.

For example, you can use a table from the Scramble database, included with the software. These tables already contain seedlists, and you can add to these tables, to use both the existing seedlists and your own. See Install Scramble Components for more information.

Seedlists from database columns

To generate seedlist(s) from a database table, you must identify three columns within the database that contains the seedlist(s):

- **Name Column**
  This column supplies the name(s) of the seedlist(s). CA TDM groups matching values in this column into seedlists, and takes the values for that seedlist from the corresponding values in the Value column.
- **Value Column(s)**
  This column(s) supplies the values for a seedlist. Values for all seedlists must be in this column(s).
- **Index Column**
  This column acts as an enumerator for each seedlist. For each seedlist it should start with 1 and increase in increments of 1. See the table below for a worked example.

Guide to creating a seedlist from a database table

This guide details how to add seedlists from data in tables in your database, for use with masking functions.

Follow these steps:

1. Click **Configuration** in the left-hand navigation pane.
2. Click **Masking** under the **Configuration** section.
   The **Masking Configuration** page opens.
3. To use the database table that you define on this page to generate masking seedlist(s), check **Use Database seedlist**.
4. To define seedlist(s) from a database table, select it/them with the fields on this page:

- **Connection Profile**
  Select the connection profile from a dropdown list of the connection profiles you define. For more information, see Create and Edit Connection Profiles.
- **Seedlist Table**
  The name of the table that contains seedlist data (by default 'gtsrc_reference_data').
- **Seedlist Name Column**
  The name of the column that contains the name(s) of the seedlist(s) (by default 'rd_ref_id').
- **Seedlist Value Columns**
  The name(s) of the column(s) that contains the values for the seedlist(s) (by default 'rd_ref_value').
- **Seedlist Index Column**
  The name of the column that contains the enumerator values for each seedlist (by default 'rd_index').

5. Click **Save** to save the options above.
   Your seedlist(s) from these columns are now available for use with masking functions that take seedlists as a parameter (e.g. HASHLOV).

Example of seedlists from database table

For the table **Seedlist_Table** below, if you use the **Seedlist_Name** column as the Name column, the **Seedlist_Value** column as the Value column, and the **Seedlist_Index** column as the Index column, you create the following 3 seedlists (each of which contains 4 values):

- UK_MALE_FIRSTNAME
- UK_FEMALE_FIRSTNAME
- UK_SURNAME

Seedlist_Table		
Seedlist_Name	Seedlist_Value	Seedlist_Index
UK_MALE_FIRSTNAME	John	1
UK_MALE_FIRSTNAME	Peter	2
UK_MALE_FIRSTNAME	David	3
UK_MALE_FIRSTNAME	Richard	4
UK_FEMALE_FIRSTNAME	Sarah	1
UK_FEMALE_FIRSTNAME	Anne	2
UK_FEMALE_FIRSTNAME	Lucy	3
UK_FEMALE_FIRSTNAME	Jane	4
UK_SURNAME	Smith	1
UK_SURNAME	Jones	2
UK_SURNAME	Bennett	3
UK_SURNAME	Wright	4

Masking Performance Optimization in CA TDM Portal

CA TDM Portal performs masking with Fast Data Masker. CA TDM Portal can run multiple instances of FDM concurrently (the maximum and default number of instances is 4). You may be able to perform your masking job faster, if you can split the job into smaller jobs that Portal can process with concurrent instances of FDM.

Assess the size of your environment

The size of the data set that you want to mask (i.e. number of tables in each database/schema, number of columns and rows in tables) has an effect on how much memory FDM needs to mask the data, and how long it takes. The amount of memory Fast Data Masker requires for a masking job generally increases linearly in relation to the number of tables, columns and rows to mask.

You can see how many rows and columns a table contains in the filter to see only larger tables.

> **TIP**
>
> You can use the PARALLEL option on the Masking Settings to set the number of parallel Java threads. Within an instance of FDM, FDM creates a Java thread for each table.

Memory use in FDM

For every 1 million rows and 100 columns to mask, 1GB of memory is generally sufficient to maintain optimum performance (see table below).

Rows	Columns	Memory Recommended
1 million	100	1GB
2 million	100	2GB
2 million	200	4GB

Allocate memory to masking instances

The **HEAPSIZE** option on the Masking Settings page controls how much memory, in MB, TDM Portal assigns to each FDM instance. The default value is 1000MB (1GB).

The allocation of less memory than this for each instance is likely to result in slower masking.

Optimize concurrent jobs in CA TDM Portal

Connection Profiles in CA TDM Portal

CA TDM Portal creates an instance of FDM for each **Connection Profile** in a masking job. A masking job can run on either of the following (see Start Masking for more information about Masking Configurations):

- Environments. These consist of data sources that it accesses through **Connection Profiles**.
- Specified **Connection Profiles**.

The creation of multiple Environments, each with one Connection Profile, **does not** improve masking efficiency. However, a Connection Profile can contain multiple schemas.

> **TIP**
>
> If you create one **Connection Profile** for each **schema**, this results in more concurrent instances of FDM (up to a maximum of 4).

For example: If you have 10 Connection Profiles in your masking job, and each Connection Profile contains one schema (and your maximum number of instances is 4) CA TDM creates 4 instances of FDM (with one schema on each instance). The other 6 schemas are queued until one of the instances completes the job. See Masking Jobs for example of CA TDM's behaviour when you cancel a job with multiple FDM instances.

Connection Profiles	Schemas per Connection Profile	Maximum concurrent FDM instances
1	4	1

1	10	1
4	1	4
10	1	4

Memory Usage for concurrent masking instances

The total memory required for a masking job that contains multiple concurrent instances of FDM is equal to the sum of the memory required for each instance of FDM.

For example, if your job contains **4 instances**, and each one requires **1GB** of memory, the total memory you need is **4GB**.

Optimize performance with Scalable Masking

From TDM 4.8, the Masking Engine is available as a Docker container, which you can use to distribute a masking job's masking tasks across multiple machines and/or instances of the Masking Engine. For more information, see Scalable masking with Docker.

Each instance of the Masking Engine (which is an instance of FDM), can perform **4** tasks concurrently by default.

> **NOTE**
>
> Contact CA Support for information on how to change this value. We recommend that you leave it at 4.

The fastest way to mask data is to have enough instances of FDM active, that all your masking tasks can run concurrently. The best way to increase the total number of instances of FDM, is to increase the number of instances of the Masking Engine container, with the **--scale** parameter.

> **TIP**
>
> You can add instances of the Masking Engine to your Docker network while the network is active. To do so, increase the `--scale masking=n` parameter in the `docker-compose up` command that you use to start your Docker network, and execute the `docker-compose up` command again.

Calculate masking tasks

When you use TDM Portal with the Messaging container, the Masking service (part of TDM Portal) creates additional masking tasks according to the following logic:

- Each database/schema is one masking task.
  - Within each database/schema, any tables of more than 1,000,000 rows (by default) are split into an additional task.

> **NOTE**
>
> To change this default value, change the `application.properties` parameter **tdmweb.TDMMaskingService.tableTaskRowThreshold**.
>
> For information on how to set these properties in your TDM Portal Docker container, see Custom application.properties configuration.

Therefore, you can calculate the total number of masking tasks for a masking job with the following formula:

```
Total number of masking tasks = Number of databases or schemas + Number of tables of
 over 1,000,000 rows
```

For example:

You have a masking job that includes 2 databases, consisting of the following databases and tables:

SALES	
Table Name	Row Count
Customers	1,100,000
Suppliers	24,000
Items	250,000
Orders	1,450,000
Employees	10,000
CUSTOMER_CARE	
Table Name	**Row Count**
Customers	1,100,000
Purchases	1,300,000
Review Scores	60,000

This masking job consists of:

- 2 databases (**SALES** and **CUSTOMER_CARE**)
  The Masking service creates an additional masking task for each database.
- 4 tables with more than 1,000,000 rows each (**SALES.Customers**, **SALES.Orders**, **CUSTOMER_CARE.Customers** and **CUSTOMER_CARE.Purchases)**
  The Masking service creates an additional masking task for each table with more than 1,000,000 rows.

Therefore, the total number of masking tasks for this masking job is **6** (**2** databases + **4** tables of over 1,000,000 rows in size). To manage 6 masking tasks concurrently, with 4 tasks per Masking Engine container, you would need **2** Masking Engine containers.

You can use the following `docker-compose up` command to set the number of instances of the Masking Engine container to 2:

```
docker-compose -f docker-compose.yml -f docker-compose-messaging.yml -f docker-compose-masking.yml up -d --scale masking=2
```

Results of insufficient memory

For optimum masking performance, the system that hosts instances of FDM (either the Windows application, or the Docker container) must have enough physical memory for all concurrent instances of FDM. Less memory can result in:

- Slower performance of masking jobs
- Slower CA TDM Portal performance

Maximise memory availability

To maximize memory available to CA TDM Portal, we recommend that you schedule jobs to run at a time when memory load is lower.

Scalable Masking with Docker

From Test Data Manager 4.8, masking functionality is available for Docker-based deployments of TDM Portal. For more information on masking in Portal, see Mask Data with CA TDM Portal.

To mask data with the TDM Portal Masking Engine container. This functionality allows you to distribute your masking jobs across multiple hosts, to run concurrently.

TIP

Use the files **docker-compose-messaging.yml** and **docker-compose-masking.yml**, to add these containers to your Docker network. For more information, see Docker-compose files.

You can also use these Docker containers with TDM Portal in Windows, to distribute masking jobs.

Overview

Scalable masking is possible with TDM Portal and the following two Docker containers:

- **Message Bus Server container**
  This container contains a Java Messaging Service (JMS) queue of the tasks that constitute your masking job, and distributes them to the Masking Engine container(s). This container uses RabbitMQ messaging software.

  NOTE

  The TDM Portal Masking Service splits your masking job into tasks, to send to the Message Bus Server container.
- **Masking Engine container**
  This container starts up to four instances of the Fast Data Masker masking engine to perform each masking task. You can start multiple instances of this container.

Scale the masking service

You can run multiple instances of the Masking Engine container with the **--scale** flag in your `docker-compose up` command. The following example starts a Docker network with the **tdmweb**, **orientdb** and **messaging** containers active, and 3 instances of the **masking** container:

```
docker-compose -f docker-compose.yml -f docker-compose-messaging.yml -f docker-compose-masking.yml up -d --
scale masking=3
```

You can use this feature to optimize the performance of your masking jobs.

TIP

For more information, see Optimize performance with Scalable Masking.

Use Cases

The main use cases for scalable masking with the TDM Portal masking container are as follows:

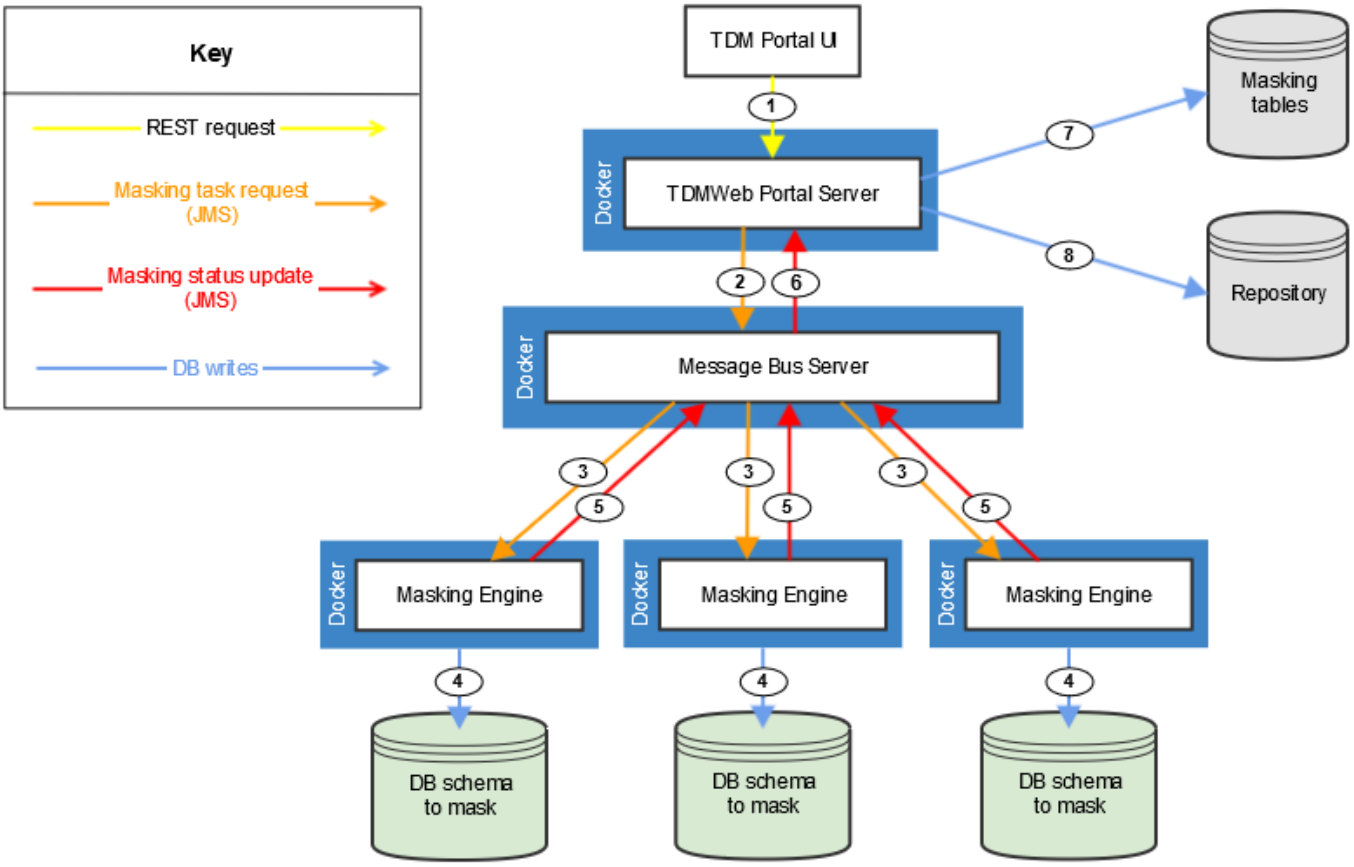
- **With TDM Portal in Docker - local masking**
  Your TDM environment runs within a Docker network. This network includes the Message Bus Server container, and the Masking Engine container(s).
- **With TDM Portal in Docker - remote masking**
  Your TDM environment runs within a Docker network. This network includes the Message Bus Server container, but the Masking Engine container(s) to which it sends tasks, are on remote hosts.
- **With TDM Portal in Windows**
  Your TDM environment runs in Windows. The Message Bus Server container and Masking Engine container(s) must be on Docker networks available to the Masking Service, but they can be local or remote.

Diagram of scalable masking operation

The diagram below illustrates the following steps:

1. User initiates masking job via REST request
2. The Masking Manager in the TDMWeb Portal server (in Windows or in Docker) resolves job information and splits the job into tasks (maximum of one per schema), which it adds to the Message Bus Server's queue.
3. Masking engine(s) pull tasks from Message Bus Server (JMS) queue.
4. Masking engine begins masking operations on database table.
5. Masking engine provides status updates and final audit via JMS queue.
6. TDM Portal Server pulls status and audit messages from JMS queue.
7. After completion, TDM Portal Server writes final audit information to masking store.
8. TDM Portal Server writes job status to repository.

Figure 43: scalable masking



Security password

For data security, a password is necessary, which must match on all services. Supply it as the following parameters or environment variables:

- In TDM Portal in Windows:
  `tdmweb.TDMMaskingService.messaging.password`
- In TDM Portal in Docker:

`MESSAGING_PASS`

- In the Messaging container:
  `DEFAULT_PASS`
- In the Masking Engine container:
  `MESSAGING_PASS`

This password can be either plain text or encrypted.

How to Implement the independent masking engine

To use the masking engine, it is necessary to perform 3 steps:

1. Configure TDM Portal's connection to the Messaging container. This can be done in two ways:
   a. Configure connection from TDM Portal in Windows to the Messaging container.
   b. Configure connection from TDM Portal in Docker to the Messaging container.
2. Configure the Messaging container's connection to TDM Portal and the Masking Engine container.
3. Configure a Masking Engine container's connection to the Messaging container

1a. Configure connection from TDM Portal in Windows to the Messaging container

To send masking jobs from TDM Portal in a Windows environment, to the Messaging container, it is necessary to add the following lines to the `application.properties` file (by default, this is at `C:\Program Files\CA\CA Test Data Manager Portal\conf`):

```
tdmweb.TDMMaskingService.messaging.host=messaging
tdmweb.TDMMaskingService.messaging.port=5671
tdmweb.TDMMaskingService.messaging.username=Admin
tdmweb.TDMMaskingService.messaging.password={cry}1hY5pZrm87PWjgPdmypDbVZnL4a108lxy8YLuUVRMCr8

# The tableTaskRowThreshold parameter is not specific to scalable masking
# tdmweb.TDMMaskingService.tableTaskRowThreshold = 1000000
```

Where

- **tdmweb.TDMMaskingService.messaging.host**
  Specifies the **hostname** of the Message Bus Server container. Default: *messaging*.
- **tdmweb.TDMMaskingService.messaging.port**
  Specifies the **port number** of the Message Bus Server container. Default: *5671*.
- **tdmweb.TDMMaskingService.messaging.username**
  Defines the username with which you access the Message Bus Server.
- **tdmweb.TDMMaskingService.messaging.password**
  Defines the password. Can be encrypted or unencrypted.
- **(Optional) tdmweb.TDMMaskingService.tableTaskRowThreshold**
  Sets the maximum number of rows to assign per instance of FDM (i.e. per container). Default: 1000000.

  NOTE

  This parameter is not specific to scalable masking. It also applies to masking jobs you perform without Docker containers.

1b. Configure connection from TDM Portal in Docker to the Messaging container

The Messaging container, with reference to the following entries under **environment** in **docker-compose.yml** (the file that you use to start the TDM Portal container):

```
services:
                tdmweb:
      ...
      hostname: tdmweb
      environment:
       - 'MESSAGING_SERVER=messaging'
       - 'MESSAGING_PORT=5671'
       - 'MESSAGING_USER=Admin'
       - 'MESSAGING_PASS={cry}1hY5pZrm87PWjgPdmypDbVZnL4a108lxy8YLuUVRMCr8'
       # - 'APPLICATION_PROP="tdmweb.TDMMaskingService.tableTaskRowThreshold=1000000"'
```

Where

- **MESSAGING_SERVER**
  Specifies the **hostname** of the Message Bus Server. Default: *messaging*.
- **MESSAGING_PORT**
  Specifies the port number of the Message Bus Server container. Default: *5671*.
- **MESSAGING_USER**
  Defines the username with which you access the Messaging container.
  Must match Messaging container environment variable **DEFAULT_USER**.
- **MESSAGING_PASS**
  Defines the password for the user with which you access the Messaging container. Can be encrypted or unencrypted.
  Must match Messaging container environment variable **DEFAULT_PASS**.

  NOTE

  For encrypted passwords, begin this value with `{cry}`
- (Optional) **APPLICATION_PROP="tdmweb.TDMMaskingService.tableTaskRowThreshold"**
  Sets `application.properties` parameter `tableTaskRowThreshold` , to set the maximum number of rows to assign per instance of FDM (i.e. per container). Default: 1000000.

2. Configure the Messaging Container's connection to TDM Portal and the Masking Engine(s)

For the Masking Engine container, it refers to the following values in the file **docker-compose-messaging.yml**:

```
services:
   messaging:
      hostname: messaging
      ports:
      - '5671:5671'
      environment:
      - 'DEFAULT_USER=Admin'
      - 'DEFAULT_PASS={cry}1hY5pZrm87PWjgPdmypDbVZnL4a108lxy8YLuUVRMCr8'
```

Where

- **hostname:messaging**
  Defines the hostname of the service. Default: *messaging*.
- **- '5671:5671'**
  Defines the port through which this container communicates. Default: *5671*.
- **- 'DEFAULT_USER'**
  Specifies the username with which to send and receive masking tasks.

Must match the value of **messaging.username** (Portal in Windows) or **MESSAGING_USER** (Portal in Docker), and **MESSAGING_USER** in the Masking Engine container.

- **- 'DEFAULT_PASS'**
  Specifies the password for the user with which to send and receive masking tasks.
  Must match the value of **messaging.password** (Portal in Windows) or **MESSAGING_PASS** (Portal in Docker), and **MESSAGING_PASS** in the Masking Engine container.

3. Configure a Masking Engine container's connection to the Messaging container

For Messaging container, they refer to the following values in the **docker-compose-masking.yml** file that you use to start your Masking Engine container(s):

```
services: masking:
      ...
      hostname: masking
      environment:
       - 'FDM_LICENSE=FDM_License_key'
       - 'MESSAGING_SERVER=messaging'
       - 'MESSAGING_PORT=5671'
       - 'MESSAGING_USER=Admin'
       - 'MESSAGING_PASS={cry}1hY5pZrm87PWjgPdmypDbVZnL4a108lxy8YLuUVRMCr8'
```

Where

- **MESSAGING_SERVER**
  Specifies the **hostname** of the Messaging container. Default: *messaging*.
- **MESSAGING_PORT**
  Specifies the port number of the Messaging container. Default: *5671*.
- **MESSAGING_USER**
  Specifies the username with which the service accepts tasks from the Messaging container.
  Must match the value of **messaging.username** (Portal in Windows) or **MESSAGING_USER** (Portal in Docker), and **DEFAULT_USER** in the Messaging container.
- **MESSAGING_PASS**
  Specifies the password for the user with which the service accepts tasks from the Messaging container.
  Must match the value of **messaging.password** (Portal in Windows) or **MESSAGING_PASS** (Portal in Docker), and **DEFAULT_PASS** in the Messaging container.

Additional Considerations

Expose logs for masking and messaging

You can expose the logs from the Message Bus Server container and the Masking Engine container, so that they are available for review independently of the container.

For more information, see Masking Engine container data volumes.

Use custom seedtables

You can use custom seedtables to mask data with the Masking Engine container.

For more information, see Use Custom Seedtables with the Masking Engine container.

Optimize masking jobs across multiple masking containers

You can use multiple Masking Engine containers to perform the masking tasks that make up your masking job concurrently.

For more information, see Optimize performance with Scalable Masking.

Tester Self-Service

As a software tester, you require the appropriate test data for your test cases, along with assurance that the data will not change or disappear while you require it. Test Data Manager provides the following self-service interfaces that let you dynamically request, reserve, and obtain the data you need to execute test cases when you need it:

CA TDM Portal

As a software tester, you use the Self Service Catalog menu in the CA TDM Portal to generate, find, and reserve test data for your test cases. The TDE defines flows to facilitate data generation and data reservation processes based on testers' requirements. You can see different flows designed for different purposes based on your user privileges. Identify the right flow that meets your requirements and then execute it to generate, find and reserve the data you need. If you do not see a flow that fits your test data requirements, contact your administrator or test data engineer to get the needed flow.

Follow the below procedures to understand how to use various self-service flows for your specific test data requirement:

- Find and Reserve Test Data Interactively
- Reserve Data with Self Service Catalog Forms

> **NOTE**
>
> If you are a Test Data Engineer, you need to Configure Test Data Reservation Service.

Find and Reserve Test Data Interactively

As a tester, you can consume forms created by the Test Data Engineer on the Self-Service Catalog tab of the CA TDM Portal to find and reserve test data interactively. For more information about how the TDE creates these forms, see Configure Dynamic Test Data Reservation Service.

As a tester, to find and reserve test data interactively, and to manage your reservations, follow these procedures:

> **WARNING**
>
> In the current version of the *find and reserve* capability, there is a limitation with the usage of the connection profiles with the Oracle database. The Oracle connection profiles created in the CA TDM Portal need users to have schema ownership. If a user does not have permissions on all the schemas used for *find and reserve*, then the functionality does not work.

The following video provides a summary of the find and reserve process for a tester: (Some TDM 4.9 features are not covered.)

Find and Reserve Test Data

As a tester, you can find and reserve test data for exclusive use for your test cases using dynamic forms from the Self-Service Catalog interface.

You can also specify multiple reservation criteria as part of a single reservation request. This ability helps you manage your reservations more efficiently. Instead of sending multiple reservation requests to reserve the required test data in an environment, you can use one single request that contains data coming from different search criteria. You can simply keep on adding your selected records to a cart by changing your reservation criteria. The cart acts as a single placeholder that consolidates all the selected records from different reservation criteria at one place. You can then review the combined list and delete any records that do not meet your testing requirement. You can finally proceed with the process of reserving those records through a single reservation request.

For example, you as a Tester want to find and reserve all the products that are shipped by CourierA or CourierB. In this case, you do not need to send two separate reservation requests—one for CourierA and other for CourierB. You can achieve this by using a single request. To do so, you use CourierA in your search criteria, find all the products that CourierA has shipped, select the required product rows, and add them to the cart. You now change the search criteria and use CourierB, find all the products for CourierB, select the rows, and add them to the same cart. The cart now includes records coming from multiple search criteria. You review your records and proceed with the reservation request.

NOTE

If email notifications are enabled for the test data reservation process, an email about the reservation status is sent to the intended recipients when the reservation process completes.

To see more context for the data, enable **Show Related Tables**. Note that Legacy Models do not support showing related tables.

- With the **Show Related Tables** feature enabled, TDM displays more columns of the Find & Reserve models. The root table is displayed in red and its related tables are blue. To choose which columns are visible in table, click the Hamburger button.
  When viewing related tables, some rows may appear duplicated. When you reserve one row, multiple related rows appear also selected in a redundant, flattened view, to give you context. However, TDM reserves only the row in the *root* table.
- The same **Show Related Tables** feature is also available under **My Reservations**, to give you more context. The root table is displayed in red and its related tables are blue. This table is similarly flattened and displays (only seemingly) duplicated lines.
  If you download the data as CSV file, and enable the **Show Related Tables** checkbox, the file also contains the related columns.
- In new models, form filters are linked to each other for related tables; the forms do not offer inapplicable combinations. For example, when you select a country, then the related city form is now pre-filtered to cities that are linked to that country. Filter linking is not supported for legacy models.

Follow these steps:

1. Access the CA TDM Portal as a tester.
2. Select the required project and version from the **Project** drop-down list.
3. Navigate to **Self Service Catalog** in the left pane.
   The **Self Service Catalog** page opens to show the available forms enabled for you.

   NOTE

   If you select the **All Projects** option, the page shows all the forms irrespective of the project version that you selected from the **Project** drop-down list. If you want to see the forms associated with the selected project version, clear the **All Projects** option.
4. Identify the form that fits your test data requirements and click the **New Request** button.
   The respective form opens.
5. From the **Environment** drop-down list, select the environment which you want to search for the required test data.
6. Enter values in the fields to specify the filter criteria to find the data. These fields correspond to the columns that the Test Data Engineer adds and makes visible, during creation of the Test Data Model.
   - **Text data types**
     TDM matches text you enter in this field with an implicit wildcard at the start and end. For example: if you enter "**car**" in this field, your results include "s**car**", "**car**d" and "s**car**f".
   - **Numeric data types**
     Choose an operator to define the search field's criteria. This can be "Equal to", "Greater than or equal to", "Less than or equal to" or "Number between (inclusive)" (adds a second field).
   - **Fields of type "Date", "Time", "Datetime" or "Timestamp"**

- Choose an operator to define the search field's criteria. This can be "Exact date", "Date on or before", "Date on or after" or "Date between (inclusive)" (adds a second field).
- Pick a date from the dropdown calendar, or specify the value in the format 'yyyy-MM-dd' (for example, 2017-07-22).

– **Drop-down lists**
Fields configured as drop-down lists auto-suggest values for that element as you type. To browse all values, press the down arrow key on your keyboard. You can check all values for which you wish to filter (this does not apply to number/date values).

WARNING

Fields configured as drop-down lists **only** filter items selected from the drop-down list. Other text in the field is not filtered.

7. Select the **Include Reserved Data** check box if you want your results to include data that is already reserved. This option is only available if the Test Data Engineer enabled it in the Test Data Model (see Create a Test Data Model). With this option selected, results include three more columns - **Status**, **Reserved By** and **Reserved On**. Data that is reserved and that matches the filter criteria, appears in results with the status 'Reserved' and the details of who reserved it, and when. You can use this information to request that the user who reserved the data clones or releases that data for you to test.

   WARNING

   Do not use others' reserved data for testing.
8. (Optional) Enable **Show Related Tables** to see more context and link tables filters.
9. Click the **Find Data** button.
   The CA TDM Portal searches the data based on the specified filter criteria and displays only the matched test data. Currently, only fields from the root entity of the test data model are displayed.
   Click the down arrow on the scroll bar to scroll down and see the complete list of matched data rows.
   To show or hide the columns to display in the UI grid, click the Column Selector icon (icon with three horizontal bars at the top right hand side of the UI grid) and select the columns as required. When you select the columns, the Tick mark indicates that the respective column is shown in the grid and the Cross mark (X) indicates that the respective column is hidden in the grid.
10. Identify the test data that matches your specific test case criteria and select the check box in the corresponding row. Select all the records that you want to add to the reservation.

    WARNING

    If any model key value is NULL in a reservation, the reservation is not allowed.
11. Click the **Add to Reservation** button.
    All selected records are added to the cart. The cart icon (basket) at the top of the table is enabled and shows the number of records that are added to it.

    TIP

    You can change your search criteria to find related data, select the appropriate records, and add them to the cart, where records from the first search criteria are already added. You can keep on adding more records to the cart by changing your criteria for the same environment. If a record is already added to the cart as a result of your previous search, the same record is shown as selected in subsequent searches.
12. Click the **Complete Reservation** button (or the cart icon at the top of the table) after you are done adding records to the cart.
    The **Items added to Reservation** dialog opens.
13. Perform the following tasks:
    a. Enter a name for your reservation.
       If you do not enter a name, the default name is used as *<Test Data Model Name_Environment Name_MM-DD_Time>*. For example, Mytestdatamodel_Myenvironment_10-28_10:34am.

The *MM-DD_Time* represents the date and time of you local computer when the reservation request is submitted.

b. Review the records that are added from different search criteria.
c. Click the Delete icon (cross) corresponding to the record that you do not want to reserve. To remove all the records, click **Remove all**.
d. Define an expiration date for this reservation. The default is 100 years from the day the reservation is made.

TIP
An administrator can customize the default expiration date by defining the `reservation_expiry_in_days = <number of days>` property in the `conf\tdmdatareservation.properties` file; set this value to 0 days to restore the default expiration date of 100 years from the day the reservation is made.

14. Click **Reserve**.
A message with a reservation name is displayed.
15. (Optional) Click the reservation name shown in the message.
The **My Reservations** page opens to show the list of submitted reservations.

NOTE
You can access your reservations from the **My Reservations** page at any time to review the reservation details and/or release the reservations.

16. Identify the reservation that you submitted. You can see the reservation in one of the following states:
– Created
– Success
– Failed
If the reservation is failed, you can see the reason under the Comments column of the respective reservation request.

You have successfully reserved the test data.

Review and Download the Reserved Test Data

As a tester, when you submit a reservation from the Self Service Catalog a reservation request is submitted for processing. You can see the reservation requests submitted in the CA TDM Portal on the My Reservations page. You can also download the reserved model keys in a CSV file.

Follow these steps:

1. Access the CA TDM Portal.
2. Click the **My Reservations** option from the left hand menu.
The My Reservation page shows the list of reservation requests which you submitted. You can see the reservation request in one of the following states:
– Created
– Success
– Failed
3. When the reservation request shows the status as "**Success**", click a reservation request under the **Name** column to see the details of the reservation.
The ***<Reservation_Name>*** page opens and displays the model keys that you have reserved.
4. Click the Download Model Keys as CSV icon (down arrow) next to the **Model Keys** table.
The **Save As** dialog opens.
5. Specify the CSV file name and the location where you want to save the CSV file. By default, the name of the CSV file is <*Reservation_Name*>.csv.
6. Click **Save**.
The CSV file is saved to the specified location.
7. Navigate to the location and review the CSV file.
The downloaded CSV file includes all the reserved model keys.

Release the Reserved Test Data

As a tester, after using the test data you reserved for your test cases, you can release the reservation so that the reserved resources are made available for any future reservations.

NOTE

If email notifications are enabled for the test data reservation process, an email about the release status is sent to the intended recipients when the release process completes.

Follow these steps:

1. Access the CA TDM Portal.
2. Click the **My Reservations** option from the left hand menu.
   The My Reservation page shows the list of reservation requests which you submitted.
3. Identify the reservation request that you want to release. You can use the search functionality to find a reservation request by its Reservation Name, Environment Name, or Test Data Model Name.
4. Click the **Release** button under the **Actions** column in the row corresponding to the reservation request. Alternatively, click the reservation request, go to reservation details page, and click **Release** button under the **Actions** column.
   A success message appears to confirm that the reservation is released.
5. Click the **Refresh** icon to verify the reservation status. The status of the reservation request, first changes to **"Invalid"** and then to **"Purged".**
   The reservation resources are successfully released and available for any future reservations.
   By default, released reservations are permanently deleted after 30 days, and this deletion process runs once every 12 hours. You can configure these values (i.e. deletion process running interval and number of days to keep the reservations in purged or failed state).For more information, see Configure CA TDM Portal for Deleting the Purged Reservations.

Reserve Data with Self Service Catalog Forms

TDEs create self service forms using the CA Agile Requirements Designer (ARD) and exposes them to CA TDM Portal. These ARD based self service forms are then available for testers as forms in the Self-Service Catalog interface. As a tester you can consume these ARD based self service forms to publish the data, to perform the test match, and to attach the test data to HPALM or Rally test cases. For more information about how the TDE creates an ARD based self service form, see Configure Test Data Reservation Service.

Follow these steps:

1. Open CA TDM Portal and log in with valid user credentials.
2. Navigate to the **Self Service Catalog** through the menu.
   The **Self Service Catalog** page opens to show the available forms.
3. Identify the form that fits your test data requirements.
4. Click the **New Request** button on the form.
5. Choose whether you want to use your **Last Used Values** or **Default Values**.
   Depending on the complexity of the test data and the request, the form contains dozens of fields or only a couple. If your last used values have been stored in the browser, they are highlighted in the form.
6. Complete all the fields on the form and click **Request**.

   NOTE

   If you leave the **Email** field blank, TDM sends the Request to your registered email address.

   The **Requests** page opens to show the list of submitted requests with all the associated jobs.
7. Find the Job ID of the request you submitted and expand to see the associated jobs mapped to your request. You can see the request and the associated jobs in one of the following states:

- Running
- Complete
- Error

When your data is available, a file is attached to the corresponding request.

8. Download the file attached to the request that contains the requested data.
   You can also access these request results from the **Submitted Requests** page.

Access Submitted Requests

As a tester, when you submit a request from the Projects, Generators or Data Catalog a job is submitted for processing. You can see the requests submitted and the associated jobs in the CA TDM Portal on the Requests page.

Follow these steps:

1. Access the CA TDM Portal.
2. Click the **Requests** option from the left hand menu.
   The Requests page shows the list of requests logically arranged with the associated jobs. Lists the Data Catalog requests by default.
3. Select the request type from the Requests drop-down list. Following are the available options:
   - **Data Catalog**
     Lists all the requests submitted related to Data Catalog.
   - **Projects**
     Lists all the requests submitted related to Projects.
   - **Generator**
     Lists all the requests submitted related to Generators
4. Find the Job ID of the request you submitted and expand to see the associated jobs mapped to your request. You can see the request and the associated jobs in one of the following states:
   - Running
   - Complete
   - Error
5. When your data is available, a file is attached to the corresponding request. Download the file attached to the request that contains the requested data.
   Note: You can view the requests by organizing them using the following functions:
   - **Sort**
     You can sort the requests by a specific column. Following are the available columns to sort the requests.
     - Job ID
     - Job Name
     - Project Name
     - Job Type
     - Job Submitted
     - Scheduled Start
     - Start Date
     - End Date
   - **Additional Information**
     Select a request or an associated job to see the additional information.
   - **Pagination**
     By default the list shows 25 requests per page. You can use pagination to see more results in the subsequent pages.

Tutorial Video

Watch the following video for a visual walk-through of a use case of Test Data Reservation using the CA TDM Portal Self Service Catalog (CA ARD Flow).

Virtual Test Data Management (vTDM)

vTDM provides a lightweight mechanism for Test Data Engineers to rapidly and cheaply provide Testers with access to Test Data. Instead of Testers sharing the same data source, they can access their own virtual copies of the test data, without locking the data for other testers. Each Clone of the Data Source is available near-instantaneously and takes virtually no space. More space is taken up only when a Tester changes data. Data copies can be automatically connected to a SQL Server instance or an Oracle Server instance. Using vTDM minimizes test duration, storage requirements, compute overhead, and therefore, overall costs.

The current approach of test data management is to generate and sub-set physical data to limit the size of data, to reduce cost of data storage space. In previous releases, the Test Data Manager tools let Testers reserve data, so other users cannot modify the test data in test reserved by other testers. Now, with Virtual Test Data Management (vTDM), Testers can work independently on virtual copies of the same test data, and increase the level of test coverage for the application under test. Virtual Test Data Management provides a markedly reduced footprint for each Tester to have the full environment, with all Testers having instant access to all the data.

Save Money and Time

In the Getting Started with vTDM video, Joe *faces the issue that creating copies of test data uses expensive storage and computing resources. His goal is to create multiple copies of test data at minimal cost.*

Joe estimates how much time he saved is based on typical speed and time values for database copy and attachment operations. The time saved value sums up the time saved for all Clones.

Joe estimates how much storage space he saved based on the space taken by each checkpoint, times the number of its Clones, for all Filesystems. The appliance increments the running total of the delta between the size of the checkpoint and the clone size.

Summary

- Each tester can create their own ’personal copy’ of test data, on demand
- vTDM performs copy on write operations at the filesystem level
- vTDM can support many data sources (RDBMS, flat-files, etc.)
- vTDM offers a public API for all operations

This section contains the following procedures:

API Reference material:

- Use APIs to Manage and Consume vTDM Clones
- TDMvDataService

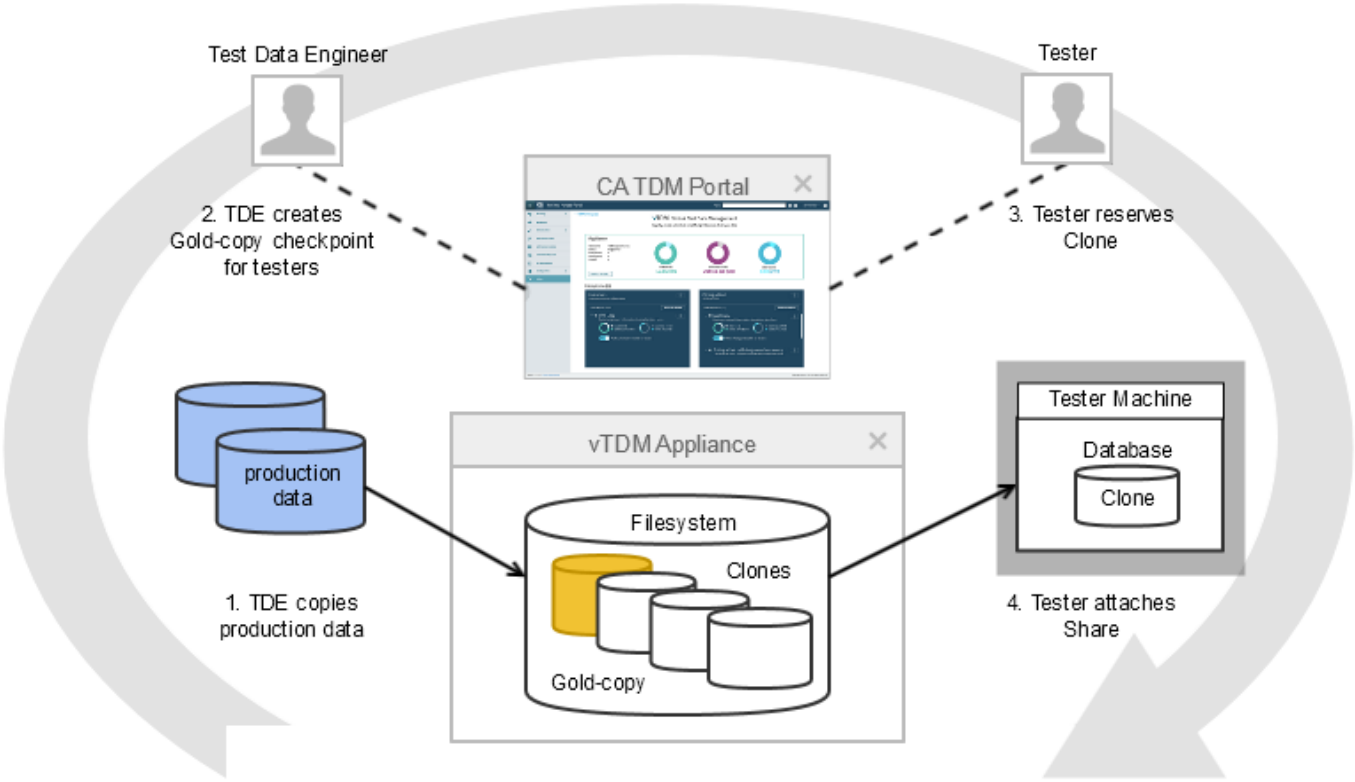
vTDM Architecture

vTDM Terminology

- **Clone** or **Virtual Data Source** refers to a virtual copy of a Filesystem or set of data.
- **Checkpoint** refers to a point in time for the data source or file system. You can take multiple checkpoints of one filesystem.
- **Production Database** refers to the database holding customers' live data, the root source for databases used in vTDM.
- **Gold-copy** refers to the masked (and possibly subsetted) database that is used as the basis for testing and Clones.
- **Appliance** refers to the Virtual Machine used to host the vTDM Filesystem. To enable vTDM, deploy an Appliance.
- **Data Source** refers to the database used for testing. This database is copied to the Filesystem and cloned.

vTDM Architecture

Figure 44: vTDM Architecture



The vTDM Appliance provides a network connected filesystems and clones environment. (1) A Test Data Engineer copies the database files to the filesystems, and (2) creates a checkpoint for testers. (3) After the testers create clones from the checkpoint, (4) they access the clones on the appliance network shares. When creating a checkpoint, the TDE specifies the DBMS on which the clone is mounted. This process is called automatic attachment of clones to database. If the TDE does not specify a database connection when creating a checkpoint, the clones for this checkpoint are flat files.

vTDM Considerations

The 4.5 release of vTDM has the following generic known limitations:

- Each CA TDM Portal instance supports only one vTDM appliance.
- The vTDM appliance is not designed to be moved to another instance of CA TDM Portal.
- By default, the vTDM appliance disk space is limited to 50GB. Add new disks to the vTDM Appliance to expand the disk space.
- vTDM supports the following databases for automatic attachment of clones to databases:
  - Microsoft SQL Server
  - Oracle 11g (Linux) Enterprise edition
  - Oracle 12c (Linux) Enterprise edition Pluggable
    Note: You can create clones in pluggable databases (PDBs) only. Creating clones in a root container is not supported.
    For more information about different architectures supported for Oracle 12c, refer to How to Copy Data from Oracle Database.
- For Oracle 11g source database, when creating a checkpoint, Oracle 11g and Oracle 12c are supported as the target database on which the clone is mounted.
- For Oracle 12c source database, when creating a checkpoint, only Oracle 12c is supported as the target database on which the clone is mounted.

vTDM Ports

The following ports are open on the vTDM Appliance by default:

Port	Type	Description
22	tcp	ssh - Needed for remote connection to the Appliance
67	udp	dhcp - Needed for DHCP control
68	udp	dhcp - Needed for DHCP control
111	tcp|udp	rpcbind - needed for NFS file sharing
445	tcp	microsoft-ds - needed for Windows file sharing
2049	tcp	nfs - needed for NFS file sharing
4045	tcp|udp	lockd - needed for NFS file sharing
8443	tcp	Appliance REST API

Install and Register the vTDM Appliance

The Test Data Engineer and System Administrator work together when installing the vTDM Appliance. You as Test Data Engineer use either the CA TDM Portal, or you script it using the vTDM REST API to register the vTDM Appliance. For more information about the API, see Use APIs to Manage and Consume vTDM Clones.

Prerequisites:

- Complete the installation of the CA TDM Portal before you install vTDM. For more information, see Install TDM Portal for Windows.
- A VMware ESXi 5.1 or later server managed by a vCenter server is required to host the appliance. VMware Workstation is not supported.
- A VMware vSphere Windows client connected to a vCenter server is required to deploy the OVA file.

Follow these steps:

Install the vTDM Appliance

Figure 45: vtdm test data engineer workflow

For more information about how to import and deploy OVA files, consult the vSphere documentation.

> **WARNING**
>
> There is no vTDM appliance for CA TDM 4.7. Use the OVA file from the vTDM appliance for CA TDM 4.6. You can find this package at support.broadcom.com/.

To import the OVA file:

1. Open the vSphere client and click **File, Deploy OVF Template**.
2. Fill in the following fields and click **Next** to proceed.
   - **Source** — Defines the OVA file.
   - **Name** — Defines the name of this virtual machine. For example, vTDMAppliance.
   - **Inventory Location** — Defines this VM's location in your data center.
   - **Host / Cluster** — Defines where you want to run this VM.
   - **Resource Pool** — Defines where within the hierarchy of the host/cluster to deploy this VM.
   - **Storage** — Defines where to store the VM files.
   - **Disk Format** — Specifies options how to store the virtual disks.
   - **Network Mapping** — Maps the network that is used in this VM to one of your **Destination Networks**.
3. Fill in the vTDM specific **Properties**:
   - **Root Password**Defines the password for the `root` account of the appliance. Administrators use this account to log in to the appliance. Default: `C45c4de$1`
   - **API Access Password**
     Defines the password for `vtdmadmin` account. Administrators use this account to access the appliance through the API.
     Default: `vtdmadmin`
   - **Share Access Password**
     Defines the password for the `vtdm` account. Testers and TDEs use this account to access the vTDM Filesystem.
     Default: `vtdm`
   - Configure IP Address, Netmask, and Route, or leave them blank to use DHCP.
     - **IP Address**
       Defines a static IP address for the appliance.
     - **Netmask**
       Defines your netmask in dotted quad notation, for example, 255.255.0.0.
     - **Route**
       Defines your default route (gateway) address.
   - **Hostname**
     Defines the hostname for the appliance.
   - **DNS Domain**

Defines your DNS domain, for example, company.com.

- **DNS Servers**
  Defines a comma-separated list of IP addresses of DNS servers.
- **Search DNS Domains**
  Defines a comma-separated list of IP addresses of DNS domains to search.

4. Review your deployment settings.Before you Power on the appliance, check if you have more than 50GB of test data and accordingly expand the disk space. For more information, see Configure Disk Capacity of the vTDM Appliance.
   - **Power on After Deployment**
     (Optional) Choose whether to automatically start the VM afterwards.
5. Click **Finish** and wait for the import to finish.
6. Start the VM.
7. (If required) Restart the VM to apply the hostname change.

Now the Appliance is ready for the Test Data Engineer to register it.

NOTE

The appliance is not designed to be moved to another instance of CA TDM Portal.

Configure Disk Capacity of the vTDM Appliance

By default the vTDM Appliance has 50GB of disk space. If you have more than 50GB of test data, you can expand the disk space by adding new disks to the vTDM Appliance.

TIP

We recommend that you add new disks to the appliance before it is Powered On for the first time.

To expand the appliance disk space after the first Power On, log in to the appliance.

Follow these steps:

- Add Disks to the vTDM Appliance
- Import Disks to the vTDM Appliance

NOTE

vTDM Appliance can continue running when adding disks to the appliance. Reboot of the vTDM Appliance is not required after expanding the disk space.

Add Disks to the vTDM Appliance

1. Log in to the vSphere client and select the vTDM Appliance virtual machine.
2. Right-click and navigate to **Edit Settings**.
   The Virtual Machine Properties dialog appears.
3. Click **Add**.
   The Add Hardware dialog opens.
4. Under **Device Type**, select **Hard Disk** and click **Next**.
5. Under **Select a Disk**, select the option **Create a new virtual disk** and click **Next**.
6. Under **Create a Disk**, select the required **Disk Size** capacity and **Disk Provisioning** option.
   For more information about disk provisioning options, refer to VMware vSphere documentation.
7. Click **Next** and **Finish** to apply your selections.

Now the disk is ready to be imported into the vTDM Appliance.

Import Disks into the vTDM Appliance

1. Log in to the vTDM Appliance as the root user.
   For example, SSH or right-click the vTDM Appliance and select the **Open Console** option in the vSphere client.
2. Run the following commands:
   a. `camcontrol rescan all` **Note:** Ignore the error message `"camcontrol: CAMIOCOMMAND ioctl failed: Invalid argument".`
   b. `/usr/local/vtdm/scripts/datapool.sh expand`
   The new disk has now been added to the vTDM Appliance.

Verify that Disks are Connected to the vTDM Appliance

To determine the disks that are connected to the vTDM Appliance, run the command `camcontrol devlist`

To obtain a list of disks that have not been imported in to the vTDM Appliance, run the command `/usr/local/vtdm/scripts/datapool.sh free-disks`

Register the Appliance using the CA TDM Portal

1. Open the CA TDM Portal.
2. Click vTDM, Get Started.
3. Enter the **Appliance Hostname**.
4. Enter your vTDM API Access **Password**.
   Default: vtdmadmin
5. Click **Register**.

The vTDM Appliance is now registered.

Unregister the Appliance using the CA TDM Portal

1. Open the CA TDM Portal.
2. Click vTDM.
3. Click the cog icon in the Appliance pane.
4. Click **Unregister**.

The vTDM Appliance is now unregistered.

If you encounter issues, continue with vTDM Troubleshooting.

To prepare and add data to use with vTDM, continue with Copy Data from vTDM Supported Data Sources.

Upgrade the vTDM Appliance

As part of the TDM upgrade process it may be required to upgrade the vTDM Appliance. You can either replace the vTDM Appliance with a new version or migrate to a new vTDM Appliance.

> **NOTE**
>
> The vTDM Appliance in version CA TDM 4.2 can only be replaced. From CA TDM version 4.3, the vTDM Appliance can be migrated to a higher version.

To replace the vTDM Appliance, follow these steps:

1. Unregister the vTDM Appliance from the upgraded instance of CA TDM Portal.
2. Deploy new version of the vTDM Appliance.
3. Register the new deployed vTDM Appliance with CA TDM Portal.

For more information about how to unregister, deploy, and register a vTDM Appliance, see Install and Register the vTDM Appliance.

WARNING

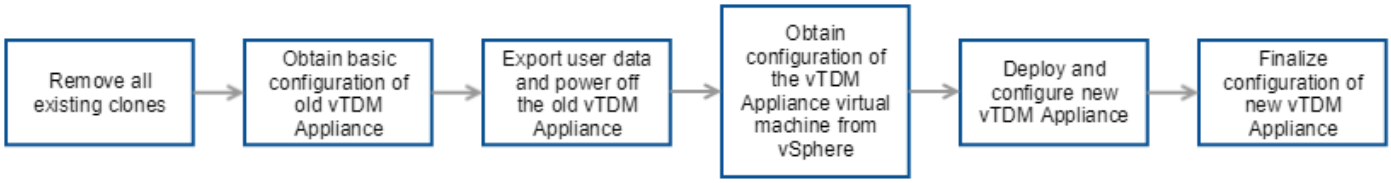
When you replace the vTDM Appliance all data such as filesystems, checkpoints, and clones are lost.

Migrating to a new vTDM Appliance preserves all the database files on filesystems with all checkpoints. However, before starting with migration you must remove all clones. For more information about how to migrate the vTDM Appliance, see Migrate the vTDM Appliance.

Migrate the vTDM Appliance

The migration procedure of the vTDM appliance preserves user data on existing filesystems including checkpoints. The basic flow for migrating the vTDM Appliance is as follows:

Figure 46: vTDM Appliance Migration Process



To migrate the vTDM Appliance from CA TDM 4.3 to later versions, follow these steps:

1. Remove All Existing Clones
2. Obtain Basic Configuration of the vTDM Appliance
3. Export User Data and Power Off the vTDM Appliance
4. Obtain Configuration of the vTDM Appliance Virtual Machine from vSphere
5. Deploy and Configure the New Version of the vTDM Appliance
6. Finalize Configuration of the New vTDM Appliance

NOTE

Do not create clones or perform any vTDM operations while the vTDM Appliance migration is in progress. Also, consider stopping the TDM Portal service to ensure a smooth migration.

Remove All Clones

You as Test Data Engineer use either the CA TDM Portal to remove clones from the vTDM Appliance, or you can write a script using the vTDM REST API. For more information about the API, see Use APIs to Manage and Consume vTDM Clones.

Example script: Remove Clones from vTDM Appliance on a Linux system

The following example script uses the vTDM API. Set the *PORTAL_URL*, *USER*, and *PASSWORD* variables to your values.

```
 #!/bin/sh
PORTAL_URL=https://tdm-portal:8443
USER=Administrator
PASSWORD=marmite
login() {
```

```
  local credentials="$(echo -n "$USER:$PASSWORD" | base64)"
  local header="Authorization: Basic $credentials"
  curl -s -k -H "$header" -X POST $PORTAL_URL/TestDataManager/user/login | jq -r .token
}
AUTH_HEADER="Authorization:Bearer $(login)"
(curl -H "$AUTH_HEADER" -X GET $PORTAL_URL/TDMvDataService/api/ca/v1/clones | jq .[].id) | while IFS= read -r
 id
do
    curl -H "$AUTH_HEADER" -X DELETE $PORTAL_URL/TDMvDataService/api/ca/v1/clones/$id &
done
```

Note: This example script requires the curl, jq, and base64 utilities installed.

Obtain Basic Configuration of the vTDM Appliance

To obtain the basic configuration of a vTDM Appliance, run the following commands. Copy the outputs off the vTDM Appliance on your local system as these outputs are referenced when configuring the new vTDM Appliance later in the migration procedure.

Follow these steps:

1. Log in to the vTDM Appliance.
2. Run the following command to obtain the vTDM Appliance properties:

   ```
   cat /usr/local/etc/default/vtdm
   ```

   Sample output:

   ```
   requestedHostname=vtdm-appliance
   gateway=
   ip=
   subnetMask=
   dnsDomainName=
   dnsServerList=
   dnsSearch=
   apiUser=vtdmadmin
   apiPassword=vtdmadmin
   shareUser=vtdm
   sharePassword=vtdm
   rootPassword=vtdm
   ```

3. Run the following command to obtain the vTDM Appliance network configuration:

   ```
   ifconfig
   ```

   Sample output:

   ```
   em0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
   options=9b<RXCSUM,TXCSUM,VLAN_MTU,VLAN_HWTAGGING,VLAN_HWCSUM>
   ether 00:50:56:94:80:5a
   inet 130.119.45.178 netmask 0xffffff00 broadcast 130.119.45.255
   nd6 options=29<PERFORMNUD,IFDISABLED,AUTO_LINKLOCAL>
   media: Ethernet autoselect (1000baseT <full-duplex>)
   status: active
   lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> metric 0 mtu 16384
   options=600003<RXCSUM,TXCSUM,RXCSUM_IPV6,TXCSUM_IPV6>
   inet6 ::1 prefixlen 128
   inet6 fe80::1%lo0 prefixlen 64 scopeid 0x2
   inet 127.0.0.1 netmask 0xff000000
   ```

```
nd6 options=21<PERFORMNUD,AUTO_LINKLOCAL>
groups: lo
```

4. Run the following commands to obtain the vTDM Appliance disk configuration:
   a. `/usr/local/vtdm/scripts/datapool.sh status`

      Sample output:

      ```
      NAME SIZE ALLOC FREE EXPANDSZ FRAG CAP DEDUP HEALTH ALTROOT
      database 209G 109M 209G - 0% 0% 1.00x ONLINE -
      da1 9.94G 27.4M 9.91G - 0% 0%
      da2 89.5G 27.3M 89.5G - 0% 0%
      da3 99.5G 26.6M 99.5G - 0% 0%
      da4 9.94G 27.2M 9.91G - 0% 0%
      ```

   b. `camcontrol devlist`

      Sample output:

      ```
      <NECVMWar VMware IDE CDR10 1.00> at scbus1 target 0 lun 0 (cd0,pass0)
      <VMware Virtual disk 1.0> at scbus2 target 0 lun 0 (pass1,da0)
      <VMware Virtual disk 1.0> at scbus2 target 1 lun 0 (pass2,da1)
      <VMware Virtual disk 1.0> at scbus2 target 2 lun 0 (pass3,da2)
      <VMware Virtual disk 1.0> at scbus2 target 3 lun 0 (pass4,da3)
      <VMware Virtual disk 1.0> at scbus2 target 4 lun 0 (pass5,da4)
      ```

Export User Data and Power Off the vTDM Appliance

To export the user data and power off the vTDM Appliance, log into the vTDM Appliance and run the following command:

```
zpool export database
poweroff
```

Obtain Configuration of the vTDM Appliance Virtual Machine from vSphere

To obtain the configuration of the vTDM Appliance virtual machine from vSphere, use the Virtual Machine Properties dialog.

Follow these steps:

1. Log into the vSphere client and select the relevant virtual machine.
2. Navigate to **Edit Settings**.
   The Virtual Machine Properties dialog displays.
3. Under the **Hardware** tab, select **Network adapter 1** and copy the MAC Address for later reference.
   Note: Ensure that the MAC address matches the MAC Address that you obtained from the network configuration.
   Example:

   ```
   00:50:56:94:80:5a
   ```

4. Select all Data Hard Disks starting from Hard Disk 2, and copy the values in the Disk File field.
   Note: Hard Disk 1 is a system disk. Do not copy this field.
   For example, the values in the Disk Field are as follows:

   ```
   [GM33P3J_ds1_raid0] vtdm-appliance/vtdm-appliance_1.vmdk
   [GM33P3J_ds1_raid0] vtdm-appliance/vtdm-appliance_2.vmdk
   [GM33P3J_ds1_raid0] vtdm-appliance/vtdm-appliance_3.vmdk
   [GM33P3J_ds1_raid0] vtdm-appliance/vtdm-appliance_4.vmdk
   ```

Deploy and Configure the New Version of the vTDM Appliance

You are now ready to deploy the OVF Template. For more information about installing the vTDM Appliance, see Install and Register the vTDM Appliance.

WARNING

Set all the vTDM Appliance properties for the new vTDM Appliance as you obtained them from the old Appliance in the previous steps.

Ensure that the properties are assigned as follows:

Old vTDM Appliance Properties	New vTDM Appliance Properties
rootPassword	Root Password
apiUser	API Access User
apiPassword	API Access Password
shareUser	Share Access User
sharePassword	Share Access Password
ip	IP Address
subnetMask	Netmask
gateway	Route
requestedHostname	Hostname
dnsDomainName	DNS Domain
dnsServerList	DNS Servers
dnsSearch	Search DNS Domains

NOTE

Do not power on the vTDM appliance yet.

Follow these steps:

1. Log in to the vSphere client and select the virtual machine for the new version of vTDM Appliance.
2. Navigate to **Edit Settings**.
   The **Virtual Machine Properties** dialog displays.
3. Under the **Hardware** tab, select **Hard disk 2** and click **Remove**.
   Wait for the Hard disk 2 to be successfully removed.
4. Click **Add**. The **Add Hardware** dialog is displayed.
5. Select **Hard Disk** and click **Next**.
6. Select **Use an existing virtual disk** and click **Next**.
   You are prompted to enter the disk file location.
   a. Enter the values that you obtained from the old Appliance virtual machine.
   b. Repeat this step for each data disk.
7. Under the **Hardware** tab, select **Network adapter 1** and select the **manual** option to restore the MAC Address manually.
   Note: Ensure that the MAC address matches the MAC Address that you obtained from the old vTDM Appliance.
8. Power on the Appliance.
   Note: Ensure that under **Options** tab, **Properties**, **Hostname** matches the Hostname that you obtained from the old vTDM Appliance.

Finalize Configuration of the New vTDM Appliance

1. Log into the new vTDM Appliance.
2. Run the following command to verify that no disks are configured:
   ```
   /usr/local/vtdm/scripts/datapool.sh status
   ```
   Expected output:
   ```
   vTDM data pool not found
   ```
3. Run the following command to import a data pool:
   ```
   zpool import database
   ```
4. Review the status of the data pool to ensure that the new vTDM Appliance configuration is same as the previous Appliance configuration.

To verify that the new vTDM Appliance configuration matches the previous version of the Appliance, run the basic configuration commands again, as described in this article, and compare the outputs. Also, confirm that the vTDM Appliance version on the vTDM dashboard is updated to 4.5.0.4.

The vTDM Appliance is migrated now. You can delete the previous version of the vTDM Appliance virtual machine in the vSphere client.

vTDM Administration

As a Test Data Engineer or Product Administrator, you are responsible for the configuration and maintenance of vTDM. This section includes administration information for vTDM.

Download a Log File of Clone Activity

You as Test Data Engineer can download a zipped **CSV** log file of recent clone creation and deletion activity.

Follow these steps:

1. Open the CA TDM Portal and click **My Clones**.
2. Click the **Chart** button to download a zipped **CSV** log file.

The **CSV** file includes details about:

- All currently active clones in vTDM.
- All clones that have been deleted in the last one year.

vTDM End-to-End Scenarios

vTDM provides a lightweight mechanism for Test Data Engineers to rapidly and cheaply provide Testers with access to Test Data. Instead of Testers sharing the same data source, they can access their own virtual copies of the test data. Each Clone of the Data Source is available near-instantaneously and takes virtually no space. More space is taken up only when a Tester changes data. Data copies can be automatically connected to a tester's database instance.

- Scenario for Microsoft SQL Server
- Scenario for Oracle 11g and Oracle 12c Linux

Scenario for Microsoft SQL Server

This example scenario uses Microsoft SQL Server. For more information about other supported databases, see vTDM Considerations.

This scenario addresses the following roles:

- Joe is a Test Data Engineer. He copies subsets of production data and provides it to testers.
- Jane is a tester. She uses the Self Service to clone the test data that was prepared by Joe.

This scenario covers the following workflow:

Provide Testers with Access to Test Data Quickly

Joe's development teams uses two-week sprints, but it takes him longer than this to gather the test data they need. His goal is to give them the data they need faster.

Joe copies a large Gold-copy database into vTDM. He uses vTDM as a lightweight mechanism to quickly make many copies available to Testers. Joe manages Gold-copies through the CA TDM Portal, and relies on the vTDM API when he wants to automate management through scripts.

Create a Filesystem

1. Open the CA TDM Portal.
2. Click vTDM.
3. Click **Add Filesystem**.
   a. Enter a name that allows your Testers to identify the Filesystem.
   b. (Optional) Enter a description.
   c. Click **Save**.

Copy Gold-copy Data to Filesystem

Joe has taken production data and masked it to remove any sensitive and other personal identifiable information. Joe connects the Filesystem to the Windows machine that hosts the SQL Server that contains this Gold-copy data, and copies the Gold-copy data to the Filesystem.

1. Open the CA TDM Portal
2. Click vTDM.
3. Click **Add Data** and follow the instructions in the dialog as the vdtm user.

1. a. Connect the share \\host\database_name.
   Example: Run the following command on the SQL Server host from a DOS command line:
   `net use \\my.vtdm.host\database_gold-copy`
   b. Enter the share access user name and password.
   Default: vtdm / vtdm. An administrator may have changed user name or password from the default.
   c. Take the Gold-copy data base offline to make sure it is not changed while you are copying the files.
   `ALTER DATABASE [gold-copy] SET OFFLINE WITH ROLLBACK IMMEDIATE`
   d. Determine the database files associated with 'gold-copy '.
   `USE [gold-copy]`
   `SELECT physical_name FROM sys.database_files` This gives the two files that you need copy to the filesystem.
   e. Copy those two files to the Filesystem.
   `copy <files>  \\my.vtdm.host\database_gold-copy`
   f. Bring the database back online.
   `ALTER DATABASE [gold-copy] SET ONLINE`

Now Joe is ready to checkpoint the Filesystem, and to make the Filesystem available to Testers.

Checkpoint the Gold-copy Data

Joe creates Checkpoints of the data that testers will want to clone. A Checkpoint refers to a point in time for the data source or file system. vTDM Checkpoints are not associated with projects. After the tester creates a Clone, the Clone is associated with the tester's current project.

Joe wants that database files are attached automatically to a SQL Server instance when testers create Clones of the checkpoint. For automatic attachment, he creates a user account that can connect to the SQL Server and has the necessary Server Roles to attach the database.

1. Open the CA TDM Portal
2. Click vTDM.
3. Click **New Checkpoint**.
   a. Enter a name that allows your testers to identify the check point.
   b. (Optional) Enter a description.
   c. Choose whether to make this checkpoint visible to testers.
   d. Configure the following options so that Clones of this checkpoint automatically attach to a tester's database:
      - **Automatically**
      - DBMS Server
      - (Optional) Port
      - (Optional) SQL Server Instance Name
      - Username — Defines the username of an account with appropriate SQL server roles.
      - Password — Defines the password for the same account.
   e. Click Test to verify the automatic database server connection.
   f. Click **Save**.

Joe has copied the Gold-copy database to the Filesystem, and set up the appliance for the Tester to use.

Joe verifies that he made the checkpoints visible by checking for the eye icon.

Joe verifies that checkpoints automatically attach by checking for a database icon.

Consume Shared Test Data

Jane and her fellow testers are often tripping over each other as they use a single shared copy of data for their testing. They would like to have their own private copies of test data, but manual copying takes too much time and space.

Jane communicates with Joe about the vTDM Filesystem and Checkpoints that she needs for her test cases, and Joe prepres the test data.

Create Clones

Jane recognizes a Gold-Copy in the Self Service Catalog by the vTDM symbol. If she doesn't see the Gold-copy she needs, she asks Joe to create it.

1. Open the CA TDM Portal and click **Self Service Catalog**.
2. Choose a project.
3. Identify the Gold-Copy that you need and click **New Clone**.
   a. Define a Clone Name.
   b. (Optional) Define a Clone description.
   c. Click **Create**.
   The new Clone is associated with the current project.
4. Review the details how to connect to this Clone.
5. (Optional) Click **Email Owner** to send a message with the connection details of a Clone to yourself, including a link to the instructions on how to manually attach a Clone to a database.

The Clones appear in the CA TDM Portal under **My Clones**.

View Clones

Each Clone is associated with a project. Open the CA TDM Portal and click **My Clones**. Here you can view details or delete Clones.

Jane, the Tester, only sees her own Clones. Joe, the Test Data Engineer, can view Clones created by any user.

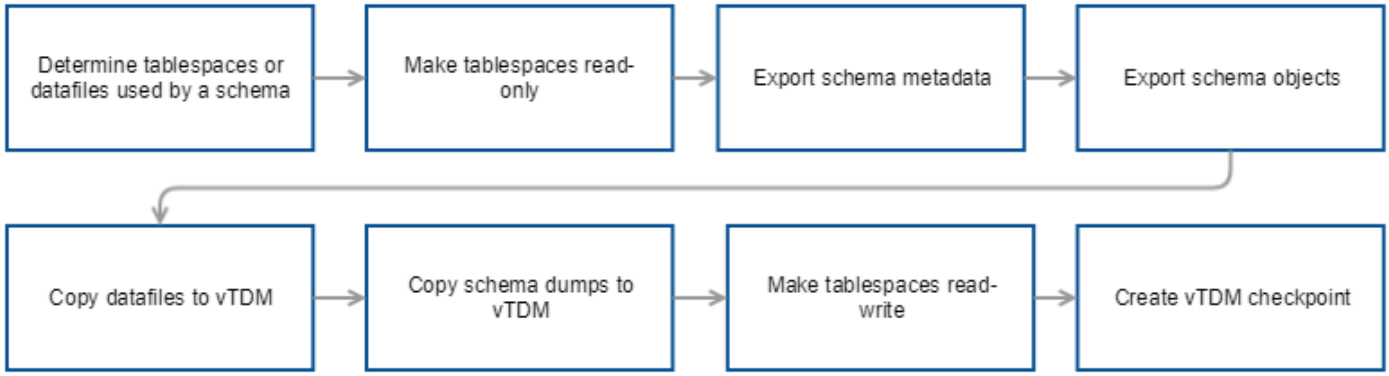
Jane identifies auto-attaching checkpoints in the Portal by a database icon.

After Jane has cloned the Gold-copy, the database files are attached automatically to her SQL Server instance.

Scenario for Oracle 11g and Oracle 12c Linux

The vTDM End-to-End Scenario for Oracle outlines the step-by-step process for a Test Data Engineer to export the Oracle schema and make it available in vTDM. The basic flow for exporting the Oracle schema is as follows:

Figure 47: TDE Oracle Export Flow



NOTE

This scenario applies to both, Oracle 11g (Linux) and Oracle 12c (Linux). For Oracle 12c (Linux), do not use *schemas* parameter to export Oracle schema. For more information, see Manage Gold-copies for Oracle Database.

Considerations

In this scenario, the schema name is *DEVUSER*, the user name is *OPERATOR* and the password is *password*. The *OPERATOR* user has the *exp_full_database* and *manage tablespace* permissions.

Use the Oracle SQL Developer to run the SQL commands and use a command shell to execute the export schema command `expdp` .

Procedure to Export Single Tablespace

Follow these steps:

1. Open Oracle SQL Developer.
2. Determine the datafiles and tablespaces used by the schema.
   `select UNIQUE(df.file_name), t.owner, t.tablespace_name from dba_tables t, dba_data_files df where t.owner='DEVUSER' and t.tablespace_name=df.tablespace_name;` The terminal prints the following output:

```
FILE_NAME                                    OWNER   TABLESPACE_NAME
/u01/app/oracle/oradata/orcl/devapp01.dbf DEVUSER DEVAPP
/u01/app/oracle/oradata/orcl/devapp02.dbf DEVUSER DEVAPP
```

3. Make the tablespace read-only.
   ```
   alter tablespace DEVAPP read only;
   ```
   The terminal prints the following output:

   ```
   tablespace DEVAPP altered.
   ```

4. Open a shell command prompt and export the schema:
   ```
   expdp operator/password schemas=DEVUSER logfile=devuser_ts.log dumpfile=devuser_ts.dmp
   transport_tablespaces=DEVAPP
   ```
   The terminal prints the following output:

   ```
   Export: Release 11.2.0.1.0 - Production on Thu Sep 14 09:09:24 2017
   Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.
   Connected to: Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit
    Production
   With the Partitioning, OLAP, Data Mining and Real Application Testing options
   Starting "OPERATOR"."SYS_EXPORT_TRANSPORTABLE_01": operator/******** schemas=DEVUSER
    logfile=devuser_ts.log dumpfile=devuser_ts.dmp transport_tablespaces=DEVAPP
   Processing object type TRANSPORTABLE_EXPORT/PLUGTS_BLK
   Processing object type TRANSPORTABLE_EXPORT/TABLE
   Processing object type TRANSPORTABLE_EXPORT/INDEX
   Processing object type TRANSPORTABLE_EXPORT/CONSTRAINT/CONSTRAINT
   Processing object type TRANSPORTABLE_EXPORT/INDEX_STATISTICS
   Processing object type TRANSPORTABLE_EXPORT/CONSTRAINT/REF_CONSTRAINT
   Processing object type TRANSPORTABLE_EXPORT/TABLE_STATISTICS
   Processing object type TRANSPORTABLE_EXPORT/POST_INSTANCE/PLUGTS_BLK
   Master table "OPERATOR"."SYS_EXPORT_TRANSPORTABLE_01" successfully loaded/unloaded
   ******************************************************************************
   Dump file set for OPERATOR.SYS_EXPORT_TRANSPORTABLE_01 is:
    /u01/app/oracle/admin/orcl/dpdump/devuser_ts.dmp
   ******************************************************************************
   Datafiles required for transportable tablespace DEVAPP:
    /u01/app/oracle/oradata/orcl/devapp01.dbf
    /u01/app/oracle/oradata/orcl/devapp02.dbf
   Job "OPERATOR"."SYS_EXPORT_TRANSPORTABLE_01" successfully completed at 09:09:56
   ```

   Note: For Oracle 12c (Linux), run the following command:
   ```
   expdp operator/password logfile=devuser_ts.log dumpfile=devuser_ts.dmp
   transport_tablespaces=DEVAPP
   ```
5. Open a shell command prompt and export the metadata.
   ```
   expdp operator/password schemas=DEVUSER logfile=devuser_meta.log
   dumpfile=devuser_meta.dmp
    include=FUNCTION,PACKAGE,PROCEDURE,SEQUENCE,SYNONYM,TYPE,VIEW,USER,ROLE_GRANT,SYSTEM_GRANT
   ```

The terminal prints the following output:

```
Export: Release 11.2.0.1.0 - Production on Thu Sep 14 09:15:41 2017
Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.
Connected to: Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit
 Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options
Starting "OPERATOR"."SYS_EXPORT_SCHEMA_01": operator/********
 schemas=DEVUSER logfile=devuser_meta.log dumpfile=devuser_meta.dmp
 include=FUNCTION,PACKAGE,PROCEDURE,SEQUENCE,SYNONYM,TYPE,VIEW,USER,ROLE_GRANT,SYSTEM_GRANT
Estimate in progress using BLOCKS method...
Processing object type SCHEMA_EXPORT/TABLE/TABLE_DATA
Total estimation using BLOCKS method: 0 KB
Processing object type SCHEMA_EXPORT/USER
Processing object type SCHEMA_EXPORT/SYSTEM_GRANT
Processing object type SCHEMA_EXPORT/ROLE_GRANT
ORA-39168: Object path FUNCTION was not found.
ORA-39168: Object path PACKAGE was not found.
ORA-39168: Object path PROCEDURE was not found.
ORA-39168: Object path SEQUENCE was not found.
ORA-39168: Object path SYNONYM was not found.
ORA-39168: Object path TYPE was not found.
ORA-39168: Object path VIEW was not found.
Master table "OPERATOR"."SYS_EXPORT_SCHEMA_01" successfully loaded/unloaded
******************************************************************************
Dump file set for OPERATOR.SYS_EXPORT_SCHEMA_01 is:
 /u01/app/oracle/admin/orcl/dpdump/devuser_meta.dmp
Job "OPERATOR"."SYS_EXPORT_SCHEMA_01" completed with 7 error(s) at 09:15:50
```

Note: You can ignore the error messages displayed for each type of metadata that is not found.

6. From the Oracle SQL Developer, obtain the location of dump files.
   ```
   select directory_path from dba_directories where directory_name='DATA_PUMP_DIR';
   ```
   The terminal prints the following output:

   ```
   /u01/app/oracle/admin/orcl/dpdump/
   ```

7. Mount the vTDM Appliance filesystem on a local directory. This step assumes that you have created a filesystem called 'oracle' already.

   ```
   mkdir -p /vtdm/mnt/oracle
   mount -t nfs -o rw,timeo=600,hard,wsize=32768,vers=3,rsize=32768 vtdm-
   appliance.domain.com:/database/oracle /vtdm/mnt/oracle
   ```

8. Copy datafiles, dump files and logfiles to the vTDM share.

   ```
   cp /u01/app/oracle/admin/orcl/dpdump/devuser* /vtdm/mnt/oracle
   ```

```
cp /u01/app/oracle/oradata/orcl/devapp01.dbf /u01/app/oracle/oradata/orcl/
devapp02.dbf /vtdm/mnt/oracle
```

9. From the Oracle SQL Developer, make the tablespace read-write.
   `alter tablespace DEVAPP read write;`
   The terminal prints the following output:

```
tablespace DEVAPP altered.
```

You can now create a checkpoint for the filesystem and start cloning. For more information, see Checkpoint the Gold-copy Data for Oracle.

Procedure to Export Multiple Tablespaces

Follow these steps:

1. Open Oracle SQL Developer.
2. Determine the datafiles and tablespaces used by the schema.
   `select UNIQUE(df.file_name), t.owner, t.tablespace_name from dba_tables t, dba_data_files df where t.owner='DEVUSER' and t.tablespace_name=df.tablespace_name;`
   The terminal prints the following output:

```
FILE_NAME                                    OWNER   TABLESPACE_NAME
/u01/app/oracle/oradata/orcl/devapp01.dbf DEVUSER DEVAPP
/u01/app/oracle/oradata/orcl/devapp02.dbf DEVUSER DEVAPP
/u01/app/oracle/oradata/orcl/devapp03.dbf DEVUSER DEVAPP2
```

3. Make the tablespaces read-only.

```
alter tablespace DEVAPP read only;
alter tablespace DEVAPP2 read only;
```

   The terminal prints the following output:

```
tablespace DEVAPP altered.
tablespace DEVAPP2 altered.
```

4. Open a shell command prompt and export the schema.
   `expdp operator/password schemas=DEVUSER logfile=devuser_ts2.log dumpfile=devuser_ts2.dmp transport_tablespaces=DEVAPP,DEVAPP2`
   The terminal prints the following output:

```
Export: Release 11.2.0.1.0 - Production on Fri Sep 15 10:20:51 2017
Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.
Connected to: Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit
 Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options
```

```
Starting "OPERATOR"."SYS_EXPORT_TRANSPORTABLE_01": operator/********
 schemas=DEVUSER logfile=devuser_ts2.log dumpfile=devuser_ts2.dmp
 transport_tablespaces=DEVAPP,DEVAPP2
Processing object type TRANSPORTABLE_EXPORT/PLUGTS_BLK
Processing object type TRANSPORTABLE_EXPORT/TABLE
Processing object type TRANSPORTABLE_EXPORT/INDEX
Processing object type TRANSPORTABLE_EXPORT/CONSTRAINT/CONSTRAINT
Processing object type TRANSPORTABLE_EXPORT/INDEX_STATISTICS
Processing object type TRANSPORTABLE_EXPORT/CONSTRAINT/REF_CONSTRAINT
Processing object type TRANSPORTABLE_EXPORT/TABLE_STATISTICS
Processing object type TRANSPORTABLE_EXPORT/POST_INSTANCE/PLUGTS_BLK
Master table "OPERATOR"."SYS_EXPORT_TRANSPORTABLE_01" successfully loaded/unloaded
******************************************************************************
Dump file set for OPERATOR.SYS_EXPORT_TRANSPORTABLE_01 is:
 /u01/app/oracle/admin/orcl/dpdump/devuser_ts2.dmp
******************************************************************************
Datafiles required for transportable tablespace DEVAPP:
 /u01/app/oracle/oradata/orcl/devapp01.dbf
 /u01/app/oracle/oradata/orcl/devapp02.dbf
Datafiles required for transportable tablespace DEVAPP2:
 /u01/app/oracle/oradata/orcl/devapp03.dbf
Job "OPERATOR"."SYS_EXPORT_TRANSPORTABLE_01" successfully completed at 10:21:22
```

Note: For Oracle 12c (Linux), run the following command:

```
expdp operator/password logfile=devuser_ts2.log dumpfile=devuser_ts2.dmp
transport_tablespaces=DEVAPP,DEVAPP2
```

5. Open a shell command prompt and export the metadata.

```
expdp operator/password schemas=DEVUSER logfile=devuser_meta2.log
dumpfile=devuser_meta2.dmp
 include=FUNCTION,PACKAGE,PROCEDURE,SEQUENCE,SYNONYM,TYPE,VIEW,USER,ROLE_GRANT,SYSTEM_GRANT
```

The terminal prints the following output:

```
Export: Release 11.2.0.1.0 - Production on Fri Sep 15 10:39:18 2017

Copyright (c) 1982, 2009, Oracle and/or its affiliates.  All rights reserved.

Connected to: Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit
 Production

With the Partitioning, OLAP, Data Mining and Real Application Testing options

Starting "OPERATOR"."SYS_EXPORT_SCHEMA_01":  operator/********
 schemas=DEVUSER logfile=devuser_meta2.log dumpfile=devuser_meta2.dmp
 include=FUNCTION,PACKAGE,PROCEDURE,SEQUENCE,SYNONYM,TYPE,VIEW,USER,ROLE_GRANT,SYSTEM_GRANT

Estimate in progress using BLOCKS method...

Processing object type SCHEMA_EXPORT/TABLE/TABLE_DATA
```

```
Total estimation using BLOCKS method: 0 KB

Processing object type SCHEMA_EXPORT/USER

Processing object type SCHEMA_EXPORT/SYSTEM_GRANT

Processing object type SCHEMA_EXPORT/ROLE_GRANT

ORA-39168: Object path FUNCTION was not found.

ORA-39168: Object path PACKAGE was not found.

ORA-39168: Object path PROCEDURE was not found.

ORA-39168: Object path SEQUENCE was not found.

ORA-39168: Object path SYNONYM was not found.

ORA-39168: Object path TYPE was not found.

ORA-39168: Object path VIEW was not found.

Master table "OPERATOR"."SYS_EXPORT_SCHEMA_01" successfully loaded/unloaded

******************************************************************************

Dump file set for OPERATOR.SYS_EXPORT_SCHEMA_01 is:

  /u01/app/oracle/admin/orcl/dpdump/devuser_meta2.dmp

Job "OPERATOR"."SYS_EXPORT_SCHEMA_01" completed with 7 error(s) at 10:39:27
```

Note: You can ignore the error messages displayed for each type of metadata that is not found.

6. From the Oracle SQL Developer, obtain the location of dump files.
   `select directory_path from dba_directories where directory_name='DATA_PUMP_DIR';`
   The terminal prints the following output:

   ```
   /u01/app/oracle/admin/orcl/dpdump/
   ```

7. Mount the vTDM Appliance filesystem on a local directory. This step assumes that you have created a filesystem called 'oracle' already.

   ```
   mkdir -p /vtdm/mnt/oracle

   mount -t nfs -o rw,timeo=600,hard,wsize=32768,vers=3,rsize=32768 vtdm-
   appliance.domain.com:/database/oracle /vtdm/mnt/oracle
   ```

8. Copy datafiles, dump files and logfiles to the vTDM share.

```
cp /u01/app/oracle/admin/orcl/dpdump/devuser*  /vtdm/mnt/oracle

cp /u01/app/oracle/oradata/orcl/devapp01.dbf   /u01/app/oracle/oradata/orcl/
devapp02.dbf  /u01/app/oracle/oradata/orcl/devapp03.dbf /vtdm/mnt/oracle
```

9. From the Oracle SQL Developer, make the tablespace read-write.

```
alter tablespace DEVAPP read write;

alter tablespace DEVAPP2 read write;
```

The terminal prints the following output:

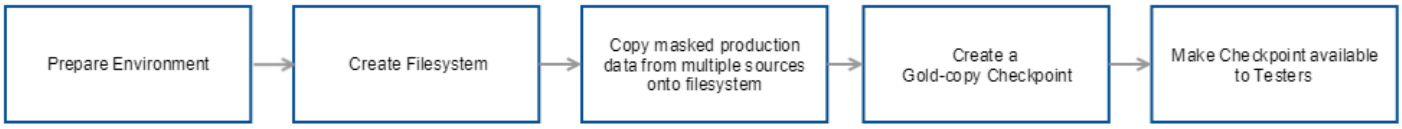
```
tablespace DEVAPP altered.

tablespace DEVAPP2 altered.
```

You can now create a checkpoint for the filesystem and start cloning. For more information, see Checkpoint the Gold-copy Data for Oracle.

Copy Data from vTDM Supported Data Sources

You have now installed and registered the Appliance, and you are now ready to prepare data to use with vTDM. The basic flow diagram for copying data from vTDM supported data sources is as follows:



Follow the respective process for the following supported data sources:

- How to Copy Data from Microsoft SQL Server
- How to Copy Data from Oracle Database
- How to Copy Data from Flat Files

How to Copy Data from Microsoft SQL Server

vTDM allows you to manage multiple copies of Microsoft SQL Server test data. In order to use Microsoft SQL Server with vTDM, perform the following procedures:

1. Prepare the Environment for Microsoft SQL Server
2. Manage Gold-copies for Microsoft SQL Server
3. Checkpoint the Gold-copy Data for Microsoft SQL Server
4. Consume Gold-copy Clones with vTDM

The Maintenance and Recovery Operations for SQL Server.

Operating Systems

vTDM supports Auto-attachment of SQL Server database files to SQL Server DBMS instances, on all Microsoft Windows supported operating systems.

Prepare the Environment for Microsoft SQL Server

Before you begin copying the Gold-copy database into vTDM, prepare your environment. You must perform certain configuration settings for Microsoft SQL Server.

Configure the SQL Server Service to Run as Local User

By default, the SQL Server Service runs with the virtual account `NT Service/SQLSERVER`, which does not have access to network shares. Ensure that you configure the service to run as a *local* user.

Follow these steps:

1. Launch the **SQL Server Configuration Manager**.
2. Click **SQL Server Services**, and open **SQL Server (MSSQLSERVER)**.
   The **Properties** panel opens.
3. Go to the **Log On** tab.
4. Configure it to **Log on as... This Account**.
   a. Enter the **Account Name** of a local SQL user, for example, "**.**`\SQLUser`".
   b. Enter the **Password**, and re-enter it to confirm the password.
   c. Click **Apply.**
5. Restart the SQL Server.

The SQL Server Service is configured to run as a local user. You can now copy the Gold-copy database into vTDM and make it available for cloning to the Tester. You can now manage Gold-copies for Microsoft SQL Server.

Manage Gold-copies for Microsoft SQL Server

As a Test Data Engineer, you copy the Gold-copy database into vTDM and make it available for cloning to the Tester. You can manage Gold-copies through either the CA TDM Portal or the vTDM API. We recommend using the CA TDM Portal, and you have the option to use the REST API from external automation scripts.

Prerequisites:

- Install and Register the vTDM Appliance

Follow these steps:

Create a Filesystem

Each appliance supports several filesystems.

1. Open the CA TDM Portal.
2. Click vTDM.
3. Click **Add Filesystem**.
   a. Enter a name that allows your Testers to identify the Filesystem.
   b. (Optional) Enter a description.
   c. Click **Save**.

Copy Gold-copy Data to Filesystem

As Test Data Engineer, you have taken production data and masked it to remove any sensitive and other personal identifiable information. In this step, you connect the Filesystem to the Windows machine that hosts the SQL Server that contains this Gold-copy data. Finally, you copy the Gold-copy data to the Filesystem.

Follow these steps:

1. Open the CA TDM Portal.
2. Click vTDM.
3. Click **Add Data** and follow the instructions in the dialog.

1. a. Connect to the share `\\my.vtdm.host\database_name` as the `vdtm` user.
   Example: Open an Administrator Command Prompt on the SQL Server host and run the following command:
   `net use z: \\my.vtdm.host\database_nam e vtdm /user:vtdm`
   b. Enter the vTDM Share Access user name and password.
   Default: vtdm / vtdm. An administrator may have changed user name or password from the default. See Install and Register the vTDM Appliance for further details.
   c. Open Microsoft SQL Studio.
   d. Determine the database files associated with 'gold-copy '.
   `USE [gold-copy]`
   `SELECT physical_name FROM sys.database_files` Identify the two files that you need to copy.
   e. Take the Gold-copy data base offline to make sure it is not changed while you are copying the files.
   `ALTER DATABASE [gold-copy] SET OFFLINE WITH ROLLBACK IMMEDIATE`
   f. Copy those two files to the Filesystem.
   Example: Open an Administrator Command Prompt on the SQL Server host and run the following command:
   `copy \\my-server\TravelDB* \\my.vtdm.host\database_name`
   g. Bring the database back online.
   `ALTER DATABASE [gold-copy] SET ONLINE`

Now you are ready to checkpoint the Filesystem and start making the Filesystem available to Testers to clone.

Checkpoint the Gold-copy Data for Microsoft SQL Server

You as Test Data Engineer create Checkpoints of the data that testers want to clone. A Checkpoint refers to a point in time for the data source or file system. vTDM Checkpoints are not associated with projects. After the tester creates a Clone, it is associated with the tester's current project.

You can choose to let Testers mount the Clone to their database server manually. Alternatively, you can specify that database files are attached automatically to a SQL Server instance when testers create Clones of the checkpoint.

For automatic attachment, for SQL Server, create a user account that has the following **Server Roles** to attach the database:

1. • dbcreator
   • public
   • serveradmin
   • sysadmin

Follow these steps:

1. Open the CA TDM Portal
2. Click vTDM.
3. Click **New Checkpoint**.
   a. Enter a name that allows your testers to identify the checkpoint.
   b. (Optional) Enter a description.

c. Choose whether to make this checkpoint visible to testers.
Tip: You identify visible checkpoints in the Portal by an eye icon.

4. Click **Next**.
   a. Choose **Microsoft SQL Server** as the DBMS Type option for how the Testers attach a Clone of this checkpoint to a database server.
   Note: The initial database selection in the Create Checkpoint dialog is based on the database file types in your filesystem.
   b. Fill in the following fields:
   Tip: You identify auto-attaching checkpoints in the Portal by a database icon.
   - • DBMS Server — Choose Microsoft SQL Server
     • (Optional) Port
     • (Optional) SQL Server Instance Name
     • Username — Defines the username of an account with appropriate server roles.
     • Password — Defines the password for the same account.
   - **None (Manual)**—Allows the Testers to choose how the Clone is consumed. The user interface provides Testers with information, username, and password, to mount the Clone to their Database Server manually.

   c. (Optional) Click **Test** to verify the automatic database server connection.
   Note: The test operation creates a clone and attempts to connect the clone to the specified SQL Server database. Ensure that the SQL Server is online during the test operation. This test operation validates the DBMS Server, Port, SQL Server Instance Name, Username and Password values.
   d. Click **Save**.

You have copied the Gold-copy database to the Filesystem, and set up the checkpoint for the Tester to use. For more information about how Tester can make copies of the Gold-copy data, see Consume Gold-copy Clones with vTDM.

NOTE

When you create a checkpoint and get the following error message

```
Checkpoint validation failed:

Details: SQL Server (MSSQLSERVER) service is using local system account (NT
Service\MSSQLSERVER). Such scenario is not supported for DB auto attach.
```

then ensure to configure the SQL Service as described in Prepare the Environment for Microsoft SQL Server.

Maintenance and Recovery Operations for SQL Server

This section describes the maintenance and recovery procedures that you use for vTDM with a Microsoft SQL Server database. We present several different types of maintenance scenarios with the appropriate recovery procedures to use.

Restarting SQL Server Database Service

If you restart a SQL Server database service or this service is automatically restarted, the SQL Server database service automatically connects to the vTDM appliance. No action is required in this scenario.

Restarting the vTDM Appliance

When the vTDM Appliance automatically restarts, the SQL Server database attempts to update a table within the Clone. An unexpected network error message is displayed. To resolve this error, after the vTDM Appliance has restarted, take the clone offline and bring it back online. Use *one* of the following ways:

- SQL Server Management Studio
  a. Right-click the vTDM Clone Database, select **Tasks** and click **Take Offline**.
  The Take Database Offline dialog opens.

b. Select the **Drop All Active Connections** checkpoint and click **OK**.
After the offline action is completed, vTDM Clone is marked as offline in the SQL Server Management Studio.
c. Right-click the vTDM Clone, select **Tasks** and click **Bring Online**.
The vTDM Clone is now online.

- SQL Server Command Line
Run the following two commands from the SQL Server command line prompt.
`alter database [clone_name] set offline with rollback immediate; alter database [clone_name] set online;`
Replace the *clone_name* with your vTDM Clone name.

The vTDM Clone is now online and usable.

Restarting the SQL Server Host

If you restart a SQL Server Host or it restarts automatically, the SQL Server Host connects to the automatically attached Clones without any errors. No action is required in this scenario.

Deleting the vTDM Appliance

When the vTDM Appliance is permanently unavailable or you are unable to access the vTDM appliance from CA TDM Portal, remove the vTDM Clones manually from the SQL Server. Use *one* of the following ways:

- SQL Server Management Studio:
  a. Right-click the vTDM Clone, select **Tasks** and click **Detach...**The **Detach Database** dialog displays.
  b. Select **Drop Connections** checkpoint and click **OK**.
  c. Wait for the action to be completed.
  An unexpected network error message is displayed indicating that the database cannot be written to.
  d. Ignore this error message and click **OK**.
  e. In the **Detach Database** dialog, click **OK**.
- SQL Server Command Line:
  a. Run the following commands from the SQL Server command line prompt.
  `EXEC sp_detach_db 'clone_name', 'true';`
  Replace the *clone_name* with your vTDM Clone name.
  b. Wait for the command to complete. You can ignore the unexpected network error message.

How to Copy Data from Oracle Database

vTDM allows you to manage multiple copies of Oracle test data and supports the following Oracle databases for automatic attachment of clones to databases:

- Oracle 11g (Linux) Enterprise edition
- Oracle 12c (Linux) Enterprise edition
  – **Non-Container Database (CDB)**Supported same as Oracle 11g (Linux) Enterprise edition.
  – **Single tenant CDB configuration**
  For a container database with a single pluggable database, you can create clones in the pluggable database (PDB) only.
  – **Multitenant CDB configuration**
  For a container database with multiple pluggable databases, you can create clones in the pluggable databases (PDBs) only.

  Note: Creating clones in system database and root container is not supported.

To use Oracle database with vTDM, perform the following procedures:

1. Prepare the Environment for Oracle Database
2. Manage Gold-copies for Oracle Database
3. Checkpoint the Gold-copy Data for Oracle
4. Consume Gold-copy Clones with vTDM

The Maintenance and Recovery Operations for Oracle.

Operating Systems

vTDM supports Oracle Gold-copy filesystems and automatically attached Oracle Cloned filesystems on the following Linux operating systems only:

- – Linux x86
  – Linux x86-64

  NOTE

  Microsoft Windows operating system is not supported for Oracle database in this release.

Oracle Database Support Considerations

Using Oracle database with vTDM has the following known limitations:

- To transport a tablespace from one platform to another, datafiles on different platforms must be in the same endian format.
- The source and target database must use the same character set and national character set.
- Only a single schema can be imported or exported at a time.
- Import of tablespaces with more than one schema is not supported.
- Encrypted tablespaces have the following limitations:
  – Before you transport an encrypted tablespace, identify the location of master encryption key. Copy the Oracle wallet or the Hardware Security Module (HSM) that includes the master encryption key to the destination database. The Oracle wallet password remains unchanged. For more information, see Oracle Database Advanced Security Administrator's Guide.
  – You cannot transport an encrypted tablespace to a database that contains an Oracle wallet for transparent data encryption. Use Oracle Data Pump to export schema objects of a tablespace and import them to the destination database. For more information about Oracle Data Pump, see Oracle Database Utilities.

Prepare the Environment for Oracle Database

Before you begin copying the Gold-copy database into vTDM, prepare your environment. You must provide certain user permissions to the privileged user, create a non-system Oracle user, and configure the number of datafiles for your Oracle instance.

Prerequisite for Exporting Oracle Database Schema

As a Test Data Engineer, you must manually export the schema metadata and datafiles from the source Oracle database. Use the Oracle Data Pump utility to export this data. The exported schema metadata and datafiles are imported into the vTDM filesystem using different names for the schema and tablespace. This supports the same schema on a single Oracle database. The exported schema metadata and datafiles together form the Gold-copy.

Before exporting the schema metadata and datafiles, follow these steps:

1. Open Oracle SQL Developer.
2. Determine the tablespaces and datafiles that are used by a user or schema.

```
select df.file_name, t.owner, t.tablespace_name from dba_tables t, dba_data_files df
where t.owner='schema' and t.tablespace_name=df.tablespace_name;
```

This query returns the file name, owner, and tablespace name.

Example:

```
FILE_NAME                                  OWNER      TABLESPACE
------------------------------------------ ---------- ----------
/u01/app/oracle/oradata/orcl/example.dbf Test       Example
```

3. Ensure that objects in a tablespace have no reference to objects in a tablespace that is not exported.
   `exec SYS.DBMS_TTS.TRANSPORT_SET_CHECK(ts_list => '`*`TABLESPACENAME`*`', incl_constraints => TRUE);`
   Wait for this operation to complete.
4. Verify if any violation has occurred.
   `select * from transport_set_violations;`
   If there are no violations, continue with Manage Gold-copies for Oracle Database.
   If there are violations, consider using a larger set of tablespaces and continue with step 2.

You are now ready to export the schema metadata and datafiles.

Requirements for Mounting Filesystems and Clones

To automatically mount and unmount filesystems and clones, certain privileges are required. Ensure that you use a Linux user account that has privileges to mount and unmount NFS filesystems. The privileged user is required when performing the following tasks:

- copy data to the vTDM filesystem
- automatically attach a clone when creating a checkpoint

Set up a User Account with Minimum Privileges

Complete the following steps as the root user to ensure that the privileged user has mount and unmount permissions for NFS filesystems:

Note: The example user account in the following steps is `vtdm`.

1. Create a standard Linux user account.
   Example: To create a new user called *vtdm* run the following commands:

```
useradd vtdm

passwd vtdm

usermod -aG wheel vtdm
```

2. Edit .bashrc (or equivalent non-interactive login script) and set ORACLE_HOME to point to the directory containing the oracle instance. Do *not* use the oracle shell script oraenv for this as it is interactive.
   Note: vTDM uses a SSH connection to the oracle dbms machine and logs in as the non-root user. This also invokes the non-interactive login script, for example, .bashrc on Linux. This script must *not* include anything interactive as this causes vTDM to fail when trying to create checkpoints or clones.
3. Define aliases for the mount and umount commands so that they are run as root using sudo. Typically, only root can use these commands, not ordinary users.
   Example: For `.bashrc`

```
# .bashrc

PATH=$PATH:$HOME/bin
export PATH
ORACLE_HOME=/u01/app/oracle/product/11.2.0/db_1
export ORACLE_HOME



# Define aliases to allow non-root user to run mount and umount

shopt -s expand_aliases
alias mount='sudo -n mount'
alias umount='sudo -n umount'
```

4. Configure sudo so that:
   - sudo can be executed from a non-interactive shell
   - requiretty is disabled
   - the new user can execute mount and unmount

   To do this, add the following to the sudoers configuration file using visudo:

```
Defaults !requiretty

vtdm    ALL= NOPASSWD: /bin/mount, /bin/umount
```

Use an Existing User Account for vTDM

vTDM uses an SSH connection to connect to the Oracle database machine. To use an existing user (for example, root), you define ORACLE_HOME in the non-interactive login script (for example, .bashrc) on the Oracle database machine.

Create the Mount Directory

Create a mount directory for vTDM mount points (NFS). Ensure that the mount directory is accessible by the user who is required to mount vTDM clones to the Oracle database and set the permissions to allow the new user to mount vTDM clones.

For example, assume that a `vtdm` user is already created and is used for mounts, by default it has the primary group ‘vtdm’ on Linux.

```
mkdir –p  /vtdm/mnt
chgrp vtdm /vtdm/mnt
chmod 775 /vtdm/mnt
chmod 755 /vtdm
```

Create a Non-system Oracle User for vTDM

A non-system Oracle user is required to export the database schema to vTDM and import the database clone from vTDM.

For exporting the Oracle database schema, a non-system user with the following permissions is required:

- Alter or Manage Tablespace
- Export Full database role (EXP_FULL_DATABASE)
- Tablespace Quota for Users Tablespace

For automatic and manual attachment, for Oracle Server, a non-system user with the following permissions is required:

- Alter or Manage Tablespace
- Import Full database role (IMP_FULL_DATABASE)
- Tablespace Quota for Users Tablespace

Follow these steps:

Run the following commands to create a non-system user:

1. Open sqlplus with create user and grant privileges.
   `sqlplus sys/password as sysdba;`
2. Create a non-system user *cloner*:
   - For Oracle 11g, run the following command:
     `CREATE USER cloner IDENTIFIED BY password;`
   - For Oracle 12c, first specify where you want to create the user.
     For example, `ALTER SESSION SET CONTAINER = PDBORCL;`
     Run the following command to create the user
     CREATE USER *cloner* IDENTIFIED BY *password*;

   Note: To obtain a list of all PDBs, run the following command:
   `SELECT NAME, OPEN_MODE FROM V$PDBS;`
3. Run the following command:
   - to provide export full database role:
     `GRANT exp_full_database to cloner;`
   - to provide import full database role: `GRANT imp_full_database to cloner;`
4. Run the following command to provide alter tablespace permissions:
   `ALTER USER cloner QUOTA 10m ON USERS;`
5. Run the following command to provide manage tablespace permissions:
   `GRANT MANAGE TABLESPACE to cloner;`

The new user *cloner* you can now use for exporting the database schema or importing the database clone from vTDM .

Configure the Oracle Server to Increase Number of Datafiles

When connecting a large number of clones to an Oracle database, you may exceed the maximum number of datafiles that are configured for your Oracle instance. You can determine the number of datafiles on your checkpoint, and your expected number of concurrent clones, and can adjust the maximum number of datafiles configurations using the steps below.

To calculate the required number of datafiles use the following equation:

The required number of datafiles for clones > (number of datafiles in checkpoint)*(expected number of concurrent clones)

For example, if you have 15 datafiles on a checkpoint with 20 concurrent clones, the minimum required number of datafiles for clones is 300.

TIP
Configure the Oracle database to a larger number of datafiles than the required number of datafiles.

The following steps include restarting the Oracle server, so ensure that you are not connecting using a Transparent Network Substrate (TNS) listener, as this connection is terminated when the service is shut down.

1. Switch the current user to Oracle.
   ```
   su oracle
   ```
2. Connect to sqlplus as follows:
   ```
   export ORACLE_SID=orcl
   sqlplus / as sysdba
   ```
3. Determine the number of allowed datafiles.
   ```
   select value from v$parameter where name = 'db_files';
   ```
   This query returns the maximum number of datafiles that are allowed in the Oracle database.
   Example:
   VALUE = 200
4. Update the number of allowable datafiles.
   ```
   alter system set db_files=400 scope=spfile;
   ```
5. Stop and restart the oracle service to save the changes.
   ```
   shutdown immediate;
   startup;
   ```
6. Verify the updated allowable datafiles.
   ```
   select value from v$parameter where name = 'db_files';
   ```
   This query returns the updated number of datafiles that are allowed in the Oracle database.
   Example:
   VALUE = 400
7. Exit sqlplus.
   ```
   quit
   ```

You have now increased the maximum allowed number of datafiles for your Oracle Instance. You can now copy?the?Gold-copy?database into vTDM and make it available for cloning to the Tester.

Manage Gold-copies for Oracle Database

As a Test Data Engineer, you copy the Gold-copy database into vTDM and make it available for cloning to the Tester. You can manage Gold-copies through either the CA TDM Portal or the vTDM API. We recommend using the CA TDM Portal, and you have the option to use the REST API from external automation scripts.

Prerequisites:

- Install and Register the vTDM Appliance

Follow these steps:

Create a Filesystem
Each appliance supports several filesystems.

1. Open the CA TDM Portal
2. Click vTDM.
3. Click **Add Filesystem**.
   a. Enter a name that allows your Testers to identify the Filesystem.
   b. (Optional) Enter a description.
   c. Click **Save**.

Copy Gold-copy Data to Filesystem

In this step, you connect the Filesystem to the Linux machine that hosts the Oracle Server that contains this Gold-copy data. Finally, you copy the Gold-copy data to the Filesystem. Ensure that you complete the prerequisites for Oracle database before creating a checkpoint. Ensure that you have prepared the environment for exporting Oracle database.

Follow these steps:

1. Open the CA TDM Portal.
2. Click vTDM.
3. Click **Add Data** and follow the instructions in the dialog.
4. Check Prerequisites.
5. Export Schema Data.
6. Copy Database Dumps to vTDM.

Check Prerequisites

1. Create a mount directory for the vTDM mount points.
   `mkdir /vtdm/mnt/filesystemname`
2. Mount the network share. Open a root terminal on the Oracle Server host and run the following command:
   `mount -t nfs -o rw,timeo=600,hard,wsize=32768,vers=3,rsize=32768 my.vtdm.host:/database/filesystemname /vtdm/mnt/filesystemname`
3. Connect to the Oracle instance using SQL Developer or SQL*Plus. Determine the tablespaces and datafiles used by a user or schema.
   Note: Ensure that the *user* has EXP_FULL_DATABASE role.
   `select distinct df.file_name, t.owner, t.tablespace_name from dba_tables t, dba_data_files df where t.tablespace_name=df.tablespace_name and t.owner='schema';` This query returns the file name, owner, and tablespace name.
   Example:
   a. File_Name: `/u01/app/oracle/oradata/orcl/example.dbf`
   b. Owner: Test
   c. Tablespace_Name: Example
4. Make the tablespaces read-only. For multiple tablespaces in a schema, run the following command for each tablespace.
   `alter tablespace TABLESPACENAME read only;`

Export Schema Data

1. Open a command line prompt and copy all datafiles referenced by the schema to a vTDM share.
   `cp /u01/app/oracle/oradata/orcl/datafilename.dbf /vtdm/mnt/filesystemname`
2. Execute 'expdp' command to export schema metadata.
   To export multiple tablespaces, use a comma separated list of tablespace names for the transport_tablespaces parameter.
   a. For *Oracle 11g (Linux)*, run the following command:
      Replace *user*, *SCHEMANAME*, *SCHEMANAME_TS*, and *TABLESPACENAME* with your values.
      `expdp user schemas=SCHEMANAME dumpfile=SCHEMANAME_TS.dmp logfile=SCHEMANAME_TS.log transport_tablespaces=TABLESPACENAME`
   b. For *Oracle 12c (Linux)*, run the following command: Replace *user*, *SCHEMANAME_TS*, and *TABLESPACENAME* with your values. `expdp user dumpfile=SCHEMANAME_TS.dmp logfile=SCHEMANAME_TS.log transport_tablespaces=TABLESPACENAME`
3. Execute 'expdp' command to export all the schema objects in the SYSTEM tablespace.
   Replace *user*, *SCHEMANAME*, and *SCHEMANAME_SYS* with your values.

```
expdp user schemas=SCHEMANAME dumpfile=SCHEMANAME_SYS.dmp logfile=SCHEMANAME_SYS.log
include=FUNCTION,PACKAGE,PROCEDURE,SEQUENCE,SYNONYM,TYPE,VIEW,USER,ROLE_GRANT,SYSTEM_GRANT
```

Copy Database Dumps to vTDM

1. Determine the database dump directory. By default, the database dumps are saved in the DATA_PUMP_DIR directory.
   ```
   select directory_path from dba_directories where directory_name='DATA_PUMP_DIR';
   ```
2. Copy the database dumps and log files from the database dump directory identified above to the vTDM share. Log files are used to distinguish between system and tablespace dumps.
   ```
   cp /u01/app/oracle/oradata/admin/orcl/dpdump/SCHEMANAME*.* /vtdm/mnt/filesystemname
   ```
3. Make all the read-only tablespaces read-write. For multiple tablespaces in a schema, run the following command for each tablespace.
   ```
   alter tablespace TABLESPACENAME read write;
   ```

Now you are ready to checkpoint the Filesystem and start making the Filesystem available to Testers to clone.

Checkpoint the Gold-copy Data for Oracle

You as Test Data Engineer create Checkpoints of the data that testers will want to clone. A Checkpoint refers to a point in time for the data source or file system. vTDM Checkpoints are not associated with projects. After the tester creates a Clone, it is associated with the tester's current project.

You can choose to let Testers mount the Clone to their database server manually. Alternatively, you can specify that database files are attached automatically to an Oracle Server instance when testers create Clones of the checkpoint.

For automatic attachment, for Oracle Server, a non-system user with Import Full database role is required. For more information, see Create a Non-system User for Automatic Attachment.

Follow these steps:

1. Open the CA TDM Portal
2. Click vTDM.
3. Click **New Checkpoint**.
   a. Enter a name that allows your testers to identify the check point.
   b. (Optional) Enter a description.
   c. Choose whether to make this checkpoint visible to testers.
      Tip: You identify visible checkpoints in the Portal by an eye icon.
4. Click **Next**.
   a. Choose **Oracle 11g (LINUX)** or **Oracle 12c (LINUX)** as the DBMS Type option for how the Testers attach a Clone of this checkpoint to a database server.
      Note: The initial database selection in the Create Checkpoint dialog is based on the database file types in your filesystem.
   b. The database files are attached automatically to an Oracle instance.
      Tip: You identify auto-attaching checkpoints in the Portal by a database icon.
      Fill in the following fields:

- • DBMS Server — Oracle 11g (LINUX) or Oracle 12c (LINUX).
  - (Optional) Port
  - (Optional) Oracle Service Name
  - Username — Defines the username of an account with appropriate server roles.
  - Password — Defines the password for the same account.
  - Privileged User — Defines the username of an account that by default has access to all commands and files on a Linux or Unix operating system.
  - Privileged User Password — Defines the password for the same account on a Linux or Unix operating system.
    Note: The Privileged User credentials are used for mounting the virtual filesystem on a DBMS system.
  - Enter and confirm an Oracle user password for all clones created from this checkpoint.
- **None (Filesystem Clone)** — Allows the Testers to choose how the Clone is consumed. The user interface will provide Testers with information, username, and password, to mount the Clone to their Database Server manually.

c. (Optional) Click **Test** to verify the automatic database server connection.
d. Click **Save**.

You have copied the Gold-copy database to the Filesystem, and set up the appliance for the Tester to use. For more information about how Tester can make copies of the Gold-copy data, see Consume Gold-copy Clones with vTDM.

Maintenance and Recovery Operations for Oracle

This section describes the maintenance and recovery procedures to be used for vTDM with an Oracle database. We present several different types of maintenance scenarios along with the appropriate recovery procedures to use.

Upgrade or Patch a VMware ESXi Server

We recommend that you delete clones or shut down the Oracle instance before upgrading or patching the VMware ESXi Server.

When upgrading or patching the VMware ESXi server, you can suspend the vTDM appliance. If the Oracle instance is not shut down, the Oracle instance uses cached data to continue working with the vTDM appliance. However, after a period of time, the Oracle instance can become unresponsive.

When the vTDM Appliance resumes working, the Oracle database will automatically connect to the vTDM Appliance and continue working as normal. If Oracle does not resume, see Restarting, Patching, or Recovering a vTDM Appliance.

Restart, Patch, or Recover a vTDM Appliance

Delete clones and shut down the Oracle instance before performing a maintenance activity on the vTDM Appliance. In case of a power outage, when the vTDM Appliance is restarted, the Oracle database will be unable to write to any vTDM datafiles. Once the Appliance is restarted, restart the Oracle database as usual.

NOTE

The Oracle instance may terminate unexpectedly if the vTDM Appliance is shut down without deleting clones or shutting down Oracle.

Upgrade or Patch an Oracle Database

If an Oracle database upgrade or patch install requires a restart of Oracle Server Host, consider the following:

1. Remove the vTDM Clones from the CA TDM Portal. For more information on how to manually remove a clone, see Remove Clone from Oracle Database Manually.
2. Shut down the Oracle database instance.

3. Perform the upgrade or patch operation.
4. After the Oracle database is upgraded or patched, restart the Oracle database.
5. Recreate and reattach the vTDM Clones in CA TDM Portal. For more information on how to manually attach a clone, see Attach Clone to Database Manually.

Restart the Oracle Database

If you restart an Oracle database for maintenance purposes, the Oracle database automatically connects to the vTDM appliance. No action is required.

Reattach vTDM Clones After Oracle Server Host Restarts

If the Oracle Server host restarts, you must complete the following steps:

1. Mount the vTDM appliance shares.
2. Start the Oracle listener by running `lsnrctl start` command.
3. Start the Oracle host by running the `startup` command from sqlplus.

Example: Determine the NFS Mounts to be reattached

To determine the NFS mounts that need to be reattached, run the `ls /vtdm/mnt` command.

```
ls /vtdm/mnt
oracle_clone1     oracle_clone2
mount <appliance>:/database/oracle_clone1 /vtdm/mnt/oracle_clone1 -o
 rw,timeo=600,hard,wsize=32768,vers=3,rsize=32768
mount <appliance>:/database/oracle_clone2 /vtdm/mnt/oracle_clone2 -o
 rw,timeo=600,hard,wsize=32768,vers=3,rsize=32768
```

Clean up the Oracle Server Host

When the vTDM appliance is permanently unavailable or you are unable to access the vTDM appliance from CA TDM Portal, you can perform a manual clean up of the Oracle Host Sever. Remove vTDM related datafiles, tablespaces, and schemas.

Follow these steps:

1. Start the Oracle database by running the following command:
   `startup nomount;`
2. Determine names of the affected datafiles and tablespaces by running the following sqlplus query:
   `select df.name, ts.name from v$datafile df, v$tablespace ts where df.ts# = ts.ts#;`
3. Determine the schema name or user name.Open the CA TDM Portal and click **My Clones**. Select a Clone and click **DETAILS.** The username displayed on the Clone details dialog is the schema name to be dropped in the step 6 of this procedure.
4. Ensure that all datafiles obtained as a result in step 2 are offline.
   a. Run the following command only once: `alter database mount;`
   b. Run the following command for each datafile:
      `alter database datafile 'datafile_name' offline drop;` **Note:**Ensure that the datafile name is entered in single quotation marks.
5. Restart the Oracle database by running the following command.

```
alter database open;
```

6. Remove the tablespaces and schemas by running the following commands for each tablespace and each schema.

```
drop tablespace tablespace_name including contents;
drop user user_name cascade;
```

The Oracle Server host is now cleaned up.

To continue using vTDM, un-register the existing vTDM Appliance and re-register a new vTDM Appliance from CA TDM Portal. For more information, see Install and Register the vTDM Appliance.

Example: Remove Datafiles and Tablespaces from Oracle Server Host

This example shows how to remove three datafiles and three tablespaces from Oracle Server Host.

```
startup nomount;

select df.name, ts.name from v$datafile df, v$tablespace ts where df.ts# = ts.ts#;

alter database mount;
alter database datafile 'first_datafile_name' offline drop;

alter database datafile 'second_datafile_name' offline drop;

alter database datafile 'third_datafile_name' offline drop;

alter database open;

drop tablespace first_tablespace_name including contents;

drop tablespace second_tablespace_name including contents;

drop tablespace third_tablespace_name including contents;
drop user user_name cascade;
```

How to Copy Data from Flat Files

The filesystem clones option allows testers to create filesystem clones. It behaves in the same way as a database connected clone except that the database is not mounted onto the clone filesystem. The filesystem clone is available to be consumed or modified by the tester as per their requirement. Filesystem clones are useful when a customer application requires file data instead of a connected database.

As a Test Data Engineer, you copy all files that represent your gold copy source onto your vTDM Appliance and make it available for cloning to the Tester. You can manage checkpoints through either the CA TDM Portal or the vTDM API. We recommend using the CA TDM Portal, and you have the option to use the REST API from external automation scripts. You can deposit files of any type on the vTDM filesystem for your checkpoint, and there are no restriction on the directory hierarchy which you create on the file system.

Prerequisites:

- Install and Register the vTDM Appliance

Follow these steps:

Create a Filesystem

Each appliance supports several filesystems.

1. Open the CA TDM Portal
2. Click vTDM.
3. Click **Add Filesystem**.
   a. Enter a name that allows your Testers to identify the Filesystem.
   b. (Optional) Enter a description.
   c. Click **Save**.

Copy Gold-copy Data to Filesystem

As Test Data Engineer, you have taken production data and masked it to remove any sensitive and other personal identifiable information. In this step, you connect the Filesystem to any compatible machine that hosts this Gold-copy data. Finally, you copy the Gold-copy data to the Filesystem.

Follow these steps:

1. Open the CA TDM Portal.
2. Click vTDM.
3. Click **Add Data** and select **Filesystem Clone**.
4. Follow the instructions in the dialog.
   a. Copy your files on to the vTDM share.

Now you are ready to checkpoint the Filesystem and start making checkpoints available to Testers to clone.

Checkpoint the Gold-copy Data

You as Test Data Engineer create Checkpoints of the data that testers will want to clone. A Checkpoint refers to a point in time for the data source or file system. vTDM Checkpoints are not associated with projects. After the tester creates a Clone, it is associated with the tester's current project.

Follow these steps:

1. Open the CA TDM Portal
2. Click vTDM.
3. Click **New Checkpoint**.
   a. Enter a name that allows your testers to identify the check point.
   b. (Optional) Enter a description.
   c. Choose whether to make this checkpoint visible to testers.
   Tip: You identify visible checkpoints in the Portal by an eye icon.
4. Click **Next**.
   a. Choose **None (Filesystem Clone)** as the DBMS Type option for how the Testers attach a Clone of this checkpoint to a database server.
5. Click **Save**.

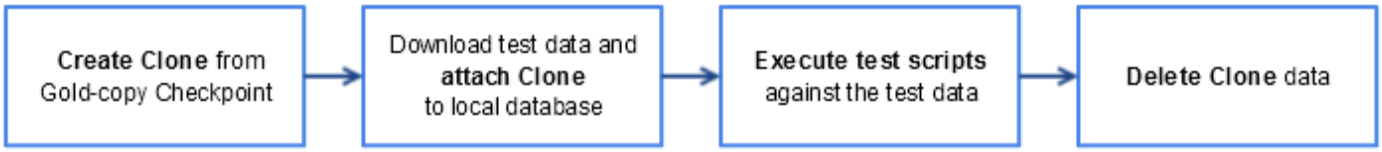
You have copied the Gold-copy files to the Filesystem, and created a checkpoint for the Tester to use. For more information about how Tester can make copies of the Gold-copy data, see Consume Gold-copy Clones with vTDM.

Consume Gold-copy Clones with vTDM

You as a Tester want to make copies of the Gold-copy data that the Test Data Engineer has prepared. Contact the Test Data Engineer for the names of the filesystem and Checkpoints that you want to clone.

Prerequisite: Copy Data from vTDM Supported Data Sources

Figure 48: vtdm tester workflow



Create Clones

You recognize a Gold-Copy in the Self Service Catalog by the vTDM symbol. If you don't see the Gold-copy you need, ask your Test Data Engineer to create it or make it available to testers.

1. Open the CA TDM Portal and click **Self Service Catalog**.
2. Choose a project.
3. Identify the Gold-Copy that you need and click **New Clone**.
   a. Define a unique Clone Name.
   b. (Optional) Define a Clone description.
   c. Click **Create**.
4. Review the details how you connect to this Clone.
   – If the clone is configured for auto-attachment, the connection details contain a JDBC connection string. The status of the clone attachment is visible on the clone tile under **My Clones**.
   – If the clone is not configured for auto-attachment, the network share connection details are populated to consume the clone as a Filesytem or flat files clone.
5. (Optional) Click **Email Me** to send a message with the connection details of a Clone to yourself, including a link to the instructions on how to manually attach a Clone to a database.
   If you receive a message that emailing is not enabled in the portal, or if the Email button is grayed out, contact your TDM administrator.

The Clones appear in the CA TDM Portal under **My Clones**.

When you create or delete a Clone, the status is displayed in the CA TDM Portal under **My Clones**. The new Clone is associated with your current project. If the status displays an error message, click **DETAILS** or the error message link to troubleshoot the issue.

If a Clone is automatically attached to a database, use the JDBC connection details to configure the application under test. To use a Filesystem Clone, use the credentials provided to connect to the vTDM share, and configure your application to connect to this share.

View Clones

Each Clone is associated with a project. Open the CA TDM Portal and click **My Clones**. Here you can view details or delete Clones.

- Testers only see their own Clones.
- TDEs can view Clones from all users.

View Return on Investment for vTDM

The vTDM section in the Test Data Manager Portal displays estimates how much time and disk space you saved by using vTDM instead of copying and attaching databases manually.

The graphs visualizes the following estimates of your returns on investment over time:

Time Saved

Displays how much time you saved by using vTDM. The estimate is based on typical constant speed and time values for database copy and attachment operations. The value sums up the time saved for all Clones. If you have not created any Clones yet, this metric is zero.

The value is derived from the following measurements and estimates:

- The number of Clones created
- The total size of the checkpoint from which the Clone was copied, in Bytes
- The actual disk usage of each Clone, include modifications after the copy, in Bytes
- Estimated disk write speed: 2.8 GB/min
- Estimated time for each manual database attachment: 6 min
- Estimated time for each manual database copy: 1 min

Storage Saved

Displays how much storage space you saved by using vTDM. This estimate is based on the space taken by each checkpoint, times the number of its Clones, for all Filesystems. The appliance increments the running total of the delta between the size of the checkpoint and the clone size. If you have not created any Clones yet, this metric is zero.

Disk Usage

Displays the current actual disk usage.

vTDM Troubleshooting

After you have installed the Appliance, make sure to register it with vTDM via the CA TDM Portal. If you encounter registration issues, use the following procedures to validate and troubleshoot your installation.

First, open the **VMWare vSphere Client** and double-check the settings under **Edit Settings, Options, vApp Options, Properties**.

Validate whether the vTDM Appliance is Working

The Portal accepts and registers the Appliance only if the Appliance is running and has a valid hostname.

Follow these steps:

1. Open a command prompt and execute the following command. Replace *hostname* by the hostname of your Appliance:
   `ping hostname`
2. Check which type of response you get:

- – Does the ping command return data?

  ```
  Pinging hostname [IP_address] with 32 bytes of data:
  Reply from IP_address: bytes=32 time=2ms TTL=126
  Reply from IP_address: bytes=32 time=1ms TTL=126
  ```

  This means the hostname is valid and in the DNS, but the vTDM Appliance service cannot be reached. Continue with the troubleshooting tip "I cannot register the Appliance although the hostname is valid".

- – Or does the ping command time out?

  ```
  Pinging hostname [IP_address] with 32 bytes of data:
  Request timed out.
  ```

  This means the hostname is valid, but the vTDM Appliance cannot be reached.

Open the VMWare vSphere Client and ensure that the vTDM Appliance is started .

– Or does the ping command return an error?

```
Ping request could not find host hostname. Please check the name and try again.
```

This error means that the hostname is not valid.
Continue with the troubleshooting tip "The vTDM Hostname is not Valid".

vTDM Troubleshooting

If you cannot register the Appliance, use the following advice for troubleshooting.

I Cannot Register the Appliance Although the Hostname is Valid

1. Verify that you are using the correct API Access password for the `vtdmadmin` account.
2. Verify that the hostname is actually that of the Appliance.
   a. Open the **VMWare vSphere Client** and locate the Appliance host in the left hand tree view.
   b. Verify that this host contains the Appliance.
      For example, confirm that Guest OS in the General tab says "Oracle Solaris 11 (64-bit)".
3. Restart the Appliance.
   a. Open the **VMWare vSphere Client** and locate the Appliance host in the left hand tree view.
   b. Right-click and choose **Power, Restart Guest**.
4. If the Appliance still does not register, please contact CA Support.

The vTDM Hostname is Not Valid

1. Look up the IP address for the Appliance.
   a. Open the **VMWare vSphere Client** and locate the Appliance host in the left hand tree view.
   b. Go to the **Summary** tab in the right hand panel.
   c. Go to the General sub panel and verify the **IP Address** and **DNS Name**.
2. Use this IP address in the vTDM UI to register the Appliance.
3. Contact your local system administrator to add this IP address and hostname to your site's DNS servers. Wait for the DNS change to propagate.

I Cannot see any Clones After Moving the vTDM Appliance to Another Instance of CA TDM Portal

The appliance is not designed to be moved from one instance of CA TDM Portal to another instance of CA TDM Portal.

I Get an Error Message that ORACLE_HOME is Not Set

Modify your non-interactive login script (for example, .bashrc) on the Oracle database machine to define ORACLE_HOME.

Advanced Troubleshooting

Only under direction of CA support should you log into the appliance via vSphere remote console as root.

- Check contents of /opt/vtdm/logs/network.log and /opt/vtdm/logs/vtdm.log
- Check output of ipadm show-if: IFNAME e1000g0 should be in state ok
- Check output of ipadm show-addr, ADDROBH e1000g0/v4 should be in state ok

Javelin

Javelin is a general-purpose automation tool that significantly reduces the time to perform key testing and Test Data Manager tasks. Use Javelin as your engine to model workflows and run sequences of commands that automate complex data migrations and application activities. For example, you can automate CA TDM activities, web testing, database scripts, web services, SSH, and more.

- An intuitive workflow modeling interface that makes it easy to parameterize key values
- An expandable framework for which you can create extensions for custom activities
- Accelerators so you can quickly create workflows for subsetting and cloning data
- Connectors that provide integration points with Datamaker and other Test Data Manager components.
- An API to call Javelin commands from other CA tools, such as CA Release Automation

For example, you can use the Bulk Copy functionality in Javelin to automate the transfer of CA TDM generated data subset extracts into their destination database. This workflow is an automated alternative to the otherwise manual process that is described in the Data Subset documentation.

Datamaker contains functionality to register and use Javelin programs. Datamaker can execute Javelin programs as an "Ad-hoc" action, or during publish as a "pre-publish" or "post-publish" action.

Install Javelin

Javelin is included in the CA TDM installer. When you install Javelin, verify the following prerequisites:

- **Operating System**
  Javelin is supported on:
  - Microsoft Windows 8
  - Microsoft Windows 8.1 enterprise editions
  - Microsoft Windows Server 2008 R2
  - Microsoft Windows Server 2012 R2 (64-bit)
  - Microsoft Windows 10 (64-bit)
  - Microsoft Windows Server 2016
- **.NET 4.5**
  Download .NET 4.5 from the following location: http://www.microsoft.com/en-gb/download/details.aspx?id=30653
- **VisualUIAVerifyNative**
  VisualUIAVerofyNative permits the extraction of IDs for automated Windows testing. VisualUIAVerofyNative is part of the Microsoft Windows SDK.
  Note: Access to the Microsoft Windows SDK is required to perform Windows Application Testing.

  NOTE

  CA TDM 4.3 supports Javelin installed in the default folder.

Upgrade Javelin

When you upgrade to CA Test Data Manager 4.3, the installer upgrades Javelin 1.5.x to Javelin 2.0. The Javelin 2.0 extensions directory is moved to `%ALLUSERSPROFILE%\CA\JavelinConfig\Extensions`. Typically, `%ALLUSERSPROFILE%` is mapped to the `C:\ProgramData` directory.

To prepare for the upgrade, back up the Javelin 1.5.x installation directory. By default, Javelin is installed in `C:\Program Files (x86)\Grid-Tools\Javelin\`. Javelin 1.5.x or earlier extension files are supported in Javelin 2.0.

When you upgrade to Javelin 2.0, ensure that:

- As part of upgrading to Javelin 2.0, you open the Javelin UI to initiate the transfer of all extensions to the new directory.
- Select the correct option in the extensions directory dialog based on if you are upgrading from Javelin 1.5.x or from Agile Designer Automator.
  Select *one* of the following appropriate options and continue to open Javelin.
  – I am upgrading from older versions of Javelin
  – I am upgrading from Agile Designer Automator to Javelin
  – I am a new user who has had neither Javelin or Agile Designer Automator installed until now
  Tip: Only new users select the third option.

  WARNING

  Downgrading from Javelin 2.0 to earlier versions is not supported.

Javelin Keyboard Shortcuts

Use the following keyboard shortcuts when modeling flows:

- **Ctrl+ E, V**
  Shows / hides the variable window.
- **Ctrl + E, F**
  Auto connects selected activities in the flowchart.
- **Crtl + E, I**
  Shows/hides the imports window.
- **Ctrl + E, N**
  Adds a variable and opens the Variables window.
- **Ctrl + E, O**
  Shows/hides the overview window.
- **Ctrl + Alt + F6**
  Moves the keyboard focus from the current UI area to the next area in a circular motion. The order is:
  - Breadcrumb navigation bar. Use left and right arrow buttons to move in breadcrumbs.
  - Designer surface
  - Arguments/Variables/Import designer

Create and Execute Visual Workflows

Javelin provides access to various automation activities to build an automation workflow. You can see the logical groups that include different activities to add to a workflow and execute those actions as per the flow design. Every action contains property values that you must provide. You can do so directly within the workflow when you drag an action into the flow using the Properties pane.

When you launch Javelin application, a flowchart is created by default and is available for your use. You can add various elements to the flowchart and design the sequence of actions to execute.

Follow these steps:

1. Verify that the new flowchart includes a start element.
2. Drag the following general workflow elements from the Toolbox as required into the flowchart:
   – Flow Chart—are the building blocks of a flow chart.
   – Control Flow—control the sequence of a flow chart with automated actions.
   – Primitives—perform operations in your flow, such as delays and method invocations.
   For more information, see Visual Work Flow Actions.
3. Drag the supported automation activities into the flowchart as needed to build your visual workflow:

- Automating Web Testing Activities
- Automating Database Activities
- Automating File System Activities
- Automating TDoD Activities
- Automating Communication Activities
- Automating Secure Shell Activities

For more information see Visual Work Flow Actions.

4. Double click on each of the elements that you added, and specify their properties.
   Properties can be one of the following types. To view the property type, mouse-over the name of the property.
   - **InArgument<Type>**
     These properties can only be passed to the action, and the value is neither changed nor assigned. It can either be a literal value or a variable.
   - **OutArgument<Type>**
     These properties are assigned by the activity.

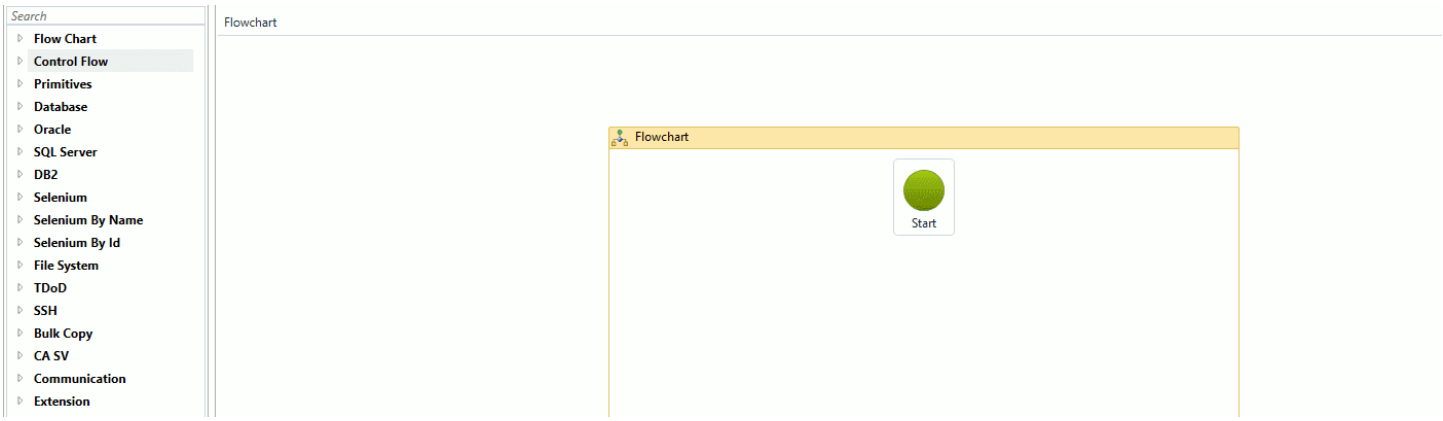
   Not all property fields are required, some are optional. If you see a red exclamation mark on an action, fill in the required property field.
5. (Optional) Create variables and assign default values.
   You can pass variables to properties of the same type, regardless of the property being InArgument or OutArgument. After completion of the action, the variable holds the data assigned by that action. Variables can be passed to an action by typing their names in the relevant field.
6. Click **File, Save**. Specify the file name and select a destination folder to save the flowchart.
7. Click **Execute, Start**.
   The flow is executed and a log is generated, showing the progress for each action added to the flow.

Visual Work Flow Actions

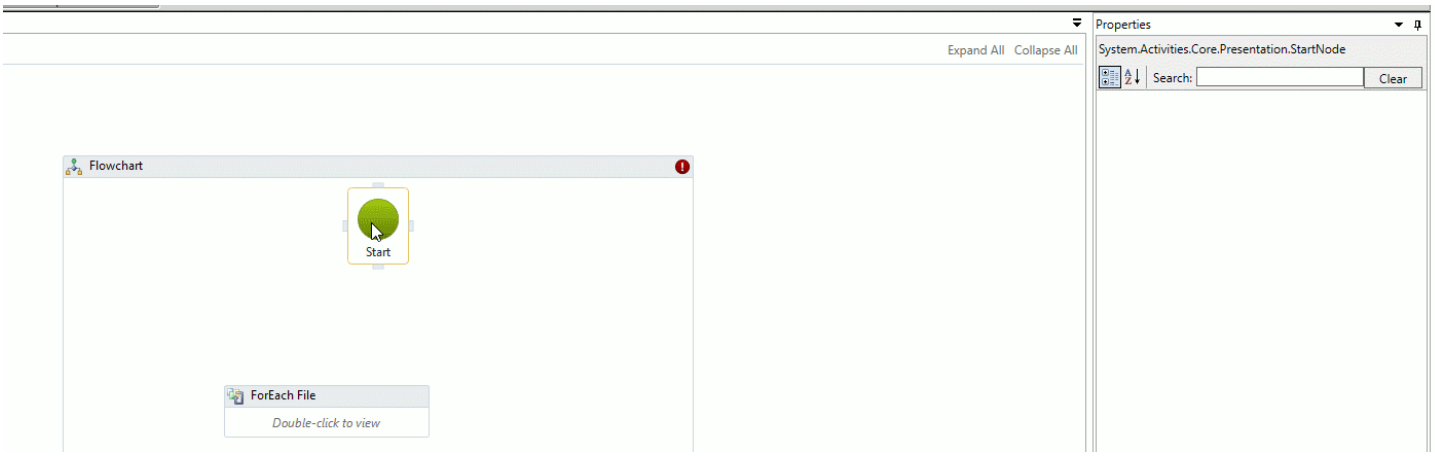
Javelin comes with a graphical user interface that lets you model workflows intuitively. The interface makes it easy to parameterize key values, and it has integration points with Datamaker and other CA Test Data Manager components.

Build workflows using the following general actions for the supported automation activities available in the Javelin toolbox.

Drag the required actions into a flowchart to lay out the visual workflow. You can change the name of an action on the canvas:



If you have changed the name of an action, you can find the original name by clicking the action and inspecting the top of the properties pane:



Flowchart

Flowchart functions ease the creation of a flow chart.

The following flow chart functions are available:

Flowchart

Inserts a sub-flow into the flow chart.

Sequence

Inserts a process block in the flow chart.

Example: #unique_531

Wait

Waits for the specified amount of time in seconds.

Pretty Print XML

Optimizes the appearance of an XML document.

- **String** — Defines an XML String
- **Output** — Stores the beautified output string
- **Include XML Declaration?** — Specifies whether to include XML Declaration.

XPath Search

Searches an XmlDocument with XPath and gets node values which qualify the input XPath.

Example: #unique_532

FlowChart API Reference

Actions related to creating flow charts.

Process

Inserts a process block into the flow chart.

Property Name	Mandatory	DataType	Description
Display Name	N	string	Name or brief description of the activity that you perform.

Pretty Print XML

Optimizes the appearance of an XML document.

Property Name	Mandatory	DataType	Description
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
Display Name	N	string	Name or brief description of the activity that you perform.
IncludeDecBool(IN)	N	Boolean	Whether to include Declaration in XML Document.
OutXMLString(OUT)	Y	string	Contains the beautified XML string.
Timeout(IN)	N	int	Duration of the timeout in seconds
XMLString(IN)	Y	string	A valid XML string to be beautified (Input)

XPATH Search

Searches an XmlDocument with XPath and gets node values which qualify the input XPath.

Property Name	Mandatory	DataType	Description
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
Display Name	N	string	Name or brief description of the activity that you perform.
FirstMatch(OUT)	N	string	Contains the first found occurrence
Timeout(IN)	N	int	Duration of the timeout in seconds
Values(OUT)	N	string[]	Contains the resultant XPath value
XMLDoc(IN)	Y	xml document	Input xmldoc on which the XPath needs to be evaluated
XPATH(IN)	Y	string	Defines the XPath

ArdDataBuilder

This action can be used in conjunction with CA TDM DataBuilder. All the properties passed to this action are with respect to the given DataBuilder flow.

Property Name	Mandatory	DataType	Description
ConfigBuilder(OUT)	Y	Ard.JavelinDataBuilder	Output DataBuilder returned from the flow
ConfigFilePath(IN)	Y	string	Path to the configuration file for DataBuilder
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
Display Name	N	string	Name or brief description of the activity that you perform
GroupId(IN)	Y	string	Group_id for the DataBuilder flow
Timeout(IN)	N	int	Duration of the timeout in seconds

Control Flow

Control flow functions let you control the sequence of a flow chart with automated actions.

The following control flow functions are available:

Decision

Inserts a decision block into the flow chart. The decision block is based on a condition and acts based on the result, which can be True or False.

Example: #unique_533

Exit

Terminates the flow chart.

For Each Object

Specifies objects and specific functions to perform on those objects. This action loops across all of the objects in an array or list of objects. Configure the type of list from the TypeArgument property.

Example: #unique_531

If

Performs an action based on a defined If condition.

Example: #unique_531

Log

Logs the activity of a process in a log file or window.

Example: #unique_534

Set Boolean

Sets a Boolean value during run time. If the value is not set in a given duration, the default value is used. Requires user input during execution.

Example: #unique_533

Set Double

Sets a double value during run time. If the value is not set in a given duration, the default value is used. Requires user input during execution.

Example: #unique_533

Set Integer

Sets an integer value during run time. If the value is not set in a given duration, the default value is used. Requires user input during execution.

Example: #unique_533

Set String

Sets a string value during run time. If the value is not set in a given duration, the default value is used. Requires user input during execution.

Example*:* #unique_533

Switch

Inserts a conditional clause (if/else) to determine alternative routes based on the input being evaluated.

Example: #unique_535

Try Catch

Surrounds an activity with a Try Catch block to handle errors and exceptions while still enabling program execution.

While

Defines a While condition that you can use to evaluate an input.

Example: #unique_536

Assert

Checks if two values are the same.

Control Flow API Reference

Control flow functions let you control the sequence of a flow chart with automated actions.

Assert

Checks if two values are the same.

Property Name	Mandatory	DataType	Description
CompareTo(IN)	Y	object	Value with which you want to compare the InValue.
ComparisonResult(OUT)	Y	comparisonresult	Contains the result of the comparison as two properties: • AreEqual – Indicates if two objects are equal or not • DifferencesString - Indicates what are the differences
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
Display Name	N	string	Name or brief description of the activity that you perform
InValue(IN)	Y	string	Value to be compared with the compareTo field.
Timeout(IN)	N	int	Duration of the timeout in seconds

Primitives

Primitive functions let you perform various basic operations in your flow, such as delays and method invocations.

The following primitive functions are available:

Assign

Assigns a value to a variable. Enter a VB expression as value.

Example: #unique_532

Delay

Waits for a specific duration of time.

Invoke Method

Invokes a method from a specific application. You can invoke a method of a .NET static class or object with or without parameters and get the result of the invocation.

- **TargetType**
- **Target Object**
- **Method Name**

Example: #unique_532

Invoke Process

Executes any external process, like launching an application.

Example: #unique_537

RegEx Matches

Performs queries using regular expressions which return matches of the input text as a string.

Get Variable Value

Gets the value of a variable.

- **VariableName** — Defines the name of the variable whose value you want to get.
- **VariableValue** — Outputs the value after execution.

Set Variable Value

Sets the value of a variable.

- **VariableName** — Defines the name of the variable whose value you want to change.
- **VariableValue** — Defines the string value or string variable.

CallJavelinFlow

Calls a Javelin flow and executes it as batch process. You can override variables in the called flow by preceding this action with a SetVariable. After execution, the log file outputs lines in the following order:

1. the flowpath
2. the temporary variables filepath used to override variables in the called flow. The file is empty if no variables are overriden.
3. the log filepath.

You can enter the following property values for this action:

- **ExitCode** — Specifies an integer variable. The exit code is output to this variable. 0 means successful execution of the flow.
- **JavelinExecutorPath** — Defines the directory where the JavelinExecutor.exe is located.
  Default: `C:\Program Files (x86)\Grid-Tools\Javelin`
- **FlowPath** — Defines the full path of the flow to be called.
- **LogToCurrentFlowLogs** — Specifies whether to log the called flow in the master flow log file.
- **WaitForExit** — Specifies whether to force the master flow to wait for the called flow to exit.

Example: #unique_538

SetVariableinCalledFlow

Overrides variables in a called flow. This action must precede CallJavelinFlow.

- **Variable Name** — Defines the name of variable in the called flow.
- **Variable Value** — Defines the new value of the variable in called flow.

Primitives API Reference

Primitive functions let you perform various basic operations in your flow, such as delays and method invocations.

InvokeMethod

Invokes a method from a specific application. You can invoke a method of a .NET static class or object with or without parameters and get the result of the invocation.This action can be used to invoke a dotnet method. E.g., load a document.

Property Name	Mandatory	DataType	Description
Display Name	N	string	Name or brief description of the activity that you perform
GenericTypeArguments	N	collection <type>	Type Arguments
MethodName	Y	string	The name of a VB.Net method that you want to call on either a static class (for example, the Create method of the Directory class) or an Object variable (for example, a String variable). Declare the variable in the Variables pane.
Parameters	N	arguments	File path
Result (OUT)	N	argument	Contains the output of the method that is being called
RunAsynchronously	N	Boolean	Go on to next block if the result is not received.
TargetObject(IN)	One of TargetObject or TargetType is mandatory.	argument	Any variable that you have created in Javelin.
TargetType(IN)	One of TargetObject or TargetType is mandatory.	type	Any .NET static class

InvokeProcess

This action can be used to invoke executable files like .bat, .exe., etc

Property Name	Mandatory	DataType	Description
Arguments(IN)	N	string	Arguments required by the process
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
Display Name	N	string	Name or brief description of the activity that you perform
ExecutableName(IN)	Y	string	Executable name
ExitCode(OUT)	N	int	Return exit code
TerminateProcessAfterTimeInSecs(IN)	N	int	Attempt to terminate the process after a given number of seconds
Timeout(IN)	N	int	Duration of the timeout in seconds
WaitForProcessToExit(IN)	N	Boolean	Synchronous by default. True means synchronous.
WorkingDirectory(IN)	Y	string	Directory path of the executable

RegexMatches

Performs queries using regular expressions which return matches of the input text as a string.

Property Name	Mandatory	DataType	Description
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
Display Name	N	string	Name or brief description of the activity that you perform
InputText(IN)	Y	string	Input to run regex match on
MatchedGroups(OUT)	Y	matched collection	Contains the matched result
Pattern(IN)	Y	string	Regex pattern
Timeout(IN)	N	int	Duration of the timeout in seconds

See also: Supported Automation Activities

Automating Database Activities

You can create Javelin workflows to automate various database activities. Javelin is compatible with both 32-bit and 64-bit databases, and supports remote database connections. You need database access to work with databases and database activities.

Javelin supports the following database activities:

Note: Javelin supports Oracle, DB2, and Microsoft SQL Server. Subset also supports these data sources. Thus, Subset can generate Javelin workflows for Oracle, Microsoft SQL Server, and DB2. Javelin has been tested on Windows and Linux.

Database

Database functions let you automate various database operations in Javelin flows. The following functions are available:

Database Query to Load Table

Issues a query to return a database table. DataTable is a .NET object that is returned based on query results. It represents in-memory data.

- **Query**

For Each Row

Performs an instruction for the specified rows. Use variables to specify an expression that operates on a database row.

Example: #unique_539

Cassandra - Execute Query

Runs a query on a Cassandra database.

- **Contact Points** — Defines a comma separated lists of host addresses of Cassandra nodes.
- **Keyspace**
- **Username**
- **Password**
- **Query**
- **Output**

Netezza - Execute Query

Runs a query on a Netezza database.

- **OleDB Provider** — Defines the server name.
- **Data Source**
- **PersistSecurityInfo** — Specifies whether to persist (true) or not (false).
- **Port**
- **Username**
- **Password**
- **Query**
- **Output**

Database Action API Reference

Database related actions. Note that in connection details related properties, you can either pass a full connection string or pass the connection field values separately (like server, username, password, etc). One of these two ways is mandatory and the other is optional. In the mandatory fields, this is listed as "Y -- or use connection string" or "Y -- or configure individual values".

Database Query to load Data Table

Javelin contains dedicated actions for commonly used databases (Oracle, MS SQL Server, DB2, and so on). We recommend to use those dedicated actions for such databases. For other generic database connections, use this action.

Property Name	Mandatory	DataType	Description
ConnectionString(IN)	Y	string	Connection string to connect with database
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
Display Name	N	string	Name or brief description of the activity that you perform
IsStoredProcedure(IN)	N	Boolean	Is it a stored proc. Values: yes/ no
OutElement(OUT)	Y	datatable	Contains the query result data table
ProviderInvariantName(IN)	Y	string	Oracle/SQLServer who ever is provider
Query(IN)	Y	string	Query to execute
Timeout(IN)	N	int	Duration of the timeout in seconds

Oracle

Oracle functions let you automate various Oracle database operations in Javelin flows.

Execute Query

OracleActivity executes a query against an Oracle database.OraclePLSQLActivity also supports Oracle PL/SQL queries.

Note: Install ODP.NET 4.0 (v 4.112.3.0) for this activity to work.

- **Server**
- **Service Name**
- **Port**
- **Username**
- **Query** — The query can be either Select, Insert, Update or Delete.
- **Output** — For Select queries, output rows are assigned to an output variable which is set in the OutDataTable property.

You can also execute Stored Procedures that exist on the server using OracleActivity.

Example: Use the following VB.net syntax in the query field to invoke a procedure with 4 parameters:

```
string.Format("begin schema.package.procedure_name (‘{0}‘, ‘{1}‘, ‘{2}‘,’{3}‘); end;", value1, value2, "test",
 "test2")
```

The first two parameters (value1 and value2) are Javelin variables. The second two parameters (test and test2) are hard-coded.

SQL Server

SQL Server functions let you automate various Microsoft SQL Server database operations in Javelin flows. The following functions are available:

Execute Query

SQLActivity executes a query against SQL Server database. You have the option to add parameters.

- **Server**
- **Database Name**
- **Username**
- **Query** — The query can be either Select, Insert, Update or Delete.
- **Output** — For Select queries, output rows are assigned to an output variable which is set in the OutDataTable property.

Examples:

- #unique_540
- #unique_541

SQL Server Action API Reference

Execute Parameterized Query

Execute SQLServer query.

Property Name	Mandatory	DataType	Description
CommandTimeOut(IN)	N	int	Time for which connection should wait for command to start returning results.
ConnectionString(IN)	Y -- or configure individual values	string	SQLServer connection string.
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
DatabaseName(IN)	Y -- or use connection string	string	Database name
DisplayName	N	string	Name or brief description of the activity that you perform
OutDataTable(OUT)	N	datatable	Contains resultant query data
Password(IN)	Y -- or use connection string	string	Password for database connection
Query(IN)	Y	string	Query to execute
Server(IN)	Y -- or use connection string	string	Database Server info
Timeout(IN)	N	int	Duration of the timeout in seconds
Username(IN)	Y -- or use connection string	string	Username for database connection

DB2

DB2 functions let you automate various database operations in Javelin flows.

Execute Query

Db2Activity executes a query against an IBM DB2 database.

Note: Install the IBM DB2.NET Data Provider version 9.7.4.4 for this activity to work.

- **Server**
- **Database Name**
- **Username**
- **Query** — The query can be either Select, Insert, Update or Delete.
- **Output** — For Select queries, output rows are assigned to an output variable which is set in the OutDataTable property.

Bulk Copy

The Bulk Copy functions automate the bulk copy of data into a database. You can use these functions to automate the copy of a Data Subset extract into its target database.

The following functions are available:

Data Reader/DB2 Data Reader

Connects to a database table to read its data. Using this we query a database and return a datatable that is held as an IDataReader object (This object must be created in the variables pane by the user). The IDataReader object can then be passed to 'DB2/Oracle/SQL Bulk Copy - Data Reader' actions to pull data from the source system, and push to the target system without loading in memory.

Specify the following required properties:

- **Server**Server name where the database is installed. Specify a server port if applicable.
- **Database**Specifies the database to connect to.
- **Schema**Specifies the schema if needed. Otherwise, leave this field empty.
- **Username/Password**Enter valid database credentials.
- **Query**Specifies the query to use to retrieve records from the database.
- **OutDataReader**Creates a variable of type IDataReader, with no default value specified, into which you can enter variable data for the bulk copy.

DB2BulkCopy, OracleBulkCopy, SQLBulkCopy - Data Reader

The IDataReader object holding the data table is passed to the relevant (DB2 or Oracle or SQL) action to pull data from the source system, and push to the target system. Drop the DB2/Oracle/SQL action inside the Data Reader Action. The Source and Target database can be different types and have different names, but the column count must be the same.

If the source and the target column names differ, specify column mappings in the column mapping field in the properties pane. Separate the column mapping with a colon, and separate multiple mappings with a comma, for example "`SourceColumn1:TargetColumn1,SourceColumn2:TargetColumn2`" . Alternatively, click the Map Columns button in the Bulk Copy action to provide column mappings.

Specify the following required properties:

- **Batch Size**Specifies how many rows are sent to the target database at once.
- **Destination Table Name**Specifies the table name to which the data needs to be transferred.
- **InDataReader**Specifies the IDataReader variable used in the OutDataReader property of Data Reader Activity.
- **Service Name (Oracle Only)**Specifies the service name of target database.
- **Username/Password**Enter valid credentials to connect to the target database.

TDM Data Subset provides an accelerator for large databases composed of many tables. For more information, see Javelin Example: Subset Bulk Copy.

InsertDataTableActivity

Reads a data table from a Data-source and inserts it into a Target table in a Database. Doesn't use the bulk copy protocol. Specify the following required properties:

- **Datasource**Specify the DataSource segment of the DB connection string, for Oracle this is just the ServerName see: https://docs.microsoft.com/en-us/sql/reporting-services/report-data/data-connections-data-sources-and-connection-strings-report-builder-and-ssrs for details about other DB types.
- **InDataReader**Specifies the IDataReader variable that holds the rows to be copied to the target.
- **UserId/Password**Enter valid credentials to connect to the target database.
- **ProviderName**Specifies the DataSourceAttribute.ProviderInvariantName property. For Sql Server this is system.data.sqlclient, for Oracle Oracle.DataAccess.Client or system.data.oracleclient
- **TableName**Name of the destination table in the target database.

Examples:

- #unique_543
- #unique_544
- #unique_545
- #unique_546
- #unique_547
- #unique_548

Bulk Copy Action API Reference

The bulk copy utility in Javelin is a fast method of moving data from one database to another, or between database types.

Note that in connection details related properties, you can either pass a full connection string or pass the connection field values separately (like server, username, password, etc). One of these 2 ways is mandatory and the other would be optional. In the mandatory fields, this is listed as "Y -- or use connection string" or "Y -- or configure individual values".

DataReader

Reads data from ODBC or OLEDB connection.

Property Name	Mandatory	DataType	Description
AdditionalConnectionParameters(IN)	N	string	You can connect to other ODBC data sources through additional connection parameters
Children	N	collectionactivity	Ignore this property. It's auto managed and adds children into collection which are dropped inside DataReader activity
CommandTimeout(IN)	N	int	timeout for executing the command
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
Database(IN)	Y	string	Database name
DisplayName	N	string	Name or brief description of the activity that you perform
IntegratedSecurity	N	Boolean	Dependent on database configuration
OdbcConnectionString(IN)	Y -- or configure individual values	string	ODBC full connection string
OleDbConnectionString(IN)	Y -- or configure individual values	string	OLE DB full connection string
OutDataReader(OUT)	Y	DataReader	Contains the resultant DataReader
Password(IN)	Y -- or use connection string	string	Password
Provider(IN)	Y	string	Provider type – sqlserver, oracle,etc
Query(IN)	Y -- or use connection string	string	Query to execute on the database
Schema(IN)	Y -- or use connection string	string	Database schema

Server(IN)	Y -- or use connection string	string	Database server name
ServiceName(IN)	N	string	Dependent on database type
Timeout(IN)	N	int	Duration of the timeout in seconds
Username(IN)	Y	string	Username

DB2 DataReader - READ

Read data from DB2 connection

Property Name	Mandatory	DataType	Description
AdditionalConnectionParameters(IN)	N	string	You can connect to other ODBC data sources through additional connection parameters. (not recommended)
Children	N	collectionactivity	Ignore this property. It's auto managed and adds children into collection which are dropped inside DataReader activity
CommandTimeout(IN)	N	int	Not needed. Can be left empty
ConnectionString(IN)	Y -- or configure individual values	string	DB2 connection string
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
DatabaseName(IN)	Y	string	Database name
DisplayName	N	string	Name or brief description of the activity that you perform
OutDataReader(OUT)	Y	DataReader	Contains the resultant DataReader
Password(IN)	Y -- or use connection string	string	Password
Provider(IN)	Y -- or use connection string	string	Database type
Query(IN)	Y	string	Query to execute on database
Server(IN)	Y -- or use connection string	string	Database server
Timeout(IN)	N	int	Duration of the timeout in seconds
Username(IN)	Y -- or use connection string	string	Username

DB2 Bulk Copy DataReader – WRITE

Writes data into DB2

Property Name	Mandatory	DataType	Description
AdditionalConnectionParameters(IN)	N	string	You can connect to other ODBC data sources through additional connection parameters. (not recommended)

BulkCopyTimeout(IN)	Y	int	Duration of the timeout for the bulk copy activity, in seconds
ColumnMappings(IN)	N	string	If source and target have different column names, then you need to specify col mappings like Sourcecol1:targetcol1,sourcecol2:targe Otherwise not needed.
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
Database(IN)	Y	string	Target database name
DefinitionTableName(IN)	Y	string	Target table within the database
DisplayName(IN)	N	string	Name or brief description of the activity that you perform
InDataReader(IN)	Y	DataReader	DataReader that contains data from source
IntegratedSecurity(IN)	N	Boolean	Dependent on database configuration
NotifyAfter(IN)	N	int	Notify after n number of rows are copied, like a log.
OutRowsCopied(OUT)	N	int	Contains the number of rows copied in the target
Password(IN)	Y	string	Password
Schema(IN)	Y	string	Schema
Server(IN)	Y	string	Server
Timeout(IN)	N	int	Duration of the timeout in seconds
TrackRecordsCount(IN)	N	Boolean	Track the number of records which are inserted during bulk copy process. Note: Setting this option to true slows down the bulk copy performance.
UserName(IN)	Y	string	Username

Oracle Bulk Copy DataReader

Writes data into Oracle database.

Property Name	Mandatory	DataType	Description
AdditionalConnectionParameters(IN)	N	string	You can connect to other ODBC data sources through additional connection parameters. (not recommended)
AvoidOutOfMemoryIssue(IN)	N	Boolean	Check avoid out of memory issue (y/n). Sent batch size to smaller numbers for this

BatchSize(IN)	N	int	How many records to copy at a time. This option works in conjunction with setting the AvoidOutOfMemoryIssue property = true.
BulkCopyTimeout(IN)	Y	int	Duration of the timeout for the bulk copy activity, in seconds
ColumnMappings(IN)	N	string	If source and target have different column names, then you need to specify col mappings like Sourcecol1:targetcol1,sourcecol2:targe Otherwise not needed.
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
DestinationTableName(IN)	Y	string	Target table within the database
DisplayName(IN)	N	string	Name or brief description of the activity that you perform
InDataReader(IN)	Y	DataReader	DataReader contains data from source
NotifyAfter(IN)	N	int	Notify after n number of rows are copied, like a log
OutRowsCopied(OUT)	N	int	Contains the number of rows copied in the target
Password(IN)	Y	string	Password
Server(IN)	Y	string	Server
ServiceName(IN)	Y	string	Service name
Timeout(IN)	N	int	Duration of the timeout in seconds
TrackRecordsCount(IN)	N	Boolean	Tracks the number of records which are inserted during bulk copy process. Note: Setting this option to true slows down the bulk copy performance.
UserName(IN)	Y	string	Username

SQL Bulk Copy DataReader

Writes data in SQL Server database.

Property Name	Mandatory	DataType	Description
AdditionalConnectionParameters(IN)	N	string	You can connect to other ODBC data sources through additional connection parameters. (not recommended)

BatchSize(IN)	N	int	How many records to copy at a time. This option works in conjunction with setting the AvoidOutOfMemoryIssue property = true.
BulkCopyTimeout(IN)	Y	int	Duration of the timeout for the bulk copy activity, in seconds
ColumnMappings(IN)	N	string	If source and target have differecnt column names, then you need to specify col mappings like Sourcecol1:targetcol1,sourcecol2:targe Otherwise not needed.
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
Database(IN)	Y	string	Database name
DestinationTableName(IN)	Y	string	Target table within the database
DisplayName(IN)	N	string	Name or brief description of the activity that you perform
InDataReader(IN)	Y	DataReader	DataReader that contains data from source
IntegratedSecurity	N	Boolean	Dependent on database configuration
KeepIdentity(IN)	N	Boolean	While bulk insert if identify values should be used as is, otherwise new values are generated for identify columns.
NotifyAfter(IN)	N	int	Notify after n number of rows are copied, like a log.
OutRowsCopied(OUT)	N	int	Contains the number of rows copied in the target
Password(IN)	Y	string	Password
Schema(IN)	Y	string	Schema
Server(IN)	Y	string	Server name
Timeout(IN)	N	int	Duration of the timeout in seconds
UserName(IN)	Y	string	Username

Teradata Insert Data DataReader

Write data into Teradata database.

Property Name	Mandatory	DataType	Description
AdditionalConnectionParameters(IN)	N	string	You can connect to other ODBC data sources through additional connection parameters. (not recommended)

ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
DataSource(IN)	Y	string	Data source name
DisplayName(IN)	N	string	Name or brief description of the activity that you perform
InDataReader(IN)	Y	DataReader	DataReader contains data from source
IntegratedSecurity	N	Boolean	Dependent on database configuration
OutRowsCopied(OUT)	N	int	Contains the number of rows copied in the target
Password(IN)	Y	string	Password
RecordsToLoadInMemory(IN)	N	int	Number of records to load to avoid out of memory issues
TableName(IN)	Y	string	Table name to insert data
Timeout(IN)	N	int	Duration of the timeout in seconds
UpdateBatchSize(IN)	N	int	Number of commands to run in a batch
UserID(IN)	Y	string	Username

Automating Web Testing Activities

You can use Javelin to automate web browser testing. For a demo, download the following example flow: #unique_549

You can add the following web testing activities to the flowchart:

Selenium

Selenium functions let you automate operations from Selenium testing suites. The following Selenium functions are available:

Click Element by Tag Name

Performs a click on a specified element, which is identified by tag name.

Click Link By Text

Follows the specified link. You can specify a (partial) link text using the IsPartialLinkText property.

Selenium Flow End

Stops the Selenium flow and closes all associated opened web browser windows.

Execute Script

Executes a command script, which can include the command with parameters, return values, and specified timeout behavior.

Element Operation / Execute Function

Performs a click, clear, or submit action operation on the provided element.

Find Elements by Class Name

Finds an element by searching CSS class names and holds a reference to found elements in the type IWebElement[] variable assigned in the OutElement property.

For Each Element

Performs sub-flow actions for each element in a loop. You can loop through elements found by class name, name, or id.

Get Desired Capability Value

Lets you enter a capability name and returns its value.

Open Browser

Opens a specified browser and navigates to the supplied URL.

NOTE

This feature supports the following browsers:

- Internet Explorer - versions **9**, **10** and **11**
- Mozilla Firefox - versions **57 and above**
- Google Chrome - all versions

RegEx is Match

Searches for specified regular expressions.

RegEx Replace

Finds and replaces a regular expression.

Select Element Operation

Inserts an argument according to a select or clear operation. You can select or clear by index, text, or value.

Switch Window

Switches to a specified window, when more than one window is opened.

Web Page Info

Returns source, title, and URL of the specified web page.

Find Element by XPath

Searches for a string in an XML script.

- **XPath** — Defines the XPath of the element you are looking for.

Selenium by ID

Selenium by ID functions provide operations based on the Selenium ID value of an element. The following Selenium by ID functions are available:

Click by ID

Click the element that you enter in the Element ID field.

- **Element ID**

Find Element by ID

Performs a search for an element based on its ID value.

- **Element ID**

Input Data by ID

Inputs data to an element based on its ID value.

- **Element ID** — Defines the target element ID
- **Value** — Defines the input value.

Selenium by Name

Selenium by Name functions provide operations based on the Name value of an element. The following Selenium by Name functions are available:

Click by Name

Click an element identified by the Element Name.

- **Element Name**

Find Element by Name

Performs a search for an element based on its Name value.

- **Element Name**

Input Data by Name

Inputs data into an element based on its Name value.

- **Element Name**

Selenium Action API Reference

Selenium functions let you automate operations from Selenium testing suites

ExecuteScript

Executes a command script, which can include the command with parameters, return values, and specified timeout behavior.

Property Name	Mandatory	DataType	Description
Command(IN)	Y	int	Command to execute the selenium script
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
DisplayName	N	string	Name or brief description of the activity that you perform
Parameters(IN)	N	datatable	Arguments to the selenium script
ReturnValue(OUT)	Y	string	Contains the return value of the Selenium scripts
Timeout(IN)	N	int	Duration of the timeout in seconds

Automating File System Activities

The File System category consists of functions to support various operations on CSV, ZIP, and other types of files. The following file system functions are available:

Read Excel

Reads all content from a given sheet to be pushed to a data table.

Import CSV to Data Table

Imports the contents of a CSV file into a data table. You can then export the table to a database.

Example: #unique_539

Export Data Table

Exports contents of an in-memory database into a CSV or Excel file. You can then load the data table by executing a query against the above listed databases.

- **Path** — Defines the output file path.

Example: #unique_550

File Activity

The following are the available operations:

- **Create**
- **Append**
- **Delete** — Deletes a file.
- **Read All Text** — Reads file content into a ReadText property which you can keep in a variable for further processing. Can be used against any plain text format. For example, .txt, .rtf, .log.
- **Move** — Cuts and pastes a file.
- **Copy** — Copies and pastes a file.

Examples:

- #unique_551
- #unique_552
- #unique_553

GZip Compress File

Compresses the specified file using GZip compression.

Example: #unique_554

GZip Decompress File

Decompresses the specified file using GZip.

XML Doc to CSV

Converts an XML file to CSV format.

- **Xml Document Path** — Defines the input file path.
- **CSV File Path** — Defines the output file path.

File System Action API Reference

The File System category consists of functions to support various operations on CSV, zip, and other types of files.

Edit Excel Cell

Updates a given cell in Excel.

Property Name	Mandatory	DataType	Description
CellAddress(IN)	Y	string	Excel cell to edit in the first place
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
DisplayName	N	string	Name or brief description of the activity that you perform
FilePath(IN)	Y	string	Path of the excel file to be edited
NewValue(IN)	Y	string	New value with which excel cell should be updated
Timeout(IN)	N	int	Duration of the timeout in seconds

WorksheetName(IN)	N	string	Worksheet name in which cell should be updated

GZip Compress File

Compresses a file with GZip.

Property Name	Mandatory	DataType	Description
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
DisplayName	N	string	Name or brief description of the activity that you perform
InputFilePath(IN)	Y	string	Path to file to be gzipped
OutputFilePath(OUT)	Y	string	Path to where output gzip file will be placed
Timeout(IN)	N	int	Duration of the timeout in seconds

GZip DeCompress File

Decompresses a gzip file.

Property Name	Mandatory	DataType	Description
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
DisplayName	N	string	Name or brief description of the activity that you perform
InputFilePath(IN)	Y	string	File path for the file to be un-GZipped (OnlyGZIP files)
OutputFilePath(OUT)	Y	string	Path to where output file will be placed
Timeout(IN)	N	int	Duration of the timeout in seconds

ReadExcel

Read contents of an Excel file.

Property Name	Mandatory	DataType	Description
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
DisplayName	N	string	Name or brief description of the activity that you perform
FilePath(IN)	Y	string	Path of the excel file to be read

OutputElement(OUT)	Y	data table	Returns the sheet data in data table element.
SheetName	Y	string	Worksheet name to be read

XMLDocToCSV

Converts and XML file into a CSV file.

Property Name	Mandatory	DataType	Description
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
CSVFilePath(OUT)	Y	string	File path of CSV file to be generated
DisplayName	N	string	Name or brief description of the activity that you perform
Timeout(IN)	N	int	Duration of the timeout in seconds
XMLDoc(IN)	Y	xmldocument	File path of XMLDoc to be converted to CSV

Automating TDoD Activities

TDoD Functions automate operations related to the Test Data on Demand user interface that lets testers request and reserve data. The following TDoD function is available:

Authenticate User - TDoD

Use this action to input your login details for TDoD.

- Service URL
- Username
- Password

Example: #unique_555

TDOD API Reference

Resolve Variable

Resolves data maker variables in Javelin so it can be used in TDOD.

Property Name	Mandatory	DataType	Description
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
Display Name	N	string	Name or brief description of the activity that you perform
LevelID(IN)	N	int	Related to Datamaker data pool in action

Password(IN)	Y	string	TDOD password
ProjectID(IN)	N	int	Related to data pool in action
ServiceURL(IN)	Y	string	TDOD server URL
Timeout(IN)	N	int	Duration of the timeout in seconds
Username(IN)	Y	string	TDOD username
VersionID(IN)	N	int	Related to data pool in action

Automating Communication Activities

Javelin includes the following functions under the **Communication** category:

Email

Sends an email using the SMTP or MS Exchange protocol.

- Configuration File — Defines the path to a configuration file in the Configuration File field.
  - #unique_556 (SMTP, GTSendMail)
  - #unique_557 (GTSendMailExchange)
- Subject — Defines the email subject line.
- Content — Defines the email body.

Examples:

- #unique_558, (SMTP, GTSendMail)
- #unique_559 (GTSendMailExchange)

REST Post, REST Get

Makes a REST call and provides an in-memory representation of the XML response. Use the provided REST Get activity for GET calls, and use the REST Post activity for PUT, POST, DELETE, PATCH, OPTIONS, MERGE, HEAD calls.

Example: #unique_552

Download File

Downloads a file from the given URL and saves it in the specified local destination directory.

Communication API Reference

Communication related actions like Email.

Email

Sends an email using the SMTP or MS Exchange protocol.

Property Name	Mandatory	DataType	Description
ConfigurationFile(IN)	Y	Config file	Path to configuration file for Email server configuration
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.

Display Name	N	string	Name or brief description of the activity that you perform
EmailContent(IN)	Y	string	HTML or Text based email content
EmailSubject(IN)	Y	string	Email subject
Timeout(IN)	N	int	Duration of the timeout in seconds

EmailExtended

Sends an email using the SMTP or MS Exchange protocol, just like Email. If no configuration file is available for Email, you can use this action. Using Email action is preferred.

Property Name	Mandatory	DataType	Description
AttachmentFilePath(IN)	N	string	File path for any attachments to go with email
BccAddress(IN)	N	string	BCC email addresses separated by semicolon
CcAddress(IN)	N	string	CC email addresses separated by semicolon
ConfigAutoDiscoverUrl(IN)	N	Boolean	Exchange auto discover service can configure email with few values from user input
ConfigExchangeServiceUrl(IN)	Y	string	Email Exchange service configuration URL, for example https://outlook.office.365.com/EWS/Exchange.asmx.
ConfigFromAddress(IN)	Y	string	Email address of the sender. This name appears to recipients as the sender address.
ConfigFromName(IN)	Y	string	Human-readable email sender name. This name appears to recipients as the sender name.
ConfigHostName(IN)	Y	string	Host name, for example "bank.com"
ConfigPassword(IN)	Y	string	Password of the email sender
ConfigPort(IN)	N	int	Port for the outgoing mail
ConfigReplyToAddress(IN)	N	string	Reply-to address config
ConfigSsl(IN)	N	Boolean	Use SSL or not
ConfigUserName(IN)	Y	string	Username of the email sender
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
DisplayName	N	string	Name or brief description of the activity that you perform.
EmailBody(IN)	N	string	Email content
EmailSubject(IN)	N	string	Email subject line

Protocol(IN)	Y	string	Email protocol. Use one of these values: SMTP, MSExchange2013, MSExchange2010, MSExchange2010_SP1, MSExchange2010_sp2, MSExchange2007_sp1
Timeout(IN)	N	int	Duration of the timeout in seconds
ToAddress(IN)	Y	string	Recipient email addresses separated by semicolon

RestPost

Makes a REST call of one of the seven request types, PUT, POST, DELETE, PATCH, OPTIONS, MERGE, HEAD.

Property Name	Mandatory	DataType	Description
BaseUrl(IN)	Y	string	Base URL
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
DisplayName	N	string	Name or brief description of the activity that you perform
IsBasicAuthentication(IN)	Depends on Request Type	Boolean	Dependent on Request Type
IsJSON(IN)	Depends on Request Type	Boolean	Dependent on Request Type
Password(IN)	Depends on Request Type	string	Dependent on Request Type
RequestContent(IN)	Y	string	Dependent on request type
RequestType(IN)	Y	string	One of the seven request types, PUT, POST, DELETE, PATCH, OPTIONS, MERGE, HEAD.
Resource(IN)	Depends on Request Type	string	Dependent on request type
ResponseStatus(OUT)	Depends on Request Type	string	Response status from the rest call
RestResponse(OUT)	Depends on Request Type	xmlDocument	Response from the rest call
RestResponseJObject(OUT)	Depends on Request Type	Jobject	Dependent on request type
Timeout(IN)	N	int	Duration of the timeout in seconds
Username(IN)	Depends on Request Type	string	Username for the rest call

DownloadFile

Downloads a file from the given URL and saves it in the specified local destination directory.

Property Name	Mandatory	DataType	Description
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.

DisplayName	N	string	Name or brief description of the activity that you perform
FilePath(IN)	Y	string	Fully qualified file path to save it on local
Timeout(IN)	N	int	Duration of the timeout in seconds
URL(IN)	Y	string	URL from where to download the file

Automating Secure Shell Activities

Secure Shell (SSH) is an encrypted network protocol for remote login, and for network services to operate securely over an unsecured network. Javelin includes the following SSH functions:

SCP – Download Directory

Downloads a directory using Secure Copy. Define the following parameters:

- Host
- Port
- User
- Password
- Remote Directory
- Local Directory

SCP – Upload Directory

Uploads a local directory using Secure Copy. Define the following parameters:

- Host
- Port
- User
- Password
- Remote Directory
- Local Directory

SFTP – Download File

Downloads a file using SFTP. Define the following parameters:

- Host
- Port
- User
- Password
- Remote Directory
- Remote File Name
- Local file path including file name

SFTP – Get Files

Gets a list of files in a given directory using SFTP. Define the following parameters:

- Host
- Port
- User
- Password
- Remote Directory
- Output file list — Contains the result.

SFTP – Upload File

Uploads a local file using SFTP. Define the following parameters:

- Host
- Port
- User
- Password
- Remote Directory
- Remote File Name
- File to Upload
- Overwrite? — Specifies whether to overwrite existing files.

SSH – Execute Command

Executes a command through secure shell and returns the Standard Output. You can assign the Standard Output to a variable and process it further. Define the following parameters:

- Host
- Port
- User
- Password
- Command Text — Defines the input command.
- Command Result — Contains the output result.
- Command Error — Contains any error messages.

SSH Action API Reference

SSH (Secure Shell) related actions like SCP, SFTP, etc.

SCP – Download Directory

Downloads a directory using Secure Copy.

Property Name	Mandatory	DataType	Description
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
DestinationDir(IN)	Y	string	Path to download to
Display Name	N	string	Name or brief description of the activity that you perform
Host(IN)	y	string	Host to connect to remote directory

Password(IN)	Y	string	Password
Port(IN)	Y	int	Port number
RemoteDir(IN)	Y	string	Path to download from
Timeout(IN)	N	string	Duration of the timeout in seconds
User(IN)	Y	string	Username to connect to remote directory

SCP – Upload Directory

Uploads a local directory using Secure Copy.

Property Name	Mandatory	DataType	Description
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
DestinationDir(IN)	Y	string	Fully qualified path to upload from
Display Name	N	string	Name or brief description of the activity that you perform
Host(IN)	Y	string	Host to connect to remote directory
Password(IN)	N	string	Password
Port(IN)	Y	int	Port number
RemoteDir(IN)	Y	string	Fully qualified path to upload to
Timeout(IN)	N	int	Duration of the timeout in seconds
User(IN)	N	string	Username to connect to remote directory

SFTP – Download File

Downloads a file using SFTP.

Property Name	Mandatory	DataType	Description
ContinueOnError	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
Display Name	N	string	Name or brief description of the activity that you perform
FileName	Y	string	Path to download the file
Host	Y	string	Host of sftp
Password	N	string	password
PasswordAuthentication	N	Boolean	Password authentication check y/n
Port	Y	int	Sftp port

PrivateKeyAuthentication	N	Boolean	If Private Key Authentication needs to be used instead of Password authentication
PrivateKeyFile	N	string	Key File in case PrivateKeyAuthentication is set to True
PrivateKeyPassPhrase	N	string	Key Pass phrase in case PrivateKeyAuthentication is set to True
ProxyHost	N	string	If proxy should be used provide host name or IP
ProxyPassword	N	string	Proxy password of Proxy user
ProxyPort	N	int	Proxy Port
ProxyType	N	string	Proxy Type - Http / None / Open / Site / Socks4 / Socks5 / User
ProxyUserName	N	string	Proxy username
RemoteDir	Y	string	Path to remote directory
RemoteFileName	Y	string	File name on remote directory to download
Timeout	N	int	Duration of the timeout in seconds
User	N	string	Username

SFTP – Get Files

Read file names from SFTP Remote dir and return the names of files in OutFiles property.

Property Name	Mandatory	DataType	Description
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
Display Name	N	string	Name or brief description of the activity that you perform
Host(IN)	Y	string	SFTP host
OutFiles(OUT)	Y	SFTP files	Contains the list of files in remote directory
Password(IN)	Y	string	Password
Port(IN)	Y	int	Port number
RemoteDir(IN)	Y	string	Fully qualified path of remote directory to get files from
Timeout(IN)	N	int	Duration of the timeout in seconds
User(IN)	Y	string	Username to connect to SFTP

SFTP – Upload File

Uploads a local file using SFTP.

Property Name	Mandatory	DataType	Description
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
Display Name	N	string	Name or brief description of the activity that you perform
FileToUpload(IN)	Y	string	Fully qualified path to upload the file from (local)
Host(IN)	Y	string	Host to connect to remote directory
Overwrite (IN)	Y	Boolean	Overwrite if file exists – yes/no
Password(IN)	Y	string	Password
Port(IN)	Y	int	Port to connect to remote directory
RemoteDir(IN)	Y	string	Dir path to upload file to
RemoteFileName(IN)	Y	string	Fully qualified path to upload the file to
Timeout(IN)	N	int	Duration of the timeout in seconds
User(IN)	Y	string	Username to connect to remote directory

SSH – Execute Command

Executes a command through secure shell and returns the Standard Output. You can assign the Standard Output to a variable and process it further.

Property Name	Mandatory	DataType	Description
Command(IN)	Y	string	Command that needs to be executed
CommandError(OUT)	N	string	Contains the output parameter that is captured if the command encounters errors
CommandResult(OUT)	Y	string	Contains the output parameter that returns the response of command
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
Display Name	N	string	Name or brief description of the activity that you perform
Host(IN)	Y	string	Name or IP of Linux or Unix machine

Password(IN)	Y	string	Password of the user that logs in to remote machine
Timeout(IN)	N	int	Duration of the timeout in seconds
User(IN)	Y	string	Username that logs in to the remote machine

Visual Flow Examples

Use the following videos and tutorials to learn about common use cases for Javelin.

The following ZIP file contains example flows for a wide range of use cases

Download Javelin Examples

Javelin Example: Copy Table from Oracle to MSSQL Database

This short video demonstrates how to configure Data Reader and SQL Bulk Copy actions in Javelin in order to copy a Table from an Oracle to an MSSQL database.

Javelin Example: Handle Exceptions in Javelin Flows

Some actions, such as SQL Server Activities, can throw exceptions. For example, if you query an SQL table with a Truncate statement, an exception might be thrown due to a foreign key constraint in the table. This video demonstrates how to use **Try Catch** actions to handle exceptions, and how to alert a user using the following methods:

- Print a message to a local file using a **File Activity**.
- Email the exception using an **Email Activity**.

Tip: Download a template for an Email Activity configuration file from Automating Communication Activities.

Javelin Example: Loop Over Files

A flowchart can use the "Files in Folder" extension to loop over all files in a directory, and perform an action. Copy this simple example flow, and use as a template to expend upon when you automate complex file operations.

In this example, you want to automate the following task:

1. Loop over the directory C:\Users\you\Documents.
2. Read each file name.
3. Append each file name to a text file C:\Users\you\Documents\filelist.txt.

Prerequisites

- Basic understanding of programming concepts (loops)
- Basic understanding of .NET framework and C# methods.
- Basic understanding of Visual Basic .NET syntax.
- Create the following variables
- Create a flow with the following variables

- `OutFilePath` with **Type** string, **Scope** sequence, and **Default** value `"C:\Users\you\Documents\filelist.txt"`.
- `FilePath` with **Type** string , **Scope** flowchart, and default value, `"C:\Users\you\Documents\"` .

Define the Input Path

1. Click **Toolbox, Extensions, Get Files in Folder**.
   The FilesInFolder block is added to the flow.
2. Connect the Start block to the FilesInFolder block.
3. Select the FilesInFolder block and open the **Variables** pane.
4. Verify that Javelin has created an `outputFileList` variable in the **Variables** pane. The `outputFileList` variable is a string array that will store file names.
5. Enter the folder path in the **Folder Path** field of the block. Do *one* of the following:
   - Enter a previously defined variable, for example `FilePath` .
   - Enter a static value, such as `"C:\Users\you\Documents\"` .
     Format: Surround static paths with double quotes.

     TIP
     Tip: Alternatively to filling in fields inside blocks on the canavas, you can also define block properties in the **Properties** pane.

Define the Loop

Create a loop that executes a file operation for each file name in the output file list. The file operation is to append text.

1. Click **Create Loop** in the FilesInFolder block.
   A For Each File block appears. This action block implements a C#-style foreach statement.
2. Connect the FilesInFolder block to the For Each File block.
3. Double-click the ForEach File block to configure the loop:
   a. Define loop parameters by entering a temporary variable (`fileName`), and the existing output variable (`outputFileList`):
      `Foreach fileName in outputFileList`
   b. Define the loop body by clicking **Toolbox, File System**. Drag and drop a **File Operation** activity into the **Sequence** body.
   c. Configure the File Operation by choosing the **Append** action from the drop-down.
      You have created an append loop.

Define the Output Path

For each file in the list, you want to append a new line containing a file name to your output file. If the file does not exist, the Append action creates an empty file. If the file exists, the Append action adds text at the end.

1. Select the Sequence that you just created, and open the **Properties** pane.
2. Enter `OutFilePath` into the **Path** field.
3. Enter the temporary loop variable (`fileName`) and a newline character into the **Content** field.
   `fileName + System.Environment.Newline`

   TIP
   expression from the .NET framework as arguments. You can load additional libraries by clicking on the imports tab (at the bottom, next to Variables and Arguments) and entering the relevant namespace.

Execute the Javelin flow

1. Return to the flowchart. Click Save.

2. Click Start to execute the flow.
   Javelin prints the log to the **Logs** window.
3. Switch to Windows Explorer, browse to C:\Users\you\Documents, and open filelist.txt in a text editor.
   The output file now contains the file list, separated by new lines.

Delete, rename, or move the output file before executing the flow again.

Javelin Example: Push CSV file into MSSQL Database Table

This example demonstrates how to automate moving a CSV file into a Table in a database. If the table already exists, the Javelin flow appends the content to the table. Otherwise, the flow creates the table and then inserts the CSV content. This example uses an MSSQL database, and it also includes advice on how to modify this flow for other database types such as Oracle or DB2.

Download and open the flow #unique_566.

Open the **Variables** window, and set the following variables with Default values and with scope Flowchart:

Variable Type	Name	Default
String	filePath	"FilePathToCSV"
String	serverName	*"YourSqlServer"*
String	dbName	"YourSqlDatabaseName"
String	TableName	"TestCases"

Subflow Details

The flow consists of three steps: Format Create Table Statement subflow, Create Table block, and Append To Table subflow.

Format Create Table Statement Subflow

This subflow formats the SQL statement that creates the table based on the column headings in the CSV file.

1. Read the CSV file from *filePath* into a string.
2. Split the string into an array using the newline character as delimiter.
   The first element of the array now contains the table header of the CSV file.
3. Split this string to get the column names and store them in a string array.
4. Use a while loop to format the content of the string array as SQL.
5. Concatenate the string array into one string.
   Note: This step is different depending on the database type.
6. Format the string as SQL.

Create Table Block

To query the database, provide an Execute SQL Statement block with server name, database name, username, and query. Use variables to store these properties.

> **TIP**
>
> If the output table already exists (because you have created it or from a previous run of the script), then bypass the first 2 steps in the flow and configure the Append to Table subflow.

Append To Table Subflow

1. Import CSV file to Datatable. By default, the reader assumes that the top row contains headers and ignores it.
2. Convert the Datatable to an IDataReader object. This object helps with memory management.
3. Use a SQL bulk copy action to push the IDataReader object to the table.

Push CSV File into Another Type of Database Table

You can use this flow as template when you want to push a CSV file to other types of databases, such as DB2 or Oracle. Change the following properties:

- In the steps where you concatenate and format the SQL query, use the syntax for the respective database type.
- Use the provided Execute Oracle Statement and Execute DB2 Statement actions and fill in the required connection details.
  For example, for Oracle, you also provide the database port.
- Use the respective Bulk Copy action for your database in the Append To Table subflow.

Javelin Example: REST Actions

This video demonstrates how to configure REST actions and File Activity actions in Javelin. Copy this simple example flow, and use as a template to expand upon when you automate complex REST calls.

1. Get the rss.xml resource off a news site via a REST GET call.
2. Store the content in a variable of type XmlDocument.
3. Pull the news headlines out of the XML using an XPath.
4. Store the news headlines in a .txt document.

Prerequisites

1. Identify the REST GET URL of the resource.
2. Identify the XPath of the info that you want to extract.
3. Create a flow.
4. Create the following variables with scope Flowchart:
   - `Titles` variable of **Type** XmlDocument to store the REST response
   - `Strings` variable of **Type** String[] to store the XPath search results.
   - `FilePath` of **Type** string to store the output file path. Set the Default to `C:\Users\you\Documents\headlines.txt`

Get a REST Resource

1. Click **Toolbox, Communication, REST Get**, and add it to your flow.
2. Select the RESTgetBasicAuthActivity block and open the **Properties** pane.
3. Specify the following properties:
   a. **Base URL**—Enter "http://rss.cnn.com/rss".
   b. **Resource**—Enter "rss.xml".
   c. **Is Basic Auth?**— Disable this if the REST call is available for anonymous users.
   d. **Username and password**—Leave these fields empty if the REST call is available for anonymous users.
   e. **REST response:**—Choose the `Titles` variable (type XmlDocument) to store the response.

For more information on REST calls, see Automating Communication Activities.

Perform an Xpath Search on the Response

1. Click **Toolbox, Flow Chart, XPath Search**, and add it to your flow.
2. Connect the RESTgetBasicAuthActivity block to the XPathValuesActivity block .
3. Select the XPathValuesActivity block and open the **Properties** pane.
4. Specify the following properties:
   – **XmlDoc**—Enter the XmlDocument variable `Titles.`
   – **XPATH**—Enter the following XPath expression:
   `"//rss/channel/item/title"`
   – **Values**—Enter the String[] variable `Strings` to store the results.

Create a File to Store the Results

1. Click **Toolbox, File System, File Operation**, and add it to your flow.
2. Connect the XPathValuesActivity block to the FileActivity block.
3. Select the FileActivity block and open the **Properties** pane.
4. Specify the following properties:
   – **Action—**Choose **Create** to create a new file.
   – **Path**—Enter the String variable `FilePath` .
   – **Content**—Initialize the file with custom text content, for example:
   `"CNN Headlines"`

Write the Values into the File

1. Click **Toolbox, Control Flow, For Each Object**, and add it to your flow.
2. Connect the XPathValuesActivity block to the ForEach block.
3. Select the ForEach block and open the **Properties** pane.
4. Set TypeArgument to String.
5. Double click the ForEach block and specify the following properties:
   a. Define the loop parameters by entering a temporary variable (`item`), and the existing output variable (`Strings`):
   `Foreach item in Strings`
   b. Define the loop body by clicking **Toolbox, File System**. Drag and drop a **File Operation** activity into the **Sequence** body.
   c. Configure the File Operation by choosing the **Append** action from the drop-down.
   You have created an append loop.
   d. Enter the temporary loop variable (`item`) and a newline character into the **Content** field.
   `System.Environment.Newline + item`
   e. Enter `FilePath` in the **Path** field.

Execute the Javelin flow

1. Return to the flowchart. Click Save.
2. Click Start to execute the flow.
   Javelin prints the log to the **Logs** window.
3. Switch to Windows Explorer, browse to `C:\Users\you\Documents\headlines.txt` , and open filelist.txt in a text editor.
   The output file now contains the requested headlines extracted from the REST call.

Javelin Example: Subset Bulk Copy

You can use the Data Orchestration Engine Javelin to perform a universal subset. The bulk copy utility in Javelin is a fast method of moving data from one database to another, or between database types.

Prerequisites

- A source database with data
- A target database where table and column names can be different to the source, but the column count must be the same as the source.
- CA TDM Data Subset

First you define the subset of data that you want to copy.

Follow these steps:

1. Open CA TDM Data Subset.
2. Open the source database, and open the SQL window.
3. Execute an SQL SELECT statement that selects the tables including sub-tables.
4. Choose **Workflow**, **Save Workflow Subset**.
   The **Choose Workflow Tables** window opens.
5. Enable the option to select the extract **From Repository**.
6. Select the extract that you just created by enabling its checkbox.
7. Click "all" to add all tables from the "No Data Tables" pane to the "All Data tables" pane.
8. Specify the target database under **Target Destination**.
9. Click **OK** and save the subset as XML file.

Next you define and perform the bulk copy operation.

Follow these steps:

1. Open Javelin and connect.
2. Choose **Home** in the Ribbon, and choose the **Import XML/CSV** accelerator. The **Import Flow** dialog opens.
3. Select "**Subset Bulk Copy XML**" as data type.
4. Browse to the subset XML file that you previously created using CA TDM Data Subset.
5. Choose **Import**.
   Javelin generates a linear flowchart of Data Reader actions, and each of these has a Bulk Copy action nested within. For more information on actions, see Automating Database Activities.
6. Choose **Home, Save As...** and save the flow as a .vwf file.
7. Click **Home, Start**.
   The log window displays progress of the bulk copy. Wait until it completes.

The data subset, with reference tables, is copied from the source to the target.

Javelin Example: Using Selenium Actions

This video demonstrates how to configure Selenium actions in Javelin.

Prerequisites

- CA TDM 3.2 or later
- Up-to-date Selenium drivers for Firefox (see Javelin Troubleshooting)

Test Task

In this tutorial, you execute a simple exemplary test flow on a browser page:

1. Open a webbrowser and navigate to a search engine
2. Search for a string and wait for the results
3. Retrieve the title of the search results page
4. Check the page title against the expected result
5. Report success or failure

Selenium Actions

The video demonstrates a flow that uses Selenium to perform the following actions:

- Open Firefox
- Find a web page element by name
- Invoke a method
- Click a button on a web page
- Delay
- Read web page info
- End the Selenium flow
- Print success message to log
- Display an exception if the test failed

Javelin Variables Declaration and Usage

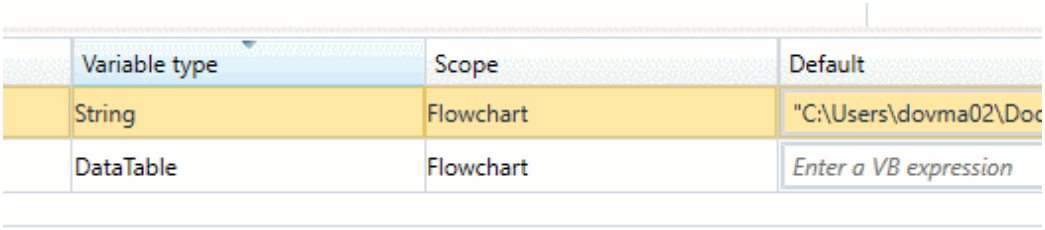
Javelin variables add flexibility in flow, and provide immense power at runtime. Use variables to store values, and activities use the stored values at runtime. You can initialize variables with a default value and you can load them from various sources, such as databases and the file system.

The **Variables** window provides access to variables and allows you to add, delete, or edit variables.

Add Variable

Click **Create Variable** and specify the following:

- **Variable Name**Specifies the variable name. Variable name must be VB.NET compliant.
- **Variable Type**Specifies the variable type. Variable Type can be any .NET data type. To browse for more exotic datatypes, double-click a type in the Variables Type column:



- **Scope**Specifies the variable scope. Scope depends on where you use the variable. Variables declared at outer scope in a flowchart are also available at inner scope. Variables declared at inner scope are not available at outer scope.
- **Default Value**Specifies a default value to a variable.
  Format: A valid Visual Basic expression

Variable Value Assignment

To assign values to variables, do one of the following:

- Enter a value in the **Default** field of the **Variables** pane.
- Use the **Primitives, Assign** action to change a variables value on the canvas.
- Change a variable values based on output from an action.
  Example: You query a database using the **SQL Server, Execute Query** action, and return a DataTable into a variable of type DataTable.
- Change variable values at runtime by using a **Primitives, Set (Boolean, Double, Integer, String)** action. The flow pauses at this action and prompts the user for input.

Use Variables

You can operate on any object using VB.net methods. Usage and operators depend on the type of variable. For example, you use the `&` operator to concatenate String values. Or, if you have a DataTable object MyDataTable, you convert it to a DataRow array using `MyDatatable.Select()`.

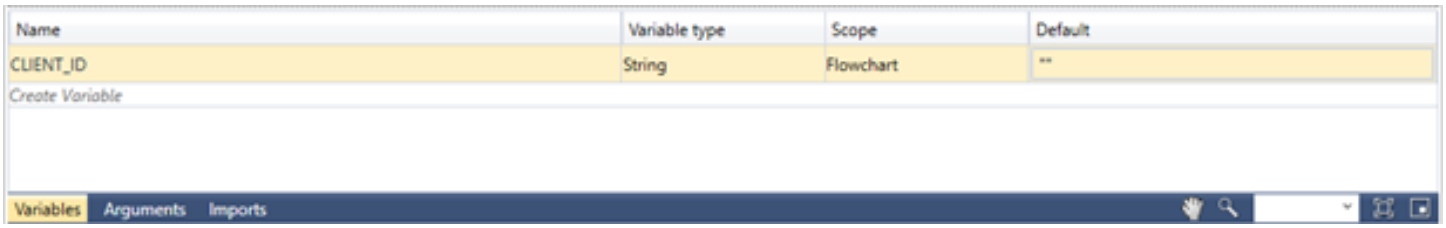
Example: Read File Into String Variable

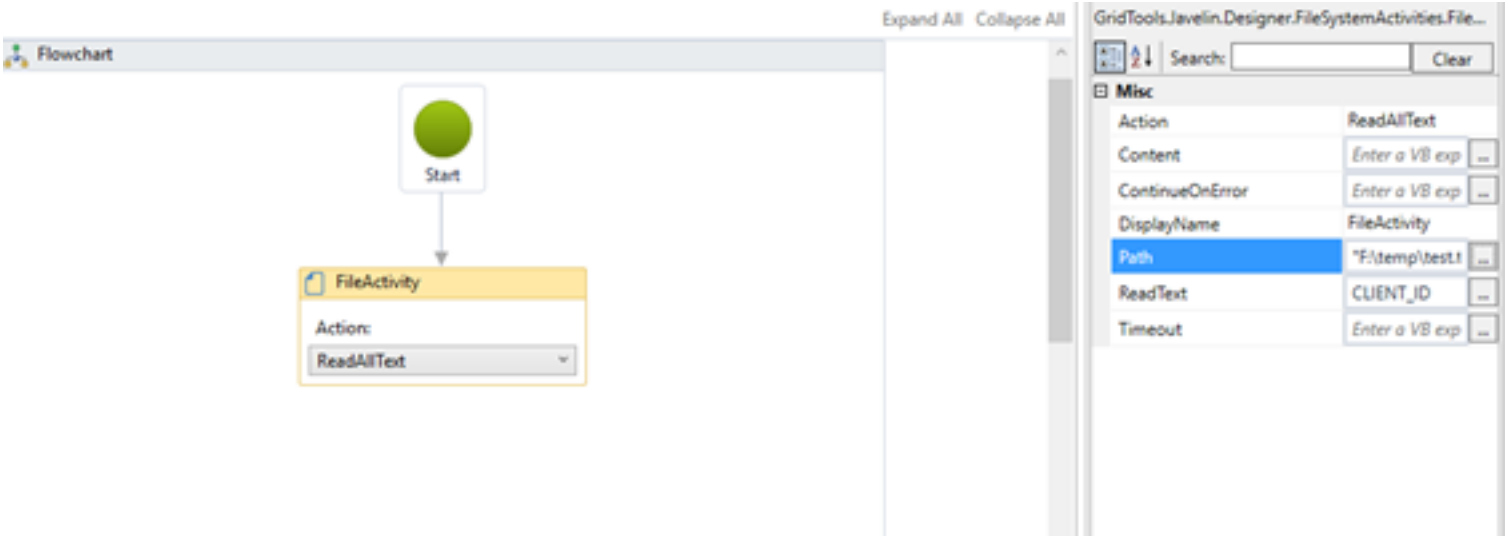
Add a string variable. You want to assign a value to it by reading content of a file, and concatenate the variable with a static string, and then print the concatenated string to the log.

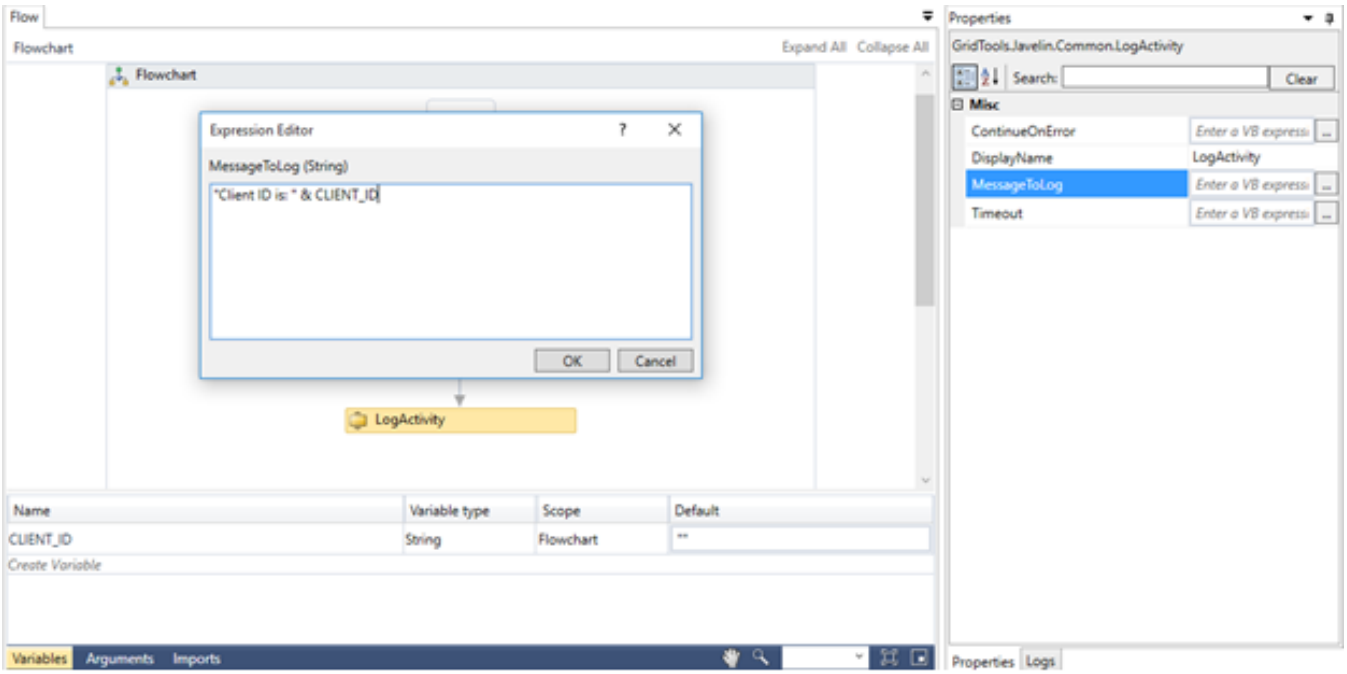
Follow these steps:

1. Declare a variable named CLIENT_ID with empty default value.
2. Drag a File Operation activity from the File System group in the Toolbox, and connect it with Start.

3. Select the ReadAllText action from the Action dropdown.
4. Specify the path of the file from which you want to read content that will be assigned to the variable.
5. Specify the CLIENT_ID in the ReadText output property.
6. Drag a Log activity from the Control Flow group in the Toolbox, and connect the FileActivity to the LogActivity.
7. Define the following MessageToLog:
   `"Client ID is: " & CLIENT_ID`
   Note: Use the Ampersand operator (&) to concatenate strings. Javelin supports VB.NET syntax in the Expression Editor.
8. Save and execute the flow.
9. Verify that the log includes the following, where 4 is the variable value which was read from the file JavelinVariable.vwf:
   Client ID is: 4





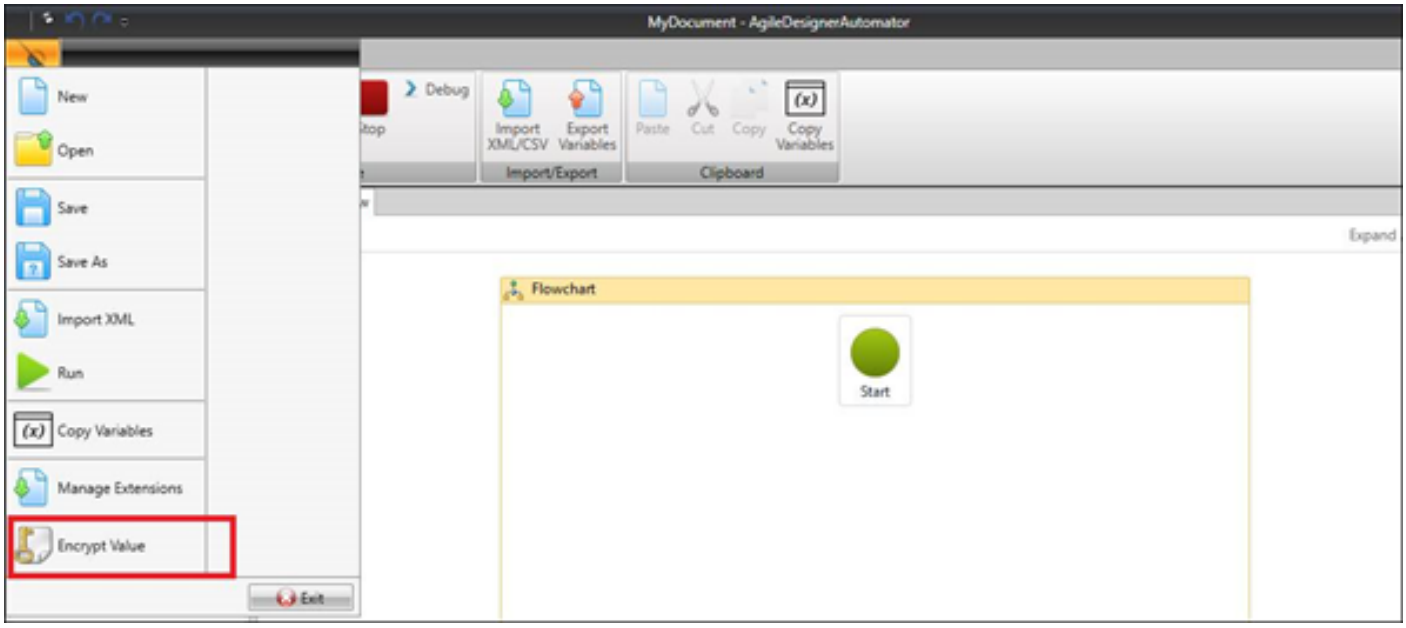


Example: Encrypt Variables

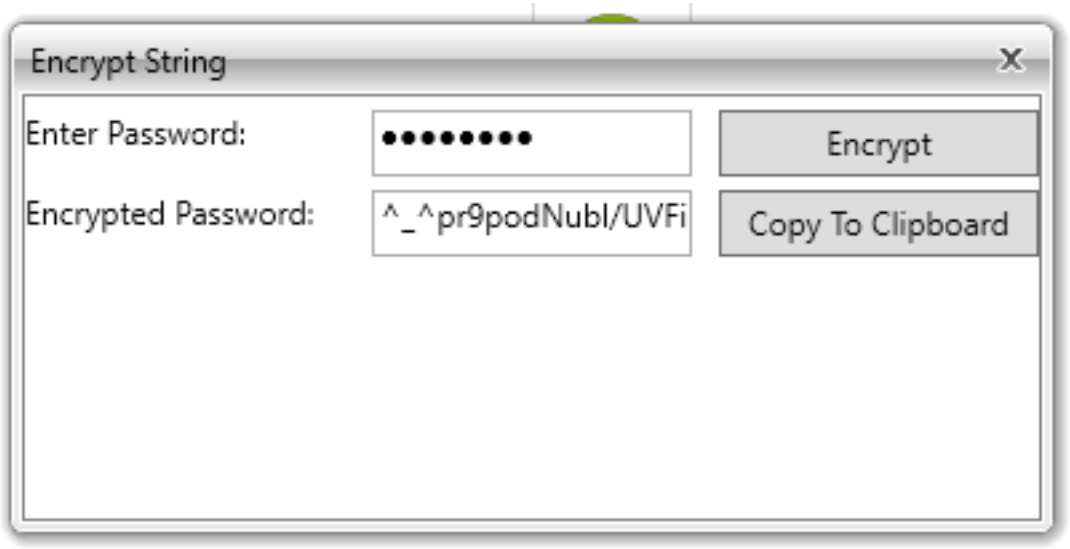
Javelin supports encrypting string variables using the `System.Security.Cryptography.AesCryptoServiceProvider` standard from the .net framework. Use this if a user wishes to input a password as a variable.

Follow these steps:

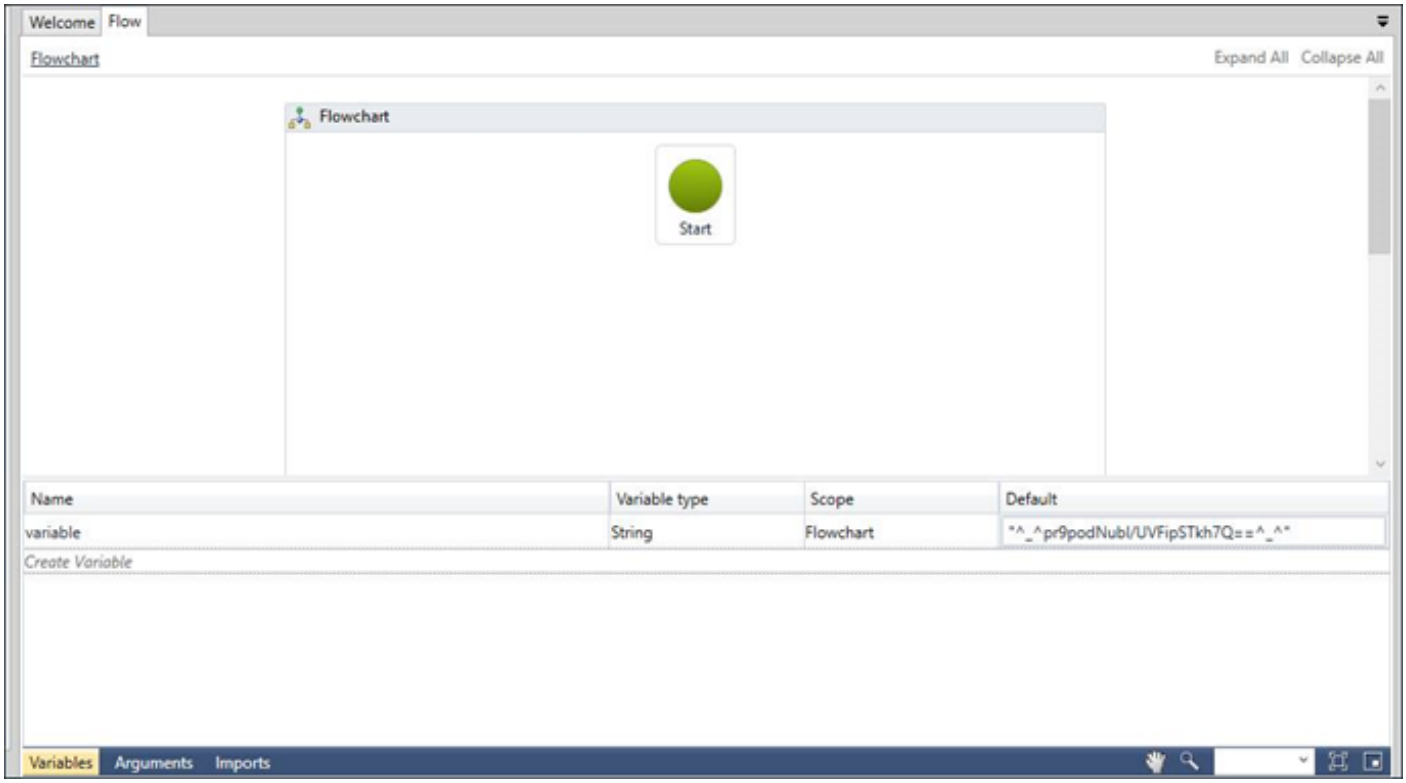
1. Click the Javelin Icon in the top left and click **Encrypt Value**.



2. Encrypt the value by entering the password and clicking Encrypt button. Copy the value to the clipboard.



3. Create a variable with the pasted encrypted password as **Default** value.



Javelin will decrypt the variable on flow execution, before executing any activity. The decrypted value is used across the flow.

Using Workflows in CA TDM Portal

As a Test Data Engineer, you want to execute actions defined through a workflow application in conjunction with a publish. You use the workflow application to define workflows, and you use CA TDM Portal to manage and execute your workflows. CA TDM Portal can execute workflow actions ad-hoc, or before or after an event.

> **NOTE**
>
> The default workflow engine for CA TDM Portal is Javelin. If a workflow engine is not installed under the default install directory, you must edit the `workflow.path.executor` parameter in application.properties file to point to the new workflow application install location. By default, the application.properties file is available at **C:\Program Files\CA\CA Test Data Manager Portal\conf**.

To manage workflows in CA TDM Portal, select a project and version, and click **Orchestration, Workflows**. The Workflows window displays all the workflows that you have added to the current project and version. You use this window to add (upload), delete, list, modify, and download workflows.

Add Workflows

1. Create the workflow file (and optionally the variables file) in the workflow application and save them locally.
2. Open CA TDM Portal and navigate to **Orchestration, Workflows.**
3. Click **New Workflow**.
4. Define a **Workflow Name**.
5. Upload the **Workflow** file in .vwf format.
6. (Optional) Upload the **Variable Content** file in .csv format.
   Tip: You can drag and drop files into the web interface, or click the upload buttons to open a file browser.
7. Click **Save**.

Edit Workflow Properties

To modify properties of added workflows, navigate to **Orchestration, Workflows** and click an entry in the **Workflows** list. You can edit the following fields:

- Rename the workflow
- Replace the workflow file (.vwf)
- Replace or remove the variable content file (.csv)

View and Modify Workflows

To view or edit the workflows, use the workflow application.

1. Download the workflow files from the CA TDM Portal.
2. Open the files in workflow application to view and modify them.
3. Save the files locally.
4. Return to CA TDM Portal, edit the flow entry, and replace the workflow and variable content files.

Execute Workflow Actions

You define and add actions with code type WORKFLOW in TDM Web Portal the same way as code types Host or SQL.

Prerequisites:

1. Select project and version
2. Navigate to **Orchestration, Workflows** and add a workflow.
3. Navigate to **Generators** and create or edit a generator.

To create and execute workflow actions for a generator, do the following:

1. Click **Actions** and create a **Publish Action** or **Table Action**.
2. Define a **Name** and **Description** for the action.
3. Select **Type: WORKFLOW**.
4. Select a **Workflow** that you have previously uploaded.
5. Define what you want to use as variables:
   – **Default Variables**Uses the variables uploaded with the flow. Use this if you call a standalone workflow that either requires no external variables, or uses a pre-defined list of variables.
   – **Published Data**Uses data from a published table to define the variables that are passed to the workflow. If there are multiple rows of published data, the action executes the workflow multiple times.
     • Select the **Table** that you publish to. You defined these tables in the generator screen.
   – **Direct SQL**Uses the results of a SQL SELECT script to define the variables that are passed to the workflow. If there are multiple rows returned from the script, the action executes the workflow multiple times.
     • Select a fixed **Connection Profile**, or configure the action to use the publish **Data Target**.
     • Enter the **SQL** script that defines the variables.
6. (Published Data and Direct SQL only) Control the maximum number of times that the workflow is executed in cases where there are multiple sets of variables.
   – **Maximum Records** — Specifies the maximum number of returned records to use as variables. If the number is less than or equal to the number of rows of available variable data, then the rows of data to use are chosen at random from the available set; otherwise all data is used.
7. Schedule workflow actions for *one or both* of the following action types:
   – **Pre** — executes the action before the publish starts.
   Note: If you use "Published Data" as variables, the action cannot run pre-publish or adhoc because the published data would not yet be available.
   – **Post** — executes the action after the publish has completed.
8. (Optional) Define one of the following success criteria if **Success Required**:
   – **Results** string
   – **Row Count** value
9. Click Save.
10. Return to the Generator and click Publish.

TIP

Click the **Execute** button next to an action to run the action adhoc.

Example: Execute a Workflow that uses published data as variables

You have published data to the specified table, and you want to use the data to generate the workflow variables file. The workflow is executed multiple times, once for each row of data that has been published. Only the published columns whose names match the ones in the registered variables file are passed. If you have set maxRecords to less than the number of published rows, then the workflow is only executed maxRecords times, with the rows of data being picked randomly from the published data.

Using Workflows for Datamaker during Publish

Datamaker contains functionality to register and use Workflows. Register Workflows by saving the Workflow file and the variables file in the Repository. You can execute a Workflow action as an "Ad-hoc" action or during publish as a "pre-publish" or "post-publish" action.

NOTE

The default workflow engine for Datamaker is Javelin.

Registering Workflows

To register and view Workflows in Datamaker, select a project version in the project explorer and click menu option **Tools**, **Register Workflows**. The Register Workflows window appears.

The Register Workflows window displays all the existing Workflows for the current project version. The following are the relevant actions:

- View the Workflow
- Retrieve Files

To register a Workflow, click on the Plus button at the top of the Register Workflows window and enter the Workflow name.

You are then prompted for the following:

- The Workflow filename.
- The default variable filename.

A registered Workflow can be viewed by selecting the program on the left pane of the Register Workflows window.

You then can do one of the following:

- Choose **View Workflow** to launch your workflow engine for viewing the flow.
- Choose **Retrieve Files** to restore the registered files for the Workflow to a selected folder.

Manipulating the Workflow

By right-clicking the program name in the left pane of the Register Workflows, a menu appears. The menu choices and descriptions for manipulating the Workflow are:

- Edit

The registered files of an existing Workflow can be replaced by selecting the program in the left pane of the Register Workflows window. Next right click the menu option Edit. It is possible to replace either the flow file or the default variable file or both.

- Copy

The Copy menu option copies the selected Workflow to any project version in the current repository. A project version is selected by opening the **project selection** screen.

- Rename

The Rename menu option allows renaming of a selected Workflow.

- Delete

The Delete menu option is used to delete the selected Workflow after confirmation. A Workflow which is already used in an action cannot be deleted.

Workflow Actions

Adding an action

A new Workflow action can be added the same way as other action types by selecting Workflow action code type.

Follow these steps:

1. Select a Workflow code type. In the **Registered Workflow** drop-down list, select a Workflow.
2. Select the stored Workflow.

A script can be added.

If adding a Post-Publish action, select the box **Use Published Data as variables** to create the variable file for the registered Workflow. If not, the default variable file is used.

Workflow Action Script

A Workflow action allows for adding a script to retrieve variables for the selected Workflow. Set the DB connection for the added script. The script is checked when the action is saved. If a problem is encountered or there are any variables that are not set by the script, a message appears. The script is run when the Workflow action is executed. For more information about the Variable file, refer to Variables File section.

Use Published Data as variables:

This option is only available for Post-Publish actions. When the box is selected, the Table drop down list is enabled to select a publish table. The published data for the selected table is used to retrieve the variables for the selected Workflow. For more information about the Variable file, refer to the Variables File section.

When using publish data for variables, if Publish Table is not selected when publishing data, a table must be selected (with a pop-up window). When using remote or batch publish, if the publish table is not selected, the default variable file is used for the Workflow execution.

Workflow action, Maximum Records to Use:

When using a script or published data, the number of records to use are set. For example, if "Maximum record to use" is 3, three random records are selected from the data and the Workflow is executed three times using this data.

Edit, Delete Workflow Action:

In the Maintain project screen, the right click menu options for a Workflow action allows for editing and deletion of a Workflow action.

Workflow Action Execution, Ad-hoc, Pre and Post publish:

In the Maintain project screen, the right click menu option, "execute" allows Ad-hoc execution of Workflow. In the case of using a script, the data from the script is used to create the variable file. Otherwise the default variable file (if loaded) is used for the execution of the Workflow action. You are asked to set the folder for saving the Workflow execution log file.

In the case of publishing data, the Workflow action is executed if the action is not an ad-hoc action. The information about these executions can be found in the publish log file. The restored Workflow files and log file can be found in the Publish Data directory.

Workflow actions can be also executed with a Batch engine.

Variables File:

When registering a Workflow, the default variables file can be loaded optionally if a script is not defined or publish data is not selected. If the default variables file is loaded, the Workflow action is executed using the default variables, otherwise the action is executed with no variables file.

Structure of the Variables file:

The variables file is a CSV file should have only one record of data. This example file has three columns:

The Name column contains the names of the variables.

The Value column contains the data for the variables named.

The Scope column is the Flowchart by default.

Name	Value	Scope
ID	100	Flowchart
Firstname	Abc	Flowchart

Lastname	Xyz	Flowchart
Email	abc@com	Flowchart

Variables file using a script or published data:

To create the variables file dynamically, one random record is selected from the result (script or the selected published table):

ID	Lastname	Firstname	Middlename	Email	Age
200	Brown	John	George	john@ca.com	30

In the case of having a default variables file, the variable names of the record which are the same as the default file are selected. In this example the variables file will be as follows:

Name	Value	Scope
ID	200	Flowchart
Firstname	John	Flowchart
Lastname	Brown	Flowchart

Note: No record for Email
If no default variables file is registered, the variables file will be as follows:

Name	Value	Scope
ID	200	Flowchart
Lastname	Brown	Flowchart
Firstname	John	Flowchart
Middlename	George	Flowchart
Email	john@ca.com	Flowchart
Age	30	Flowchart

Import Extensions into Javelin

To create a Javelin flow using different applications, you import the relevant extensions in to Javelin.

Prerequisites:

1. Save all files.
2. Clean and build the solution.

Follow these steps:

1. Launch the Javelin application and click the Javelin logo (in the Ribbon menu icon at top left corner).
2. Click **Manage Extensions**.
3. Locate and open the required *.dll* file from the following directory:
   `C:\Program Files (x86)\Grid-Tools\Javelin`
4. Click **Show Extensions**.
5. Select the actions, provide an existing or a new group name for Javelin Toolbox.
6. Click **Import Selected**, click **OK.**
7. Restart Javelin.
   You can now see the imported activities in the toolbox under the specified extension.

Develop and Deploy Custom Extensions

As a test data engineer, you may need to develop custom extensions for the CA Test Data Manager Javelin utility. In the following scenario, you develop an extension that adds two input numbers and binds the result to a variable.

Prerequisites

- Visual Studio 2012 (or higher) development environment
- Access to required Javelin DLLs available in Javelin installation directory
- C# knowledge

Optionally, download the .cs and .xaml files used in this example for reference:

- content/dam/broadcom/techdocs/us/en/assets/docops/tdm/addtwonumbers.cs
- content/dam/broadcom/techdocs/us/en/assets/docops/tdm/addtwonumbersdesigner.xaml

Create Project in Visual Studio

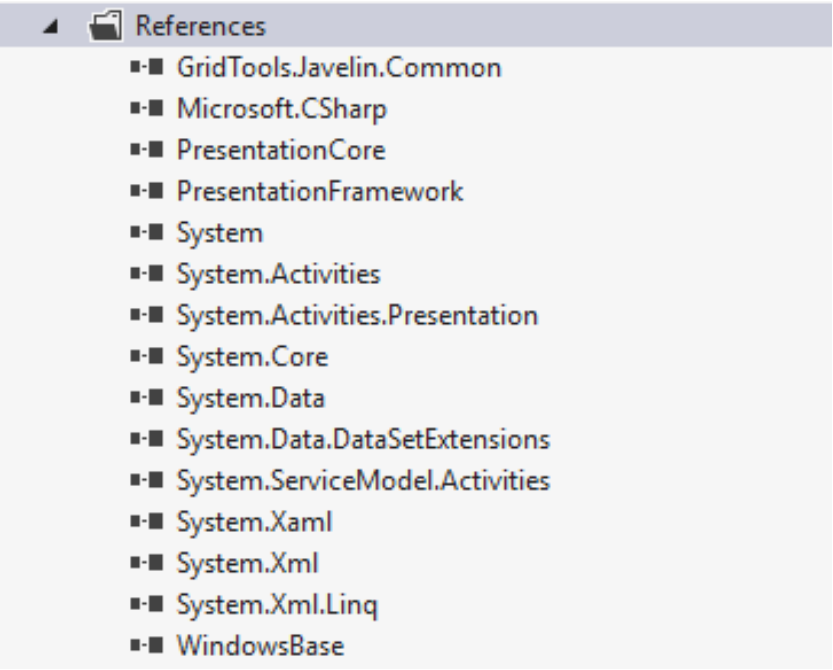
Note: This procedure refers to the Visual Studio 2012 user interface. If you are using a different version of Visual Studio, refer to specific product documentation regarding UI navigation.

1. Launch Visual Studio
2. Click **File, New, Project**.
   The New Project dialog opens.
3. Expand **Installed, Templates**, and select **Visual C#**.
4. Select **Class Library** from the installed templates list.
5. Define the following, and click **OK**.
   - **Name**
     Specifies the name of your Javelin extension. For example, AddTwoNumbersExtension.
   - **Location**
     Specifies the path to save the solution. For example, your home directory.

Define Project References

1. Expand the project in **Solution Explorer**. Right click **References** and click **Add Reference**.
   The **Reference Manager** dialog opens.
2. Go to **Assemblies, Framework** in the Reference Manager dialog.
3. Select the following from the References list:
   - PresentationCore
   - PresentationFramework
   - System.Activities
   - System.Activities.Presentation
   - System.Drawing
   - System.Drawing.Design
   - System.Xaml
   - System.ServiceModel
   - WindowsBase
4. Browse and select the file GridTools.Javelin.Common.dll from the Javelin installation folder.
5. Click OK.
   GridTools.Javelin.Common is added to the References.

The References should now look like this:



Develop the Code

1. Expand the project from the Solution Explorer. Right-click `Class1.cs` and give it a name, for example, to `AddTwoNumbers.cs` .
2. Click **Yes** when Visual Studio prompts you to refactor the class name accordingly.
3. Extend the class `JavelinNativeActivity` and inherit and override its `Execute` method:

```
public class AddTwoNumbers : JavelinNativeActivity
{
 protected override void Execute(System.Activities.NativeActivityContext context)
 {
   try
   {
      base.Execute(context);
    }
   catch (Exception ex)
   {
      BusinessException bex = new BusinessException(ex.Message, DisplayName, ex.InnerException);
      if (!continueOnErrorInExecMode)
      {
          throw bex;
      }
   }
  }
}
```

4. Declare `InArgument` for each input that the user provides to the activity.
   Examples: `InArgument<string>` for string type, `InArgument<int>` for int type, `InArgument<DataTable>` for DataTable type.
5. Declare `OutArgument` for each output that you set after the execution of the activity.

Examples: `OutArgument<string>` for string type, `OutArgument<int>` for int type, `OutArgument<DataTable>` for DataTable type and so on.

6. Use the the attribute `RequiredArgument` to mark the necessary arguments as required, and to add constraints to the activity:

```
using GridTools.Javelin.Common;
using System;
using System.Activities;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace CA.Javelin.Extension
{
    public class AddTwoNumbers : JavelinNativeActivity
      {
        [RequiredArgument]
         public InArgument<int> FirstNumber { get; set; }
         [RequiredArgument]
         public InArgument<int> SecondNumber { get; set; }
         [RequiredArgument]
         public OutArgument<int> TheSum { get; set; }
          protected override void Execute(System.Activities.NativeActivityContext context)
          {
            try
              {
                base.Execute(context);
                 int tempFirstNumber = FirstNumber.Get(context);
                int tempSecondNumber = SecondNumber.Get(context);
                 int result = tempFirstNumber + tempSecondNumber;
                 TheSum.Set(context, result);
              }
                catch (Exception ex)
                   {
                    BusinessException bex = new BusinessException(ex.Message, DisplayName,
 ex.InnerException);
                    if (!continueOnErrorInExecMode)
                          {
                            throw bex;
                           }
                    }
           }
       }
}
```

7. Save the code and build the solution.

After building the solution, all the required files are available in the default projects folder. Typically the default projects folder is
`..\Documents\Visual Studio 2012\Projects\`*`Solution Name`*`\`*`Project Name`*`\bin\Debug\`

(Optional) Configure the User Interface of the Extension

You can choose to customize the user interface for your extension now. The following procedure demonstrates how to add a custom icon to the toolbox and to the canvas widget, and how to lay out more user-friendly input fields on the canvas.

Tip: If you decide to implement the extension without a custom UI, you can skip to the section titled "Deploy the Custom Extension."

Define a Custom Toolbox Icon

The following section describes how to override the default bitmap image shown for an extension in the toolbox palette.

1. Find an image (.png format) that you want to use as an icon. In this example, we use the file resfolder.png.
2. Create a folder called 'images' in the solution folder, and reference the folder in the solution.
3. Open the build **Properties**, expand **Advanced,** and set the **Build Action** to build as **Embedded Resource**.
4. Expand the project in **Solution Explorer**. Right click **References** and click **Add Reference**. Select the following from the References list and click OK:
   – System.Drawing
   – System.Drawing.Design
5. Open the class file 'AddTwoNumbers.cs' and add the following lines:
   a. A Using System.Drawing statement
   b. An Internal class EmbeddedResourceFinder
   c. A ToolboxBitmap attribute and a reference to the the image to be used when displaying activity in Toolbox Control.

```
[ToolboxBitmap(typeof(EmbeddedResourceFinder),
"CA.Javelin.Extension.images.resfolder.png")]
 using System;
 using System.Activities;
 using System.Collections.Generic;
 using System.ComponentModel;using System.Drawing;
 using System.IO;
 using System.Linq;
 using System.Text;
 using System.Threading.Tasks;
 using GridTools.Javelin.Common;

   internal class EmbeddedResourceFinder{
   }

 namespace CA.Javelin.Extension
 {
  [ToolboxBitmap(typeof(EmbeddedResourceFinder), "CA.Javelin.Extension.images.resfolder.png")]
  public class AddTwoNumbers : JavelinNativeActivity
  {
```

Let Users Input Argument Values on the Canvas

You can choose to customize the extension widget so that a user can input argument values on the Javelin canvas. The less user-friendly alternative is that users enter properties directly in the properties pane.

1. Right-click the solution in the solution explorer, click **Add, New Item, Visual C#, Workflow, Activity Designer**. Name the file AddTwoNumbersDesigner.xaml and click **Add**.
   You added an activity designer file to your project.
2. Open the activity designer file in the Visual Studio window.

3. Define your default namespace:
   `x:Class="CA.Javelin.Extension.AddTwoNumbersDesigner"`
4. Return to your AddTwoNumbers.cs file and add the following code:
   `[Designer(typeof(AddTwoNumbersDesigner))]` You have specified the class used to implement design time services for a component, which in this case is the default namespace we just defined on our activity designer.
5. Return to the AddTwoNumbersDesinger.xaml file and add the following code:

```
<sap:ActivityDesigner x:Class="CA.Javelin.Extension.AddTwoNumbersDesigner"
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  xmlns:s="clr-namespace:System;assembly=mscorlib"
  xmlns:sap="clr-namespace:System.Activities.Presentation;assembly=System.Activities.Presentation"
  xmlns:sapv="clr-namespace:System.Activities.Presentation.View;assembly=System.Activities.Presentation"
  xmlns:sapc="clr-
namespace:System.Activities.Presentation.Converters;assembly=System.Activities.Presentation" Height="82"
 Width="243">
```

   You declared namespaces and set the physical size of the extension widget on the canvas. In this example, we set the height to 82 and the width to 243
   Tip: Click and drag the widget corners on the canvas to adjust the size manually.
6. Add an `sap:ActivityDesigner.Resources` resources block and use the resource dictionary element to create a key for the `ArgumentToExpressionConverter` :

```
<sap:ActivityDesigner.Resources>
        <ResourceDictionary>
            <sapc:ArgumentToExpressionConverter x:Key="ArgumentToExpressionConverter"/>
        </ResourceDictionary>
</sap:ActivityDesigner.Resources>
```

7. Define the row and column definitions for the widget using the grid layout panel as shown in the following code snippet:

```
<Grid Margin="0,1,0,-75">
   <Grid.ColumnDefinitions>
       <ColumnDefinition/>
       <ColumnDefinition Width="0*"/>
   </Grid.ColumnDefinitions>
   <Grid Height="90" VerticalAlignment="Top" >
       <Grid.RowDefinitions>
           <RowDefinition Height="42*" />
           <RowDefinition Height="5" />
           <!--<RowDefinition Height="*" />
           <RowDefinition Height="5" />-->
           <RowDefinition Height="28*" />
           <RowDefinition Height="15*"/>
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
           <ColumnDefinition Width="46*"/>
           <ColumnDefinition Width="30*"/>
           <ColumnDefinition Width="6"/>
           <ColumnDefinition Width="9"/>
           <ColumnDefinition Width="5"/>
           <ColumnDefinition Width="71*"/>
           <ColumnDefinition Width="32*"/>
           <ColumnDefinition Width="30*"/>
```

```
        </Grid.ColumnDefinitions>
...
```

8. Add one Textblock element for each argument that you want to input on the canvas, and their position within the grid. In this example, you want to input two arguments on the canvas, the first and second number.
   Tip: Click and drag the elements on the canvas to position them as required.

```
<TextBlock Text="First Number:" Grid.Row="0" Grid.Column="0" Grid.ColumnSpan="2" Margin="0,-4,0,27"
 HorizontalAlignment="Left" Width="76"/>  <TextBlock Text="Second Number:" Grid.Row="0" Grid.Column="0"
 Grid.ColumnSpan="5" Margin="0,26,0,0" Grid.RowSpan="2"/>
```

9. Define an instance of the ExpressionTextBox Class (and its position within the grid) to create a field on the widget where users input the argument for the `InArgument<int>FirstNumber` object in the AddTwoNumbers.cs file.

```
<sapv:ExpressionTextBox
  HintText="First Number"
  Expression="{Binding Path=ModelItem.FirstNumber, Mode=TwoWay, Converter={StaticResource
 ArgumentToExpressionConverter}, ConverterParameter=In }"
  ExpressionType="{Binding ModelItem.Properties[FirstNumber].PropertyType.GenericTypeArguments[0]}"
  OwnerActivity="{Binding Path=ModelItem}"
  Margin="24,-4,0,0"
  Grid.Row="0"
  Grid.Column="5"
  MaxLines="1" Height="23" VerticalAlignment="Top" Grid.ColumnSpan="3" />
```

   If you now hover the mouse to the right of the 'First Number' text on the widget, you see a rectangular box.

10. Define a second instance of the ExpressionTextBox Class (and its position within the grid) to create a field on the widget where users input the argument for the `InArgument<int> SecondNumber` object in the AddTwoNumbers.cs file:

```
<sapv:ExpressionTextBox
  HintText="Second Number"
  Expression="{Binding Path=ModelItem.SecondNumber, Mode=TwoWay, Converter={StaticResource
 ArgumentToExpressionConverter}, ConverterParameter=In }"
  ExpressionType="{Binding ModelItem.Properties[SecondNumber].PropertyType.GenericTypeArguments[0]}"
  OwnerActivity="{Binding Path=ModelItem}"
  Margin="24,19,-2,0"
  MaxLines="1" Height="23" VerticalAlignment="Top" Grid.Column="5" Grid.ColumnSpan="3" />
```

   If you now hover the mouse to the right of the 'Second Number' text on the widget, you see a rectangular box.

The code used for the bindings is from a standard template given on MSDN. For more information, see msdn.microsoft.c

Define a Custom Widget Icon

The following section describes how to override the default bitmap image shown for your extension's widget on the canvas.

1. Add an ActivityDesigner.icon element and the following snippet to the code. Add the desired image to your images file in your solution (in this example, resfolderIcon.png).

```
<sap:ActivityDesigner.Icon>
        <DrawingBrush>
            <DrawingBrush.Drawing>
                <ImageDrawing>
                    <ImageDrawing.Rect>
                        <Rect Location="5,5" Size="5,5" />
                    </ImageDrawing.Rect>
                    <ImageDrawing.ImageSource>
```

```
                        <BitmapImage UriSource="/Ca.Javelin.Extension;component/images/
resfolderIcon.png" />
                    </ImageDrawing.ImageSource>
                </ImageDrawing>
            </DrawingBrush.Drawing>
        </DrawingBrush>
    </sap:ActivityDesigner.Icon>
```

2. Open the Solution Explorer and click the Image Properties. Expand **Advanced,** and set the **Build Action** to **Resource**.

Deploy the Custom Extension

1. Save all files.
2. Clean and build the solution.

Follow these steps:

1. Copy the AddTwoNumbersExtension.dll file to the following directory:
   `C:\Program Files (x86)\Grid-Tools\Javelin\Extensions`
2. Launch the Javelin application and click the Javelin logo (in the Ribbon menu icon at top left corner).
3. Click Manage Extensions.
4. Locate and open the AddTwoNumbersExtension.dll file from the following directory:
   `C:\Program Files (x86)\Grid-Tools\Javelin\Extensions`
5. Click Show Extensions.
6. Select the actions, provide an existing or a new group name for Javelin Toolbox.
7. Click **Import Selected**, then click **OK.**
8. Restart Javelin.
   You can now see the imported activities in the toolbox under the specified group.

Create a Flow Using Your Custom Extension

After you write and deploy the extension, you want to use it in a flow. The example extension adds two integers and binds the result to a variable

1. Open Javelin.
2. Click the Number Operations dropdown in the Toolbox, and choose AddTwoNumbers.
3. Drag the AddTwoNumbers action onto the canvas and connect it to the start node.
4. Enter two integers into the widget, for example 5 and 7.
5. Click the variables pane.
6. Create an integer variable to hold the answer. For example, name the variable TheAnswer.
7. Enter this TheAnswer variable in ‘TheSum’ field in the properties pane.
8. Add a log action to output ‘TheAnswer’ to the log.
9. Join the log action to the AddTwoNumbers action.
10. Click on the log action, go to the properties pane, and enter the following string in the MessageToLog field:
    `"The Sum of these two numbers is: " & TheAnswer`
11. Save and run the flow.
12. Click on the log pane to view the output.

Extensions Actions API Reference

GetFilesinFolder

Reads files list in a given folder.

Property Name	Mandatory	DataType	Description
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
DisplayName	N	string	Name or brief description of the activity that you perform.
InputFolder(IN)	Y	string	Fully qualified path of folder to read from
OutputFiles(IN)	Y	string	Returns list of files in Input Folder
Timeout(IN)	N	int	Duration of the timeout in seconds

JavaScript Alert

While you are testing through the UI, some alerts may be displayed. Use this action to handle those pop-ups.

Property Name	Mandatory	DataType	Description
Action	Y	string	It's a drop-down with one of the following options: "Accept", "Dismiss", or "Send Text". Specify the action you want to do on the alert dialog
ContinueOnError(IN)	N	Boolean	If the step encounters an error, should the workflow move on or stop. Set this to True if you want it to continue on error.
DisplayName	N	string	Name or brief description of the activity that you perform.
OutAlertText(OUT)	N	string	An output parameter, containing the text that was displayed in the label of the alert dialog.
TextToSend	N	string	If you selected "Send Text" as the action, this is the text that is sent. If you selected any other action, this property is not used.
Timeout(IN)	N	int	Duration of the timeout in seconds

Run Javelin in Batch Mode

For many Test Data Engineers, it is desirable to run Javelin in batch mode. This allows us the option of bypassing the Javelin UI for reasons of convenience and memory efficiency. To do this is straightforward, though it does require some basic understanding of command line syntax.

1. Create a flow and declare your variables in the variables pane.
2. Create a CSV file and define your variables in three columns named *Name, Value, Scope.*
3. Create an XML file that defines the paths to the flow and variables files.

```
<?xml version="1.0" encoding="UTF-8"?>
<GTJavelinJob>
    <FlowPath>C:\Users\name\myflow.vwf</FlowPath>
    <VariableFilePath>C:\Users\name\myvariables.csv</VariableFilePath>
    <LogFile>C:\Users\name\mylogfile.log</LogFile>
    <outdir>C:\Users\name\mydirectory</outdir>
</GTJavelinJob>
```

4. Point the Javelin executable at the XML file.
   `"C:\Program Files (x86)\Grid-Tools\Javelin\JavelinExecutor.exe" file="C:\Users\name \myautomation.xml"`

Example

You want to run a flow that looks in a directory specified by *filePath,* and outputs the names of all the files in that directory to a .txt file specified by *OutFilePath*.

> **TIP**
>
> Tip: You can write the XML file and the CSV file manually, or write a script that generates them. If you generate the files through a script, you may want to pass command line arguments to the script. In this example, we pass two paths into the script. In this case, use %*argument_number* to reference the commandline arguments. For example, %1 stands for the first command line argument, %2 stands for the second.

1. You create the flow.
2. You declare the variables *filePath* and *OutFilePath* in the flow.
3. You create the following batch file (suffix .cmd):
   a. The batch file reads two command line arguments that set the variables *filePath* and *OutFilePath.*
   b. The batch file generates a CSV file that defines the variables as follows:

Name	Value	Scope
filePath	%1	Flowchart
OutFilePath	%2	Flowchart

   c. The batch file creates the XML file that defines where Javelin looks for the flow and variables files.
4. You open the command prompt and execute the batch file. You pass it command line arguments, in this example, filePath and OutFilePath.
   The batch file runs Javelin.

```
REM @echo off
if [%1]==[]  goto error
```

```
if [%2]==[]  goto error

(
echo ^<?xml version="1.0" encoding="UTF-8"?^>
echo ^<GTJavelinJob^>
echo ^<FlowPath^>C:\Users\myname\Documents\Javelin\FilesInFolderBatch\FilesInFolder.vwf^</FlowPath^>
echo ^<VariableFilePath^>C:\Users\myname\Documents\Javelin\FilesInFolderBatch\FilesinFolder_runjav.csv^</
VariableFilePath^>
echo ^<LogFile^>FilesinFolder_runjav.log^</LogFile^>
echo ^<outdir^>C:\Users\myname\Documents\Javelin\FilesInFolderBatch\^</outdir^>
echo ^</GTJavelinJob^>
) > C:\Users\myname\Documents\Javelin\FilesInFolderBatch\FilesinFolder_runjav.xml

(
echo Name,Value,Scope
)   > C:\Users\myname\Documents\Javelin\FilesInFolderBatch\FilesinFolder_runjav.csv
(
echo filePath , %1, Flowchart
)   >> C:\Users\myname\Documents\Javelin\FilesInFolderBatch\FilesinFolder_runjav.csv
(
echo OutFilePath , %2, Flowchart
)   >> C:\Users\myname\Documents\Javelin\FilesInFolderBatch\FilesinFolder_runjav.csv


"C:\Program Files (x86)\Grid-Tools\Javelin\JavelinExecutor.exe" file="C:\Users\myname\Documents\Javelin
\FilesInFolderBatch\FilesinFolder_runjav.xml"
goto end

:error
echo.
echo ** ERROR SPECIFYING PARAMETERS
echo.
echo ** TO RUN: filePath OutFilePath
echo.
:end
```

The Following annotations describe what each line does:

- Line 8: Specify the path to the VWF flow file.
- Line 9: Specify the path to the CSV variables file.
- Line 10: Specify the name of the log file
- Line 11: Specify the directory where the log file is created
- Line 13: Specify the name of an XML file into which lines 6-13 are printed. Later, we point the Javelin Executable at this file.
  The > command creates the file if it exists, or overwrites if it does not exist.
- Lines 16-17: Write the strings *Name, Value, Scope* into the heading row of the CSV file.
- Lines 19-23: Append the first and second command line argument as rows in the CSV file. These rows define the variables.
- Line 26: Point the Javelin executable at the XML file created in line 13.

TIP

Download the command script example here:

- #unique_575

Javelin Troubleshooting

TIP

Syntax errors: Javelin uses standard VB.net syntax. Errors messages are standard .Net messages and you can find them quickly by a web search.

Publish from Javelin as an Active Directory User

You want to publish from Javelin using an Active Directory user. If the Administrator user has created and added default jobs in Datamaker for a Data Pool, and if Datamaker, Remote Publish, and TDoD are integrated with AD, follow this procedure:

1. Join the test machine to the AD Domain in which Javelin is installed.
2. Log in to the test machine as the AD user as which you want to perform the Javelin publish.
3. Launch Javelin and connect to Datamaker using GTService or TDMService.
4. Drag and drop the data pool to the canvas, and choose the default job that was created by the Administrator user from the drop-down.
5. Create 3 variables in Javelin and define the following default values:
   - username—Defines the AD user name.
   - password—Defines the AD user password.
   - jobXml—Defines the job and its properties.
   a. a. Edit the data pool and go to the **Properties** tab.
      b. Copy the entire property value of "JobXml" and paste it as default value into the variable `jobXml` .
      c. Replace "Administrator" with "[USERNAME]" in the copied text (for example, <username>[USERNAME]</username>).
6. Edit the data pool, go to the **Properties** tab, and enable the Use Variable option.
   a. Set the Username property value to `username.`
   b. Set the Password property value to `password.`
   c. Set the JobXml property value to `jobXml.Replace("[USERNAME]", username).`
7. Save the Javelin flow and submit the publish job.

If you want to perform the publish as a different AD user, repeat these steps.

Socket Timeout Opening Firefox Driver

When I open Firefox, the following error message appears:

Failed to start up socket within 45000.

Reason:

Firefox has been upgraded and other tied-in products require updates.

Action:

1. a. Go to http://selenium-release.storage.googleapis.com/index.html
   b. Open the latest version folder (currently 2.48) and extract the file "selenium-dotnet-<version>.zip". For example, "selenium-dotnet-2.48.0.zip".
   c. Copy the WebDriver.dll and the WebDriver.Support.dll from the extract, and replace the files in the Javelin installation folder.
   d. Restart Firefox

Shredder Actions Not Supported Through CA TDM Portal

Javelin does not support using Shredder actions through CA TDM Portal.

Javelin Debugging

LogActivity Actions

The most common and quickest way to debug a Javelin flow is to use the "LogActivity" action. This approach is similar to a developer who prints a variable to the console when debugging code.

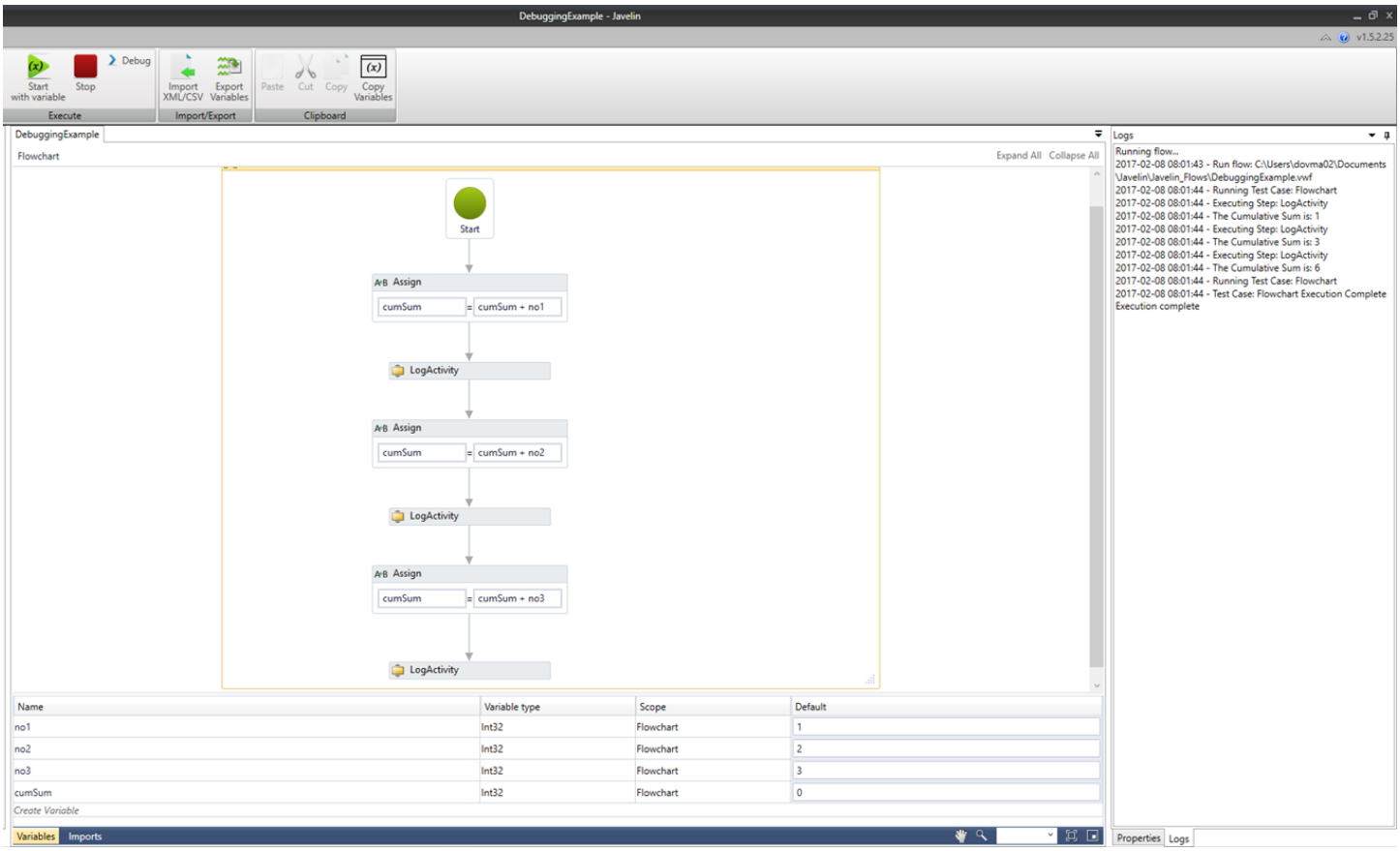
1. Place the "LogActivity" action strategically within the flow.
2. Open the Properties pane of the LogActivity and type in the MessageToLog field what you want to print to the log.
   Format: Use VB.net syntax.
   Example: Log the value of some variable in the flow.
3. Run the flow.
4. View the Logs pane.

TIP

Syntax errors: If you have entered incorrect VB.net syntax, a red exclamation mark appears on the offending action. Hover over this action and read the error message. The errors are standard .Net messages and you can find them quickly by a web search.

Example: LogActivity Actions

In this example, the purpose of the flow is to sum up the numbers 1, 2, and 3. These values are assigned to the integer variables no1, no2, and no3 respectively. The variable cumSum represents the cumulative sum of these variables. You place a log activity between each addition to track the cumulative sum. Run the flow and view the log pane. You see how the variable is printed after every addition, thus allowing you to monitor its evolution throughout the flow.



TIP

Every time Javelin runs, it stores the log from the flow as a .txt file in a directory. To inspect logs, open the path `%appdata%\Grid-Tools\Javelin\logs` in the file explorer.

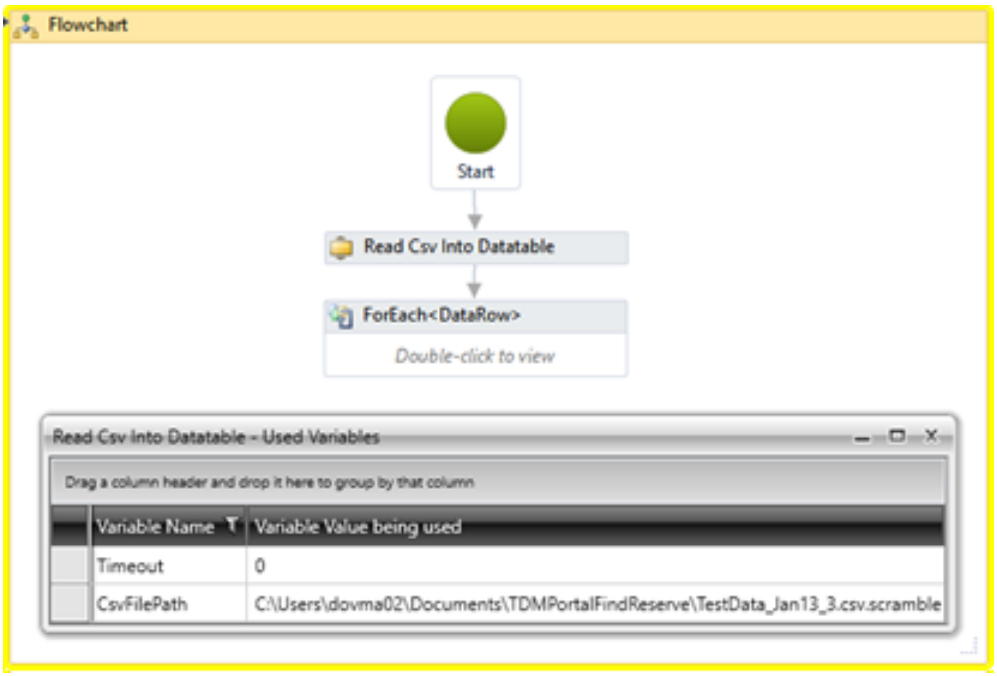
Run Javelin in Debug Mode

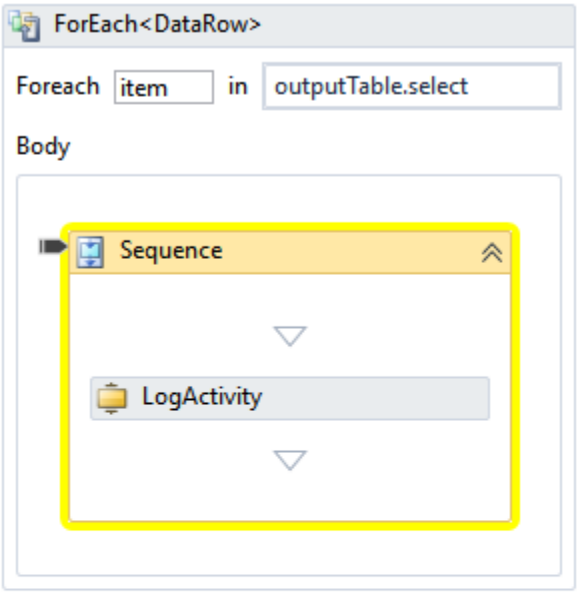
You can achieve rigorous debugging by running Javelin in debug mode. Open a flow and click the **Debug** button on the **Home** ribbon.

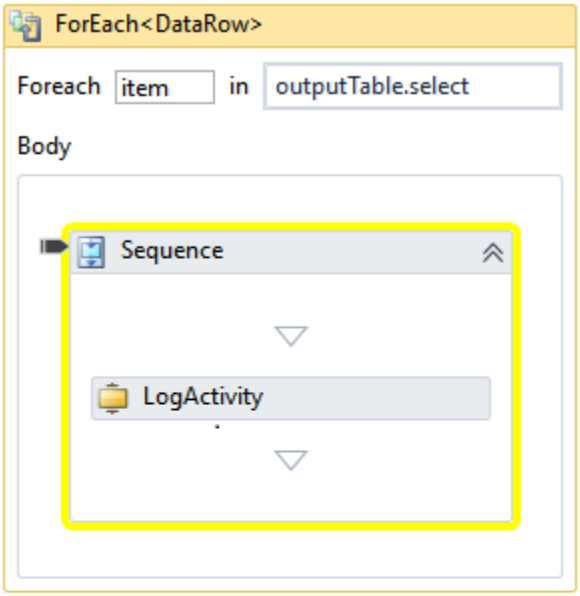
In debug mode, the flow breaks before execution of each action. For a recursive action, that is a ForEach action, the flow breaks at each iteration. When the flow is stopped, a message box appears and lists all variable names and their values that input or output from that action. When the user closes this box, the action executes, and the flow moves to the next action. Thus you can monitor the evolution of variables throughout the flow.

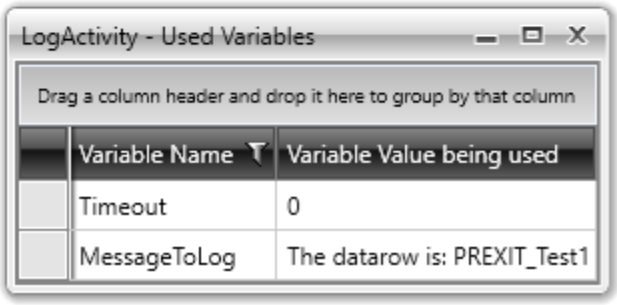
Example: Debug Mode

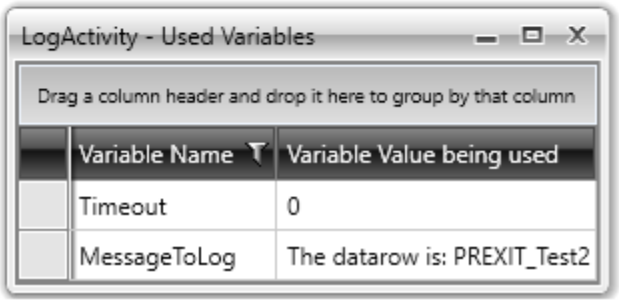
The following example flow reads a CSV file into a DataTable variable, iterates through the DataTable, and prints the first element in each row to the log. The three following screenshots show the flow and message box for three debugger breaks. The fourth screenshot is the log.











Debugging flow...
2017-03-24 08:26:48 - Run flow: C:\Users
\Documents\Javelin\Javelin_Flows
\AccessElementsOfDatarow.vwf
2017-03-24 08:26:48 - Running Test Case: Flowchart
2017-03-24 08:26:48 - Executing Step: Read Csv Into
Datatable
2017-03-24 08:26:52 - Executing Step: LogActivity
2017-03-24 08:26:54 - The datarow is: PREXIT_Test1
2017-03-24 08:26:54 - Executing Step: LogActivity
2017-03-24 08:26:56 - The datarow is: PREXIT_Test2

TIP

Sometimes the debugger Message box can hide behind the Javelin flow. Press alt-tab to find it again.

Support for DB2 on the IBM System AS-400 and z/OS

Javelin does not support DB2 drivers on IBM AS-400 and z/OS.

```
Activity could not be loaded because of errors in the Xaml
```

This error message means that the action that this error has replaced is not compatible with the current version of Javelin that you are running. To fix, change to a compatible version.

Oracle Bulk Copy Writer UserName Password Incorrect When Copying to Oracle RAC

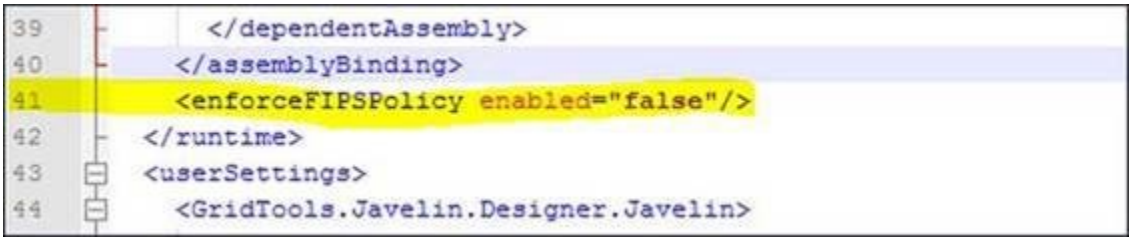
Problem:

When using the Bulk Copy Writer action copy to an Oracle RAC cluster database, Javelin outputs an incorrect Username/Password error, even if both are correct.

Solution:

if you are 100% certain that you have entered the correct Username and password, follow these steps:

1. Go to the directory C:\Program Files (x86)\Grid-Tools\Javelin\.
2. Open the file JavelinExecutor.exe.config.
3. Change the *enabled* attribute in the *enforceFIPS*Policy element. (shown in screenshot)



Javelin Cannot Trigger Mainframe Jobs from Windows

The following article describes how to submit mainframe jobs from Windows using FTP: Submitting Jobs and Retrieving the Output Via FTP by Lionel B. Dyck

More information and support:

- IBM Knowledge Center: Transferring data using the File Transfer Protocol (FTP) > Interfacing with JES
- IBM Knowledge Center: Transferring data using the File Transfer Protocol (FTP) > Interfacing with JES > Steps for submitting a job

Mainframe

This section contains information specific to use of CA Test Data Manager with Mainframe data sources.

For information about installing Test Data Manager on the mainframe, see Mainframe Installation and Upgrade.

Mainframe System Requirements

Installation

See Mainframe Installation and Upgrade

The Mainframe

Test Data Manager mainframe functionality supports masking, subsetting, and synthetic data generation in z/OS mainframe environments.

For masking, a standard set of seed data is supplied. This seed data can be stored in DB2 tables or in a VSAM KSDS, if DB2 is not available.

Operating System

In addition to installing components on z/OS, you need to install Datamaker and GT Subset on Windows in order to use Test Data Manager in mainframe environments.

Data Source Types

Test Data Manager supports subsetting and masking against DB2, VSAM, ISAM, flat files, and IMS.

For more information about prerequisite and system requirements, see System Requirements for Mainframe Installation.

Working with DB2 Data Sources

To work with DB2 Data Sources, you must first register the required tables into a project version in Datamaker. Optionally, you can then profile the data available in the registered tables before continuing to mask, subset, or generate data to those tables.

Register DB2 Tables

Test Data Manager can directly connect to a DB2 database using a connection profile. You must have DB2 Connect installed on the Test Data Manager server to establish this connection.

Once connected, you register the tables with Test Data Manager and can optionally profile the tables to detect personally identifiable information (PII).

Configure Connection Profile

Use DB2 Connect to configure a Datamaker connection profile to your z/OS DB2 datasource.

You may receive an error similar to the following:

```
odbc: -1031
SQLSTATE = 58031
```

```
[IBM] [CLI Driver] SQL1031N The database directory does not exist in the file system
 specified. SQLSTATE = 58031
```

If you do receive this error, perform the following steps in DB2 to fix the connection:

1. Eliminate the current DB2 Connect setup.
2. Start IBM DB2.
3. Access the Command Line Processor and enter the following command:

```
connect to db user userid (replace as required)
```

4. Enter the password when prompted.
   If the connection is successful, the DB2 connect setup is eliminated.

Register Tables

Register the tables that you want to mask to an appropriate project in Datamaker.

Follow these steps:

1. In the toolbar, click **Projects** and select **Register** from the drop-down list.
   The **Register into Project: File Version: <version>** screen opens.
2. Select **Database Table** from the list of options.
3. In the **Register Tables** window, use CTRL + Click to select the tables you want to register.

Profile Table Contents

Profile the contents of the selected tables.

Follow these steps:

1. In the tool bar, click **Data Profiler** and select **Sample Table Data** from the drop-down list.
2. Click and highlight the tables you want and **select Sample Data** from the drop-down list.
3. (Optional) If required, connect to the source or target database.
4. Select your required options in the **Sample Options** window, and click **OK** to sample the selected tables.

Masking DB2 Data Sources

You can use TDM to mask DB2 data sources. To do so, it is necessary to:

- Create transformation maps in Datamaker.
  Transformation maps define the masking rules to use on each table.
- Export the transformation maps to a file that you can transfer to the mainframe.
- Perform the masking job on the mainframe using batch jobs.

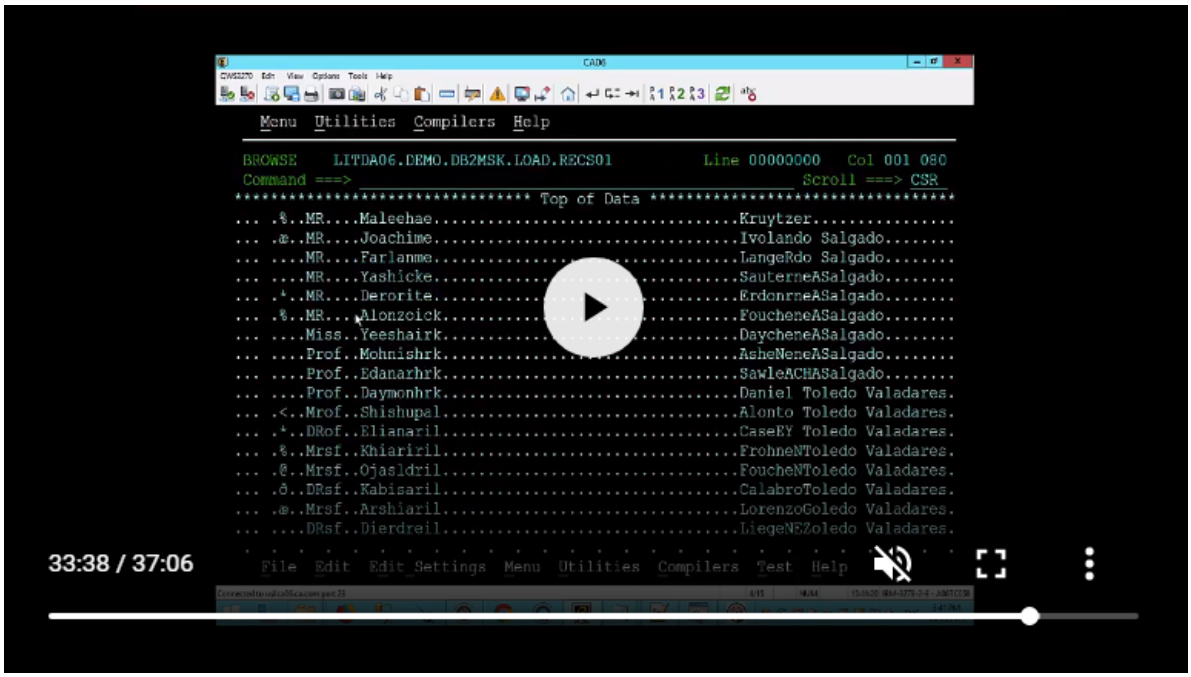
You can also mask mainframe data in-flight, i.e. store the masked data in memory and write to a new location. For more information, see Masking DB2 data sources in Mainframe z/OS.

Masking DB2 data sources in Mainframe z/OS

As a Tester, you need to mask a set of columns in DB2 for z/OS tables, and generate the necessary masking and sub-setting rules for these DB2 for z/OS tables. You want to mask and subset different sets of datasets that have been

exported from the production DB2 for z/OS subsystems based on the rules that the test data engineer has defined. This scenario shows how to set up and generate masking and sub-setting rules, and execute the mainframe processes.

CA Communities video: TDM z/OS: In-Flight Data Masking (and Subset) for DB2 z/OS



Requirements

You need to fulfill the following requirements before you can use the CA TDM Mainframe toolkit:

- CA TDM 4.x installed
- Mainframe user access
  - TSO access
  - ISPF editor access
  - DB2 instance access
- Mainframe user needs to have FTP capabilities
- QWS3270 or equivalent installed
- DB2 Connect v10.x or better (or DB2 standard with DB2 Connect feature)

Install CA TDM Mainframe Support

The CA Test Data Manager for Mainframe package is composed of CA TDM mainframe objects (PGMs and JCL procs). These mainframe binaries are needed to perform in-place or in-flight masking natively in the mainframe.

1. Log into the CA Support site https://support.ca.com.
2. Click **Download Management**, enter "CA Test Data Manager", and browse the **Product Downloads**.
3. Download the following (or a newer) package:
   CA Test Data Manager Mainframe DB2 Add On MVS, Version 5.4.13, Service Pack 7
   The **Download Manager** opens.
4. Download the file **CA Test Data Manager For Mainframe 5.4.13** using your preferred method.
   The file is saved under a numeric name, for example, GEN500000000001207.zip.

For details how to install the mainframe toolkit, see Mainframe Installation and Upgrade.

Mainframe Data Masking

Data masking hides or obfuscates sensitive and classified data. The goal is to protect data that is used for purposes such as development, testing, and QA cycles. Data masking is a standard practice that is often required for compliance with national and international data protection legislation.

To perform the necessary data masking natively in the mainframe, you use Datamaker transformation maps to mask the data. The approach that you select depends on your business requirements and feasibility. You can adopt one of the following approaches to masking, depending on which stage the data is masked at:

- **In-place masking**
  A typical scenario for in-place masking is that the production data is copied over to a staging area. You use DataMaker to create a transformation map with the necessary rules, upload this transformation map, and use the RUNJCL(GTXMSK) JCL procedure pointing to this staging database, and it masks the data that resides there. You then copy this masked data over to different testing environments as required.
  For more information, see Mask DB2 Tables in Place.
- **In-flight masking**
  In a typical scenario for in-flight masking, you use Datamaker transformation maps and Subset scripts. You first define a transformation map (Oracle or MSSQL) in Datamaker, and create masking functions for the columns you want to mask. You use the Subset interface to create the masked export scripts. These scripts perform masking as they export the source data to a dump file. The dump file (which contains masked data) is then imported into the target database. Testers can use the same database, which now includes masked data, for testing.
  For more information, see Mask and Unload DB2 Tables.

The mainframe data masking facilities are designed to help you with the masking of DB2 datasets natively in the mainframe environment. These facilities provide you with consistent, robust, and repeatable methodologies for securing sensitive data.

The following table shows the more common mainframe programs that you use for in-place and in-flight masking. Before you transfer the XMI files, pre-allocate these files in the mainframe based off the following values, and define them as partitioned dataset files (PDS).

Program Name	JCL Proc	Purpose	Link to Flow Diagram
GTXMSK	RUNJCL(GTXMSK)	The program that performs the in-place masking	GTXMSK Flow Diagram
GTXMSKL	RUNJCL(GTXMSKL)	The program that performs the in-flight masking	GTXMSKL Flow Diagram
GTXMSKF	GRIDT01.LOADLIB	The program that performs the in-place masking of a flat file	GTXMSKF Flow Diagram

These programs require the uploaded members in the following datasets:

Dataset	Member	Purpose
LIB.MAPCSV	Transformation_map_name	This dataset contains the transformation map rules that are used for the in-place and in-flight masking.
LIB.SUBS	Subset name	This file contains the subset members that you generated and that are used for the in-flight masking effort.

In-Flight Masking Scenario

In the following in-flight masking scenario, you create a transformation map, set up a subset job, and attach the subset job before you generate the transformation map. These files are uploaded to the mainframe and placed in the two datasets listed above.

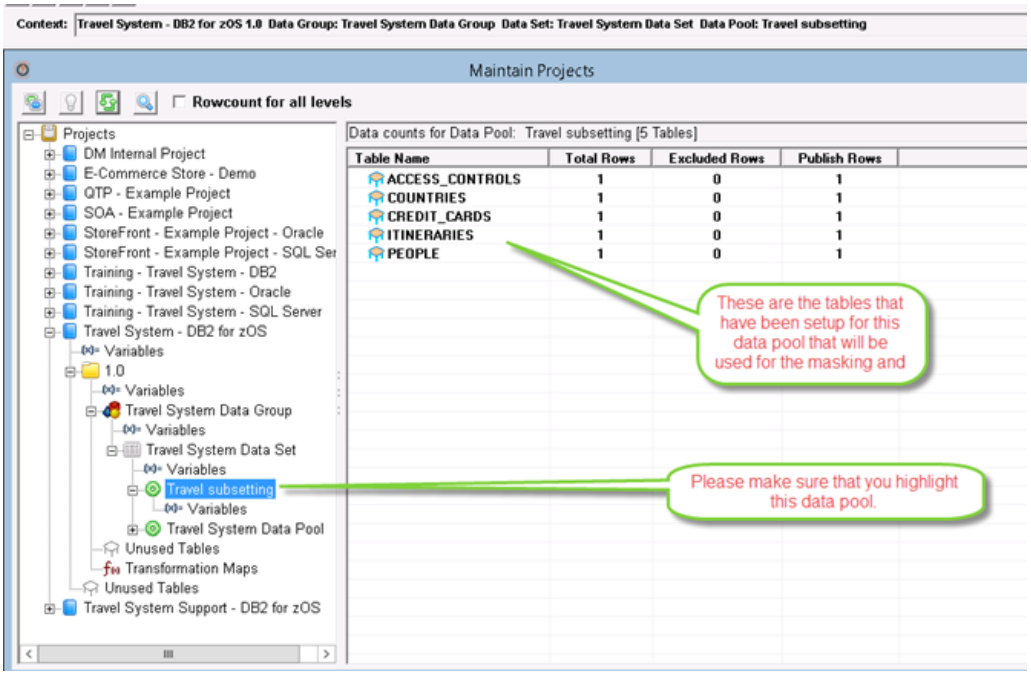
TIP

Use the JCL proc TDMDBLD to load the masked data back into another DB2 subsystem. See Appendix A for details.

Create Transformation Map and Subset

1. Launch GT Datamaker. Click the **Maintain Projects** button.

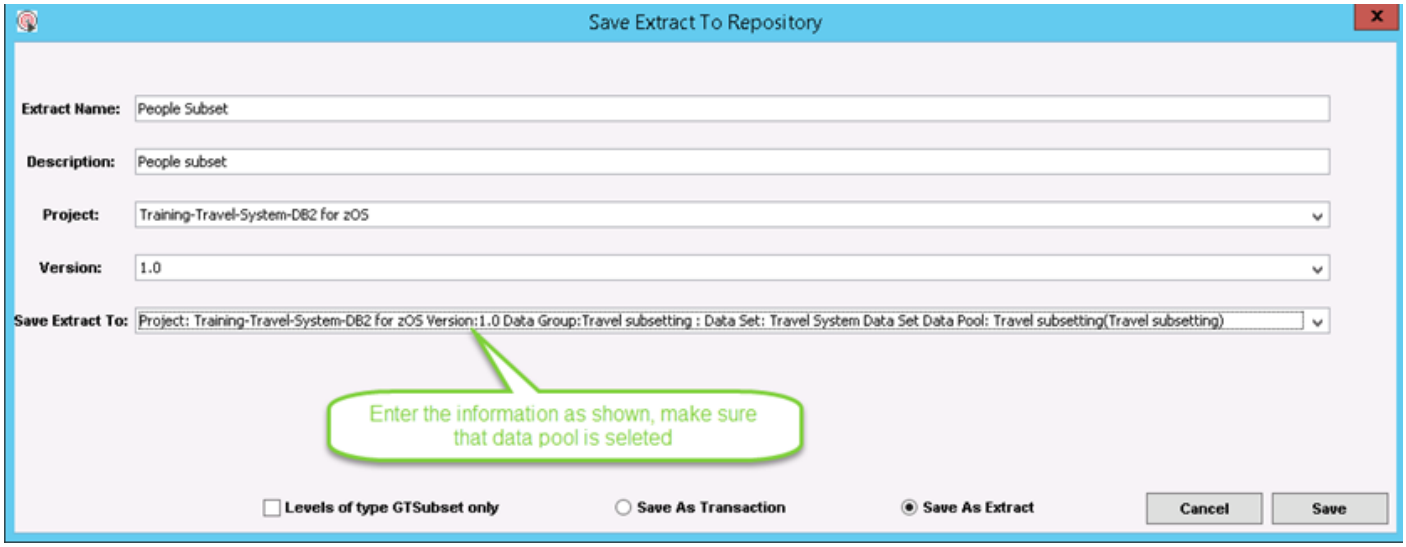
   You will see the available projects in the **Maintain Projects** window.
2. Pay close attention to the DB2 for z/OS project. There are two schemas in this scenario:
   – The source schema is pointed to GRIDDEMO.
   – The target schema, which will be very relevant to us is pointing to the TRAVELDEV schema.
   – The target tables are already present in the target schema.



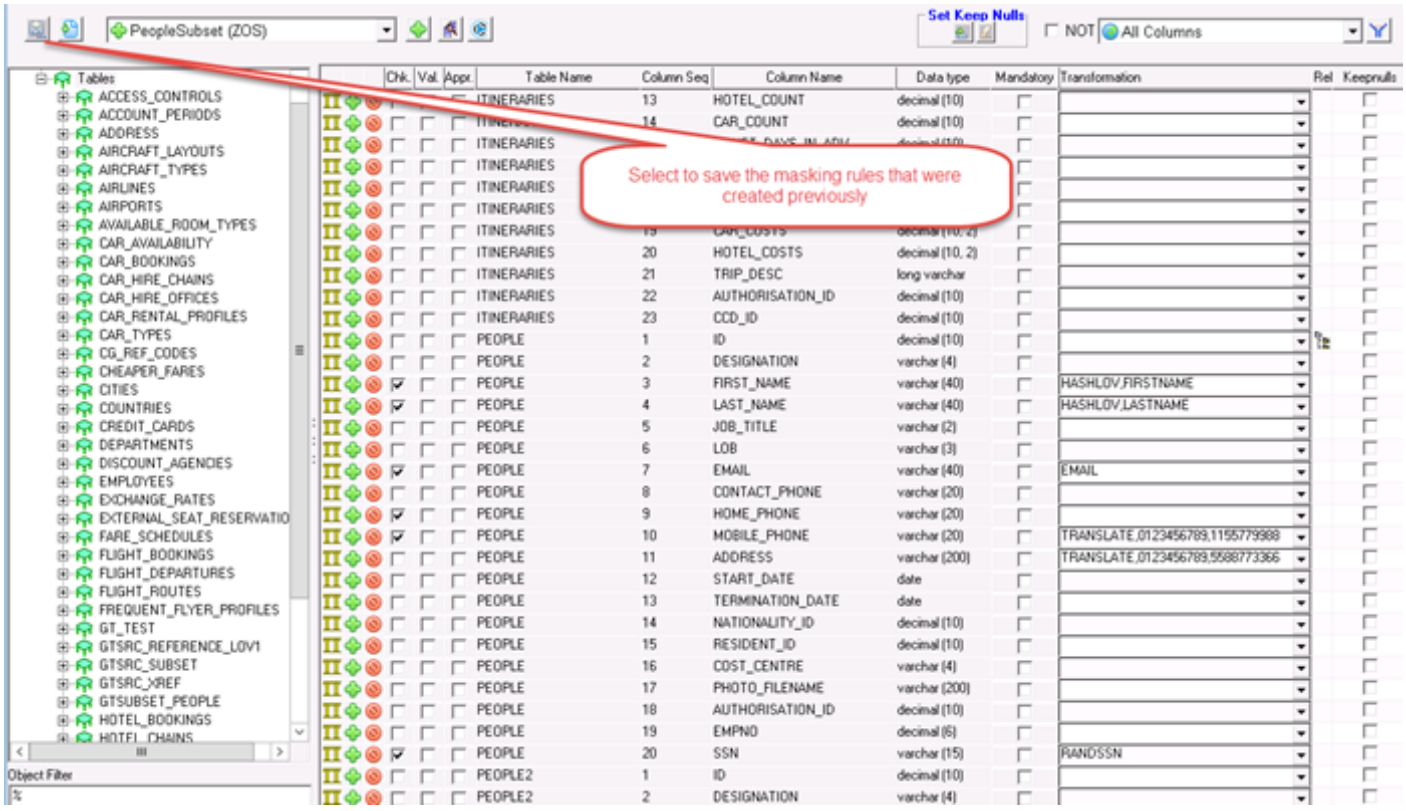
3. Click **Data Subset**, **Design Extracts and Transactions** from the menu to create the subset that is used as part of this use case.
   GT Subset opens.
4. Select Schema: TRAVELDEV
5. Select Table: PEOPLE
6. In the SQL Window, create a subset that is used in the same schema that provided the data, which is to be subset and masked.
   Complete the SQL statement:

```
SELECT * FROM TRAVELDEV.PEOPLE where traveldev.people.cost_centre = 'AAAA'
```

7. Click the **Save Extract to the TDM Repository** button.
   The **Save Extract to the TDM Repository** window opens.
8. Save the "PeopleSubset" subset to the following data pool:



9. Return to Data Maker and refresh the project tree.
   You see that the subset extract "PeopleSubset" has been placed under the "Travel subsetting" node.
10. Click the Transformation Maps node in the project tree.
11. Double-click the "PeopleSubset" transformation map in the right pane to select it.
12. Create a set of masking rules that is to be converged with the subset extract that you created previously.



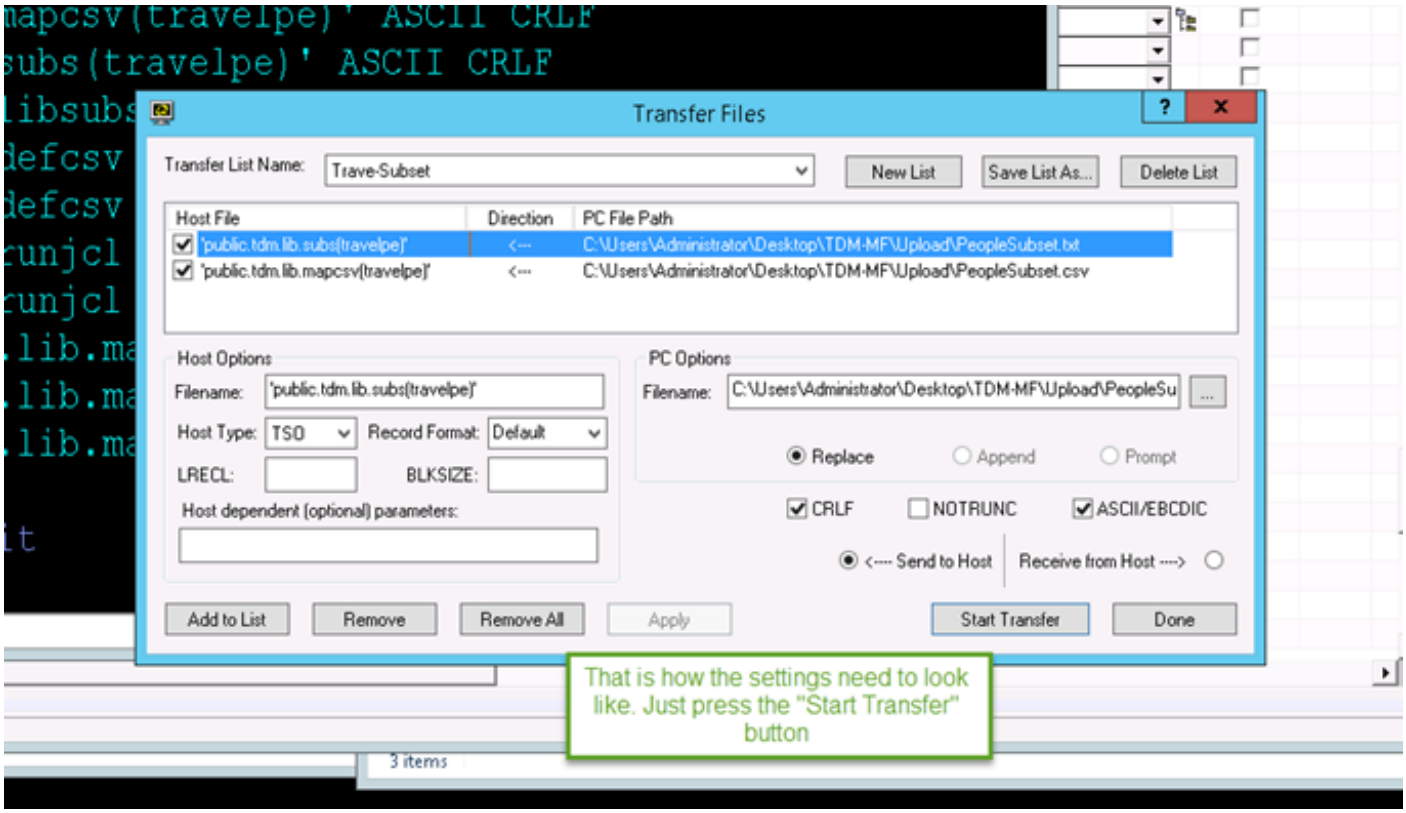
13. Scroll down until you see the "People" table, and the fields that have been selected to be masked.

14. Click **Save** on the left side of the dialog.
    a. Save the masking rules as the "CSV-ZOS" file type.
    b. Save the file as PeopleMask.csv, for example to your desktop, and remember the path. This is important, since this is the location from which you will be uploading the files to the mainframe.
    c. Select "Yes" when prompted if you would like to attach a subset to the transformation map.
    d. Select the subset extract that you created previously and click OK.
    e. Select "JOIN SQL" as the subset condition when prompted. Click the green checkmark.

    A dialog displays success and a file path.
15. Open the indicated directory and verify that it contains the following files:
    – PeopleMask.csv
    – PeopleSubset.csv
    – PeopleSubset.txt

Transfer the Files

1. Open QWS3270 and connect to the mainframe system using your mainframe credentials.
2. Select the option to enter the **TSO** environment, and enter **ISPF** to start at the base menu.
3. Select option 6 to enter the **TSO Command Line**.
4. Click **Tools, Batch Transfer Files** in the QWS3270 menu.
   The **Transfer Files** window opens.
5. Start the batch job that has been preset already. Press the "**Start Transfer**" button.



6. Wait for the transfers to completed, and press Done to close the batch transfer dialog.

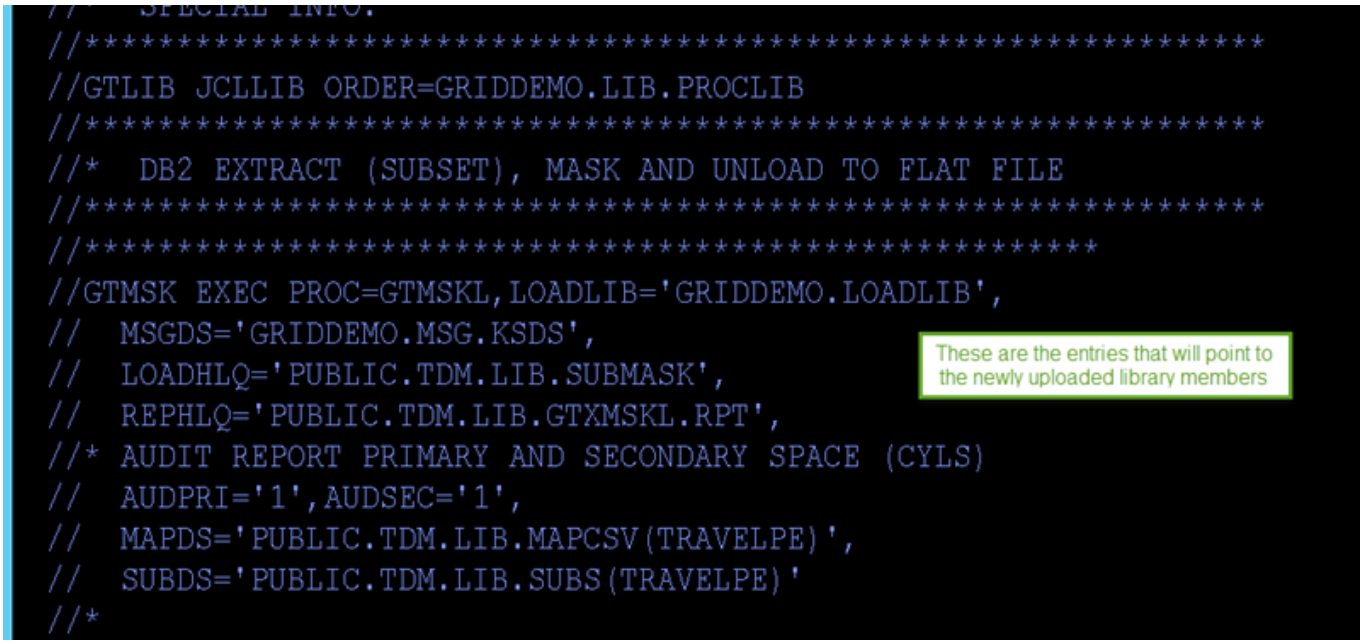
Review the Files

1. Press PF3 to return to the main menu.

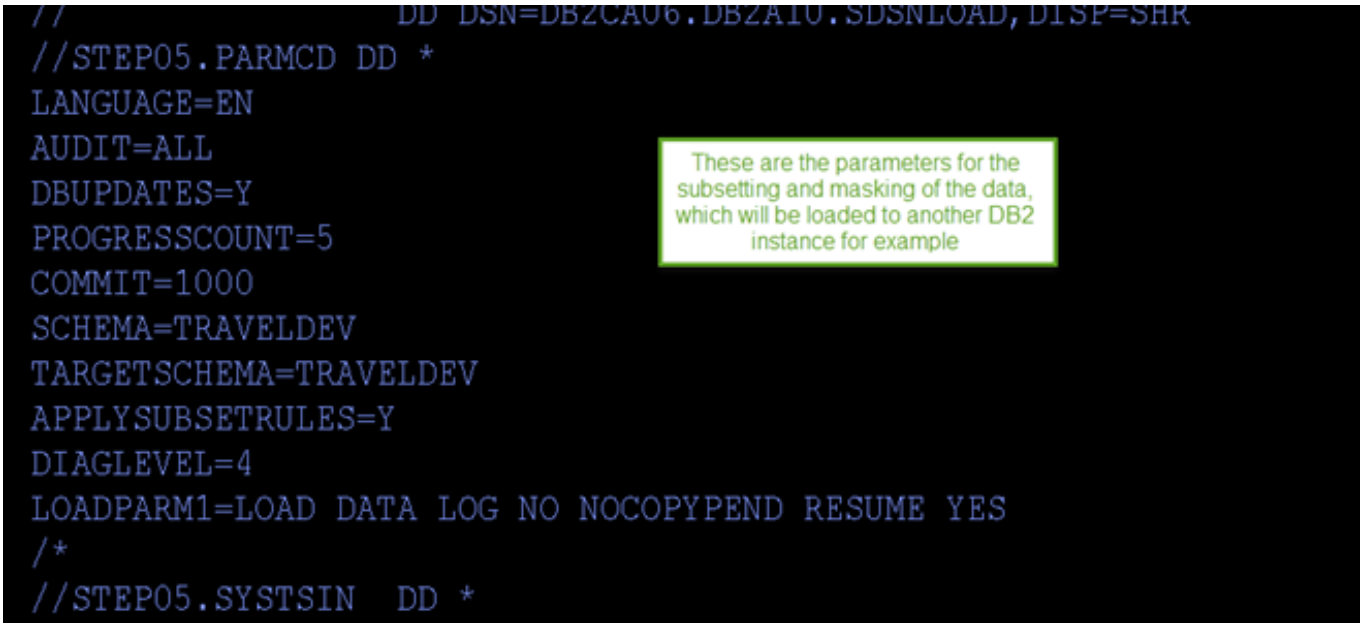
2. Select option 3.4 to display the data sets.
3. The dsnname level that you start at is "PUBLIC.TDM.*" This is just an example, your entries could be starting with "GRIDT01.LIB".
4. After you have the list of datasets display, browse the dataset library PUBLIC.TDM.LIB.MAPCSV. Look at the masking file contents for the file that we just uploaded.
5. Inside PUBLIC.TDM.LIB.MAPCSV, look for "TRAVELPE", and browse the contents of this file.
6. After looking at the contents of the masking file, press PF3 twice to return to the dataset listing.
7. Browse the PUBLIC.TDM.LIB.SUBS for the subset rules that you uploaded.
8. Inside the PUBLIC.TDM.LIB.SUBS dataset, browse the TRAVELPE member, which contains all the subset rules.
9. After reviewing these subset rules, press PF3 twice to return to the dataset listing.

Update the Procedure and Submit the Job

1. At the dataset listing, browse the PUBLIC.TDM.LIB.RUNJCL dataset library.
2. Browse the contents of the "GTXMSKL" JCL procedure. This procedure performs the unloading of the data based off the subset rules, and masks the resulting data files.
   a. Update the proc with the map and sub member names.
   b. Update the "LOADHLQ" and "REPHLQ" entries.



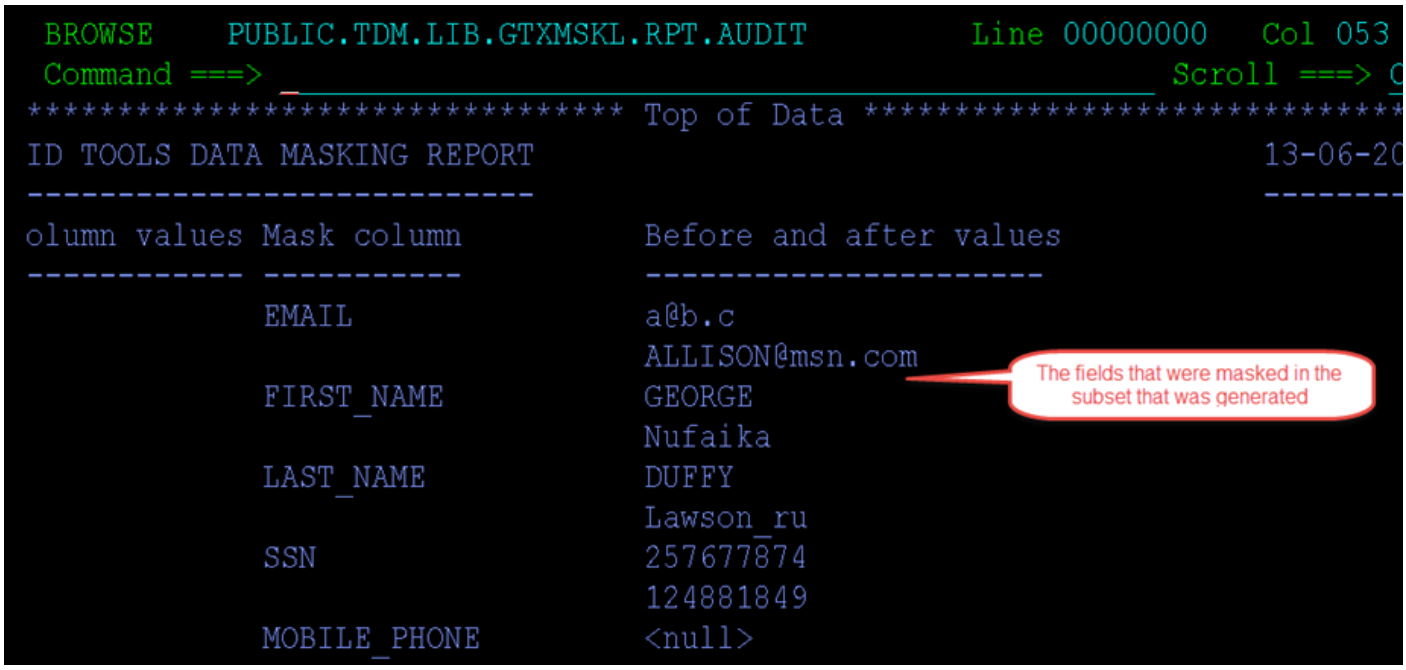
   c. Update the "schema" and "targetschema".



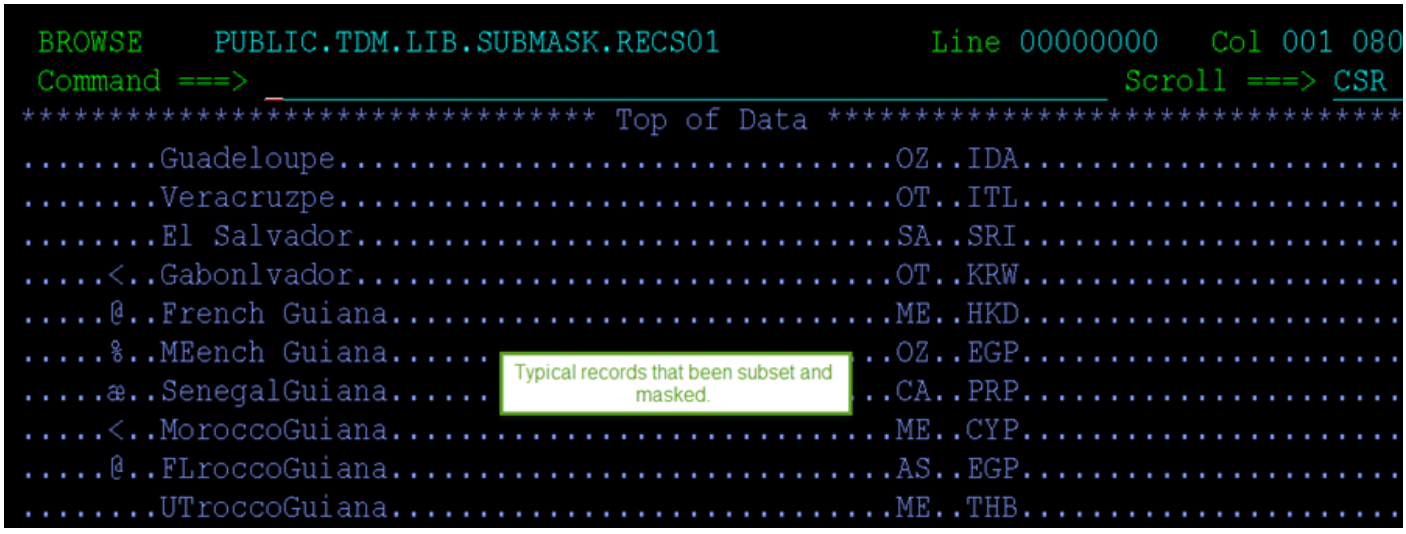
3. Enter "Submit" at the command line in the JCL procedure.
   You get a message that the JCL procedure has been submitted to the JOB subsystem.
4. Wait about 45 seconds and the following files appear:
   - Load data card PUBLIC.TDM.LIB.SUBMASK.CARDS
   - Data files PUBLIC.TDM.LIB.SUBMASK.RECS01, 02, 03...
   - Report file PUBLIC.TDM.LIB.GTXMSKL.RPT.REPT
   - Audit file PUBLIC.TDM.LIB.GTXMSKL.RPT.AUDIT

Verify the Output

1. Browse the REPT file to get a status report of the unloading and masking of the data.
2. Press PF3 once to return to the main dataset listing.
3. Look at the AUDIT report, which shows you the fields that were masked with their original and masked values.



4. Press PF3 to return to the dataset listings.
5. Browse the load data card, so you can see how the load command with the supporting tables is organized.
6. Press PF3 to return to the dataset listing.
7. Browse one of the data files that was generated. These are rows that will be loaded back into the target database schema that was pre-defined in the GTXMKSL procedure.



8. Press PF3 to return to the dataset listing.

Load Data

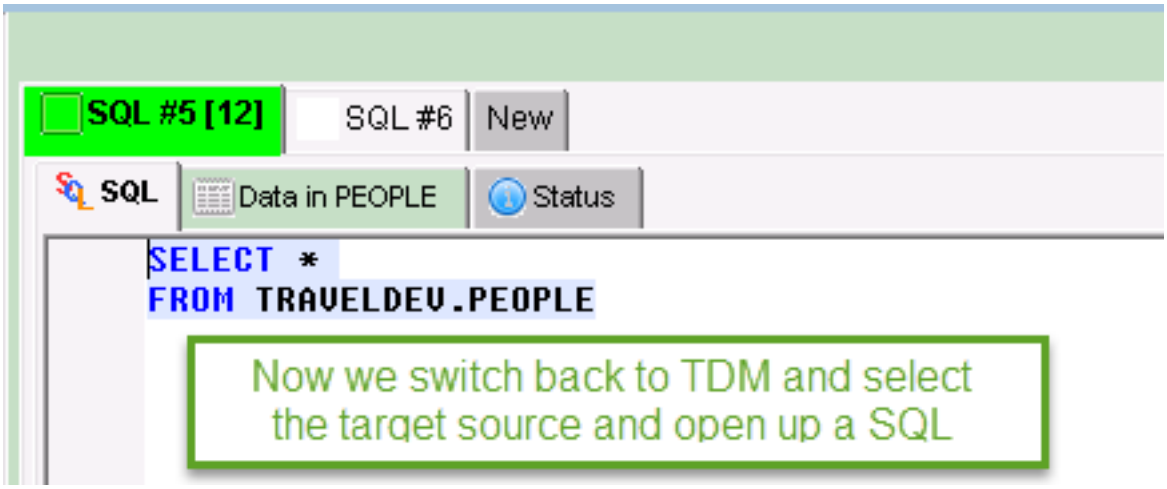
1. Browse the PUBLIC.TDM.LIB.RUNJCL dataset library again.
2. Browse the TDMLODDB member. This JCL procedure contains the necessary instructions to load data into the target schema based off the previously generated load card and data files.
3. Update the selected JCL proc copy and update the following entries:

- DB2 subsystem info
- SYSIN
- SYSREC01…SYSREC15

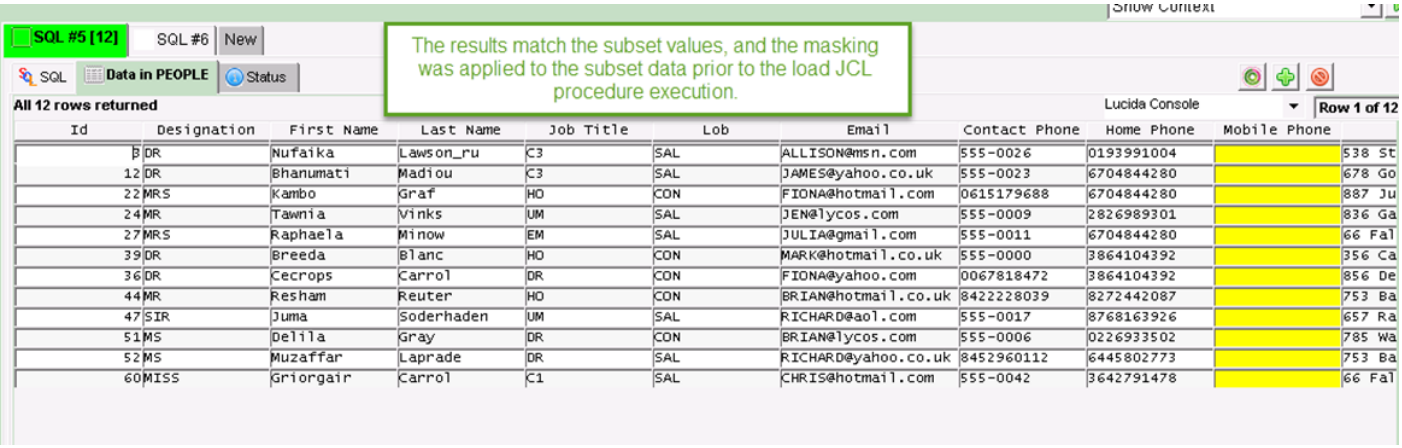
```
//* DB2 LOAD OF MASKED FILES
//*********************************************************************
//* ------------------------------------------*
//LOAD     EXEC DSNUPROC,SYSTEM='C10V',COND=(4,LT)
//STEPLIB  DD DSN=C10V.PRIVATE.SDSNEXIT,DISP=SHR
//         DD DSN=C10V.RUNLIB.LOAD,DISP=SHR
//         DD DSN=DB2CA06.DB2A10.SDSNLOAD,DISP=SHR
//SYSOUT   DD SYSOUT=*
//SYSIN    DD DSN=PUBLIC.TDM.LIB.SUBMASK.CARDS,DISP=OLD
//SYSREC01 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS01,DISP=OLD
//SYSREC02 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS02,DISP=OLD
```

These are the parameters that will be executed as part of the DB2 DSNUPROC taking into account the generated load card and data files

4. Enter the submit command and press Enter.
   You receive a message that the job has been submitted. If the message that you receive has a MAXCC value of 04 or less, then the job has completed successfully.
5. Go back to GT Data Maker to execute the SQL statement.



6. Verify that the values that you selected have been masked as necessary.



Id	Designation	First Name	Last Name	Job Title	Lob	Email	Contact Phone	Home Phone	Mobile Phone	
3	DR	Nufaika	Lawson_ru	C3	SAL	ALLISON@msn.com	555-0026	0193991004		538 St
12	DR	Bhanumati	Madiou	C3	SAL	JAMES@yahoo.co.uk	555-0023	6704844280		678 Go
22	MRS	Kambo	Graf	HO	CON	FIONA@hotmail.com	0615179688	6704844280		887 Ju
24	MR	Tawnia	Vinks	UM	SAL	JEN@lycos.com	555-0009	2826989301		836 Ga
27	MRS	Raphaela	Minow	EM	SAL	JULIA@gmail.com	555-0011	6704844280		66 Fal
39	DR	Breeda	Blanc	HO	CON	MARK@hotmail.co.uk	555-0000	3864104392		356 Ca
36	DR	Cecrops	Carrol	DR	CON	FIONA@yahoo.com	0067818472	3864104392		856 De
44	MR	Resham	Reuter	HO	CON	BRIAN@hotmail.co.uk	8422228039	8272442087		753 Ba
47	SIR	Juma	Soderhaden	UM	SAL	RICHARD@aol.com	555-0017	8768163926		657 Ra
51	MS	Delila	Gray	DR	CON	BRIAN@lycos.com	555-0006	0226933502		785 Wa
52	MS	Muzaffar	Laprade	DR	SAL	RICHARD@yahoo.co.uk	8452960112	6445802773		753 Ba
60	MISS	Griorgair	Carrol	C1	SAL	CHRIS@hotmail.com	555-0042	3642791478		66 Fal

Best Practices

The following best practices help you in being successful in masking DB2 datasets.

DB2 Authorizations

Make sure that you have sufficient rights to the DB2 schemas (read/write/alter authorizations). At the same time make sure that you have set up DB2 Connect, and tested this connection from the system where CA TDM is installed.

Add an ODBC entry to CA TDM that points to the DB2 subsystem in the mainframe. For more information, see System Requirements for Mainframe Installation.

Planning

Before you start your in-flight masking, plan the process:

- Select the proper entries in the transformation maps.
- Make sure that you have tested your subsets in GT Subset, which you can access from the start menu or via GT DataMaker.
- Verify that you have proper access to the mainframe with the proper datasets authorizations as described in the requirements section.

JCL Procedures

Create a copy of the GTXMSKL procedure for a specific subset/transformation map.

Main options

The following two options are key, if you want to just mask the data in-flight with no subset. Or you could just subset the data with no masking, if you so choose.

- No subset – You need to change the following entry: "`SUBDS=NULLFILE` "
- No masking – You need to change the following entry: "`MAPDS=NULLFILE` "

Report and Audit files

To make sure that you differentiate the in-flight masking job, it is important that you change the following entry to the type of masking that you are doing.

- Report path – Change the following entry "`REPHLQ=GRIDT01.LIB.GTXMSKL` ".
  For example, set that value to "`REPHLQ=mypath.lib.inflgt.rpt` ".
- Size of files – If you are processing a lot of fields and a lot of data, the size of the files is controlled via the following line in the JCL procedure:
  ```
  //* AUDIT REPORT PRIMARY AND SECONDARY SPACE (CYLS)
  // AUDPRI='1',AUDSEC='1',
  ```
  Change the value from 1 to at least 10 to provide you with the necessary space for the entries in the report and audit files.

Default Parameters

The following "shipped" parameters are included in the GTXMSKL JCL procedure.

- `LANGUAGE=EN`
- `AUDIT=ALL`
- `DBUPDATES=Y`

Set this to N initially, so that you can test how the job executes without changing the database. Set this option to Y to run the job and actually update the database.

- `PROGRESSCOUNT=5`
- `COMMIT=1000`
- `SCHEMA=source schema`
  Defines the source schema that provides the data to be masked.
- `TARGETSCHEMA=` *target schema*Defines the target schema that is scheduled to receive the masked or subset schema.
- `APPLYSUBSETRULES=Y`
- `LOADPARM1=LOAD DATA LOG NO NOCOPYPEND RESUME YES`
  Specifies the instruction set that is part of the job card that you create. Please review the DB2 load parameters, just in case you need to change these entries.

For more detailed information about all the valid parameters, see GTXMSKL Parameters

To obtain additional diagnosis messages when the job executes, you can change the entry below to the value shown, be default this value is set to 1.

```
DIAGLEVEL=4
```

Recommended Parameters

Define these additional parameters and use them in the JCL procedure job.

- `HASHTYPE=JAVA`
  Specifies that the masking hash used by the ZOS is the same as FDM. This is a requirement for consistent masking with FDM.
- `LOADPARM1=LOAD DATA LOG NO NOCOPYPEND REPLACE`
  By using these values for the creation of the DB2 load card, it replaces all the existing data in the target schema. This assumes that the DDL between source and target schemas are the same.
- `PAGELIMIT=200`
  If you are masking very large datasets, then it is important that you change the report page limit from 50 to at least 200.

How to Handle large datasets

To perform in-flight masking on more than 20 tables at a time, modify the "`GRIDT01.LIB.PROCLIB(GTMSKL )`" template procedure to be able to handle these large datasets.

Make the following changes to the template:

Update the delete section:

Add additional entries based off the entry below. Replace the DD20 with DD21, and so on.

```
//DD20 DD DSN=&LOADHLQ..RECS20,
// DISP=(MOD,DELETE),SPACE=(TRK,0),
// MGMTCLAS=TSO,STORCLAS=TSO
```

Update the create file section:

Add additional files based off the snippet below. Make a copy starting from DD20, and rename the new section DD21, and so on.

```
//DD20 DD DSN=&LOADHLQ..RECS20,
// UNIT=SYSDA,DISP=(NEW,CATLG,CATLG),
// SPACE=(CYL,(10,10)),
// DCB=(RECFM=VB,LRECL=31996,BLKSIZE=32000),
```

```
// MGMTCLAS=TSO,STORCLAS=TSO
```

Update storage capacity and location:

1. Update the `MGMTCLAS` and `STORCLAS` with the correct volume to use and with enough storage space available.
2. Modify the entry "`SPACE=(CYL,(10,10))` ". Change the values of 10 to at least 100 to make sure that there is enough space for the each of the sequential files, possibly even larger, as needed.
3. Add the same files in the "TDMLODDB" to be able to load the data back into the new DB2 subsystem.

Running Multiple JCL Jobs

We recommend submitting one JCL job for each set of tables or subsets, based on your masking needs. If you need to generate several sets of masking jobs, we recommended creating multiple copies of the JCL procedure and running these JCL procedures in parallel.

Appendix A

The JCL proc TDMDBLD loads the masked data back into another DB2 subsystem. Place the JCL procedure in the RUNJCL dataset. Update the job card info and the DB2 subsystem info.

```
//DB2LODDB JOB (002200000),'DATAMAKER',                                 00001007
//        CLASS=K,MSGCLASS=X,NOTIFY=&SYSUID                             00002008
/*JOBPARM S=CA06                                                        00003000
//*                                                                     00004000
//GTLIB JCLLIB ORDER=PUBLIC.TDM.LIBPROC                                 00005005
//*************************************************************        00150000
//* DB2 LOAD OF MASKED FILES                                            00160000
//*************************************************************        00170000
//* ----------------------------------------*                           00180000
//LOAD     EXEC DSNUPROC,SYSTEM='C10V',COND=(4,LT)                      00190000
//STEPLIB  DD DSN=C10V.PRIVATE.SDSNEXIT,DISP=SHR                        00200000
//         DD DSN=C10V.RUNLIB.LOAD,DISP=SHR                             00210000
//         DD DSN=DB2CA06.DB2A10.SDSNLOAD,DISP=SHR                      00220000
//SYSOUT   DD SYSOUT=*                                                  00230000
//SYSIN    DD DSN=PUBLIC.TDM.LIB.SUBMASK.CARDS,DISP=OLD                 00231005
//SYSREC01 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS01,DISP=OLD                00232005
//SYSREC02 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS02,DISP=OLD                00233005
//SYSREC03 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS03,DISP=OLD                00234005
//SYSREC04 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS04,DISP=OLD                00235005
//SYSREC05 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS05,DISP=OLD                00236005
//SYSREC06 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS06,DISP=OLD                00237005
//SYSREC07 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS07,DISP=OLD                00238005
//SYSREC08 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS08,DISP=OLD                00239005
//SYSREC09 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS09,DISP=OLD                00239105
//SYSREC10 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS10,DISP=OLD                00239205
//SYSREC11 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS11,DISP=OLD                00239305
//SYSREC12 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS12,DISP=OLD                00239405
//SYSREC13 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS13,DISP=OLD                00239505
//SYSREC14 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS14,DISP=OLD                00239605
//SYSREC15 DD DSN=PUBLIC.TDM.LIB.SUBMASK.RECS15,DISP=OLD                00239705
//SYSTSPRT DD SYSOUT=*                                                  00450000
//SYSPRINT DD SYSOUT=*                                                  00460000
//SYSUT1 DD DSN=&&SYSUT1,                                               00470000
// DISP=(,PASS),                                                        00480000
```

```
// SPACE=(4096,(20,20),,,ROUND)                                  00490000
//SORTOUT DD DSN=&&SORTO,                                        00500000
// DISP=(,PASS),UNIT=SYSDA,                                      00510000
// SPACE=(4096,(20,20),,,ROUND)                                  00520000
//SYSMAP DD DSN=&&SYSMA,                                         00530000
// DISP=(,PASS),UNIT=SYSDA,                                      00540000
// SPACE=(4096,(20,20),,,ROUND)                                  00550000
```

NOTE

For more information, see:

- Mask and Unload DB2 Tables
- Creating Extract Definitions for DB2 Subset
- Executing DB2 Subsetting
- DB2 Subsetting With Masking
- GTXMSKL Parameters

Create Transformation Maps for DB2 Masking

To mask a DB2 source, you first create a transformation map with masking rules, and transfer it to the mainframe.

1. Select **Projects** in the toolbar and select **Transformation Maps** in the drop-down list.
2. Click the green plus icon to add a new Transformation Map, or select existing maps from the drop-down list
   The **Transformation Maps** window opens.
3. Click the green plus icon on the right to insert a new row in which to enter the new map.
4. Click **DBMS** and select **ZOS** from the drop-down list.
5. Enter a name and description for the Transformation Map.
6. Click the **save** icon and click the green check mark icon to exit.
7. Select the drop-down and enter the rules into the **Transformation** column to create your masking rules. For more information about masking functions, see Masking Functions for Mainframe.
   Note: You can use the profiling data to help design your masking rules.
8. Click the **column** icon, to view the profile results.
9. Select a filter category from the drop-down list in the top right. Then click the **filter** icon to filter the columns in the Transformations Map screen.
   You have defined all your masking rules.
10. Click on the **save** icon to save the rules to file. Select CSV ZOS as the file type.
11. If a DB2 Subset definition is available in the repository for that project or version, Datamaker prompts you to extract that subset now.
    Choose one of the following Subset Conditions:
    - (for Mainframe) Inner Join Select -- INNER JOIN
    - Exists Select -- WHERE EXISTS SELECT
    - Join SQL

The extracted transformation map is saved as a .csv file and the saved subset as a .txt file.

Transfer these files to the mainframe using FTP or any similar utility program.

- Store the transformation map in a PDS with an LRECL of at least 255 characters.
- Save the subset .txt file to a dataset with RECFM=VB and with a large LRECL. The exact required record length depends on the complexity of the subsetting rules being used.

Executing Masking (DB2 Data Sources)

Once appropriate masking rules have been defined in a transformation map they can be executed either by masking tables in place, or by masking the data and writing out the results to files to be loaded back into a DB2 instance. In this case, the source tables are unchanged by the masking.

Mask DB2 Tables in Place

To run DB2 in-place file masking, JCL is supplied in the installation package as GTXMSK. This job uses procedure GTMSKDB.

You can use the following parameters for the JCL procedure:

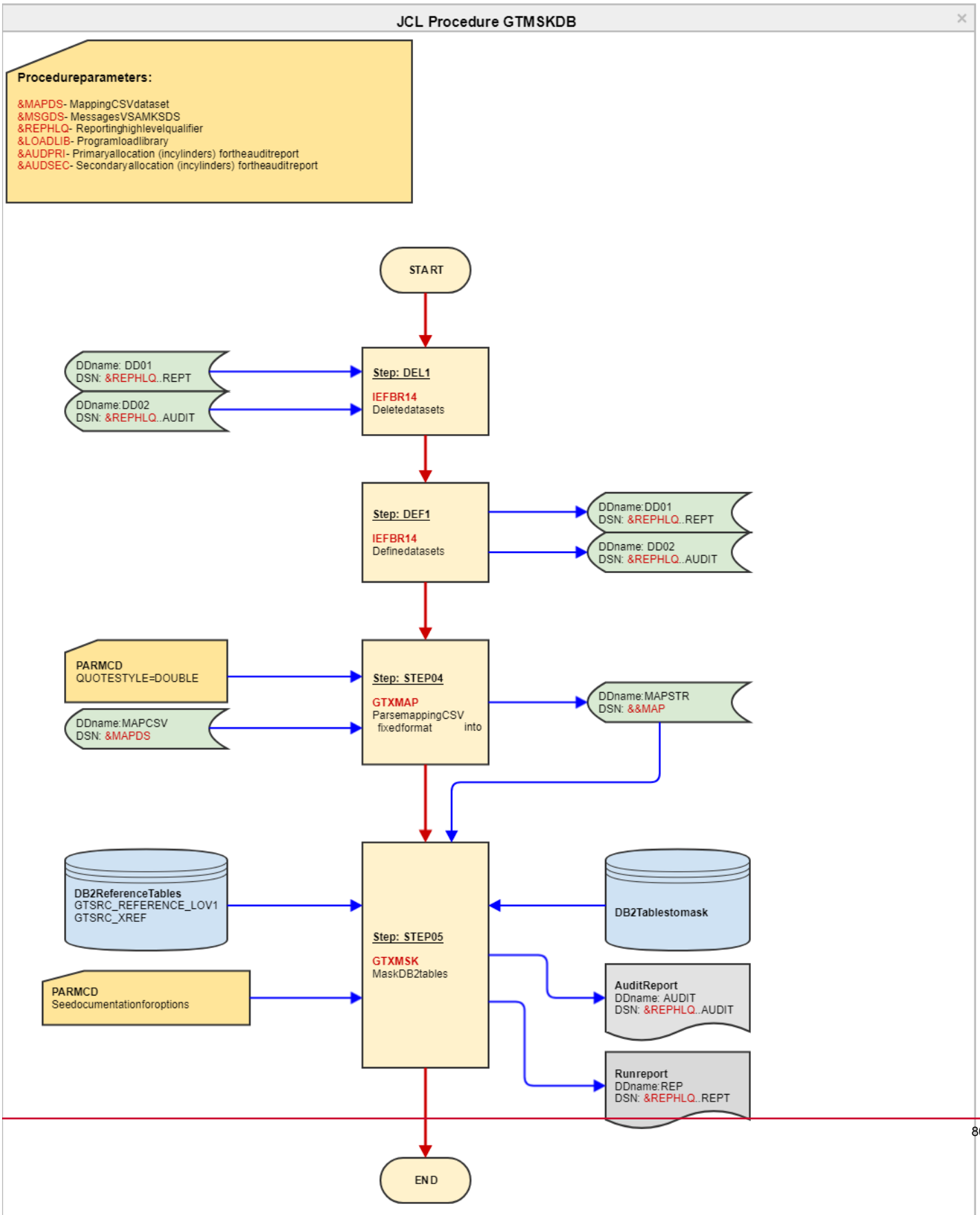
- **LOADLIB**
  Names the load library that contains programs GTXMAP and GTXMSK.
- **MSGDS**
  Names the VSAM data set containing the TDM error messages.
- **REPHLQ**
  Gives the high-level dataset name qualifier that is used for the audit and report files.
- **MAPDS**
  Names the dataset that contains the mapping CSV (masking rules).
- **AUDPRI**
  The primary space allocation (in cylinders) for the audit report.
- **AUDSEC**
  The secondary space allocation (in cylinders) for the audit report.

The masking job contains the following steps:

- Runs IEFBR14 to delete and define the report and audit files.
  Note: The report file is allocated with SPACE=(CYL,(1,1)) which should be sufficient for most runs. If many error or warning messages are produced, you might need to increase the space allocation.
- Runs GTXMAP to read and parse the mapping CSV, and write the mapping CSV out to a fixed record format file. For more information, see GTXMAP Parameters.
- Runs GTXMSK to apply the masking/subsetting rules that are specified in the mapping CSV. For more information, see the section GTXMSK Parameters.

GTXMSK Flow Diagram

Figure 49: GTXMSK_flow2



GTXMSK Parameters

Auditing

The audit file contains the following information:

- Table name
- Unique column values for the rows that are being masked
- Name of the column being masked
- Old and new values for each column being masked

If the AUDIT parameter is not supplied then no audit data will be produced.

- **AUDIT=ALL**
  Specifies that all masked rows are detailed in the audit report.
- **AUDIT=ROW*nnn***
  Specifies the number of rows to be audited.
  Example: ROW1000 will audit the first 1000 rows.
- **AUDIT=SAMPLE*nnn***
  Specifies the interval at which rows are audited.
  Example: SAMPLE100 will audit every 100th row.
- **PAGELIMIT=*nnn***
  Specifies the audit output page limit
  Values: A positive number that indicates the page limit
  Default: 50
  Note: Default is regardless of the audit settings. To override the default, specify the number of audit pages to produce as *nnn*.

Cross-Reference

- **CASEINSENSITIVEXREF=**
  Specifies that cross-referencing is done regardless of case.
  Values: Y, N
  Default: N
- **TRIMMEDXREF=**
  Specifies that values are trimmed before cross-referencing.
  Values: Y, N
  Default: N

Dates

- **BADDATESTRING=*ccyymmdd***
  Specifies the date to replace unparseable dates for date functions.
  Default: Unparseable dates are not masked.
- **BASECENTURY=*nn***
  Specifies how BASECENTURY indicates the starting point for the century digit If the record definitions contain dateformats with a single digit century.
  Example: BASECENTURY=19 and a dateformat of "CYYMMDD","1880729" is interpreted as 29th July 1988.
- **CDATE=*ccyymmdd***
  Specifies to override the current date for the purposes of date calculation functions.
- **HIGHDATE=*ccyymmdd***

Specifies to override the highest date that offset date functions will process.

- **LOWDATE=*ccyymmdd***
  Specifies to override the lowest date that offset date functions will process.

Shuffling

Shuffling happens when the SHUFFLE function is specified in a transformation map. Shuffling is a two-phase process, in the first phase column values for which SHUFFLE is specified are used to populate a seedlist in GTSRC_REFERENCE_LOV1, in the second phase SHUFFLE functions are converted to SEQLOV functions which refer to the seedlist just created. Setting SHUFFLEONLY=Y results in just the first phase of the shuffle being executed, that is, a seedlist will be created but will not be used to update column values.

- **SHUFFLEDISTINCT=**
  Specifies the shuffle values that are created.
  Values: N (creates all values in the shuffle), Y (creates distinct values for the shuffle)
  Default: N
- **SHUFFLELIMIT=nnn**
  Specifies to only select nnn values for the shuffle.
- **SHUFFLEONLY=**
  Specifies to update GTSRC_REFERENCE_LOV1 with shuffle values and not update the database.
  Values: Y, N
  Default: N

Other

- **BLANKSASNULLS=**
  Specifies to treat fields that contain blanks as null.
  Values: N, Y
  Default: N
  Note: Where the mapping CSV includes the Keepnulls=Y option, blank fields are retained.
- **CASEINSENSITIVEFORMATENCRYPT=**Ignores case of input data.
  Values: Y, N
- **CASEINSENSITIVEHASH=**
  Specifies if the value that is hashed by functions HASHLOV and HASHLOV1 is converted to upper case before hashing.
  Values: N, Y
  Default: N
  Note: This parameter is the same as CASEINSENSITIVEHASHLOV. The program will use the last parameter that is read into the program. Required for consistent masking with FDM.
- **CASEINSENSITIVEHASHLOV=**
  Specifies if the value that is hashed by functions HASHLOV and HASHLOV1 is converted to upper case before hashing.
  Values: N, Y
  Default: N
- **CASEINSENSITIVESEED=**
  Specifies if RANDLOV1, SEQLOV1, and HASHLOV1 seed value lookup is case-sensitive.
  Values: N, Y
  Default: N
- **COMMIT=*nnn***
  Specifies the commit frequency for updates to the cross-reference or seed tables.
  Values: 1000, nnn

Default: 50000

- **DODBUPDATES=**
  Specifies whether the tables to be masked are updated. Using DODBUPDATES=N.
  Values: Y, N (Lets you validate masking rules and view audit results without committing changes)
  Default: Y
- **HASHTYPE=**
  Sets the hashing algorithm to use with the HASHLOV function.
  Values: ASM, JAVA
  Default: ASM
  Note: If JAVA is specified with the hashing on, zOS will produce the same hash value as FDM. JAVA option is required for consistent masking with FDM.
- **LANGUAGE=**
  Specifies the two-character language code used for output messages.
  Values: EN, DE, ES, IT
  Default: EN
- **LOWERCASEKEY=**(Optional) A key to be used in encrypting lower case letters.
- **NUMERICKEY=**
  (Optional) A key to be used in encrypting numbers.
- **ORDERBY=**
  N/Y. Decides whether or not selected data will be ordered by primary key columns.
  Values: Y, N
  Default: N
- **PROCESSCOUNT=nnn**
  Specifies the number of rows per table to be processed.
- **PROGRESSCOUNT=nnn**
  After each nnn rows are processed, the program will write out a line to the SYSOUT file, this may be useful to monitor the progress of a long running job.
- **TRIMMEDHASHLOV=**
  Specifies if the value that is hashed by functions HASHLOV and HASHLOV1 have leading and trailing blanks trimmed before hashing. Required for consistent masking with FDM.
  Values: Y, N
  Default: N
- **UPPERCASEKEY=**
  (Optional) A key to be used in encrypting upper case letters.
- **VALIDATEONLY=**Specifies if the input mapping CSV and parameters files are validated and any errors are reported.**Values**: N, Y**Default:** N

NOTE

The rules that are specified in the mapping CSV are not applied to the input file.

Mask and Unload DB2 Tables

To run DB2 masking and unloading with optional subsetting, JCL is supplied in the installation package as GTXMSKL. This job uses procedure GTMSKL.

You can use the following parameters for the JCL procedure:

- **LOADLIB**

Defines the load library that contains programs GTXMAP, GTXMSKL.

- **MSGDS**
  Defines the VSAM data set containing the TDM error messages.
- **REPHLQ**
  Gives the high-level dataset name qualifier that is used for the audit and report files.
- **LOADHLQ**
  Gives the High Level Qualifer to hold the load card and extracted data.
- **MAPDS**
  Defines the dataset that contains the mapping CSV (masking rules).
- **SUBDS (optional)**
  Defines the dataset that contains the Subset rules (if not being used - set to 'NULLFILE').
- **AUDPRI**
  Defines the primary space allocation (in cylinders) for the audit report.
- **AUDSEC**
  Defines the secondary space allocation (in cylinders) for the audit report.

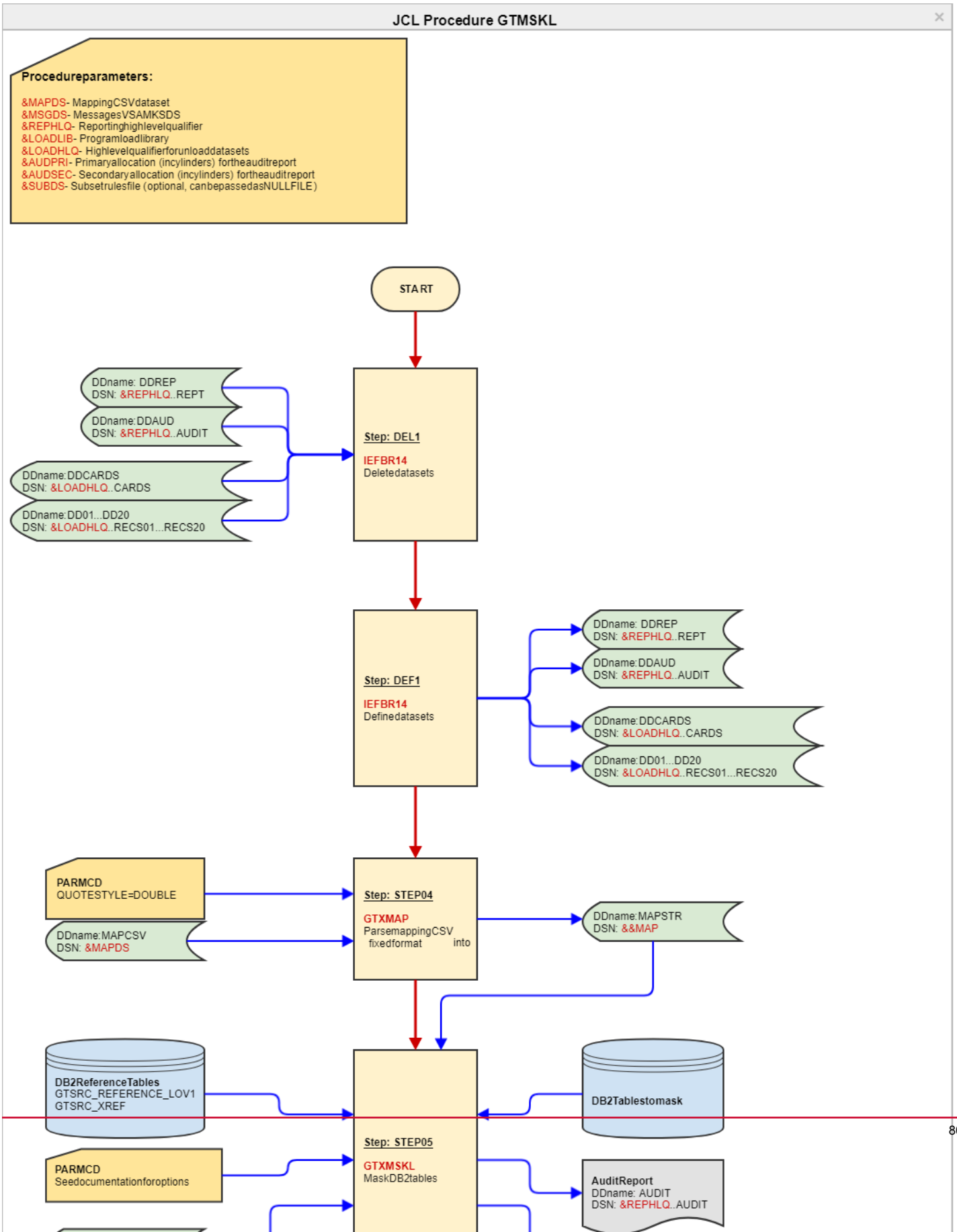
The masking job contains the following steps:

1. Runs IEFBR14 to delete the report, audit, DB2 load control cards, and the DB2 load data dataset(s).
2. Creates the report, audit, DB2 load control cards and the DB2 load data dataset(s).
   Note: The report file is allocated with SPACE=(CYL,(1,1)) which should be sufficient for most runs. If many error or warning messages are produced, increase the space allocation. The space allocation for the output load datasets is hard-coded in the JCL procedure. The space allocation may require adjustment to suit a specific run. 20 load datasets are defined in the JCL procedure. Each dataset contains data from one table. The masking program can handle up to 99 datasets. If more than 20 datasets are subsetted or masked in one run, amend the JCL.
3. Runs GTXMAP to read and parse the mapping CSV, and writes the mapping CSV out to a fixed record format file. For more information, see GTXMAP Parameters.
4. Runs GTXMSKL to apply the masking/subsetting rules that are specified in the mapping CSV. For more information, see GTXMSK and GTMSKL Parameters.

The output CARDS and SYSRECnn files from the job are in the required input format for the DSNUPROC utility which can be used to load the data into a DB2 instance. An example job using DSNUPROC is supplied in the installation package as GTXMSKL2.

GTXMSKL Flow Diagram

Figure 50: GTXMSKL_flow



GTXMSKL Parameters

Parameters for GTXMSKL are the same as those for GTXMSK. For more information about the GTXMSK parameters, see GTXMSK Parameters. In addition, for GTXMSKL you can set the following parameters:

Extra Parameters

- **ALIAS=**
  Specifies the alias for all of the subset tables. To be used when the subset SQL has been manually created / edited and the SQL uses an alias for the subset table.
  Do not use for SQL generated by GTSUBSET as the alias for the subset table has already been defined.
  Default: blank
- **APPLYSUBSETRULES=**
  Specifies that the subsetting rules supplied in the dataset for DDNAME SUBSET are used to restrict the output from the program.
  Values: Y, N
  Default: N
- **(Optional) FILECOUNT=**Specifies the maximum number of output files to which to write data. If the number of output files specified is less than the number of tables being processed, then some files will include more than one table.
  Values: 1 - 99 inclusive **Default:** 99
- **LOADPARM1=**
  Specifies whether to supply the first DB2 load utility control card for each table to be masked or subsetted.
  Note: If this parameter is not supplied, the program uses the following string as the control card: LOAD DATA REPLACE LOG YES. For possible options, see the IBM DB2 Utility Guide and reference section on LOAD.
- **LOADPARM2=**
  (Optional) Specifies whether to supply extra load utility control cards. For possible options, see the IBM DB2 LOADPARM3= Utility Guide and Reference section on LOAD.
- **TARGETSCHEMA=** *name*
  Specifies the name of the schema into which subsetted or masked data is loaded. This value supplied for this parameter is used to qualify table names in the load utility control cards dataset that are written by the program.
  Special values:
  - **ASINPUT**
    Each output schema has the same name as the input schema.
  - **PFX:** *output_*
    Each output schema has the same name as the input schema, prefixed with the text immediately after it (for example, '*output_*').

Subsetting DB2 Data

Creating Extract Definitions for DB2 Subset

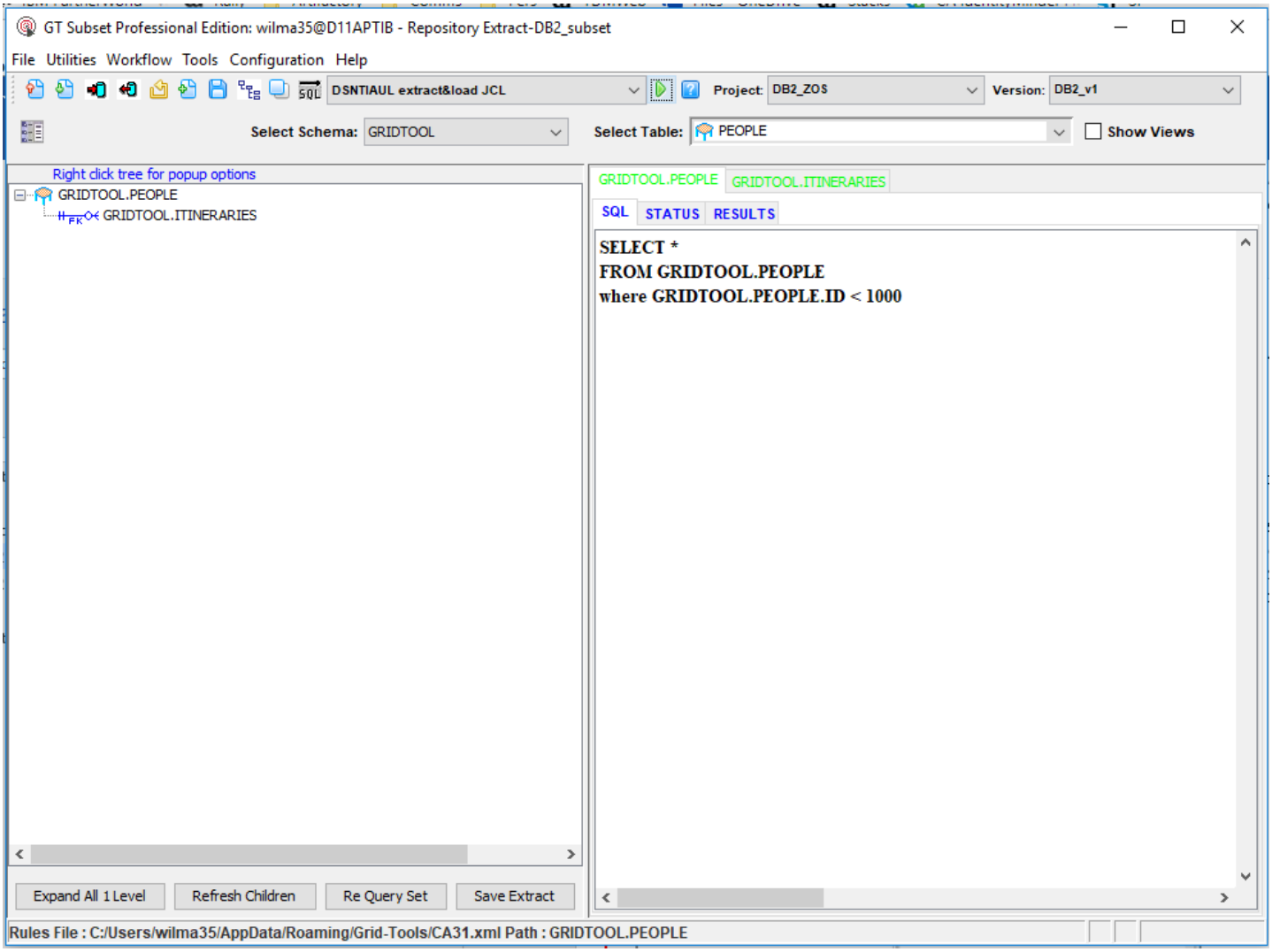
In order to subset DB2 data you need to connect GT Subset to your database via JDBC and then create an extract definition. For more information about connecting DB2 data to GT Subset, see Subset Production Data.

Executing DB2 Subsetting

The extract definition created in GT Subset is executed by running batch JCL on the mainframe. You can either simply create a subset or create subset with masked data.

DB2 Subsetting Without Masking

Creating a subset without masking is achieved by running DSNTIAUL in batch to read data from DB2 and write it out to files. The DB2 Load utility can then be run to load the data into the target database. The JCL for the unload and load jobs can be created by using a JCL template in GT Subset. Within the Datamaker installation directory, under Templates\DB2, a template called DSNTIAUL_extract and load_JCL.xml is supplied with the installation. This template should be amended as per your JCL standards and dataset naming conventions. For more information about how templates work and what you can do with them, see Using Templates to Generate Scripts.



This is the extract and load template:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<FILE>
  <!--Parameters - these are used to substitute values into the output file--
>
  <PARMS>UNLOAD JOB CARD=//GRIDT01X JOB 'DSNTIAUL',CLASS=A,NOTIFY=&amp;SYSUID</PARMS>
  <PARMS>LOAD JOB CARD=//GRIDT01X JOB 'DSNTIAUL',CLASS=A,NOTIFY=&amp;SYSUID</PARMS>
```

```
  <PARMS>DSN HLQ=GRIDT01.DSN</PARMS>
  <PARMS>SRC SCHEMA=GRIDTOOL</PARMS>
  <PARMS>TGT SCHEMA=GRIDT01</PARMS>
  <!--This is the name of the output file containing unload JCL--
>
  <FILENAME>[ACTION NAME].jcl</FILENAME>
  <LINEWIDTH>72</LINEWIDTH>
  <TEXT>[UNLOAD JOB CARD]</TEXT>
  <TEXT>//* ------------------------------------------*</TEXT>
  <TEXT>//DEL EXEC PGM=IEFBR14</TEXT>
  <TEXT>//DDCARDS DD DSN=[DSN HLQ].CARDS,</TEXT>
  <TEXT>// DISP=(MOD,DELETE),SPACE=(TRK,0)</TEXT>
  <!--Allocate datasets to contain the table data--
>
  <ALLEXTRACT>
    <TEXT>//DD[GCOUNTER1] DD DSN=[DSN HLQ].RECS[GCOUNTER2],</TEXT>
    <TEXT>// DISP=(MOD,DELETE),SPACE=(TRK,0)</TEXT>
  </ALLEXTRACT>
  <ALLDATA>
    <TEXT>//DD[GCOUNTER1] DD DSN=[DSN HLQ].RECS[GCOUNTER2],</TEXT>
    <TEXT>// DISP=(MOD,DELETE),SPACE=(TRK,0)</TEXT>
  </ALLDATA>
    <TEXT>//* ------------------------------------------*</TEXT>
    <TEXT>//STEP1 EXEC PGM=IKJEFT01,DYNAMNBR=20</TEXT>
    <TEXT>//STEPLIB DD DISP=SHR,DSN=DSN810.SDSNEXIT</TEXT>
    <TEXT>// DD DISP=SHR,DSN=DSN810.SDSNLOAD</TEXT>
    <TEXT>// DD DISP=SHR,DSN=CEE.SCEERUN</TEXT>
    <TEXT>//SYSOUT DD SYSOUT=*</TEXT>
    <!--The SYSPUNCH dataset will be written with load utility control cards--
>
    <TEXT>//SYSPUNCH DD DSN=[DSN HLQ].CARDS,</TEXT>
    <TEXT>// UNIT=SYSDA,SPACE=(800,(15,15)),DISP=(NEW,CATLG,CATLG),</TEXT>
    <TEXT>// DCB=(RECFM=FB,LRECL=120,BLKSIZE=1200)</TEXT>
    <!--Data from each table is written to a separate dataset--
>
  <ALLEXTRACT>
    <TEXT>//SYSREC[GCOUNTER3] DD DSN=[DSN HLQ].RECS[GCOUNTER4],</TEXT>
    <TEXT>// UNIT=SYSDA,DISP=(NEW,CATLG,CATLG),</TEXT>
    <TEXT>// SPACE=(4096,(500,100)),</TEXT>
    <TEXT>// DCB=(RECFM=FB,BLKSIZE=6480)</TEXT>
  </ALLEXTRACT>
  <ALLDATA>
    <TEXT>//SYSREC[GCOUNTER3] DD DSN=[DSN HLQ].RECS[GCOUNTER4],</TEXT>
    <TEXT>// UNIT=SYSDA,DISP=(NEW,CATLG,CATLG),</TEXT>
    <TEXT>// SPACE=(4096,(500,100)),</TEXT>
    <TEXT>// DCB=(RECFM=FB,BLKSIZE=6480)</TEXT>
```

```
  </ALLDATA>
    <TEXT>//SYSTSPRT DD SYSOUT=*</TEXT>
    <TEXT>//SYSPRINT DD SYSOUT=*</TEXT>
    <!--SYSIN contains the extract queries created by GT Subset--
>
    <TEXT>//SYSIN DD *</TEXT>
    <ALLEXTRACT>
    <TEXT>[SP]</TEXT>
    <TEXT>SELECT * FROM [OWNER].[TABLE] [QUERY1];</TEXT>
    </ALLEXTRACT>
  <ALLDATA>
    <TEXT>[SP]</TEXT>
    <TEXT>SELECT * FROM [OWNER].[TABLE];</TEXT>
  </ALLDATA>
    <TEXT>/*</TEXT>
    <TEXT>//SYSTSIN DD *</TEXT>
    <TEXT>DSN SYSTEM(DB8G) RETRY(0) TEST(0)</TEXT>
    <TEXT>RUN PROGRAM(DSNTIAUL) PLAN(DSNTIB81) -</TEXT>
    <TEXT> PARMS('SQL') -</TEXT>
    <TEXT> LIB('DSN810.RUNLIB.LOAD')</TEXT>
    <TEXT>END</TEXT>
    <TEXT>/*</TEXT>
    <!--Create another output file to hold the load utility JCL--
>
    <NEWFILENAME>[ACTION NAME]L.jcl</NEWFILENAME>
    <LINEWIDTH>72</LINEWIDTH>
    <TEXT>[LOAD JOB CARD]</TEXT>
    <TEXT>//PROCLIB JCLLIB ORDER=DSN810.PROCLIB</TEXT>
    <TEXT>//* ------------------------------------------*</TEXT>
    <TEXT>//EDIT EXEC PGM=GTMOD</TEXT>
    <TEXT>//STEPLIB DD DISP=SHR,DSN=GRIDT01.LOADLIB</TEXT>
    <TEXT>//SYSOUT DD SYSOUT=*</TEXT>
    <TEXT>//FILEI DD DISP=SHR,DSN=[DSN HLQ].CARDS</TEXT>
    <TEXT>//FILEO DD DSN=&amp;amp;&amp;amp;CARDS,DCB=WILMA35.TEMP.CARDS,</TEXT>
    <TEXT>// DISP=(NEW,PASS,DELETE),SPACE=(TRK,(1,1))</TEXT>
    <!--The control cards file refers to the source schema, change this to the target
 schema--
>
    <TEXT>//SYSIN DD *</TEXT>
    <TEXT>&lt;FROM&gt;[SRC SCHEMA].&lt;/FROM&gt;</TEXT>
    <TEXT>&lt;TO&gt;>[TGT SCHEMA].&lt;/TO&gt;</TEXT>
    <TEXT>/*</TEXT>
    <TEXT>//*</TEXT>
    <TEXT>//* ------------------------------------------*</TEXT>
    <TEXT>//LOAD EXEC DSNUPROC,SYSTEM='DB8G',COND=(4,LT)</TEXT>
    <TEXT>//SYSOUT DD SYSOUT=*</TEXT>
```

```
    <TEXT>//SYSIN DD DSN=&amp;amp;&amp;amp;CARDS,DISP=(OLD,DELETE)</TEXT>
  <ALLEXTRACT>
    <TEXT>//SYSREC[GCOUNTER5] DD DSN=[DSN HLQ].RECS[GCOUNTER6],</TEXT>
    <TEXT>// DISP=OLD</TEXT>
  </ALLEXTRACT>
  <ALLDATA>
    <TEXT>//SYSREC[GCOUNTER5] DD DSN=[DSN HLQ].RECS[GCOUNTER6],</TEXT>
    <TEXT>// DISP=OLD</TEXT>
  </ALLDATA>
    <TEXT>//SYSTSPRT DD SYSOUT=*</TEXT>
    <TEXT>//SYSPRINT DD SYSOUT=*</TEXT>
    <TEXT>//SYSUT1 DD DSN=CSYSUT1,DISP=(,PASS),</TEXT>
    <TEXT>// SPACE=(4096,(20,20),,,ROUND)</TEXT>
    <TEXT>//SORTOUT DD DSN=&amp;amp;&amp;amp;SORT1,DISP=(,PASS),</TEXT>
    <TEXT>// SPACE=(4096,(20,20),,,ROUND)</TEXT>
</FILE>
```

For a subset extract with just two tables the following template creates the extract job:

```
//GRIDT01X JOB 'DSNTIAUL',CLASS=A,NOTIFY=&SYSUID
//* ----------------------------------------*
//DEL EXEC PGM=IEFBR14
//DDCARDS DD DSN=GRIDT01.DSN.CARDS,
// DISP=(MOD,DELETE),SPACE=(TRK,0)
//DD00 DD DSN=GRIDT01.DSN.RECS00,
// DISP=(MOD,DELETE),SPACE=(TRK,0)
//DD01 DD DSN=GRIDT01.DSN.RECS01,
// DISP=(MOD,DELETE),SPACE=(TRK,0)
//* ----------------------------------------*
//STEP1 EXEC PGM=IKJEFT01,DYNAMNBR=20
//STEPLIB DD DISP=SHR,DSN=DSN810.SDSNEXIT
// DD DISP=SHR,DSN=DSN810.SDSNLOAD
// DD DISP=SHR,DSN=CEE.SCEERUN
//SYSOUT DD SYSOUT=*
//SYSPUNCH DD DSN=GRIDT01.DSN.CARDS,
// UNIT=SYSDA,SPACE=(800,(15,15)),DISP=(NEW,CATLG,CATLG),
// DCB=(RECFM=FB,LRECL=120,BLKSIZE=1200)
//SYSREC00 DD DSN=GRIDT01.DSN.RECS00,
// UNIT=SYSDA,DISP=(NEW,CATLG,CATLG),
// SPACE=(4096,(500,100)),
// DCB=(RECFM=FB,BLKSIZE=6480)
//SYSREC01 DD DSN=GRIDT01.DSN.RECS01,
// UNIT=SYSDA,DISP=(NEW,CATLG,CATLG),
// SPACE=(4096,(500,100)),
// DCB=(RECFM=FB,BLKSIZE=6480)
//SYSTSPRT DD SYSOUT=*
```

```
//SYSPRINT DD SYSOUT=*
//SYSIN DD *

SELECT *
FROM GRIDTOOL.PEOPLE
where GRIDTOOL.PEOPLE.ID < 1000 ;

SELECT *
FROM GRIDTOOL.ITINERARIES
where (AUTHORISATION_ID)
in (
select L0.ID
from (
SELECT *
FROM GRIDTOOL.PEOPLE L0
where L0.ID < 1000) L0) ;
/*
//SYSTSIN DD *
DSN SYSTEM(DB8G) RETRY(0) TEST(0)
RUN PROGRAM(DSNTIAUL) PLAN(DSNTIB81) -
 PARMS('SQL') -
 LIB('DSN810.RUNLIB.LOAD')
END
/*
```

The tables in the target database should be defined before this job is run as follows:

```
//GRIDT01X JOB 'DSNTIAUL',CLASS=A,NOTIFY=&SYSUID
//PROCLIB JCLLIB ORDER=DSN810.PROCLIB
//* ----------------------------------------*
//EDIT EXEC PGM=GTMOD
//STEPLIB DD DISP=SHR,DSN=GRIDT01.LOADLIB
//SYSOUT DD SYSOUT=*
//FILEI DD DISP=SHR,DSN=GRIDT01.DSN.CARDS
//FILEO DD DSN=&&CARDS,DCB=WILMA35.TEMP.CARDS,
// DISP=(NEW,PASS,DELETE),SPACE=(TRK,(1,1))
//SYSIN DD *
<FROM>GRIDTOOL.</FROM>
<TO>>GRIDT01.</TO>
/*
//*
//* ----------------------------------------*
//LOAD EXEC DSNUPROC,SYSTEM='DB8G',COND=(4,LT)
//SYSOUT DD SYSOUT=*
//SYSIN DD DSN=&&CARDS,DISP=(OLD,DELETE)
//SYSREC00 DD DSN=GRIDT01.DSN.RECS00,
```

```
// DISP=OLD
//SYSREC01 DD DSN=GRIDT01.DSN.RECS01,
// DISP=OLD
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DSN=CSYSUT1,DISP=(,PASS),
// SPACE=(4096,(20,20),,,ROUND)
//SORTOUT DD DSN=&&SORT1,DISP=(,PASS),
// SPACE=(4096,(20,20),,,ROUND)
```

DB2 Subsetting With Masking

To combine masking with subsetting, follow these steps:

1. Create your extract definition using GT Subset.
2. Save the extract in the Datamaker repository.
3. Create your masking rules using the Datamaker Transformation Maps dialog.
4. When you save the transformation map to a file you are prompted to attach a subset.
5. Datamaker creates two files, a transformation map csv and a subset rules file.
   These files should be transferred to the mainframe and used as input to the DB2 Mask and Unload job. For more information, see Mask and Unload DB2 Tables.
6. If you wish you can use template GTXMSKL_jcl from GT Subset to generate the jobs required.
   Modify this template as required by your mainframe environment.

Data Generation for DB2

You can generate synthetic test data for zOS DB2 using Datamaker. For more information, see Generate Synthetic Test Data.

Working with Mainframe Files or IMS Segments

In order to work with files, Test Data Manager needs a definition that describes the record layouts used in a file, and, for multi-record files the conditions under which a given layout is applicable. Test Data Manager uses an Advanced File Layout (AFL) to describe files. This is an internal format which is derived either from COBOL or PL1 copy books.

Test Data Manager does not work directly with IMS databases. To mask IMS databases, the segment data should be dumped to flat files, which can be masked and then used to reload a database. For more information, see How to Parse IMS Database Copybooks and Mask Data

Create an Advanced File Layout (AFL) with File Definition Manager

You should use the File Definition Manager to create an Advanced File Layout (AFL).

Configure File Definition Manager

Configure File Definition Manager before you use it to create an Advanced File Layout.

Follow these steps:

1. Open File Definition Manager and click **Configuration**.

2. Specify the **Language** by choosing *one* of the following:
   - **COBOL**
   - **PL1**
3. If you intend to generate synthetic data for the file you should select the method that you will be using for data generation. For the **Create Data Using option** select one of the following:
   - **DB2**To generate file data via DB2 tables. In this case the File Definition Manager outputs just one file: a **Z/OS AFL**. For more information, see Generate Synthetic Mainframe File Data using DB2 Tables.
   - **FD File**To generate data directly to a file. If you use this option the File Definition Manager will output three files:
     - A **Z/OS AFL**, which describes the file as it exists on the mainframe.
     - A **_DG** file, which describes the file as it is generated in Windows.
     - A **G-T Excel file definition**, which is the object you should register in Datamaker or the Portal. For more information see Generate Synthetic Mainframe File Data using File Definitions.
4. (Optional) Define the **Left Margin**. Specify the number of characters at the start of each line of a copybook to ignore during parsing. For Cobol copy books the File Definition manager parses the data found in columns 8 to 72 of the copy book, for PL1 the data found in columns 1 to 72 is used. If the input copy books match this requirement Left Margin should be left as 0.
5. In **Copybook Location** specify where the copy book files you are using have been saved.
6. In **Output Location specify where you want the File Definition Manager outputs to be saved**.
7. In **Definitions Location** specify where you want file definitions to be saved. The File Definition Manager creates an XML file to store all the meta-data associated with a file, this XML file is saved in the Definitions Location.
8. (Optional) Define **Pre-processor Replacements**.
   The File Definition Manager only recognizes valid Cobol or PL1 declarative statements. If your copy book(s) contain invalid characters, for example colons in variable names, you can specify a replacement for these to be made before the copy book is parsed. In addition you can replace tab characters (identified by the literal <TAB>) with spaces.
9. Define the **Pre-processor copy keyword.** If you have nested copy books, supply the keyword used to signal the inclusion of a nested copybook. Note that for nested copy books to be processed correctly the the copy books should all have the same file extension.
   Example: INCLUDE, COPY
   Default: COPY
10. Click **Update** to save your changes, and close the Configuration window.

The configuration file **FileDefinitionManagerConfig.txt** is created at `~/.fdm/`

Log files for File Definition Manager are created at `~/.fdm/log`

Prepare Copybooks

The File Definition Manager is not a full language parser and only recognizes valid data item declarations. If your copy book contains other language elements, or invalid declarations you should edit the copybook before attempting to parse it with the Definition Manager.

It isn't possible to determine from a copy book where a record layout begins and ends. The File Definition Manager makes the assumption that a level-1 declaration marks the start of a record, and that all the data items enclosed by the level-1 declaration are part of the record. It may be necessary to edit copy books to match this assumption.

If your copy books don't include a level-1 declaration you may be able to avoid editing the copy books themselves by creating an enclosing copy book. For example, if you have a file with two records defined by copy books COBCB1 and COBCB2 as follows:

COBCB1

```
 05 RECORD1-STRUCTURE.
    10 FIELD-1 PIC S9(4) COMP.
```

```
   10 FIELD-2 PIC X(32).
   10 FIELD-3 PIC X(10).
```

COBCB2

```
05 RECORD2-STRUCTURE.
   10 FIELD-1 PIC X(1).
   10 FIELD-2 PIC X(32).
   10 FIELD-3 PIC X(55).
```

You could create the following copy book to add level-1 declarations:

```
01 RECORD-1.
   COPY COBCB1
01 RECORD-2.
   COPY COBCB2
```

Parse copybooks

To create a new file definition, follow these steps:

1. Click **New Definition**.
2. Define a **Definition Name** for this set of copy books this will be used to identify the outputs from the File Definition Manager.
3. Click **Add Copybook**. Include one or more copybooks from the location that you set in the configuration step.
   Tip: To exclude a copybook from parsing, highlight a copybook line, and click **Remove Copybook**.
   The copybooks selected for parsing are displayed.
4. Click **Parse Copybooks**. Wait for the log panel to confirm that the parser has completed.
5. Click **Save** and close the **New File Definition** window.
   The configuration data, copy book details, and parse results are saved in an XML file in the Definitions Location that you set in the configuration step.

Enhance Parsed Output

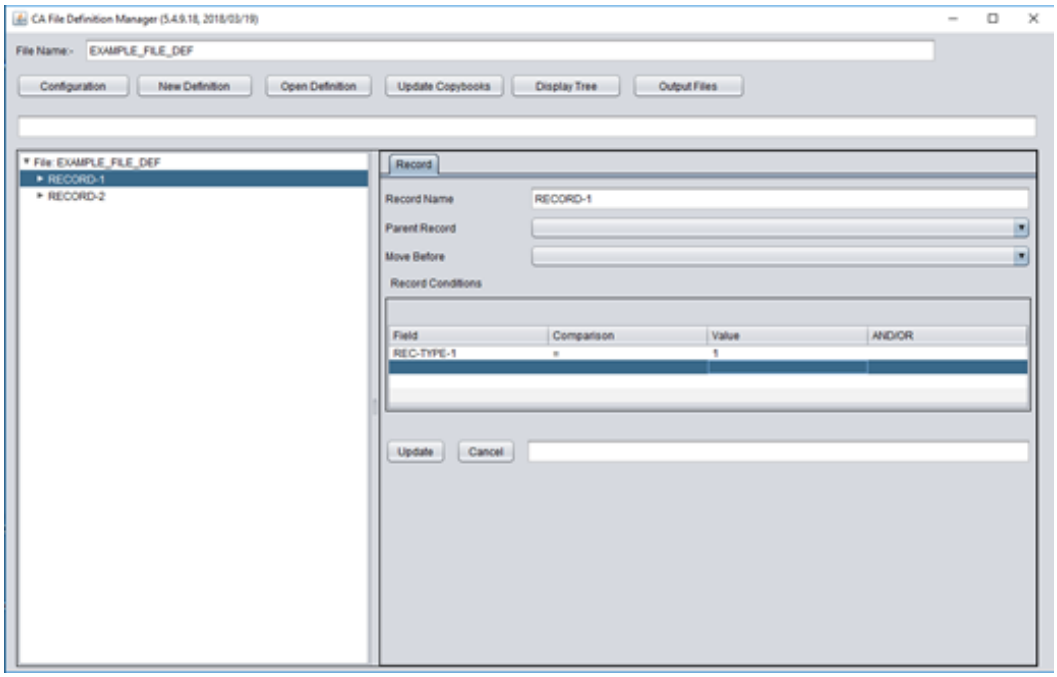
In most cases copy books do not supply all the meta-data required to fully define a file and its record layouts, it is necessary to manually supply the missing information.

Click on the Display Tree button to show the record structures derived from parsing the copy books.

If there were no level-1 declarations in the copy books then a "TEMPLATE_RECORD" will be displayed. You cannot edit or delete the TEMPLATE_RECORD. The template is ignored when producing outputs from the File Definition Manager. The TEMPLATE_RECORD is a dummy template from which you copy structures to create your required record layouts. Right click on the "File" element at the top of the tree structure and select "add record", this will add an empty record element to the structure. You can then copy and paste elements from the TEMPLATE_RECORD to your new record to create the required record layout.

If a file contains multiple record layouts it is necessary to define the conditions under which a given layout applies.

Highlight a record in the tree structure and in the Record Conditions box specify when the record layout applies. You can define a record as a header or trailer, or supply field comparisons.

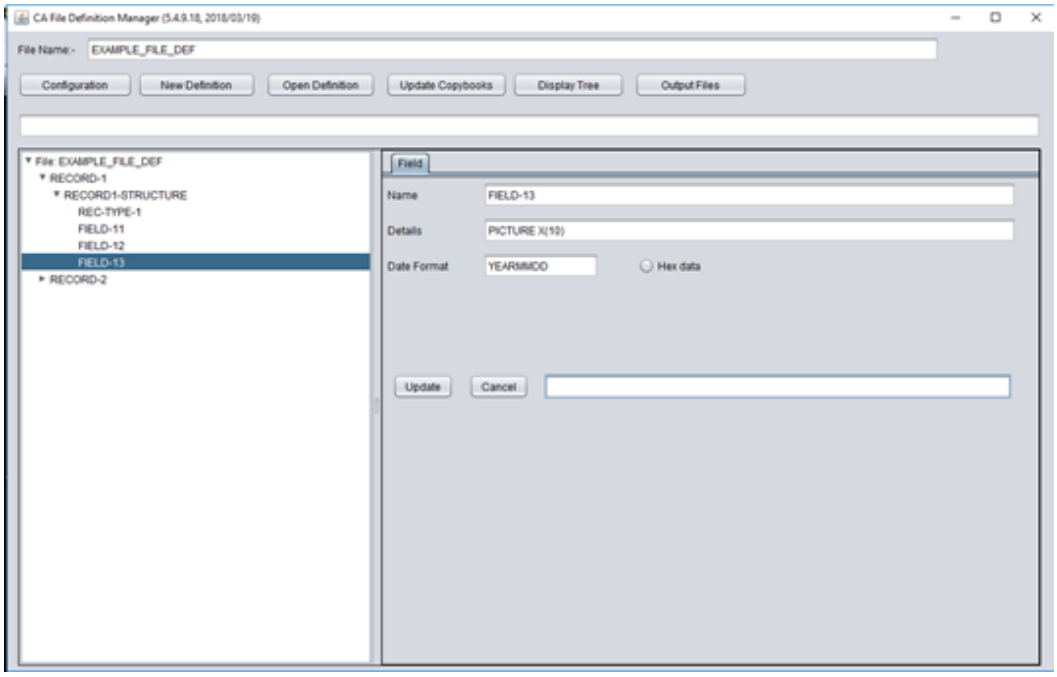


For a record you can also define hierarchical relationships by supplying a parent record, and you can specify the order in which records occur within the file.

Having added record details click on Update to save these.

If any records in the file contain date fields you should specify the date format for these.

Navigate to the date field in the tree structure and enter the date format.



If there are any fields in the file for which you want to be able to generate non-display hex values you should mark these by selecting the Hex data radio button for the field.

Troubleshooting: How to Correct Parser Errors

If the utility encounters a problem, it displays an error message.

TIP

If the utility continues processing after encountering the first error, more errors might result. Investigate the first error before you investigate later errors. Correcting the first error might clear up later errors.

Example: Parser Error Interpretation

The `ParserErr.txt` file contains, for example, the following message:

```
C:/Grid-Tools/mainframe/zOS_CopybookParser/Parser/example.txt
line 2:10 missing DOT at '05'
```

The line number refers to lines within the file `example.txt` . This line is the output of the first phase of the parse. This phase attempts to remove comment lines from the input copybooks. The following details are the file contents:

```
01 ADDR-REC
    05 REC-ID PIC X.
     05 ADDR-ID PIC S9(9) COMP.
     05 EMPLOYEE-ID PIC S9(9) COMP.
     05 ADDR-LINE-1 PIC X(40).
     05 ADDR-LINE-2 PIC X(40).
     05 ADDR-LINE-3 PIC X(40).
     05 POST-CODE PIC X(12).
     05 CITY PIC X(30).
     05 STATE PIC X(25).
     05 COUNTRY PIC X(4).
01 CCARD-REC.
     05 REC-ID PIC X.
     05 CARD-ID PIC S9(9) COMP.
     05 CARD-NO PIC X(30).
     05 TYPE PIC X(2).
     05 EXPIRY PIC X(10).
01 PEOPLE-REC.
     05 REC-ID PIC X.
     05 ID PIC S9(9) COMP.
     05 DESIGNATION PIC X(4).
     05 FIRST-NAME PIC X(40).
     05 LAST-NAME PIC X(40).
     05 ADDRESS-LINE PIC X(200).
```

The parser is indicating an error in line two, although line one is the one that is missing a period in the end. The line numbering can be off by one because the parser continues until a non-blank character, the "0" on line two, is found. Add the missing period in line *one* to resolve the error, and parse the copybook again.

Unable to Resolve Parser Errors?

If you are unable to resolve the problem, forward the following information to CA Support:

- The contents of the Output directories
- The copybook that you are attempting to parse

Create File Definitions

Once you are happy with your file definition click on **Output Files** to produce the definition files for use in TDM.

If you selected **Create data using DB2** in the File Definition Manager configuration, one file will be written to the Output location set in the configuration, this file will be suffixed `_ZOS.AFL.DM.txt` .

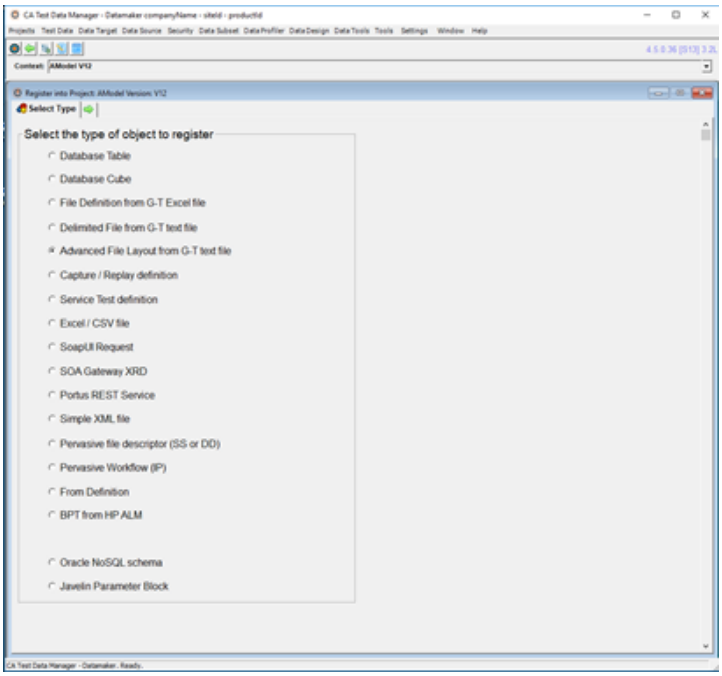
If you selected Create data using FD file three files will be written:

- An AFL suffixed `_ZOS.AFL.DM.txt`
- An AFL suffixed `_DG.AFL.DM.txt`
- A G-T Excel file definition (with a .xls extension)

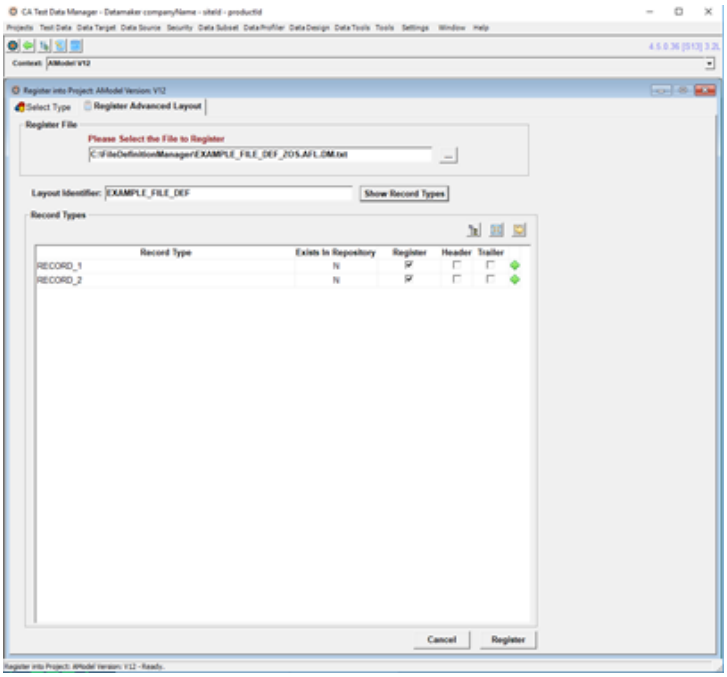
Register File Layouts

To create masking rules for files using the Transformation Maps dialog in Datamaker you should register the ZOS AFL (suffix _ZOS.AFL.DM.txt) created by the File Definition Manager.

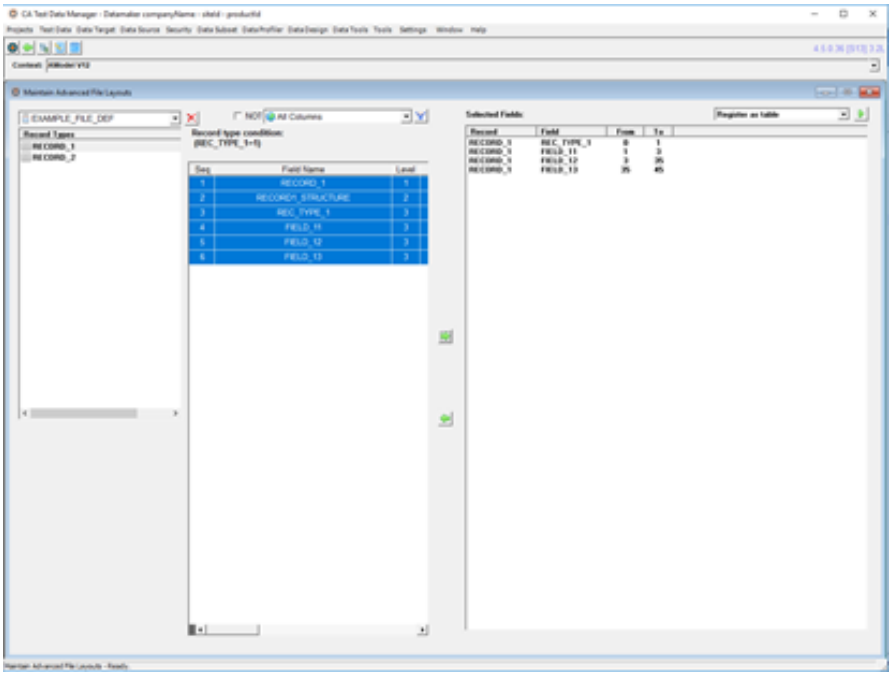
Register the file as an Advanced File Layout from G-T text file.



Navigate to your AFL and click Show Record Types, followed by Register.



For each record layout highlight all the fields and then Register as table.



If you selected Create data using FD File in your File Definition Manager configuration then you should register the output Excel file as a File Definition from G-T Excel file. This registered object should only be used for data generation, not for masking.

Profile z/OS Files

You can run a batch job to profile or sample file data. The output from this job is a CSV file which can be loaded into Datamaker to aid in designing masking or generation rules.

NOTE

If you do not load profile data during the file registration process, you can load this data later.

Profile (Sample) Flat Files

To run flat file profiling, JCL is supplied in the installation package as GTXPRO. This job uses procedure GTPRO.

You can supply the following parameters to the JCL procedure:

- **LOADLIB**
  Names the load library that contains programs GTXDEF, GTXPRO1 and GTXPRO2.
- **INFILE**
  Names the file to be profiled.
- **DEFFILE**
  Names the dataset that contains the record definition CSV (Advanced File Layout).
- **PROFILE**
  Names the dataset to contain the output profiling data.
- **REPHLQ**
  Gives the high-level dataset name qualifier to be used for the audit and report files.
- **SP1PRI,SP1SEC**
  The primary and secondary space allocation (in cylinders) for the first work dataset that is used by program GTXPRO1.

  NOTE
  The required size of this dataset depends on the number of fields that are profiled. A value of "1" for both parameters is likely to be sufficient in most cases.
- **SP2PRI,SP2SEC**
  The primary and secondary space allocation (in cylinders) for the second work dataset used by program GTXPRO1.

  NOTE
  The required size of this dataset depends on the number of non-numeric fields values that are profiled. The required size of this dataset depends on the number of numeric field values that are profiled. The larger the sample size, the larger the space allocation for this dataset needs to be.
- **SP3PRI,SP3SEC**
  The primary and secondary space allocation (in cylinders) for the third work dataset that is used by program GTXPRO1. The required size of this dataset depends on the number of numeric field values that are profiled. The larger the sample size, the larger the space allocation for this dataset needs to be.

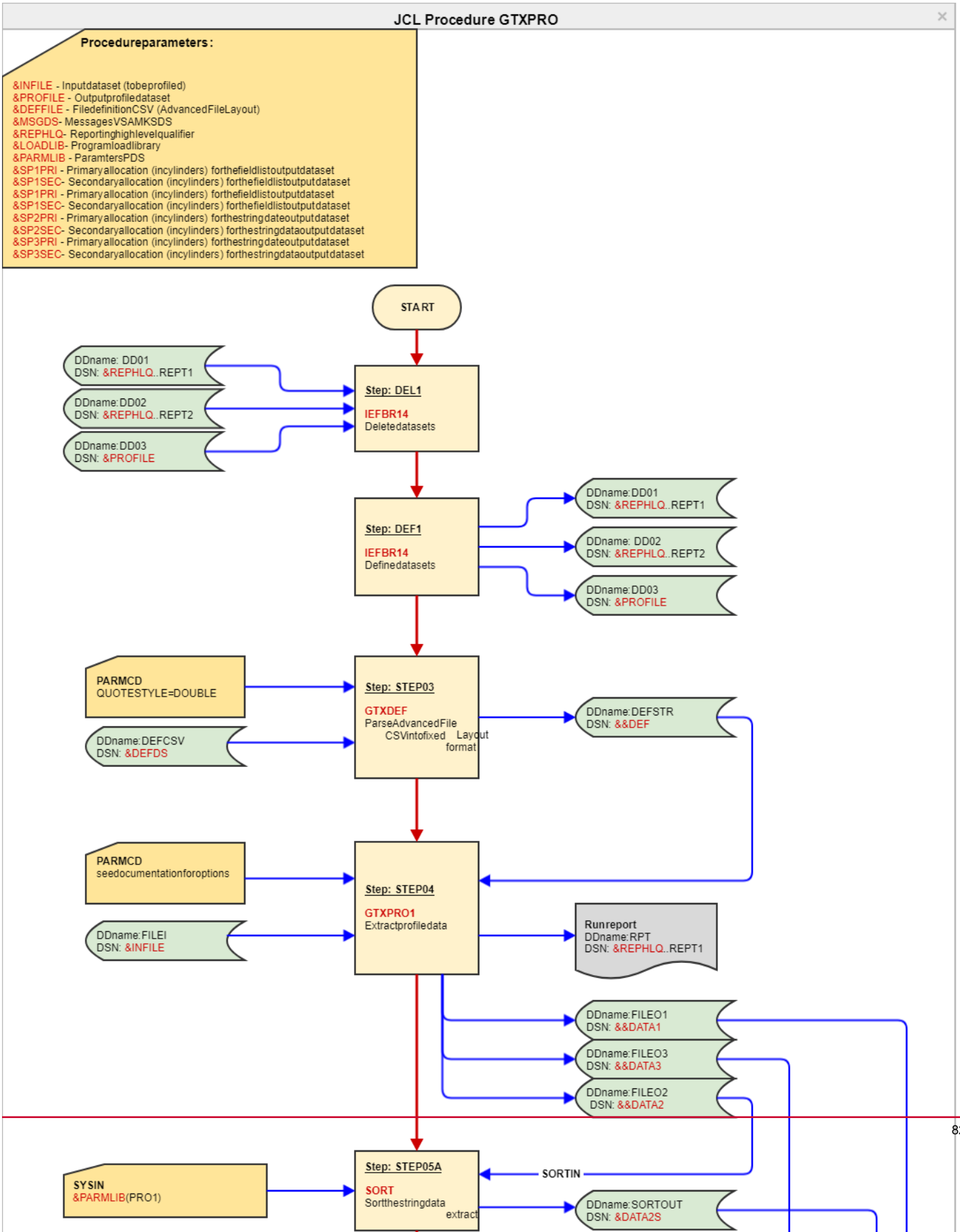
The profiling job contains the following steps:

1. IEFBR14 to delete the report files and the output profiling data file.
2. IEFBR14 to define the report files and the output profiling data file.

   NOTE
   The output profiling data is allocated with SPACE=(CYL,(1,1)) which should be sufficient for most runs. If a large number of error or warning messages are produced, you may need to increase the space allocation.
3. Runs GTXDEF to read the record definition CSV, parse it, and write it out to a fixed record format file. See GTXDEF Parameters.
4. Runs the profile extract program GTXPRO1. See GTXPRO1 Parameters.
5. Sorts non-numeric field data items from GTXPRO1.
6. Sorts numeric data items from GTXPRO1.
7. Runs the profile analysis program GTXPRO2. See GTXPRO2 Parameters.

GTXPRO Flow Diagram

Figure 51: gtxpro



GTXPRO1 Parameters

Subset

The following parameters let you control the sample size that is profiled:

- **SUBSET=ALL**
  Specifies all records in the input file to be profiled
- **SUBSET=ROWnnn**
  Specifies the number of records that are sampled starting with the first record. Once sufficient rows have been sampled, the program stops processing the input file.
  Example: ROW1000 samples the first 1000 records.

  NOTE
  For multi-record format files, the first 1000 records of each type that are found in the file are sampled. The resultant pctscan value will match the rows read to meet the above requirement (which may not be the whole file).
- **SUBSET=SAMPLEnnn**
  Specifies the interval at which records are sampled
  Example: SAMPLE100 samples every 100th record.

  NOTE
  For multi-record format files, every 100th record of each type that is found in the file is sampled.

Dates

- **BASECENTURY=nn**
  Specifies how BASECENTURY indicates the starting point for the century digit If the record definitions contain dateformats with a single digit century
  Example: BASECENTURY=19 and a dateformat of "CYYMMDD","1880729" is interpreted as 29th July 1988

Other

- **FIELD=**
  Restricts the sampling to specific fields
  Values: A record name and a field name separated by a period (.).
  Example: RECA.FIELDB
  Default: By default, the program samples data from all record types that are defined in the input record definition CSV. The program also samples data from all fields that are defined for those records.

  NOTE
  The record name or field name can be replaced by an asterisk to act as a wild card.

  Examples: *.FIELDB", "RECA.*

  NOTE
  You can repeat the FIELD parameter up to 5000 times.
- **LANGUAGE=**
  Specifies the two-character language code that is used for output messages.
  Values: EN, DE, ES, IT
  Default: EN
- **PAGELIMIT=nnn**
  Specifies the number of pages that the output report file contains.

  NOTE
  Note: Use this parameter to override the default page limit.

Values: 50, nnn
Default: 50

GTXPRO2 Parameters

Profiling Options

The following parameters let you control the type of profiling that is performed:

NOTE

Full profiling is the default. For example, all of the implemented functionality is used.

- **ANALYSIS=**
  Specifies whether to analyse field values and assign categories.
  Values: Y, N
  Default: Y
- **AVERAGE=**
  Specifies whether to determine the average value for each numeric field.
  Values: Y, N
  Default: Y
- **BOTTOM_PERC_PERCENT=**
  Defines the bottom percentile
  Values: 90, nn
  Default: 90
- **BOTTOM_PERC_VALUE=**
  Specifies whether to determine the bottom percentile value for each field. The bottom percentile used defaults to 10.
  Values: Y, N
  Default: Y
- **DISTINCTCOUNT=**
  Specifies whether to determine the number of distinct values for each field.
  Values: Y, N
  Default: Y
- **DISTINCT_VALUES=**
  Specifies whether to produce a list of distinct values.
  Values: Y, N
  Default: Y

  NOTE
  By default, the maximum number of distinct values that are output is 50.
- **INVALIDCOUNT=**
  Specifies whether to determine the number of invalid values for each field.
  Values: Y, N
  Default: Y
- **MAXIMUM=**
  Specifies whether to determine the maximum value for each field.
  Values: Y, N
  Default: Y
- **MEDIAN=**
  Specifies whether to determine the median value for each field.
  Values: Y, N
  Default: Y

- **MINIMUM=**
  Specifies whether to determine minimum value for each field.
  Values: Y, N
  Default: Y
- **STDDEV=**
  Specifies whether to determine the standard deviation for each numeric field.
  Values: Y, N
  Default: Y
- **TOP_PERC_PERCENT=**
  Defines the top percentile.
  Values: a number *nn*
  Default: 90
- **TOP_PERC_VALUE=**
  Specifies whether to determine the top percentile value for each field.

  NOTE
  The top percentile used defaults to 90

  Values: Y, N
  Default: Y

Output Format

The following parameters control the format of the CSV file that is written.

NOTE

We recommend that you use the default settings. For the default settings, transfer the output CSV to Windows in binary mode for loading into Test Data Manager.

- **ADDCRLF=**
  Specifies whether each record that is written is suffixed with X'0D0A

  NOTE
  This parameter is only applicable if CONVERTTOASCII is set to Y. If Y, each record written is suffixed with X'0D0A'.

  Values: Y, N
  Default: Y
- **BLANKNONDISPLAY=**
  Specifies which non-display characters are converted to blanks
  Values: 7 (converts non-display characters in the 7-bit ASCII character set), 8 (converts non-display characters in the 8-bit ASCII character set), N.
  Default: 7
- **CONVERTTOASCII=**
  Specifies whether to convert EBCDIC data in the input file to ASCII.
  Values: Y, N
  Default: Y

Other

- **LANGUAGE=**
  Specifies the two-character language code that is used for output messages
  Values: EN, DE, ES, IT

Default: EN

- **PAGELIMIT=**
  Specifies the number of pages that the output report file contains

  NOTE
  Use this parameter to override the default page limit.

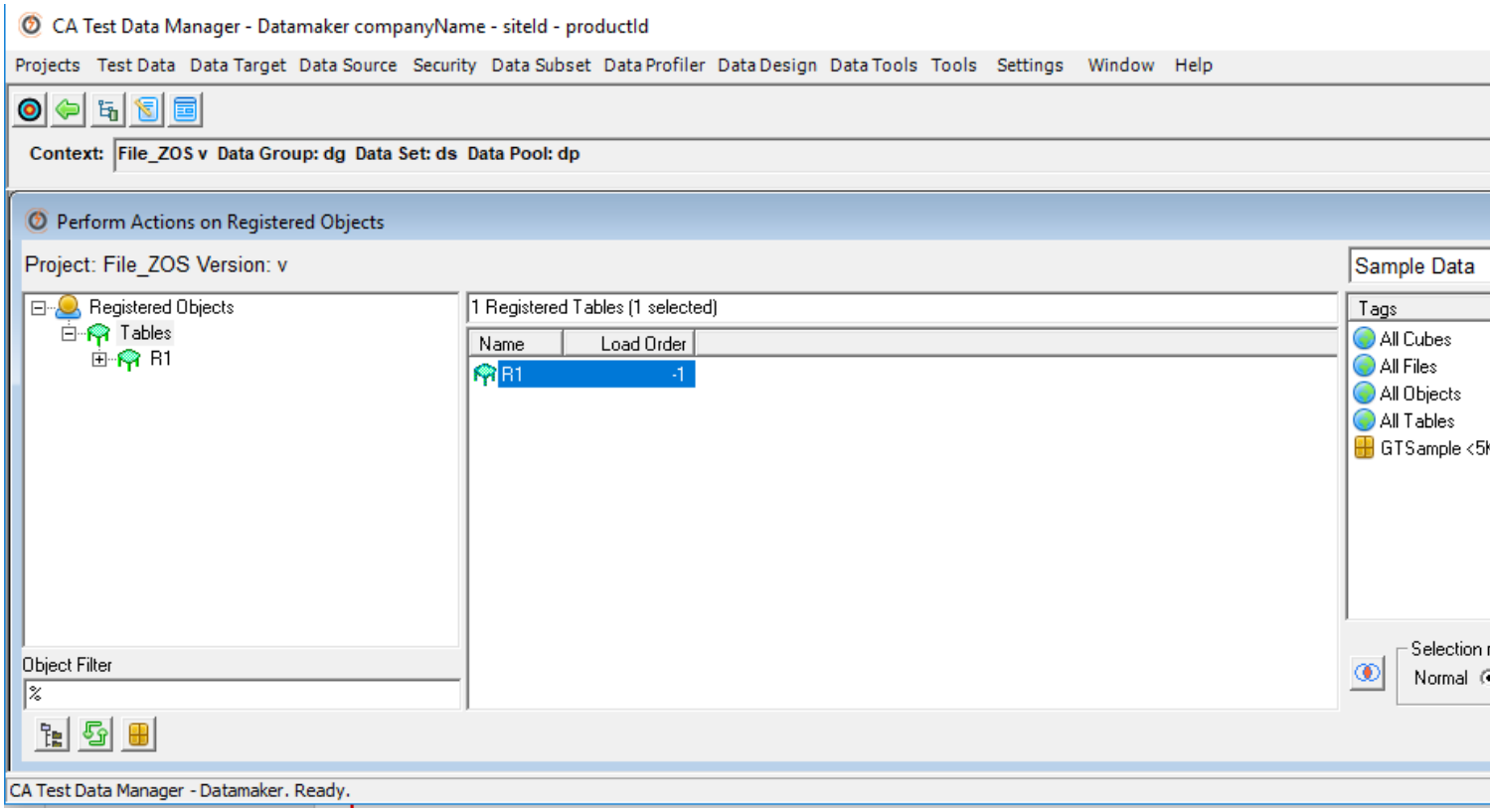
  Note: Use this parameter to override the default page limit.
  Values: a number
  Default: 50

Loading Profile Data into Datamaker

After you run the profiling job, transfer the output CSV to Windows.

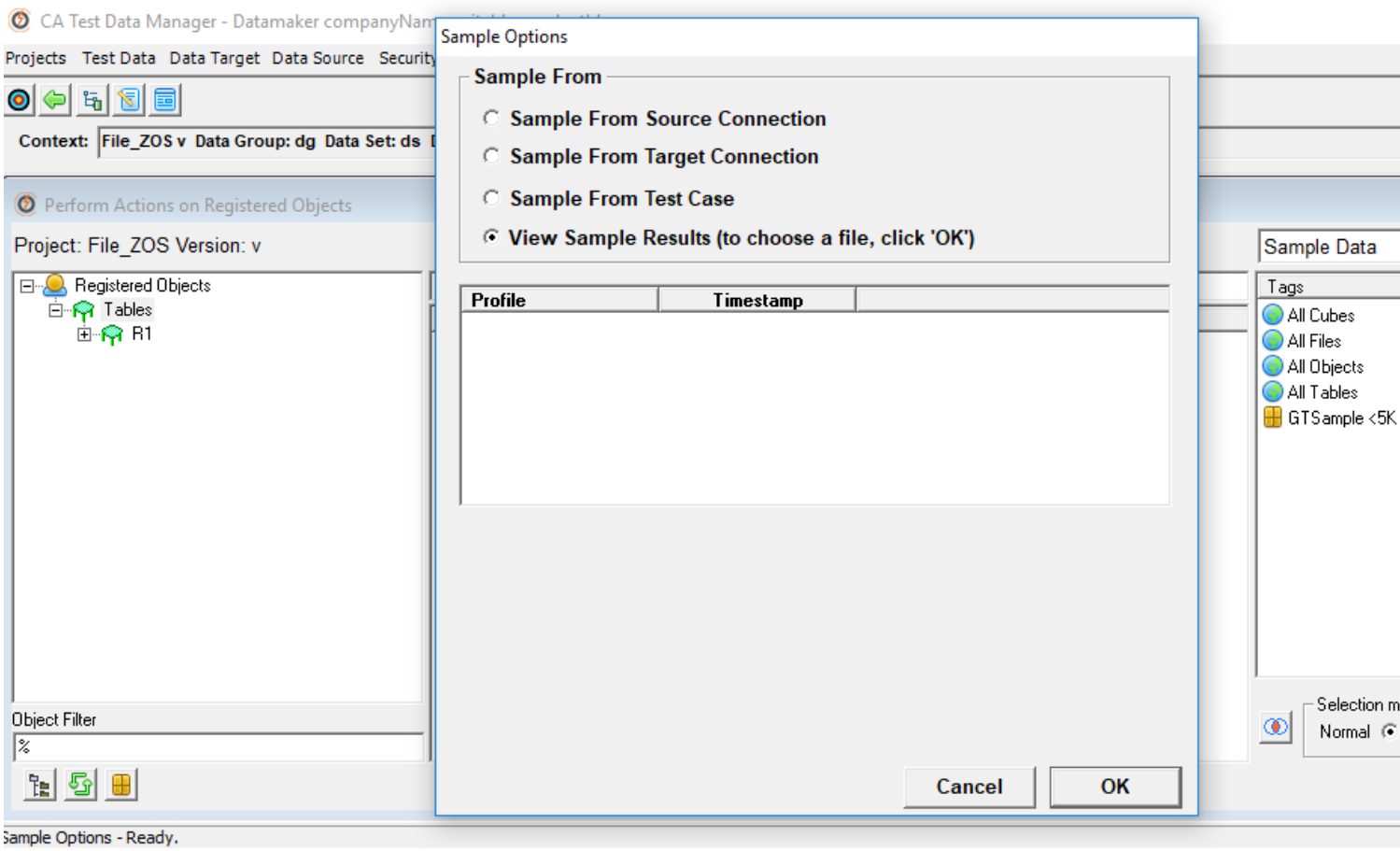
Follow these steps:

1. Open Datamaker.
2. Select **Project/Actions for Registered Objects**.
   The project/version context should be where the Advanced File Layout describes that the file was registered.
3. Highlight the registered objects that corresponds to the file.



4. Select Sample Data from the drop-down actions list.

   NOTE
   If prompted for source and target connection details, ignore the prompt window and close it.
5. Select the **View Sample Results** option, and click **OK** to load the sample data csv file.



Your sample data is now loaded.

Masking Files

In order to mask files you need an Advanced File Layout describing the file, and a Transformation Map giving the masking rules. The Advanced File Layout (suffixed _zOS.AFL.DM.txt) should be registered in Datamaker as an "Advanced File Layout from G-T text file", and it should also be transferred to the mainframe where it can be read by the masking job.

Add Seed Lists to DB2 zOS SeedList Tables

As a Tester, you have installed the TDM mainframe toolkit and use seed lists in the DB2 database for your z/OS installation. As you start working with different hash values or seed list entries, you find that you want to add new entries to the seed list.

You can edit the seed list that is part of the TDM repository in DataMaker in the main menu by clicking **Tools, Maintain Seed Data**. The **Maintain Seed List** dialog gives you a comprehensive overview of the lists that are used by Test Data Manager. The dialog displays the rows and columns included seed data, which covers domains such as countries, country codes, postal codes, persons' and street names, email providers, days of the week, currencies, credit cards types, and many more.

Prerequisites

The tables being used for the creation of extra seed list entries are based off the TDM repository kit 3.2.11 that became GA starting with TDM 4.5.

1. Install the TDM mainframe toolkit.
2. install the DB2 reference table. For more information, see Install DB2 Reference Data.

Create a New TDM DB2 z/OS Seed List

You want to add new seed list entries for DB2 for z/OS. You can add entries to the seed lists that are part of the TDM (GTREP) repository. You access the seed list editor through the GT Data Maker user interface.

Create schema and table

1. Create a DB2 z/OS schema, for example, GRIDT01.
2. Create a table in the DB2 z/OS schema, for example, "GTSRC_REFERENCE_LOV1".

Copy an Existing Seedlist

1. Launch GT DataMaker and select **Tools, Maintain Seed Data** to open the Seed Data Maintenance dialog.
2. Select an entry and all its corresponding entries.
3. Export the results as a CSV file.
4. Open the exported CSV file in Microsoft Excel.
5. Replace the single quote with a hash symbol (#). This step is necessary, if there are entries in the rows that contain a single quote.
6. Save the file as a spreadsheet in .xls or .xlsx format.

Prepare SQL Statements

1. Create a new sheet tab named "SQL".
2. Populate cells in the "SQL" tab to create insert statements for your SQL Server seed list table, starting from row 2.
3. Create a new sheet tab named "SQL-DB2".
4. Populate cells in the "SQL-DB2" tab to create insert statements for the SQL Server version of the gtsrc reference lov1 seed list table, starting from row 2.
5. Create a new sheet tab named "DB2".
6. Copy from "DB2" row 2 on down in the "DB2" tab.
7. Save the spreadsheet.

Add the Seedlist Entries

1. Save each tab to a corresponding CSV file. Name the files after the tab names.
2. Edit each of the CSV files, and replace the previously inserted hash sign `#` with two single quotes `''` .
3. Save the updated files as SQL files with .sql suffix.
4. Execute the insert statements in the GT Data Maker target to update the corresponding database type based off its database profile.
5. Update the available data functions where you will be adding the new hashlov, randlov, and seqlov functions that are associated with the new seed list.

You have added a new seed list entries for your DB2 for z/OS TDM mainframe toolkit installation.

Keep in mind that these new seed list entries are used by the TDM mainframe toolkit programs and JCL procedures.

Example: Adding a New Seed List Entry Set

In this example, you create a new seed list for cities in the State of Colorado, USA. Adapt this example to your use case.

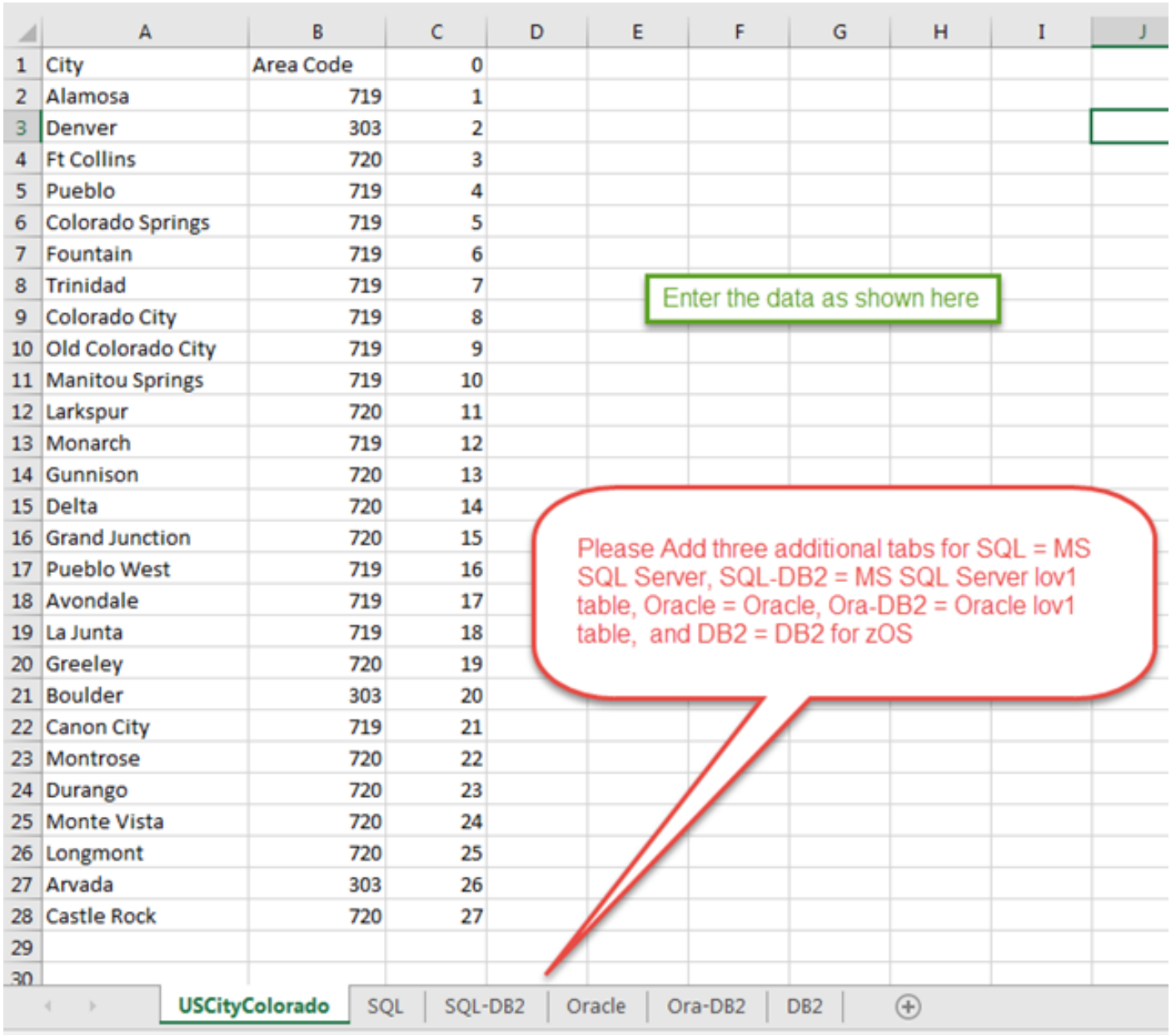
Copy an Existing Seedlist

1. Launch GT DataMaker.
2. Connect with a user name and password of an account that has permissions to work with the seed list editor. Typically, you use the Test Data Manager administrator or equivalent.
3. Connect Data Target and Data Source.
4. Click **Tools, Maintain Seed Data** to open the **Maintain Seed Data** dialog.
5. Look in the Seed Data Type pane for an existing seed list to use as a base for the new seed list.
   In this example, we choose the US City list.
6. Right-click the "US City" column title on the right-hand side, and choose **Export** from the context menu.
   The Export to CSV dialog opens.
7. Provide a name for this export file, for example, "USCityColorado.csv", and click **Save**.
8. Open the saved CSV file with Microsoft Excel.
9. Re-save the file as an Microsoft Excel Workbook in *.xlsx format.

Prepare SQL Statements

If you are using MS SQL Server as your repository. Enter the data for the MS SQL Server level repository. If your GT Data Maker repository is Oracle based, fill in the Oracle tabs instead.

1. Delete all the entries that were exported, and use the data in the following screenshot as example. Here we are creating a brand new seed list. In this example, we want a city, area code, and index number columns. These rows represent the entries:



	A	B	C
1	City	Area Code	0
2	Alamosa	719	1
3	Denver	303	2
4	Ft Collins	720	3
5	Pueblo	719	4
6	Colorado Springs	719	5
7	Fountain	719	6
8	Trinidad	719	7
9	Colorado City	719	8
10	Old Colorado City	719	9
11	Manitou Springs	719	10
12	Larkspur	720	11
13	Monarch	719	12
14	Gunnison	720	13
15	Delta	720	14
16	Grand Junction	720	15
17	Pueblo West	719	16
18	Avondale	719	17
19	La Junta	719	18
20	Greeley	720	19
21	Boulder	303	20
22	Canon City	719	21
23	Montrose	720	22
24	Durango	720	23
25	Monte Vista	720	24
26	Longmont	720	25
27	Arvada	303	26
28	Castle Rock	720	27
29			
30			

2. Create two more sheets named either SQL and SQL-DB2, or Oracle and Ora-DB2 respectively; and create one sheet named DB2.
3. (SQL Server only) Go to the SQL workbook. Enter the data for the MS SQL Server level repository.

```
Column A
Insert into Scramble.dbo.gtsrc_reference_data (rd_ref_id, rd_ref_value, rd_ref_value2, rd_index)
 values('US CITY COLORADO'
Column B
=CONCAT("'",USCityColorado!A2,"'")
Column C
=CONCAT("'",USCityColorado!B2,"'")
Column D
=CONCAT(USCityColorado!C2,");")
```

The result should like this screenshot:

1					
2	Insert into Scramble.dbo.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Alamosa'	'719'	1);	
3	Insert into Scramble.dbo.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Denver'	'303'	2);	
4	Insert into Scramble.dbo.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Ft Collins'	'720'	3);	
5	Insert into Scramble.dbo.gtsrc_reference_data (rd_re	s ('US City Colorado'	'Pueblo'	'719'	4);
6	Insert into Scramble.dbo.gtsrc_reference_data (rd_re	s ('US City Colorado'	'Colorado Springs	'719'	5);
7	Insert into Scramble.dbo.gtsrc_reference_data (rd_re	s ('US City Colorado'	'Fountain'	'719'	6);
8	Insert into Scramble.dbo.gtsrc_reference_data (rd_re	s ('US City Colorado'	'Trinidad'	'719'	7);
9	Insert into Scramble.dbo.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Colorado City'	'719'	8);	
10	Insert into Scramble.dbo.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Old Colorado Cit	'719'	9);	
11	Insert into Scramble.dbo.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Manitou Springs'	'719'	10);	
12	Insert into Scramble.dbo.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Larkspur'	'720'	11);	

Switch over the "SQL" tab and setup row 2 based on the values listed below. After you have done that, please copy row 2 down to row 28.

4. (SQL Server only) Switch to the "SQL-DB2" sheet, and enter the data as shown, on row 2. Then copy the entries down to row 28.

```
Column A
Insert into Scramble.dbo.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values
 ('US CITY COLORADO'
Column B
=CONCAT("'",USCityColorado!A2,"'")
Column C
=CONCAT("'",USCityColorado!B2,"'")
Column D
=USCityColorado!C2
Column E
17788);
```

The result should like this screenshot:

1					
2	Insert into Scramble.dbo.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Alamosa'	'719'	1	17788);
3	Insert into Scramble.dbo.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Denver'	'303'	2	17788);
4	Insert into Scramble.dbo.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Ft Collins'	'720'	3	17788);
5	Insert into Scramble.dbo.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Pueblo'	'719'	4	17788);
6	Insert into Scramble.dbo.gtsrc_reference_lov1 (rl_ref_id,	US City Colorado'	'Colorado S	'719'	5 17788);
7	Insert into Scramble.dbo.gtsrc_reference_lov1 (rl_ref_id,	US City Colorado'	'Fountain'	'719'	6 17788);
8	Insert into Scramble.dbo.gtsrc_reference_lov1 (rl_ref_id,	US City Colorado'	'Trinidad'	'719'	7 17788);
9	Insert into Scramble.dbo.gtsrc_reference_lov1 (rl_ref_id,	US City Colorado'	'Colorado C	'719'	8 17788);
10	Insert into Scramble.dbo.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Old Colora	'719'	9	17788);
11	Insert into Scramble.dbo.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Manitou Sp	'719'	10	17788);
12	Insert into Scramble.dbo.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Larkspur'	'720'	11	17788);
13	Insert into Scramble.dbo.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Monarch'	'719'	12	17788);

Switch over the "SQL-DB2" tab and setup row 2 based on the values listed below. After you have done that, please copy row 2 down to row 28.

5. (Oracle Server only) Switch to the "Oracle" sheet, and enter the data as shown, on row 2; then copy the entries down to row 28.

```
Column A
Insert into scramble.gtsrc_reference_data (rd_ref_id, rd_ref_value, rd_ref_value2, rd_index) values('US
 CITY COLORADO'
Column B
=CONCAT("'",USCityColorado!A2,"'")
Column C
=CONCAT("'",USCityColorado!B2,"'")
Column D
=CONCAT(USCityColorado!C2,");")
```

The result should like this screenshot:

2	Insert into scramble.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Alamosa'	'719'	1);
3	Insert into scramble.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Denver'	'303'	2);
4	Insert into scramble.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Ft Collins'	'720'	3);
5	Insert into scramble.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Pueblo'	'719'	4);
6	Insert into scramble.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Colorado Sp	'719'	5);
7	Insert into scramble.gtsrc_reference_data (rd_r	es ('US City Colorado'	'Fountain'	'719'	6);
8	Insert into scramble.gtsrc_reference_data (rd_r	es ('US City Colorado'	'Trinidad'	'719'	7);
9	Insert into scramble.gtsrc_reference_data (rd_r	es ('US City Colorado'	'Colorado Ci	'719'	8);
10	Insert into scramble.gtsrc_reference_data (rd_r	es ('US City Colorado'	'Old Colorac	'719'	9);
11	Insert into scramble.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Manitou Sp	'719'	10);
12	Insert into scramble.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Larkspur'	'720'	11);
13	Insert into scramble.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Monarch'	'719'	12);
14	Insert into scramble.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Gunnison'	'720'	13);
15	Insert into scramble.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Delta'	'720'	14);
16	Insert into scramble.gtsrc_reference_data (rd_ref_id,rd_ref_value,rd_ref_value2,rd_index) values ('US City Colorado'	'Grand Junct	'720'	15);

Switch over the "Oracle" tab and setup row 2 based on the values listed below. After you have done that, please copy row 2 down to row 28.

6. (Oracle Server only) Switch to the "Ora-DB2" sheet, and enter the data starting from row 2, then copy the entries to row 28.

```
Column A
Insert into scramble.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values
 ('US CITY COLORADO'
Column B
=CONCAT("'",USCityColorado!A2,"'")
Column C
=CONCAT("'",USCityColorado!B2,"'")
Column D
=USCityColorado!C2
Column E
17788);
```

The result should like this screenshot:

1					
2	Insert into scramble.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Alamosa'	'719'	1	17788);
3	Insert into scramble.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Denver'	'303'	2	17788);
4	Insert into scramble.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Ft Collins	'720'	3	17788);
5	Insert into scramble.gtsrc_reference_lov1 (rl_re	es ('US City Colorado'	'Pueblo'	'719'	4	17788);
6	Insert into scramble.gtsrc_reference_lov1 (rl_re	es ('US City Colorado'	'Colorado	'719'	5	17788);
7	Insert into scramble.gtsrc_reference_lov1 (rl_re	es ('US City Colorado'	'Fountain'	'719'	6	17788);
8	Insert into scramble.gtsrc_reference_lov1 (rl_re	es ('US City Colorado'	'Trinidad'	'719'	7	17788);
9	Insert into scramble.gtsrc_reference_lov1 (rl_re	es ('US City Colorado'	'Colorado	'719'	8	17788);
10	Insert into scramble.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Old Color	'719'	9	17788);
11	Insert into scramble.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Manitou S	'719'	10	17788);
12	Insert into scramble.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Larkspur'	'720'	11	17788);
13	Insert into scramble.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Monarch'	'719'	12	17788);
14	Insert into scramble.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Gunnison	'720'	13	17788);

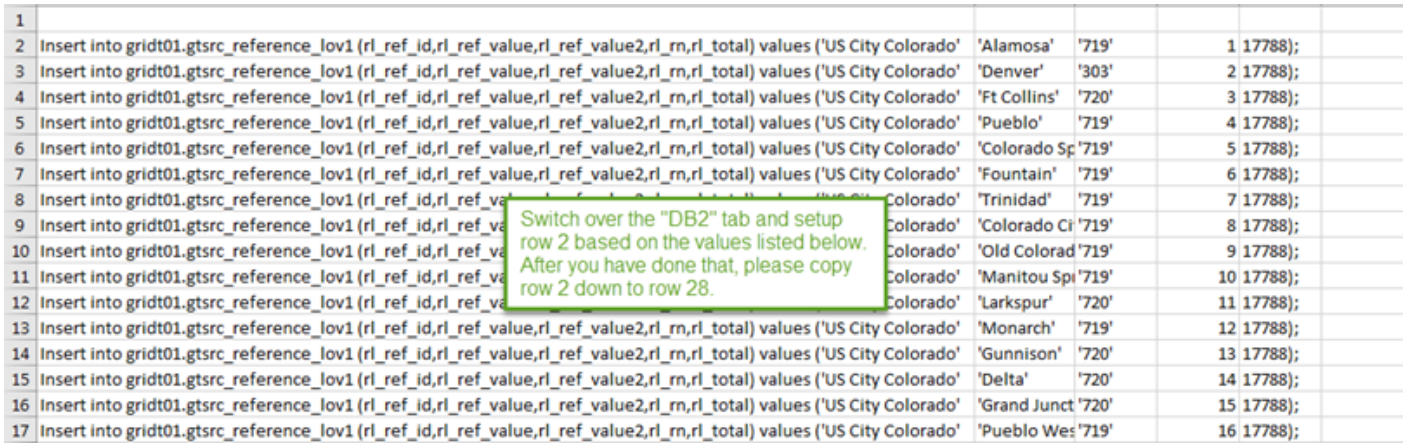
Switch over the "Ora-DB2" tab and setup row 2 based on the values listed below. After you have done that, please copy row 2 down to row 28.

7. Go to the DB2 sheet, and enter the data in each of the columns listed starting at row 2. Copy these entries up to row 28.
   Note: Change the schema name "gridt01" to the schema where your TDM MF tables were installed.

```
Column A
Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US
 CITY COLORADO'
Column B
=CONCAT("'",USCityColorado!A2,"'")
Column C
=CONCAT("'",USCityColorado!B2,"'")
Column D
```

```
=USCityColorado!C2
Column E
17788);
```

The result should like this screenshot:



1					
2	Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Alamosa'	'719'	1	17788);
3	Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Denver'	'303'	2	17788);
4	Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Ft Collins'	'720'	3	17788);
5	Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Pueblo'	'719'	4	17788);
6	Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Colorado Sp	'719'	5	17788);
7	Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Fountain'	'719'	6	17788);
8	Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_va	Colorado'	'Trinidad'	'719'	7	17788);
9	Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_va	Colorado'	'Colorado Ci	'719'	8	17788);
10	Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_va	Colorado'	'Old Colorad	'719'	9	17788);
11	Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_va	Colorado'	'Manitou Spi	'719'	10	17788);
12	Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_va	Colorado'	'Larkspur'	'720'	11	17788);
13	Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Monarch'	'719'	12	17788);
14	Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Gunnison'	'720'	13	17788);
15	Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Delta'	'720'	14	17788);
16	Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Grand Junct	'720'	15	17788);
17	Insert into gridt01.gtsrc_reference_lov1 (rl_ref_id,rl_ref_value,rl_ref_value2,rl_rn,rl_total) values ('US City Colorado'	'Pueblo Wes	'719'	16	17788);

8. Save all your changes to the sheets.
9. Verify that you have saved the spreadsheet after all the changes that you have made.
10. Switch to the "SQL" tab and save the "SQL" sheet in CSV-UTF-8 format under the name `USCityColorado-SQLServer.csv`.
11. Repeat this process and save the remaining sheets in this format.
    Now you have three files with the following names:
    - (SQL Server only) `USCityColorado-SQLServer.csv`
    - (SQL Server only) `USCityColorado-SQLServer-DB2-LOV1.csv`
    - (Oracle only) `USCityColorado-Oracle.csv`
    - (Oracle only) `USCityColorado-Oracle-DB2-LOV1.csv`
    - USCityColorado-DB2
12. Close the spreadsheet, but don't save any additional changes.

Enable the SQL statements

1. Open Windows Explorer.
2. Rename the three files and change their .csv suffix to an .sql suffix.
3. Open each of the SQL files in a plain-text editor.
4. Perform a global search and replace: Replace the double quotes by an empty string to delete the double quotes.
   The SQL statements are now no longer commented out.
5. Save your changes and close the text editor.

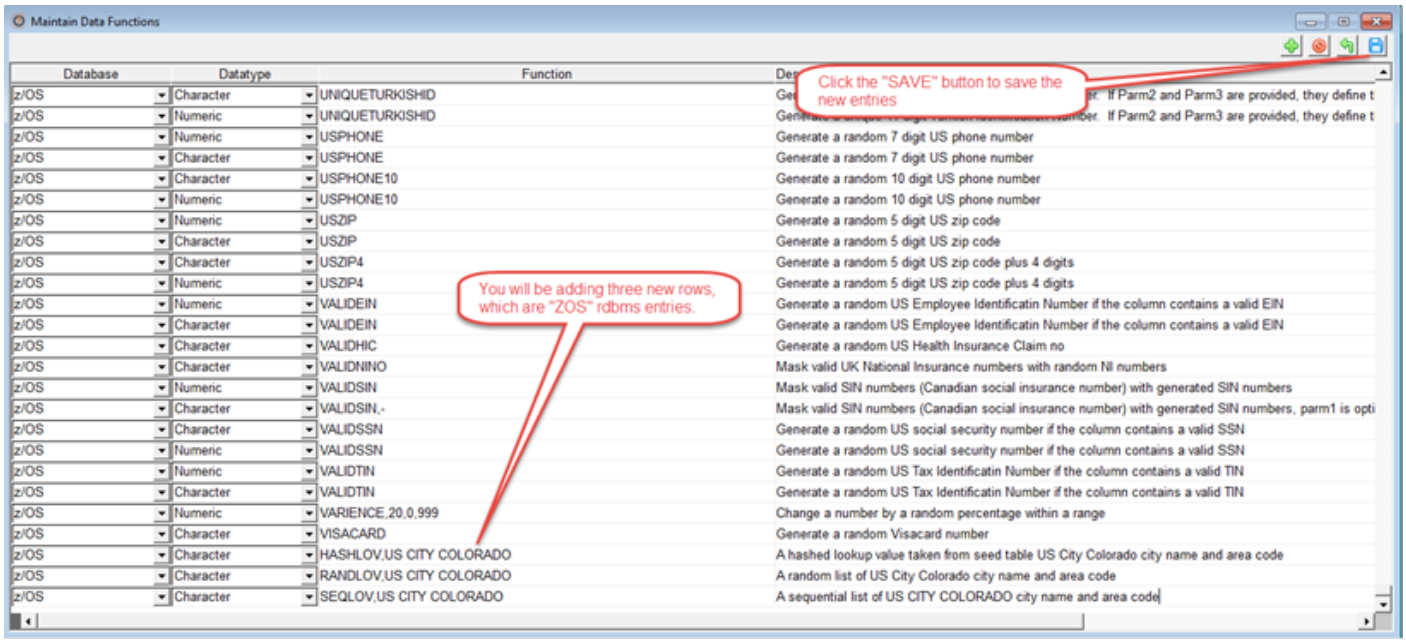
Add the Seedlist Entries

1. Launch GT DataMaker. Perform one of the following procedures:
   - SQL Server only:
     a. Connect to the MS SQL Server SQL window, where the GT rep and scramble databases have been installed.
     b. Execute the SQLServer and SQLServer-DB2-LOV1 scripts to add the new seed list.
   - Oracle only:
     a. Connect to the Oracle data source where the scramble database has been installed.
     b. Execute the Oracle and Oracle-DB2-LOV1 scripts to add the new seed list.
2. Open an SQL window to the DB2 for z/OS subsystem.

3. Execute the DB2 SQL script that adds the additional seed list to the gtsrc_reference_lov1 table.
4. Click Save.

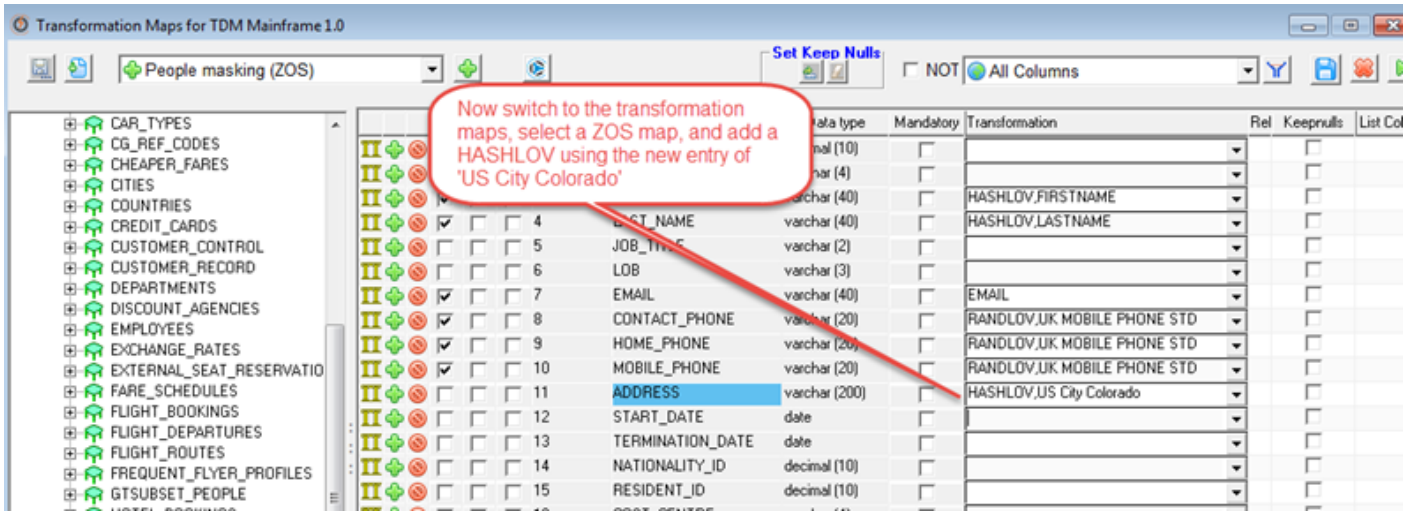
Add Data Functions

1. Click **Tools, Maintain Data Functions**.
2. Add three new rows for each of the available *LOV functions.
   - `HASHLOV,US CITY COLORADO`
     A hashed lookup value taken from seed table US City Colorado city name and area code
   - `RANDLOV,US CITY COLORADO`
     A random list of US City Colorado city name and area code
   - `SEQLOV,US CITY COLORADO`
     A sequential list of US CITY COLORADO city name and area code
3. Bring up the data functions dialog, so you can add the *LOV functions for the newly created seed list.



Database	Datatype	Function	Description
z/OS	Character	UNIQUETURKISHID	Ge… If Parm2 and Parm3 are provided, they define t
z/OS	Numeric	UNIQUETURKISHID	Gen… ber. If Parm2 and Parm3 are provided, they define t
z/OS	Numeric	USPHONE	Generate a random 7 digit US phone number
z/OS	Character	USPHONE	Generate a random 7 digit US phone number
z/OS	Character	USPHONE10	Generate a random 10 digit US phone number
z/OS	Numeric	USPHONE10	Generate a random 10 digit US phone number
z/OS	Numeric	USZIP	Generate a random 5 digit US zip code
z/OS	Character	USZIP	Generate a random 5 digit US zip code
z/OS	Character	USZIP4	Generate a random 5 digit US zip code plus 4 digits
z/OS	Numeric	USZIP4	Generate a random 5 digit US zip code plus 4 digits
z/OS	Numeric	VALIDEIN	Generate a random US Employee Identificatin Number if the column contains a valid EIN
z/OS	Character	VALIDEIN	Generate a random US Employee Identificatin Number if the column contains a valid EIN
z/OS	Character	VALIDHIC	Generate a random US Health Insurance Claim no
z/OS	Character	VALIDNINO	Mask valid UK National Insurance numbers with random NI numbers
z/OS	Numeric	VALIDSIN	Mask valid SIN numbers (Canadian social insurance number) with generated SIN numbers
z/OS	Character	VALIDSIN,-	Mask valid SIN numbers (Canadian social insurance number) with generated SIN numbers, parm1 is opti
z/OS	Character	VALIDSSN	Generate a random US social security number if the column contains a valid SSN
z/OS	Numeric	VALIDSSN	Generate a random US social security number if the column contains a valid SSN
z/OS	Numeric	VALIDTIN	Generate a random US Tax Identificatin Number if the column contains a valid TIN
z/OS	Character	VALIDTIN	Generate a random US Tax Identificatin Number if the column contains a valid TIN
z/OS	Numeric	VARIENCE,20,0,999	Change a number by a random percentage within a range
z/OS	Character	VISACARD	Generate a random Visacard number
z/OS	Character	HASHLOV,US CITY COLORADO	A hashed lookup value taken from seed table US City Colorado city name and area code
z/OS	Character	RANDLOV,US CITY COLORADO	A random list of US City Colorado city name and area code
z/OS	Character	SEQLOV,US CITY COLORADO	A sequential list of US CITY COLORADO city name and area code

4. Update the gtsrc_reference_* tables.
5. Create a ZOS transformation map that includes the new seed list. In this scenario, we are performing a HASHLOV using the ‘US City Colorado’ seed list.

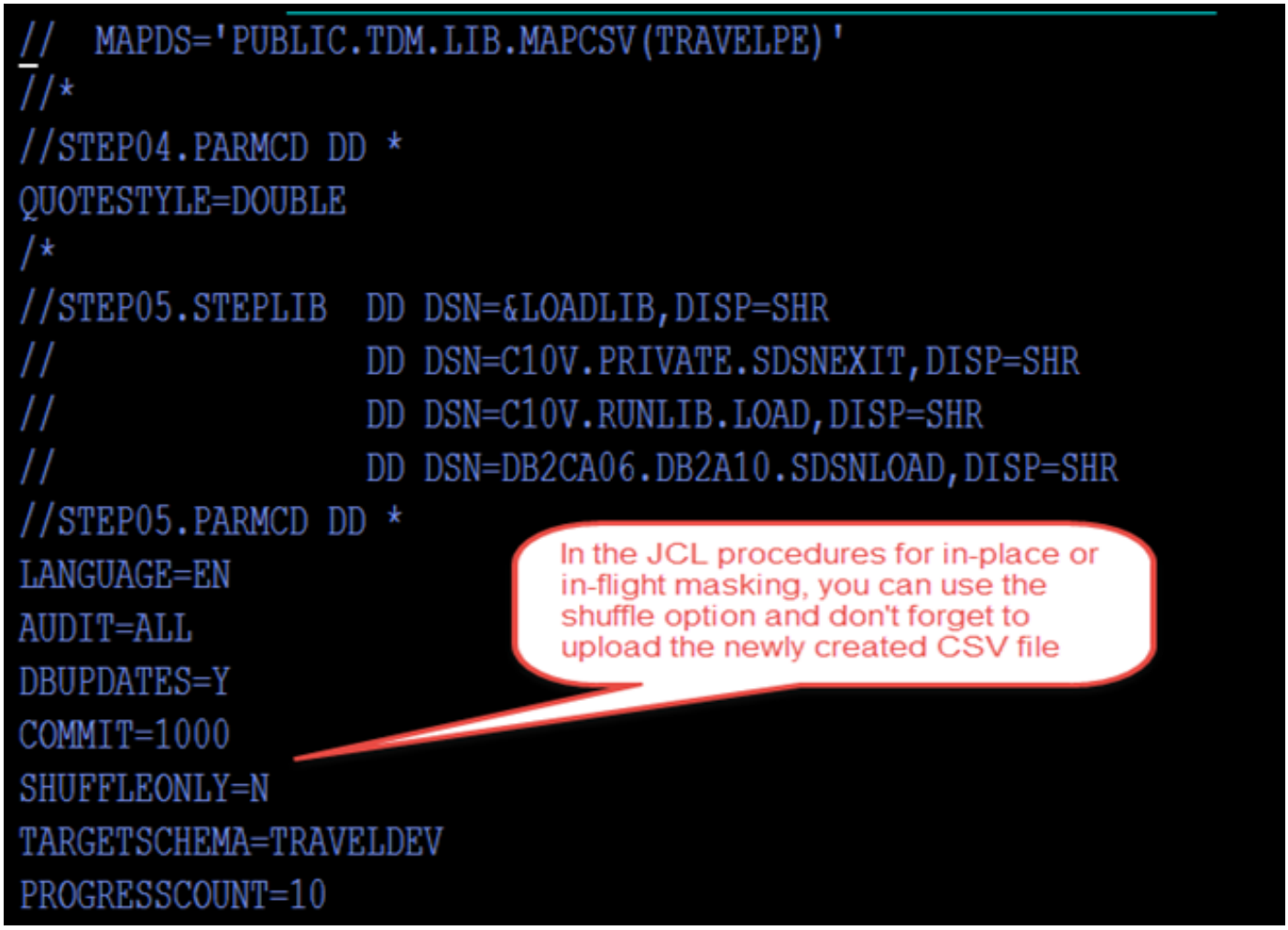


6. The first reference column for this seed list is used for the masking effort.
7. Save the ZOS version of the transformation map, and verify that the entry for the new seed list is there.

```
"Table","Column","Function","Parm1","Parm2","Parm3","Parm4","KeepNulls","Dateformat","Cross
PEOPLE,FIRST_NAME,HASHLOV,FIRSTNAME,,,,N,,,,,,,,
PEOPLE,LAST_NAME,HASHLOV,LASTNAME,,,,N,,,,,,,,
PEOPLE,EMAIL,EMAIL,,,,,N,,,,,,,,
PEOPLE,CONTACT_PHONE,RANDLOV,UK MOBILE PHONE STD,,,,N,,,,,,,,
PEOPLE,HOME_PHONE,RANDLOV,UK MOBILE PHONE STD,,,,N,,,,,,,,
PEOPLE,MOBILE_PHONE,RANDLOV,UK MOBILE PHONE STD,,,,N,,,,,,,,
PEOPLE,ADDRESS,HASHLOV,US City Colorado,,,,N,,,,,,,,
```

In the generated CSV file, you will see the new seedlist entry ready

8. Upload this csv file into the mainframe and follow the standard procedures to include this transformation map for in-place or in-flight masking.



Best Practices

The following best practices will help you in being successful in masking DB2 datasets.

DB2 Authorizations

Make sure that you have sufficient rights to the DB2 schemas (read/write/alter authorizations).

Make sure that you have set up DB2 Connect and tested this connection from the system where CA TDM is installed. Add an ODBC entry to TDM that points to the DB2 subsystem in the mainframe.

Planning

When you create a new seed list:

- We recommend that you look at the shipped seed lists and use a seed list that is very close to the type of seed list that you want to add.
- Create a new rl_total number for the new seed list!

When you update an existing seed list:

- Plan the additional entries that you want to add to an existing list.
- Update the rl_total number!

Testing

Before rolling out your new seed list into production, make test runs to make sure that the seed list performs the masking using the correct values.

NOTE

More information:

- Install DB2 Reference Data
- Mask Files (Using Seedlists Stored in DB2)
- Create Seed Data from a Cube
- Seed Lists
- Propagate Seed List Data Across Masking Engines

Create File Transformation Maps - Masking

The steps required to create and save Transformation Maps for flat files is the same as for DB2 targets. For more information see, Create DB2 Transformation Maps.

Executing Masking (Flat File sources)

Inputs to the file masking are the Advanced File Layout describing the file, the Transformation Map CSV containing the masking rules, and the file to be masked. The job allocates and writes to a new file with the masked data. The masked data file is a sequential file, if the input to the job is a VSAM cluster then the masked data file should be used to populate a new VSAM file using the IDCAMS utility.

Mask Files (Using Seedlists Stored in DB2)

To run flat file masking, JCL is supplied in the installation package as GTXMSKF. This job uses procedure GTMSKF.

For VSAM file masking the JCL is supplied as GTXMSKFV. This job uses procedure GTMSKFV.

The following are only differences between the jobs for masking sequential files and VSAM files:

- **Sequential files** - The output dataset that contains the masked data is allocated with the same DCB parameters as the input dataset
- **VSAM files** - The JCL procedure includes parameters for the LRECL, RECFM, and BLKSIZE of the output dataset as noted below.

You can supply the following parameters to the JCL procedure:

- **LOADLIB**
  Names the load library that contains programs TDMXDF, TDMXMP and TDMXMKF
- **MSGDS**
  TDM messaging file
- **INDS**
  Names the file to be masked
- **MAPDS**
  Names the dataset that contains the mapping CSV (masking rules)
- **DEFFDS**

Names the dataset that contains the record definition CSV (Advanced File Layout)

- **REPHLQ**
  Gives the high-level dataset name qualifier to be used for the audit and report files
- **AUDPR**
  The primary space allocation (in cylinders) for the audit report
- **AUDSEC**
  The secondary space allocation (in cylinders) for the audit report
- **INDSPR**
  The primary space allocation (in cylinders) for the output masked file (default 1)
- **INDSSEC**
  The secondary space allocation (in cylinders) for the output masked file (default 1)

To mask VSAM files, also supply the following parameters:

- **LRECL**
  The logical record length of the output masked file
- **RECFM**
  The record format of the output masked file
- **BLKSIZE**
  The block size of the output masked file

The masking job contains the following steps:

- Runs IEFBR14 to delete and define the report, audit files, and output masked files.

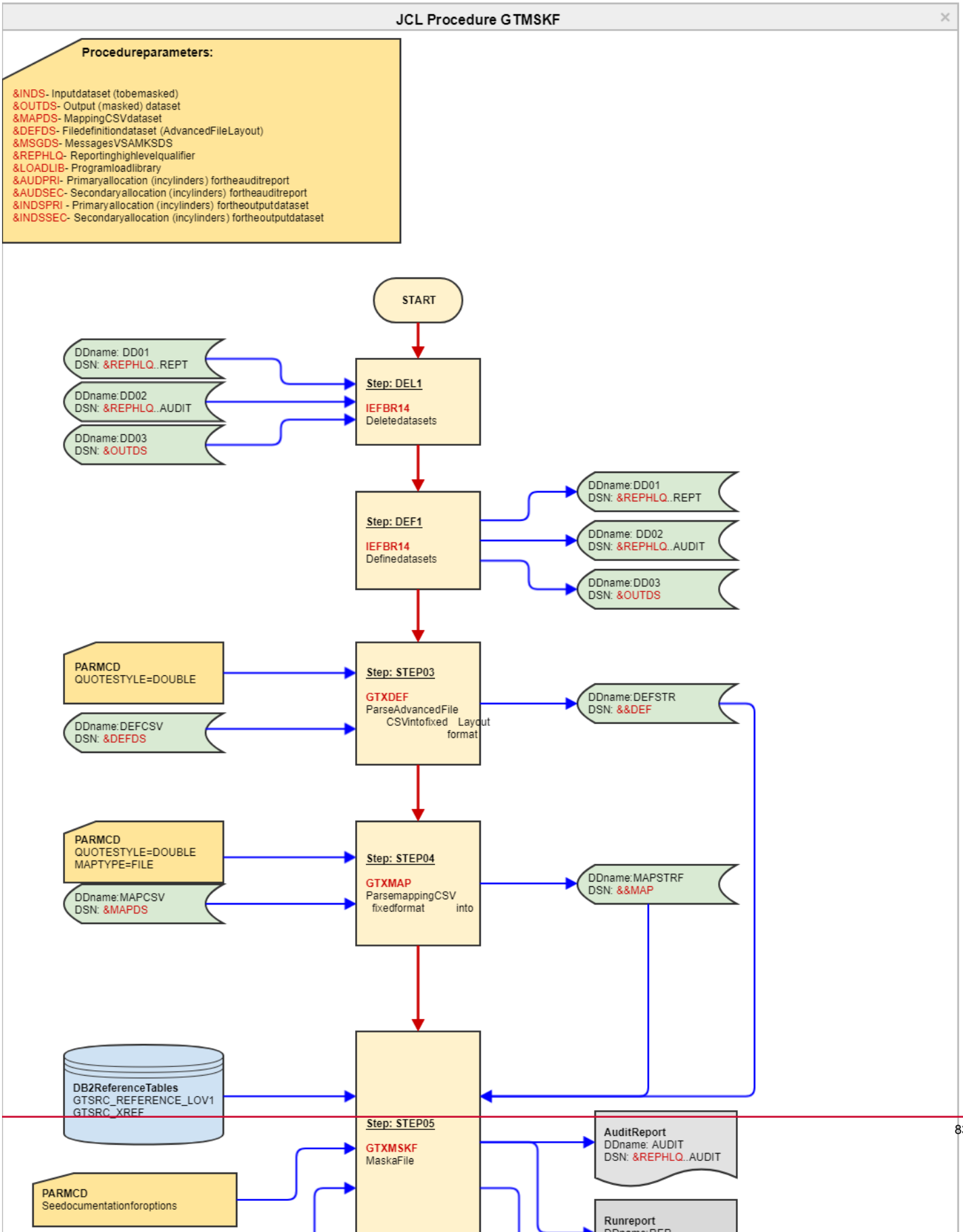
  NOTE

  The report file is allocated with SPACE=(CYL,(1,1)) which should be sufficient for most runs. If many error or warning messages are produced, you might need to increase the space allocation.

  For GTXMSKF the output masked file uses the same DCB information as the input file. The dataset name is the input file name that is suffixed with ".MASKED".
- Runs GTXDEF to read and parse the record definition CSV, and write the CSV out to a fixed record format file. See the GTXDEF Parameters.
- Runs GTXMAP to read and parse the mapping CSV, and write the CSV out to a fixed record format file. See the section GTXMAP Parameters.
- Runs GTXMSKF to apply the masking/subsetting rules that are specified in the mapping CSV. See the section GTXMSKF Parameters

GTXMSKF Flow Diagram

Figure 52: gtxmskf



GTXMSKF Parameters

GTXMSKF can run with DB2 (the default) or VSAM lookup tables. To specify the lookup tables, use the VSAMLOOKUP parameter below.

Auditing

The audit file contains the following information:

- Record name
- Record sequence within the input file for a record that is being masked
- Name of the field that is being masked, and the old and new values for that field.

  NOTE
  No audit data is produced by default.

- **AUDIT=ALL**
  Specifies all masked fields are detailed in the audit report.
- **AUDIT=ROW*nnn***
  Specifies the number of records to be audited.
  Values: A positive number
  Example: ROW100 will audit the first 100 records.
- **AUDIT=SAMPLE*nnn***
  Specifies the interval at which records are audited
  Values: A positive number
  Example: SAMPLE100 will audit every 100th record
- **PAGELIMIT=nnn**
  Specifies the audit output page limit.
  Values: A positive number that indicates the page limit.
  Default: 50
  Note: Default is regardless of the audit settings. To override the default, specify the number of audit pages to produce as *nnn*.

Cross-Reference

- **CASEINSENSITIVEXREF=**
  Specifies that cross-referencing is done regardless of case
  Values: N, Y
  Default: N
- **TRIMMEDXREF=**
  Specifies that values are trimmed before cross-referencing
  Values: N, Y
  Default: N

Dates

- **BADDATESTRING=*ccyymmdd***
  Specifies the date to replace unparseable dates for date functions.
  Default: Unparseable dates are masked
- **BASECENTURY=*nn***

Specifies how BASECENTURY indicates the starting point for the century digit if the record definitions contain dateformats with a single digit century
Example: BASECENTURY=19 and a dateformat of "CYYMMDD","1880729" is interpreted as 29th July 1988.

- **CDATE=*ccyymmdd***
  Specifies to override the current date for the purposes of date calculation functions.
- **HIGHDATE=*ccyymmdd***
  Specifies to override the highest date that offset date functions will process.
- **LOWDATE=*ccyymmdd***
  Specifies to override the lowest date that offset date functions will process

Shuffling

Shuffling happens when the SHUFFLE function is specified in a transformation map. Shuffling is a two-phase process, in the first phase, field values for which SHUFFLE is specified are used to populate a seedlist in GTSRC_REFERENCE_LOV1, in the second phase SHUFFLE functions are converted to SEQLOV functions which refer to the seedlist just created. Setting SHUFFLEONLY=Y will result in just the first phase of the shuffle being executed, that is, a seedlist will be created but will not be used to update field values.

- **SHUFFLEDISTINCT=**
  Specifies shuffle values that are created
  Values: N (creates all values in the shuffle), Y (creates distinct values for the shuffle)
  Default: N
- **SHUFFLELIMIT=nnn**
  Specifies to only select nnn values for the shuffle.
- **SHUFFLEONLY=**
  Specifies to update GTSRC_REFERENCE_LOV1 with shuffle values and not write to the output file
  Values: N, Y.
  Default: N

Other

- **BLANKSASNULLS=**
  Specifies to treat fields that contain blanks as null.
  Values: N, Y
  Default: N

  NOTE

  Where the mapping CSV includes the Keepnulls=Y option, blank fields are retained.
- **CASEINSENSITIVEFORMATENCRYPT=**Ignores case of input data.
  Values: Y, N
- **CASEINSENSITIVESEED=**
  Specifies if RANDLOV1, SEQLOV1, and HASHLOV1 seed value lookup is case-sensitive.
  Values: N, Y
  Default: N
- **CASEINSENSITIVEHASHLOV=**
  Specifies if the value that is hashed by functions HASHLOV and HASHLOV1 is converted to upper case before hashing
  Values: N, Y
  Default: Y
- **COMMIT=*nnn***

Specifies the commit frequency for updates to the cross-reference or seed tables
Values: 1000, nnn
Default: 50000

- **HASHTYPE=**
  Sets the hashing algorithm to use with the HASHLOV function.
  Values: ASM, JAVA
  Default: ASM

  NOTE

  If JAVA is specified, zOS will produce the same hash value as FDM (for consistent lookups).

- **KEEPINVALID=**
  Specifies whether numeric or date fields with invalid data are retained unchanged in the output file
  Values: N, Y
  Default: N
- **LANGUAGE=*LC***
  Specifies the two-character language code used for output messages.
  Values: EN, DE, ES, IT
  Default: EN
- **LOWERCASEKEY=**(Optional) A key to be used in encrypting lower case letters.
- **NUMERICKEY=**(Optional) A key to be used in encrypting numbers.
- **PROCESSCOUNT=*nnn***
  Specifies the maximum number of records in the file to be processed

  NOTE

  : This parameter is not the maximum number of records to be read.

- **REPORTINVALID=**
  Specifies if numeric or date fields with invalid data are reported
  Values: N, Y
  Default: N
- **TRIMMEDHASHLOV=**
  Specifies if the value that is hashed by functions HASHLOV and HASHLOV1 have the leading and trailing blanks trimmed before hashing. Required (Y) for consistent masking with FDM.
  Values: N, Y
  Default: N
- **UPPERCASEKEY=**
  (Optional) A key to be used in encrypting upper case letters.
- **VALIDATEONLY=**
  Specifies if the input mapping CSV and parameters files are validated and any errors are reported
  Values: N, Y
  Default: N

  NOTE

  The rules that are specified in the mapping CSV are not applied to the input file.

- **VSAMLOOKUP=**
  Specifies if all lookups, XREFs, and Subsetting (using SUBSETLIST) are performed using VSAM datasets.
  Values: N, Y
  Default: N (Lookups are performed using DB2 tables)
- **WHEREASSUBSET=**
  Specifies how WHERE functions affect the masking functions

Values: N (WHERE functions that are specified in the mapping CSV restrict the application of the masking functions to those records that satisfy the WHERE condition), Y (a subset of the input file is written that contains only those records that satisfy WHERE conditions)
Default: N

Mask Files (Using Seedlists Stored in VSAM)

To run flat file masking, JCL is supplied in the installation package as GTXMSKVS. This job uses procedure GTMSKVS.

NOTE

If the output file you want to create has different DCB parameters to the input file you want to mask, you can uncomment the following line in the JCL file GTXMSKVS to override the input file's DCB parameters, with the appropriate values for your output file:

```
// *DEF1.DD03 DD DCB=(LRECL=XXX,BLKSIZE=XXXX,RECFM=XX)
```

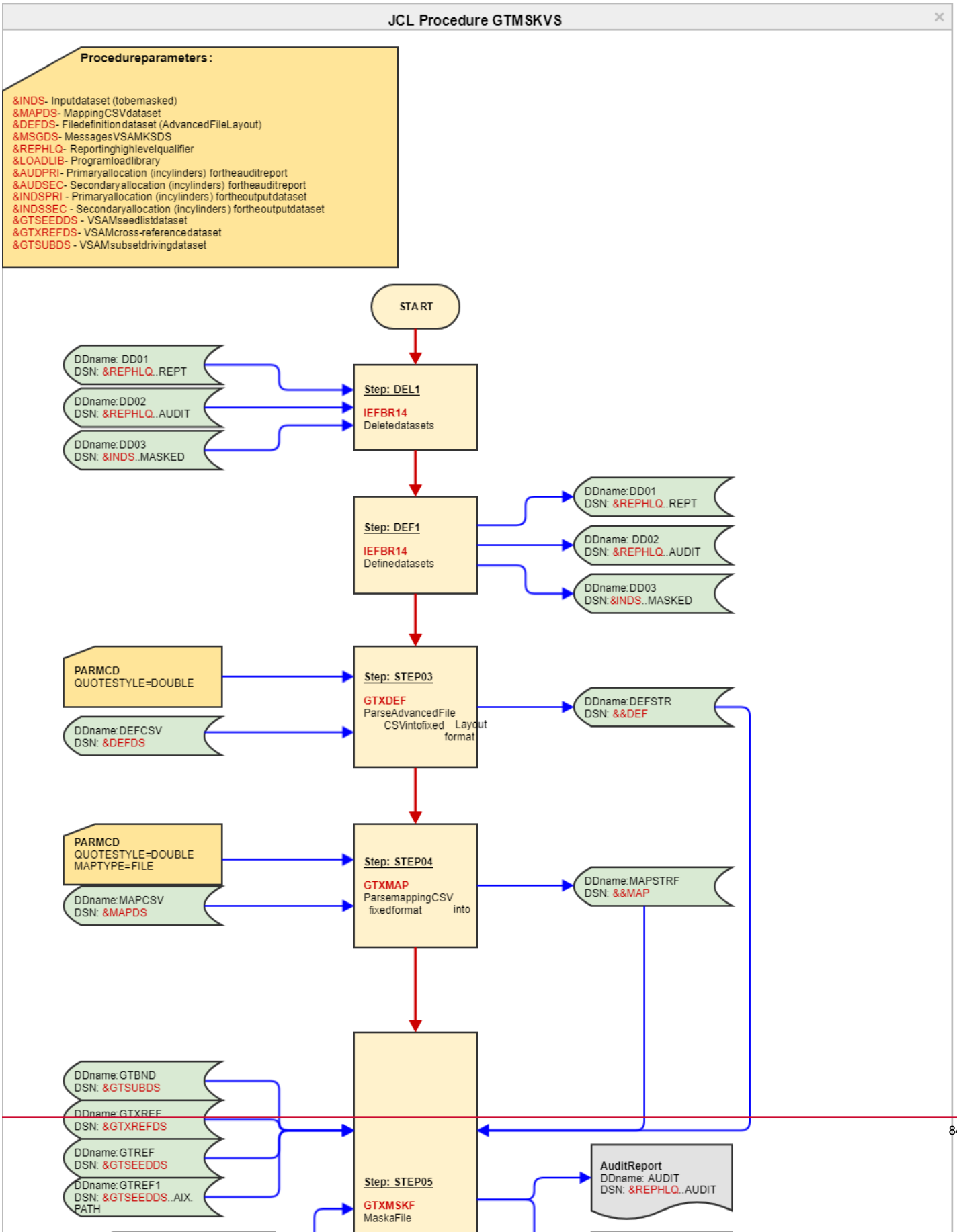
In the PARMCD input for program GTXMSKF, in this job parameter VSAMLOOKUP should be set to Y.

Parameters to the JCL procedure are the same as for GTMSKF procedure, see Mask Files (Using Seedlists Stored in DB2), but in addition you should supply the following parameters:

- **GTSEEDDS**
  The VSAM seedlist cluster name.
- **GTSUBDS**
  The VSAM subset driving cluster name.
- **GTXREFDS**
  The VSAM cross-reference cluster name.

GTXMSKVS Flow Diagram

Figure 53: GTXMSKVS



Subsetting Files

To subset files you need to create a transformation map with WHERE clauses and run the file masking job with the PARMCD option WHEREASSUBSET=Y. When the masking program runs, only records which satisfy the WHERE clauses supplied are written to the output file. You can apply masking at the same time as subsetting but note that if you need masking to be conditional, based on WHERE clauses, this does not work. In this case, run the file masking job twice, once to subset the file, and then to mask the subsetted file.

NOTE

The WHERE clauses you can use with files are limited in their syntax, they are not equivalent to SQL where clauses. See Mask Flat Files Using WHERE Clauses.

Generate Synthetic Mainframe File Data

There are a number of challenges with generating synthetic data for mainframe files:

- **Mainframe specific data types without LUW equivalents**
  For example packed decimal or binary numerics: for generation to a file, a conversion process may be necessary to format numerics correctly.
- **Redefinitions**
  If part of a record is defined in multiple ways, then for data generation you need to choose which definition to use.
- **Arrays**
  You cannot register an array object in TDM, for synthetic data generation an array should be represented by multiple rows in a table.
- **Large numbers of fields**
  Some file records contain hundreds or thousands of fields which can be difficult to manage in generating data.

There are two different ways you can generate synthetic file data:

- Register and generate data for a G-T Excel File Definition and then "publish to file (FD)". See Generate Synthetic Mainframe File Data using File Definitions.
- Define DB2 tables to hold file data, register and generate to these tables and then run a mainframe batch job to create a file from the tables. See Generate Synthetic Mainframe File Data using DB2 Tables.

Data Generation with File definitions

Using file definitions means that:

- a file conversion process is required to format the published file correctly.
- the G-T Excel file definition created by the File Definition Manager contains separate tabs for each redefined/redefining data item.
- the G-T Excel file definition created by the File Definition Manager contains separate tabs for each array.
- in the File Definition Manager you can split data items into separate tabs if there is a large number of fields.

Data Generation with DB2 Tables

Using DB2 tables means that:

- no file conversion is required
- redefined/redefining data item columns can be set to null which means that separate tables are not required to handle these
- separate tables are used to hold array data
- in defining the tables you can spit data items into separate tables to cater for large numbers of fields.

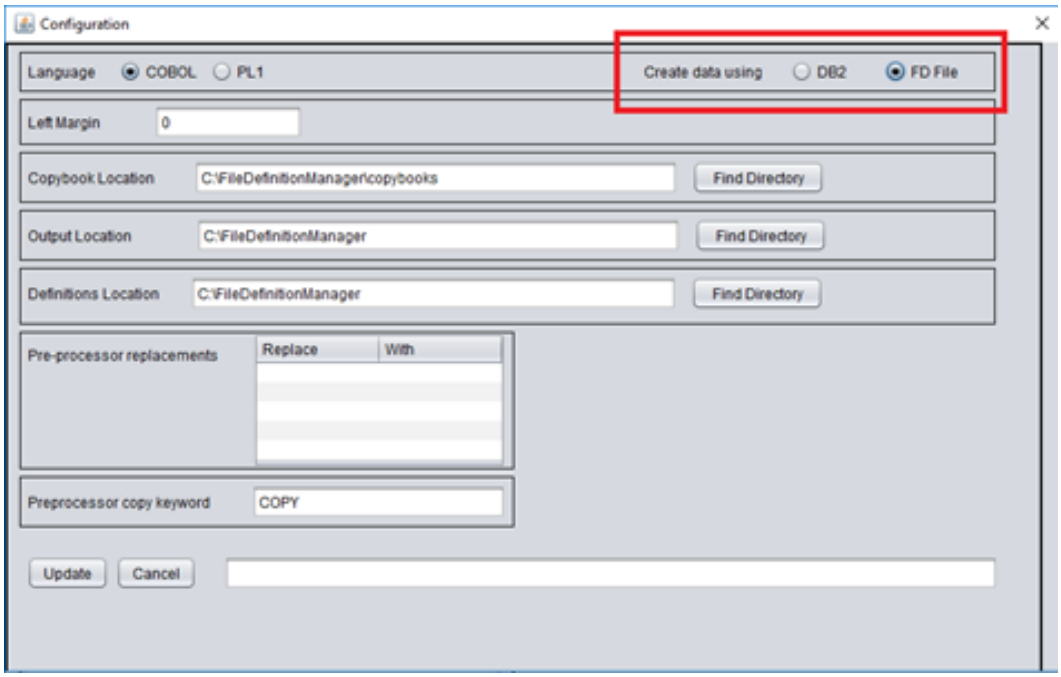
With the exception of simple record layouts, we recommend the use of DB2 tables for data generation.

Generate Synthetic Mainframe File Data using File Definitions

This page gives an overview of synthetic mainframe file data generation with file definitions.

File Definition Manager Use

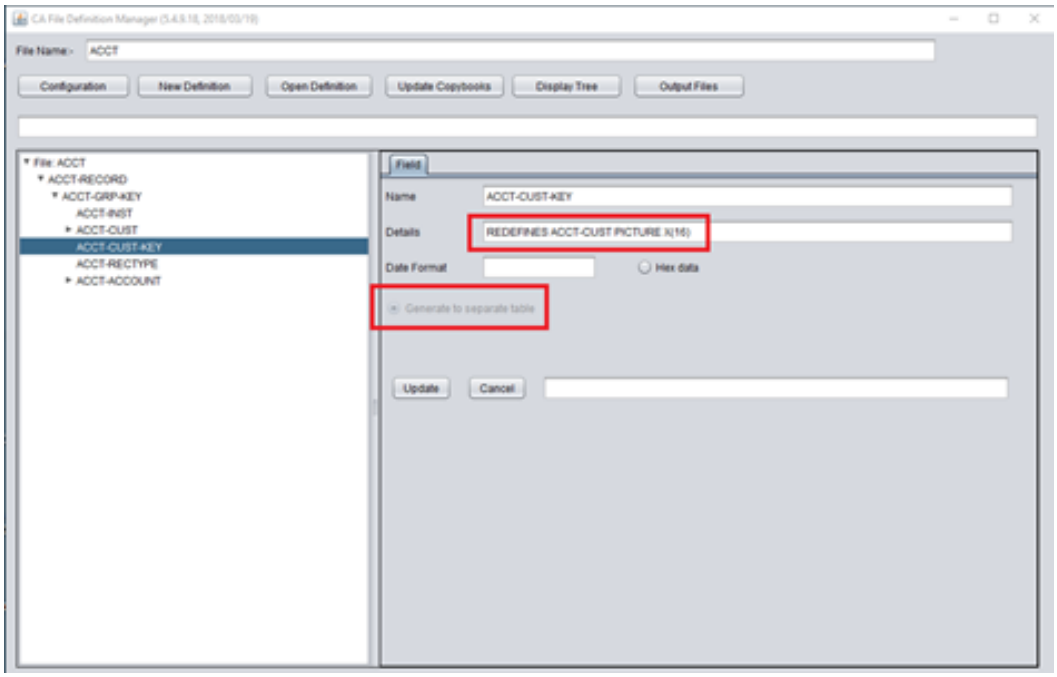
In your File Definition Manager configuration, you should specify that you intend to create data using a file definition.



To allow you to generate data for array elements, and to allow you choose between different definitions of the same part of a record the File Definition Manager will automatically mark array data items and redefined or redefining data items as belonging to separate tables.

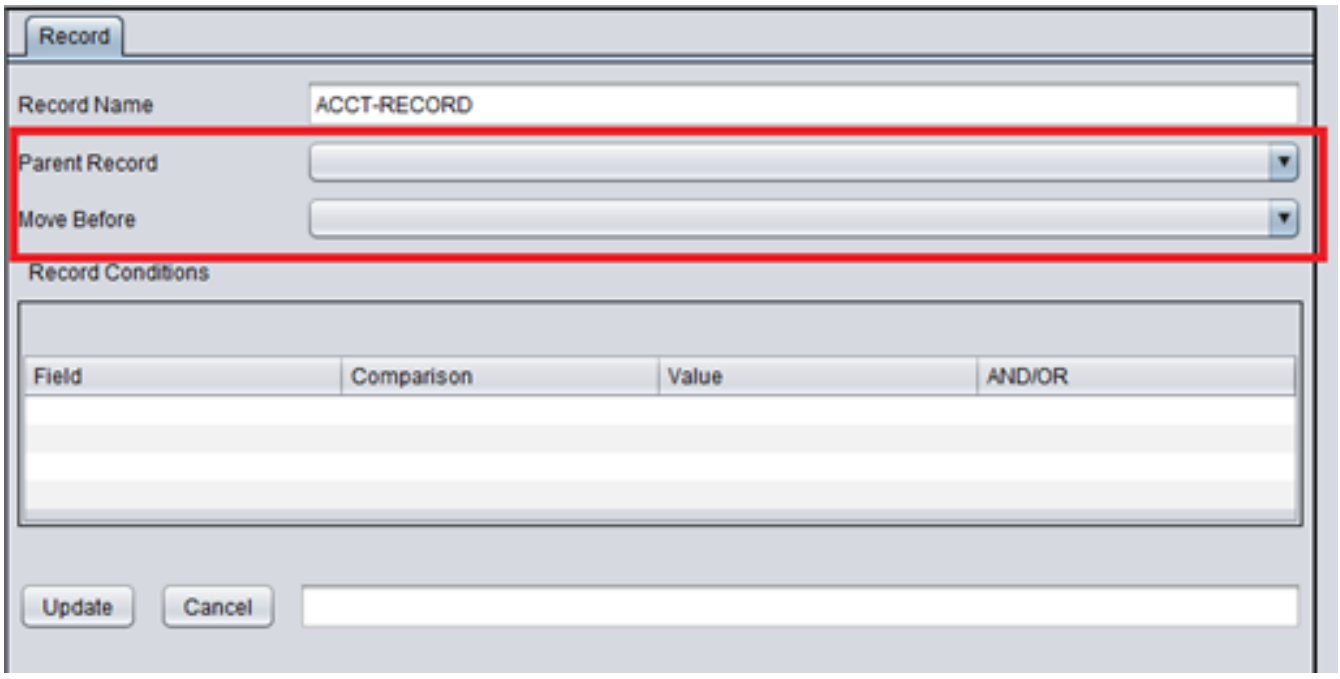
For example, for the following copy book, both ACCT-CUST, which is redefined, and ACCT-CUST-KEY, which redefines ACCT-CUST are marked as belonging to separate tables:

```
01 ACCT-RECORD.
   03 ACCT-GRP-KEY.
      05 ACCT-INST PIC S9(04) COMP.
      05 ACCT-CUST.
         07 ACCT-ALPHA PIC X(14).
         07 ACCT-ACCUM PIC S9(03) COMP-3.
         07 ACCT-CUST-KEY REDEFINES ACCT-CUST PIC X(16).
       05 ACCT-RECTYPE PIC X(01).
       05 ACCT-ACCOUNT.
         07 ACCT-ACCT-INST PIC 9(04) COMP.
         07 ACCT-APPL PIC 99.
          07 ACCT-BRANCH PIC S9(05) COMP-3.
          07 ACCT-CLASS PIC S9(03) COMP-3.
          07 ACCT-ACCT PIC S9(8) COMP.
```



You can explicitly specify that items should have their own table for data generation by enabling the **Generate to separate table** radio button.

For multi-record format files if there is a hierarchical relationship between the different record types, and if the order of records is important, you should record this information with the **Parent Record** and **Move Before** options in the details tab for the records.



When you click on the output files button in File Definition manager you will get three outputs – an Advanced File Layout (AFL) suffixed _ZOS.AFL.DM.txt, an AFL suffixed _DG.AFL.DM.txt and an Excel spreadsheet.

For data generation the Excel spreadsheet should be registered in Datamaker as a File Definition from G-T Excel file. To allow records to be split into separate tables for generation and reassembled correctly after publishing the File Definition Manager adds a 32 character table identifier field (called _RECNAME) to each table. In addition, to facilitate the correct ordering of separate parts of the record the File Definition Manager may create "link" tables which don't contain any fields other than the table identifier field.

The G-T Excel file for the above example contains six tabs as follows, the first two of these are link records, the remaining tabs hold the data items in the record split according to the redefinitions present:

Field Name	Mandatory	From	To	Length	Format	Column Name	Value Start	Value End	Default
1	M	1	32	32	a	_RECNAME	ACCT_RECORD	ACCT_RECORD	ACCT_RECORD

Field Name	Mandatory	From	To	Length	Format	Column Name	Value Start	Value End	Default
2	M	1	32	32	a	_RECNAME	ACCT_GRP_KEY	ACCT_GRP_KEY	ACCT_GRP_KEY

Field Name	Mandatory	From	To	Length	Format	Column Name	Value Start	Value End	Default
3	M	1	32	32	a	_RECNAME	ACCT_RECORD#1	ACCT_RECORD#1	ACCT_RECORD#1
4		33	38	6	n	ACCT_INST			

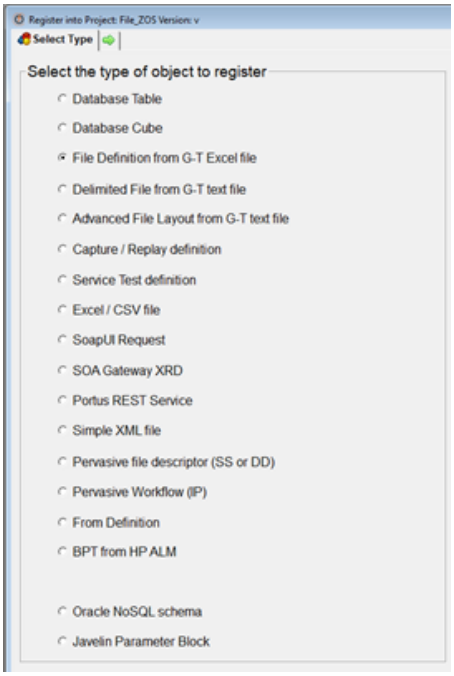
Field Name	Mandatory	From	To	Length	Format	Column Name	Value Start	Value End	Default
5	M	1	32	32	a	_RECNAME	ACCT_RECORD#2	ACCT_RECORD#2	ACCT_RECORD#2
6		33	46	14	a	ACCT_ALPHA			
7		47	50	4	n	ACCT_ACCUM			

Field Name	Mandatory	From	To	Length	Format	Column Name	Value Start	Value End	Default
8	M	1	32	32	a	_RECNAME	ACCT_RECORD#3	ACCT_RECORD#3	ACCT_RECORD#3
9		33	48	16	a	ACCT_CUST_KEY			

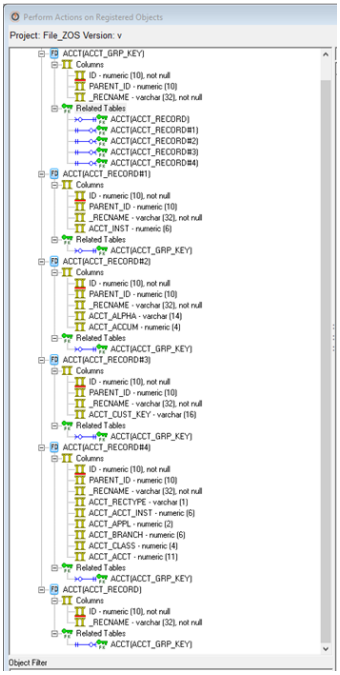
Field Name	Mandatory	From	To	Length	Format	Column Name	Value Start	Value End	Default
10	M	1	32	32	a	_RECNAME	ACCT_RECORD	ACCT_RECORD	ACCT_RECORD
11		33	33	1	a	ACCT_RECTYPE			
12		34	39	6	n	ACCT_ACCT_INST			
13		40	41	2	n	ACCT_APPL			
14		42	47	6	n	ACCT_BRANCH			
15		48	51	4	n	ACCT_CLASS			
16		52	61	11	n	ACCT_ACCT			

Registering the G-T Excel File Definition

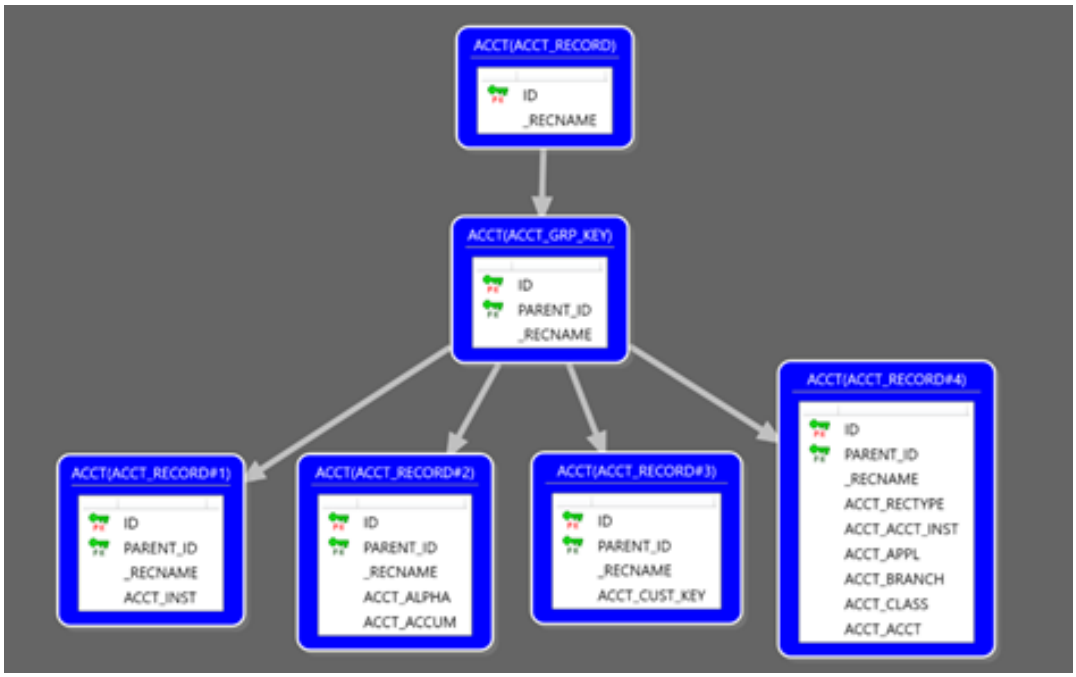
The Excel file created by the File Definition Manager should be registered as a File Definition from G-T Excel file.



In the registration process primary key columns (named ID) and foreign key columns (named PARENT_ID) are added to the tables to record the relationships between them:



The table relationships can also be viewed using GT Diagrammer:



File Conversion Processing

Because of the restructure of the record that happens when redefinitions and arrays are split out, and because of mainframe specific data types, such as packed decimal, which do not have a Windows equivalent, a file conversion process is necessary to create a mainframe file from data published to file by Datamaker, or to create a file which can be imported into a datapool.

Implementation

File conversion utility programs are supplied in the mainframe TDM installation. Unpack the contents of the provided zip file. Unpack and copy the FileConversion directory to a directory on your computer. To more easily read and edit the scripts, we recommend you keep the path to the directory structure short.

NOTE

Because the applications create work files in the directory from which they run, assign write privileges to the applications.

The following sub-directories are located under the FileConversion directory:

Executables
Contains the FujitsuCobol programs and example command files.

Work
Contains temporary work files

NOTE

These files are useful for error diagnosis.

Runtime

Contains the FujitsuCobol documentation and runtime install. The runtime is free to distribute and install.

NOTE

Execution of the runtime installer "FujitsuNetCOBOL.exe" is required by the conversion programs.

The GTXGEN.exe conversion program reads a source file that is described by an advanced DM.txt file. The conversion program then creates a target file that is described by another advanced DM.txt file.

Fields in the DM.txt files are matched by name and the data is converted and moved from the source to the target.

Example batch scripts are supplied as follows:

- **ConvertA2E.cmd**
  Converts file TestfileGen.txt from ASCII to EBCDIC format
- **ConvertE2A.cmd**
  Converts the file TestFile_From_ZOS.txt from EBCDIC to ASCII

Parameters

- **EXEDIR**
  Indicates the directory that contains the file conversion programs
- **WORKDIR**
  Indicates the working directory that contains environment setup files/input files to the programs.

  NOTE

  If this directory does not exist, the .cmd script creates the directory.
- **LOGFILE**
  Indicates the location and name of the log file

  NOTE

  This file is useful for error resolution.
- **ERRORFILE**
  Indicates the location and name of the error file.

NOTE

This file is deleted at the end of the run If the processing is successful.

- **SOURCEDMTXT**
  Indicates the advanced file layout for the file that you want to print
- **INPUTFILE**
  Indicates he file that you want to print
- **INPUTMODE**
  Indicates the file transfer method
  Values: LS (Windows style with line delimiters), B - FTP (blocked mode files from zOS), S - FTP (stream mode files from zOS)
- **INPUTRECFM**
  Indicates the input file record format
  Not required if INPUTMODE=LS, otherwise:
  Values: V (Varying length), VB (Varying length blocked), F (Fixed length), FB (Fixed length blocked)

  NOTE

  This parameter is not required if INPUTMODE=LS
- **INPUTLRECL**
  Indicates the input file record length
  Values: A positive number that indicates the maximum record length for V and VB files, or the record length for F and FB files

  NOTE

  This parameter is not required if INPUTMODE=LS
- **INPUTCODESET**
  Indicates the input set code files
  Values: ASCII, EBCDIC
- **OUTPUTFILE**
  Indicates the location of the printed files
- **OUTPUTMODE**
  Indicates the file transfer method
  Values: LS (Windows style with line delimiters), B (FTP blocked mode files fro zOS)
- **OUTPUTRECFM**
  Indicates the output file record format
  Values: V (Varying length), VB (Varying length blocked), F (Fixed length), FB (Fixed length blocked)

  NOTE

  This parameter is not required if INPUTMODE=LS
- **OUTPUTLRECL**
  Indicates the output file record length
  Values: A positive number that indicates the maximum record length for V and VB files, or the record length for F and FB files

  NOTE

  This parameter is not required if OUTPUTMODE=LS
- **OUTPUTCODESET**
  Indicates the output set code files
  Values: ASCII, EBCDIC

- **DIAGLEVEL**
  Indicates the diagnostic level
  Values: 0 (No diagnostics), 1, 2, 3, 4 (Highest level of diagnostics)

Creating a data file to import into a data pool

Once you have created and registered a G-T Excel file definition it may be useful to aid and guide generation to import some example data into a data pool.

Follows these steps:

1. Transfer an example file from the mainframe to Windows, the transfer should be done in binary mode, if the file has a varying record length it is important that the record length prefixes are included in the transferred file (this can be achieved by using the FTP option "quote mode b").
2. Run the file conversion process to create a version of the file that can be imported.
   For a varying record length file the file conversion parameters would be similar to the following example:

```
set EXEDIR=C:\FileConversion\Executables
set WORKDIR=C:\FileConversion\Work
set LOGFILE=C:\FileConversion\Work\Log.log
set ERRORFILE=C:\FileConversion\Work\Error.error

set SOURCEDMTXT=C:\FileConversion\ACCT_ZOS.AFL.DM.txt --AFL describing the file on
 ZOS
set INPUTFILE=C:\FileConversion\acctVB.dat --the transferred file name
set INPUTMODE=B --indicates that the file was transferred in Block mode (with length
 prefixes)
set INPUTRECFM=VB --the record format of the file on the mainframe
set INPUTLRECL=100 --the record length of the file on the mainframe
set INPUTCODESET=EBCDIC

set TARGETDMTXT=C:\FileConversion\ACCT_DG.AFL.DM.txt --AFL describing the file as
 registered in Datamaker
set OUTPUTFILE=c:\FileConversion\acctvb_from_zos.txt --the file output by the
 conversion
set OUTPUTMODE=LS --the file will be line sequential, ie with line terminators
set OUTPUTRECFM= --record format is not applicable to Windows files
set OUTPUTLRECL= --record length is not applicable to Windows files
set OUTPUTCODESET=ASCII
```

   For a fixed record length file the INPUT parameters to the conversion would be similar to the following

```
set INPUTFILE=C:\FileConversion\acctFB.dat --the transferred file name
set INPUTMODE=S --indicates that the file was transferred in stream mode (no length
 prefixes)
set INPUTRECFM=FB --the record format of the file on the mainframe
set INPUTLRECL=100 --the record length of the file on the mainframe
set INPUTCODESET=EBCDIC
```

3. In the projects view in Datamaker right click on a data pool within the project/version where you registed the G-T Excel file definition and select **Import External File Data**. You can then import the file output by the conversion program.

It is important to note that if the file definition includes redefinitions of fields then the imported data will include multiple data items belonging to the same part of a record. The example layout given above includes the following redefinition:

```
05 ACCT-CUST.
   07 ACCT-ALPHA PIC X(14).
   07 ACCT-ACCUM PIC S9(03) COMP-3.
05 ACCT-CUST-KEY REDEFINES ACCT-CUST PIC X(16).
```

The file created by the conversion process will include data corresponding to ACCT-CUST and data corresponding to ACCT-CUST-KEY.

Creating a Mainframe File from A Published File

For guidance on setting up data generation rules see the section Generate Synthetic Test Data.

Once you have created your generation rules you should publish "to file (FD)". The published file will be in Windows format and needs to be converted into a mainframe format by running the file conversion program.

For a varying record length file the file conversion parameters would be similar to the following:

```
set EXEDIR=C:\FileConversion\Executables
set WORKDIR=C:\FileConversion\Work
set LOGFILE=C:\FileConversion\Work\Log.log
set ERRORFILE=C:\FileConversion\Work\Error.error

set SOURCEDMTXT=C:\FileConversion\ACCT_DG.AFL.DM.txt – AFL describing the file as
 registered in Datamaker
set INPUTFILE=C:\FileConversion\published_acct.txt – the file published by Datamaker
set INPUTMODE=LS  – the file is sequential, ie with line terminators
set INPUTRECFM= - record format is not applicable to Windows files
set INPUTLRECL= - record length is not applicable to Windows files
set INPUTCODESET=ASCII

set TARGETDMTXT=C:\FileConversion\ACCT_ZOZ.AFL.DM.txt – AFL describing the file on ZOS
set OUTPUTFILE=c:\FileConversion\published_acct.dat – the file output by the conversion
set OUTPUTMODE=B – indicates that the output file should include length prefixes
set OUTPUTRECFM=VB – the record format of the file on the mainframe
set OUTPUTLRECL=100 – the record length of the file on the mainframe
set OUTPUTCODESET=EBCDIC
```

For a fixed record length file the OUTPUT parameters to the conversion would be similar to the following:

```
set OUTPUTFILE=c:\FileConversion\published_acct.dat – the file output by the conversion
set OUTPUTMODE=S – indicates that the output file should not include length prefixes
set OUTPUTRECFM=FB – the record format of the file on the mainframe
set OUTPUTLRECL=100 – the record length of the file on the mainframe
set OUTPUTCODESET=EBCDIC
```

[C:0reating a Mainframe File from A Published File

05 ACCT-CUST.

07 ACCT-ALPHA PIC X(14).

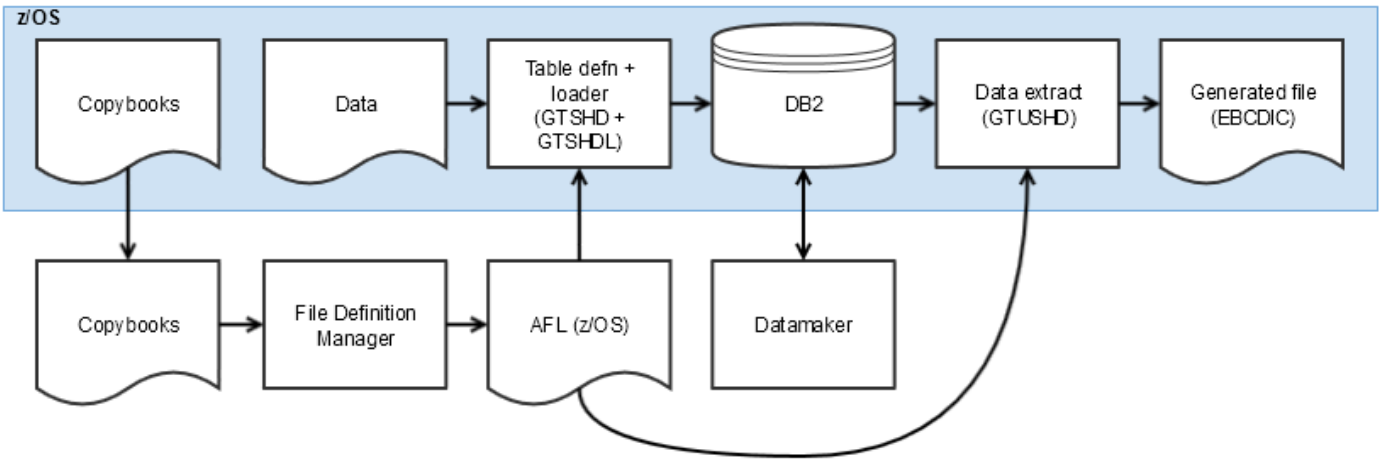
07 ACCT-ACCUM PIC S9(03) COMP-3.

05 ACCT-CUST-KEY REDEFINES ACCT-CUST PIC X(16).

Generate Synthetic Mainframe File Data using DB2 Tables

The aim of "shredding" and "unshredding" file data is to simplify the process for mainframe file processing. Shredding loads data, unshredding creates a file with DB2 data. The approach described here is an improvement over the approach described in Create an Advanced File Layout (AFL) with File Definition Manager. It requires fewer data transfers and does not require you to create the GT Excel file and generation AFL files, and go through the file conversion program.

Figure 54: generating synthetic mainframe data process



Follow these steps:

1. Transfer copybooks to Windows.
2. Parse copybooks to produce one z/OS AFL file.
   For more information, see Create an Advanced File Layout (AFL) with File Definition Manager.
3. Transfer AFL file to z/OS.
4. Run z/OS shredder program to define DB2 tables and load example data.
   For more information, see Define Tables for Data Generation for Mainframe Files
   a. Run JCL GTXSHD.
   b. Run JCL GTXSHDL.
5. Register DB2 tables in Datamaker.
6. Set up data generation rules.
7. Publish to DB2.
8. Run z/OS unshredder program to read DB2 and create a file.
   For more information, see Create Mainframe Files from Data Stored in DB2 Tables
   a. Run JCL GTXUSHD.

Architecture

Every table is created with a primary key consisting of columns BUNDLE_ID and BIIN_ID. Rows relating to a given physical file all have the same BUNDLE_ID, BIIN_ID is unique within a table. Column FK_BIIN_ID is used to link tables using foreign keys.

For a multi-record file, the order of the records in the file is established using the foreign keys on the tables storing the file data. If there is a hierarchical structure to the records, and if you recorded this in the File Definition Manager by

specifying the parent-child relationships between records, then this is used to set up foreign key relationships. If there are any records without a parent, then the shredding program (GTXSHD) creates a link table to act as a parent to all the records in the file. This link table contains just two columns: BUNDLE_ID and BIIN_ID.

Example:

For a file with a CUSTOMER record type and an ACCOUNT record type, but no parent-child relationship, you have the following data:

LINK_TABLE data

Bundle_ID	BIIN_ID
1	1
1	2
1	3
1	4

CUSTOMER data

Bundle_ID	BIIN_ID	FK_BIIN_ID	Customer_Name
1	1	1	John Smith
1	2	4	Jane Doe

ACCOUNT data

Bundle_ID	BIIN_ID	FK_BIIN_ID	Account_No
1	1	2	12345678
1	2	3	99999991

After unshredding, the file contains:

1. Customer record for John Smith
2. Account record 12345678
3. Account record 99999991
4. Customer record for Jane Doe

The BIIN_ID order on the parent table (LINK_TABLE) determines the order in which child table data appears in the file.

If the CUSTOMER record has been defined as the parent of the ACCOUNT record:

CUSTOMER data

Bundle_ID	BIIN_ID	Customer_Name
1	1	John Smith
1	2	Jane Doe

ACCOUNT data

Bundle_ID	BIIN_ID	FK_BIIN_ID	Account_No
1	1	1	12345678
1	2	2	99999991
1	3	2	99999992

After unshredding, the file contains:

1. Customer record for John Smith
2. Account record 12345678
3. Customer record for Jane Doe
4. Account record 99999991
5. Account record 99999992

The BIIN_ID order on the parent table (CUSTOMER) determines the order.

Limits and Tips

- **DB2 table limit:** Currently, the maximum number of DB2 tables the processing can handle is 200. It is possible that this processing limit will be increased in the future.

  TIP

  Calculate the base number of tables required for a given file by adding the number of arrays defined in the record layouts to the number of record types in the file. However, this number may be increased depending upon the maximum number of columns allowed for each table.
- **Null columns:** Redefined and redefining fields in a record layout appear as separate columns in the DB2 tables. If you load data into the tables using program GTXSHD, the program will attempt to populate all of these columns. Effectively this means that you will have multiple versions of the data in the redefined field. When "unshredding" (creating a file with the DB2 data) GTXSHD2, the program creating the file, processes each column in the order that it comes across it, and puts the data from that column into a record. Consequently, the last column that is supplying data determines what appears in a redefined field.

  TIP

  This program does not move data if a column is null, so to specify which version of the data you want to use, set to null the columns which are not applicable.
- **Arrays:** Arrays are supported up to 3 levels of nesting.
- **Dynamic arrays:** Dynamic arrays are supported, but not nested dynamic arrays. For dynamic arrays, verify that the count field occurs before the first dynamic array in the record. Count fields must be elementary (non-arrayed) fields.
- **Table Names:** To ensure that unique DB2 table names are created, use a unique table name prefix for each file definition. Supply the table name prefix as parameter TABLENAMEPREFIX to program GTXSHD.

Define Tables for Data Generation for Mainframe Files

Program GTXSHD writes DDL to create DB2 tables for data generation for mainframe files.

In addition, given an example file, the program creates control cards and data files to load data into the tables. The data load is done using the IBM DSNUPROC utility, an example of the use of this utility is given in GRIDT01.LIB.RUNJCL(GTSHDL). The process of defining file data in DB2 tables is referred to as "shredding" because the data for a single file may be represented by multiple tables.

JCL is supplied in the installation package GRIDT01.LIB.RUNJCL(GTXSHD). This job uses procedure GRIDT01.LIB.PROCLIB(GTSHD).

JCL Parameters

You can use the following parameters for the JCL procedure:

- **LOADLIB**

Defines the load library that contains programs GTXDEF and GTXSHD.

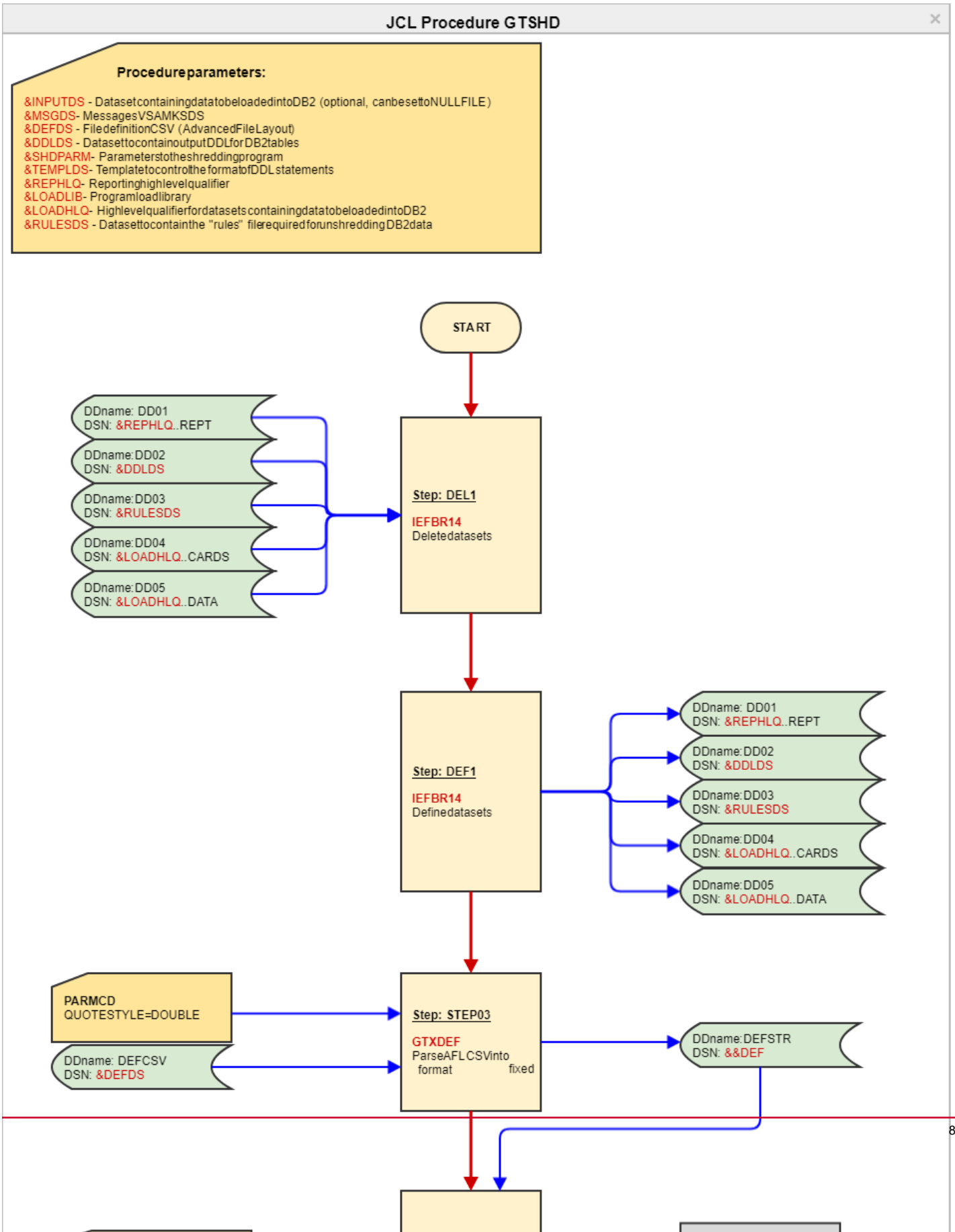
- **MSGDS**
  Defines the VSAM data set containing the TDM error messages.
- **REPHLQ**
  Gives the high-level dataset name qualifier that is used for the report files.
- **LOADHLQ**
  Gives the High Level Qualifier for DSNUPROC load card and data datasets.
- **PCYL**
  Gives the primary space allocation (in cylinders) for the DSNUPROC data.
- **SCYL**
  Gives secondary space allocation (in cylinders) for the DSNUPROC data.
- **SHDPARM**
  Defines the dataset that contains the parameters to GTXSHD (see below for possible parameters).
- **INPUTDS**
  Defines the dataset that containing data to be loaded into DB2.
- **DEFDS**
  Defines the dataset that contains the Advanced File Layout for the file.
- **TEMPLDS**
  Defines the dataset containing the template used to create DB2 DDL (see below).
- **DDLDS**
  Gives the name for the dataset to contain DB2 DDL statements.
- **RULESDS**
  Gives the name for the dataset to contain shredding "rules". This dataset is used by jobs GTXSHD1 and GTXSHD2.

The job performs the following steps:

1. Deletes the report, DB2 DDL, DB2 load control cards, DB2 load data and "rules" datasets.
2. Creates the above datasets.
3. Runs GTXDEF to read and parse the Advanced File Layout CSV, and write the CSV out to a fixed record format file.
4. Runs GTXSHD.

GTXSHD Flow Diagram

Figure 55: GTXSHD_flow



GTXSHD Parameters

- **BASECENTURY**
  Specifies the starting point for the century digit if the record definitions contain date formats with a single digit century. Example: BASECENTURY=19 and a date format of "CYYMMDD","1880729" is interpreted as 29th July 1988
- **BUNDLEID**
  All the DB2 tables defined include a "bundle_id" column which is part of the primary key. This is an integer which identifies all the table rows belonging to a given instance of a file. The supplied value is used in the DSNUPROC data dataset. The default value is zero.
- **MAXCHARSIZE**
  For string fields or columns, this parameter specifies the maximum size of char field to use in the DDL created. Fields or columns larger than this value are defined as varchar columns.
  Default: 255
- **MAXCOLSIZE**
  For string fields, this parameter specifies the maximum size of field that is included in the DDL created. Fields larger than the given value are ignored.
  Default: 4046
- **MAXCOLSINTABLE**
  The program creates separate tables for arrayed data items, but will also create separate tables where there are a large number of fields.
  Example: If a record contains 500 data items, and MAXCOLSINTABLE is set to 250, the record is represented by two tables each containing 250 columns.
  Default: 99
- **SEQUENCESTART**
  Optionally, table names can include a sequence number (see TABLENAMEPREFIX and VIEWNAMEPREFIX). By default the sequence starts at zero, but this can be changed by setting SEQUENCESTART.
- **BLANKSASNULLS**
  Specifies whether to represent string fields containing blanks by null columns when data files for loading into DB2.
  Default: N (blanks are not treated as nulls).
  Values: Y or N.
- **LOWSASNULLS**
  Specifies whether to represent string fields containing low values (X’00’) by null columns when data files for loading into DB2.
  Default: N (low values are not treated as nulls).
  Values: Y or N.
- **FORBITDATA**
  Specifies whether string fields defined in the Advanced File Layout as containing hex data should be defined in the DDL created as being "FOR BIT DATA". If FORBITDATA is not set to Y, then fields defined as containing hex data are represented by char columns twice the length of the field, and the column contains the hex representation of the string value.
  Values: Y or N.
- **LOADPARM1**
  The DSNUPROC load utility supports numerous options, by default, the control card for the utility is written with options "LOAD DATA RESUME YES LOG YES". You can override this option by supplying values for LOADPARM1 (and optionally for LOADPARM2 and LOADPARM3). For supported options see the section on Load in IBM’s DB2 for z/OS Utility Guide and Reference documentation.
- **LOADPARM2**
  See LOADPARM1
- **LOADPARM3**
  See LOADPARM1
- **TABLEOWNER**

Specifies the table owner used in the DDL created. For more information, see the section in the DDL Template.

- **VIEWOWNER**
  Specifies the table owner used in the DDL created. For more information, see the section in the DDL Template.
- **TABLENAMEPREFIX**
  In the created DDL, tables are given names based on data items defined in the Advanced File Layout. Optionally, you can supply a prefix to this name. If the supplied prefix contains the string "{TABLE_NO}" this will be replaced by a sequence number, starting with the value supplied in SEQUENCESTART.
- **VIEWNAMEPREFIX**
  In the created DDL views are given names based on data items defined in the Advanced File Layout. Optionally, you can supply a prefix to this name. If the supplied prefix contains the string "{TABLE_NO}" this will be replaced by a sequence number, starting with the value supplied in SEQUENCESTART.

DDL Template

An example DDL template is supplied in GRIDT01.LIB.PARM(TEMPL). Amend this template as required:

```
/*
{FOR_EACH_TABLE_B}
DROP TABLE {TABLE_OWNER}.{TABLE_NAME};
{END_FOR_EACH_TABLE_B}
*/
{FOR_EACH_TABLE}
SET CURRENT SQLID =  'DBA';
CREATE TABLESPACE STAB{TABLE_NO}
   IN DBNAME
   USING STOGROUP DATASTOGRP
   PRIQTY 533520 SECQTY 54000
   ERASE  NO
   FREEPAGE 0  PCTFREE 5
   GBPCACHE CHANGED
   TRACKMOD YES
   LOGGED
   SEGSIZE 64
   MAXPARTITIONS 10
   BUFFERPOOL BP11
   LOCKSIZE ANY
   LOCKMAX SYSTEM
   CLOSE NO
   COMPRESS NO
   CCSID      EBCDIC
   DEFINE YES
   MAXROWS 255;
GRANT USE OF TABLESPACE DBNAME.STAB{TABLE_NO} TO {TABLE_OWNER};

SET CURRENT SQLID='{TABLE_OWNER}';

CREATE TABLE
  {TABLE_OWNER}.{TABLE_NAME}(
  {FOR_EACH_COLUMN}
    {COLUMN_NAME}
    {COLUMN_TYPE}
  {END_FOR_EACH_COLUMN}
```

```
  )
  IN DBNAME.STAB{TABLE_NO}
  AUDIT NONE
  DATA CAPTURE NONE
  WITH RESTRICT ON DROP
  CCSID      EBCDIC
  NOT VOLATILE
  APPEND NO
;
SET CURRENT SQLID = 'DBA';

CREATE UNIQUE INDEX {TABLE_OWNER}.T{TABLE_NO}
  ON {TABLE_OWNER}.{TABLE_NAME}(
     BUNDLE_ID ASC, BIIN_ID ASC)
  USING STOGROUP INDXSTOGRP
  PRIQTY 12960 SECQTY 1440
  FREEPAGE 0 PCTFREE 10
  GBPCACHE CHANGED
  CLUSTER
  COMPRESS NO
  BUFFERPOOL BP12
  CLOSE NO
  COPY NO
  DEFER NO
  DEFINE YES;

ALTER TABLE {TABLE_OWNER}.{TABLE_NAME}
  ADD PRIMARY KEY (BUNDLE_ID,BIIN_ID);

{PARENT_EXISTS_START}
ALTER TABLE {TABLE_OWNER}.{TABLE_NAME}
  ADD FOREIGN KEY (BUNDLE_ID,FK_BIIN_ID)
  REFERENCES {TABLE_OWNER}.{PARENT_NAME} (BUNDLE_ID,BIIN_ID);
{PARENT_EXISTS_END}

GRANT DELETE,INSERT,SELECT,UPDATE
  ON TABLE {TABLE_OWNER}.{TABLE_NAME}
  TO {VIEW_OWNER} WITH GRANT OPTION;

SET CURRENT SQLID='{VIEW_OWNER}';

CREATE VIEW {VIEW_OWNER}.{VIEW_NAME} AS
 SELECT
 {FOR_EACH_COLUMN}
  {COLUMN_NAME}
 {END_FOR_EACH_COLUMN}
 FROM {TABLE_OWNER}.{TABLE_NAME}
 WITH CHECK OPTION;

GRANT SELECT, UPDATE, INSERT, DELETE
 ON TABLE {VIEW_OWNER}.{VIEW_NAME}
 TO  DBA WITH GRANT OPTION;
```

```
GRANT SELECT, UPDATE, INSERT, DELETE
 ON TABLE {VIEW_OWNER}.{VIEW_NAME}
 TO TEAM;
{END_FOR_EACH_TABLE}
```

Within the template, directives to program GTXSHD are enclosed in braces ("{" and "}"). Anything not enclosed in braces is written out to the DDL dataset by the program unchanged.

{FOR_EACH_TABLE}, {FOR_EACH_TABLE_B} and {FOR_EACH_COLUMN} represent loops in the program. You must end these loops in the template using {END_FOR_EACH_TABLE}, {END_FOR_EACH_TABLE_B} and {END_FOR_EACH_COLUMN}.

{FOR_EACH_TABLE} is a loop executed for every table created by GTXSHD. The template must include a table loop to define the tables used, and also the primary and foreign keys for the tables. Every table must have a primary key defined:

```
ALTER TABLE {TABLE_OWNER}.{TABLE_NAME}
  ADD PRIMARY KEY (BUNDLE_ID,BIIN_ID);
```

{VIEW_OWNER} and {TABLE_OWNER} are replaced in the output DDL using the supplied PARMCD values for TABLEOWNER and VIEWOWNER.

Within the template, a loop through the tables being created is signaled by {FOR_EACH_TABLE} and {END_FOR_EACH_TABLE}. This loop is executed in the order in which the tables are created. {FOR_EACH_TABLE_B} and {FOR_EACH_TABLE_B} loop in reverse table creation order, which is useful for dropping tables if required.

For a table loop iteration, you can refer to the table sequence number by using {TABLE_NO}, the table name using {TABLE_NAME}, and the view name using {VIEW_NAME}. {TABLE_NO}, {TABLE_NAME} and {VIEW_NAME} are replaced by appropriate values in the output DDL. {VIEW_NAME} and {TABLE_NAME} are generated by the program and include any suffix specified in the PARMCD using the TABLENAMEPREFIX and VIEWNAMEPREFIX parameters.

Within a table loop, you can refer to the columns belonging to that table by using a column loop signaled by {FOR_EACH_COLUMN} and {END_FOR_EACH_COLUMN}.

Within a column loop, you can refer to {COLUMN_NAME} and {COLUMN_TYPE}, the data type of the column. Do not alter the column name (for example, by prefixing or suffixing it) or column type – it is important that these values are set as determined by the program.

{PARENT_EXISTS_START} and {PARENT_EXISTS_END} allow you to specify output lines which should only appear if a given table is the child of another table. Use this to specify foreign key relationships.

The definition of views in the template is optional, but it must include table definitions along with associated primary key and foreign key constraints. The following example shows the simplest possible usable template:

```
{FOR_EACH_TABLE}
CREATE TABLE
  {TABLE_OWNER}.{TABLE_NAME}(
  {FOR_EACH_COLUMN}
    {COLUMN_NAME}
    {COLUMN_TYPE}
  {END_FOR_EACH_COLUMN}
  )
;

ALTER TABLE {TABLE_OWNER}.{TABLE_NAME}
  ADD PRIMARY KEY (BUNDLE_ID,BIIN_ID);

{PARENT_EXISTS_START}
ALTER TABLE {TABLE_OWNER}.{TABLE_NAME}
```

```
  ADD FOREIGN KEY (BUNDLE_ID,FK_BIIN_ID)
  REFERENCES {TABLE_OWNER}.{PARENT_NAME} (BUNDLE_ID,BIIN_ID);
{PARENT_EXISTS_END}

{END_FOR_EACH_TABLE}
```

Create Mainframe Files from Data Stored in DB2 Tables

JCL is supplied in the installation package GRIDT01.LIB.RUNJCL(GTXUSHD). This job uses procedure GRIDT01.LIB.PROCLIB(GTUSHD).

This job reads data from DB2 tables and uses it to create a "target" file. This process is referred to as "unshredding".

You can use the following parameters for the JCL procedure:

- **LOADLIB**
  Defines the load library that contains programs GTXDEF and GTXSHD1 and GTXSHD2.
- **MSGDS**
  Defines the VSAM data set containing the TDM error messages.
- **REPHLQ**
  Gives the high-level dataset name qualifier that is used for the report files.
- **DEFDS**
  Defines the dataset that contains the Advanced File Layout for the file.
- **RULESDS**
  Defines the dataset containing the shredding "rules". This dataset is used by created by job GTXSHD.
- **FILEDS**
  Gives the name of a dataset to contain the mainframe file being created.
- **RECFM**
  The record format of the mainframe file being created.
- **LRECL**
  Defines the logical record length of the target file being created.
- **BLK**
  Defines the block size of the target file being created.
- **SP1**
  Defines the primary space allocation (in cylinders) of the target file being created.
- **SP2**
  Defines the secondary space allocation (in cylinders) of the target file being created.

The job performs the following actions:

1. Deletes the report, DB2 data and target datasets.
2. Creates the above datasets.
3. Runs GTXDEF to read and parse the Advanced File Layout CSV, and write the CSV out to a fixed record format file.
4. Runs GTXSHD1
5. Runs GTXSHD2

GTXSHD1 Parameters

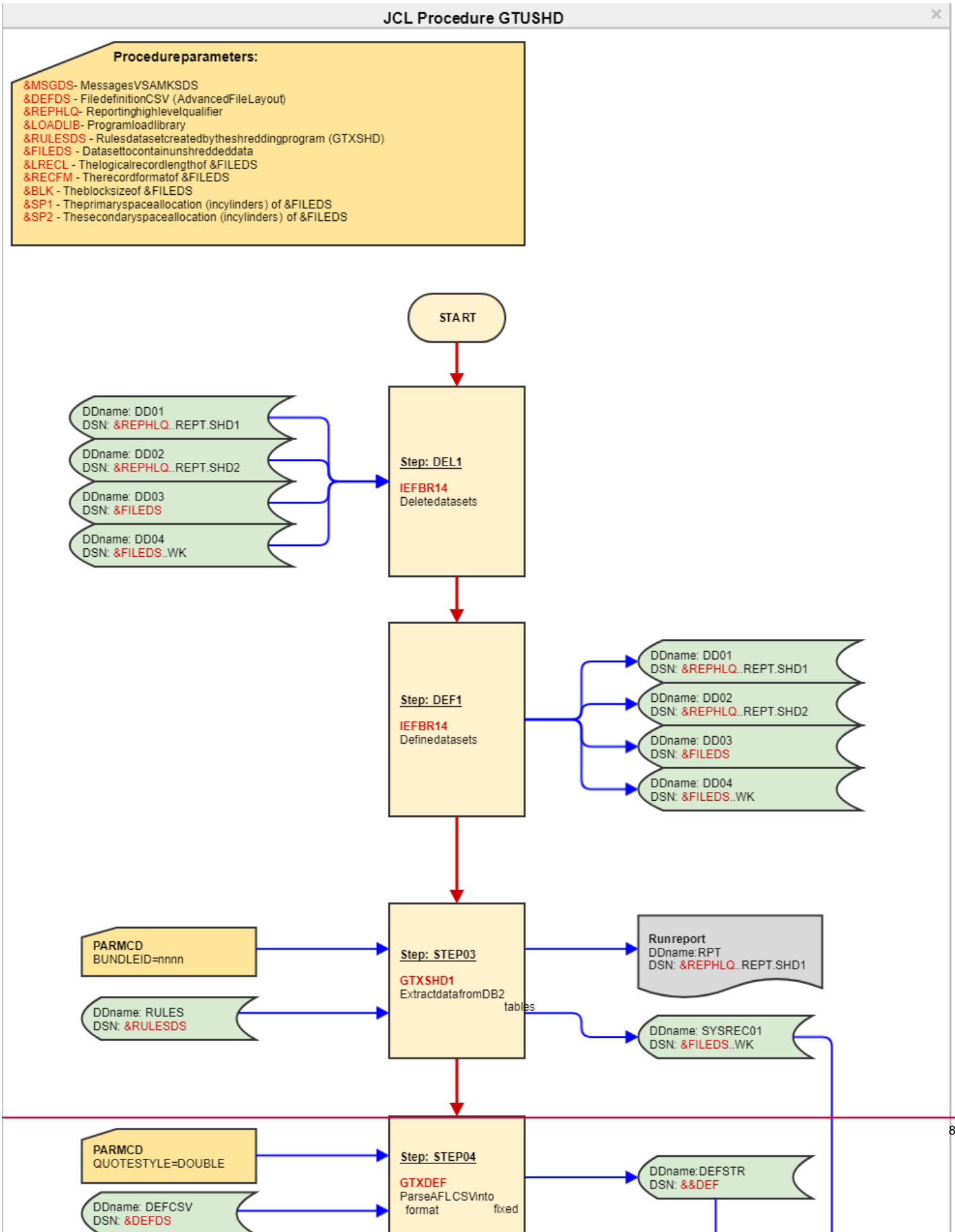
- **BUNDLEID**
  Defines the bundle_id primary key column value for which to extract data from the DB2 tables.
- **TABLEOWNER**
  The owner of the tables from which to extract data.

GTXSHD2 Parameters

- **BASECENTURY**
  Indicates the starting point for the century digit if the record definitions contain date formats with a single digit century.Example: BASECENTURY=19 and a date format of "CYYMMDD","1880729" is interpreted as 29th July 1988
- **OUTPUTRECFM**
  Defines the record format of the output file.
  - V — varying length
  - VB — varying length blocked, this is the default
  - F — fixed length
  - FB — fixed length blocked
- **OUTPUTLRECL**
  Defines the record length of the output file.
- **INITASBLANK**
  Initializes output records to blanks characters (x'40'). If set to "N", output records are initialized to low values (x'00'). Values: "Y" or "N".

GTXUSHD Flow Diagram

Figure 56: GTXUSHD



Mainframe Test Match Data Extract

You can use Mainframe Datamaker to extract data from the following sources:

- z/OS flat files
- VSAM files
- IMS databases

Create a Data Extract Transformation Map

The data extract job takes a transformation map CSV file as input. This file specifies the data items to extract.

To create transformation maps in Datamaker, navigate to **Projects**, **Transformation Maps**.

Create a transformation map with a DBMS of **ZOS**, and with the **ordered** option selected.

For each required field for test matching, apply the TESTMARTDATA or TESTMARTKEY. Both functions take one of the following optional parameters:

- **BIT**
  The field is converted to a bit representation in the data extract
- **HEX**
  The field is converted to a hex representation
- **DEC**
  The field is converted to a decimal representation

When the CSV file is exported for a given record type, ensure that TESTMARTKEY fields appear before TESTMARTDATA records. Set this order when you create the transformation map.

For files with multiple record types, there is a separate table in the extract for each record type. The fields are selected in this table. For a given record type, all the selected fields are included in an extract data row. If the AFL specifies parent-child relationships, an extract data row includes all ancestor records fields that are selected with TESTMARTKEY. This case is shown Example 3, Extract from an IMS Database.

Once the transformation map is created, export the map with a type of CSV – ZOS. Transfer the map to z/OS for the data extract program to read.

Load Data Into the Target Database

Transfer the files output by the data extract job out of z/OS. For data files (DDnames SYSRECnn), perform the transfer in binary mode. The data is loaded into Oracle using SQLLDR, or into SQL Server using BCP.

Run the z/OS Data Extract Job

The data extract job is a batch job that runs on z/OS.

To run flat file profiling, JCL is supplied in the installation package as GRIDTOOL.TDM549.RUNJCL(GTXTMT). This package is reproduced in Appendix H. This job uses procedure GRIDTOOL.TDM549.PROCLIB(GTXTMT).

The job has 2 pre-steps DEL1 and DEF1 which delete and define the output Test Match files. The job is set up for 3 test match load and data files. You can increase this by amending the DEL1, DEF1 and STEP05 overides.

You can supply the following parameters to the JCL procedure:

- **LOADLIB**
  Names the load library that contains programs GTXDEF, GTXMAP, and GTXTMT.
- **INDS**

Names the file from which data is to be extracted.

- **MAPDS**
  Names the dataset that contains the mapping CSV (which specifies the fields to be extracted).
- **DEFFDS**
  Names the dataset that contains the record definition CSV (Advanced File Layout).
- **TABS**
  Names the output dataset to contain table DDL statements.
- **REPHLQ**
  Gives the high-level dataset name qualifier to be used for the audit and report files.

The masking job contains the following steps:

1. (JCL) IEFBR14 to delete the load and data files.
2. (JCL) IEFBR14 to define the load and data files.
3. IEFBR14 to delete the report and table DDL files.

   NOTE

   The report file is allocated with SPACE=(CYL,(1,1)) which should be sufficient for most runs. If a large number of error or warning messages are produced, you may need to increase the space allocation.
4. Same as step 1.
5. Runs GTXDEF to read and parse the record definition CSV, and write the CSV out to a fixed record format file.
6. Runs GTXMAP to read and parse the mapping CSV, and write the CSV out to a fixed record format file.
7. Runs GTXTMT to extract the required data items. See the section GTXTMT?Parameters.

   NOTE

   The number of datasets that are written to by GTXTMT, and the space requirements for these datasets, vary depending on the extract rules that are supplied in the mapping CSV and on the target DBMS. For this reason the submitted JCL contains delete and define steps and DD statements for some datasets.

Oracle and SQLServer

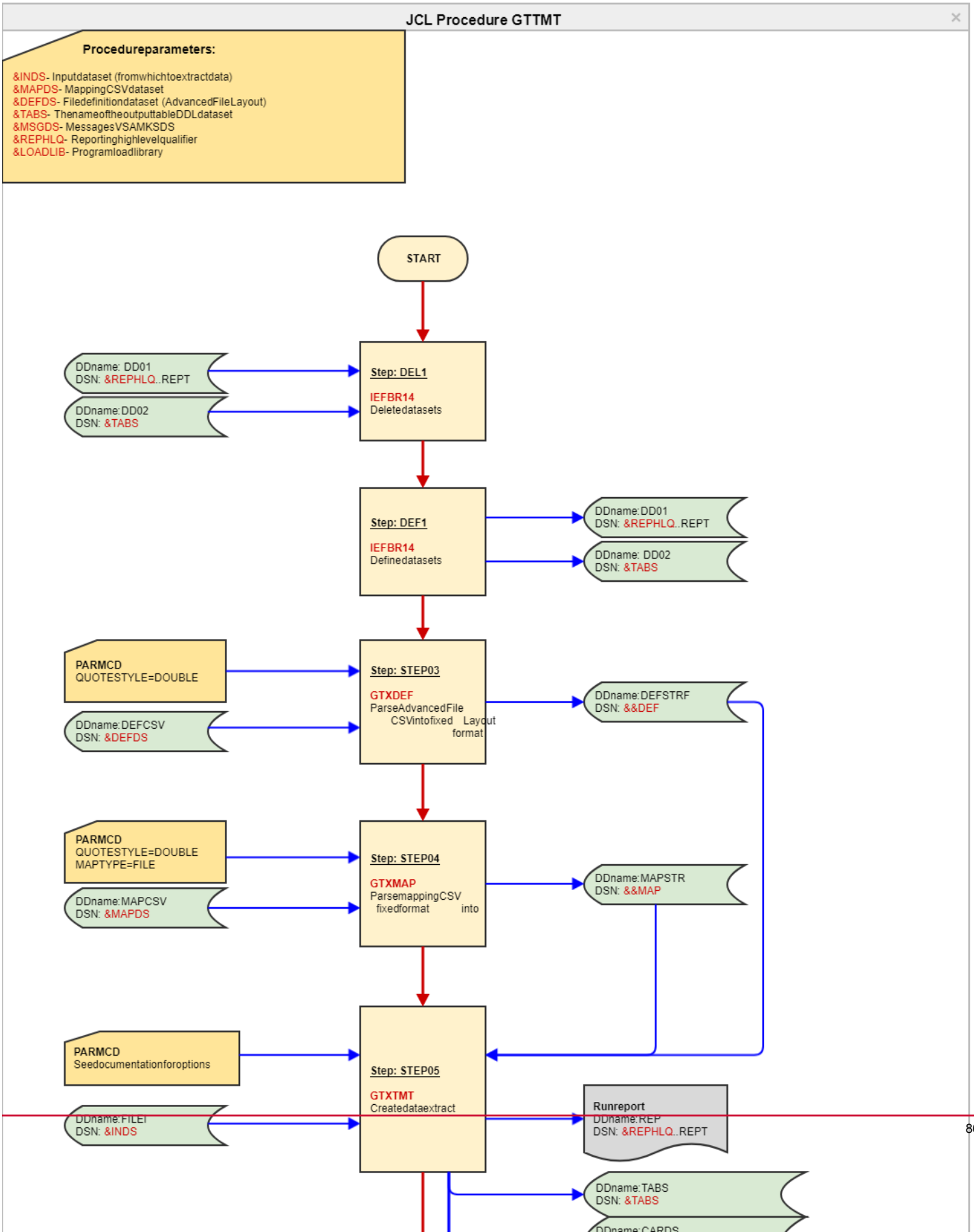
- Target datasets for DD name TABS and CARDS are always written to. The TABS dataset is allocated in the JCL procedure, the CARDS dataset should be allocated in the submitted JCL with LRECL=512 and RECFM=FB.
- Targets the data extracted is written to datasets with a DD name in the format SYSREC*nn*, where nn is a 2-digit number between 01 and 99. These datasets should have RECFM=VB. There will be one dataset corresponding to each record type for which an extract rule has been specified in the mapping CSV. The DD names for these datasets will be numbered sequentially starting at 01 in the order in which the record types are defined in the Advanced File Layout. For example if the mapping CSV gives extract rules for RECORD_A, RECORD_B and RECORD_C, and the Advanced File Layout contains a definition for RECORD_A followed by one for RECORD_C then RECORD_B, regardless of the order of the extract rules in the mapping CSV, data for RECORD_A will be written to DD name SYSREC01, for RECORD_C to DD name SYSREC02 and for RECORD_B to DD name SYSREC03. The maximum record length for these datasets depends upon the number and datatypes of the fields being extracted. The required space allocation will depend on the maximum record length and the number of records in the input file.

SQLServer

- As a target for each SYSRECnn dataset that is used, a corresponding CARDS*nn* dataset is written to. These datasets should have LRECL=126 and RECFM=FB. A space allocation of TRK(1,1) should be sufficient in almost all cases.

GTXTMT flow diagram

Figure 57: gtxtmt



GTXTMT Parameters

General

- **BLANKNONDISPLAY=**
  Specifies which non-display characters are converted to blanks
  Values: 7 (converts non-display characters in the 7-bit ASCII character set), 8 (converts non-display characters in the 8-bit ASCII character set), N
  Default: 7
- **BLANKSASNULLS=**
  Specifies whether to treat string fields that contain only blank characters as null fields.
  Values: N, Y
  Default: N
- **CHARASVCHAR=**
  Specifies whether to treat character fields as varying character fields. See TRIM.
  Values: N, Y
  Default: N
- **CONVERTTOASCII=**
  Specifies whether to convert EBCDIC data in the input file to ASCII.
  Values: N (for Oracle targets), Y (For SQLServer targets)
  Default: N
  Note: Use N for Oracle targets, and use Y for SQLServer targets.
- **LOADCHARSET=**
  Sets the character set that is specified in the control cards to the target database load job
  Default: WE8EBCDIC500
  Oracle target: The parameter cards file includes the line "CHARACTER SET WE8EBCDIC500" if the default is used.
  SQLServer target: The supplied value is used to populate the Column collation field in the program format files output. The recommended setting is "Latin1_General_CI_AS".
- **LOADERRPREFIX=**
  Supplies a string, if required, that is used to prefix the name of error files in the load program control cards.
  Oracle target: The program produces control cards that name an error file as SYSRECnn.bad and a discard file as SYSRECnn.dsc.
  Example:

  ```
  BADFILE 'SYSREC01.bad'
  DISCARDFILE 'SYSREC01.dsc'
  ```

  With "LOADERRPREFIX=C:\SQLLDR\ERRORS\" these lines appear as:

  ```
  BADFILE 'C:\SQLLDR\ERRORS\SYSREC01.bad'
  DISCARDFILE 'C:\SQLLDR\ERRORS\SYSREC01.dsc'
  ```

  SQLServer target: The program creates bcp command line parameters that name an error file.
  Example:

  ```
  -e "ERROR01.txt"
  ```

  With "LOADERRPREFIX=C:\BCP\ERRORS\" this line appears as

  ```
  -e "C:\BCP\ERRORS\ERROR01.txt"
  ```

- **LOADFILEPREFIX=**
  Supplies a string, if required, that is used to prefix the name of data files in the load program control cards.
  Oracle target: The program produces control cards that name a data file as SYSRECnn.dat.
  Example:

  ```
  INFILE 'SYSREC01.dat'
  ```

  With "LOADERRPREFIX=C:\SQLLDR\DATA\" this line appears as:

```
INFILE 'C:\SQLLDR\DATA\SYSREC01.dat'
```

SQLServer target: The program creates bcp command line parameters naming an input file
Example:

```
in "SYSREC01.dat"
```

With "LOADERRPREFIX=C:\BCP\DATA\" this line appears as

```
in "C:\BCP\DATA\SYSREC01.dat"
```

- **TABLEARRAYS=**
  Specifies whether the program extracts a single instance of any field within an array as part of the record structure.
  Values: N (the instance is extracted), Y (The program creates an extra output file per array that contains a field to be extracted. The file includes the load card and table definition.)
  Default: N
  Notes:
  – Specify the primary record TESTMARTKEY information, and group the array fields after the standard record fields, before you set TABLEARRAYS=Y. The additional table contains the selected fields, the TESTMARTKEY column data, and a subscript column per array dimension.
  – This option shreds any selected array data into separate tables from the original record.
- **TABLEPREFIXSTRING=**
  The program output includes target table names derived from the record names given in the Advanced File Layout record definitions. If a TABLEPREFIXSTRING is supplied, this is prefixed to the target table names.
- **TARGETDBMS=**
  Specifies the the target DBMS.
  Values: ORACLE, SQLSERVER
  Default: ORACLE
  Note: Only Oracle and SQLServer are currently supported.
- **TESTMSCHEMA=**
  Specifies the target database schema.
- **TRIM=**
  Specifies whether, and from where, to trim blanks from string field values. By default, values are not trimmed.
  Values: N (Blanks are not trimmed), R (Trim blanks from the end of strings), L (Trim blanks from the start of strings), B (Trim blanks from the end and from the start of strings)
  Default: N

Oracle specific parameters

The following parameters are only applicable for Oracle targets. For example, TARGETDBMS=ORACLE.

- **LOADENDIAN= BYTEORDER BIG**
  Specifies whether the value that is supplied is included in the control cards to SQLLDR.

SQLServer specific parameters

The following parameters are only applicable for SQLServer targets. For example, TARGETDBMS=SQLSERVER.

- **SQLSERVERARGS=**
  Specifies whether to supply required bcp command line arguments in addition to arguments that are automatically generated by the program.
  Example: "SQLSERVERARGS=-v" results in "-v" being appended to the bcp command lines output.
- **TESTMDATABASE=**
  Specifies the test data target database.
  Values: The name of the target database.

- **TESTMPASSWORD=**
  Specifies If bcp uses a user ID and password to run.
  Values: User ID
- **TESTMSERVER=**
  Specifies to which instance of SQLServer to connect.
  Note: The value supplied is included in the bcp command line parameters that are prefixed with "-S".
- **TESTMTRUSTED=**
  Specifies if bcp uses a trusted connection to run.
  Values: N, Y (Trusted connection is used)
  Default: N
- **TESTMUSER=**
  (SQLServer only) Specifies if bcp uses a user ID and password to run.
  Values: User ID

Dates

- **BASECENTURY=*nn***
  Specifies how BASECENTURY indicates the starting point for the century digit if the record definitions contain dateformats with a single digit century.
  Example: BASECENTURY=19 and a dateformat of "CYYMMDD","1880729" is interpreted as 29th July 1988.
- **DATEFORMAT=YEAR-MM-DD**
  The date format that is used by the target DBMS when date columns are populated.

Diagnostics

- **DIAGLEVEL=*n***
  Specifies the volume of diagnostics that are produced.
  Values: 0 (diagnostics are written to SYSOUT), 1, 2, 3, 4 (highest level of diagnostics)
- **PROGRESSCOUNT=*nnnn***
  Specifies the frequency to write the number of rows that are read and the time.
  Note: For every <nnnn> rows that are read, a line is written to SYSOUT that contains the number of rows and the time.

Other

- **LANGUAGE=**
  Specifies the two-character language code that is used for output messages.
  Values: EN, DE, ES, IT
  Default: EN

Examples

The following scenarios illustrate how to extract test matching data from the following sources:

- A single record file
- A multi-record file
- An IMS database containing customer data.

Example 1 – Extract from a File with One Record Type

In this scenario, a file with a single record type has copybook layout seen in Appendix A.This copybook is parsed to create an AFL named SINGLE_FILE_zOS.AFL.DM.txt (see Appendix G).

1. In Datamaker, navigate to **Project** and **Register**
2. Register an AFL in Project TestMatchExtract, Version SingleFile.
3. Run a profiling job.
4. Set up a transformation map to describe the fields to extract.
5. Create a transformation map with a **ZOS** DBMS.
   Note: Because the file has only one record type, there are no parent-child relationships. No parent-child relationships indicate that the TESTMARTKEY function is not used. Do not select the **Ordered** option when you create the map.

The fields that you select for extract depend on an analysis of the sampling results./ The selected fields also depend don the application where the file is used.

In this example, REC1_ALPHA and REC1_ACCUM are key fields. These fields have a TESTMARTDATA selected rather than TESTMARTKEY. TESTMARTKEY only affects the output from the extract job if the AFL contains parent-child relationships. REC1_STATUS is a single-byte field which can contain non-display characters, for this reason the TESTMARTDATA function is given a parameter value of "HEX".
Once the data extract fields are specified, exported the transformation map with a type of CSV-ZOS. Transfer the map to z/OS for input to the data extract program.

Outputs If *TARGETDMBS=SQLSERVER*

The outputs from the extract program include a table definition (output to DDname TABS):

```
CREATE TABLE
  [TEST].[dbo].[GTTM_REC1_RECORD](
    [REC1_RECORD_REC1_ALPHA] [char](14) NULL,
    [REC1_RECORD_REC1_ACCUM] [decimal](3,0) NULL,
    [REC1_RECORD_REC1_STATUS] [char](2) NULL,
    [REC1_RECORD_REC1_STATE] [char](2) NULL,
    [REC1_RECORD_REC1_DOB] [date] NULL,
    [REC1_RECORD_REC1_SEX] [char](1) NULL,
    [REC1_RECORD_REC1_MARITAL] [char](1) NULL,
    [REC1_RECORD_REC1_OCCUP_CD] [char](5) NULL,
    [REC1_RECORD_REC1_EDUC_LVL] [char](2) NULL,
    [REC1_RECORD_REC1_INC_CLASS] [char](5) NULL,
    [REC1_RECORD_REC1_OWN_RENT_CD] [char](1) NULL
 )
```

A BCP command file (output to DDname CARDS):-

```
bcp "TEST.dbo.GTTM_REC1_RECORD" in "SYSREC01.dat" -f "CARDS01.txt" -e "ERROR01.txt" -q -
T
```

A BCP format file (output to DDname CARDS01):-

```
9.0
11
1       SQLCHAR             0       14      "\t"      1 REC1_RECORD_REC1_ALPHA
Lat-in1_General_CI_AS
```

```
2        SQLCHAR             0       41      "\t"      2 REC1_RECORD_REC1_ACCUM
Lat-in1_General_CI_AS
3        SQLCHAR             0       2       "\t"      3 REC1_RECORD_REC1_STATUS
Lat-in1_General_CI_AS
4        SQLCHAR             0       2       "\t"      4 REC1_RECORD_REC1_STATE
Lat-in1_General_CI_AS
5        SQLCHAR             0       11      "\t"      5 REC1_RECORD_REC1_DOB
Lat-in1_General_CI_AS
6        SQLCHAR             0       1       "\t"      6 REC1_RECORD_REC1_SEX
Lat-in1_General_CI_AS
7        SQLCHAR             0       1       "\t"      7 REC1_RECORD_REC1_MARITAL
Lat-in1_General_CI_AS
8        SQLCHAR             0       5       "\t"      8 REC1_RECORD_REC1_OCCUP_CD
Lat-in1_General_CI_AS
9        SQLCHAR             0       2       "\t"      9 REC1_RECORD_REC1_EDUC_LVL
Lat-in1_General_CI_AS
10       SQLCHAR             0       5       "\t"      10 REC1_RECORD_REC1_INC_CLASS
Lat-in1_General_CI_AS
11       SQLCHAR             0       1       "\t\r\n"  11 REC1_RECORD_REC1_OWN_RENT_CD
Lat-in1_General_CI_AS
```

A data file (output to DDname SYSREC01), is not reproduced here.

Output If *TARGETDMBS=ORACLE*

The outputs from the extract program include a table definition (output to DDname TABS):

```
CREATE TABLE TESTM.GTTM_REC1_RECORD (      REC1_RECORD_REC1_ALPHA          CHAR (00014)
     ,    REC1_RECORD_REC1_ACCUM          DECIMAL ( 3, 0)                       ,     REC1_RECORD_REC1_STATUS
          CHAR (00002)                           ,     REC1_RECORD_REC1_STATE          CHAR (00002)
                       ,     REC1_RECORD_REC1_DOB           DATE                                    ,
 REC1_RECORD_REC1_SEX          CHAR (00001)                           ,     REC1_RECORD_REC1_MARITAL
      CHAR (00001)                          ,     REC1_RECORD_REC1_OCCUP_CD          CHAR (00005)
                  ,     REC1_RECORD_REC1_EDUC_LVL          CHAR (00002)                           ,
 REC1_RECORD_REC1_INC_CLASS         CHAR (00005)                          ,     REC1_RECORD_REC1_OWN_RENT_CD
         CHAR (00001)  );
```

A SQLLDR command file (output to DDname CARDS):-

```
LOAD DATA
CHARACTERSET WE8EBCDIC500 BYTEORDER BIG
INFILE 'C:\FILE\PREFIX\SYSREC01.dat' "VAR 5"
BADFILE 'C:\ERR\PREFIX\SYSREC01.bad'
DISCARDFILE 'C:\ERR\PREFIX\SYSREC01.dsc'
REPLACE
  INTO TABLE TESTM.GTTM_REC1_RECORD
 WHEN (1:2) = '01'
 (
    REC1_RECORD_REC1_ALPHA                              POSITION (00003:00016)
       CHAR
```

```
        NULLIF(00017)='?'                                          ,
    REC1_RECORD_REC1_ACCUM                          POSITION (00018:00019)
       DECIMAL ( 3, 0)
        NULLIF(00020)='?'                                          ,
    REC1_RECORD_REC1_STATUS                         POSITION (00021:00022)
       CHAR
        NULLIF(00023)='?'                                          ,
    REC1_RECORD_REC1_STATE                          POSITION (00024:00025)
       CHAR
        NULLIF(00026)='?'                                          ,
    REC1_RECORD_REC1_DOB                            POSITION (00027:00036)
       DATE "YYYY-MM-DD"
        NULLIF(00037)='?'                                          ,
    REC1_RECORD_REC1_SEX                            POSITION (00038:00038)
       CHAR
        NULLIF(00039)='?'                                          ,
    REC1_RECORD_REC1_MARITAL                        POSITION (00040:00040)
       CHAR
        NULLIF(00041)='?'                                          ,
    REC1_RECORD_REC1_OCCUP_CD                       POSITION (00042:00046)
       CHAR
        NULLIF(00047)='?'                                          ,
   REC1_RECORD_REC1_EDUC_LVL                        POSITION (00048:00049)
       CHAR
        NULLIF(00050)='?'                                          ,
   REC1_RECORD_REC1_INC_CLASS                       POSITION (00051:00055)
      CHAR
        NULLIF(00056)='?'                                          ,
   REC1_RECORD_REC1_OWN_RENT_CD                     POSITION (00057:00057)
      CHAR
        NULLIF(00058)='?'
)
```

A data file (output to DDname SYSREC01), not reproduced here.

Example 2 – Extract from a File with Multiple Record Types

In this scenario, a file has three record types. The cookbook type layouts of the types are noted in Appendix C. In this case, the file is a VSAM KSDS. The file might also be a flat file with the same record types.
The copybooks are parsed to create an AFL named MULTI_FILE.AFL.DM.txt (see Appendix H).
Because the file contains multiple record types, edit the AFL to record the record type conditions. Use the copybook editor to edit the AFL.

1. In Datamaker, navigate to **Project** and **Register**
2. Register an AFL in Project TestMatchExtract, Version MultiFile.
3. Run a profiling job.
   The transformation map can now be set up to describe the fields to extract.
4. Create a transformation map with a **ZOS** DBMS to describe the fields to extract.

Note: Because the file has only one record type, there are no parent-child relationships. The lack of parent-child relationships indicate that the TESTMARTKEY function is not used. Do not select the **Ordered** option when you create the map.

Create the transformation map with a **ZOS** DBMS.The file has multiple record types, but no parent-child relationships are added. Because the file is a VSAM KSDS, each record that relates to a customer contains the customer key. In this case, do not use the TESTMARTKEY function, and do not select the **Ordered** option when you create the map.

The fields to select for extract depend on analysis of the sampling results. The field selected also depends on the application in which the file is used.
In this example, each record has key fields RECn_ALPHA and RECn_ACCUM (where n = 1, 2 or 3). These fields are included in the extract to join data items from different records for the same customer.

Once the data extract fields are specified, exported the transformation map with a type of CSV-ZOS. Transfer the map to z/OS for input to the data extract program.

Outputs If *TARGETDMBS=SQLSERVER*

The outputs from the extract program include a three table definitions (output to DDname TABS):-

```
CREATE TABLE
   [TESTM].[dbo].[GTTM_REC1_RECORD](
      [REC1_RECORD_REC1_ALPHA] [char](14) NULL,
      [REC1_RECORD_REC1_ACCUM] [decimal](3,0) NULL,
      [REC1_RECORD_REC1_STATUS] [char](2) NULL,
      [REC1_RECORD_REC1_STATE] [char](2) NULL,
      [REC1_RECORD_REC1_DOB] [date] NULL,
      [REC1_RECORD_REC1_SEX] [char](1) NULL,
      [REC1_RECORD_REC1_MARITAL] [char](1) NULL,
      [REC1_RECORD_REC1_OCCUP_CD] [char](5) NULL,
      [REC1_RECORD_REC1_EDUC_LVL] [char](2) NULL,
      [REC1_RECORD_REC1_OWN_RENT_CD] [char](1) NULL
 )
CREATE TABLE
   [TESTM].[dbo].[GTTM_REC2_RECORD](
     [REC2_RECORD_REC2_ALPHA] [char](14) NULL,
     [REC2_RECORD_REC2_ACCUM] [decimal](3,0) NULL,
     [REC2_RECORD_REC2_EFF_DT] [date] NULL,
     [REC2_RECORD_REC2_EXP_DT] [date] NULL,
     [REC2_RECORD_REC2_STATE] [char](2) NULL
)
CREATE TABLE
  [TESTM].[dbo].[GTTM_REC3_RECORD](
    [REC3_RECORD_REC3_ALPHA] [char](14) NULL,
    [REC3_RECORD_REC3_ACCUM] [decimal](3,0) NULL,
    [REC3_RECORD_REC3_ACCT_INST] [decimal](5,0) NULL,
    [REC3_RECORD_REC3_APPL] [decimal](2,0) NULL,
    [REC3_RECORD_REC3_BRANCH] [decimal](5,0) NULL,
    [REC3_RECORD_REC3_CLASS] [decimal](3,0) NULL,
    [REC3_RECORD_REC3_ACCT] [decimal](21,0) NULL
)
```

A BCP command file (output to DDname CARDS):-

```
bcp "TESTM.dbo.GTTM_REC1_RECORD" in "SYSREC01.dat" -f "CARDS01.txt" -e "ERROR01.txt" -q
 -T
bcp "TESTM.dbo.GTTM_REC2_RECORD" in "SYSREC02.dat" -f "CARDS02.txt" -e "ERROR02.txt" -q
 -T
bcp "TESTM.dbo.GTTM_REC3_RECORD" in "SYSREC03.dat" -f "CARDS03.txt" -e "ERROR03.txt" -q
 -T
```

Three BCP format files (output to DDnames CARDS01, CARDS02 and CARDS03):

```
9.0
11
1       SQLCHAR             0       14      "\t"     1     REC1_RECORD_REC1_ALPHA
Latin1_General_CI_AS
2       SQLCHAR             0       41      "\t"     2     REC1_RECORD_REC1_ACCUM
Latin1_General_CI_AS
3       SQLCHAR             0       2       "\t"     3     REC1_RECORD_REC1_STATUS
Latin1_General_CI_AS
4       SQLCHAR             0       2       "\t"     4     REC1_RECORD_REC1_STATE
Latin1_General_CI_AS
5       SQLCHAR             0       11      "\t"     5     REC1_RECORD_REC1_DOB
Latin1_General_CI_AS
6       SQLCHAR             0       1       "\t"     6     REC1_RECORD_REC1_SEX
Latin1_General_CI_AS
7       SQLCHAR             0       1       "\t"     7     REC1_RECORD_REC1_MARITAL
Latin1_General_CI_AS
8       SQLCHAR             0       5       "\t"     8     REC1_RECORD_REC1_OCCUP_CD
Latin1_General_CI_AS
9       SQLCHAR             0       2       "\t"     9     REC1_RECORD_REC1_EDUC_LVL
Latin1_General_CI_AS
10      SQLCHAR             0       5       "\t"     10    REC1_RECORD_REC1_INC_CLASS
Latin1_General_CI_AS
11      SQLCHAR             0       1       "\t\r\n" 11    REC1_RECORD_REC1_OWN_RENT_CD
Latin1_General_CI_AS

9.0
5
1       SQLCHAR             0       14      "\t"      1     REC2_RECORD_REC2_ALPHA
Latin1_General_CI_AS
2       SQLCHAR             0       41      "\t"      2     REC2_RECORD_REC2_ACCUM
Latin1_General_CI_AS
3       SQLCHAR             0       11      "\t"      3     REC2_RECORD_REC2_EFF_DT
Latin1_General_CI_AS
4       SQLCHAR             0       11      "\t"      4     REC2_RECORD_REC2_EXP_DT
Latin1_General_CI_AS
5       SQLCHAR             0       2       "\t\r\n"  5     REC2_RECORD_REC2_STATE
Latin1_General_CI_AS
```

```
9.0
7
1       SQLCHAR             0       14      "\t"      1     REC3_RECORD_REC3_ALPHA
Latin1_General_CI_AS
2       SQLCHAR             0       41      "\t"      2     REC3_RECORD_REC3_ACCUM
Latin1_General_CI_AS
3       SQLCHAR             0       41      "\t"      3     REC3_RECORD_REC3_ACCT_INST
Latin1_General_CI_AS
4       SQLCHAR             0       41      "\t"      4     REC3_RECORD_REC3_APPL
Latin1_General_CI_AS
5       SQLCHAR             0       41      "\t"      5     REC3_RECORD_REC3_BRANCH
Latin1_General_CI_AS
6       SQLCHAR             0       41      "\t"      6     REC3_RECORD_REC3_CLASS
Latin1_General_CI_AS
7       SQLCHAR             0       41      "\t\r\n"  7     REC3_RECORD_REC3_ACCT
Latin1_General_CI_AS
```

Three data files (output to DDnames SYSREC01, SYSREC02 and SYSREC03), not reproduced here.

Outputs If *TARGETDMBS=ORACLE*

The outputs from the extract program include three table definitions (output to DDname TABS):-

```
CREATE TABLE TESTM.GTTM_REC1_RECORD (
    REC1_RECORD_REC1_ALPHA
     CHAR (00014)                  ,
    REC1_RECORD_REC1_ACCUM
      DECIMAL ( 3, 0)              ,
    REC1_RECORD_REC1_STATUS
     CHAR (00002)                  ,
    REC1_RECORD_REC1_STATE
     CHAR (00002)                  ,
    REC1_RECORD_REC1_DOB
      DATE                         ,
    REC1_RECORD_REC1_SEX
     CHAR (00001)                  ,
    REC1_RECORD_REC1_MARITAL
     CHAR (00001)                  ,
    REC1_RECORD_REC1_OCCUP_CD
     CHAR (00005)                  ,
    REC1_RECORD_REC1_EDUC_LVL
     CHAR (00002)                  ,
    REC1_RECORD_REC1_OWN_RENT_CD
     CHAR (00001)
   );
CREATE TABLE TESTM.GTTM_REC2_RECORD (
    REC2_RECORD_REC2_ALPHA
```

```
     CHAR (00014)                  ,
    REC2_RECORD_REC2_ACCUM
      DECIMAL ( 3, 0)              ,
    REC2_RECORD_REC2_EFF_DT
      DATE                         ,
    REC2_RECORD_REC2_EXP_DT
      DATE                         ,
    REC2_RECORD_REC2_STATE
     CHAR (00002)
   );
CREATE TABLE TESTM.GTTM_REC3_RECORD (
    REC3_RECORD_REC3_ALPHA
     CHAR (00014)                  ,
    REC3_RECORD_REC3_ACCUM
      DECIMAL ( 3, 0)              ,
    REC3_RECORD_REC3_ACCT_INST
      DECIMAL ( 5, 0)              ,
    REC3_RECORD_REC3_APPL
      DECIMAL ( 2, 0)              ,
    REC3_RECORD_REC3_BRANCH
      DECIMAL ( 5, 0)              ,
    REC3_RECORD_REC3_CLASS
      DECIMAL ( 3, 0)              ,
    REC3_RECORD_REC3_ACCT
      DECIMAL (21, 0)
   );
```

A SQLLDR command file (output to DDname CARDS):-

```
LOAD DATA
CHARACTERSET WE8EBCDIC500 BYTEORDER BIG
INFILE 'C:\FILE\PREFIX\SYSREC01.dat' "VAR 5"
BADFILE 'C:\ERR\PREFIX\SYSREC01.bad'
DISCARDFILE 'C:\ERR\PREFIX\SYSREC01.dsc'
INFILE 'C:\FILE\PREFIX\SYSREC02.dat' "VAR 5"
BADFILE 'C:\ERR\PREFIX\SYSREC02.bad'
DISCARDFILE 'C:\ERR\PREFIX\SYSREC02.dsc'
INFILE 'C:\FILE\PREFIX\SYSREC03.dat' "VAR 5"
BADFILE 'C:\ERR\PREFIX\SYSREC03.bad'
DISCARDFILE 'C:\ERR\PREFIX\SYSREC03.dsc'
REPLACE
  INTO TABLE TESTM.GTTM_REC1_RECORD
  WHEN (1:2) = '01'
   (
    REC1_RECORD_REC1_ALPHA                          POSITION (00003:00016)
      CHAR
        NULLIF(00017)='?'                                                ,
    REC1_RECORD_REC1_ACCUM                          POSITION (00018:00019)
```

```
    DECIMAL ( 3, 0)
      NULLIF(00020)='?'                                          ,
  REC1_RECORD_REC1_STATUS                          POSITION (00021:00022)
    CHAR
      NULLIF(00023)='?'                                          ,
  REC1_RECORD_REC1_STATE                           POSITION (00024:00025)
    CHAR
      NULLIF(00026)='?'                                          ,
  REC1_RECORD_REC1_DOB                             POSITION (00027:00036)
    DATE "YYYY-MM-DD"
      NULLIF(00037)='?'                                          ,
  REC1_RECORD_REC1_SEX                             POSITION (00038:00038)
    CHAR
      NULLIF(00039)='?'                                          ,
  REC1_RECORD_REC1_MARITAL                         POSITION (00040:00040)
    CHAR
      NULLIF(00041)='?'                                          ,
  REC1_RECORD_REC1_OCCUP_CD                        POSITION (00042:00046)
    CHAR
      NULLIF(00047)='?'                                          ,
  REC1_RECORD_REC1_EDUC_LVL                        POSITION (00048:00049)
    CHAR
      NULLIF(00050)='?'                                          ,
  REC1_RECORD_REC1_OWN_RENT_CD                     POSITION (00051:00051)
    CHAR
      NULLIF(00052)='?'
 )
INTO TABLE TESTM.GTTM_REC2_RECORD
WHEN (1:2) = '02'
 (
  REC2_RECORD_REC2_ALPHA                           POSITION (00003:00016)
    CHAR
      NULLIF(00017)='?'                                          ,
  REC2_RECORD_REC2_ACCUM                           POSITION (00018:00019)
    DECIMAL ( 3, 0)
      NULLIF(00020)='?'                                          ,
  REC2_RECORD_REC2_EFF_DT                          POSITION (00021:00030)
    DATE "YYYY-MM-DD"
      NULLIF(00031)='?'                                          ,
  REC2_RECORD_REC2_EXP_DT                          POSITION (00032:00041)
    DATE "YYYY-MM-DD"
      NULLIF(00042)='?'                                          ,
  REC2_RECORD_REC2_STATE                           POSITION (00043:00044)
    CHAR
      NULLIF(00045)='?'
 )
```

```
INTO TABLE TESTM.GTTM_REC3_RECORD
WHEN (1:2) = '03'
 (
  REC3_RECORD_REC3_ALPHA                          POSITION (00003:00016)
    CHAR
      NULLIF(00017)='?'                                                ,
  REC3_RECORD_REC3_ACCUM                          POSITION (00018:00019)
    DECIMAL ( 3, 0)
      NULLIF(00020)='?'                                                ,
  REC3_RECORD_REC3_ACCT_INST                      POSITION (00021:00023)
    DECIMAL ( 5, 0)
      NULLIF(00024)='?'                                                ,
  REC3_RECORD_REC3_APPL                           POSITION (00025:00026)
    DECIMAL ( 2, 0)
      NULLIF(00027)='?'                                                ,
  REC3_RECORD_REC3_BRANCH                         POSITION (00028:00030)
    DECIMAL ( 5, 0)
      NULLIF(00031)='?'                                                ,
  REC3_RECORD_REC3_CLASS                          POSITION (00032:00033)
    DECIMAL ( 3, 0)
      NULLIF(00034)='?'                                                ,
  REC3_RECORD_REC3_ACCT                           POSITION (00035:00045)
    DECIMAL (21, 0)
      NULLIF(00046)='?'
 )
```

Three data files (output to DDnames SYSREC01, SYSREC02 and SYSREC03), not reproduced here.

Example 3 – Extract from an IMS Database

In this scenario, an IMS database has three segment types. The copybook type layout is noted in in Appendix D, E, and F. SEG1 is the root segment with a key given by SEG1-CUST-KEY. SEG2 and SEG3 are children of SEG1. This example uses a simple database).
The copybooks are parsed to create an AFL named IMS.AFL.DM.txt (see Appendix I). The **IMS** button is checked when the parser is run. The parser prefixes each segment layout with an eight character field named SEGNAME.

Because the file contains multiple record/segment types, edit the AFL to record the record type conditions. Use the copybook editor to edit the AFL. Use the SEGNAME field in the record conditions. The data extract program runs against a flat file that contains segment data. Prefix each record in the file with an eight character field to hold the segment name.

Unlike the earlier VSAM KSDS example, in this case there is no key on each record/segment to join different segments for a given customer. The key is only present on the root segment. To specify the segment hierarchy, set parent records for all segments other than the root.

1. In Datamaker, navigate to **Project** and **Register.**
2. Register an AFL in Project TestMatchExtract, Version IMS.
3. Run a profiling job (not shown here)
4. Set up the transformation map to describe the fields to extract.
5. Create a transformation map with a **ZOS** DBMS

The file has multiple record/segment types, and we added parent-child relationships. Because the TESTMARTKEY function is used to tag key fields, the map is created with the **Ordered** option. The TESTMARTKEY fields are saved as the first fields in the map.

Which fields you select for extract depends on an analysis of the sampling results. The fields selected also depend on application in which the file is used.

SEG1 is the root segment and has a key that is made up of SEG1_ALPHA and SEG1_ACCUM. These items are selected for extract with the TESTMARTKEY function. This extract means that these values from the root segment are included in data extract rows for dependent segments.

Outputs If *TARGETDMBS=SQLSERVER*

The outputs from the extract program include a three table definitions (output to DDname TABS):

```
CREATE TABLE
  [TESTM].[dbo].[GTTM_SEG1](
    [SEG1_SEG1_ALPHA] [char](14) NULL,
    [SEG1_SEG1_ACCUM] [decimal](3,0) NULL,
    [SEG1_SEG1_STATUS] [char](2) NULL,
    [SEG1_SEG1_STATE] [char](2) NULL,
    [SEG1_SEG1_DOB] [decimal](9,0) NULL,
    [SEG1_SEG1_SEX] [char](1) NULL,
    [SEG1_SEG1_MARITAL] [char](1) NULL,
    [SEG1_SEG1_OCCUP_CD] [char](5) NULL,
    [SEG1_SEG1_EDUC_LVL] [char](2) NULL,
    [SEG1_SEG1_OWN_RENT_CD] [char](1) NULL
  )
CREATE TABLE
  [TESTM].[dbo].[GTTM_SEG2](
    [SEG1_SEG1_ALPHA] [char](14) NULL,
    [SEG1_SEG1_ACCUM] [decimal](3,0) NULL,
    [SEG2_SEG2_EFF_DT] [decimal](9,0) NULL,
    [SEG2_SEG2_EXP_DT] [decimal](9,0) NULL,
    [SEG2_SEG2_STATE] [char](2) NULL
  )
CREATE TABLE
  [TESTM].[dbo].[GTTM_SEG3](
    [SEG1_SEG1_ALPHA] [char](14) NULL,
    [SEG1_SEG1_ACCUM] [decimal](3,0) NULL,
    [SEG3_SEG3_ACCT_INST] [decimal](5,0) NULL
    [SEG3_SEG3_APPL] [decimal](2,0) NULL,
    [SEG3_SEG3_BRANCH] [decimal](5,0) NULL,
    [SEG3_SEG3_CLASS] [decimal](3,0) NULL
  )
```

A BCP command file (output to DDname CARDS):

```
bcp "TESTM.dbo.GTTM_SEG1" in "SYSREC01.dat" -f "CARDS01.txt" -e "ERROR01.txt" -q -T
bcp "TESTM.dbo.GTTM_SEG2" in "SYSREC02.dat" -f "CARDS02.txt" -e "ERROR02.txt" -q -T
bcp "TESTM.dbo.GTTM_SEG3" in "SYSREC03.dat" -f "CARDS03.txt" -e "ERROR03.txt" -q -T
```

Three BCP format files (output to DDnames CARDS01, CARDS02 and CARDS03):-

```
9.0
10
1       SQLCHAR             0       14      "\t"          1     SEG1_SEG1_ALPHA
Latin1_General_CI_AS
2       SQLCHAR             0       41      "\t"          2     SEG1_SEG1_ACCUM
Latin1_General_CI_AS
3       SQLCHAR             0       2       "\t"          3     SEG1_SEG1_STATUS
Latin1_General_CI_AS
4       SQLCHAR             0       2       "\t"          4     SEG1_SEG1_STATE
Latin1_General_CI_AS
5       SQLCHAR             0       41      "\t"          5     SEG1_SEG1_DOB
Latin1_General_CI_AS
6       SQLCHAR             0       1       "\t"          6     SEG1_SEG1_SEX
Latin1_General_CI_AS
7       SQLCHAR             0       1       "\t"          7     SEG1_SEG1_MARITAL
Latin1_General_CI_AS
8       SQLCHAR             0       5       "\t"          8     SEG1_SEG1_OCCUP_CD
Latin1_General_CI_AS
9       SQLCHAR             0       2       "\t"          9     SEG1_SEG1_EDUC_LVL
Latin1_General_CI_AS
10      SQLCHAR             0       1       "\t\r\n"      10    SEG1_SEG1_OWN_RENT_CD
Latin1_General_CI_AS
9.0
5
1       SQLCHAR             0       14      "\t"          1     SEG1_SEG1_ALPHA
Latin1_General_CI_AS
2       SQLCHAR             0       41      "\t"          2     SEG1_SEG1_ACCUM
Latin1_General_CI_AS
3       SQLCHAR             0       41      "\t"          3     SEG2_SEG2_EFF_DT
Latin1_General_CI_AS
4       SQLCHAR             0       41      "\t"          4     SEG2_SEG2_EXP_DT
Latin1_General_CI_AS
5       SQLCHAR             0       2       "\t\r\n"      5     SEG2_SEG2_STATE
Latin1_General_CI_AS
9.0
6
1       SQLCHAR             0       14      "\t"          1     SEG1_SEG1_ALPHA
Latin1_General_CI_AS
2       SQLCHAR             0       41      "\t"          2     SEG1_SEG1_ACCUM
Latin1_General_CI_AS
3       SQLCHAR             0       41      "\t"          3     SEG3_SEG3_ACCT_INST
Latin1_General_CI_AS
4       SQLCHAR             0       41      "\t"          4     SEG3_SEG3_APPL
Latin1_General_CI_AS
5       SQLCHAR             0       41      "\t"          5     SEG3_SEG3_BRANCH
```

```
Latin1_General_CI_AS
6       SQLCHAR             0       41      "\t\r\n"     6     SEG3_SEG3_CLASS
Latin1_General_CI_AS
```

Three data files (output to DDnames SYSREC01, SYSREC02 and SYSREC03), not reproduced here.

Outputs If *TARGETDMBS=ORACLE*

```
CREATE TABLE TESTM.GTTM_SEG1 (
    SEG1_SEG1_ALPHA
     CHAR (00014)                          ,
    SEG1_SEG1_ACCUM
      DECIMAL ( 3, 0)                      ,
    SEG1_SEG1_STATUS
     CHAR (00002)                          ,
    SEG1_SEG1_STATE
     CHAR (00002)                          ,
    SEG1_SEG1_DOB
      DECIMAL ( 9, 0)                      ,
    SEG1_SEG1_SEX
     CHAR (00001)                          ,
    SEG1_SEG1_MARITAL
     CHAR (00001)                          ,
    SEG1_SEG1_OCCUP_CD
     CHAR (00005)                          ,
    SEG1_SEG1_EDUC_LVL
     CHAR (00002)                          ,
    SEG1_SEG1_OWN_RENT_CD
     CHAR (00001)
   );
CREATE TABLE TESTM.GTTM_SEG2 (
    SEG1_SEG1_ALPHA
     CHAR (00014)                          ,
    SEG1_SEG1_ACCUM
      DECIMAL ( 3, 0)                      ,
    SEG2_SEG2_EFF_DT
      DECIMAL ( 9, 0)                      ,
    SEG2_SEG2_EXP_DT
      DECIMAL ( 9, 0)                      ,
    SEG2_SEG2_STATE
     CHAR (00002)
   );
CREATE TABLE TESTM.GTTM_SEG3 (
    SEG1_SEG1_ALPHA
     CHAR (00014)                          ,
    SEG1_SEG1_ACCUM
```

```
      DECIMAL ( 3, 0)                        ,
    SEG3_SEG3_ACCT_INST
      DECIMAL ( 5, 0)                        ,
    SEG3_SEG3_APPL
      DECIMAL ( 2, 0)                        ,
    SEG3_SEG3_BRANCH
      DECIMAL ( 5, 0)                        ,
    SEG3_SEG3_CLASS
      DECIMAL ( 3, 0)
  );
```

A SQLLDR command file (output to DDname CARDS):-

```
LOAD DATA
CHARACTERSET WE8EBCDIC500 BYTEORDER BIG
INFILE 'C:\FILE\PREFIX\SYSREC01.dat' "VAR 5"
BADFILE 'C:\ERR\PREFIX\SYSREC01.bad'
DISCARDFILE 'C:\ERR\PREFIX\SYSREC01.dsc'
INFILE 'C:\FILE\PREFIX\SYSREC02.dat' "VAR 5"
BADFILE 'C:\ERR\PREFIX\SYSREC02.bad'
DISCARDFILE 'C:\ERR\PREFIX\SYSREC02.dsc'
INFILE 'C:\FILE\PREFIX\SYSREC03.dat' "VAR 5"
BADFILE 'C:\ERR\PREFIX\SYSREC03.bad'
DISCARDFILE 'C:\ERR\PREFIX\SYSREC03.dsc'
REPLACE
  INTO TABLE TESTM.GTTM_SEG1
  WHEN (1:2) = '01'
   (
    SEG1_SEG1_ALPHA                          POSITION (00003:00016)
      CHAR
        NULLIF(00017)='?'                                  ,
    SEG1_SEG1_ACCUM                          POSITION (00018:00019)
      DECIMAL ( 3, 0)
        NULLIF(00020)='?'                                  ,
    SEG1_SEG1_STATUS                         POSITION (00021:00022)
      CHAR
        NULLIF(00023)='?'                                  ,
    SEG1_SEG1_STATE                          POSITION (00024:00025)
      CHAR
        NULLIF(00026)='?'                                  ,
    SEG1_SEG1_DOB                            POSITION (00027:00031)
      DECIMAL ( 9, 0)
        NULLIF(00032)='?'                                  ,
    SEG1_SEG1_SEX                            POSITION (00033:00033)
      CHAR
        NULLIF(00034)='?'                                  ,
```

```
  SEG1_SEG1_MARITAL                   POSITION (00035:00035)
    CHAR
      NULLIF(00036)='?'                                    ,
  SEG1_SEG1_OCCUP_CD                  POSITION (00037:00041)
    CHAR
      NULLIF(00042)='?'                                    ,
  SEG1_SEG1_EDUC_LVL                  POSITION (00043:00044)
    CHAR
      NULLIF(00045)='?'                                    ,
  SEG1_SEG1_OWN_RENT_CD               POSITION (00046:00046)
    CHAR
      NULLIF(00047)='?'
 )
INTO TABLE TESTM.GTTM_SEG2
WHEN (1:2) = '02'
 (
  SEG1_SEG1_ALPHA                     POSITION (00003:00016)
    CHAR
      NULLIF(00017)='?'                                    ,
  SEG1_SEG1_ACCUM                     POSITION (00018:00019)
    DECIMAL ( 3, 0)
      NULLIF(00020)='?'                                    ,
  SEG2_SEG2_EFF_DT                    POSITION (00021:00025)
    DECIMAL ( 9, 0)
      NULLIF(00026)='?'                                    ,
  SEG2_SEG2_EXP_DT                    POSITION (00027:00031)
    DECIMAL ( 9, 0)
      NULLIF(00032)='?'                                    ,
  SEG2_SEG2_STATE                     POSITION (00033:00034)
    CHAR
      NULLIF(00035)='?'
 )
INTO TABLE TESTM.GTTM_SEG3
WHEN (1:2) = '03'
 (
  SEG1_SEG1_ALPHA                     POSITION (00003:00016)
    CHAR
      NULLIF(00017)='?'                                    ,
  SEG1_SEG1_ACCUM                     POSITION (00018:00019)
    DECIMAL ( 3, 0)
      NULLIF(00020)='?'                                    ,
  SEG3_SEG3_ACCT_INST                 POSITION (00021:00023)
    DECIMAL ( 5, 0)
      NULLIF(00024)='?'                                    ,
  SEG3_SEG3_APPL                      POSITION (00025:00026)
    DECIMAL ( 2, 0)
```

```
      NULLIF(00027)='?'                                          ,
   SEG3_SEG3_BRANCH                          POSITION (00028:00030)
     DECIMAL ( 5, 0)
      NULLIF(00031)='?'                                          ,
   SEG3_SEG3_CLASS                           POSITION (00032:00033)
     DECIMAL ( 3, 0)
      NULLIF(00034)='?'
  )
```

Three data files (output to DDnames SYSREC01, SYSREC02 and SYSREC03), not reproduced here.

Appendix A - REC1 Copybook

```
      ****************************************************************

      *    RECORD TYPE 1 (CUSTOMER DATA)

      ****************************************************************

       01  REC1-RECORD.

           03  REC1-GRP-KEY.

               05  REC1-INST                 PIC 9(04) COMP.

               05  REC1-CUST.

                   07  REC1-ALPHA            PIC X(14).

                   07  REC1-ACCUM            PIC 9(03) COMP-3.

               05  REC1-CUST-KEY             REDEFINES REC1-CUST

                                             PIC X(16).

               05  REC1-RECTYPE              PIC X(01).

               05  REC1-FILLER               PIC X(19).

           03  REC1-COMMON-DATA.

               05  REC1-TYPE                 PIC X(01).

               05  REC1-STATUS               PIC X(01).

               05  REC1-HOUSEHOLD            PIC X(10).
```

```
           05  REC1-MAINT-TYPE            PIC 9(03).
           05  REC1-NAME                  PIC X(40).
           05  REC1-TIN                   PIC S9(10) COMP-3.
           05  REC1-BRANCH                PIC S9(05) COMP-3.
           05  REC1-CITY                  PIC X(35).
           05  REC1-STATE                 PIC X(02).
           05  REC1-PROV                  PIC X(02).
           05  REC1-ZIP                   PIC 9(05).
           05  REC1-POSTAL                PIC X(10).
           05  REC1-COUNTRY-CD            PIC X(02).
       03  REC1-DEMOGRAPHICS.
           07  REC1-DOB                   PIC S9(09) COMP-3.
           07  REC1-SEX                   PIC X(01).
           07  REC1-MARITAL               PIC X(01).
           07  REC1-OCCUP-CD              PIC X(05).
           07  REC1-HPHONE                PIC S9(10) COMP-3.
           07  REC1-BPHONE                PIC S9(10) COMP-3.
           07  REC1-EMPL-DT               PIC S9(09) COMP-3.
           07  REC1-EDUC-LVL              PIC X(02).
           07  REC1-INC-CLASS             PIC X(05).
           07  REC1-OWN-RENT-CD           PIC X(01).
      *-----------------------------------------------------------------*
```

Appendix B - REC2 Copybook

```
      ******************************************************************
```

```
*    RECORD TYPE 2 (CUST ALT ADDRESS)
*****************************************************************
 01  REC2-RECORD.
     03  REC2-GRP-KEY.
         05  REC2-INST                  PIC 9(04) COMP.
         05  REC2-CUST.
             07  REC2-ALPHA             PIC X(14).
             07  REC2-ACCUM             PIC 9(03) COMP-3.
         05  REC2-CUST-KEY              REDEFINES REC2-CUST
                                        PIC X(16).
         05  REC2-RECTYPE               PIC X(01).
         05  REC2-ALT-CODE              PIC X(01).
         05  REC2-FILLER                PIC X(18).
     03  REC2-CDM-GRP-DATA.
         05  REC2-EFF-DT                PIC S9(09) COMP-3.
         05  REC2-EXP-DT                PIC S9(09) COMP-3.
         05  REC2-CDM-DATA.
             07  REC2-CITY-STATE.
                 09  REC2-CITY          PIC X(35).
                 09  REC2-STATE         PIC X(02).
                 09  REC2-COUNTRY-CD    PIC X(02).
*---------------------------------------------------------------*
```

Appendix C - REC3 Copybook

```
*****************************************************************
 * RECORD TYPE 3 (ACCOUNT RELATION)
 *****************************************************************
 01 REC3-RECORD.

    03 REC3-GRP-KEY.

        05 REC3-INST PIC 9(04) COMP.

        05 REC3-CUST.

            07 REC3-ALPHA PIC X(14).

            07 REC3-ACCUM PIC 9(03) COMP-3.

        05 REC3-CUST-KEY REDEFINES REC3-CUST PIC X(16).

        05 REC3-RECTYPE PIC X(01).

        05 REC3-ACCOUNT.

            07 REC3-ACCT-INST PIC 9(04) COMP.

            07 REC3-APPL PIC 99.

            07 REC3-BRANCH PIC 9(05) COMP-3.

            07 REC3-CLASS PIC 9(03) COMP-3.

            07 REC3-ACCT PIC 9(18) COMP.

        05 REC3-ACCT-KEY REDEFINES REC3-ACCOUNT PIC X(17).

*---------------------------------------------------------------*
```

Appendix D - SEG1 Copybook

```
      ****************************************************************

      *    SEGMENT TYPE 1 (CUSTOMER DATA)

      ****************************************************************

       01  SEG1.

           03  SEG1-GRP-KEY.
```

```
    05  SEG1-INST                  PIC 9(04) COMP.

    05  SEG1-CUST.

        07  SEG1-ALPHA             PIC X(14).

        07  SEG1-ACCUM             PIC 9(03) COMP-3.

    05  SEG1-CUST-KEY              REDEFINES SEG1-CUST

                                   PIC X(16).

03  SEG1-COMMON-DATA.

    05  SEG1-TYPE                  PIC X(01).

    05  SEG1-STATUS                PIC X(01).

    05  SEG1-HOUSEHOLD             PIC X(10).

    05  SEG1-MAINT-TYPE            PIC 9(03).

    05  SEG1-NAME                  PIC X(40).

    05  SEG1-TIN                   PIC S9(10) COMP-3.

    05  SEG1-BRANCH                PIC S9(05) COMP-3.

    05  SEG1-CITY                  PIC X(35).

    05  SEG1-STATE                 PIC X(02).

    05  SEG1-PROV                  PIC X(02).

    05  SEG1-ZIP                   PIC 9(05).

    05  SEG1-POSTAL                PIC X(10).

    05  SEG1-COUNTRY-CD            PIC X(02).

03  SEG1-DEMOGRAPHICS.

    07  SEG1-DOB                   PIC S9(09) COMP-3.

    07  SEG1-SEX                   PIC X(01).

    07  SEG1-MARITAL               PIC X(01).

    07  SEG1-OCCUP-CD              PIC X(05).
```

```
          07  SEG1-HPHONE                PIC S9(10) COMP-3.

          07  SEG1-BPHONE                PIC S9(10) COMP-3.

          07  SEG1-EMPL-DT               PIC S9(09) COMP-3.

          07  SEG1-EDUC-LVL              PIC X(02).

          07  SEG1-INC-CLASS             PIC X(05).

          07  SEG1-OWN-RENT-CD           PIC X(01).

*-----------------------------------------------------------------*
```

Appendix E - SEG2 Copybook

```
*****************************************************************

*    SEGMENT TYPE 2 (CUST ALT ADDRESS)

*****************************************************************

 01  SEG2.

     03  SEG2-ALT-CODE              PIC X(01).

     03  SEG2-CDM-GRP-DATA.

         05  SEG2-EFF-DT                PIC S9(09) COMP-3.

         05  SEG2-EXP-DT                PIC S9(09) COMP-3.

         05  SEG2-CDM-DATA.

             07  SEG2-CITY-STATE.

                 09  SEG2-CITY          PIC X(35).

                 09  SEG2-STATE         PIC X(02).

                 09  SEG2-COUNTRY-CD    PIC X(02).

*-----------------------------------------------------------------*
```

Appendix F - SEG3 Copybook

```
      ****************************************************************

      *    SEGMENT TYPE 3 (ACCOUNT RELATION)

      ****************************************************************

       01  SEG3.

           03  SEG3-ACCOUNT.

               05  SEG3-ACCT-INST          PIC 9(04) COMP.

               05  SEG3-APPL               PIC 99.

               05  SEG3-BRANCH             PIC 9(05) COMP-3.

               05  SEG3-CLASS              PIC 9(03) COMP-3.

               05  SEG3-ACCT               PIC 9(18) COMP.

           03 SEG3-ACCT-KEY                REDEFINES SEG3-ACCOUNT

                                            PIC X(17).

      *---------------------------------------------------------------*
```

Appendix G - Single File AFL

```
0,LOGICALFILE=SINGLE_FILE,0.01,2014.02.18 13:04:12,HEADER=N,TRAILER=N,,,,,

1,RECNAME=REC1_RECORD,,,,,,,,,

3,REC1_RECORD,195,0,1,Structure,,,,,

3,REC1_GRP_KEY,38,0,2,Structure,,,,,

3,REC1_INST,2,0,3,"BinaryUnsigned:5,0",,,,,

3,REC1_CUST,16,2,3,Structure,,,,,

3,REC1_ALPHA,14,2,4,String,,,,,

3,REC1_ACCUM,2,16,4,"PackedUnsigned:3,0",,,,,
```

```
3,REC1_CUST_KEY,16,2,3,String,,Redef:REC1_CUST,,,

3,REC1_RECTYPE,1,18,3,String,,,,,

3,REC1_FILLER,19,19,3,String,,,,,

3,REC1_COMMON_DATA,120,38,2,Structure,,,,,

3,REC1_TYPE,1,38,3,String,,,,,

3,REC1_STATUS,1,39,3,String,,,,,

3,REC1_HOUSEHOLD,10,40,3,String,,,,,

3,REC1_MAINT_TYPE,3,50,3,"Numeric:999",,,,,

3,REC1_NAME,40,53,3,String,,,,,

3,REC1_TIN,6,93,3,"PackedSigned:10,0",,,,,

3,REC1_BRANCH,3,99,3,"PackedSigned:5,0",,,,,

3,REC1_CITY,35,102,3,String,,,,,

3,REC1_STATE,2,137,3,String,,,,,

3,REC1_PROV,2,139,3,String,,,,,

3,REC1_ZIP,5,141,3,"Numeric:99999",,,,,

3,REC1_POSTAL,10,146,3,String,,,,,

3,REC1_COUNTRY_CD,2,156,3,String,,,,,

3,REC1_DEMOGRAPHICS,37,158,2,Structure,,,,,

3,REC1_DOB,5,158,3,"PackedSigned:9,0",,,,DATEFORMAT:YYYYMMDD,

3,REC1_SEX,1,163,3,String,,,,,

3,REC1_MARITAL,1,164,3,String,,,,,

3,REC1_OCCUP_CD,5,165,3,String,,,,,

3,REC1_HPHONE,6,170,3,"PackedSigned:10,0",,,,,

3,REC1_BPHONE,6,176,3,"PackedSigned:10,0",,,,,

3,REC1_EMPL_DT,5,182,3,"PackedSigned:9,0",,,,,
```

```
3,REC1_EDUC_LVL,2,187,3,String,,,,,

3,REC1_INC_CLASS,5,189,3,String,,,,,

3,REC1_OWN_RENT_CD,1,194,3,String,,,,,
```

Appendix H - Multi File AFL

```
0,LOGICALFILE=MULTI_FILE,0.01,2014.02.18 15:31:48,HEADER=N,TRAILER=N,,,,,

1,RECNAME=REC1_RECORD, ,,,,,,,,

2,REC1_RECTYPE,1,18,3,String,,EQ,1,,

3,REC1_RECORD,195,0,1,Structure,,,,,

3,REC1_GRP_KEY,38,0,2,Structure,,,,,

3,REC1_INST,2,0,3,"BinaryUnsigned:5,0",,,,,

3,REC1_CUST,16,2,3,Structure,,,,,

3,REC1_ALPHA,14,2,4,String,,,,,

3,REC1_ACCUM,2,16,4,"PackedUnsigned:3,0",,,,,

3,REC1_CUST_KEY,16,2,3,String,,Redef:REC1_CUST,,,

3,REC1_RECTYPE,1,18,3,String,,,,,

3,REC1_FILLER,19,19,3,String,,,,,

3,REC1_COMMON_DATA,120,38,2,Structure,,,,,

3,REC1_TYPE,1,38,3,String,,,,,

3,REC1_STATUS,1,39,3,String,,,,,

3,REC1_HOUSEHOLD,10,40,3,String,,,,,

3,REC1_MAINT_TYPE,3,50,3,"Numeric:999",,,,,
```

```
3,REC1_NAME,40,53,3,String,,,,,

3,REC1_TIN,6,93,3,"PackedSigned:10,0",,,,,

3,REC1_BRANCH,3,99,3,"PackedSigned:5,0",,,,,

3,REC1_CITY,35,102,3,String,,,,,

3,REC1_STATE,2,137,3,String,,,,,

3,REC1_PROV,2,139,3,String,,,,,

3,REC1_ZIP,5,141,3,"Numeric:99999",,,,,

3,REC1_POSTAL,10,146,3,String,,,,,

3,REC1_COUNTRY_CD,2,156,3,String,,,,,

3,REC1_DEMOGRAPHICS,37,158,2,Structure,,,,,

3,REC1_DOB,5,158,3,"PackedSigned:9,0",,,,DATEFORMAT:YYYYMMDD,

3,REC1_SEX,1,163,3,String,,,,,

3,REC1_MARITAL,1,164,3,String,,,,,

3,REC1_OCCUP_CD,5,165,3,String,,,,,

3,REC1_HPHONE,6,170,3,"PackedSigned:10,0",,,,,

3,REC1_BPHONE,6,176,3,"PackedSigned:10,0",,,,,

3,REC1_EMPL_DT,5,182,3,"PackedSigned:9,0",,,,,

3,REC1_EDUC_LVL,2,187,3,String,,,,,

3,REC1_INC_CLASS,5,189,3,String,,,,,

3,REC1_OWN_RENT_CD,1,194,3,String,,,,,

1,RECNAME=REC2_RECORD, ,,,,,,,,

2,REC2_RECTYPE,1,18,3,String,,EQ,2,,

3,REC2_RECORD,87,0,1,Structure,,,,,

3,REC2_GRP_KEY,38,0,2,Structure,,,,,

3,REC2_INST,2,0,3,"BinaryUnsigned:5,0",,,,,
```

```
3,REC2_CUST,16,2,3,Structure,,,,,

3,REC2_ALPHA,14,2,4,String,,,,,

3,REC2_ACCUM,2,16,4,"PackedUnsigned:3,0",,,,,

3,REC2_CUST_KEY,16,2,3,String,,Redef:REC2_CUST,,,

3,REC2_RECTYPE,1,18,3,String,,,,,

3,REC2_ALT_CODE,1,19,3,String,,,,,

3,REC2_FILLER,18,20,3,String,,,,,

3,REC2_CDM_GRP_DATA,49,38,2,Structure,,,,,

3,REC2_EFF_DT,5,38,3,"PackedSigned:9,0",,,,DATEFORMAT:YYYYMMDD,

3,REC2_EXP_DT,5,43,3,"PackedSigned:9,0",,,,DATEFORMAT:YYYYMMDD,

3,REC2_CDM_DATA,39,48,3,Structure,,,,,

3,REC2_CITY_STATE,39,48,4,Structure,,,,,

3,REC2_CITY,35,48,5,String,,,,,

3,REC2_STATE,2,83,5,String,,,,,

3,REC2_COUNTRY_CD,2,85,5,String,,,,,

1,RECNAME=REC3_RECORD, ,,,,,,,,

2,REC3_RECTYPE,1,18,3,String,,EQ,3,,

3,REC3_RECORD,36,0,1,Structure,,,,,

3,REC3_GRP_KEY,36,0,2,Structure,,,,,

3,REC3_INST,2,0,3,"BinaryUnsigned:5,0",,,,,

3,REC3_CUST,16,2,3,Structure,,,,,

3,REC3_ALPHA,14,2,4,String,,,,,

3,REC3_ACCUM,2,16,4,"PackedUnsigned:3,0",,,,,

3,REC3_CUST_KEY,16,2,3,String,,Redef:REC3_CUST,,,
```

```
3,REC3_RECTYPE,1,18,3,String,,,,,

3,REC3_ACCOUNT,17,19,3,Structure,,,,,

3,REC3_ACCT_INST,2,19,4,"BinaryUnsigned:5,0",,,,,

3,REC3_APPL,2,21,4,"Numeric:99",,,,,

3,REC3_BRANCH,3,23,4,"PackedUnsigned:5,0",,,,,

3,REC3_CLASS,2,26,4,"PackedUnsigned:3,0",,,,,

3,REC3_ACCT,8,28,4,"BinaryUnsigned:21,0",,,,,

3,REC3_ACCT_KEY,17,19,3,String,,Redef:REC3_ACCOUNT,,,
```

Appendix I - IMS AFL

```
0,LOGICALFILE=IMS,0.01,2014.02.18 17:06:50,HEADER=N,TRAILER=N,,,,,

1,RECNAME=SEG1,,,,,,,,,

2,SEGNAME,8,0,1,String,,EQ,SEG1,,

3,SEGNAME,8,0,1,String,,,,,

3,SEG1,175,8,1,Structure,,,,,

3,SEG1_GRP_KEY,18,8,2,Structure,,,,,

3,SEG1_INST,2,8,3,"BinaryUnsigned:5,0",,,,,

3,SEG1_CUST,16,10,3,Structure,,,,,

3,SEG1_ALPHA,14,10,4,String,,,,,

3,SEG1_ACCUM,2,24,4,"PackedUnsigned:3,0",,,,,

3,SEG1_CUST_KEY,16,18,3,String,,Redef:SEG1_CUST,,,

3,SEG1_COMMON_DATA,120,34,2,Structure,,,,,

3,SEG1_TYPE,1,34,3,String,,,,,

3,SEG1_STATUS,1,35,3,String,,,,,

3,SEG1_HOUSEHOLD,10,36,3,String,,,,,
```

```
3,SEG1_MAINT_TYPE,3,46,3,"Numeric:999",,,,,

3,SEG1_NAME,40,49,3,String,,,,,

3,SEG1_TIN,6,89,3,"PackedSigned:10,0",,,,,

3,SEG1_BRANCH,3,95,3,"PackedSigned:5,0",,,,,

3,SEG1_CITY,35,98,3,String,,,,,

3,SEG1_STATE,2,133,3,String,,,,,

3,SEG1_PROV,2,135,3,String,,,,,

3,SEG1_ZIP,5,137,3,"Numeric:99999",,,,,

3,SEG1_POSTAL,10,142,3,String,,,,,

3,SEG1_COUNTRY_CD,2,152,3,String,,,,,

3,SEG1_DEMOGRAPHICS,37,154,2,Structure,,,,,

3,SEG1_DOB,5,154,3,"PackedSigned:9,0",,,,,

3,SEG1_SEX,1,159,3,String,,,,,

3,SEG1_MARITAL,1,160,3,String,,,,,

3,SEG1_OCCUP_CD,5,161,3,String,,,,,

3,SEG1_HPHONE,6,166,3,"PackedSigned:10,0",,,,,

3,SEG1_BPHONE,6,172,3,"PackedSigned:10,0",,,,,

3,SEG1_EMPL_DT,5,178,3,"PackedSigned:9,0",,,,,

3,SEG1_EDUC_LVL,2,183,3,String,,,,,

3,SEG1_INC_CLASS,5,185,3,String,,,,,

3,SEG1_OWN_RENT_CD,1,190,3,String,,,,,

1,RECNAME=SEG2,SEG1,,,,,,,,

2,SEGNAME,8,0,1,String,,EQ,SEG2,,

3,SEGNAME,8,0,1,String,,,,,
```

```
3,SEG2,50,8,1,Structure,,,,,

3,SEG2_ALT_CODE,1,8,2,String,,,,,

3,SEG2_CDM_GRP_DATA,49,9,2,Structure,,,,,

3,SEG2_EFF_DT,5,9,3,"PackedSigned:9,0",,,,,

3,SEG2_EXP_DT,5,14,3,"PackedSigned:9,0",,,,,

3,SEG2_CDM_DATA,39,19,3,Structure,,,,,

3,SEG2_CITY_STATE,39,19,4,Structure,,,,,

3,SEG2_CITY,35,19,5,String,,,,,

3,SEG2_STATE,2,54,5,String,,,,,

3,SEG2_COUNTRY_CD,2,56,5,String,,,,,

1,RECNAME=SEG3,SEG1,,,,,,,,

2,SEGNAME,8,0,1,String,,EQ,SEG3,,

3,SEGNAME,8,0,1,String,,,,,

3,SEG3,17,8,1,Structure,,,,,

3,SEG3_ACCOUNT,17,8,2,Structure,,,,,

3,SEG3_ACCT_INST,2,8,3,"BinaryUnsigned:5,0",,,,,

3,SEG3_APPL,2,10,3,"Numeric:99",,,,,

3,SEG3_BRANCH,3,12,3,"PackedUnsigned:5,0",,,,,

3,SEG3_CLASS,2,15,3,"PackedUnsigned:3,0",,,,,

3,SEG3_ACCT,8,17,3,"BinaryUnsigned:21,0",,,,,

3,SEG3_ACCT_KEY,17,16,2,String,,Redef:SEG3_ACCOUNT,,,
```

How to Parse IMS Database Copybooks and Mask Data

The purpose of the CA TDM FM for IMS Integrator is to extract and mask data from an IMS database without having to work on the mainframe yourself. CA TDM FM for IMS Integrator uses CA File Master Plus for IMS to access IMS

databases on the mainframe, and it uses CA Mainframe Datamaker to mask the extracted IMS data. The Data Collector service for z/OS with the TDM FM Integration plugin drives the interactions between the components of this solution.

Work together with your System Programmer and Database Admin to initialize the setup once for each user.

Masking an IMS database involves 3 steps:

1. Extracting data from the source IMS database into a flat file.
2. Masking the data in the flat file.
3. Reloading the masked data into the target IMS database.

To process flat files in CA TDM Datamaker, you have to register the record layouts for the files. These definition files are suffixed *DM.txt. You can create the DM.txt definition files manually. The z/OS files might come with custom COBOL or PL1 declaration copybooks that provide the definitions.

If you are masking more than one segment type in the database, the flat file that contains the extracted data is a multi-format file, in which one record represents an instance of a segment. For such a multi-format file, an Advanced File Layout file contains record definitions for all record types in one file. A layout file is suffixed *AFL.DM.txt.

TIP

For your convenience, your Test Data Manager installation includes the required utilities as Windows executables and as generic .jar files. The Integrator is packaged with the File Definition Manager component. The parser is written in Java and requires a JVM (Java Virtual Machine) to run. The executables do not require any installation steps.

- CATDMFMforIMSIntegrator.exe and .jar
- File Definition Manager -- FileDefinitionManager.exe and .jar

Follow these steps:

Verify Prerequisites

Work with your System Programmer to fulfill the following prerequisites.

1. Verify that you have a recent JVM (Java Virtual Machine) installed to run the utilities.
2. Install and deploy the Data Collector. For more information, see Deploy and Configure the Data Collector for z/OS.
3. Find the JOB card template in the `templates` directory of the Data Collector for IMS home directory. CA provides this JOB card template to generate valid JCLs.
4. Review the JOB card template and adapt it to your system and security environment, if needed. This template applies to all users.
   Tip: ISPF option 3.17 lists the z/OS UNIX directory in a way that is easy to navigate.
5. Provide a TDM FM User Space for each user of the integration service to store per-user configuration, including created JCLs, AFL files, MAP files. The TDM FM User Space instance ID is the high-level qualifier (HLQ) of these user-specific data sets.
   Allocate the following data sets by hand for each user:
   – *HLQ*.RUNJCL – Stores all JCLs – library (fixed block, record length 80)
   – *HLQ*.FM.REPT – Stores reports of CA File Master extract or reload operations – library (fixed block ANSI (FBA), record length 133)
   – *HLQ*.TDM.AFL – is the data set for Advanced File Layout files – library (fixed block, record length 120)
   – *HLQ*.TDM.MAP – is the data set for Transformation Map files – library (fixed block, record length 255)
   – *HLQ*.DB.*dbdName* – the data set for the extracted database data – sequential
6. Work with your Database Admin to locate and prepare your copybooks. For more information, see Create an Advanced File Layout (AFL) with File Definition Manager.

NOTE

Note: The generated jobs are submitted in the security context of the user who runs the utilities.

Create Connection Profile

You create connection profiles to store commonly used connections to remote IMS Databases. You can create source and target connection profiles, so that you can extract from, and import data into, the same or different databases.

Follow these steps:

1. Launch the CA TDM FM for IMS Integrator from your local computer.
2. Open the **Connection Profile** tab and specify the following information:
   - **Connection Profile List**
     Select either "No Connection Profile" or "Select Connection Profile" and fill in the fields to create a connection profile.
   - **Connection Profile Name**
     Defines the name under which to save this connection profile.
   - **Data Set Prefix**
     Defines the high level qualifier of your TDM FM User Space.
   - **Server Name**
     Defines the server name where Data Collector for z/OS is installed.
   - **Port Number**
     Defines the port number of the server where Data Collector for z/OS is installed.
   - **User Name**
     Defines the user name for your TDM FM user space.
   - **Password**
     Defines your password.
   - **Sysplex Name**
     Defines the name of the sysplex where the Data Collector for z/OS is running.
   - **z/OS Name**
     Defines the LPAR name where the Data Collector for z/OS is running.
   - **File Master Clist**
     Defines the CA File Master executable that you want to run.
     Click **GET IMS ENV List** to retrieve the list of IMS Environments to populate the following two drop-downs.
   - **IMS ENV DSN**Defines the partitioned data set name with IMS environments definitions in its members.
   - **IMS ENV List**Select the IMS environment that contains the database definition.
   - **Access Database**Defines whether to access the database using Program Specification Blocks (PSB) or Database Descriptors (DBD). Ask your DB Admin for details.
     - **Using PSB** — Click **Get PSB List** to select a database. **PSB List** defines the name of the Program Specification Block which contains the Program Communication Block (PCB) for the selected database. Click **Get PCB List** to select a PCB.
     - **Using DBD** — Click **Get Database List** to select a database.
   - **Account Code**(Optional) Defines the account code, if your JOB card needs one.
   - **Programmer Name**(Optional) Defines the programmer name, if your JOB card needs it.
3. Click **New**.
   The connection profile for the IMS database server is stored.

Edit a Connection Profile

For details on the fields, see Create a Connection Profile.

1. Launch the CA TDM FM for IMS Integrator from your local computer.

2. Open the **Connection Profile** tab.
3. Select an existing profile from the **Connection Profile List**. The list is empty if you have not created any connection profiles yet.
4. Edit the fields and click **Save**.

Delete a Connection Profile

1. Launch the CA TDM FM for IMS Integrator from your local computer.
2. Open the **Connection Profile** tab.
3. Select the profile from the **Connection Profile List**. The list is empty if you have not created any connection profiles yet.
4. Click **Delete**.

Get Remote Copybook Extract

The CA TDM FM for IMS Integrator uses connection profiles to store connection details for your IMS database server. This remote connection lets you get all unparsed copybook extract files under a given IMS layout data set name. Contact your Database Admin for details.

Follow these steps:

1. Launch the CA TDM FM for IMS Integrator from your local computer.
2. Specify the following information under the **Get Remote Copybook Extract** tab:
   - **IMS Layout DSN**
     Defines the IMS Layout Data Set Name where your copybooks are stored on the mainframe.
   - **Select Connection Profile**
     Defines the connection profile that connects to the remote IMS database server.
3. Click **Get Copybook Extract**
   Wait for the extract to complete, and then click **OK** in the success message dialog.
4. Click **Parse Copybooks**.
   The File Definition Manager utility opens. The copybook location that you defined in the IMS Integrator is passed on to the File Definition Manager configuration.
5. Use the File Definition Manager to generate AFL files. Before you can use them, use File Definition Manager to review and prepare them.
   For more information, see Create an Advanced File Layout (AFL) with File Definition Manager.

The File Definition Manager creates the following files:

- ***_DG_Source.AFL.DM.txt** — The AFL file that defines the record layout according to the ASCII layout.
- ***_zOS.AFL.DM.txt** — The EBCDIC version of the AFL file that is used in Mainframe Datamaker.
- **XLS file —** The spreadsheet contains one sheet per record type in the parsed copybooks parse. Use this spreadsheet to register the File Definition in Datamaker.

Register Parsed Copybook (Advanced File Layout)

Follow these steps:

1. Verify the AFL files before you use them. Make sure you have edited the layouts to add information that the parser cannot derive. Make these edits before you register the layouts or transfer them to z/OS.
2. Register the zOS.AFL.DM.txt files in Datamaker. These files are used to design file masking and subsetting.
3. Transfer the zOS.AFL.DM.txt layouts to z/OS to execute file masking and subsetting. The masking and subsetting programs read these files in z/OS.

TIP

DG_Source.AFL.DM.txt facilitates file conversion between mainframe and windows code pages. For more information, see Mainframe File Conversion.

Register the Advanced File Layout in TDM Datamaker

1. Launch CA TDM Datamaker.
2. Select a project version context.
3. Click **Project, Register**.

Select one of the following Copybook Parser Layouts to register to Datamaker:

- **File generation**
  Register **File Definition from G-T Excel file**
- **File Masking and subsetting**
  Register **Advanced File Layout from G-T text file**

Register File Definition from G-T Excel File

For files that contain a single record type:

1. Open the *.csv file in the layout directory in Excel.
2. Save as Excel 97-2003 workbook.
   Note: Save the file as *.xls, not as *.xlsx.

For files that contain multiple record types:

1. Open `LoadCSV.xls` in the z/OS_CopybookParser directory. This spreadsheet contains an Import CSVs macro. Use this macro to import all CSVs in cell A1 of the directory.
2. Enter the directory that contains the CSVs into cell A1, and select **Import CSVs**.
   A new spreadsheet opens with all of the CSVs that are imported in their own worksheet. An extra row in A1 is inserted in the first worksheet. This row provides extra information about the file.
   This file contains the following parameters:
   – PARAMETERS
   – HEADER=N
   – TRAILER=N
   – STYLE=COMPLEX,ID_OFFSET=
   – ID_LENGTH=1
3. Modify the following parameters to define the record identifier in the file:
   OFFSET
   Defines the start position of the record identifier
   ID_LENGTH =
   Defines the length of the record identifier.
   Note: In this example, the record identifier is `REC_ID` , so `OFFSET = 1` and `ID_LENGTH=1` .
4. Save the spreadsheet as" Excel 97-2003 workbook (*.xls, not as *.xlsx).
   To register single records or multiple record files into Datamaker with the correct Project in the context, select Projects, Register.
5. Proceed to Register Advanced Layout.

Register Advanced File Layout from G-T Text File

1. Select **Advanced File Layout from G-T text file** and click the green arrow to the right of **Select Type**.
2. Click the ellipsis (...) at the end of the **Please Select the File to Register** field.

3. Browse for the AFL to register, and select **Show Record Types** to show the records types within the selected AFL. You are prompted with options to clear and register each Record Type.
4. Use the "**Register**" button to register all record types and proceed to the **Advanced file Layout** screen.
   Note: You can also access the **Advanced file Layout** screen through the **Menu bar**, **Tools**, **Manage Advanced File Layouts**.
5. Select the first record type to display the Fields for that record.
6. Highlight all the record fields that require masking. Use the arrows to move the fields within the **Selected Fields** list.
   Note: All fields are typically selected. If you moved multiple fields at once, the screen might refresh several times.
7. Select the fields and click **Register as table** to register the record type and the fields as a table.
8. Select the next Record Type and repeat the process.

Extract Data from IMS Database

Using a connection profile that establishes connection between the CA TDM FM for IMS Integrator and IMS database server, you extract the data from specified IMS Database so that you mask the data and then reload the masked data to the same database or to another database.

Follow these steps:

1. Launch the CA TDM FM for IMS Integrator and open the **Extract Data** tab.
2. Select a **Connection Profile** that establishes the remote connection to the source IMS Database.
3. Specify the following parameters in the **Fill Details to Extract Data** section. Consult with your CA File Master Plus Admin.
   - **Job Name**
     Defines the job named needed for your JOB card. Limit: seven characters.
   - **IMS Layout DSN**(Optional) Defines the data set where copybooks (layouts for your database) are stored.
     Custom Record Layout DSN
     (Optional) Defines the DSN of the partitioned data set in which Custom Record Layouts are stored.
   - **Segment(s)**
     (Optional) Defines a comma separated list of only the segments which you want to extract.
   - **Selection Criteria**(Optional) Defines the selection criteria for optional filters. See CA File Master Plus documentation.
   - **Segment CrossRef**(Optional) Defines the Data Set Name of the partitioned data set in which Segment Cross-Reference parm members are stored.
   - **DSN Lists**(Optional) Defines the Data Set Name List PDS. Each DSN List is a list of DSNs saved to a member in the DSN List PDS. You use DSN Lists to resolve DSN fields on any of the product's panels.
4. Click the **Create Job Definition** button. The Utility creates job definitions.
5. Select the **Job Definition ID** in the **Submit Extract Job** section, and then click the **Submit Job** button.
6. Verify the **Job Status**:
   - **Job Definition ID**
   - **Job Instance ID**
   - **Job Status**
7. Click **Refresh Job Status** until the utility confirms that the Job Status is completed. The data is extracted on the mainframe.
8. Click **Get Job Output** to save a copy of the job log in a text file on your local computer for further review.
9. Click **Open Directory** to access the file directory where the job log is placed.

Mask the Extracted Data

You can now mask the data that you extracted from the IMS Database, according to the masking rules that you generated into the CSV file using TDM Datamaker.

Follow these steps:

1. Open the **Mask Data** tab in the CA TDM FM for IMS Integrator.
2. Select a Connection Profile that establishes the remote connection to the source IMS Database.
3. Specify the following parameters under the **Fill Details to Mask Data** section:
   - **Job Name**
     Defines the job name needed for the JOB card.
   - **TDM Masking Data Set Prefix**
     Defines the high-level qualifier (HLQ) of data sets where Mainframe Datamaker is installed. For more information, see the Mainframe Installation and Upgrade section.
   - **AFL Member Name**
     Defines the target member name into which the AFL CSV file is transferred. eight characters.
   - **Map Member Name**
     Defines the target member name into which the transformation map CSV file is transferred. eight characters.
   - **Common Options**
     Specifies optional common options.
     - **Primary Cyls Masked** — defines the space for the output masked data file.
       Type: Integer
       Default: Define the default value in the *tdmId*.PROCLIB(GTMSKVS) JCL.
     - **Secondary Cyls Masked** — defines the space for the output masked data file.
       Type: Integer
       Default: Define the default value in the *tdmId*.PROCLIB(GTMSKVS) JCL.
     - **Trimmed XRef** — specifies whether to trim values before cross-reference lookup.
       Type: Boolean
       Default: false
     - **Trimmed HashLov** — specifies whether to trim values before using in HASHLOV function.
       Type: Boolean
       Default: false
     - **Case InsensitiveXRef** — specifies whether to do cross-reference lookups case-insensitively.
       Type: Boolean
       Default: false
   - **Advanced Options**
     Specifies optional advanced options.
     - **Primary Cyls Audit** — defines the space for the output audit file. Type: Integer.
       Default: Define the default value in the *tdmId*.PROCLIB(GTMSKVS) JCL.
     - **Secondary Cyls Audit** — defines the space for the output audit file. Type: Integer.
       Default: Define the default value in the *tdmId*.PROCLIB(GTMSKVS) JCL.
     - **Base Century** — defines the base century of the year. Only required if you use abbreviated dates with a one-digit century. Type: Integer
     - **AFL Quote Style** — defines the quote style to use in the AFL file. Options are SINGLE or DOUBLE.
       Type: String
       Default: DOUBLE
     - **Map Quote Style** — defines the quote style to use in the map file. Options are SINGLE or DOUBLE.
       Type: String
       Default: DOUBLE
     - **Language** — defines the language for output messages. Options are EN, DE, ES, IT. FR is not available.
       Type: String
       Default: EN
     - **Page Limit** — defines the maximum number of pages produced in audit and report output files.
       Type: Integer

Default: 50

- **Bad DateString** — defines the value to use in place of fields which are supposed to contain a valid date but do not. Type: String
- **Current Date** — defines the current date to use in processing. Format: CCYYMMDD.
  Type: String
  Default: the system date
- **Low Date** — defines the minimum date to be output during masking.
  Format: dd/mm/yyyy
  Type: String
  Default: 01/01/1800
- **High Date** — defines the maximum date to be output during masking.
  Format: dd/mm/yyyy
  Type: String
  Default: 31/12/2200
- **Hash Type** — defines the type of hashing to use, either "ASM" or "JAVA". ASM is much faster. JAVA (written in COBOL) is slower, but matches the algorithm used by FDM.
  Type: String
  Default: "ASM"
- **Commit** — defines the database commit frequency.
  Type: Integer
  Default: 1000
- **Process Count** — defines the number of records to mask.
  Type: Integer
- **Case Insensitive Seed** — specifies whether to match *LOV1 lookups case insensitively.
  Type: Boolean
  Default: false
- **Case Insensitive HashLov** — specifies whether to return the same HashLov value for strings which match, regardless of case.
  Type: BooleanDefault: false
- **Case Insensitive Hash** — specifies whether to return the same hash value for strings which match, regardless of case.
  Type: Boolean
  Default: false
- **Blank As Null** — specifies whether to treat blank fields as null.
  Type: Boolean
  Default: false
- **Keep Invalid** — specifies not to mask fields that contain invalid data.
  Type: BooleanDefault: false

– **Diagnostic Options**
Specifies optional diagnostic options.

- **Audit** — defines the audit frequency. We recommended using ALL only when testing masking, due to the volume of output.
  Type: String
  Options: "`ALL`" or "`ROW`*j* " or "`SAMPLE`*j* ", where *j* is an integer
- **Report Invalid** — specifies whether to report invalid values as they are found during masking.
  Type: Boolean
  Default false.
- **Validate Only** — specifies not to apply masking, and to validate merely program inputs.
  Type: Boolean
  Default: false

4. Click the **Create Job Definition** button.

5. Specify the following parameters under the **Submit Mask Job** section:
   a. **Job Definition ID**
      Select the job.
   b. **Advance File Layout Location**
      Click **Browse ADL File** to select a file from your hard drive.
   c. **Transformation Map File Location**
      Click **Browse Transformation Map** and upload the Transformation Map File.
6. Click **Submit Job**. The utility reads the files, saves them to the members on the mainframe, and submits the job.
7. Verify the Job Status section for the following:
   – **Job Definition ID**
   – **Job Instance ID**
   – **Job Status**
8. Click **Refresh Job Status** until the utility confirms that the Job Status is completed.
9. Click **Get Job Output** to save the copy of job log in a text file on your local computer. Part of the job log is the masking report and audit.
10. Click **Open Directory** to access the file directory where the copy of job log is placed.

Reload Data into IMS Database

After you have masked the data that you extracted from the IMS Database, you can reload that data into the same source database, or to another target database to maintain both the original data and the masked data.

Follow these steps:

1. Open the **Reload Data** tab in the CA TDM FM for IMS Integrator.
2. Select a **Source Connection Profile** that establishes the remote connection to the source IMS Database that has the masked data.
3. Select a **Target Connection Profile** that establishes the remote connection to the target IMS Database where you want to place the masked data.
   Tip: If you want to overwrite the original data in the source database with the masked data, select the same connection profile for both source and target.
4. Specify the following parameters under the **Fill Details to Mask Data** section:
   – **Job Name**
      Defines the name of the JOB card.
      Limit: seven characters.
   – **IMS Layout DSN**
      (Optional) Defines the data set where layouts for our database are stored.
   – **Custom Record Layout DSN**
      (Optional) Defines the data set name of the partitioned data set in which Custom Record Layouts are stored.
   – **Segment CrossRef**
      (Optional) Defines the data set name of the partitioned data set in which Segment Cross-Reference parm members are stored.
   – **DSN Lists**
      (Optional) Defines the Data Set Name List PDS. Each DSN List is a list of DSNs saved to a member in the DSN List PDS. You use DSN Lists to resolve DSN fields on any of the product's panels.
5. Click **Create Job Definition**.
6. Select the **Job Definition ID** under the **Submit Reload Job** section and click **Submit Job**.
7. Verify the Job Status section for the following:
   – **Job Definition ID**
   – **Job Instance ID**
   – **Job Status**

8. Click **Refresh Job Status** until the utility confirms that the Job Status is completed.
9. Click **Get Job Output** to save the copy of the job log and report in a text file on your local computer.
10. Click **Open Directory** to access the file directory where the copy of job log is placed.

Masking Functions for Mainframe

This page details the masking functions that are currently available for use on the Mainframe:

Click to expand table of contents...

ADD

Add a fixed value in Parm1 to the original value.
Applies to Field Type: Numeric
Parm1 (Mandatory): Must contain a numeric value.

ADDDAYS

Add the number of days specified by Parm1 to the existing value.
Applies to Field Type: DB2 Date. Field with DATEFORMAT specified in the AFL.
Parm1 (Mandatory): Must contain an integer.

ADDPERCENT

Add the percentage specified by Parm1 to the existing numeric value.
Applies to Field Type: Numeric
Parm1 (Mandatory): Must contain a numeric value.

ADDRANDOM

Add a random number between Parm1 and Parm2 to the existing value.
Applies to Field: Type Numeric
Parm1 (Mandatory): Must contain a numeric value.
Parm2 (Mandatory): Must contain a numeric value greater than Parm1.

ADDRANDOMDAYS

Add a random number of days between Parm1 and Parm2 to the existing value.
Applies to Field Type: DB2 Date. Field with DATEFORMAT specified in the AFL.
Parm1 (Mandatory): Must contain a integer value.
Parm2 (Mandatory): Must contain a integer value greater than Parm1.

ADDRANDOMHOURS

Add a random number of hours between Parm1 and Parm2 to the existing value.
Applies to Field Type: DB2 Date. Field with DATEFORMAT specified in the AFL
Parm1 (Mandatory): Must contain a integer value.
Parm2 (Mandatory): Must contain a integer value greater than Parm1.

ADDRANDOMMINUTES

Add a random number of minutes between Parm1 and Parm2 to the existing value.
Applies to Field Type: DB2 Date
Note: Field with DATEFORMAT specified in the AFL.

Parm1 (Mandatory): Must contain a integer value.
Parm2 (Mandatory): Must contain a integer value greater than Parm1.

ADDRANDOMMONTHS

Add a random number of months between Parm1 and Parm2 to the existing value.
Applies to Field Type: DB2 Date. Field with DATEFORMAT specified in the AFL.
Parm1 (Mandatory): Must contain a integer value.
Parm2 (Mandatory): Must contain a integer value greater than Parm1.

ADDRANDOMSECONDS

Add a random number of seconds between Parm1 and Parm2 to the existing value.
Applies to Field Type: DB2 Date. Field with DATEFORMAT specified in the AFL.
Parm1 (Mandatory): Must contain a integer value.
Parm2 (Mandatory): Must contain a integer value greater than Parm1.

ADDRANDOMYEARS

Add a random number of years between Parm1 and Parm2 to the existing value.
Applies to Field Type: DB2 Date. Field with DATEFORMAT specified in the AFL.
Parm1 (Mandatory): Must contain a integer value.
Parm2 (Mandatory): Must contain a integer value greater than Parm1.

AMEXCARD

Generate a random American Express (AMEX) card number.
Applies to Field Type: Character
Required Parameters: None

ANAHTAR

Generate an account number from the data fields in the input parameters.
Applies to Field Type: Character
Parm1 (Mandatory): Must contain field name (or internal variable) containing First Name data.
Parm2 (Mandatory): Must contain field name (or internal variable) containing Last Name data.
Parm3 (Mandatory): Must contain field name (or internal variable) containing the Date of Birth data field. If file masking, the format must be YYYY-MM-DD

AND

Only applicable to flat files.
Note: See the Use of WHERE Clauses when Processing Flat Files section.
Applies to Field Type: All
Required Parameters: None

CHARHASH

Create a Hash value based on Parm1 (the hash key).
Applies to Field Type: Character
Parm1 (Mandatory): Must contain an integer as the hash key.

CHECKIBAN

Overwrites the IBAN check digit with the correct check digit value.

Note: To be used in after other masking functions masking specific sections of the IBAN. For example, account number.
Applies to Field Type: Character
Required Parameters: None

CHECKRUT

Social security numbers in Chile (RUT) follow a special format with a check digit at the end which is dependent on the first 8 digits of the number.
Note: This function populates the check digit only for the data field.
Applies to Field Type: Character
Parm1 (Mandatory): Must contain field name (or internal variable) containing the RUT.

COMBINEVALS

All mappings below this for the same table and column have their values concatenated and placed back in that column. Blanks are trimmed from the start and end of values before concatenation.
Note: This function will not work for masking fields inside array structures.
Applies to Field Type: Character
Required Parameters: None

CONCAT

Concatenate the values in Parm1 through Parm4.
Applies to Field Type: Character
Required Parameters: None, all optional
Parm1 (Optional): Literal value to appear at the start of the concatenation
Parm2 (Optional): Field name (or internal variable)
Parm3 (Optional): Literal value
Parm4 (Optional): Field name (or internal variable)

CREDITCARD

Mask the last five digits of a credit card number and recalculate the check digit.
Applies to Field Type: Character, Numeric
Required Parameters: None

CREDITCARDKEEPTYPE

Retain the first seven digits of an existing credit card number. The eighth digit is the value supplied in Parm1. All remaining digits are incremented each time the function is used.
Applies to Field Type: Character, Numeric
Parm1 (Mandatory): Integer

DELETE

Delete the row or record to which the function applies. In most cases, you use this function in conjunction with a selection function (for example, WHERE, AND or OR).
Applies to Field Type: Character, Numeric, DB2 Date
Required Parameters: None

DOB

Adjust a date, by between plus or minus the number of days given by Parm1. Without altering the age in comparison to the current data (as set by system date, or by the CDATE PARMCD option).
Applies to Field Type: DB2 Date. Field with DATEFORMAT specified in the AFL.

Parm1 (Optional): Integer value for range of number of days to affect the date.
Default: 30

DOD

Adjust a date, by between plus or minus the number of days given by Parm1. Without altering the age in comparison to the current data (as set by system date, or by the CDATE PARMCD option).
Applies to Field Type: DB2 Date. Field with DATEFORMAT specified in the AFL.
Parm1 (Optional): Integer value for range of number of days to affect the date.
Default: 30.

EMAIL

Generate a random email address.
Applies to Field Type: Character
Required Parameters: None

EXIT

Invoke a user-written masking function. For more information, see User Functions - Specification & Calling.
Applies to Field Type: Character, Numeric, DB2 Date
Parm1 (Mandatory): Name of the exit function

FILL

Fill the column with the value given in Parm1.
Applies to Field Type: Character
Parm1 (Mandatory): Character to be used as the filler

FIXED

Fill the column with the value given in Parm1.
Applies to Field Type: Character, Numeric, DB2 Date
Parm1 (Mandatory): Value to be assigned to the field. This must be a valid data value of the field. For example, there must be a numeric value if masking a numeric field.

FIXEDDAY

Fix the day part of a date to value in Parm1.
Applies to Field Type: DB2 Date. Field with DATEFORMAT specified in the AFL.
Parm1 (Mandatory): Integer value to be assigned to day value of field.

FORMATENCRYPT

Consistently mask the given column values with the original format. The function produces unique values as long as the original values are also unique, which makes it ideal for masking key columns.

Parameters

None of the following parameters are mandatory:

- PARM1
  (Optional) Specifies the number of start characters to ignore.
- PARM2
  (Optional) Specifies the number of end characters to ignore.
- PARM3

(Optional) (If PARM1 and PARM2 are not set) Specifies the number of start characters to mask.

- PARM4
  (Optional) (If PARM1 and PARM2 are not set) Specifies the number of end characters to mask.

Applies to: Characters and Numbers.

Review the following considerations:

- For numeric columns, FORMATENCRYPT ignores the first digit of input values. This is to avoid the generation of a masked value with leading zeroes, which databases typically truncate, and which can then become identical to another value.
  This rule does not apply to character columns, because databases do not truncate character values.
- This function does not mask the first occurrence of a lowercase character. It retains that letter *as is*. For example, `aBCd` to `aWKj` or `BaB` to `WaJ`.
  To address this issue, ensure that you enter a lowercase key for the LOWERCASEKEY masking option; for example, `htjugtvffc`. Additionally, verify that the key does not start with the character `a`.

FORMATENCRYPT1

Consistently masks the given column values with the original format. The function produces unique values as long as the original values are also unique, which makes it ideal for masking key columns.

This function works pretty much as FORMATENCRYPT, but it has an extended set of 20 group keys against the unique set of keys from FORMATENCRYPT. Each masking group key is split in upper, lower, and numeric keys in a random sequence of characters, giving in total 60 character keys to be used in the masking process. This function also allows the users to set a particular Masterkey to be mixed with those 20 standard keys in other to obtain customized masking results. It also has the ability to work with an EXTENDED character map, listed at the end of this description.

Parameters

None of the following parameters is mandatory:

- PARM1
  (Optional) Specifies the number of starting characters to be ignored.
- PARM2
  (Optional) Specifies the number of ending characters to be ignored.
- PARM3
  (Optional) (If PARM1 and PARM2 are not set) Specifies the starting position of the input string to be masked.
- PARM4
  (Optional) (If PARM1 and PARM2 are not set) Specifies the ending position of the input string to be masked.
- PARM5
  (Optional) Defines a user-defined Masterkey to be mixed with the set of the standard 20 masking group keys that are embedded in the product, generating a customized set of 20 masking group keys. If the Masterkey is not defined, the standard set of 20 masking group keys will be used. The Masterkey length can be up to 255 characters, but only the first 20 characters of the Masterkey will be used to be mixed with the product keys, following the same rule of the Windows version.
- PARM6
  (Optional) Defines the Ignored Chars, that means, which characters of the original input string will not be masked.
  Example:
  Original input: ABCDE
  Masked output: LIXUV
  Masked output with Ignored chars 'BD': L**B**X**D**V.
- (Optional) PARM7

Specifies Excluded Chars, that means, which characters will not be present in the output string. As the excluded characters will be not present in the output map of characters, the masking algorithm will produce a completely different result from those where they are not defined.
Example:
Original input: ABCDE
Masked output: LIXUV
Masked output with Excluded chars = 'IU': LJAVX

Job parameter:

To use the EXTENDED characters map, set the following parameter in the JCL:

```
FORMATENCRYPTEXTENDEDCHARS=Y
```

Applies to:

Characters and Numbers.

> **NOTE**
> The Masterkey, Ignored and Excluded chars can be defined individually for each Column or Field. This means, that you can set a different Masterkey for each column in a table, so results can vary upon need. Because of this, when the Excluded characters feature is used, the masking process uses more CPU because the output map needs to be redone for each given column referred to in the MAPCSV file.

Extended Characters Map:

À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï	ð	ñ	ò	ó	ô	õ	ö	ø	ù	ú	û	ü	ý	þ	ÿ

FORMATHASH

Hashes lowercase letters to lowercase letter, uppercase letters to uppercase letters and digits to digits. All other characters remain the same.
Applies to Field Type: Character
Required Parameters: None
Example: "`ABC/123-dur~678` " becomes "`VRT/529-cas~210` "

FORMATLUHN

The FORMATLUHN function consistently changes the current value, preserving the format (letters to letters, digits to digits). The digits are used to calculate the check digit, which then replaces the last digit in the resulting masked value. The function produces unique values as long as the original values are also unique.

Parameters:

- **Starting From Position**
  (Optional) Defines the initial character position where to start masking, counting from 1. By default it masks all characters up to the last. Can be used together with **Number of Digits to Mask**.
- **Number of Digits to Mask**
  (Optional) Defines the number of digits to mask. By default it starts at the first character. Can be used together with **Starting From Position**.
- **Number of Last Digits to Mask**
  (Optional) Defines the number of digits to mask, counting backwards from the end. The last digit will be the checksum and does not count. If **Start From Position** or **Number of Digits to Mask** are defined, **Number of Last Digits to Mask** is ignored.

Applies to: Number and Character

Example: The values in the `PART_NUMBER` column are masked by using the FORMATLUHN function; for example, `ABC/123-A1` to `VJI/802-E9` , `DEF/456-B1` to `YML/135-F4` , `GHI/123-C3` to `BPO/802-G9` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
PARTS	PART_NUMBER	FORMATLUHN			

Example: I want to format "1234567890" starting from position "3". The function returns "1297432113", where "12" remains unchanged, "3456789" is encrypted to "9743211", and the checksum is calculated as 3 and replaces the last digit. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
STAFF	ID_NUMBER	FORMATLUHN	3		

Example: I want to format "2" digits of "1234567890". The function returns "6834567894", where "3456789" remains unchanged, the first 2 digits "12" are encrypted to 68, and the checksum is calculated as 4 and replaces the last digit. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
STAFF	ID_NUMBER	FORMATLUHN		2	

Example: I want to format the last "2" digits of "1234567890". The function returns "1234567431", where "1234567" remains unchanged, "89" is encrypted to "43", and the checksum is calculated as 1 and replaces the last digit. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
STAFF	ID_NUMBER	FORMATLUHN			2

FORMATMASK

Mask a value retaining the original format. Only character a..z, A..Z and 0..9 are masked.
Applies to Field Type: Character
Parm1 (Mandatory): Masking format. E.g. "`AA99999`" denotes two alphabetic characters followed by five numeric characters

GENCARD

Generate a random credit card number.
Applies to Field Type: Character, Numeric
Required Parameters: None

HASH

Create a Hash value based on current value and Parm2 (the hash key).
Applies to Field Type: Character, Numeric
Parm1 (Mandatory): Integer number of digits required in the return value. Maximim value is 18.
Parm2 (Mandatory): Integer value representing the Hash key

HASHABN

Hash Australian Business Number with valid check digits.
Applies to Field Type: Character
Required Parameters: None

HASHACN

Hash Australian Company Number with valid check digit.
Applies to Field Type: Character
Required Parameters: None

HASHCARD

Hash a credit card number.
Applies to Field Type: Character, Numeric
Required Parameters: None

HASHCARD1

Hash a credit card number using FORMATENCRYPT1. This function masks digits only. If the data to be masked contains characters and digits, use the start and end position parameters to exclude non-numeric characters from masking. If the input value does not have a valid length for a credit card number, the number will be encrypted and its length will remain invalid. If the input value is null, 0, or 0000000000000000, the value will not be changed.
Applies to Field Type: Character, Numeric

Parm1 - Card Start Position (Optional): Position where the credit card number starts in the String. Default value is 1.
Example: ABC1234567890123456. In this example, Card Start Position is 4.
Parm2 - Card End Position (Optional): Position where the credit card number ends in the String. Default value is 16.
Example: 1234567890123456ABC. In this example, Card End Position is 17.
Parm3 - Mask Start Position (Optional): Position in the credit card number where FDM will start masking. Default value is 1.
Example: 1234567890123456. In this example, Mask Start Position is 4.
Parm4 - Mask End Position (Optional): Position in the credit card number where FDM will stop masking. Default value is 16.
Example: 1234567890123456. In this example, Mask End Position is 14.
Parm5 - Master Key (Optional): Custom user key that will be mixed with the internal key set as it is defined for FORMATENCRYPT1. For more information, see FORMATENCRYPT1.
Parm6 - Custom End Digits (Optional): A sequence of digits in the end of the credit card number that will not be masked. The check digit will not be updated.
Example: 1234567890123000. In this example, Custom End Digits is 00.

HASHDOB

Hash a date keeping the original age.
Applies to Field Type: Date
Required Parameters: None

HASHLOV

Hash current value to consistently pick a value from the seed list or table in Parm1.
Applies to Field Type: Character, Numeric
Parm1 (Mandatory): Seedlist name (from GTSRC_REFERENCE_LOV1.RL_REF_ID).
Parm2 (Optional): Integer between 1 and 30, default is 1. This identifies which column of data (RL_REF_VALUE to RL_REF_VALUE30) is returned.

Parm3 (Optional): Field name that contains the value to be hashed. Default: The field to be masked.

Example

SSN_COL in table SSN_TAB is hashed to give the row to return from GTSRC_REFERENCE_LOV1 where RL_REF_ID = "ADDRESSES".

Table: SSN_TAB
Column: ADDR_LINE1
Function: HASHLOV
Parm1: ADDRESSES
Parm2: 1
Parm3: 1 SSN_COL
Parm4: N/A

Example

There is a cross-reference table with RX_REF_ID = "PERSON_SSN" and containing PERSON_IDs as old values. The new value from the cross-reference table is hashed to give the row to return from GTSRC_REFERENCE_LOV1.
If no entry is found in the cross-reference table, then the masking program produces a warning message and the value in ADDR_LINE1 is unchanged.

Table: PERSON_TAB
Column: ADDR_LINE1
Function: HASHLOV
Parm1: ADDRESSES
Parm2: 1
Parm3: XREF, PERSON_SSN(PERSON_ID)
Parm4: N/A

Associated Options Settings for HASHLOV

Set the following parameter options in the PARMCD input DD file:

- **TRIMMEDHASHLOV=Y/N**Y — Leading and trailing blanks of the value to hash are trimmed.
  Default: N
- **CASEINSENSITIVEHASHLOV=Y/N**Y — The value to hash is converted to upper case.
  Default: N
- **HASHTYPE=JAVA/ASM**JAVA — The hashing algorithm is the same as that used by FDM. This algorithm ensures consistency between FDM and zOS masking.
  ASM — An assembler hashing algorithm is used. The ASM algorithm is quicker than the JAVA algorithm.

Important HASHLOV Requirements

To ensure that mainframe DB2 masking (HASHLOV) is consistent with FDM masking (HASHLOV), you must set the following options:

Mainframe

- – TRIMMEDHASHLOV=Y
  – HASHTYPE=JAVA
  – CASEINSENSITIVEHASHLOV=Y

FDM

- – TRIMMEDHASHLOV=Y
  – SEEDTABLEINDEXCOLUMN=rl_rn

Note: Only when using seedlist lookups from mainframe. Specify the index columns in the options file, to ensure that SDM uses the same index columns as the mainframe hashing.

To ensure that mainframe file masking (HASHLOV) matches the FDM masking, you must set the following options:

Mainframe

- – HASHTYPE=JAVA

FDM

- – SEEDTABLEINDEXCOLUMN=rl_rn
  Note: Only when using seedlist lookups from mainframe. Specify the index columns in the options file, to ensure that SDM uses the same index columns as the mainframe hashing.

HASHLOV1

Hash current value to consistently pick a value from seed list or table in Parm1 where RL_REF_VALUE equals field value from.
Applies to Field Type: Character, Numeric
Parm1: (Mandatory): Seedlist name (from GTSRC_REFERENCE_LOV1.RL_REF_ID).
Parm2: (Mandatory): Field name used to restrict the seedlist rows to those where (GTSRC_REFERENCE_LOV1.RL_REF_VALUE matches the value in the given field.
Parm3: (Optional): Integer between 1 and 30 (default 1). This identifies which column of data (RL_REF_VALUE to RL_REF_VALUE30) is to be returned.
Parm4: (Optional) Field name containing the value to be hashed or a reference to a cross reference table to give the value to hash. Default is the field to be masked.

Example

SSN_COL in table SSN_TAB is hashed to give the row to return from GTSRC_REFERENCE_LOV1 where RL_REF_ID = "ADDRESSES" and RL_REF_VALUE = the value from PCODE.

Table: SSN_TAB
Column: ADDR_LINE1
Function: HASHLOV1
Parm1: ADDRESSES
Parm2: PCODE
Parm3: 2
Parm4: SSN_COL

Example

There is a cross-reference table with RX_REF_ID = "PERSON_SSN" and containing PERSON_IDs as old values. The new value from the cross-reference table is hashed to give the row to return from GTSRC_REFERENCE_LOV1. Note – if no entry is found in the cross-reference table then the masking program produces a warning message and the value in ADDR_LINE1 is unchanged.

Table: PERSON_TAB
Column: ADDR_LINE1
Function: HASHLOV1
Parm1: ADDRESSES
Parm2: PCODE
Parm3: 2
Parm4: XREF, PERSON_SSN(PERSON_ID)

Associated Options Settings for HASHLOV1

Set the following parameters options in the PARMCD input DD file:

- **TRIMMEDHASHLOV=Y/N**Y — Trim the leading and trailing blanks of the value to hash.
  Default: N
- **CASEINSENSITIVEHASHLOV=Y/N**Y — Convert the value to hash to upper case.
  Default: N
- **HASHTYPE=JAVA/ASM**JAVA — The hashing algorithm is the same as that used by FDM. This algorithm ensures consistency between FDM and zOS masking.
  ASM — An assembler hashing algorithm is used. The ASM algorithm is quicker than the JAVA algorithm.
  Default: ASM

  WARNING

  Because of differences in architecture, there is no guarantee that results from FDM and Mainframe (DB2 or File) masking match.

HASHPHONE4

Hash the last 4 digits of a phone number.
Applies to Field Type: Character, Numeric
Required Parameters: None

HASHRUT

Takes an existing RUT number (Chilean Social Security number) and hashes the first 8 digits, then adds the appropriate check digit to the end.
Applies to Field Type: Character
Required Parameters: None

HASHTURKISHID

Masks an existing 11-digit Turkish Identification Number.
Required Parameters: None
Applies to Field Type: Numeric, Character

HASHTURKISHTAXID

The HASHTURKISHTAXID function masks an existing 10-digit Turkish Tax Identification Number.

Parameters: None

Applies to: Numeric, Character

HASHUSSSN

Takes an existing US Social Security Number (SSN), hashes the 9 digits, then returns the hashed value in either 999999999 format (if supplied in that format), or in 999-99-9999 format.
Applies to Field Type: Character, Numeric
Required Parameters: None

IGNORE

Mask using a cross reference value, if found. If not, mask using the fixed value supplied in Parm1. If no value is supplied in Parm1, the column value is unchanged. If no cross reference value is found, the cross reference table is not updated, for example, no new cross reference rows are inserted.
Applies to Field Type: Character, Numeric, Date
Parm1 (Optional): Fixed value to mask the column if no cross reference entry is found

INTRANGE

Mask the column using a random integer between Parm1 and Parm2.
Note: If the column or field accepts decimal values, you can also use the NUMERICRANGE masking functions.
Applies to Field Type: Character, Numeric, Date
Parm1 (Mandatory): Integer value representing the lower value of the potential range.
Parm2 (Mandatory): Integer value representing the upper value of the potential range.

MASTERCARD

Mask the field with a random Mastercard credit card number.
Applies to Field Type: Character, Numeric
Required Parameters: None

MOD11

Strips the first 15 non-numeric characters from the field and calculates the mod11 check digit on the numeric. Where valid (not 10), the check digit is applied to the last numeric in the field. Otherwise if the calculated check digit is invalid, then either it increments the digit position specified by Parm2 and performs the check digit calculation again, or (if Parm2 = X) it marks the check digit as 'X'.
Applies to Field Type: Character, Numeric
Parm1 (Mandatory): Integer value of the weighting to be used in the mod11 calculation. Exclude weighting on the check digit. For example, 21212121212
Parm2 (Mandatory): Contains *one* of the following:

- The digit position (from right) that is incremented if an invalid check digit was calculated.
- 'X' indicating an invalid check digit (value of 10) – It places an X in the check digit position. (Available for character fields.)

Parm3 (Optional): Integer value to add prior to check digit calculation to affect the mod11 calculation.

NOR-SSN-CHECK

Attempts to generate check digits for a Norwegian social security number. The field being masked should contain a Norwegian social security number. The masked value includes valid check digits.
Note: Because the check digit algorithm uses modulus 11, the function may change digits 7 to 9 of the input number to give a valid social security number.
Applies to Field Type: Character, Numeric
Required Parameters: None

NOR-SSN-DOB

Extracts the date of birth for a given Norwegian social security number and assigns to the columns or field.
Applies to Field Type: Character, DB2 Date
Parm1 (Mandatory): Field name containing a valid Norwegian social security number, or a reference to a cross reference table containing a valid Norwegian social security number.

Example

SSN_COL in table SSN_TAB contains a valid social security number. The date of birth for this number is used to mask DOB. If SSN_COL contains 30129900063 then the value in DOB will be 1999-30-12.

Table: SSN_TAB
Column: DOB
Function: NOR-SSN-DOB
Parm1: ADDRESSES
Parm2: SSN_COL

Parm3: N/A
Parm4: N/A

Example

There is a cross-reference table with RX_REF_ID = "PERSON_SSN" and containing PERSON_IDs as old values and SSNs as new values. PERSON_ID in table PERSON_TAB is used to lookup the SSN from the cross-reference. The date of birth for this SSN is used to mask DOB. Note – if no entry is found in the cross-reference table then the masking program produces a warning message and the value in DOB is unchanged. If PERSON_ID contains 12345 and PERSON_SSN in GTSRC_XREF contains a row with RX_OLD_VALUE = 12345 and RX_NEW_VALUE = 30129900063 then the value in DOB will be 1999-30-12.

Table: PERSON_TAB
Column: DOB
Function: NOR-SSN-DOB
Parm1: XREF, PERSON_SSN(PERSON_ID)
Parm2: N/A
Parm3: N/A
Parm4: N/A

NOR-SSN-DOB-SEX-D

Interrogates a Norwegian social security number to return a string consisting of date of birth (in format CCYY-MM-DD) concatenated with a sex indicator ("M" or "F") and a D-number indicator ("D" or " ").
Applies to Field Type: Character
Parm1 (Mandatory): Field name that contains a valid Norwegian social security number, or a reference to a cross reference table that contains a valid Norwegian social security number.

Example

SSN_COL in table SSN_TAB contains a valid social security number. The date of birth for this number is used to mask DOB. If SSN_COL contains 30129900063 then the value in DOB will be 1999-30-12F.

Table: SSN_TAB
Column: DOB
Function: NOR-SSN-DOB-SEX-D
Parm1: SSN_COL
Parm2: N/A
Parm3: N/A
Parm4: N/A

Example

There is a cross-reference table with RX_REF_ID = "PERSON_SSN" and containing PERSON_IDs as old values and SSNs as new values. PERSON_ID in table PERSON_TAB is used to lookup the SSN from the cross-reference. The date of birth for this SSN is used to mask DOB. Note – if no entry is found in the cross-reference table then the masking program produces a warning message and the value in DOB is unchanged. If PERSON_ID contains 12345 and PERSON_SSN in GTSRC_XREF contains a row with RX_OLD_VALUE = 12345 and RX_NEW_VALUE = 30129900063 then the value in DOB will be 1999-30-12.

Table: PERSON_TAB
Column: DOB
Function: NOR-SSN-DOB
Parm1: XREF PERSON_SSN(PERSON_ID)
Parm2: N/A
Parm3: N/A
Parm4: N/A

NOR-SSN-MASK

Masks a Norwegian social security number retaining the date of birth and sex. The function works by calculating all the possible valid numbers for the given date of birth and sex and the assigning one of these numbers as the masked value. The function is deterministic and will retain uniqueness.
Applies to Field Type Character: Numeric
Required Parameters: None
Parm1: Integer seed value to determine which number is selected as the masked value.

Example

Table: SSN_TAB
Column: SSN
Function: NOR-SSN-MASK
Parm1: N/A
Parm2: N/A
Parm3: N/A
Parm4: N/A

Example

Table: SSN_TAB
Column: SSN
Function: NOR-SSN-MASK
Parm1: 12345
Parm2: N/A
Parm3: N/A
Parm4: N/A

NOR-SSN-REVERSE

Reverses the date part of a Norwegian social security number so that the format of the number is YYMMDDNNNCC rather than DDMMYYNNNCC.
Applies to Field Type: Numeric
Required Parameters None

Example

If SSN_COL contains 30129900063, then it is masked with 99123000063.

Table: SSN_TAB
Column: SSN
Function: NOR-SSN-REVERSE
Parm1: N/A
Parm2: N/A
Parm3: N/A
Parm4: N/A

NUMERICRANGE

Mask the column using a random number with decimal precision given between Parm1 and Parm2. Used to mask numeric with decimal precision.
Applies to Field Type: Numeric
Parm1 (Mandatory): Decimal number representing the lower value of the potential range.
Parm2 (Mandatory): Decimal number representing the upper value of the potential range.

NUMHASH

Hashes a numeric value in a character column as digits. The key is defined as Parm1 is the key. Function is to produce the same output as the DM variant.
Applies to Field Type: Character
Parm1 (Mandatory): Integer value used as the Hash key
Parm2 (Optional): Integer value representing the maximum length to be returned
Parm3 (Optional): Integer value representing the minimum length to be returned

OR

Only applicable to flat files. For more information, see Mask Flat Files Using WHERE Clauses.
Applies to Field Type: All
Required Parameters: None

PARTMASK

Mask only alphabetic (Parm1 ='C') or numeric (Parm1 = 'N') with randomly selected characters. The case of alphabetic characters is retained.
Applies to Field Type: Character
Parm1 (Mandatory): Specify one of the following:

- 'C' — Mask only characters.
- 'N' — Mask only numeric.

PHONE_01

Replaces digits 0...9 with digits in Parm1 or a fixed replacement value. The 4th, 5th, and 6th digits in the field are each masked to value '5'. This ensures the masked value cannot be reverse engineered. If there are less than 6 digits, the field is returned unmasked.
Note: The function masks data within the first 20 characters of the field, any subsequent data is left unmasked. This deviates from function for SDM/FDM, but 20 characters should be sufficient to hold a phone number.
Applies to Field Type: Character
Parm1 (Optional): 10 digit integer to affect the masking of the integers.

POSITIONMASK

The POSITIONMASK function masks a value based on positional rules, that you define in Parm1. Separate each rule with a hyphen (no spaces).

Parameters:

- Parm1
  Specifies the rules for each position in the output value, from the following formula:
  - RD*nnn*L: Rando*nnn*m digit at position *nnn* from left.
  - RD*nnn*R: Random digit at position *nnn* from right.
  - RA*nnn*L: Random alphabetic character at position *nnn* from left.
  - RA*nnn*R: Random alphabetic character at position *nnn* from right.
  - RC*nnn*L: Random alphanumeric character at position *nnn* from left.
  - RC*nnn*R: Random alphanumeric character at position *nnn* from right.
  - F#*nnn*L: Fixed digit (#) at position *nnn* from left.
  - F#*nnn*R: Fixed digit (#) at position *nnn* from right.
  - Fa*nnn*L: Fixed alphabetic character (*a*) at position *nnn* from left.
  - F*a nnn*R: Fixed alphabetic character (*a*) at position *nnn* from right.

Notes

- For any position for which you do not provide a rule, the original value remains as the output value.
- If the old value is null or all blanks, the function skips the row.

Applies to: Character

Example: The function masks the first three characters of each value of the `PHONE_NUMBER` column in the table `PEOPLE` , with the fixed value `9`. The rest of the digits remain as their original values. The following table shows the usage:

Table	Column	Function	Parm1
PEOPLE	PHONE_NUMBER	POSITIONMARK	F9001L-F9002L-F9003L

Therefore, the resultant masked value is `999XXXXXX` , where `X` is the existing value.

For example, `1235553283` becomes `9995553283` , and `9238974398` becomes `9998974398` .

RANDEIN

Generate a random EIN (US Employer Identification Number).
Applies to Field Type: Character, Numeric
Required Parameters: None

RANDHIC

Generate a random HIC (US Health Insurance Claim) number.
Applies to Field Type: Character
Required Parameters: None

RANDLOV

Mask the field value with the randomly selected values from the seed table.
Applies to Field Type: Character, Numeric
Parm1 (Mandatory): Seedlist name (matching GTSRC_REFERENCE_LOV1.RL_REF_ID).
Parm2 (Optional): Integer between 1 and 30. Default is 1. This identifies which column of data (RL_REF_VALUE to RL_REF_VALUE30) is returned.

Example

Table: SSN_TAB
Column: STATE
Function: RANDLOV
Parm1: US STATE, ZIP CITY, COUNTY
Parm2: N/A
Parm3: N/A
Parm4: N/A

Example

Table: SSN_TAB
Column: ZIP
Function: RANDLOV
Parm1: US STATE, ZIP CITY, COUNTY
Parm2: 2
Parm3: N/A
Parm4: N/A

Example

The value in SSN_TAB.STATE is hashed to provide the lookup value against the seedlist "US STATE ZIP CITY COUNTY" and the 1st reference value column is returned (RL_REF_VALUE). The same hashed value is used for SSN_TAB.ZIP and the 2nd reference value column is returned (RL_REF_VALUE2). As SSN_TAB2 is a different table / record, the value in SSN_TAB2.STATE is hashed to provide the lookup value against the seedlist "US STATE ZIP CITY COUNTY" and the 1st reference value column is returned (RL_REF_VALUE).

Table: SSN_TAB
Column: ZIP
Function: RANDLOV
Parm1: US STATE, ZIP CITY, COUNTY
Parm2: 2
Parm3: N/A
Parm4: N/A

RANDLOV1

Mask data by randomly picking a value from the seed list or table in Parm1 where RL_REF_VALUE equals field value from Parm2.
Applies to Field Type: Character, Numeric
Parm1 (Mandatory): Seedlist name from GTSRC_REFERENCE_LOV1.RL_REF_ID.
Parm2 (Mandatory): Field name used to restrict the seedlist rows to those where the GTSRC_REFERENCE_LOV1.RL_REF_VALUE matches the value in the given field.
Parm3 (Optional): Integer between 1 and 30. Default is 1. This identifies which column of data (RL_REF_VALUE to RL_REF_VALUE30) is returned.
Note: RANDLOV1 returns data from the same seedlist row on subsequent calls to RANDLOV1 within the same record. This permits the usage of RANDLOV1 to return consistent data across multiple columns for a single record.

RANDLUHN

Generate a random number with a valid Luhn check digit.
Applies to Field Type: Character, Numeric
Parm1 (Mandatory): Integer specifying the number of digits (max 18) including the check digit.

RANDNINO

Generate a random UK National Insurance (NI) number.
Applies to Field Type: Character
Parm1 (Optional): Separator character.

> **NOTE**
>
> RANDNINO adds the character you choose as a separator to the output value, in the pattern '`AB_##_##_##_A`' (where '_' is the separator).

RANDTIN

Generate a random TIN (US Tax Identification Number).
Applies to Field Type: Character, Numeric
Required Parameters: None

RANDOMDATE

Generate a random date between Parm1 and Parm2.
Applies to Field Type: DB2 Date, Field with DATEFORMAT specified in the AFL

Parm1 (Mandatory) Date in format CCYYMMDD specifying the lower of the date range
Parm2 (Mandatory) Date in format CCYYMMDD specifying the upper of the date range

RANDOMDAY

Randomly change the day between Parm1 and Parm2 leaving the year and month unchanged. The returned value will be a valid date.
Applies to Field Type: DB2 Date, Field with DATEFORMAT specified in the AFL
Parm1 (Optional): Integer specifying the lower range of the day value (default 1)
Parm2 (Optional): Integer specifying the upper range of the day value (default 31)

RANDOMTXT

Generate random text with a length between Parm1 and Parm2.
Applies to Field Type: Character
Parm1 (Mandatory): Integer specifying the minimum length of the returned text
Parm2 (Mandatory): Integer specifying the maximum length of the returned text

RANDRUT or RUT

Generate a random Chilean Social Security number.
Applies to Field Type: Character
Required Parameters: None

RANDSIN

Generate a random SIN (Canadian Social Insurance) number.
Applies to Field Type: Character, Numeric
Parm1 (Optional): Character separator used to break the 9 digit number into 3 sets.
Example: If you specify "-", the generated number will be in the following format: 999-999-999

RANDSSN

Generate a random SSN (US Social Security Number).
Applies to Field Type: Character, Numeric
Parm1 (Optional): Character separator used as a separator in the generated SSN.
Example: If you specify "-", the generated number will be in the following format: 999-99-999

RANDHIC

Description Generate a random TIN (US Tax Identification Number).
Applies to Field Type: Character
Required Parameters: None

REPLACE

Searches the field values for the character pattern mentioned in Parm1 and replaces it with the character pattern mentioned in Parm2. The replace operation is case sensitive.
Applies to Field Type: Character
Parm1 (Mandatory): Character pattern to be searched for in the field
Parm2 (Mandatory): Character pattern to replace with in the field

Example

The pattern 'Ab', when found in the column ADDRESS_1, is replaced by '23', while 'a', if found in the column ADDRESS_2, is also replaced by '23'.

Table	Column	Function	Parm1	Parm2
SSN_TAB	ADDRESS_1	REPLACE	Ab	23
SSN_TAB	ADDRESS_2	REPLACE	a	23

RJUST

Strip blanks from the right of the string and right justify the data in the field, padding to the left with blanks. You use this function most likely in conjunction with substr functionality.
Applies to Field Type: Character
Required Parameters: None

SEQLOV

Mask the field value with the sequentially selected values from the seed table.
SEQLOV returns data from the same seedlist row on subsequent calls to SEQLOV within the same record. This permits the usage of SEQLOV to return consistent data across multiple columns for a single record.
Applies to Field Type: Character, Numeric
Parm1 (Mandatory): Seedlist name (matching GTSRC_REFERENCE_LOV1.RL_REF_ID).
Parm2 (Optional): Integer between 1 and 30 (default 1). This identifies which column of data (RL_REF_VALUE to RL_REF_VALUE30) is to be returned.

SEQLOV1

Mask data by sequentially picking a value from seed list or table in Parm1 where RL_REF_VALUE equals field value from Parm2.
SEQLOV1 returns data from the same seedlist row on subsequent calls to SEQLOV1 within the same record. This permits the usage of SEQLOV1 to return consistent data across multiple columns for a single record.
Applies to Field Type: Character, Numeric
Parm1 (Mandatory): Seedlist name (from GTSRC_REFERENCE_LOV1.RL_REF_ID).
Parm2 (Mandatory): Field name used to restrict the seedlist rows to those where (GTSRC_REFERENCE_LOV1.RL_REF_VALUE matches the value in the given field.
Parm3 (Optional): Integer between 1 and 30 (default 1). This identifies which column of data (RL_REF_VALUE to RL_REF_VALUE30) is to be returned.

SEQNUMBER

Generate a sequential number starting at Parm1.
Applies to Field Type: Numeric
Parm1 (Optional): Integer specifying the start number of the sequence.
Parm2 (Optional): Character string identifying the sequence, allowing for multiple sequences within 1 masking run.

SHUFFLE

Shuffle the values in the specified column for the entire table / file. The shuffled values are used to populate the specified column. Shuffle works in three phases.

1. Any previous data for that shuffle name (Parm1) is deleted from the seedlist lookup table (GTSRC_REFERENCE_LOV1).
2. The values in the column are used to populate the seedlist lookup table using the value in Parm1 as the seedlist name (RL_REF_ID). Rows are written to this table in random order.

3. The SHUFFLE is treated as a SEQLOV function to read values from the created table.

Applies to Field Type: Character, Numeric, DB2 Date
Parm1 (Mandatory): Name of the seedlist to be stored in GTSRC_REFERENCE_LOV1
Parm2 (Optional): Integer specifying the column in GTSRC_REFERENCE_LOV1 to store the value. You can create a multi-value SHUFFLE seedlist by supplying column numbers (between 1 and 30) in Parm2.
Example

Parm2=1 stores the data in column RL_REF_VALUE, which in this example is ADDRESS_1.

Table: SSN_TAB
Column: ADDRESS_1
Function: SHUFFLE
Parm1: ADDRESS
Parm2: 1
Parm3: N/A
Parm4: N/A

Example

Table: SSN_TAB
Column: CITY
Function: SHUFFLE
Parm1: ADDRESS
Parm2: 2
Parm3: N/A
Parm4: N/A

Example

The following shuffles the content of the 3 address fields, keeping the data in the 3 fields together as per the original record.

Table: SSN_TAB
Column: POSTCODE
Function: SHUFFLE
Parm1: ADDRESS
Parm2: 3
Parm3: N/A
Parm4: N/A

SQLFUNCTION

Mask the data with the output from the SQL function (only applicable when masking DB2).
Applies to Field Type: Character, Numeric, DB2 Date
Parm1 (Mandatory): Valid SQL function which when executed in a SELECT statement will generate the masked value to be used.

STRING-FUNC

Applies the function specified in Parm1 to a string. Available functions are "REVERSE", "UPPER", or "LOWER".
Applies to Field Type: Character
Parm1 (Mandatory): Specify one of the following:

- "REVERSE" — Reverse the characters in the string.
- "UPPER" — Convert to upper case.
- "LOWER" — Convert to lower case.

Parm2 (Optional): Integer specifying the start position of the function in the string.
Parm3 (Optional): Integer specifying the length from Parm2 to perform the function upon.

Example

Table: TAB
Column: COL
Function: STRING-FUNC
Parm1: UPPER
Parm2: N/A
Parm3: N/A
Parm4: N/A

Example

Table: TAB
Column: COL
Function: STRING-FUNC
Parm1: LOWER
Parm2: N/A
Parm3: N/A
Parm4: N/A

Example

Table: TAB
Column: COL
Function: STRING-FUNC
Parm1: REVERSE
Parm2: N/A
Parm3: N/A
Parm4: N/A

Example

Table: TAB
Column: COL
Function: STRING-FUNC
Parm1: UPPER
Parm2: 3
Parm3: 2
Parm4: N/A

Example

Table: TAB
Column: COL
Function: STRING-FUNC
Parm1: LOWER
Parm2: 3
Parm3: 2
Parm4: N/A

Example

COL contains "abcDEF" and is masked with "abDcEF".

Table: TAB
Column: COL

Function: STRING-FUNC
Parm1: REVERSE
Parm2: 3
Parm3: 2
Parm4: N/A

SUBSETLIST

Only applicable when masking files. This function is applied to the *whole* file regardless of WHERE clauses and before applying any masking functions.
The value from the field is stored in the GTSRC_SUBSET DB2 table.
Applies to Field Type Character, Numeric
Parm1 (Mandatory): Name of the subset list, that is, the value for GTSRC_SUBSET.BUNDLE_ID.
Parm2 (Optional): "Y" – Deletes the entries for the subset list in Parm1 from table GTSRC_SUBSET before inserting any values.

SUBSETLISTSQL

Only applicable when masking files. This function is applied to the *whole* file regardless of WHERE clauses and before applying any masking functions.
The value(s) returned by the SQL supplied in Parm3 and Parm4 are stored in the GTSRC_SUBSET DB2 table.
Applies to Field Type: Character, Numeric
Parm1 (Mandatory): Name of the subset list (i.e. the value for GTSRC_SUBSET.BUNDLE_ID)
Parm2 (Optional): "Y" – Deletes the entries for the subset list in Parm1 from table GTSRC_SUBSET before inserting any values.
Parm3 (Mandatory): SQL to execute (first 256 characters)
Parm4 (Mandatory): SQL to execute (continuation of Parm3 for another 256 characters). Use Parm3 and Parm4 to specify SQL statements with a length of up to 512 characters.

SWEDISHID

Generates Swedish Identity number (not necessarily unique).
If Parm1 contains a value, it is used as the source date value. Any nulls are kept and reported. If Parm1 is empty, the function uses a date range between Parm3 and Parm4.

Applies to Field Type: Character

Parm1 (Optional): Data field used to generate the ID (format CCYY-MM-DD). Either a character field with a date format of CCYY-MM-DD, or DB2 Date field. In Addition:

- MM can be in the range:
  - 1 to 12 (personnummer) or
  - > 20 (organisationsnummer)
- DD can be in the range:
  - 01 – 31 (personnummer) or
  - 61 to 91 (samordningsnummer)

Parm1 lets you account for formatting differences in Swedish IDs and for cases where you have to mask `organisationsnummer` or `samordningsnummer` values. For either of these values, define an internal variable for Parm1 and assign the value to that variable to avoid the value appearing as an invalid date.
Parm2 (Optional): Format of the ID number to return

- YYYYMMDD-XXXX (default)
- YYYYMMDDXXXX
- YYMMDD-XXXX
- YYMMDDXXXX

Parm3: (Mandatory): Date Value (CCYY-MM-DD) specifying the low value of random date range to use (if no Parm2).
Parm4: (Mandatory): Date Value (CCYY-MM-DD) specifying the upper value of random date range to use (if no Parm2).

TRANSLATE

Translate individual characters from the value given in Parm1 to the value given in Parm2.
Applies to Field Type: Character, Numeric
Parm1: (Mandatory) Character(s) to be replace from.
Parm2: (Mandatory) Character(s) to be replace to. Must contain same number of characters as Parm1.

TRANSLATERAND

Randomly translate individual characters from the values given in Parm1 through Parm4. For each parameter supplied if a character in the string to be masked is found in the parameter then it is replaced by another character from the parameter.
Applies to Field Type: Character
Parm1: (Mandatory) Character(s) to replace, at least two characters must be supplied in Parm1.
Parm2: (Optional) Characters to replace.
Parm3: (Optional) Characters to replace.
Parm4: (Optional) Characters to replace.
ExampleTRANSLATERAND,ABCDEFGHIJKLMNOPQRSTUVWXYZ,01234567890 results in all upper-case letters being replaced by another upper-case letter, and all digits being replaced by another digit.

TRANSPOSE

Convert characters consistently to other character, a to c, b to d etc.
Applies to Field Type: Character
Parm1: (Mandatory) Integer used as the transposition key
Parm2: (Optional) "Y" specifies that only alphabetic characters (a-z, A-Z) are transposed. Default is to transpose alphanumeric characters.

TURKISHID

Calculate and apply the check digit to a Turkish national identity number.
Applies to Field Type: Character
Required Parameters: None

UNIQUETURKISHID

Generates a unique 11-digit Turkish Identification Number.
Applies to Field Type: Numeric, Character
Parm1: Specifies whether you want to generate a sequence. Values: Y or N. Default: N.
Parm2: Defines the start value of the sequence if Parm1 is Y. Default: 100000000.
Parm3: Defines the end value of the sequence if Parm1 is Y. Default: 999999999.

UNIQUETURKISHTAXID

The UNIQUETURKISHTAXID function generates a unique 10-digit Turkish Tax Identification Number.

Applies to: Numeric, Character

Parameters:

- **Parm1**
  Specifies whether you want to generate a sequence. Values: Y or N. Default: N.
- **Parm2**
  Defines the start value of the sequence if Parm1 is Y. Default: 1111111111.
- **Parm3**
  Defines the end value of the sequence if Parm1 is Y. Default: 9999999999.

USPHONE

Masks the column with an auto-generated 7digit US Phone number of the format xxxxxxx.
Applies to Field Type: Character, Numeric
Required Parameters: None

USPHONE10 or USPHONE(10)

Masks the column with an auto-generated 10digit US Phone number of the format xxxxxxxxxx.
Applies to Field Type: Character, Numeric
Required Parameters: None

USZIP

Masks the columns with an auto-generated 5-digit US Zip code
Applies to Field Type: Character, Numeric
Required Parameters: None

USZIP4 or USZIP+4

Masks the columns with an auto-generated 9-digit US Zip code (format: xxxxxxxxx).
Applies to Field Type: Character, Numeric
Required Parameters: None

VALIDHIC

Tests for a valid US Health Insurance Claim (HIC) number. If the data is a valid HIC, then it will be replaced with a new value, otherwise it will leave the bad number as it is.
Applies to Field Type: Character
Required Parameters: None

VALIDNINO

Tests for a valid UK National Insurance Number (NINO). If the data is a valid NINO, then it will be replaced with a new value, otherwise it will leave the bad number as it is.
Applies to Field Type: Character
Parm1 (Optional): Separator character.

> **NOTE**
>
> VALIDNINO expects the character you choose as a separator, to appear in the input string in the pattern `'AB_##_##_##_A'` (where '_' is the separator). The separator applies to all values in the column - all separators in a column must be the same for the check to be valid.

VALIDRUT

Tests for a valid Chilean Social Security (RUT) number. If the data is a valid RUT, then it will be replaced with a new value, otherwise it will leave the bad number as it is.
Applies to Field Type: Character
Required Parameters: None

VALIDSIN

Tests for a valid Canadian Social Insurance Number (SIN). If the data is a valid SIN, then it will be replaced with a new value, otherwise it will leave the bad number as it is.
Applies to Field Type: Character, Numeric
Parm1 (Optional): Character separator used to break the 9 digit number into 3 sets. For example, "-", the generated number will be in the following format: 999-999-999

VALIDSSN

Identifies whether a column contains a valid SSN (United States Social Security Number), and if yes, masks with a generated SSN, keeping the original layout.
Applies to Field Type: Character, Numeric
Required Parameters: None

VALIDTIN

Tests for a valid US Tax Identification Number (TIN). If the data is a valid TIN, then it will be replaced with a new value, otherwise it will leave the bad number as it is.
Applies to Field Type: Character, Numeric
Required Parameters: None

VARIENCE or VARIANCE

Variant values are generated based on Parm1 (% value) and then added or subtracted to the field values.

Applies to Field Type: Numeric
Parm1 (Mandatory): Integer between 1 and 99 (percentage variance)
Parm2 (Optional): Minimum permitted value
Parm3 (Optional): Maximum permitted value
Example: If the column value is 100, then Parm1 applies 60% varience; a random number is generated between 40 and 160. However Parm2 (minimum permitted value) of 50 and Parm3 (maximum permitted value) of 150 would ensure that the generated random value lies in the range 50 - 150 instead of 40 – 160

VISACARD

Generates a random VISA card number.
Applies to Field Type: Character, Numeric
Required Parameters: None

WHERE

The WHERE function allows you to restrict your obfuscation to only certain rows in the table. This will allow you to mask, for example, Male names and Female names differently based on the GENDER column. The WHERE function does not require you to enter a column. Parm1 contains the SQL WHERE clause used to sub-select the table data.
Applies to Field Type: N/A
Parm1: (Mandatory) Valid SQL selection clause (DB2 only)

WARNING

Because of differences in architecture, there is no guarantee that results from FDM and Mainframe (DB2 or File) masking will match.

Internal Numeric variables

Numeric internal variables are likely to be most useful for computing totals and counts when processing files.

There are nine signed numeric variables available which can be used in masking. These variables are called `GT__CTR_1 ... GT__CTR_9`. Note the double underscore between GT and CTR.

All internal variables have a precision of 18 and are initialized by the masking program to zero.

Note: If you set an internal variable to a field in an array, specify the subscript of the field explicitly.

The three functions used specifically with internal numeric variables are described in this article.

SETCTR

SETCTR sets an internal numeric variable.

- The "table" column of the mapping csv defines a target table or record for masking.
- The "column" column of the mapping csv defines an internal numeric variable.
- Set either parameter 1 or parameter 2. If both parameters 1 and 2 are populated, then the value from parameter 1 is used to set the variable.
  - Parameter 1 defines a numeric column or an internal numeric variable to which you want to set the variable.
  - Parameter 2 defines an integer to which you want to set the variable. Parameter 2 can be positive or negative.
    Note: If parameter 2 includes a decimal point, only the integer portion of the number is stored.

SETCTR Examples

Example 1:

FIELD_B is a packed decimal number with 2 decimal places containing 2.34.

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
RECORD_A	GT__CTR_1	SETCTR	FIELD_B			

After you execute the function, GT__CTR_1 contains 234. Note that the scale of FIELD_B is ignored: All the digits it contains are stored in the internal variable.

Example 2:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
RECORD_A	GT__CTR_1	SETCTR		123		

After you execute the function, GT__CTR_1 contains 123.

Example 3:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
RECORD_A	GT__CTR_1	SETCTR		1.23-		

After you execute the function, GT__CTR_1 contains -1.

Example 4:

FIELD_B is a packed decimal number with 2 decimal places containing 2.34.

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
RECORD_A	GT__CTR_1	SETCTR	FIELD_B			
RECORD_A	GT__CTR_2	SETCTR	GT__CTR_1			

After you execute the functions, GT__CTR_1 and GT__CTR_2 both contain 234.

ADDTOCTR

ADDTOCTR adds to an internal numeric variable.

- The "table" column of the mapping csv defines a target table or record for masking.
- The "column" column of the mapping csv defines an internal numeric variable.
- Set either or both parameters 1 and 2. If both parameters 1 and 2 are populated, then both values are added to variable.
- Parameter 1 defines a numeric column or an internal numeric variable to which you want to set the variable.
- Parameter 2 defines an integer to which you want to set the variable. Parameter 2 can be positive or negative.
  Note: If parameter 2 includes a decimal point, only the integer portion of the number will be used.
- Parameter 3 (Optional) specifies whether you want to add or subtract. Specify "-" to subtract the values from the internal variable. The default is to add the values to the internal variable.

ADDTOCTR examples

Example 1:

FIELD_B is a packed decimal number with 2 decimal places containing 2.34

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
RECORD_A	GT__CTR_1	SETCTR		0		
RECORD_A	GT__CTR_1	ADDTOCTR	FIELD_B			

After you execute the functions, GT__CTR_1 contains 234. Note that the scale of FIELD_B is ignored: All the digits it contains are stored in the internal variable.

Example 2:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
RECORD_A	GT__CTR_1	SETCTR		0		
RECORD_A	GT__CTR_1	ADDTOCTR	FIELD_B	123		

After you execute the functions, GT__CTR_1 contains 357 (that is, 234 plus 123).

Example 3:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
RECORD_A	GT__CTR_1	SETCTR		0		
RECORD_A	GT__CTR_2	SETCTR		5		
RECORD_A	GT__CTR_1	ADDTOCTR	GT__CTR_2	3	-	

After you execute the functions, GT__CTR_1 contains -8 (that is, -5 minus 3).

ASSIGNCTR

ASSIGNCTR masks a column with the value held by an internal numeric variable.

- Parameter 1 defines the name of an internal numeric variable.
- Parameter 2 (optional) defines an integer in the range -17 to 17. This parameter is only required if the scale of the target differs from the scale of any values added to the variable.

ASSIGNCTR Examples

Example 1:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
RECORD_A	NUM_FIELD	ASSIGNCTR	GT__CTR_1			

If GT__CTR_1 contains 12345, and NUM_FIELD is defined as packed decimal with a precision of 7 and scale of 0, then NUM_FIELD is set to 12345. If NUM_FIELD is defined as packed decimal with a precision of 7 and scale of 2, then NUM_FIELD is set to 123.45.

Example 2:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
RECORD_A	NUM_FIELD	ASSIGNCTR	GT__CTR_1	2		

If GT__CTR_1 contains 12345, and NUM_FIELD is defined as packed decimal with a precision of 7 and scale of 0, then NUM_FIELD is set to 1234500. If NUM_FIELD is defined as packed decimal with a precision of 7 and scale of 2, then NUM_FIELD is set to 12345.00.

Example Use Case

You are masking a file which contains detail records with an amount field, and a trailer record with a total field. You want to mask the amount with random data, and update the total to reflect the changed amount fields. The detail record is called DETAIL, with a field called AMOUNT defined as packed decimal (7,2). The trailer record is called TRAILER with a field called TOTAL defined as packed decimal (15,2).

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
DETAIL	AMOUNT	NUMERICRANGE	1.00	99999.99		
DETAIL	GT__CTR_1	ADDTOCTR	AMOUNT			
TRAILER	TOTAL	ASSIGNCTR	GT__CTR_1			

Before the masking program reads any records, GT__CTR_1 is initialized to zero.

For each DETAIL record read by the program, AMOUNT is set to a random number between 1.00 and 99999.99. The random number used to mask AMOUNT is added to GT__CTR_1. Note that the order in which the functions appear in the mapping csv is important: To increment GT__CTR_1 with the masked value, the ADDTOCTR function must come after the NUMERICRANGE function. When the program reads a trailer record, the value in GT__CTR_1 is used to mask TOTAL. Because AMOUNT and TOTAL have the same scale, there is no need to set parameter 2 for the ASSIGNCTR function.

Internal String Variables

There are nine varying character strings that can be used in masking. These variables are called `GT__STR_1 ... GT__STR_9` . Note the double underscore between GT and STR.

All internal variables can contain 254 characters. They are initialized by the masking program to have a zero length.

Note: If you set an internal variable to a field in an array, specify the subscript of the field explicitly.

The two functions used specifically with internal string variables are described in this article. In addition, internal string variables can be used in conjunction with normal masking functions (see the Example Use Case below).

SETSTR

SETSTR sets an internal string variable.

- The "table" column of the mapping csv names a target table or record for masking.
- The "column" column of the mapping csv names an internal string variable.
- Set either parameter 1 or parameter 2. If both parameters 1 and 2 are populated, then the value from parameter 1 is used to set the variable.
  - Parameter 1 names a character column or an internal string variable to which you want to set the variable.
  - Parameter 2 is a literal to which you want to set the variable.
  - Parameter 3 (Optional) specifies how to trim the source string before being assigned to the target internal string variable. Choose one of the following:
    - **B** — specifies to trim blanks from both the left and right of the string.
    - **R** — specifies to trim blanks from the right of the string.
    - **L** — specifies to trim blanks from the left of the string.

SETSTR Examples

Example 1:

COL is a 10 character string containing `"  abc       "` .

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
TAB	GT__STR_1	SETSTR	COL		B	

After you execute the function, GT__STR_1 will contain "`abc` ".

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
TAB	GT__STR_1	SETSTR		"def"		

After you execute the function, GT__STR_1 will contain "def".

Example 2:

COL is a 10 character string containing `"  abc       "` .

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
TAB	GT__STR_1	SETSTR	COL			
TAB	GT__STR_2	SETSTR	GT_STR_1		B	

After you execute the function, GT__STR_1 will contain `"  abc       "` , GT__STR_2 will contain `"abc"` .

Example 3:

WARNING

From v5.0.8 (of the masking program), SUBSTR for the function SETSTR behaves differently to all other functions. It now assigns the substring of the Parm1 value into the internal variable.

COL is a 10 character string containing "1234567890".

Table	Column	Function	Parm1	Substr Start	Substr length
TAB	GT__STR_1	SETSTR	COL	2	7

After you execute the function, GT__STR_1 will contain "2345678".

ASSIGNSTR

ASSIGNSTR masks a column with the value held by an internal string variable.

- Parameter 1 defines the name of an internal string variable.
- No other parameters are required.

ASSIGNSTR Examples

Example 1:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
TAB	COL	ASSIGNSTR	GT__STR_1			

After you execute the function, COL contain the string currently held by GT__STR_1.

Example Use Case

The aim of this usecase is to ensure that if "John Doe" in table PERSON is masked with "Joe Blogg", then in table EMPLOYEES "John" is masked with "Joe", and "Doe" is masked with "Bloggs".

- The PERSON table has column FULL_NAME, which contains a first name and a surname, separated by a space. For example "John Doe".
- The EMPLOYEES table has columns FIRST_NAME and SURNAME. For example "John" in FIRST_NAME and "Doe" in SURNAME.

In the seed table, there is a seedlist called "FULL NAME", which holds the first name concatenated with the surname in value column 1, first name in value column 3, and surname in value column 4. A row in the seed table might appear as follows:

RL_REF_ID	RL_REF_VALUE	RL_REF_VALUE3	RL_REF_VALUE4
FULL NAME	Joe Bloggs	Joe	Bloggs

The following mappings will achieve the desired result:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PERSON	FULL_NAME	HASHLOV	FULL NAME	1		
EMPLOYEES	GT__STR_1	SETSTR	FIRST_NAME		B	
EMPLOYEES	GT__STR_2	SETSTR	SURNAME		B	
EMPLOYEES	GT__STR_1	CONCAT		GT__STR_1	" "	GT_STR_2

EMPLOYEES	FIRST_NAME	HASHLOV	FULL NAME	3	GT__STR_1	
EMPLOYEES	SURNAME	HASHLOV	FULL NAME	4	GT__STR_1	

Mask Flat Files Using WHERE Clauses

When processing flat files, you can use "WHERE" clauses in two ways:

- By default, the WHERE clauses restrict the records to which masking will be applied.
- The WHERE clauses can alternatively indicate which records are included in the output file. Supply the following option in the PARMCD dataset to the masking set:
  `WHEREASSUBSET=Y`

On the **Transformation Maps** screen, enter a clause in the following format in the **WHERE clause** column:

```
WHERE-CLAUSE :: COMPARISON (BOOLEANOP COMPARISON)*
COMPARISON   :: field-name COMPARISONOP value
COMPARISONOP :: {=|<>|<=|<|>=|>|IN|NOT IN}
BOOLEANOP    :: {AND|OR}
```

The clause has the following parameters:

- ***field-name*** — Defines the field name for the comparison. Choose one of the following:
  - ***name*** — Defines the name of a field defined for the record being masked.
  - **GT-SUBSET-SAMPLE=*n*** — Defines the subset consisting of each *n*th record from the input file.
  - **GT-SUBSET-ROW= *n*** — Defines the subset consisting of the first *n* rows from the input file.
    Example: `GT-SUBSET-SAMPLE=1000` results in a subset consisting of each 1000th record from the input file.
    Example: `GT-SUBSET-ROW=1000` results in a subset consisting of the first 1000 rows from the input file.
  - ***array*[*n*:*m*]** — (Arrayed elements only) Use subscripts to refer to arrayed elements. Enclose the subscript in square brackets. Separate index values for different dimensions by a colon.
    Example: `names[1:2]`, `streets[3]`
  - ***array*[ANY]** — (Arrayed elements only) The condition evaluates to true if any element in the array evaluates to true.
  - ***array*[ALL]** — (Arrayed elements only) The condition evaluates to true if all elements in the array evaluate to true.
- ***value*** — Defines the comparison value. The data type can be string, quoted string, or numeric.

The WHERE clause is evaluated from left to right.

Operator Usage:

Boolean AND operators have higher precedence than OR operators.

Each 'WHERE' clause can have only one 'AND' and one 'OR' clause.

You can apply the IN or NOT IN to a 'WHERE', 'AND' or 'OR' clause.

The IN comparison operator lets you test against a list of values held in the GTSRC_SUBSET table. The value following IN must correspond to a value in GTSRC_SUBSET.BUNDLE_ID. You can populate GTSRC_SUBSET independently of the masking program, or via the SUBSETLIST or SUBSETLISTSQL functions.

User Functions - Specification and Calling

How to Call User Functions

You can call user-written functions (either COBOL or PL/1) by specifying the EXIT masking function and the program name in parm1.

The user program must have the following linkage and entry sections specified:

COBOL Linkage

If the called program is written in COBOL, define the linkage section as follows:

```
**********************************************************
LINKAGE SECTION.
01 RETURN-CD.
05 STATUS-CD PIC 99.
05 GT--MESSAGE.
49 GT--MESSAGE-LEN PIC S9(04) COMP.
49 GT--MESSAGE-STR PIC X(254).
01 EXIT-PARMS.
05 VAL-NUMERIC PIC S9(18) COMP-3.
05 VAL-STRING.
10 VAL-STRING-LEN PIC S9(4) COMP.
10 VAL-STRING-STR PIC X(4000).
10 PARM-ARRAY OCCURS 4 TIMES.
15 PARM-LEN PIC S9(4) COMP.
15 PARM-STR PIC X(256).
15 PARM-NUMERIC PIC S9(18) COMP-3.
15 PARM-NUMERIC-SCALE PIC 99.
**********************************************************
PROCEDURE DIVISION USING RETURN-CD, EXIT-PARMS.
```

PL/1 Entry

If the called program is written in PL1, define the linkage section as follows:

```
EXIT: PROC ( RETURN_CD, EXIT_PARMS ) OPTIONS ( MAIN, COBOL );
DCL 1 RETURN_CD,
05 STATUS_CD PIC '99',
05 GT_MESSAGE_STR CHAR(254) VARYING,
DCL 1 EXIT_PARMS,
05 FILLER_1 CHAR(2),
05 VAL_NUMERIC FIXED DEC(15),
05 VAL_STRING CHAR(4000) VARYING,
10 PARM_ARRAY(4),
15 PARM_STR CHAR(256) VARYING,
15 FILLER_2 CHAR(2),
15 PARM_NUMERIC FIXED DEC(15),
15 PARM_NUMERIC_SCALE PIC '99';
```

Parameters:

- **VAL_NUMERIC** (for numeric data types) — Contains the numeric value of the column being masked. If the column is a non-integer, the decimal point is implied. The masking program updates this parameter with the required masked value.
- **VAL_STRING** — Contains the string value of the column being masked. The masking program updates this parameter with the required masked value.
- **PARM_STR** — Contains the string value of the parameter entered in the mapping CSV. If no value was entered, PARM_STR has a zero length.
- **PARM_NUMERIC** — Contains the numeric value of the parameter entered in the mapping CSV. Note that for non-integer values, the decimal point is implied (and is given by PARM_NUMERIC_SCALE). In addition, the precision and scale of PARM_NUMERIC is adjusted to match that of the column being masked.
  For example, if the column being masked is an integer, and PARM_STR contains "123.4", then PARM_NUMERIC contains 123, and PARM_NUMERIC_SCALE contains 0.

Reporting errors or warnings

Before the exit program is invoked:

- RETURN_CD is set to zero, and
- GT_MESSAGE_STR has a length of zero.

Before the program returns:

- It sets RETURN_CD to 4 to indicate that a warning was issued, or
- It sets RETURN_CD to 8 or higher to indicate that an error occurred.

GT_MESSAGE_STR is populated with an appropriate warning or error message. The message is printed in the masking report.

Utility Programs

These are the programs that TDM Mainframe offers for performing additional tasks.

Copybook pre-processor (GTXCPY)

If the format of your copybooks does not conform to the standards that TDM requires, you can use the Copybook Pre-processor to replace incompatible definitions, with compatible ones.

Parameters:

- **LOADLIB**
  The load library that contains the program GTXCPY. Default: GRIDTOOL.LIB.LOADLIB
- **CNFDS**
  Library that lists definitions to replace, and the new definitions with which to replace them. Default: GRIDTOOL.LIB.CONFIGThis library must contain 2 columns, with the definitions to replace in the first column, and the new definitions in the second column.
  Sample GRIDTOOL.LIB.CONFIG library:

```
ZZZZZ999.99 COMP-3. X(06).

-------99.          A(09).

+++.                999.

$(10).99.           X(12).
```

```
$$$,$$$,$$9.00.      X(10)9.99.

---------9.          X(05).

'MALAYCREDIT'.       'XX-------XX'.

'SC_CCMSHK_CR'.      'AA-BBBBBB-CC'.

'+08.00'.            '-08.00'.

9(06).999999.        9(06).9(06).
```

- **SRCDS**
  Copybook library that contains the definitions you want to replace. Default: GRIDTOOL.LIB.SRCCOPY
- **REPDS**
  Copybook library to which to write the contents of the dataset defined by **SRCDS**, with new definitions as per the contents of the dataset defined by **CNFDS**. Default: GRIDTOOL.LIB.REPCOPY
- **CNFIG**
  Config rules member.
- **SRCCB**
  Source copybook member
- **REPCB**
  Replace copybook member.

Dump Data From DB2 Tables (DB2)

To dump data from DB2 to flat files, JCL is supplied in the installation package as GTXDMP. This job uses procedure GTDMP.

The following parameters can be supplied to the JCL procedure:

- **LOADLIB**
  Names the load library containing program GTXDMP.
- **REPHLQ**
  High level qualifier for the report.
- **AFL**
  Names the dataset to contain the Advanced File Layout for the dumped data.
- **DATA**
  Names the file to contain the dumped data.
- **LRECL**
  The logical record length of the file to contain the dumped data.
- **RECFM**
  The record format of the file to contain the dumped data.
- **BLK**
  The block size for the file to contain the dumped data.
- **CYL**
  The space allocation (in cylinders) for the file to contain the dumped data.

The dump job contains the following steps:

1. IEFBR14 to delete the report, Advanced File Layout, and data output files.
2. IEFBR14 to define the report, Advanced File Layout, and data output files.

3. GTXDMP to execute the input SQL, create an Advanced File Layout, and write the results of the SQL execution to DATA

GTXDMP Parameters

Output Formatting

- **AFLNAME=**
  Specifies the supplied string that is used as the logical file name that is used in the Advanced File Layout produced by the program. The file name suffixed with "_REC" as the record name in the AFL.
  Limits: 32 characters maximum
- **DELIMITER=**
  Specifies if the result set data items are delimited by the indicated single character.
  Default: No delimiters
- **NULLPREFIX=**
  Specifies if each result set data item in the output data is prefixed by a one-byte null-flag field.
  Values: N, Y
  Default: N
  Note: Set to Y if a returned data item is null.
- **FIELDNAME_nnn=**
  Specifies the supplied string to be used as the name of the field at the specified position in the output AFL field definition.
  Values: nnn represents a 3 digit positive number
  Limits: 32 characters maximum length
  Note: By default, the program uses column names in the output AFL. If the query does not include identifiable columns, fields names F_001 to F_999 are used.

Diagnostics

- **DIAGLEVEL=n**
  Specifies the volume of diagnostics that are produced.
  Values: 0 (diagnostics are written to SYSOUT), 1, 2, 3, 4 (highest level of diagnostics)
- **PROGRESSCOUNT=nnnn**
  Specifies the interval of rows at which a line is written to SYSOUT that contains the number of rows read and the time.

Other

- **LANGUAGE=**
  Specifies the two-character language code that is used for output messages.
  Values: EN, DE, ES, IT
  Default: EN

Print Flat Files (DB2/VSAM)

To run flat file printing, JCL is supplied in the installation package as GTXPRT. This job uses procedure GTPRT.

You can supply the following parameters to the JCL procedure:

- **LOADLIB**
  Names the load library that contains programs GTXDEF and GTXPRT.
- **MSGDS**
  TDM Message content VSAM.
- **INFILE**
  Names the file to be printed.
- **DEFFILE**
  Names the dataset that contains the record definition CSV (Advanced File Layout).
- **REPHLQ**
  Gives the high-level dataset name qualifier to be used for the audit and report files.
- **REPPRI**
  The primary space allocation (in cylinders) for the print output.
- **REPSEC**
  The secondary space allocation (in cylinders) for the print output.
- **LRECL**
  The logical record length for the print output. Set this parameter to a minimum of 40, plus the logical record length of the file to be printed.
- **RECFM**
  The record format of the print output dataset.
- **BLK**
  The block size of the print output dataset.

The print job contains the following steps:

1. IEFBR14 to delete the report and output print files.
2. IEFBR14 to define the report and output print files.
3. GTXDEF to read and parse the record definition CSV, and write the CSV out to a fixed record format file. See the section GTXDEF Parameters.
4. GTXPRT to read the input file and write a formatted representation of the file contents to the output print file. See the section GTXPRT Parameters.

GTXPRT Parameters

Subset Selection

The following parameters let you control the number of records that are printed.

Note: By default, all fields in all records that are identified by the record definition CSV are printed. Since the default results in high output, we recommend to print only a small subset of a file.

- **SELECT=RECORD:*nnn***
  Specifies the number of the record to print.
  Example: RECORD1000 will print just the 1000th record in the input file.
- **SELECT=SAMPLE:*nnn***
  Specifies the interval at which records are sampled.
  Example: SAMPLE100 will sample every 100th record.
  Note: For multi-record format files, every 100th record of each type that is found in the file will be sampled.

Output Control

- **FIELD=**
  Restricts the printing to specific fields
  Values: A record name and a field name separated by a period (.).

Example: RECA.FIELDB
Default: By default, the program samples data from all record types that are defined in the input record definition CSV. The program also samples data from all fields that are defined for those records.
Note: The record name or field name can be replaced by an asterisk to act as a wild card.
Examples: *.FIELDB", "RECA.*
Note: You can repeat the FIELD parameter up to 100 times.

- **INPUTCODESET=**
  Specifies the format for the codeset input. **Values:** EBCDIC, ASCII
  Default: EBCDIC
- **INVALIDONLY=**
  Specifies whether to report only invalid field values.
  Values: Y (only invalid field values are reported), N
  Default: Y
  Note: By default, all field values for all selected records are printed.
- **PRINTHEX=**
  Specifies whether the output report contains a hex representation of the file contents.
  Values: Y, N
  Default: Y

Dates

- **BASECENTURY=*nn***
  Specifies how BASECENTURY indicates the starting point for the century digit if the record definitions contain dateformats with a single digit century.
  Example: BASECENTURY=19 and a dateformat of "CYYMMDD","1880729" is interpreted as 29th July 1988.

Other

- **LANGUAGE=**
  Specifies the two-character language code that is used for output messages.
  Values: EN, DE, ES, IT
  Default: EN
- **PAGELIMIT=*nnn***
  Specifies the number of pages that the output report file contains.
  Note: Use this parameter to override the default page limit.
  Values: a number *nnn*
  Default: 50

Mainframe Messages

Use the search box to search for a message ID:

Mainframe Messages

Test Data Manager messages include one of the following suffixes in the message identifier:

- **I**
  Informational messages.
- **W**
  Warning messages.
- **E**

Error messages.
- **S**
  Severe error messages.
- **D**
  Debugging messages.

0001E0

Validation failed, processing abandoned
Reason:
Various
Action:
Check for other messages indicating the reason for failure

0002I0

All masking will be reported in audit file
Reason:
Parameter card option AUDIT=ALL used
Action:
None – information only

0003I0

Audit reporting row count used %1
Reason:
Parameter card option AUDIT=ROWnnnn (where nnnn is an integer) used
Action:
None – information only

0004I0

Audit reporting sample used %1
Reason:
Parameter card option AUDIT=SAMPLEnnnn (where nnnn is an integer) used
Action:
None – information only

0005I0

No audit reporting will be performed
Reason:
No AUDIT parameter card option specified
Action:
None – information only

0006I0

WHERE conditions will be used to apply subsetting
Reason:
Parameter card setting WHEREASSUBSET=Y
Action:

None – information only

0007I0

Commit frequency used %1
Reason:
Parameter card COMMIT set
Action:
None – information only

0008I0

Invalid fields will be retained
Reason:
Parameter card option KEEPINVALID set to Y
Action:
None – information only

0009I0

Invalid fields will not be retained
Reason:
Parameter card option KEEPINVALID set to N or is absent
Action:
None – information only

0010I0

Invalid fields will be reported
Reason:
Parameter card option REPORTINVALID set to Y
Action:
None – information only

0011I0

Invalid fields will not be reported
Reason:
Parameter card option REPORTINVALID set to N or is absent
Action:
None – information only

0012I0

Shuffle tables will be populated, but no masking will be done
Reason:
Parameter card option SHUFFLEONLY set to Y
Action:
None – information only

0013I0

Validation will be performed, but no masking will be done
Reason:
Parameter card option VALIDATEONLY set to Y
Action:
None – information only

0014I0

Shuffle tables will be populated using distinct values
Reason:
Parameter card option SHUFFLEDISTINCT set to Y
Action:
None – information only

0015I0

Shuffle limit used %1
Reason:
Parameter card option SHUFFLELIMIT=nnnnn (where nnnnn is an integer) used
Action:
None – information only

0016I0

Blanks in string fields will be treated as nulls
Reason:
Parameter card option BLANKSASNULLS set to Y
Action:
None – information only

0017I0

Seed schema used %1
Reason:
Parameter card option SEEDSCHEMA used
Action:
None – information only

0018I0

XREF schema used %1
Reason:
Parameter card option XREFSCHEMA used
Action:
None – information only

0019I0

Cross referencing will be case insensitive
Reason:

Parameter card option CASEINSENSITIVEXREF set to Y
Action:
None – information only

0020I0

LOV1 functions will be case insensitive
Reason:
Parameter card option CASEINSENSITIVESEED set to Y
Action:
None – information only

0021I0

Cross referenced values will be trimmed
Reason:
Parameter card option TRIMMEDXREF set to Y
Action:
None – information only

0022I0

Date used to replace invalid dates %1
Reason:
Parameter card option BADDATESTRING used
Action:
None – information only

0023I0

Current date used %1
Reason:
Parameter card option CDATE used
Action:
None – information only

0024I0

High date used %1
Reason:
Parameter card option HIGHDATE used
Action:
None – information only

0025I0

Low date used %1
Reason:
Parameter card option LOWDATE used
Action:
None – information only

0026I0

Language selected %1
Reason:
Parameter card option LANGUAGE used
Action:
None – information only

0027E0

Maximum number of mappings (%1) exceeded
Reason:
More masking rules than the program can process have been supplied in the mapping CSV
Action:
Reduce the number of masking rules applied in one run, by splitting the masking into two or more runs

0028E0

Maximum number of map blocks (%1) exceeded
Reason:
More blocks (different tables/records, or tables/records with WHERE clauses) than the program can process have been supplied in the mapping CSV
Action:
Reduce the number of blocks masked in one run, by splitting the masking into two or more runs

0029E0

Maximum number of record definitions (%1) exceeded
Reason:
More record types are being masked than the program can process
Action:
Reduce the number of records masked in one run, by splitting the masking into two or more runs

0030E0

Maximum number of fields used in mappings (%1) exceeded
Reason:
More field types are being masked than the program can process
Action:
Reduce the number of fields masked in one run, by splitting the masking into two or more runs

0031E0

Maximum number of record type definitions (%1) exceeded
Reason:
More conditions defining a record type have been supplied than the program can handle
Action:
Check that the AFL (Advanced File Layout) / DEFCSV definitions are correct

0032E0

Maximum number of field definitions (%1) exceeded

Reason:
More fields are defined than the program can handle
Action:
Reduce the number of fields masked in one run, by editing the AFL (Advanced File Layout) / DEFCSV and splitting the masking into two or more runs

0033E0

Record definition record type (%1) not recognised
Reason:
An unrecognised record type has been found in the file definition (AFL / DEFCSV)
Action:
Check that the file definition (AFL / DEFCSV) being used is correct and that the file definition parser program (GTXDEF) has completed correctly

0034E0

Error opening file %1
Reason:
Error encountered attempting to open a file
Action:
Check that the JCL being run is correct, that the file referred to exists, is accessible and has appropriate DCB information

0035E0

Error closing file %1
Reason:
Error encountered attempting to close a file
Action:
Check that the JCL being run is correct, that the file referred to exists, is accessible and has appropriate DCB information

0036I0

%1 records processed for %2
Reason:
Information about the number of records processed in a block of masking operations
Action:
None – information only

0037I0

WHERE %1
Reason:
Information about the WHERE clauses used to define a block of masking operations
Action:
None – information only

0038I0

%1 shuffle rows written for %2
Reason:
Information about the number of rows/records written in a shuffle operation

Action:
None – information only

0039W0

Dynamic array count is invalid for record no %1, record not masked
Reason:
The dynamic array count found in a record is not numeric, is negative or does not match the record length
Action:
Check that the record definition (AFL / DEFCSV) is correct; that the dynamic array count field is correctly defined; and that the file being processed matches the record definition. If the file contains invalid data consider correcting the file prior to masking

0040W0

Record no %1, invalid date, field %2 contains %3
Reason:
A field with a dateformat supplied contains an invalid date
Action:
Check that the record definition ((AFL / DEFCSV) is correct, and that the file being processed matches the definition. If the file contains invalid data consider correcting the file prior to masking

0041I0

Shuffle seedlist used %1
Reason:
Mapping CSV includes a SHUFFLE operation
Action:
None – information only

0042I0

Seedlist used %1
Reason:
Mapping CSV includes a seedlist function (RANDLOV, SEQLOV, RANDLOV1, SEQLOV1, HASHLOV, HASHLOV1)
Action:
None – information only

0043E0

Record no %1, invalid date, field %2 contains %3
Reason:
The mapping CSV names a record which is not defined in the supplied record definition (AFL / DEFCSV)
Action:
Check that the correct mapping CSV and record definition files are being used

0044E0

Field definition for %1 not supplied
Reason:
The mapping CSV names a field which is not defined in the supplied record definition (AFL / DEFCSV)
Action:

Check that the correct mapping CSV and record definition files are being used

0045E0

A maximum of 30 columns can be handled in one shuffle
Reason:
The same shuffle identifier is being used from more than 30 columns in one operation
Action:
Correct the mapping CSV

0046E0

Error applying %1 to %2
Reason:
The masking function supplied in the mapping CSV is not supported, is not appropriate to the datatype of the column/field being masked, or the required function parameters have not been correctly supplied
Action:
Check for other messages indicating the precise cause of the error. Correct the mapping CSV

0047E0

Parameter %1 must be numeric, valued supplied=%2
Reason:
A masking function parameter which should be numeric is not
Action:
Correct the mapping CSV

0048E0

Parameter %1 must be set
Reason:
A mandatory parameter to a masking function is missing
Action:
Correct the mapping CSV

0049E0

Field must be numeric
Reason:
Attempting to apply a numeric masking function to a non-numeric field
Action:
Correct the mapping CSV

0050E0

Field must be an integer
Reason:
Attempting to apply an integer masking function to a non-integer field
Action:
Correct the mapping CSV

0051E0

Field must be a date
Reason:
Attempting to apply a date masking function to a non-integer field
Action:
Correct the mapping CSV

0052E0

Field must be a character type
Reason:
Attempting to apply a string masking function to a non-string field
Action:
Correct the mapping CSV

0053E0

Substring only supported for character types
Reason:
The substr start or substr length fields have been set for a masking function against a column/field which is not a character type
Action:
Correct the mapping CSV

0054E0

Parameter %1 should be a date in format %2
Reason:
A function requiring a date as a parameter has not been supplied with a valid date in the correct format
Action:
Correct the mapping CSV

0055E0

Unsupported function %1
Reason:
The mapping CSV names a masking function which is not supported
Action:
Correct the mapping CSV

0056E0

Seedlist %1 not found
Reason:
The mapping CSV includes a seedlist function (RANDLOV, SEQLOV, RANDLOV1, SEQLOV1, HASHLOV, HASHLOV1) which names a seedlist which cannot be found on the seedlist table
Action:
Check that the correct seedlist name has been supplied in the mapping CSV.
Check that the SEEDSCHEMA specified in the PARMCD file is correct.
Check that the program has been bound correctly and has access to the seedlist table (GTSRC_REFERENCE_LOV1) / VSAM

0057E0

Parameter %1 must be an integer, value supplied=%2
Reason:
The named masking function requires an integer parameter, but this has not been correctly supplied
Action:
Correct the mapping CSV

0058E0

Only one shuffle per block can be specified, %1 and %2 defined
Reason:
More than one SHUFFLE function has been defined for a given block
Action:
Split the run into two or more runs each of which defines only one SHUFFLE per block

0059E0

Parameter %1 must be unique for each field in a multi-field shuffle
Reason:
A SHUFFLE function has been incorrectly defined in the mapping CSV
Action:
Correct the mapping CSV

0060E0

Job started at %1 and ended at %2
Reason:
The job step has completed
Action:
None – information only

0061S0

Unrecoverable error, processing ending
Reason:
A serious error has occurred in the run
Action:
Check for other error messages and take appropriate action

0062E0

Option %1 should be set to Y or N
Reason:
An option in the PARMCD file which should be set to "Y" or "N" contains another value
Action:
Correct the PARMCD file

0063E0

Option %1 should be a positive integer
Reason:

An option in the PARMCD file which should provide an integer contains another value
Action:
Correct the PARMCD file

0064E0

Option %1 not recognized
Reason:
The PARMCD file contains an invalid option
Action:
Correct the PARMCD file

0065E0

Option %1 should contain a valid date in format %2
Reason:
An option in the PARMCD file which should provide a date contains another value
Action:
Correct the PARMCD file

0066E0

Options %1 and %2 cannot both be Y
Reason:
Inconsistent options have been given in the PARMCD file
Action:
Correct the PARMCD file

0067E0

SQL error executing %1
Reason:
An error has occurred in executing an SQL query
Action:
Check for other error messages and take appropriate action. Check that the program has been correctly bound and has the required authority to execute the query

0068E0

SQLCODE=%1
Reason:
An error has occurred in executing an SQL query
Action:
Check for other error messages and take appropriate action. Check that the program has been correctly bound and has the required authority to execute the query

0069E0

SQLERRMC=%1
Reason:
An error has occurred in executing an SQL query
Action:

Check for other error messages and take appropriate action. Check that the program has been correctly bound and has the required authority to execute the query

0070I0

Report page limit set to %1
Reason:
PAGELIMIT PARM option set
Action:
None – information only

0071E0

Module %1 update error %2
Reason:
An error occurred updating GTSRC_REFERENCE_LOV1 or GTSRC_XREF
Action:
Check that the program has been correctly bound and has the required authority to execute the query

0072E0

Module %1 delete error %2
Reason:
An error occurred deleting from GTSRC_REFERENCE_LOV1 or GTSRC_XREF
Action:
Check that the program has been correctly bound and has the required authority to execute the query

0073E0

Module %1 insert error %2
Reason:
An error occurred inserting into GTSRC_REFERENCE_LOV1 or GTSRC_XREF
Action:
Check that the program has been correctly bound and has the required authority to execute the query

0074E0

Module %1 error getting schema %2
Reason:
An error occurred attempting to get the current schema
Action:
Check that the program has been correctly bound and has the required authority to execute the query

0075E0

Module %1 error setting schema %2
Reason:
An error occurred attempting to set the current schema
Action:
Check that the program has been correctly bound and has the required authority to execute the query

0076E0

Module %1 open error %2
Reason:
An error occurred attempting to open a cursor
Action:
Check that the program has been correctly bound and has the required authority to execute the query

0077E0

Module %1 fetch error %2
Reason:
An error occurred attempting to fetch a row from a cursor.
Action:
Check that the program has been correctly bound and has the required authority to execute the query

0078E0

Seed table %1 not found
Reason:
An error occurred attempting to fetch a row from a cursor
Action:
Check that the program has been correctly bound and has the required authority to execute the query

0079E0

Module %1 count error
Reason:
An error occurred executing a SELECT COUNT query against GTSRC_REFERENCE_LOV1
Action:
Check that the program has been correctly bound and has the required authority to execute the query

0080E0

Unrecognised masking function %1
Reason:
An unsupported masking function has been found in the mapping CSV
Action:
Correct the mapping CSV

0081E0

Value cannot be used to mask a numeric field - %1
Reason:
A masking function has returned a value which can't be used to update a numeric field
Action:
Correct the mapping CSV

0082E0

Parm%1 must contain %2 or %3
Reason:

Incorrect parameters have been supplied to a masking function in the mapping CSV
Action:
Correct the mapping CSV

0083E0

Length of Parm%1 must equal length of Parm%2
Reason:
Incorrect parameters have been supplied to a masking function in the mapping CSV
Action:
Correct the mapping CSV

0084W

Record no %1, field %2, invalid -%3
Reason:
A record contains an invalid field (either a non-numeric value in a numeric field or an invalid date in a date field)
Action:
Correct the input file, correct the record definition (DM.txt), or accept the warning and continue

0085E

Min >= max for random function
Reason:
Internal program error
Action:
Refer to support @grid-tools.com

0086E

Seed value <0 for random function
Reason:
Internal program error
Action:
Refer to support @grid-tools.com

0087E

Requested seedlist ID invalid %1
Reason:
Attempting to use a seedlist which doesn't exist (function RANDLOV, RANDLOV1, SEQLOV, SEQLOV1, HASHLOV, HASHLOV1 or SHUFFLE)
Action:
Correct the mapping CSV; add the missing seedlist data

0088E

Requested seedlist column invalid
Reason:
Attempting to access a seedlist (function RANDLOV, RANDLOV1, SEQLOV, SEQLOV1, HASHLOV, HASHLOV1 or SHUFFLE) with a column number < 1 or > 30
Action:

Correct the mapping CSV

0089E

Too many sequences used in one run
Reason:
More SEQNUMBER masking functions used than the program can support
Action:
Split the masking into two runs

0090E

Variance parm must be between 1 and 99
Reason:
The VARIENCE (or VARIANCE) masking function has been used with an invalid parameter
Action:
Correct the mapping CSV

0091E

Requested lookup value invalid
Reason:
The value supplied to RANDLOV1, SEQLOV1 or HASHLOV1 as the lookup value cannot be used
Action:
Correct the mapping CSV

0093I

Subset schema used %1
Reason:
SUBSETSCHEMA option supplied in PARMCD file
Action:
None – information only

0094I

Schema used %1
Reason:
SCHEMA option supplied in PARMCD file
Action:
None – information only

0095E

Maximum number of mappings in one block (%1) exceeded
Reason:
More masking rules for one block (i.e., for one table or a table with a WHERE clause) than the program can process have been supplied in the mapping CSV
Action:
Reduce the number of masking rules per block applied in one run, by splitting the masking into two or more runs

0096E

Maximum number of columns in one block (%1) exceeded
Reason:
A table contains more columns than the program caters for.
Action:
Refer to support@grid-tools.com

0097E

Maximum number of columns in one bloc k (%1) exceeded
Reason:
Mapping CSV names a table which can't be found in the catalogue tables
Action:
Check that the mapping CSV is correct, that the program is correctly bound and has authority to access the table named

0098E

Column not found %1
Reason:
Mapping CSV names a column which can't be found in the catalogue tables
Action:
Check that the mapping CSV is correct, that the program is correctly bound and has authority to access the table holding the column named. Check the schema is appropriate for that table.

0099E

Table to mask must have a primary key, unique index or unique columns defined - %1
Reason:
A table being masked does not have a primary key or unique index
Action:
In the mapping CSV supply a list of unique columns for the table

0100E

Maximum select statement size (%1) exceeded
Reason:
Mapping CSV names a column which can't be found in the catalogue tables
Action:
Check that the mapping CSV is correct, that the program is correctly bound and has authority to access the table holding the column named

0101E

Maximum Update statement size (%1) exceeded
Reason:
A generated SQL update statement is larger than the program can process
Action:
Reduce the number of mappings being applied in one run by splitting the masking into two or more runs

0102E

Column must be numeric
Reason:
Attempting to apply a numeric masking function to a non-numeric column
Action:
Correct the mapping CSV

0103E

Column must be an integer
Reason:
Attempting to apply an integer masking function to a non-numeric column
Action:
Correct the mapping CSV

0104E

Column must be a date
Reason:
Attempting to apply a date masking function to a non-date column (or a column without a dateformat supplied in the mapping CSV)
Action:
Correct the mapping CSV

0105E

Column must be a character type
Reason:
Attempting to apply a string masking function to a non-string column
Action:
Correct the mapping CSV

0106W

Masking of a TIME-type column not supported
Reason:
Attempt to mask a column with a TIME datatype
Action:
This is not currently supported. Correct the mapping CSV

0107W

Masking of a FLOAT-type column not supported
Reason:
Attempt to mask a column with a FLOAT datatype
Action:
This is not currently supported. Correct the mapping CSV

0108E

Maximum length of buffer (%1) exceeded

Reason:
Internal program error
Action:
Contact support@grid-tools.com

0109I

%1 rows processed for %2
Reason:
Information about the number of rows process in a given mapping block
Action:
None – information only

0110E

Invalid date (%1) found in processing shuffle
Reason:
A shuffle includes a date column (either with a date datatype or with a dateformat in the mapping CSV), and a row/rows contains an invalid date
Action:
Correct the data, or accept the error and continue

0111I

%1 shuffle rows added for block %2
Reason:
Information about the number of rows process in a given SHUFFLE block
Action:
None – information only

0112W

Data conversion required by SQL function not supported – function
ignored
Reason:
An SQLFUNCTION specified in the mapping CSV requires a data conversion in order to apply the result to the target column
Action:
In the mapping CSV specify the SQLFUNCTION including the required data conversion

0113I

Target schema %1
Reason:
TARGETSCHEMA option supplied in the PARMCD file
Action:
None – information only

0114E

Maximum number of tables (%1) exceeded
Reason:

More tables are specified in the mapping CSV or in the subset rules than the program can support
Action:
Reduce the number of tables processed in one masking/subsetting job by splitting the operation into two or more jobs

0115E

Error reading file %1
Reason:
Error encountered reading a file
Action:
Check that the file exists, is named correctly in the JCL and has appropriate DCB information

0116W

Table to mask not in subset, masking ignored for %1
Reason:
Attempting to apply a subset (APPLYSUBSETRULES=Y is supplied as an option in the PARMCD file) as well as masking, but a table named in the masking rules is not included in the subset
Action:
Check that the correct subset rules are being used for the given mapping CSV

0117E

Column has unsupported datatype %1
Reason:
Attempting to unload a table which includes a column with an unsupported datatype (for example a CLOB or BLOB)
Action:
Refer to support@grid-tools.com

0118W

%1 datatype not supported, %2 will not be exported
Reason:
Attempting to unload a table which includes a column with an unsupported datatype (for example a CLOB or BLOB)
Action:
Refer to support@grid-tools.com

0119E

Maximum number of columns in one table (%1) exceeded
Reason:
A table contains more columns than the program can process
Action:
Refer to support@grid-tools.com

0120E

Lookup column not found %1
Reason:
A lookup function (RANDLOV1, SEQLOV1 or HASHLOV1) refers to a column which can't be found in the catalog tables
Action:

Check that the mapping CSV is correct, that the program is correctly bound and has authority to access the table holding the column named

0121E

Shuffle processing not supported
Reason:
Attempting to apply a SHUFFLE function using the DB2 unload masking/subsetting program (GTXMSKL)
Action:
Use the DB2 mask in-place program (GTXMSK) to populate a seedlist for the SHUFFLE (create a mapping CSV with the required SHUFFLE and run the program with the SHUFFLEONLY=Y option in the PARMCD file), then use the SEQLOV function in the DB2 unload run in place of the SHUFFLE function

0122I

%1 records written for table %2
Reason:
Information about the number of records written for a given subset/masking operation
Action:
None – information only

0123E

QUOTESTYLE option should specify "SINGLE" or "DOUBLE"
Reason:
QUOTESTYLE option in the PARMCD file contains a value other than "SINGLE" or "DOUBLE"
Action:
Correct the supplied PARMCD options

0124E

Line %1 – expected file definition record not found
Reason:
Invalid file definition (AFL/ DEFCSV file)
Action:
Correct the file definition

0125E

More than %1 fields defined for record %2
Reason:
Record definition (DM.txt file) defines more fields for a record than the program can process
Action:
Refer to support@grid-tools.com

0126E

Line %1 – unexpected record type found
Reason:
Invalid file definition (AFL/ DEFCSV file)
Action:
Correct the file definition

0127E

More than %1 record type conditions defined for record %2
Reason:
Record definition (AFL /DEFCSV) defines more type conditions for a record than the program can process
Action:
Check that the file definition is correct. Refer to support @grid-tools.com

0128E

Line %1 – fewer than %2 columns present in CSV
Reason:
Invalid file definition (AFL /DEFCSV)
Action:
Correct the file definition

0129E

Line %1 – referenced field not found - %2
Reason:
Invalid record definition (AFL / DEFCSV) dynamic array count field referred to not defined
Action:
Correct the file definition

0130E

Line %1 – boolean operator missing from type definitions
Reason:
Record definition (AFL / DEFCSV) type condition missing an expected boolean operator ("AND" or "OR")
Action:
Correct the file definition

0131E

Line %1 – value for key field %2 must be numeric
Reason:
Record definition (AFL / DEFCSV) type condition includes a non-numeric value as the comparator for a numeric field
Action:
Correct the file definition

0132E

Line %1 – value for key field %2 must be a date in format %3
Reason:
Record definition (AFL / DEFCSV)) type condition includes a non-date value as the comparator for a date field
Action:
Correct the file definition

0133E

Line %1 – expected keyword %2 missing
Reason:

Invalid file definition (AFL / DEFCSV)
Action:
Correct the file definition

0134E

Line %1 – expected keyword %2 but found %3
Reason:
Invalid file definition (AFL / DEFCSV)
Action:
Correct the file definition

0135E

Line %1 – value for %2 missing
Reason:
Invalid file definition (AFL / DEFCSV)
Action:
Correct the file definition

0136E

Line %1 – value for %2 must be "Y" or "N"
Reason:
Invalid file definition (AFL / DEFCSV)
Action:
Correct the file definition

0137E

Line %1 – allowable boolean operators are "AND" and "OR"
Reason:
Record definition (AFL / DEFCSV) type condition has an invalid boolean operator
Action:
Correct the file definition

0138E

Line %1 – comparison operator missing
Reason:
Record definition (AFL / DEFCSV) type condition missing expected comparison operator (valid operators are "EQ", "NE", "GE", "GT", "LE" and "LT")
Action:
Correct the file definition

0139E

Line %1 – allowable comparison operators are "EQ", "NE", "GT", "LT",
"GE" and "LE"
Reason:
Record definition (AFL / DEFCSV) type condition missing expected comparison operator (valid operators are "EQ", "NE", "GE", "GT", "LE" and "LT")

Action:
Correct the file definition

0140E

Line %1 – %2 must be numeric
Reason:
Record definition (AFL / DEFCSV) contains a non-numeric value where a numeric value is expected
Action:
Correct the file definition

0141E

Line %1 – %2 must be an integer
Reason:
Record definition (AFL / DEFCSV) contains a non-integer value where a numeric value is expected
Action:
Correct the file definition

0142E

Line %1 – invalid datatype found - %2
Reason:
Record definition (AFL / DEFCSV) contains an unrecognised datatype. Recognised datatypes are "STRUCTURE", "STRING", "VARSTRING", "PACKEDSIGNED", "PACKEDUNSIGNED", "BINARYSIGNED", "BINARYUNSIGNED", "FLOATBIN", "FLOATDEC", "NUMERIC", "ZONED", "BIT", "VARBIT" and "POINTER"
Action:
Correct the file definition

0143E

Line %1 – maximum CSV field length exceeded
Reason:
Invalid file definition (AFL / DEFCSV)
Action:
Correct the file definition

0144E

CSV header doesn't include "%1" column
Reason:
Invalid mapping CSV
Action:
Correct the file definition

0145E

Line %1 – fewer fields given than defined in the CSV header
Reason:
Invalid mapping CSV
Action:
Correct the file definition

0146E

Line %1 – more than %2 records defined
Reason:
More records defined (in the AFL / DEFCSV) than the program can process
Action:
Split the profiling job into more than one job each processing fewer record types

0147I

No subsetting will apply
Reason:
SUBSET= option supplied in PARMCD file
Action:
None – information only

0148I

Subset row count used %1
Reason:
SUBSET=ROW:NNN option supplied in PARMCD file
Action:
None – information only

0149I

Subset sample used %1
Reason:
SUBSET=SAMPLE:NNN option supplied in PARMCD file
Action:
None – information only

0150E

No fields specified for sampling
Reason:
The FIELD= option specified in the PARMCD file in conjunction with the supplied record definition (AFL / DEFCSV) doesn't result in any fields being selected for sampling
Action:
Correct the PARMCD options supplied and/or the record definition used

0151W

Definition not found for field %1
Reason:
Field specified in the FIELD= option in the PARMCD file is not defined in the supplied record definition (AFL / DEFCSV)
Action:
Correct the PARMCD options supplied and/or the record definition used

0152I

Fields to report for record %1

Reason:
Information about the fields selected for sampling
Action:
None – information only

0153I

Reco r ds of type %1 read %2
Reason:
Information about the number of records read
Action:
None – information only

0154I

Records of type %1 SAMPLED %2
Reason:
Information about the number of records sampled
Action:
None – information only

0155W

Dynamic array count is invalid for record no %1, record not
sampled
Reason:
The dynamic array count found in a record is not numeric, is negative or does not match the record length
Action:
Check that the record definition (ADL / DEFCSV) is correct, that the dynamic array count field is correctly defined, and that the file being processed matches the record definition. If the file contains invalid data consider correcting the file prior to profiling

0156W

More than %1 fields selected only %2 will be sampled
Reason:
More fields selected for sampling than the program can process
Action:
Split the profiling job into more than one run each handling a smaller number of records/fields

0157W

Fields should be specified as RECORD. FIELD % will be ignored
Reason:
FIELD= option incorrectly specified in the PARMCD file
Action:
Correct the PARMCD options

0158I

Profiling option used %1 %2
Reason:

Information about the options used in profiling
Action:
None – information only

0159W

Dynamic array count is invalid %1
Reason:
The dynamic array count found in a record is not numeric, is negative or does not match the record length
Action:
Check that the record definition (AFL / DEFCSV) is correct, that the dynamic array count field is correctly defined, and that the file being processed matches the record definition. If the file contains invalid data consider correcting the file prior to printing

0160W

Record length (%1) , defined record length (%2)
Reason:
The length of a record is less than expected
Action:
Check that the record definition (AFL / DEFCSV) is correct for the file being processed

0161E

Target dynamic array count is invalid %1
Reason:
A valid value has not been supplied for a dynamic array count field in the target record
Action:
Check that the source and target record definitions are correct and that the value to be assigned to a target dynamic array count field is valid

0162W

No target definition supplied for %1, this record type will not be
written
Reason:
A record in the source record definition is not defined in the target record definition
Action:
If the record should be written to the target then correct the target record definition

0163W

No target definition supplied for %1, this field will not be populated
Reason:
A field in the source record definition is not defined in the target record definition
Action:
If the record should be written to the target then correct the target record definition

0164W

No source definition supplied for %1, this record will be initialised
Reason:

A record in the target record definition is not defined in the source record definition
Action:
If the record should be populated from the source then correct the source record definition

0165W

No source definition supplied for %1, this field will be initialised
Reason:
A field in the target record definition is not defined in the source record definition
Action:
If the field should be populated from the source then correct the source record definition

0166E

Error writing file %1
Reason:
An error occurred attempting to write to a file
Action:
Check that the file exists, that the dataset name has been correctly given in the JCL and that the file has appropriate DCB characteristics

0167W

Output record length (%1) exceeds LRECL, record will be truncated
Reason:
The target record definition defines a record which is longer than the LRECL of the output file
Action:
Correct the target record definition of the output file LRECL

0168I

%1 mapped to %2
Reason:
A source field has been mapped to a target field
Action:
None – information only

0169I

Records of type %1 written %2
Reason:
Information about the number of records written
Action:
None – information only

0170E

Record no %1 too short to contain field %2, masking rule ignored
Reason:
The record definition (AFL / DEFCSV) defines a field which doesn't exist for a given record
Action:
Check that the record definition is correct for the file being masked

0171E

Option %1 must be supplied
Reason:
A required PARMCD option has not been supplied
Action:
Correct the PARMCD file

0172E

COMBINEVALS cannot be used in conjunction with substring
Reason:
Substring cannot be used within the COMBINEVALS block
Action:
Remove the Substring from the mapping

0173E

Nested brackets not supported in file WHERE clauses
Reason:
Only 1 level of brackets is permitted
Action:
Reduce the number of brackets

0174E

Invalid file WHERE clause
Reason:
Closing bracket ")" found without opening bracket "("
Action:
Correct the WHERE statement logic

0175I

Bulking factor %1
Reason:
Option BULKINGFACTOR supplied
Action:
None – information only

0176E

Invalid subscript reference %1
Reason:
Either the array doesn't exist; or the subscript is out of the array bounds
Action:
Correct the subscript in the mapping file

0177E

Invalid column datatype for subset list processing %1
Reason:

Column datatype is not permitted for subset list processing
Action:
Convert the column data into a numeric or char / varchar format for the subset list processing.

0178E

Invalid SQL for subset list processing %1
Reason:
SQL executed produced an SQL error
Action:
Correct the provided SQL

0179E

Invalid SUBSTR specification %1 %2
Reason:
Substring start and length information don't conform (E.g. start provided and no length)
Action:
Correct the substring details ensure either no substring info or both start and length are provided

0180E

Invalid parameter %1 for function %2
Reason:
Function has been specified incorrectly
Action:
Correct the function

0181E

Invalid PARMCD option %1 valid values are %2
Reason:
Invalid option provided
Action:
Correct the parameter with a valid value (listed in the message)

0182E

Invalid %1 record definition - value must be a positive integer
Reason:
The comparison values for **HEADER** or **TRAILER** must be a positive integer
Action:
Amend the **HEADER** or **TRAILER** definition

0183E

Invalid %1 record definition - comparison operator must be %2
Reason:
The comparison operator is not one of the accepted values
Action:
Change it to be one of the values listed in the message

0184I

PARMCD option %1, setting %2
Reason:
Lists the supplied parameters and their settings
Action:
None – information only

0185E

Record out of sequence, no %1, type %2
Reason:
Child record missing a parent record
Action:
Check the input file / parent child rules for the file

0190E

Parameter 1 must be an internal string variable reference %1 found
Reason:
For function ASSIGNSTR: internal string variables are of the format GT__STR_n (where n is a integer 1-9)
Action:
Update the internal variable name to meet the format (note 2 underscores between GT and STR

0191E

Parameter 1 must be an internal numeric variable reference %1 found
Reason:
For function ASSIGNCTR: internal numeric variables are of the format GT__CTR_n (where n is a integer 1-9)
Action:
Update the internal variable name to meet the format (note 2 underscores between GT and CTR

0192E

Target of SETSTR must be an internal string variable reference %1 found
Reason:
For function SETSTR: internal string variables are of the format GT__STR_n (where n is a integer 1-9)
Action:
Update the internal variable name to meet the format (note 2 underscores between GT and STR

0193E

Target of SETCTR must be an internal numeric variable reference %1 found
Reason:
For function SETCTR: internal numeric variables are of the format GT__CTR_n (where n is a integer 1-9)
Action:
Update the internal variable name to meet the format (note 2 underscores between GT and CTR

0194E

Target of ADDTOCTR must be an internal numeric variable reference %1 found
Reason:

For function ADDTOCTR: internal numeric variables are of the format GT__CTR_n (where n is a integer 1-9)
Action:
Update the internal variable name to meet the format (note 2 underscores between GT and CTR

0195I

All lookup/xref will be performed via VSAM
Reason:
Parameter option VSAMLOOKUP=Y was used. This forces all lookups to be performed via VSAM datasets instead of the normal DB2.
Action:
None – information only

0196I

Only the first %1 rows will be processed per table (PROCESSCOUNT)
Reason:
Parameter option PROCESSCOUNT was used to limit the processing performed by the extract job. The job will stop processing after it has processed (mask or subset or both) %1 rows of each table present in the subset / mapping file.
Action:
None – information only

0197I

Only the first %1 rows will be processed per block (PROCESSCOUNT)
Reason:
Parameter option PROCESSCOUNT was used to limit the processing performed by the masking job. The job will stop processing after it has processed (mask) %1 rows of each block in the mapping file.
Action:
None – information only

0198I

Processing stopped for %1 after %2 entries (PROCESSCOUNT)
Reason:
Parameter option PROCESSCOUNT was used to limit the processing performed by the masking job. The job reached the PROCESSCOUNT limit and stopped processing any further entries for that table.
Action:
None – information only

0199I

Only the first %1 records will be processed (PROCESSCOUNT)
Reason:
Parameter option PROCESSCOUNT was used to limit the processing performed by the masking job. The job will stop processing after it has processed (mask or subset) %1 records of the file. The job may read more than %1 records before it has processed the required number of records.
Action:
None – information only

0200I

Processing stopped after %1 records (PROCESSCOUNT)
Reason:
Parameter option PROCESSCOUNT was used to limit the processing performed by the masking job. The job reached the PROCESSCOUNT limit and stopped the run.
Action:
None – information only

0201I

Processing stopped after %1 rows (PROCESSCOUNT)
Reason:
Parameter option PROCESSCOUNT was used to limit the processing performed by the masking job. The job reached the PROCESSCOUNT limit and stopped the run.
Action:
None – information only

0202E

Invalid IBAN country code or invalid IBAN length
Reason:
Data in the field masked by CHECKIBAN contained invalid country code or was incorrect length for the country.
Action:
Check the data / masking of that field

Perform Mainframe Masking Jobs With Brightside

You can use another product, CA Brightside, to perform Test Data Manager masking jobs on data stored in DB2 tables on the Mainframe, without the need to submit them directly on the Mainframe.

> **WARNING**
>
> The same functionality is available with the ZOWE Command Line Interface. To use ZOWE instead of Brightside, replace all instances of `bright` in **mask.sh**, with `zowe` .

You can use Brightside to perform two different types of Mainframe masking:

- **In-place masking (GTXMSK)**
  This program masks data in the database where it is stored (program overwrites original data).
  For more information on this process in TDM, see Mask DB2 Tables in Place.
  For a guide to this process with CA Brightside, see In-place Mainframe masking with CA Brightside.
- **In-flight masking (GTXMSKL)**
  This program extracts data (or a subset of data) from DB2 tables, masks the data, and stores it in a dataset, from which you can load the data into another DB2 database.
  For more information on this process in TDM, see Mask and Unload DB2 Tables.
  For a guide to this process with CA Brightside, see In-flight Mainframe masking with CA Brightside.

Prerequisites

To mask mainframe data, you need to use TDM Datamaker (for Windows) to create the following:

- Transformation maps
- (In-flight masking only) Subset files

To mask Mainframe data with TDM, without submitting commands directly to the Mainframe, you need the following:

- CA Brightside (Community or Enterprise Edition)

  TIP

  For more information on installation of Brightside Command-Line Interface, see Installing CLI in the Brightside Enterprise documentation.

- CA Test Data Manager Mainframe DB2 Add On MVS 5.4.14

Mainframe masking workflow

To mask mainframe data, the typical workflow is as follows:

1. **Set up masking rules in Datamaker**.
   Creates a transformation map (**.csv** file).
2. (In-flight masking only) **Create a subset in GT Subset**. This requires these steps:
   a. Setup subset conditions to export a **.ext** file.
   b. Import your transformation map (.csv) and subset (.ext) files into Datamaker.
   c. Generate **subset.csv** and **subset.txt** files.
3. **Upload the transformation map (and subset file if necessary) to the Mainframe.**

   TIP

   You can perform this step with the CA Brightside command **`zos-files upload file-to-data-set`**.
   See Command Groups - zos-files.

   Example: `bright zos-files upload file-to-data-set map.csv "TDMHLQ.MAPCSV(MAPCSV)"`
4. **Customize the relevant JCL**. This is either GTXMSK (in-place masking) or GTXMSKL (in-flight masking).
   You perform this step either locally, or directly on the Mainframe. For these guides, we assume that you edit the file locally and upload it to the Mainframe with CA Brightside.
5. **Upload JCL file to the Mainframe.**

   TIP

   You can perform this step with the CA Brightside command **`zos-files upload file-to-data-set`**.
   See Command Groups - zos-files.

   Example: `bright zos-files upload file-to-data-set GTXMSKL.txt "TDMHLQ.JCL(GTXMSKL)"`
6. **Submit JCL file to the Mainframe.**

   TIP

   You can perform this step with the CA Brightside command **`zos-jobs submit data-set`**.
   See Command Groups - zos-jobs.

   Example: `bright zos-jobs submit data-set "TDMHLQ.JCL(GTXMSKL)"`
7. (In-flight masking only) **Submit DB2LOAD JCL to write masked data to your target DB2 database.**

   TIP

   You can perform this step with the CA Brightside command **`zos-jobs submit data-set`**.
   See Command Groups - zos-jobs.

   Example: `bright zos-jobs submit data-set "TDMHLQ.JCL(DB2LOAD)"`
8. (Optional) When the masking job is complete, you can review spool, report and audit files for the program(s) you submit.

TIP

You can perform these steps with CA Brightside. You can download reports and audit files with the CA Brightside command **`zos-files download ds`** . See Command Groups - zos-files.

Example: `bright zos-files download ds "$TDM_HLQ.TEMP.REPT"`

In-flight Mainframe masking with CA Brightside

You can mask data on the Mainframe in-flight, with CA Brightside. This page describes a typical in-place masking scenario and explains how the **mask.sh** script uses CA Brightside to perform this task.

Requirements

To mask Mainframe data in-place, you need the following:

- CA Test Data Manager Mainframe DB2 Add On MVS 5.4.14.
- CA Brightside (Community or Enterprise Edition).
- Valid Mainframe authorization.
- Transformation map.
- Subset file.
- GTXMSKL program, customized for your masking job.
- Datasets on the Mainframe under your HLQ, to receive the files you upload.

Execution

GTXMSKL program

To execute an in-place masking job on Mainframe, you need to upload:

- Transformation map (**.csv** file).
- Subset file (**.txt** file).
- **GTXMSKL.txt** JCL.

The GTXMSKL program includes the following hardcoded parameters:

- JOBCARD
- TDM loadlib, jcllib
- TEMP HLQ for reports and audit (REPHLQ)
- DSN for transformation map (MAPDS)
- DB2 steplib (STEP05.STEPLIB)

Mask.sh script

To execute a masking job on Mainframe, you can modify and execute the provided **mask.sh** bash script. This script performs the following actions:

1. **Mask Data**
   a. Uploads the following files to datasets on the Mainframe:
      - Transformation map (.csv).
      - Subset file (.txt).
      - GTXMSKL.txt.
   b. Submits GTXMSKL JCL to Mainframe to start masking job.

c. Generates local spool files, report and audit file for GTXMSKL.

2. **Upload masked data to DB2 database**
   a. Clears target DB2 table.
   b. Uploads DB2LOAD.txt to a dataset on the Mainframe.
   c. Submits DB2LOAD JCL to Mainframe to write masked data to DB2 target table.
   d. Generates local spool files for DB2LOAD.
   e. Executes SQL command to show contents of target table.

mask.sh script

mask.sh

```
#! /bin/env bash




TDM_HLQ=BROADCOM.TDM




#upload jcl

echo "Uploading MASK JCL GTXMSKL.txt -> $TDM_HLQ.JCL(GTXMSKL)"

bright zos-files upload file-to-data-set GTXMSKL.txt "$TDM_HLQ.JCL(GTXMSKL)"




#upload map file

echo "Uploading maping file map.csv -> $TDM_HLQ.MAPCSV(MAPCSV)"

bright zos-files upload file-to-data-set map.csv "$TDM_HLQ.MAPCSV(MAPCSV)"




#submit our job
```

```
jobid=$(bright zos-jobs submit data-set "$TDM_HLQ.JCL(GTXMSKL)" --rff jobid --rft
 string)




echo "Submitted masking job, JOB ID is $jobid"




#wait for it to go to output

status="UNKNOWN"

while [[ "$status" != "OUTPUT" ]] ; do

    echo "Checking status of job $jobid"

    status=$(bright zos-jobs view job-status-by-jobid "$jobid" --rff status --rft
 string)

    echo "Current status is $status"

    sleep 5s

done;




echo "Job completed in OUTPUT status. Final result of job: "

bright zos-jobs view job-status-by-jobid "$jobid"




# get a list of all of the spool files for our job now that it's in output

spool_ids=$(bright zos-jobs list spool-files-by-jobid "$jobid" --rff id --rft table)
```

```
mkdir -p jobs/${jobid}
FILE_PREFIX=./jobs/${jobid}/GTXMSKL



# save each spool ID to a custom file name
while read -r id; do
    bright zos-jobs view spool-file-by-id "$jobid" ${id} > ${FILE_PREFIX}_spool_$id.txt
    echo "Saved spool DD to ${FILE_PREFIX}_spool_$id.txt"
done <<< "$spool_ids"



echo "Downloading report $TDM_HLQ.TEMP.REPT -> ${FILE_PREFIX}_report.txt"
bright zos-files download ds "$TDM_HLQ.TEMP.REPT" -f "${FILE_PREFIX}_report.txt"
if  grep -q 'RC=0000' ${FILE_PREFIX}_spool_2.txt
then
    echo "in-flight masking finished OK, downloading audit file $TDM_HLQ.TEMP.AUDIT ->
 ${FILE_PREFIX}_audit.txt"
    bright zos-files download ds "$TDM_HLQ.TEMP.AUDIT" -f "${FILE_PREFIX}_audit.txt"
else
    echo "Masking failed"
    exit 255
fi
```

```
#clean target db2 table because load must bu done into an empty table ------------

echo "cleaning target table vlcvi01.gt_test"

bright db2 execute sql --query "delete from BROADCOM.gt_test"




#run load back to db2 ---------------

FILE_PREFIX=./jobs/${jobid}/DB2LOAD




#upload jcl

echo "Uploading MASK JCL DB2LOAD.txt -> $TDM_HLQ.JCL(DB2LOAD)"

bright zos-files upload file-to-data-set DB2LOAD.txt "$TDM_HLQ.JCL(DB2LOAD)"




#submit our job

jobid=$(bright zos-jobs submit data-set "$TDM_HLQ.JCL(DB2LOAD)" --rff jobid --rft
 string)




echo "Submitted masking job, JOB ID is $jobid"
```

```
#wait for it to go to output

status="UNKNOWN"

while [[ "$status" != "OUTPUT" ]] ; do

    echo "Checking status of job $jobid"

    status=$(bright zos-jobs view job-status-by-jobid "$jobid" --rff status --rft
 string)

    echo "Current status is $status"

    sleep 5s

done;




echo "Job completed in OUTPUT status. Final result of job: "

bright zos-jobs view job-status-by-jobid "$jobid"




# get a list of all of the spool files for our job now that it's in output

spool_ids=$(bright zos-jobs list spool-files-by-jobid "$jobid" --rff id --rft table)




# save each spool ID to a custom file name

while read -r id; do

    bright zos-jobs view spool-file-by-id "$jobid" ${id} > ${FILE_PREFIX}_spool_$id.txt

    echo "Saved spool DD to ${FILE_PREFIX}_spool_${id}.txt"

done <<< "$spool_ids"
```

```
#show content of table ------------

echo "content of table source table otherDB_tdm514.gt_test"

bright db2 execute sql --query "select * from otherDB_tdm514.gt_test where gt_no=1"

echo "content of table source table BROADCOM.gt_test"

bright db2 execute sql --query "select * from BROADCOM.gt_test where gt_no=1"
```

Parameters

mask.sh takes the following parameter:

- **TDM_HLQ**
  Defines the connection to the DB2 database you want to mask.

Brightside commands

mask.sh includes the following CA Brightside commands:

- Upload the file `GTXMSKL.txt` (in the directory from which you execute the command) to the Mainframe dataset `$TDM_HLQ.JCL(GTXMSKL)` :

  ```
  bright zos-files upload file-to-data-set GTXMSKL.txt "$TDM_HLQ.JCL(GTXMSKL)"
  ```

- Upload the file `map.csv` (in the directory from which you execute the command) to the Mainframe dataset `$TDM_HLQ.MAPCSV(MAPCSV)` :

  ```
  bright zos-files  upload file-to-data-set map.csv "$TDM_HLQ.MAPCSV(MAPCSV)"
  ```

- Submit the dataset `$TDM_HLQ.JCL(GTXMSKL)` for execution :

  ```
  jobid=$(bright zos-jobs submit data-set "$TDM_HLQ.JCL(GTXMSKL)" --rff jobid --rft
   string)
  ```

- Return the status of the job you submit, as a string :

  ```
  status=$(bright zos-jobs view job-status-by-jobid "$jobid" --rff status --rft string)
  ```

- Return a table of all spool files for the job :

  ```
  spool_ids=$(bright zos-jobs list spool-files-by-jobid "$jobid" --rff id --rft table)
  ```

- Save a spool file to a local file (with the name **./${jobid}_spool_$id.txt**) :

```
bright zos-jobs view spool-file-by-id "$jobid" ${id} > ./${jobid}_spool_$id.txt
```

- Download report in dataset **$TDM_HLQ.TEMP.REPT** to a local file (with the name **${jobid}_report.txt**) :

```
bright zos-files download ds "$TDM_HLQ.TEMP.REPT" -f "${jobid}_report.txt"
```

- Download audit in dataset **$TDM_HLQ.TEMP.AUDIT** to a local file (with the name **${jobid}_audit.txt**) :

```
bright zos-files download ds "$TDM_HLQ.TEMP.AUDIT" -f "${jobid}_audit.txt"
```

- Delete all columns from target table :

```
bright db2  execute sql --query "delete from BROADCOM.gt_test"
```

- Upload the file `DB2LOAD.txt` (in the directory from which you execute the command) to the Mainframe dataset `$TDM_HLQ.JCL(DB2LOAD)` :

```
bright zos-files upload file-to-data-set DB2LOAD.txt "$TDM_HLQ.JCL(DB2LOAD)"
```

- Submit the dataset `$TDM_HLQ.JCL(DB2LOAD)` for execution :

```
jobid=$(bright zos-jobs submit data-set "$TDM_HLQ.JCL(DB2LOAD)" --rff jobid --rft
 string)
```

- Execute the SQL query "`select * from BROADCOM.gt_test where gt_no=1` " on the DB2 database to which you connect CA Brightside.

```
bright db2 execute sql --query "select * from BROADCOM.gt_test where gt_no=1"
```

In-place Mainframe masking with CA Brightside

You can mask data on the Mainframe in-place, with CA Brightside. This page describes a typical in-place masking scenario and explains how the **mask.sh** script uses CA Brightside to perform this task.

Requirements

To mask Mainframe data in-place, you need the following:

- CA Test Data Manager Mainframe DB2 Add On MVS 5.4.14.
- CA Brightside (Community or Enterprise Edition).
- Valid Mainframe authorization.
- Transformation map.
- GTXMSK program, customized for your masking job.
- Datasets on the Mainframe under your HLQ, to receive the files you upload.

Execution

GTXMSK program

To execute an in-place masking job on Mainframe, you need to upload:

- Transformation map (**.csv** file).
- **GTXMSK.txt** JCL.

The GTXMSK program includes the following hardcoded parameters:

- JOBCARD
- TDM loadlib, jcllib
- TEMP HLQ for reports and audit (REPHLQ)
- DSN for transformation map (MAPDS)
- DB2 steplib (STEP05.STEPLIB)

Mask.sh script

To execute a masking job on Mainframe, you can modify and execute the provided **mask.sh** bash script. This script performs the following actions:

1. Uploads the following files to datasets on the Mainframe:
   - Transformation map (.csv).
   - GTXMSK.txt JCL.
2. Submits GTXMSK JCL to Mainframe to start masking job.
3. Generates local spool files and report, from files on Mainframe.

mask.sh script

mask.sh

```
#! /bin/env bash




TDM_HLQ=BROADCOM.TDM
```

```
#upload jcl

echo "Uploading MASK JCL GTXMSK.txt -> $TDM_HLQ.JCL(GTXMSK)"

bright zos-files upload file-to-data-set GTXMSK.txt "$TDM_HLQ.JCL(GTXMSK)"




#upload map file

echo "Uploading maping file map.csv -> $TDM_HLQ.MAPCSV(MAPCSV)"

bright zos-files upload file-to-data-set map.csv "$TDM_HLQ.MAPCSV(MAPCSV)"




#submit our job

jobid=$(bright zos-jobs submit data-set "$TDM_HLQ.JCL(GTXMSK)" --rff jobid --rft string)




echo "Submitted masking job, JOB ID is $jobid"




#wait for it to go to output

status="UNKNOWN"

while [[ "$status" != "OUTPUT" ]] ; do

    echo "Checking status of job $jobid"

    status=$(bright zos-jobs view job-status-by-jobid "$jobid" --rff status --rft
 string)

    echo "Current status is $status"
```

```
    sleep 5s
done;



echo "Job completed in OUTPUT status. Final result of job: "
bright zos-jobs view job-status-by-jobid "$jobid"



# get a list of all of the spool files for our job now that it's in output
spool_ids=$(bright zos-jobs list spool-files-by-jobid "$jobid" --rff id --rft table)



# save each spool ID to a custom file name
while read -r id; do
    bright zos-jobs view spool-file-by-id "$jobid" ${id} > ./${jobid}_spool_$id.txt
    echo "Saved spool DD to ./${jobid}_spool_${id}.txt"
done <<< "$spool_ids"



echo "Downloading report $TDM_HLQ.TEMP.REPT -> ${jobid}_report.txt"
bright zos-files download ds "$TDM_HLQ.TEMP.REPT" -f "${jobid}_report.txt"
if  grep -q 'RC=0000' ./${jobid}_spool_2.txt
then
```

```
    echo "in-place masking finished OK, downloading audit file $TDM_HLQ.TEMP.AUDIT ->
 ${jobid}_audit.txt"

    bright zos-files download ds "$TDM_HLQ.TEMP.AUDIT" -f "${jobid}_audit.txt"

fi
```

Parameters

mask.sh takes the following parameter:

- **TDM_HLQ**
  The HLQ where you want to perform your masking job.

Brightside commands

mask.sh includes the following CA Brightside commands:

- Upload the file `GTXMSK.txt` (in the directory from which you execute the command) to the Mainframe dataset `$TDM_HLQ.JCL(GTXMSK)` :

  ```
  bright zos-files upload file-to-data-set GTXMSK.txt "$TDM_HLQ.JCL(GTXMSK)"
  ```

- Uploads the file `map.csv` (in the directory from which you execute the command) to the Mainframe dataset `$TDM_HLQ.MAPCSV(MAPCSV)` :

  ```
  bright zos-files upload file-to-data-set map.csv "$TDM_HLQ.MAPCSV(MAPCSV)"
  ```

- Submit the dataset `$TDM_HLQ.JCL(GTXMSK)` for execution :

  ```
  jobid=$(bright zos-jobs submit data-set "$TDM_HLQ.JCL(GTXMSK)" --rff jobid --rft
   string)
  ```

- Return the status of the job you submit, as a string :

  ```
  status=$(bright zos-jobs view job-status-by-jobid "$jobid" --rff status --rft string)
  ```

- Get a list of all spool files for the job :

  ```
  spool_ids=$(bright zos-jobs list spool-files-by-jobid "$jobid" --rff id --rft table)
  ```

- Save a spool file (from **spool_ids**) to a local file (with the name **./${jobid}_spool_$id.txt**) :

  ```
  bright zos-jobs view spool-file-by-id "$jobid" ${id} > ./${jobid}_spool_$id.txt
  ```

- Download dataset **$TDM_HLQ.TEMP.REPT** to a local file (with the name **${jobid}_report.txt**) :

  ```
  bright zos-files download ds "$TDM_HLQ.TEMP.REPT" -f "${jobid}_report.txt"
  ```

- Downloads audit in dataset **$TDM_HLQ.TEMP.AUDIT** to a local file (with the name **${jobid}_audit.txt**) :

```
bright zos-files download ds "$TDM_HLQ.TEMP.AUDIT" -f "${jobid}_audit.txt"
```

Reference

This section contains reference information such as functions, messages, actions, and more.

Data Generation Functions and Parameters

This page contains a comprehensive list of the Datamaker functions and their parameters.

Use double quotes to pass literal commas and spaces in arguments. Arguments to functions can be double quoted, but the behavior between Datamaker and CA TDM Portal varies. CA TDM Portal always strips double quotes, but Datamaker may or may not. Differences are noted in the function list. In Datamaker, @seedlist, @date, @ibann, @jdate fail if supplied with a double quoted string. Single quotes are always returned.

> **NOTE**
>
> Functions, variables, and column references embedded in a quoted string are still evaluated. Although unintuitive, this behavior is particular to Datamaker and is replicated in CA TDM Portal.

Click to expand table of contents...

Boolean Expressions

A Boolean expression can be a constant representing `true` or `false`.

A `true` value is one of the following:

- the words `true` or `yes`
- a number which is not 0 or -1
- something else which is none of the false values below.

A `false` value is one of the following:

- the numbers 0 or -1
- the words `false`, `no`, or `null`
- an empty string
- a string whose first character is N, n, F, f, I, I, ! or ?.

A Boolean function is one of @and, @not and @or. For more information, see the functions on this page.

A Boolean expression consists of the following:

```
<left hand operand> <logical operator> [<right hand operand>]
```

An operand can be any string or numeric expression made of constants, functions, variables or column references. All these operators work with both numbers and strings. If the operands are strings, then a lexicographic comparison is made, for example, A is less than Z. If the operands are floating point numbers, then tests for equality are done within a fixed precision of 1.0e-10, because floating point numbers do not have a guaranteed accuracy.

A logical operator can be one of:

- = equality test
- <> inequality test
- IS NULL tests if the left hand operand is NULL or empty. This operator does not require a right hand operand.
- > greater than
- < less than
- >= greater or equal to
- <= less than or equal to

In Datamaker, do not enclose just one operand in quotes; for example, the expression `"a"=a` returns false.

A Boolean expression is only evaluated when used in a function that requires a Boolean expression. They are not evaluated when used as an argument to a macro. For example:

- In `@if(a=b,yes,nno)@` , the expression "`a=b` " will be evaluated to true or false.
- In `@randchars(4,4,a=b)@` , the expression "`a=b` " is read literally as the characters "a", "=" and "b".

11PROOF(SEQUENCE, SIGN)

Finds the next Elf-Proef number from the sequence, according to the Dutch bank account number validation method.

Parameters:

- SEQUENCE — name of the sequence to use. The sequence is created if it does not already exist.
- SIGN — the arithmetic operator + or -

Return value: the Elf-Proef number

Example: @11proof(mysequence,+)@

Example result: 100000010

ABS(NUMBER)

Returns the absolute value of the argument.

Parameters:

- NUMBER — an integer or floating point number.

Return value: An absolute value

Example: @abs(-23)@

Example result: 23

ADD(NUMBER1,NUMBER2)

Adds two values together.

Parameters:

- Two expressions evaluating to integer or floating point numbers.

Return value: The sum of the two numbers.

Example: @add(1,1)@

Example result: 2

Example: @add(~v1~,~v2~)@

Example result: The value of variable v1 plus v2.

ADDCHECKSUM(NUMBER, METHOD)

Adds special LUHN, VERHOEFF, or CB checksum.

Parameters:

- NUMBER — a number
- METHOD — name of the checksum method to be used; one of:
  - LUHN
  - VERHOEFF
  - CB

Return value: The input number with added checksum.

Example: @addchecksum(343, luhn)@

Example result: 3434

ADDDAYS(DATE, DAYS)

Adds a number of days to a date, or subtracts them, if it is a negative number.

Parameters:

- DATE — A date in a date format specified through `@string()@`.
- DAYS — the number of days to add. Can be a positive or negative number.

Return value: The new date in project date format. You set the project-wide Date Format in the project settings. By default, YMD.

Example: @adddays(@string(06/05/12,DD/MM/YY)@,2)@

Example result: 2017-05-08

ADDLUHN(NUMBER)

Adds a Luhn checkdigit. The Luhn algorithm or Luhn formula, also known as the "modulus 10" or "mod 10" algorithm, is a simple checksum formula used to validate a variety of identification numbers, such as credit card numbers, IMEI numbers, National Provider Identifier numbers in US, and Canadian Social Insurance Numbers.

Parameters:

- NUMBER — a number of any length

Return value: number + Luhn checkdigit

Example: `@addluhn(123)@`

Example result: 1230

Examples:

```
@addluhn(@randlov(0, @list(34,37)@)@@nextval(AMEX_SEQ, 100000000000)@)@          ;AMEX
 Card
@addluhn(@randlov(0, @list(51,52,53,54,55)@)@@nextval(MC_SEQ, 1000000000000)@)@
  ;MASTER Card
@addluhn(4@nextval(VS13_SEQ,10000000000)@)@                                      ;VISA
 13 digit
@addluhn(4@nextval(VS16_SEQ,10000000000000)@)@                                   ;VISA
 16 digit
```

ADDMICROSECS(TIMESTAMP, MICROSECONDS)

Adds a number of microseconds to a timestamp. The timestamp resolution is 1 millisecond, so adding 1 microsecond will not change the timestamp, but adding 1000 microseconds will.

Parameters:

- TIMESTAMP — a timestamp,
- MICROSECONDS — the number of microseconds to add.

Return value: new timestamp in project timestamp format

Example: @addmicrosecs(~STIMESTAMP~,1000)@

ADDMILLISECS(TIMESTAMP, MILLISECONDS)

Adds a number of milliseconds to a timestamp. The function will use the project format to interpret the timestamp first and then a number of other formats until it finds one that matches. If the timestamp format cannot be recognized, the function fails. For example "08/08/2016 03:36:20.000".

Parameters:

- TIMESTAMP — a timestamp,
- MILLISECONDS — a number of milliseconds to add.

Return value: The new timestamp in project timestamp format

Example: @addmillisecs(~STIMESTAMP~,136)@

Example result: 08/08/2014 03:36:20.136

ADDMOD97(NUMBER)

Adds a modulo97 checksum code to a number.

Parameters:

- NUMBER — A long integer number

Return value: The input value followed by the two checksum digits.

Example: @addmod97(100)@

Example result: 10003

ADDMONTHS(DATE, MONTHS)

Adds a number of months to a date. The function interprets dates in the project format first, followed by a number of other formats, until a match is found.

Parameters:

- DATE — a date
- MONTHS — the number of months to add. Can be negative

Return value: a new date in project format.

Example: @addmonths(05/05/2020,3)@

Example result: 05/08/2020

ADDRAND(NUMBER,MIN,MAX)

Addsa random number between a specified minimum and maximum to a number. If MAX is less than MIN, the value returned is NUMBER plus MIN minus 1.

Parameters:

- NUMBER — a long integer number
- MIN — minimum value for a random number
- MAX — maximum value for a random number

Return value: a new number

Example: @addrand(1,1,5)@

Example result: 2,3,4,5,6

ADDRANDDAYS(DATE,MIN,MAX)

Adds a random number of days between the min and max to the date. If MAX is less than MIN, the value returned is the DATE plus the MIN minus 1.

Parameters:

- DATE — a date in project date format
- MIN — minimum number of days to add
- MAX — maximum number of days to add

Return value: a new date in project format.

Example: @addranddays(@date(13/04/1977,DD/MM/YYYY)@,3,10)@

Example result: 22/04/1977

ADDSECONDS(DATETIME, SECONDS)

Adds a number of seconds to a datetime value.

Parameters:

- DATETIME — a date
- SECONDS — number of seconds to add

Return value: a new datetime

Example: @addseconds(~SDATE~,1)@

Example result: 2008/07/04 00:00:01

ADDSECONDS(TIME, SECONDS)

Adds a number of seconds to a time.

Parameters:

- TIME — a time in project format, usually hh:mm:ss
- SECONDS — the number of seconds to add.

Return value: the new time in project format

Example: @addseconds(01:02:00,1)@

Example result: 01:02:01

ADDVERHOEFF(NUMBER)

Adds a checksum using the Verhoeff algorithm.

Parameters:

- NUMBER — a number

Return value: a new number followed by a checksum digit.

Example: @addverhoeff(1234)@

Example result: 12340

ADDYEARS(DATE, YEARS)

Adds a number of years to a date. IF the DATE does not match the project format, the functions attempts to interpret the DATE in other formats.

Parameters:

- DATE — a date in project format
- YEARS — the number of years to add

Return value: a new date in project format.

Example: @addyears(2008/07/01, 1)@

Example result: 2009/07/01

ALPHANUM(STRING)

Remove all non-alphanumeric characters from a string.

Parameters:

- STRING — a character string to be cleaned up

Return value: a new string containing only letters and digits

Example: @alphanum(12345%%abc)@

Example result: 12345abc

ASC(STRING)

Returns the ASCII character code of the first character of the string. This is actually a Unicode code point which will be mostly the same as ASCII for European language strings.
If the string is double quoted, CA Datamaker returns the code of the double quote. CA TDM Portal returns the code for the first character after the double quote.

Parameters:

- STRING — a character string

Return value: a new string. If the input was empty, it returns an empty result.

Example: @asc(Apple)@, **Result**: 65

ASLIST(LIST[,COLUMN1][,COLUMN2][,QUOTED])

Generates a list of comma separated strings from the specified list function.

Usage in TDOD: Specify the two optional columns to define a drop-down variable for use in TDOD screens. Such variables have two attributes: a key value (the value of the variable) and a display value (the value shown in the drop down list in the TDOD UI). COLUMN1 defines the key value and COLUMN2 defines the display value. The list items are returned in the format:
`<COLUMN1> [ <COLUMN1> ]`
TDOD recognizes the item in square brackets as the display value.

Parameters:

- LIST — A list function such as @SEEDLIST or @SQLLIST that generates a list of items.
- COLUMN1 — The name or number of a column 1 in the list. If omitted, column 1 of the LIST is chosen. Used to define a drop-down variable for TDOD screens.
- COLUMN2 — The name of another column in the list. Used to define a drop-down variable for TDOD screens.
- QUOTED — Set this to sq to return the list items single quoted. Set this to dq to return the list items double quoted. This optional parameter is useful when generating lists for use in SQL expressions.

Example: @aslist(@seedlist("DayOfWeek")@,dq)@

Example result: "Monday","Tuesday","Wednesday"…

ASLIST(DIRLIST(DIRECTORY[,EXTENSION[,NAMEFILTER]]))

Generates a list of comma separated file names from the given directory. This function is not supported in TDM Portal.

Parameters:

- DIRECTORY — A path to a directory whose files you want to list.
- EXTENSION — The file extension of the files that you want to list. If ommited, files with any extension are listed.
- NAMEFILTER — A (sub)string of the names of the files that you want to list. If ommited, files with any names are listed. You need to specify an extension.

Example: @aslist(@dirlist(c:\temp)@)@

Example result: Lists any file in the c:\temp folder

Example: @aslist(@dirlist(c:\temp,txt)@)@

Example result: Lists any file in the c:\temp folder with the extension ".txt".

Example: @aslist(@dirlist(c:\temp,txt,proc)@)@

Example result: Lists any file in the c:\temp folder with the extension ".txt" whose name contains "proc".

AND (BOOLEAN, BOOLEAN[,BOOLEAN...])

Performs a logical AND function on a variable list of arguments.

Parameters:

- a list of Boolean valued expressions. For more information, see the Boolean Expressions appendix.

Return value: a Boolean value, either true or false

Example: @AND (A, B)@ where A=true and B=false@

Example result: false

ATSIGN()

Returns the value '@'. This function is useful where you need a literal at-sign, but in a context where you want to avoid it being interpreted as a function invocation.

Example: A@atsign()@B

Example result: A@B

BOOLEANCOMPARE(LEFT, RIGHT, OPERATOR)

Compares the left and right values using an operator. If left and right values are numbers, a number comparison is used with the operator and if left or right values are a string, a string comparison is used.

Parameters:

- LEFT — a string or a number value
- RIGHT — a string or a number value
- OPERATOR— used to perform comparison between the right and left values. An operator can be one of:
  - > greater than
  - < less than
  - >= greater than or equal to
  - <= less than or equal to
  - = equality
  - <> inequality

Example: @booleancompare(263, 318, <)@

Example result: true

CARET()

Returns the value '^ '. This function is useful where you need a literal caret character, but in a context where you want to avoid it being interpreted as a column reference.

Example: A@caret()@B

Example result: A^B

CASE(TEST1,VALUE1,[TESTn,VALUEn...], ELSEVALUE)

Returns the value associated with the first test that is true, otherwise return the ELSEVALUE.

Parameters:

- TEST1 — a boolean expression. For more information, see the Boolean Expressions appendix.
- VALUE1 — the expression whose value you want returned if TEST1 is true
- TESTn, VALUEn — (Optional) Additional tests and values
- ELSEVALUE — an expression whose value is returned if all tests fail.

Return value: the value of the expression whose test was true.

Example: @case(1=2,A,2=2,B,C)@

Example result: B

CHAR(CODENUMBER)

Returns the character for a specified code number.

Parameters:

- CODENUMBER — A Unicode code point. For English, this is the same as ASCII.

Return value: a single character

Example: @char(65)@

Example result: A

COLLAPSE(STRING)

Removes all space, tab, carriage return, and new line characters from a string.

Parameters:

- STRING — a character string to be cleaned up.

Return value: a new string

Example: @collapse(hello there)@

Example result: hellothere

CONVBASE(NUMBER, BASE)

Converts the number to a different base.

Parameters:

- NUMBER — a decimal number
- BASE — the new base to convert to.

Return value: a new number in the requested base.

Example: @convbase(999,16)@

Example result: 3E7

CONVBASE(NUMBER,BASE,DIGITS)

Converts the string to a different base, represented by specified digits.
The number of digits is equal to the base. For base 8, provide 8 digits. If the number of digits is greater than the base, then only the first digits are taken. If the number of digits is less than the base, then the digits are cycled until the required number is reached. For example, if you specify "AB" for base 5, then the digits are expanded to "ABABA".

Parameters:

- NUMBER — a decimal number to be converted
- BASE — the new base as a decimal number
- DIGITS — a list of digits to represent the converted number in. If you do not define any DIGITS, then the function attempts to guess the digits from the base specified. For example, if the base is 8, then it chooses 1,2,3,4,5,6,7,0. The least significant digit must be last.

Return value: the converted number as string

Example: @convbase(6,3,*&^)@

Example result: &*

COUNT(VALUE[,VALUE...])

Counts the number of items in the list or related rows in another table.

Parameters:

- VALUE — a list of items

Return value: **the** number of items in the list

Example: @count(1,A,Z,G,X)@ @

Example result: 5

Example: @count(^ITEM.ITEMTOT^)@

Example result: 4 (the number of rows in the column "ITEMTOT".

COUNTLIST(@SQLLIST(CONNECTION, SQL)@)

Returns the number of rows that are returned by specified SQL query.

Parameters: A SQL query of the following type @SQLLIST(CONNECTION, SQL)@

- CONNECTION — Defines the dbms connection type. Choose one of the following:
  - R — Repository
  - S — Source
  - T — Target
  - P*profilename* — connection profile, for example `Pmyconprof` .
- SQL — A SQL query. Only select statements are supported.

Return value: number of rows returned by the query

Example: @COUNTLIST(@SQLLIST(Ptravel, SELECT * FROM TRAVEL.COUNTRIES)@)@

Example result: 293

COUNTLIST(@SEEDLIST(SEEDNAME)@)

Returns the number of rows in a seed list. See also Seed Lists.

Parameters:

- SEEDNAME — name of a seed list. In Datamaker, do not enclose this string in quotes.

Return value: number of rows in the list

COUNTLIST(@PRIORPUBLISHKEYLIST(LD_ID, TABLENAME)@)

Returns the number of rows contained in the key list of a saved publish job. A publish job can use the saved columns feature to save the data in various columns for use in another publish job to be carried out at a later time. The priorpublishkeylist function retrieves that data. Use the countlist function to determine how many rows were saved. The data is saved in the repository. If you run the job several times, the function looks for the latest data.

Parameters:

- LD_ID — The id of the generator that performed the publish. Datapainter inserts this id automatically when composing the expression containing countlist.
- TABLENAME — the name of the table whose columns were saved.

Return value: number of rows saved from the named table

Example: @COUNTLIST(@PRIORPUBLISHKEYLIST(1007, PEOPLE)@)@

COUNTLIST(@ALLPAIRS(LIST, LIST,[LIST], ALL_COMBINATIONS)@)

(Datamaker only) Returns the number of rows that are returned by all combinations of all pairs in specified LISTs. This function is not supported in TDM Portal 4.0.

Parameters: an SQL query of the following type @ALLPAIRS(LIST, LIST[,LIST], ALL_COMBINATIONS)@) where

- LIST — list
- ALL_COMBINATIONS — constant

Return value: number of rows

COUNTWADL(URL, XPATH)

(Datamaker only) Returns the number of rows that are returned by the REST call that match an XPATH. This function is not supported in TDM Portal.

Parameters:

- URL — specifies the REST call to be made. Only GET is supported.
- XPATH — xpath of the XML element to find

Return value: number of rows

Example: @countwadl(http://server:5091/Service?LIST, Customer/CustomerId)@

CUSTOMLUHN(NUMBER,INDEX,LENGTH,[SEEDVALUE])

Returns a Luhn number based on a custom set of digits. For more information, see also the ADDLUHN function.

Parameters:

- NUMBER — A custom set of digits that stays the same in each generated number,
- INDEX — the position in the number where the additional digits are inserted to generate a Luhn number,
- LENGTH — the target length of the Luhn number,
- SEEDVALUE — an optional seed value to control uniqueness.

Return value: a LUHN number

Example 1: You want to create a Luhn number that starts with 12345 and ends with 56789. You also want the Luhn number to be 16 digits long. The length of prefix and postfix is 5+5=10 digits. You want to generate the remaining 6 digits. You use the function with the following parameters: `@customluhn(1234556789, 5, 6)@`. The function returns "`12345`13626`56789`".

Example 2: You want to generate a unique list of Luhn values. You define the seedvalue parameter to use the function nextval starting from 1000:

`@CUSTOMLUHN(1234556789,5,6,@(nextval(listA, 1000))@)@`. Each time the customluhn function is called, the seedvalue is incremented by 1. The function returns the following:

```
1234501000156789 << seedvalue =1000
1234501001656789 << seedvalue =1001
1234501002056789 << seedvalue =1002
1234501003656789 << seedvalue =1003
1234501004456789 << seedvalue =1004
```

DATE(STRING)

Interprets the input string as a date and converts it to the format defined for the project.

Parameters:

- STRING — A date string. In Datamaker, do not enclose this string in quotes.

Return value: The input date in the project format.

Example (where project format is YYYY-MM-DD): @date(26/february/2018)@

Example result: 2018-02-26

DATE(DATESTRING, FORMAT)

Interprets a string as a date using the format specified and converts it to the format defined for the project..

Parameters:

- DATESTRING — A date string in any format,
- FORMAT — The DATESTRING's date format, for example, dd/mm/yyyy, mm/dd/yyyy, DD-MM-YY. For more information see, date format.

Return value: The input date in the project format.

Example: @date(26/02/2018, DD/MM/YYYY)@ and the project format is YYYY-MM-DD.

Example result: 2018-02-26

DATETIME(DATESTRING)

Converts a date to the datetime format defined for the project, eg yyyy-mm-dd HH:MM:SS.

Parameters:

- DATESTRING — A date string in any format. In Datamaker, do not enclose this string in quotes.

Return value: the input date reformatted as a datetime.

Example: @datetime(26/february/2008)@

Example result: 2008-02-26 00:00:00

DATETIME(STRING, FORMAT)

Returns a specific string as a date and a time, using the format that is specified.

Parameters:

- STRING — a date and time in any format,
- FORMAT — the new date time format.

Return value: date and time string with specified format.

Example: @datetime(12/07/1993, MMDDYYYY)@

Example result: 07121993

DAYSAFTER(STARTDATE, ENDDATE)

Returns the difference in days between the end date and the start date.

Parameters:

- STARTDATE — starting date,
- ENDDATE — ending date

Return value: The number of days between start and end.

Example: @daysafter(@date(26/10/2009,DD/MM/YYYY)@,@date(31/12/2016,DD/MM/YYYY)@)@

Example result: 3

DBLQUOTE()

Returns a double quote character. Use this function in expressions where a quote could be ambiguous.

Return value: a double quote character

Example: @dblquote()@

Example result: "

DIVIDE(NUMBER1,NUMBER2)

Divides the first number by the second. The function expects integer or floating point numbers.

Parameters:

- NUMBER1 — dividend,
- NUMBER2 — divisor.

Return value: the quotient

Example: @divide(6,3)@

Example result: 2

DOB(MINAGE,MAXAGE,DATE)

Determines a date of birth, given a random age between a minimum and a maximum, on a specified date.

Parameters:

- MINAGE — minimum age,
- MAXAGE — maximum age,
- DATE — the date at which someone has a randomly selected age in the specified range. In Datamaker, do not enclose this string in quotes.

Return value: a date of birth in project date format

Example: What is your birthday, if you were a random age between 4 and 99 years old on the 8th of December 1901? @dob(4,99,1901-12-08)@

Example result: 1861-09-18

DOW(DATE)

Returns the day of the week for the date specified.

Parameters:

- DATE — a date

Return value: a number representing a day of the week. 1 for Sunday, 2 for Monday, 3 for Tuesday, 4 for Wednesday, 5 for Thursday, 6 for Friday, 7 for Saturday

Example: @dow(@date(14/09/1972,DD/MM/YYYY)@)@

Example result: 5

EBCDIC(STRING)

(Datamaker only) Returns the EBCDIC representation of the given string. Useful for generating flat file data for mainframe systems. Not supported on TDM Portal.

Parameters:

- STRING — a string to be converted to ebcdic

Return value: string representation

Example: @ebcdic(Datamaker)@

ELEMENT(STRING, DELIM, ELEMENTNO)

Returns a specified item in a list of items.

Parameters:

- STRING — a string containing a list of items separated by a delimiter character,
- DELIM — the character separating items in the list,
- ELEMENTNO — number of the item to return, starting at 1.

Return value: the requested list item. If ELEMENTNO is out of the range of the list, then the function returns "".

Example: @element("hello,there,folks", "," , 3)@

Example result: folks

ENVVAR(ENV)

(Datamaker only) Reads an environment variable. This function is not supported in TDM Portal 4.0.

Parameters:

- ENV — The name of an environment variable

Return value: The value of this environment variable

Example: @envvar(WINDIR)@

Example result: C:\WINDOWS

EXECSQL(CONNECTION, SQL)

Executes a SQL select query against the repository, a source, a target, or a profile. The SQL can be any SQL compatible with the connection and can include references to variables and functions. The query must return a single row. If more than one column is queried, then the values are returned as a comma separated list.

Parameters:

- CONNECTION — Defines the dbms connection type. Choose one of the following:
  - R — Repository
  - S — Source
  - T — Target
  - P*profilename* — connection profile, for example `Pmyconprof` .
- SQL — A SQL select query that is compatible with the dbms underlying the specified connection.

Return value: the result of the query

Example: @execsql(PTravel_AW - Personal DB, select distinct expiration_date from credit_cards)@

Result: 1999-12-31

EXECSQLCOUNT(CONNECTION, SQL)

Returns the number of records returned by the specified SQL query, using the specified connection.

Parameters:

- CONNECTION — connection type - R(Repository), S(Source), or T(Target), or P*profilename* a connection profile.
- SQL — a SQL select query that is compatible with the dbms underlying the specified connection.

Return value: the number of rows returned by the query

Example: @execsqlcount(PTravel_AW - Personal DB, select * from countries)@

Example result: 293

EXECSQLPROC(CONNECTION,PROCNAME,

PARAMNAME1,PARAMDIRECTION1,PARAMVALUE1,

[PARAMNAME2,PARAMDIRECTION2,PARAMVALUE2,...],PARAMOUTNAME)

Execute a stored procedure against a specified connection with a list of parameters, and return the result.

Parameters:

- CONNECTION — specify a Connection Profile. Use prefix 'P'.
  Syntax: "P*profilename*" connects to Connection Profile 'profilename'.
- PROCNAME — Name of the stored procedure to be executed.
- PARAMNAME — Name of the first parameter.
- PARAMDIRECTION — Each parameter can be either an input parameter, or an output parameter, or both.
  - (SQL Server) IN, OUT, OUTPUT
  - (Oracle) IN, IN OUT, OUT
  - (DB2 and Teradata) IN, OUT, INOUT
- PARAMVALUE — The value of the parameter named in PARAMNAME.
- (Optional) You can have any number of parameters by repeating PARAMNAME, PARAMDIRECTION and PARAMVALUE.
- PARAMOUTNAME — Name of the output parameter name to be returned as the value of the EXECSQLPROC function.

Return Value: Value that is returned by the stored procedure as defined by PARAMOUTNAME.

Example: @execsqlproc(PTravelX,dbo.extractCounter,paramin1,in,'Test',paramin2,in,100,paramout1,output,0,paramout2,output,0
This executes the stored procedure "dbo.extractCounter" against the connection "TravelX". The procedure is supplied with the input parameters paramin1='Test', paramin2=100 and the output parameter paramout2. The value of this parameter is returned as the value of the execsqlproc function call.

Example result: 200

EXP(POWER)

Returns the natural exponent of a specified number.

Parameters:

- POWER — an integer or a floating point number

Return value: natural exponent of the input

Example: @exp(1)@

Example result: 2.718281828459045, because e to the power of 1 is e.

EXP(NUMBER, POWER)

Raises the specified number to a power.

Parameters:

- NUMBER — an integer or floating point number,
- POWER — an integer or floating point number.

Return value: the specified number raised to the power

Example: @exp(10,2)@

Example result: 100

FINNISHPERSONALID(GENDER)

Returns a random Finnish 11-digit ID based on gender.

Parameters:

- GENDER — Either F for female, M for male, or U for Unisex.

Return value: A random Finnish ID of the format "DDMMYYCZZZQ"

Example: @finnishpersonalid(M)@

Example result: 220754-905Y

FINNISHPERSONALID(DATE,NUMBER)

Returns a fixed Finnish 11-digit ID based on DOB and ZZZ number.

Parameters:

- DATE — date of birth in yyyy-mm-dd format
- NUMBER — a personal identification number from 0 to 999.

Return value: A fixed Finnish ID of the format "DDMMYYCZZZQ"

Example: @finnishpersonalid(3,67)@

Example result: 010100-067L

FINNISHPERSONALID(GENDER,STARTDATE,ENDDATE)

Returns a sequential list of Finnish 11-digit IDs, starting from the start date up to the end date. For any given date, 500 Finnish IDs are generated for either M or F gender, and a 1000 IDs in the case of U.

Parameters:

- GENDER — F female, M male, or U Unisex.
- START DATE — a start date of birth in yyyy-mm-dd format
- END DATE — an end date of birth in yyyy-mm-dd format. Must be equal to or greater than the start date.

Return value: A sequential list of Finnish IDs

Example: @countlist(@finnishpersonalid(m,1,23)@)@

Example result: Returns a count of all IDs in the range: 500

Example: @seqlov(1,@finnishpersonalid(m,1,23)@)@

Example result: Returns the next ID value: 010100-001F

FORMATENCRYPTNUMBER(NUMBER)

The FORMATENCRYPTNUMBER function takes a number, and encrypts it into another valid number.

Parameters:

- NUMBER

Return value: a number

Example: `@formatencryptnumber(123456)@`

Example result: `191368`

FORMATENCRYPTSTRING(STRING,[CASESENSITIVE])

The FORMATENCRYPTSTRING function takes a string and returns an encrypted string of the same length. The case of letters is preserved. Numbers are encrypted to numbers. Other characters (such as white space and punctuation) remain unchanged.

Parameters:

- STRING
- Valid values for the second parameter are casesensitive and caseinsensitive.
  - CASESENSITIVE — (Default) The function encrypts lower and upper case letters to different letters. This means, aaaAAA is encrypted as agrVIG.
  - CASEINSENSITIVE — The function encrypts lower and upper case letters to the same letter. This means, aaaAAA is encrypted as agrAGR.

Return value: a string

Example: `@formatencryptstring(FormatEncryptIsCo0l)@`

Example result: `AoxdiwMgqfjhiOsGg7d`

Example: `@formatencryptstring(FormatEncryptIsCo0l,CASEINSENSITIVE)@`

Example result: `FuiudmSbnjnplAcGm7l`

GETSQL(PROGNAME)

Returns the specified stored SQL from the repository. This function is useful in conjunction with @sqllist()@, especially for very long queries.

Parameters:

- PROGNAME — name of the sql program to fetch.

Return value: the SQL statements

Example: Fetch the SQL statements stored under the name "getpets". @getsql(getpets)@

Example result: SELECT name, size, type, colour, tail_length FROM dbo.pets

GROUP(ITEMNO, GROUPSIZE)

If you have a number of items grouped into blocks of a certain size, then this function returns the number of the group that a particular item belongs to. This is computed by 1 + (itemno-1)/groupsize.

Parameters:

- ITEMNO — the number of the item
- GROUPSIZE — number of items in each group.

Return value: number of the group the items belongs to.

Example: @group(11,75)@

Example result: 1, because item number 11 belongs to the first group, because that contains the first 75 items.

GROUP(ITEMNO,FROMGROUPSIZE,TOGROUPSIZE,OFFSET)

Transforms an offset from one group to another.

Parameters:

- ITEMNO — the number of the item,
- FROMGROUPSIZE — the size of the group that the items belongs to,
- TOGROUPSIZE — size of the target group that the item is moving to,
- OFFSET — a number added to the new group number.

Return value: the number of the new group

Example: @group(3,5,10,2)@

Example result: 2

GUID(COLLAPSE)

Generates a globally unique identifier (GUID).

Parameters:

- COLLAPSE — optional keyword which, if specified, removes all dashes from the generated guid. If omitted, or if the argument is not the word "collapse", then the guid is returned with dashes.

Return value: a new GUID.

Example: @guid(collapse)@

Example result: EF33B0BEA5BD44BBB2EDD676CFD64D46

Example: @guid()@

Example result: 3F2504E0-4F89-11D3-9A0C-0305E82C3301

HASH(NUMBER, MAXVAL)

Returns a hash value for a number.

Parameters:

- NUMBER — the number to be hashed,
- MAXVAL — maximum hash value allowed.

Return value: the hash value

Example: @hash(123456,345678)@

Example result: 86656

HASH(NUMBER,MAXVAL,SEED)

Creates a hash value for a number.

Parameters:

- NUMBER — the number to be hashed,
- MAXVAL — maximum allowed hash value,
- SEED — an integer value to be used for randomization of the hash.

Return value: hash value

Example: @hash(42,100)@

Example result: 13

HASHLOV(PERCNULL, @SQLLIST(SRCCONNECTION,SQL)@, SOURCECOLUMN, @SQLLIST(SEEDCONNECTION,SQL)@, SEEDCOLUMN, CASEINSENSITIVEHASH)

Consistently returns a hashed selection from a list of values. The list of values from which to derive the output is the column defined as *SOURCECOLUMN*, in the results of the first SQLLIST function. The list of values that make up the hash is the column defined as *SEEDCOLUMN*, in the results of the second SQLLIST function.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- **@SQLLIST(*SRCCONNECTION,SQL*)@** - defines the Connection Profile *SRCCONNECTION* from which to derive the list (or list of lists) of values to be hashed, and the SQL statement *SQL* to generate that list (or list of lists). See SQLLIST for all internal parameters.
- SOURCECOLUMN - the name or index (starting from 0), of the column in the output of *@SQLLIST(SRCCONNECTION,SQL)@*, that you want to use for the source list.
- **@SQLLIST(*SEEDCONNECTION,SQL*)@** - defines the Connection Profile *SEEDCONNECTION* from which to derive the list (or list of lists) of values make up the hash, and the SQL statement *SQL* to generate that list (or list of lists). See SQLLIST for all internal parameters.
- SEEDCOLUMN - the name or index (starting from 0), of the column in the output of *@SQLLIST(SEEDCONNECTION,SQL)@*, that you want to use for the hash.
- CASEINSENSITIVEHASH - defines whether to convert input string to upper case before hash. Allowed values: Y (default) or N.

Return value: A hashed value from the list (or possibly a null).

Example: @hashlov(0,@sqllist(PHR,SELECT * FROM HR.EMP_TEST)@,FIRST_NAME,@sqllist(PScrambleDB,select * from dbo.gtsrc_reference_data where rd_ref_id = 'FIRSTNAME')@,rd_ref_value,Y)@

HASHLOV(PERCNULL, @SQLLIST(SRCCONNECTION,SQL)@, SOURCECOLUMN, @SEEDLIST(SEEDNAME)@, SEEDCOLUMN, CASEINSENSITIVEHASH)

Consistently returns a hashed selection from a list of values. The list of values from which to derive the output is the column defined as *SOURCECOLUMN*, in the results of the SQLLIST function. The list of values that make up the hash is the column defined as *SEEDCOLUMN*, in the results of the SEEDLIST function.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- **@SQLLIST(*SRCCONNECTION,SQL*)@** - defines the Connection Profile *SRCCONNECTION* from which to derive the list (or list of lists) of values to be hashed, and the SQL statement *SQL* to generate that list. See SQLLIST for internal parameters.
- SOURCECOLUMN - the name or index (starting from 0), of the column in the output of *@SQLLIST(SRCCONNECTION,SQL)@*, that you want to use for the source list.
- **@SEEDLIST(*SEEDNAME*)@** - defines the category *SEEDNAME* to extract from **gtrep_reference_data**. The output of this function is a list, or a list of lists. See SEEDLIST for more information.
- SEEDCOLUMN - the name or index (starting from 0), of the column in the output of *@SEEDLIST(SEEDNAME)@*, that you want to use for the hash.
- CASEINSENSITIVEHASH - defines whether to convert input string to upper case before hash. Allowed values: Y (default) or N.

Return value: A hashed value from the list (or possibly a null).

Example: @hashlov(0, @sqllist(PHR,SELECT * FROM HR.EMP_TEST)@, FIRST_NAME, @seedlist(UK Male First Name)@, 0, Y)@

HASHSIGN()

Returns the value '#'. This function is useful where you need a literal hash sign, but in a context where you want to avoid it being interpreted as a variable parameter invocation. For more information, see Create and Manage Variables.

Example: @pos(#string#, @hashsign()@)@

Example result: Finds a hash sign in a string, in a context when the pos() function is used as the value of a parameterized variable.

IBAN()

Creates a random IBAN for a random country.

Return value: a new IBAN

Example: @iban()@

Example result: GL9622756107472084

IBAN(ISOCOUNTRYCODE)

Creates a random IBAN with specified country code.

Parameters:

- ISOCOUNTRYCODE — For more information, see IBAN formats by country (Wikipedia). The following country codes are not supported: AL, BE, BA, EE, FO, FI, IT, MK, NO, PT, RS, SI. In Datamaker, do not enclose this string in quotes.

Return value: a new random IBAN

Example: @iban(GB)@

Example result: GB94TOCG73393021580682

IBAN(ISOCOUNTRYCODE, BANKCODE)

Creates a random IBAN with specified country code and bank codes.

Parameters:

- ISOCOUNTRYCODE — A country code. For more information, see IBAN(ISOCOUNTRYCODE),
- BANKCODE — A bank code.

Return value: a new random IBAN

Example: @iban(GB, BARC)@

Example result: GB08BARC57552380947394

IBAN(ISOCOUNTRYCODE,BANKCODE,ACCOUNT)

Creates a random IBAN with a specified country code, bank code, and account number.

Parameters:

- ISOCOUNTRYCODE — A country code. For more information, see IBAN(ISOCOUNTRYCODE),
- BANKCODE — A bank code, ACCOUNT: An account number.

Return value: IBAN

Example: @iban(GB,BARC,41021810000222)@

Example result: GB26BARC41021810000222

IF(BOOLEANEXPR,TRUEEXPR,FALSEEXPR)

If the boolean expression is true, this function returns the value of one expression, otherwise it returns the other.

Parameters:

- BOOLEANEXPR — an expression returning a Boolean value. For more details, see the section on Boolean Expressions.
- TRUEEXPR — An expression whose value is returned if BOOLEANEXPR is true,
- FALSEEXPR — An expression whose value is returned if BOOLEANEXPR is false.

Return value: Either TRUEEXPR or FALSEEXPR.

Example: @if(1<0,Smaller,Bigger)@

Example result: Bigger

IFNULL(EXPR, NULLEXPR)

Evaluatse an expression and if it is NULL or empty, return the value of another expression. If not, return the value of the original expression.

Parameters:

- EXPR — An expression,
- NULLEXPR — An expression.

Return value: An expression

Example: @ifnull(@randlov(0, H)@, hello)@

Example result: hello - because "@randlov(0, H)@" returned an empty string, so the function returns the value of NULLEXPR.

Example: @ifnull(@execsql(Pxyz,select name from pets where id=123)@, hello)@

Example result: tiggy – because the select statement returned a result which is not NULL, so this result is returned.

JDATE(DATE[, PRECISION])

Converts a date to a Julian date with a certain precision.

Parameters:

- DATE — a date in any of the usual formats. In Datamaker, do not enclose this string in quotes.
- PRECISION — (optional argument) number of decimal places in the result.

Return value: the corresponding Julian date rounded to the defined precision.

Example: @jdate(25-02-2016,1)@

Example result: 2457443.5

Example: @jdate(25-02-2016)@

Example result: 2457443.500000

LASTDAY(DATE)

Returns the last day of the month that the specified date is in.

Parameters:

- DATE — A date in any of the usual formats. In Datamaker, do not enclose this string in quotes.

Return value: the date of the last day of the month in which the date resides.

Example: @lastday(12/04/2054)@

Example result: 2054-04-30, because the last day of April is the 30th.

LEFT(STRING, LENGTH)

Returns the leftmost characters from the string.

Parameters:

- STRING — a string to be split,
- LENGTH — number of characters to return.

Return value: a substring starting at position 1 in the string.

Example: @left(ashok, 2)@

Result: as

LEFTPAD(STRING, CHARTOPAD, LENGTH)

Pads a string up to a specified length by adding characters to the start. If the string is already the length or longer then no characters are added.

Parameters:

- STRING — a string to be padded
- CHARTOPAD — a string to add repeatedly to the left side of the string. Typically a single character, such as space or zero, but you can pad with any length string.
- LENGTH — the total length of the padded string

Return value: a string padded with characters up to the specified length

Example: `@leftpad(ABCDE, 1, 10)@`

Example result: `11111ABCDE`

Example: `@leftpad(ABCDE, , 10)@`

Result: `ABCDE` (Prefixed spaces)

LEFTTRIM(STRING, CHARTOTRIM)

Remove all occurences of a characters from the left side of a string.

Parameters:

- STRING — a string expression whose value is to be trimmed,
- CHARTOTRIM — the character to remove from left of string. Usually a single character is used, but any length string can be used and that string will be trimmed.

Return value: a possibly shorter string

Example: @lefttrim(++++++hello,+)@

Example result: hello

Example: @lefttrim(ABCDEFGHIJKLMNOPQRSTUVWXYZ, ABC)@

Example result: DEFGHIJKLMNOPQRSTUVWXYZ

LENGTH(STRING)

Returns the number of characters in a string.

Parameters:

- STRING — a string expression. In Datamaker, do not enclose this string in quotes, otherwise they count towards the length.

Return value: number of characters in that string

Example: @length(Contrafibularity)@

Result: 16

LOG(NUMBER)

Returns the natural log of a number.

Parameters:

- NUMBER — a numeric expression

Return value: the natural log of the value of the expression

Example: @log(10)@

Example result: 2.302585092994046

LOGTEN(NUMBER)

Returns the log to the base 10 of a number.

Parameters:

- NUMBER — a numeric expression

Return value: base 10 log of the expression

Example: @logten(1000)@

Example result: 3

LOV(@SQLLIST(CONNECTION, SQL)@, ROW)

Return a specific row from an SQL query.

Parameters:

- SQLLIST — a call to the sqllist function which executes a select statement, returning one or more rows,
- ROW — the number of the row required, starting at 1.

Return value: the required row. if more than one colum is selected, then the results are returned separated by commas.

Example: @lov(@sqllist(Ptravel, select name from cities)@,4)@

Example result: Canterbury

LOWER(STRING)

Return a string in lower case. If the string is double quoted, CA Datamaker returns the quotes as well, but CA TDM Portal does not.

Parameters:

- STRING — a string expression

Return value: a lower case string

Example: @lower(HELLO)@

Example result: hello

LUHN(LENGTH)

Returns a random number of the given length, including a check digit in the end, using the Luhn algorithm.

Parameters:

- LENGTH — length of the number to be returned including checksum digit.

Return value: a number

Example: @luhn(6)@

Example result: 459818

MAX(VALUE1,VALUE2[...,VALUEn])

Returns the maximum value in a list of numbers or strings. If the list contains strings that cannot be converted to numbers, the function makes a string comparison on all items. The maximum value is the one that is lexograpically greater than the others. For example, Z is greater than A. If all list items can be converted to numbers, the function makes a numeric comparison.

Parameters:

- VALUE1…VALUEn — a list of numeric or string expressions. If empty, the function fails.

Return value: The largest item in the list.

Example: @max(^PASSENGER1_MONEY^)@

Example result: the largest value in the money column.

Example: @max(1,2,3,4,5)@

Example result: 5

MEAN(VALUE1,VALUE2, …VALUEn)

Returns the mean of a list of numeric values.

Parameters:

- VALUE1… VALUEn — a list of numeric expressions

Return value: the mean of the list

Example: @mean(1,2,3)@

Example result: 2

MID(STRING, STARTPOS)

Returns a substring starting from the given position, up to the end of the string.

Parameters:

- STRING — a string expression.
- STARTPOS — starting position (first character is position 1).

Return value: a substring

Example: @mid(ABCDEFGHIJK, 2)@

Example result: BCDEFGHIJK

MID(STRING,STARTPOS,LENGTH)

Returns a substring starting from the given position and of the specified length.

Parameters:

- STRING — a string expression. In Datamaker, do not enclose this string in quotes, otherwise the quotes are included in the start position.
- STARTPOS — starting position (first character is position 1). If the position is past the end of the string, an empty string is returned.
- LENGTH — the required length of the string. If zero, then an empty string is returned.

Return value: a substring

Example: @mid(ABCDEFGHIJKL, 3, 4)@

Example result: CDEF

MIN(VALUE1,VALUE2,…VALUEn)

Return the minimum value in a list of numbers or strings. If the list contains strings that cannot be converted to numbers, then a string comparison is made on all items. The minimum value is the one that is lexograpically less than the others. For example, A is less than Z.
If all list items can be converted to numbers then a numeric comparison is made.

Parameters:

- VALUE1…N — a list of numeric or string expressions. If empty, the function fails.

Return value: the smallest item in the list.

Example: @min(^PASSENGER1_MONEY^)@

Example result: 100

Example: @min(1,2,3,4,5)@

Example result: 1

MOD(NUMBER1, NUMBER2)

Returns the modulo of two numbers. This is the remainder of the first number divided by the second number.

Parameters:

- NUMBER1 and NUMBER2 — numeric expressions

Return value: remainder of NUMBER1 divided by NUMBER2

Example: @mod(5,2)@

Example result: 1

MULTIPLY(NUMBER1, NUMBER2)

Multiplies the two numbers.

Parameters:

- NUMBER1 and NUMBER2 — numeric expressions

Return value: value of NUMBER1 multiplied by NUMBER2

Example: @multiply(2,3)@

Example result: 6

NEXTSTRINGVAL(SEQUENCE, STARTVAL, DIGITS)

Returns the next value in the specified sequence as a string using a list of digits. The sequence is created if it does not exist and assumes a starting value of startval. The list of digits is used to represent a number in a base equal to the length of the string. For example, abcd represents base 4 using those digits, [0-9] would represent decimal and [0-9][A-F] would represent hexadecimal. The sequence is used to remember the previous value as a decimal number. The function increments the value of the sequence and converts the decimal value to a number in the requested base using the digits.

Parameters:

- SEQUENCE — the name of a sequence,
- STARTVAL — starting value of the sequence if it does not already exist,
- DIGITS — a list of characters to be used to represent the next value. You can specify letters, numbers and ranges in the form [a-z], for example abc[0-9].

Return value: a string representing the next value in the SEQUENCE using the supplied digits.

Example: @nextstringval(myseq,7,12345670)@

Example result: 10 (This example returned the next value in myseq, assuming base 8 (octal), and starting the sequence at 7.)

NEXTVAL(SEQUENCE)

Return the next value in the specified sequence. The sequence is created if it does not exist. The sequence starts at 1.

Parameters:

- SEQUENCE — name of a sequence

Return value: the next value in the sequence

Example: @nextval(mysequence)@

Example result: 1

NEXTVAL(SEQUENCE, STARTVAL)

Return the next value in the specified sequence. The sequence is created if it does not exist and assumes a starting value of startval. If no startval is given, sequence always starts from 1.

Parameters:

- SEQUENCE — name of a sequence,
- STARTVAL — a numeric expression giving the starting value

Return value: the next value in the sequence

Example: @nextval(EXAMPLE, 10)@

Example result: 10

NOT(BOOLEAN)

Applies a logical NOT operation to a Boolean expression.

Parameters:

- BOOLEAN — a boolean expression

Return value: true when BOOLEAN=false and false when BOOLEAN=true

Example: @not(1=1)@

Example result: false

OCCURS(STRING, STRINGTOFIND)

Counts the number of occurrences of a string in another string.

Parameters:

- STRING — a string expression,
- STRINGTOFIND — a string expression to search for

Return value: number of occurrences of STRINGTOFIND

Example: @occurs(hello hello there folks,hello)@

Example result: 2

OCCVAL(NUMBER %STRING,NUMBER%STRING[,NUMBER%STRING...])

Return a random value from a list of strings. The list is composed using arguments of the form: number%string. In Datamaker, do not enclose these arguments in quotes.

Parameters:

- NUMBER — the number of times to repeat the following string
- STRING — a string expression to be repeated

Return value: a random value from the composed list.

Example: @occval(100%A,300%B)@

Example result: B, because the function assumes a list composed of 100 A's and 300 Bs and then selects a random item from that list.

OR(BOOLEAN1, BOOLEAN2[,BOOLEAn...])

Perform a logical OR operation on a list of Boolean expressions.

Parameters:

- BOOLEAN1...BOOLEANn — a Boolean expression

Return value: true if any expression is true, and false if they are all false.

Example: @or(~var1~,1=1,1=2,1=3)@

Example result: true

PERCVAL(N%STRING, N%STRING[,N%STRING...])

Returns a random value from a list of values generated using a percentage. Each argument has the form: <percentage of the total list>%<string expression>. All the percentages must add up to 100 otherwise the function fails. For example, an argument list of "30%fred,70%jim" would assume a list that contains 30% freds and 70% jims.
The difference between @randlov(0, @perclist(50%DEBIT,50%CREDIT)@)@ and @percval(50%DEBIT,50%CREDIT)@ is stability. Multiple calls to the former, within a row, always returns the same value, whereas multiple calls to the latter, within a row, do not. This is true for all @randlov() functions - they all return the same value in the same row.
In Datamaker, do not enclose these arguments in quotes.

Parameters:

- N — the percentage of the associated STRING present in the list,
- STRING — a string expression.

Return value: A random value from a list

Example: @percval(10%A,90%B)@

Example result: B, because the function assumes a list composed of 10 A's and 90 Bs and then selects a random item from that list.

POS(STRING, STRINGTOFIND)

Returns the first position of stringtofind in a string.

Parameters:

- STRING — a string expression to be searched,
- STRINGTOFIND — a string expression to find

Return value: First position of STRINGTOFIND

Example: @pos(^LAST_NAME^,a)@

Example result: 3 (where LAST_NAME=Stacey)

POS(STRING, STRINGTOFIND, STARTPOS)

Return the first position of stringtofind in the string starting at startpos.

Parameters:

- STRING — a string expression to be searched,
- STRINGTOFIND — a string expression to find

Return value: First position of STRINGTOFIND after STARTPOS

Example: @pos(banana,a,3)@

Example result: 4

RANDCHARS(MINLEN,MAXLEN,CHARLIST)

Returns a random string of a length between minlen and maxlen, consisting only of the specified characters.

Parameters:

- MINLEN — minimum length of the resulting string,
- MAXLEN — maximum length,
- CHARLIST — a string

Return value: a string of characters to be used in the random string

Example: @randchars(3,10, AEIOURSTLNE)@

Example result: EEOLES

RANDDATE(MIN, MAX)

Returns a random date between the minimum and maximum values.

Parameters:

- MIN — minimum date,
- MAX — maximum date

Return value: A random date in project format

Example: @randdate(2009/01/01,2010/01/01)@

Example result: 2009/12/07

RANDDIGITS(MINLEN, MAXLEN)

Returns a random string of digits of a length between minlen and maxlen.

Parameters:

- MINLEN — minimum length of the resulting string,
- MAXLEN — maximum length

Return value: String of digits

Example: @randdigits(3,10)@

Example result: 2103

RANDEXP(MIN, MAX, MEAN)

Returns a positive number on an exponential distribution with a mean as specified. Only values between min and max are returned.

Parameters:

- MIN — minimum value,
- MAX — maximum value,
- MEAN — mean value

Return value: A positive number

Example: @randexp(1,10,4)@

Example result: 5

RANDHEX(MINBYTES, MAXBYTES)

Returns a random hex string of a random length between the minimum and maximum length.

Parameters:

- MINBYTES — minimum length of resulting string,
- MAXBYTES — maximum length

Return value: A random hexadecimal string

Example: @randhex(1,4)@

Example result: 8A8172

RANDLOV(PERCNULL,@DIRLIST(DIRECTORY)@)

(Datamaker only) Returns a random value (with percentage of nulls that are specified by percnull) from the list of files that are contained in the specified directory. All @randlov() functions return the same value in the same row.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.,
- DIRECTORY — the directory to search for files

Return value: A random value from the list of files in the named directory

Example: @randlov(0, @dirlist(C:\TEMP)@)@

Example result: a random filename from the c:\temp directory.

RANDLOV(PERCNULL, @LIST(STRING, STRING[,STRING])@)

Returns a random selection from a list of values. All @randlov() functions return the same value in the same row.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- LIST — a list of string expressions and values that are separated by commas. In Datamaker, quoted strings are returned with their quotes; CA TDM Portal removes them.

Return value: A randomly chosen value from the list or maybe a null.

Example: @randlov(0, @list(Devon,Cornwall,Surrey)@)@

Example result: Cornwall

RANDLOV(PERCNULL, @OCCLIST(N%STRING, N%STRING[,N%STRING…])@)

Returns a random value (with percentage of nulls that are specified by percnull) from the specified occurrence list. All @randlov() functions return the same value in the same row.

Parameters:

- PERCNULL — percentage of nulls,
- OCCLIST — occurrence list,
- N — a percent value

Return value: A random value from a list

Example: @randlov(0, @occlist(50%VI,50%MC)@)@

Example result: MC

RANDLOV(PERCNULL, @OCCLIST(N%STRING, N%STRING[,N%STRING…])@)

Returns a random value from the specified occurrence list. All @randlov() functions return the same value in the same row.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- OCCLIST — an occurrence list. This is a list created from a set of items in the form: <number of items>%<a string expression>. For example, 10%A,20%B would create a list containing 10 A's and 20 B's.

Return value: A random value from the list or NULL if percnull > 0.

Example: @randlov(0, @occlist(50%VI,50%MC)@)@

Example result: MC

RANDLOV(PERCNULL, @PERCLIST(N%STRING, N%STRING[,N%STRING])@)

Return a random selection from a list of values generated by a @PERCLIST. The difference between @randlov(0, @perclist(50%DEBIT,50%CREDIT)@)@ and @percval(50%DEBIT,50%CREDIT)@ is stability. Multiple calls to the former, within a row, always returns the same value, whereas multiple calls to the latter, within a row, do not. This is true for all @randlov() functions - they all return the same value in the same row.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- PERCLIST — a percentage occurrence list. This is a list created from a set of items in the form: <% of items>%<a string expression>. For example, 10%A,90%B would create a list containing 10% A's and 90% B's.

Return value: A random value from the list or NULL if percnull > 0.

Example: @randlov(0, @perclist(10%Devon,20%Cornwall,70%Surrey)@)@

Example result: Cornwall

RANDLOV(PERCNULL,@SEEDLIST(SEEDNAME)@[,SEEDCOLUMN,][INVALIDVAL])

Return a random value from a column in a seed list. All @randlov() functions return the same value in the same row. Some seed lists have more than one column, for example, DayOfWeek contains two columns, one for the day number and the other for the day name. See also Seed Lists.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- SEEDNAME — name of a seed list, eg DayOfWeek. If the seed list has more than one column then the first column is chosen. In Datamaker, do not enclose this string in quotes.
- INVALIDVAL — (optional) a value from the seed list that will not be returned. If a randomly chosen item in the seed list has this value then it is skipped and another value is chosen.
- SEEDCOLUMN — (optional) the column in the SEEDLIST to be fetched. This can be the column number or name.

Return value: A random value from the specified column in the seed list, or NULL if percnull > 0.

Example: @randlov(0,@seedlist(streetname)@,1)@

Example result : 9 St Thomas Street

Example: @randlov(0,@seedlist(DayOfWeek)@,Numeric Day)@

Example result : 2

Example: @randlov(0,@seedlist(month)@)@

Example result: April

RANDLOV(PERCNULL, @SQLLIST(CONNECTION, SQL)@[,COLUMN][, INVALIDVAL])

Returns a random value from a sql query. All @randlov() functions return the same value in the same row.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- CONNECTION — Defines the dbms connection type. Choose one of the following:
  - R — Repository
  - S — Source
  - T — Target
  - P*profilename* — connection profile, for example `Pmyconprof` .
- SQL — A SQL query. Only select statements are supported.
- COLUMN — (optional argument). The number or name of the column to be returned. If this is omitted and multiple columns are present in the query then those columns are returned separated by commas.
- INVALIDVAL — (optional argument). A invalid value that is never returned. If a randomly chosen row has this value then it is skipped and another row is chosen.

Return value: a random item from the query or maybe a null if percnull > 0

Example: @randlov(0, sqllist(Ptravel, select names from cities))@

Example result: Albury

Example: @randlov(0, sqllist(Ptravel, select distinct expiration_date from credit_cards),1,2000/01/01)@

Example result: 2008/05/01

RANDLOV(PERCNULL, @PRIORPUBLISHKEYLIST(LD_ID, TABLENAME)@)

Returns a random row from the key list of a prior publish (for level ID (LD_ID) and table (TABLENAME)). All @randlov() functions return the same value in the same row.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- LD_ID — the id of the generator that performed the publish. This is automatically inserted by datapainter when composing the expression containing countlist.
- TABLENAME — the name of the table whose columns were saved.
- COLUMN — (optional argument). The number or name of the column to be returned. If this is omitted and multiple columns are present in the query then those columns are returned separated by commas.
- INVALIDVAL — (optional argument). A invalid value that is never returned. If a randomly chosen row has this value then it is skipped and another row is chosen.

Return value: A random row from a previously published table

Example: @randlov(0, @priorpublishkeylist(1007, PEOPLE)@)@

RANDLOV(PERCNULL, @PRIORTESTMATCHLIST(LD_ID, TESTNAME)@)

(Datamaker only) Returns a random row from the key list of a prior test match for level ID (LD_ID) and test name (TESTNAME). All @randlov() functions return the same value in the same row. This function is only supported in Datamaker and not in TDM Portal 4.0.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- LD_ID — the id of the generator that performed the publish. This is automatically inserted by datapainter when composing the expression containing countlist.
- TESTNAME — test name

Return value: A random row from the test match results

Example: @randlov(0, @priortestmatchlist(4000,TEST1)@)@

RANDLOV(PERCNULL, @WADLLIST(URL, COLUMNNAME)@)

(Datamaker only) Returns a random row from a REST call. This function is only supported in Datamaker and not in TDM Portal 4.0.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- URL — a REST UIRL to be called. Only GET is supported.
- COLUMNNAME — name of the element or an XPATH of the element to be returned

Return value: A random row

Example: @randlov(0, wadllist(http://server:5091/Service?LIST, CustomerName)@)@

RANDLOV(PERCNULL, SOURCES[,INVALIDVAL])

(Datamaker only) This function allows you to specify a list of sources (A to J) from which the list is drawn. This function is only supported in Datamaker and not in TDM Portal 4.0.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- SOURCES — sources(A to J)
- INVALIDVAL — (optional argument) An invalid value that is never returned. If one is chosen, then it is skipped and another is chosen.

Return value: A random item from the list

Example: @randlov(0, G)@,

Example result: Base US Visa

Example: @randlov(0,G,UK-US Visa)@

Example result: Standard US Visa

RANDNORM(MIN,MAX,MEAN,STDDEV)

Returns a normally distributed number between the min and max values with the specified mean and standard deviation.

Parameters:

- MIN — minimum value
- MAX — maximum value
- MEAN — mean value
- STDDEV — standard deviation

Return value: a floating point value

Example: @randnorm(1,100,50,12)@

Example result: 43.1

RANDNULL(PERCNULL, EXPR)

Returns the value of the expression or randomly a null.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- EXPR — a numeric or string expression

Return value: the value of the expression or maybe a null if percnull > 0

Example: @randnull(50, @randtext(1,20)@)@

Exam result: yuniuidfniub

RANDRANGE(MIN, MAX,[WIDTH])

Returns a random integer between min and max.

Parameters:

- MIN — minimum allowed
- MAX — maximum allowed
- WIDTH — (optional argument) if specified then the result is zero padded up to this width

Return value: A random integer

Example: @randrange(1,2)@,

Example result: 1,2

Example: @randrange(100,1000,8)@,

Example result: 00000234

RANDTEXT(MINLEN,MAXLEN[,CASE])

Create random text of a length between the min and max.

Parameters:

- MIN — minimum length of result
- MAX — maximum length
- CASE — (optional argument) If omitted, capitalized text is assumed.
  - U for upper case text,
  - M for capitalized text (Default)
  - L for lower case text.

Return value: a random string

Example: @randtext(3,50, M)@,

Example result: Ghjhj GGY hhjh

RANDTIME(MIN, MAX)

Returns a random time between a min and max.

Parameters:

- MIN — minimum time allowed in the format hh:mm
- MAX — maximum time allowed

Return value: A random time

Example: @randtime(00:00:00,23:59:59)@

Example result: 12:34:01

RANDVAL(SEQUENCE, MAXVAL)

Generates a more or less unique but not sequential number. The function can take any sequence name. The sequence is created if it does not exist. Remember that the sequence is in the repository, not the target.

Parameters:

- SEQUENCE — name of a sequence. If the sequence does not exist, it is created.
- MAXVAL — maximum allowed value. The parameter defines the size of the range of values the function returns. The smaller maxval, the greater the rate at which duplicates are generated.

Return value: A unique number

Example: @randval(myseq, 999999)@

Example Result: 1245 depending on how many times it is called.

REPEAT(EXPR,OCCURS,SEPARATOR)

Generates a string by evaluating an expression a number of times and concatenating the values separated by a character. The expression is evaluated multiple times as it may return a different result each time (for example, a random value from a seed list). It is not assumed that it has a constant value.

Parameters:

- EXPR — a numeric or string expression
- OCCURS — the number of times to repeat
- SEPARATOR — the separator character or string

Return value: A string containing a set of values of the expression.

Example: @repeat(hello,5,",")@

Example result: hello,hello,hello,hello,hello

REPLACE(STRING, STRINGTOREPLACE1, REPLACEMENTSTRING1, STRINGTOREPLACE2, REPLACEMENTSTRING2, etc)

Returns value of parameter STRING, with each occurence of substring STRINGTOREPLACE# replaced with REPLACEMENTSTRING#. This function takes a minimum of 3 parameters, but has no maximum limit on the number of parameters. The number of parameters must be odd (i.e. the STRING to search for, plus any number of pairs of STRINGTOREPLACE# and REPLACEMENTSTRING# parameters). This function is case-sensitive.

Parameters:

- STRING — a string expression to be edited
- STRINGTOREPLACE1 — the first substring in the string to be replaced.
- REPLACEMENTSTRING1 — the string to replace the string STRINGTOREPLACE1
- STRINGTOREPLACE2 — the second substring in the string to be replaced.
- REPLACEMENTSTRING2 — the string to replace the string STRINGTOREPLACE2
- etc...

Return value: a string

Example: @replace(King Kong, K, M, ng, la)@

Example result: Mila Mola

REVERSE(STRING)

Returns the reverse of a string.

Parameters:

- STRING — a string expression. In Datamaker, quoted strings are returned with their quotes. Portal removes them.

Return value: A string

Example: @reverse(ABC)@

Example result: CBA

RIGHT(STRING, LENGTH)

Return the specified number of characters from the right of a string.

Parameters:

- STRING — a string expression
- LENGTH — number of characters to return.

Return value: A substring ending at the right of the source string

Example: @right(hello, 2)@

Example result: lo

RIGHTPAD(STRING,CHARTOPAD[,LENGTH])

Pads the string to the required length by adding chartopad to the right.

Parameters:

- STRING — a string expression
- CHARTOPAD — padding character or string
- LENGTH — (optional argument) the required string length. If omitted, the column width is used.

Return value: A string padded to the required length

Example: @rightpad(ABCDE,#,10)@

Result: ABCDE#####

RIGHTTRIM(STRING, CHARTOTRIM)

Trims a substring from the right-hand end of a string.

Parameters:

- STRING — a string expression.
- CHARTOTRIM — character to trim

Return value: A trimmed string

Example: @righttrim(hello*****,*)@

Example result: hello

ROUND(NUMBER, DECPLACES)

Round a number up to a number of decimal places.

Parameters:

- NUMBER — a numeric expression
- DECPLACES — the number of decimal places

Return value: a number

Example: @round(1.262776,2)@

Example result: 1.27

Example: @round(1.55,0)@

Example result: 2

Example:@round(1.55,1)@

Example result: 1.6

Example: @round(1.551,1)@

Example result:1.6

Example: @round(1.449,1)@

Example result:1.4

SECONDSAFTER(STARTDATETIME, ENDDATETIME)

Return the number of seconds between a start and end datetime or time.

Parameters:

- STARTDATETIME — start time or datetime in project or other format, eg 20-01-2017 14:42
- ENDDATETIME — end time datetime

Return value: The number of seconds between those datetimes.

Example: @secondsafter(26/10/2008:01:30:56,26/11/2008:01:30:59)@

Example result: 2678403

Example: @secondsafter(01:30:56,01:30:59)@

Example result: 3

SEEDLIST(SEEDNAME[, COLUMNINDEX])

Returns a list (or list of lists) of values from the table **gtrep_reference_data** in gtrep. This table contains many seedlists, separated into categories by the value in column **rd_ref_id**. SEEDLIST returns all **rd_ref_value_**# columns (starting from **rd_ref_value_1**) in a category defined by parameter *SEEDNAME*.

The optional parameter *COLUMNINDEX* defines the index of a column in the seedlist.

Parameters:

- SEEDNAME - name of the seedlist category you want to return. Must match a value present in **rd_ref_id**.
- (Optional) COLUMNINDEX - index (starting from 0) of the column in the seedlist that you want to return.

Return value: A list (or list of lists) of values (one list per column in category).

Example: @seedlist(US Zip-Codes, 1)@

Example result: Second column (i.e. "State") of seedlist 'US Zip-Codes'.

SEQLOV(PERCNULL, @ALLPAIRS(LIST1,LIST2,[LIST], ALL_COMBINATIONS)@, COLUMN)

(Datamaker only) Returns the next item from a list of all combinations of pairs for LIST1, LIST2... for a particular column COLUMN. Each time the function is called, the next item is returned. When the end of the list is reached, the list wraps back to the beginning. This is only supported in Datamaker and not TDM Portal 4.0.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- @ALLPAIRS — a list generated from all combinations of two lists
- COLUMN — name or number of the column to be returned if the lists have multiple columns.

Return value: the next item from the all pairs list.

Example: @seqlov(0, @allpairs(@list(abc,bcd,def)@, @list(kkk,llll,mmm)@, N)@,1)@,

Example Result: [(abc, kkk), (abc,llll), (abc,mmm), (bcd,kkk), (bcd, llll), (bcd,mmm), (def, kkk), (def, llll), (def, mmm)]

SEQLOV(PERCNULL,@DIRLIST(DIRECTORY)@)

Returns a sequential value (with percentage of nulls that are specified by percnull) from the list of files that are contained in the specified directory.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- DIRECTORY — the directory to search for files

Return value: the next value from the list of files in the named directory.

Example: @seqlov(0, @dirlist(C:\TEMP)@)@

Example result: c:\temp\fred.txt

SEQLOV(PERCNULL, @LIST(STRING, STRING[,STRING])@)

This function allows for sequential selection from a list of values. It includes any values of interest inside the @LIST(...)@ construct, separated by commas.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- @LIST: a list of string expressions and values separated by commas

Return value: The next value from the list or maybe a null.

Example: @seqlov(0, @list(Devon,Cornwall,Surrey)@)@

Example result: Devon – the next call would return Cornwall.

SEQLOV(PERCNULL, @OCCLIST(N%STRING, N%STRING[N%STRING…])@)

Returns a the next sequential value from an occurrence list.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- OCCLIST — an occurrence list. This is a list created from a set of items in the form: <number of items>%<a string expression>. For example, 10%A,20%B would create a list containing 10 A's and 20 B's.

Return value: The next value from the list or maybe a null if percnull > 0.

Example: @seqlov(0, @occlist(50%VI,50%MC)@)@

Example result: VI

SEQLOV(PERCNULL, @PERCLIST(N%STRING, N%STRING[,N%STRING])@)

Return the next sequential item from a list of values generated by a @PERCLIST.

The difference between @randlov(0, @perclist(50%DEBIT,50%CREDIT)@)@ and @percval(50%DEBIT,50%CREDIT)@ is stability. Multiple calls to the former, within a row, always returns the same value, whereas multiple calls to the latter, within a row, do not. This is true for all @randlov() functions - they all return the same value in the same row.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- PERCLIST — a percentage occurrence list. This is a list created from a set of items in the form: <% of items>%<a string expression>. For example, 10%A,90%B would create a list containing 10% A's and 90% B's.

Return value: The next value from the list or NULL if percnull > 0.

Example: @seqlov(0, @perclist(10%Devon,20%Cornwall,70%Surrey)@)@

Example result: Devon

SEQLOV(PERCNULL, @PRIORPUBLISHKEYLIST(LD_ID, TABLENAME)@[, COLUMN][,INVALIDVAL])

(Datamaker only) Returns the next sequential item from the key list of a prior publish (for level ID (LD_ID) and table (TABLENAME)). All @seqlov() functions return the same value in the same row. This function is only supported in Datamaker and not in TDM Portal 4.0.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- LD_ID — the id of the generator that performed the publish. This is automatically inserted by datapainter when composing the expression containing countlist.
- TABLENAME — the name of the table whose columns were saved.
- COLUMN — (optional argument). The number or name of the column to be returned. If omitted, the first column is assumed.
- INVALIDVAL — (optional argument). A invalid value that is never returned. If a randomly chosen row has this value then it is skipped and another row is chosen.

Return value: A random row from a previously published table

Example: @seqlov(0, @priorpublishkeylist(1007, PEOPLE)@,1)@

SEQLOV(PERCNULL, @PRIORTESTMATCHLIST(LD_ID, TESTNAME)@[, COLUMN])

(Datamaker only) Returns a sequential row from the key list of a prior test match for level ID (LD_ID) and test name (TESTNAME). (COLUMN) is the column number from the list to extract values from. This function is only supported in Datamaker and not in TDM Portal 4.0.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- LD_ID — the id of the generator that performed the publish. This is automatically inserted by datapainter when composing the expression containing countlist.
- TESTNAME — the name of the testmatch previously run.
- COLUMN — (optional argument). The number or name of the column to be returned. If omitted, the first column is assumed.
- INVALIDVAL — (optional argument). A invalid value that is never returned. If a randomly chosen row has this value then it is skipped and another row is chosen

Return value: The next sequential value

Example: @seqlov(0, @priortestmatchlist(4000,TEST1)@,1)@

SEQLOV(PERCNULL,@SEEDLIST(SEEDNAME[,S])@,SEEDCOLUMN, INVALIDVAL)

Returns the next sequential value from a column in a seed list. All @seqlov() functions return the same value in the same row. Some seed lists have more than one column, for example, DayOfWeek contains two columns, one for the day number and the other for the day name. See also Seed Lists.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- SEEDNAME — name of a seed list, eg DayOfWeek. If the seed list has more than one column then the first column is chosen. In Datamaker, do not enclose this string in quotes.
- S — (optional) If the literal argument S is specified then the seed list is randomly shuffled before returning the first item. Subsequent calls do not re-shuffle.
- INVALIDVAL — (optional) a value from the seed list that will not be returned. If a randomly chosen item in the seed list has this value then it is skipped and another value is chosen.
- Return value — A random value from the seed list or NULL if percnull > 0.
- SEEDCOLUMN — the column in the SEEDLIST to be fetched. This can be the column number or name.

Return value: The next sequential value from the specified column in the SEEDLIST, or possibly a null value if percnull > 0.

Example: @seqlov(0,seedlist(streetname),1,Oxford Circus)@

Example result: 10 Wall Street

SEQLOV(PERCNULL, @SQLLIST(CONNECTION, SQL[,S])@ [,COLUMN] [,INVALIDVAL])

Return the next sequential value from a sql query. All @seqlov() functions return the same value in the same row.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- CONNECTION — Defines the dbms connection type. Choose one of the following:
  - R — Repository
  - S — Source
  - T — Target
  - P*profilename* — connection profile, for example `Pmyconprof` .
- SQL — A SQL query. Only select statements are supported.

  WARNING

  The SQL query that SEQLOV takes as an input, must not include the keyword '**`Compute_`** '. If you have a column that contains this keyword, assign the column an alias (without special characters).

  Example of a call that throws an error: `@seqlov(0,@sqllist(MyConnection,select` **`Compute_AgeInYears`** `from dbo.testTable)@,~COLUMN_NAME~)@`

Example of a workaround with alias: `@seqlov(0,@sqllist(MyConnection,select **Compute_AgeInYears as ComputeAge** from dbo.testTable)@,~COLUMN_NAME~)@`

- S — (optional) If the literal argument S is specified then the rows returned by the query are randomly shuffled before returning the first item. Subsequent calls do not re-shuffle.
- COLUMN — (optional) The number or name of the column to be returned. If this is omitted and multiple columns are present in the query then those columns are returned separated by commas.
- INVALIDVAL — (optional) An invalid value that is never returned. If an item has this value then it is skipped and another item is chosen.

Return value: The next sequential item from the query or maybe a null if percnull > 0

Example: @randlov(0, sqllist(Ptravel, select names from cities))@

Example: @seqlov(0, sqllist(Ptravel, select distinct expiration_date from credit_cards),1, 2000/01/01)@

Example result: 2008/05/01 (2000/01/01 will never be returned)

Example: @seqlov(0, sqllist(Ptravel, select distinct expiration_date from credit_cards,S),1)@

Example r esult: 2010/11/01 (the list is shuffled first)

Example: @seqlov(0, sqllist(Ptravel, select names from cities, S))@

Example r esult: Aalborg

SEQLOV(PERCNULL,@UITESTLIST(LENGTH)@, REPEATTYPE)

Returns a sequential value from the UI Test List specified by UITESTLIST. Values have the specified maximum length (exception is hex strings, which represent bytes). Find the list of Valid Values for RepeatType .

Parameters:

- PERCNULL — percentage of nulls
- UITESTLIST — UI Test List
- LENGTH — length
- REPEATTYPE — RepeatType Values

Return value: The next sequential value rom the uitest list

Example: @seqlov(0, @uitestlist(5)@, NUMERIC)@

SEQLOV(PERCNULL, @WADLLIST(URL, COLUMNNAME)@)

(Datamaker only) Returns the next sequential row from a REST call. This function is only supported in Datamaker and not in TDM Portal 4.0.

Parameters:

- PERCNULL — percentage of nulls to be expected over a large number of calls. For example, a value of 50% means that there is a 50-50 chance of an "address line 2" value being a null.
- URL — a REST URL to be called. Only GET is supported.
- COLUMNNAME — name of the element or an XPATH of the element to be returned

Return value: The next sequential row from the results of the REST call

Example: @seqlov(0, wadllist(http://server:5091/Service?LIST, CustomerName)@)@

SEQLOV(PERCNULL, SOURCES[,INVALIDVAL])

Selects a sequential list of values from the source, a percentage of the values are designated as null. Source can be A to J.

Parameters:

- PERCNULL — percentage of nulls
- SOURCES — source (A to J)
- INVALIDVAL — (optional) An invalid value that is never returned. If an item has this value then it is skipped and another item is chosen.

Return value: The next sequential list value from the list

Example: @seqlov(0, G)@, m

Example r esult: Base US Visa

SIGN(NUMBER)

Returns the sign of a number.

Example: @sign(-110)@

Example result: - (that is, the character 'minus')

SINE(COUNTER, PERIOD)

Return the sine of the counter with respect to the period, that is, it returns sin((2+counter+π)/period)

Parameters:

- COUNTER — a numeric expression
- PERIOD — a numeric expression

Return value: the value

Example: @sine(4,32)@

Example r esult: 0.7071067811865475

SINE(MAGNITUDE,COUNTER,PERIOD)

Returns the value of magnitude*sin(2+counter + π)/period).

Parameters:

- MAGNITUDE — a numeric expression
- COUNTER — a numeric expression
- PERIOD — a numeric expression

Return value: the sine value

Example: @sine(4,23,12)@

Result: -2.000000000000006

SNGLQUOTE()

Returns a single quote character. Use this function in expressions where a quote could be ambiguous.

Return value: a single quote character

Example: @snglquote()@

Example result: '

SQLHEXLIST(CONNECTION, SQL[,S])

(Mainframe/DB2 only) Execute a SQL query on a specified connection and return a list. Use the sqlhexlist() function for DB2 tables on mainframe only. Use this function only inside the SEQLOV function. If you try to use it outside, it will only return a list identification string (for example Q1) which only the LOV functions understand. In Datamaker, sqlhexlist() functions exactly the same way as sqllist(). It is in Datamaker purely for validation purposes.

Data Types:

The columns are varchar, nvarchar, char, or nchar data types, and storing binary data (or packed data). For other data types, sqlhexlist() falls back to sqllist().

Syntax:

The select statement in sqlhexlist() has a strict syntax. The select list is either 'select *' or 'select *the-column-name*'. The statement does not support multiple column selection.

Parameters:

- CONNECTION — Defines the dbms connection type. Choose one of the following:
  - R — Repository
  - S — Source
  - T — Target
  - P*profilename* — connection profile, for example Pmyconprof.
- SQL — A SQL query. Only select statements are supported.
- S — (optional) If the literal argument S is specified then the rows returned by the query are randomly shuffled.

Examples:

Example 1: `@seqlov(0,@sqlhexlist(Pdb2-mainframe-CA31,"SELECT * FROM TDMTEST.HEX_DATA")@,c_varchar)@`

Example 1: `@seqlov(0,@sqlhexlist(Pdb2-mainframe-CA31,"SELECT c_varchar FROM TDMTEST.HEX_DATA")@,c_varchar)@`

SQLLIST(CONNECTION, SQL[,S])

Execute a SQL query on a specified connection and return a list. This function can only be used inside functions such as SEQLOV, RANDLOV or HASHLOV, which pick an item from a list. If you try to use it outside, it will only return a list identification string (for example Q1) which only the LOV functions understand.

Parameters:

- CONNECTION — Defines the dbms connection type. Choose one of the following:
  - R — Repository
  - S — Source
  - T — Target
  - P*profilename* — connection profile, for example `Pmyconprof` .
- SQL — A SQL query. Only select statements are supported.
- S — (optional) If the literal argument S is specified then the rows returned by the query are randomly shuffled.

STDDEV(NUMBER[,NUMBER…])

Returns the standard deviation for the specified list of numbers.

Parameters:

- NUMBER… — a list of numeric expressions

Return value: the standard deviation of the list

Example: @stddev(23,42)@

Example r esult: 9.5

STRING(DATE, DATEFORMAT)

Returns a date in the given date format. Parts of the date can be returned, not just whole dates.

Parameters:

- DATE — a date - usually a system or user variable or date valued function or a column reference.
- DATEFORMAT — the required date format, e.g. MM for just a month number, DD-MM-YYYY for a date

Return value: The date in the new format

Example: @string(~SDATE~, MM)@

Example r esult: 07

STRING(NUMBER, NUMBERFORMAT)

Convert the number into string that is based on the number format. This function is not supported in TDM Portal 4.0: it simply returns the number unchanged.

Parameters:

- NUMBER — A numeric expression
- NUMBERFORMAT — format of a number. This is a string of digits chosen from this list:
  - 9 — a corresponding digit from the number
  - . — a decimal point
  - 0 — a leading zero or a digit

Return value: A formatted number

Example: @string(45,0000)@

Example result: 0045

STRING(TIME, TIMEFORMAT)

Returns the specified time as a string in the specified format.

Parameters:

- TIME — a time, usually a system or user variable or time valued function or a column reference.
- TIMEFORMAT — the required time format.

Return value: Time as a string

Example: @string(~STIME~, HH:MM:SS)@

Example r esult: 16:54:23

SUBTRACT(NUMBER1,NUMBER2)

Subtracts one number from another.

Parameters:

- NUMBER1,2 — numeric expressions

Return value: NUMBER1-NUMBER2

Example: @subtract(3,1)@

Example result: 2

SUM(NUMBER[,NUMBER…])

Returns the sum of a list of numbers.

Parameters:

- NUMBER… — a list of numeric expressions

Return value: The sum of the list of number values.

Example: @sum(1,2,3)@

Example r esult: 6

SWEDEN_SSN(SEPARATOR, GENDER)

Returns a Swedish Social Security Number with a specified separator and gender.

Parameters:

- SEPARATOR — a separator character; must be either + or – (minus).
- GENDER — the string Male or Female or M or F

Return value: A Swedish Social Security Number

Example: @sweden_ssn(|, Female)@

Example r esult: 281002|403

SWEDEN_SSN(DOB, GENDER)

Returns a Swedish Social Security Number with the specified date of birth and gender.

Parameters:

- DOB — date of birth
- GENDER — the string Male or Female or M or F

Return value: A string representing the Swedish Social Security Number as a string

Example: @sweden_ssn(1901-03-06, Male)@

Example r esult: 010306+191

TCKID([NUMBER[, WIDTH]])

(Datamaker only) Returns an 11-digit Turkish National ID, according to the Turkish MERNIS project. If you provide the optional NUMBER parameter, the first 4 digits are randomly generated, and the next 5 digits are fixed values. The optional WIDTH parameter defines the number of characters that will be fixed values. This function is supported in Datamaker and CA TDM Portal 4.1.

Parameters:

- NUMBER — a number
- WIDTH — a number.

Default: If you provide no parameters, all first 9 digits are randomly generated.

Example: @tckid(@nextval(abc)@)@ generates 99,999 unique numbers.

Example: @tckid(@nextval(abc)@, 8)@ generates 99,999,999 values.

Return value: a Turkish ID

TCKTAXID()

Returns a 10 digit generated Turkish Tax ID.

Example: @tcktaxid()@

Result: 9821890865

TILDE()

Returns the tilde character ~. This function is useful in expressions where the tilde might be confused with a variable reference.

TIME(DATETIME)

Returns the time portion of a datetime in the project format.

Parameters:

- DATETIME — a datetime in one of a number of formats. In Datamaker, quoted strings will fail to be interpreted as a date time.

Return value: the time part of the DATETIME

Example: @time(26/02/2008:12:23:22)@

Example result: 12:23:22

TRIM(STRING, CHARTOTRIM)

Trims a substring from both ends of a string.

Parameters:

- STRING — a string expression
- CHARTOTRIM — characters to trim from each end

Return value: a possibly shorter string

Example: @trim(***hello*****,*)@

Example result: hello

UK_NINO()

Returns a random UK National Insurance number.

Example: @uk_nino()@

Result: WW212145A

UPPER(STRING)

Upper-cases a string. If the string is double quoted, Datamaker returns the quotes as well, but TDM Portal does not.

Parameters:

- STRING — a string expression

Return value: the upper cased STRING

Example: @upper(hello)@

Example result: HELLO

US_SSN(SEPARATOR)

Return a random US social security number with the specified separator.

Parameters:

- SEPARATOR — a character

Return value: US social security number with specified SEPARATOR

Example: @us_ssn(-)@

Example result: 729-60-7933

VERHOEFF(LENGTH)

Return a random number of a specified length with a Verhoeff checksum.

Parameters:

- LENGTH — number of digits required

Return value: a number with Verhoeff checksum test

Example: @verhoeff(123)@

Example result: 468900650533243403923815375827290457006704602854739263036564645281919189074682062920432239894241555

WORDCAP(STRING)

Capitalizes a string. If the string is double quoted, Datamaker returns the quotes as well, but TDM Portal does not.

Parameters:

- STRING — a string to capitalize

Return value: an upper case STRING

Example: @wordcap(hello folks)@

Example result: Hello Folks

XLOOKUP(LIST, OLDVAL[, DEFAULTVALUE])

Retrieves the new value that is associated with the given old value from the given cross reference list. Use xref() to set up a cross reference list. Useful for maintaining referential integrity across multiple databases when performing updates.

If the cross reference list does not include the old value, either a NULL is returned or the default value is returned. Note that the DEFAULTVALUE is an optional parameter.

Parameters:

- LIST — a cross reference list,
- OLDVAL — value
- DEFAULTVALUE — value

Return value: new value or default value if old value is not in the list

Example: @xlookup(PEOPLE, Joe Bloggs)@

@xlookup(PEOPLE, Joe Bloggs, John Smith)@

XREF(LIST,OLDVAL,NEWVAL)

Adds the pair (old value, new value) to the given cross reference list. Returns the new value. Use xlookup() to retrieve these values later.
You typically use @xref and @lookup in a datapool that you have constructed to clone data between a source and target. Typically an identifying ID on a column is read from the source, and a new ID is generated to be used on the target. The xref function maps the old (source value) and new ID (generated value) to one another. In a later expression, you use xlookup to retrieve the mapping of the two values.

Parameters:

- LIST — a cross reference list
- OLDVAL — value
- NEWVAL — value

Return value: new value

Example: @xref(PEOPLE, Joe Bloggs, Vincent Van Gogh)@

XREFPERSIST(TABLENAME, COLUMNNAME , OLDVAL, NEWVAL)

Performs the same basic function as xref, but additionally stores the specified old and new values in the repository. The mappings of OLDVAL to NEWVAL are stored in a list with the name TABLENAME.COLUMNNAME. Xrefpersist stores the old and new values in the gtrep_pj_key_values table when the publish completes. CA TDM portal stores only the table-column combinations that use xrefpersist in the repository. In CA Datamaker, using @xrefpersist once causes all values used in both @xref and @xrefpersist to be stored to the repository. Use the @xlookup method to query the mapping of values later.

Parameters:

- TABLENAME — table name where this value has been generated,
- COLUMNNAME — column name where this value has been generated,
- OLDVAL — value,
- NEWVAL — value.

Return value: new value

Example: @xrefpersist(PEOPLE, NAME, Joe Bloggs, Vincent Van Gogh)@

Function Date Formats

These are the valid date formats to be used as function parameters in Datamaker:

```
DD-MMM-YYYY, DD:MMM:YYYY, DD MMM YYYY, DD/MMM/YYYY, DD.MMM.YYYY, DDMMMYYYY,

DD-MM-YYYY,  DD:MM:YYYY,  DD MM YYYY,  DD/MM/YYYY,  DD.MM.YYYY,  DDMMYYYY,
```

```
DD-MMM-YY,    DD:MMM:YY,    DD MMM YY,    DD/MMM/YY,    DD.MMM.YY,    DDMMMYY,
DD-MM-YY,     DD:MM:YY,     DD MM YY,     DD/MM/YY,     DD.MM.YY,     DDMMYY,
MMM-DD-YYYY, MMM:DD:YYYY, MMM DD YYYY, MMM/DD/YYYY, MMM.DD.YYYY, MMMDDYYYY,
MM-DD-YYYY,   MM:DD:YYYY,   MM DD YYYY,   MM/DD/YYYY,   MM.DD.YYYY,   MMDDYYYY,
MMM-DD-YY,    MMM:DD:YY,    MMM DD YY,    MMM/DD/YY,    MMM.DD.YY,    MMMDDYY,
MM-DD-YY,     MM:DD:YY,     MM DD YY,     MM/DD/YY,     MM.DD.YY,     MMDDYY,
YYYY-DD-MMM, YYYY:DD:MMM, YYYY DD MMM, YYYY/DD/MMM, YYYY.DD.MMM, YYYYDDMMM,
YYYY-DD-MM,   YYYY:DD:MM,   YYYY DD MM,   YYYY/DD/MM,   YYYY.DD.MM,   YYYYDDMM,
YY-DD-MMM,    YY:DD:MMM,    YY DD MMM,    YY/DD/MMM,    YY.DD.MMM,    YYDDMMM,
YY-DD-MM,     YY:DD:MM,     YY DD MM,     YY/DD/MM,     YY.DD.MM,     YYDDMM,
YYYY-MMM-DD, YYYY:MMM:DD, YYYY MMM DD, YYYY/MMM/DD, YYYY.MMM.DD, YYYYMMMDD,
YYYY-MM-DD,   YYYY:MM:DD,   YYYY MM DD,   YYYY/MM/DD,   YYYY.MM.DD,   YYYYMMDD,
YY-MMM-DD,    YY:MMM:DD,    YY MMM DD,    YY/MMM/DD,    YY.MMM.DD,    YYMMMDD,
YY-MM-DD,     YY:MM:DD,     YY MM DD,     YY/MM/DD,     YY.MM.DD,     YYMMDD
```

Function Sources

This is the list of values of Sources used in Datamaker Functions:

A Values from DDL

B Values from Production

C Values from Development

D Invalid Values

E Used Values from Test Data Repository

F Used Values from Data Target

G Used Values from Data Source

H Related Keys from Test Data Repository

I Related Keys from Data Target

J Related Keys from Data Source

Function Time Formats

These are the time formats accepted as function parameters in Datamaker:

```
HH-MM, THH:MMZ, HH:MM, HH MM, HH.MM, HHMM

HH-MM-SS, THH:MM:SSZ, HH:MM:SS, HH MM SS, HH.MM.SS, HHMMSS

HH-MM-SS.FFF, THH:MM:SS.FFFZ, HH:MM:SS.FFF, HH:MM:SS:FFF, HH MM SS.FFF, HH.MM.SS.FFF,
 HHMMSSFFF, HHMMSS.FFF,

HH-MM-SS.FFFFFF, THH:MM:SS.FFFFFFZ, HH:MM:SS.FFFFFF, HH:MM:SS:FFFFFF, HH MM SS.FFFFFF,
 HH.MM.SS.FFFFFF, HHMMSSFFFFFF, HHMMSS.FFFFFF,

HH-MM-SS.FFFFFFFFF, HH:MM:SS.FFFFFFFFF, HH:MM:SS:FFFFFFFFF, HH MM SS.FFFFFFFFF,
 HH.MM.SS.FFFFFFFFF, HHMMSSFFFFFFFFF, HHMMSS.FFFFFFFFF
```

Values for REPEATYPE Functions

Below is a list of Valid Values for REPEATTYPE functions and their meanings:

TEXT: Valid Text

TEXT-SPACE: Text and Spaces

TEXT-SPECIALCHAR: Special Characters and Text

SPECIALCHAR: Special Characters

SINGLESPACE (sic): Single Space

NUMERIC: Numbers Stored As Text

EMPTY: Empty

ALPHA-NUMERIC: Alphanumeric Values

ALLSPACE: All spaces

STD-ASC: Standard ASCII Values (between 32 and 127 inclusive)

NONSTD-ASC: Non-Standard ASCII Values (ASCII < 32 or > 127)

HEXCHAR: Hex Characters

Create Custom Masking Functions

If the out-of-the-box functions that FDM provides do not meet your needs, you can write your own custom masking functions. You also choose to implement custom functions if, for example, your masking logic is proprietary. FastDataMasker provides a framework that lets you write custom masking functions as Java plugins.

> **NOTE**
>
> We recommend to make backups of your custom functions and of the configuration file `custom_config.xml` . During updates or upgrades of TDM, the configuration file `custom_config.xml` is overwritten. The "`custom` " folder inside the FDM install folder is not deleted or changed.

This article contains the following procedures and example files:

Implement Custom Functions

1. Open the IDE of your choice, for example, Eclipse, and create a Java project.
2. Go to the FDM install folder in the Explorer.
   Default: `C:\Program Files\Grid-Tools\FastDataMasker`
3. Copy the file `Fastdatamasker.jar` and add it to the project's build path.
4. Create a Java class which implements the interface `com.grid_tools.products.datamasker.IMaskFunction`.
   Examples: Create class `GambianID`.
5. Override the `mask()` method:
   a. Note that the `mask()` method takes varargs:
      - `arg[0]` holds the original value that you want to mask.
      - `arg[1]` holds Parm 1
      - `arg[2]` holds Parm 2
      - `arg[3]` holds Parm 3
      - `arg[4]` holds Parm 4
   b. Modify the original value through the `mask()` method, based on your masking requirements. You can use the `args[]` parameters that you have passed.
   c. Return the masked value.
6. Build a JAR file from this source and name it.
   Example: Build `gambianid.jar`.
7. Create a folder called "custom" in the FDM install folder.
   Default: `C:\Program Files\Grid-Tools\FastDataMasker`
8. Place the `gambianid.jar` file in the "custom" folder.

Before you can use custom functions, you have to configure them.

Example File CustomFunction.java (Template)

```
package com.example;
import com.grid_tools.products.datamasker.IMaskFunction;

public class CustomFunction implements IMaskFunction {
    @Override
    public Object mask(Object... args) {
        String originalValue = (String) args[0];
        Boolean isFoo = Boolean.parseBoolean((String) args[1]);
        String maskedValue;
        if (isFoo) {
            maskedValue = originalValue.concat((String) args[2]);
        } else {
            maskedValue = originalValue.replaceAll((String) args[3], (String) args[4]);
        }
        return maskedValue;
    }
}
```

Example File GambianID.java

This example class uses two parameters, args[1] and args[2]. They correspond to parm1 and parm2.

```
package com.example;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Date;
import java.util.Random;

import com.grid_tools.products.datamasker.IMaskFunction;

public  class GambianID implements IMaskFunction {

    /**
    * In The Gambia, the National Identification Number (NIN)
    * consists of 11 digits in the form DDMMYY-PG-##CS.
    * DD MM YY indicates date of birth
    * PG indicates place of issuance and nationality
    * ## is a serial number and also indicates sex
    * CS is a check sum.
    */
    @Override
    public Object mask(Object... args) {
        // Assuming original value is a String
        // Parts of original value may be a driving factor in obtaining the masked value
        String originalValue = (String) args[0];
        // Assume date of birth format as yyyy-MM-dd
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
        // Date of Birth comes from parm 1
        Date dob = null;
        try {
            dob = sdf.parse((String) args[1]);
        } catch (ParseException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        Calendar calendar = Calendar.getInstance();
        calendar.setTime(dob);
        Integer dd = calendar.get(Calendar.DATE);
        Integer mm = calendar.get(Calendar.MONTH) + 1;
        Integer yyyy = calendar.get(Calendar.YEAR);
        // Form the DDMMYY part of the ID.  Assume that string manipulations are done.
        StringBuilder maskedValue = new StringBuilder();
        maskedValue.append(dd).append(mm).append(yyyy).append("-");
        String pg = getPlaceOfIssuanceAndNationality("Serekunda", "Mandinka");
        maskedValue.append(pg).append("-");
        // Gender comes from parm 2 and takes say, M | F | U
        String gender = (String) args[2];
        Integer nn = getNN(gender);
        Integer checksum = getChecksum(dob, pg, gender);
        maskedValue.append(nn).append(checksum);
```

```
        return maskedValue.toString();
    }
    private String getPlaceOfIssuanceAndNationality(String place, String nationality) {
        return "PG";
    }
    private static Integer getNN(String gender) {
        // Assume proprietary code to generate nn based on gender
        Random random = new Random();
        Integer nn = 0;
        nn = random.nextInt(100);
        if (0 == nn) {
                nn = 1;
        }
        if ("M".equalsIgnoreCase(gender)) {
            nn = nn % 2 != 0 ? nn : nn + 1;
            nn = nn == 99 ? nn - 2: nn;
        } else if ("F".equalsIgnoreCase(gender)) {
            nn = nn % 2 == 0 ? nn : nn + 1;
        }
        return nn;
    }
    private Integer getChecksum(Date dob, String pg, String gender) {
        // Assume proprietary code to generate checksum based on various factors
        // For this example we return 0 - 9
        Random random = new Random();
        return random.nextInt(10);
    }
}
```

Example File MaskCheckDigit.java

This example class does not use the args[1] to args[4] parameters.

```
package com.example;

import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import com.grid_tools.products.datamasker.IMaskFunction;

/* A custom function with no parameters that masks the original value and appends a check digit */
public class MaskCheckDigit implements IMaskFunction {
    @Override
    public Object mask(Object... args) {
        // Get original value from args[0]
        String originalValue = (String) args[0];
        if (null == originalValue) {
            return null;
        }
        // Proprietary validations and modifications
        if (originalValue.length() < 10) {
            return originalValue;
        }
```

```
        originalValue = originalValue.substring(0, 10);
        List<Integer> digits = new ArrayList<Integer>();
        for (int i = 0; i < originalValue.length();i++) {
            if (Character.isDigit(originalValue.charAt(i))) {
                digits.add(i, Character.getNumericValue(originalValue.charAt(i)));
            } else {
                return originalValue;
            }
        }
        // Shuffle digits in original value
        Collections.shuffle(digits);
        // More proprietary code to generate check digit based on shuffled digits
        int[] multiplier = { 9, 8, 7, 6, 5, 4, 3, 2, 1, 9 };
        if (0 == digits.get(0)) {
            digits.set(0, multiplier[0]);
        }
        int checkDigit = 0;
        for (int i = 0; i < 10; i++) {
               checkDigit += (digits.get(i) * multiplier[i]);
        }
        checkDigit /= 10;
        checkDigit %= 10;
        StringBuilder maskedValue = new StringBuilder();
        for (Integer i : digits) {
            maskedValue.append(i);
        }
        maskedValue.append(checkDigit);
        return maskedValue.toString();
    }
}
```

Configure Custom Functions

1. Go to the FDM install folder.
   Default: `C:\Program Files\Grid-Tools\FastDataMasker`
2. Modify the provided example file `custom_config.xml` .
3. Add a `<function>` section for each custom function, and configure it.
   a. Define the name of the custom function and provide a description.
      This text appears on the masking definition screen in FDM for your users.
      - **name**
      - **description**
   b. Define labels for zero to four parameters. Leave unused `<parm*>` elements empty.
      The labels appear on the masking definition screen in FDM for your users.
   – • **parm1**
      - **parm2**
      - **parm3**
      - **parm4**
   a. Define to which data format the custom function can be applied.
      Examples: If you set the `<char>` element to `true` , the function masks character data. If you set the `<date>` element to `false` , the function cannot be applied to dates.

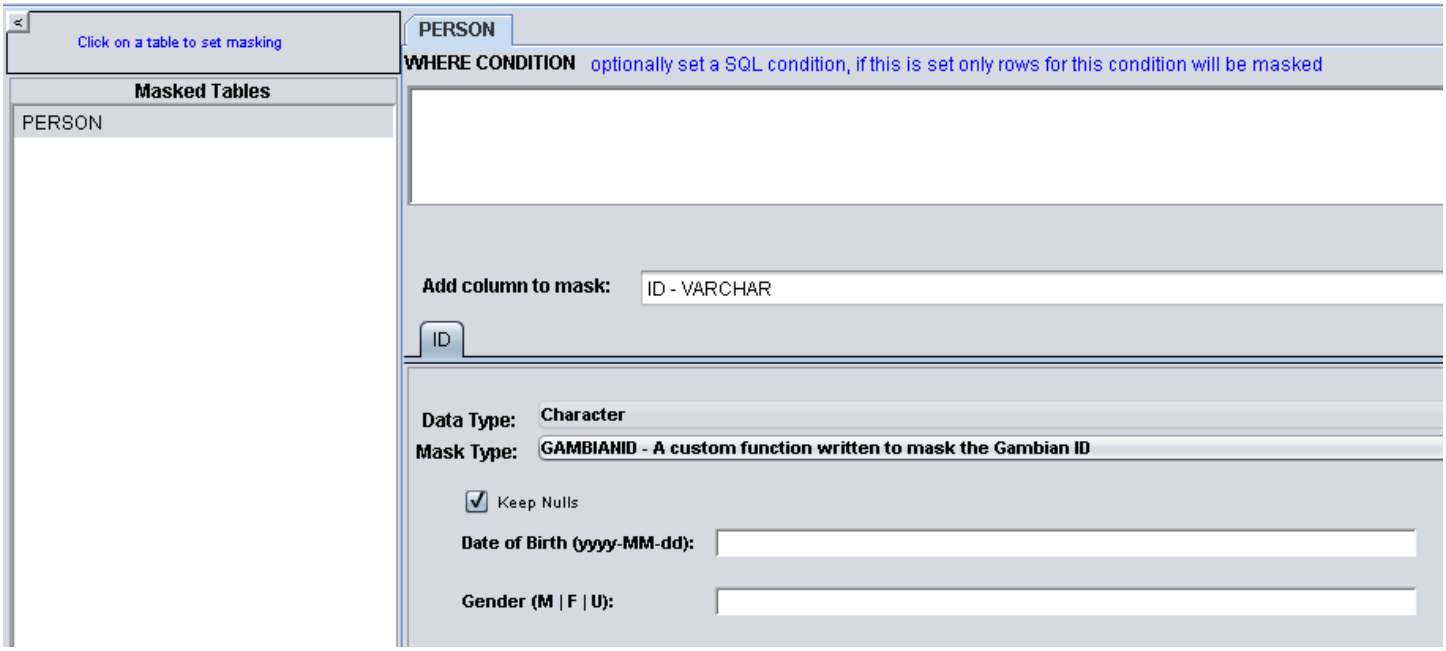
- **char**
- **number**
- **date**
- **char_date**
- **custom**

a. Provide the class path and class name of the custom function file that you placed in the "`custom` " folder.
Example: `<class_name>com.example.GambianID</class_name>`

- **class_name**

4. Save the configuration file.
5. Start FDM.

The custom functions appear on the masking definition screen in FDM. You can now use them for masking.



Example File custom_config.xml (Template)

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <functions>
        <function>
            <name>CUSTOMFUNCTION</name>
            <description>CUSTOMFUNCTION - A custom function written by the user</description>
            <parm1>Field1</parm1>
            <parm2>Field2</parm2>
            <parm3>Field3</parm3>
            <parm4>Field4</parm4>
            <char>true</char>
            <number>true</number>
            <date>false</date>
            <char_date>false</char_date>
            <custom>true</custom>
            <class_name>com.example.CustomFunction</class_name>
```

```
        </function>
    </functions>
</configuration>
```

Example File custom_config.xml with MaskCheckDigit and GambianID

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <functions>
        <function>
            <name>MASKCHECKDIGIT</name>
            <description>MASKCHECKDIGIT - A custom function with no parameters that masks the original value
 and appends a check digit.</description>
            <parm1></parm1>
            <parm2></parm2>
            <parm3></parm3>
            <parm4></parm4>
            <char>true</char>
            <number>true</number>
            <date>false</date>
            <char_date>false</char_date>
            <custom>true</custom>
            <class_name>com.example.MaskCheckDigit</class_name>
        </function>
        <function>
            <name>GAMBIANID</name>
            <description>
GAMBIANID - A custom function with two parameters that masks the Gambian ID.</description>
            <parm1>Date of Birth (yyyy-MM-dd):</parm1>
            <parm2>Gender (M | F | U):</parm2>
            <parm3></parm3>
            <parm4></parm4>
            <char>true</char>
            <number>false</number>
            <date>false</date>
            <char_date>false</char_date>
            <custom>true</custom>
            <class_name>com.example.GambianID</class_name>
        </function>
    </functions>
</configuration>
```

Masking Functions and Parameters

This page includes information about the supported masking functions and their required parameters.
Table of Contents...

ACCT_01

The ACCT_01 function substitutes digits in the original value with appropriate digits present in Parm1. Each digit in the original value represents a corresponding position in Parm1. And, the digit present at that position in Parm1 is used to replace the digit in the original value.

If Parm1 is not provided, the default value of `2749503168` is used. Regardless of the value for Parm1, the last digit in the masked number is always set to `9` .

The length of the Parm1 value must always include ten digits.

Parameters

- Parm1 (Optional)
  Specifies the digits to use for replacing the original value.

Applies to: Numeric and Character

Example: Consider the following example to understand this function:

Original value: `7564936295`

You want to mask this original value with the value provided in Parm1.

Value in Parm1: `6721843283`

The following tables shows positions of digits in Parm1:

Position	0	1	2	3	4	5	6	7	8	9
Value	6	7	2	1	8	4	3	2	8	3

To mask digit `7` in the original value with a digit from Parm1, the function identifies that the digit to be masked in the original value is `7` . Using that digit as a reference, the function finds the position `7` in Parm1. It then finds the digit present at the position `7` in Parm1. In this case, the digit at position `7` is `2` . So, the function replaces digit `7` in the original value with digit `2` . The original number now becomes **`2`**`564936295`**`9`** .

Similarly, the function takes the next digit in the original value, which is digit `5` . The function identifies the digit present at position `5` in Parm1, which is `4` . It then replaces digit `5` in the original value with digit `4` . The original number now becomes **`24`**`64936295`**`9`** .

By following the same logic, the final masked number becomes **`2438313239`** . The last digit is always set to 9 in the masked value.

The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
ACCOUNT	ACCOUNT_NO	ACCT_01	6721843283		

Note: For values that include both digits and characters (for example, `abd123xy28`), the function masks only digits. The function does not mask characters.

ADD

The ADD function adds a fixed value specified in Parm1 to the original value. The ADD function also adds the fixed value specified in Parm1 to dates in a character field.

Note: In a Microsoft SQL Server environment, the ADD function does not add the user-specified number of days to dates. For example, consider a scenario where the Hire_date column contains the value `2001-07-07` (YYYY-MM-DD format). In this case, if you use the ADD function with the value as `10` days, the function does not add `10` days to the existing date. To address this issue, download and extract the `sqljdbc_4.0.2206.100_enu.exe` file from https://www.microsoft.com/en-in/download/details.aspx?id=11774. Then, copy the `sqljdbc4.jar` file from the extracted location to the `%FDM%/lib` folder. Now, if you use the ADD function, it works as expected in your Microsoft SQL Server environment.

Parameters

- Parm1
  Specifies the fixed value to add.

Applies to: Numeric, Character, and Date

Example: The `SHIP_TO_ADDRESS_ID` column in the table `ORDERS` has `5` added to the existing value. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	KeepNull
ORDERS	SHIP_TO_ADDRESS_ID	ADD	5			N

The following are the original and masked values for this example:

- `101` to `106`
- `102` to `107`
- `103` to `108`

ADDDAYS

The ADDDAYS function adds a random number of days between 1 and the value specified in Parm1 to the existing value.

Parameters

- Parm1
  Specifies the number of days to add.

Applies to: Date and Character

Example: The `TEST_DATE` column in the table `CREDIT_CARD` adds between `1` and `14` days to the existing value. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
CREDIT_CARD	TEST_DATE	ADDDAYS	14			

The following are the original and masked values for this example:

- `1988-01-07` to `1988-01-19`
- `1982-03-01` to `1982-03-13`
- `1962-05-06` to `1962-05-18`
- `1963-01-21` to `1963-02-02`

ADDPERCENT

The ADDPERCENT function adds a fixed percentage value provided in Parm1 to the original value.

Parameters

- Parm1
  Specifies the percentage value to add.

Applies to: Numeric

Example: The `PRICE` column in the table `ORDERS` has `10%` added to the existing value. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNull
ORDERS	PRICE	ADDPERCENT	10				N

The following are the original and masked values for this example:

- `100.00` to `110.00`
- `200.00` to `220.00`
- `300.00` to `330.00`
- `400.00` to `440.00`

ADDRANDOM

The ADDRANDOM function adds a random value between Parm1 and Parm2 to the existing value.

Parameters

- Parm1
  Specifies the minimum value to use.
- Parm2
  Specifies the maximum value to use.

Applies to: Numeric

Example: The `UNIT_PRICE` column in the table `ORDER_ITEMS` has a value between `-4` and `4` added to the existing value. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNull
ORDER_ITEMS	UNIT_PRICE	ADDRANDOM	-4	4			N

The following are the original and masked values for this example:

- `112` to `114`
- `107` to `104`
- `109` to `107`
- `115` to `117`

ADDRANDOMDAYS

The ADDRANDOMDAYS function adds a random number of days between Parm1 and Parm2 to the existing value. The function does not mask the bad data. Example of bad data includes any data that does not match the supplied date format. For example, date format is `YYYYMMDD` and the data to be masked is `010101` .

Parameters

- Parm1
  Specifies the minimum value to use.
- Parm2
  Specifies the maximum value to use.

Applies to: Date

Example: The TEST_DATE column in the table CREDIT_CARD has a value between 10 and 100 days added to the existing value. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
CREDIT_CARD	TEST_DATE	ADDRANDOMDAYS	10	100		

The following are the original and masked values for this example:

- 2015-01-02 to 2015-03-25
- 2004-08-28 to 2004-11-26
- 2013-02-02 to 2013-03-25

ADDRANDOMHOURS

The ADDRANDOMHOURS function adds a random number of hours between Parm1 and Parm2 to the existing value.

Parameters

- Parm1
  Specifies the minimum value to use.
- Parm2
  Specifies the maximum value to use.

Applies to: Date

Example: The CREATION_DATE column in the table SHIPPING_OPTIONS_BASE has a value between 6 and 9 hours added to the existing value. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
SHIPPING_OPTIONS_BASE	CREATION_DATE	ADDRANDOMHOURS	6	9		

ADDRANDOMMINUTES

The ADDRANDOMMINUTES function adds a random number of minutes between Parm1 and Parm2 to the existing value.

Parameters

- Parm1
  Specifies the minimum value to use.
- Parm2
  Specifies the maximum value to use.

Applies to: Date

Example: The CREATION_DATE column in the table SHIPPING_OPTIONS_BASE has a value between 30 and 45 minutes added to the existing value. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
SHIPPING_OPTIONS_BASE	CREATION_DATE	ADDRANDOMMINUTES	30	45		

ADDRANDOMSECONDS

The ADDRANDOMSECONDS function adds a random number of seconds between Parm1 and Parm2 to the existing value.

Parameters

- Parm1
  Specifies the minimum value to use.
- Parm2
  Specifies the maximum value to use.

Applies to: Date

Example: The `CREATION_DATE` column in the table `OPTIONS_BASE` has a value between `4` and `13` seconds added to the existing value. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
OPTIONS_BASE	CREATION_DATE	ADDRANDOMSECONDS	4	13		

ADDRANDOMYEARS

The ADDRANDOMYEARS function adds a random number of years between Parm1 and Parm2 to the existing value.

Parameters

- Parm1
  Specifies the minimum value to use.
- Parm2
  Specifies the maximum value to use.

Applies to: Date and Numeric

Example: The `CREATION_DATE` column in the table `OPTIONS_BASE` has a value between `6` and `9` years added to the existing value. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
OPTIONS_BASE	CREATION_DATE	ADDRANDOMYEARS	6	9	

AES128DECRYPT

The DECRYPT function creates a decrypted version of a column based on the key in Parm1. It uses the decryption algorithm DES with key length 128.

Parameters

- Parm1
  Specifies the encryption key to use.

Applies to: Character

AES128ENCRYPT

The ENCRYPT function creates an encrypted version of a column based on the key in Parm1. It uses the encryption algorithm AES with key length 128.

Parameters

- Parm1
  Specifies the encryption key to use.

Applies to: Character

Note: The encrypted version is longer than the original value. So ensure that the column width can accommodate the new value.

AES256DECRYPT

The DECRYPT function creates a decrypted version of a column based on the key in Parm1. It uses the decryption algorithm DES with key length 256.

Parameters

- Parm1
  Specifies the encryption key to use.

Applies to: Character

AES256ENCRYPT

The ENCRYPT function creates an encrypted version of a column based on the key in Parm1. It uses the encryption algorithm AES with key length 256.

Parameters

- Parm1
  Specifies the encryption key to use.

Applies to: Character

Note: The encrypted version is longer than the original value. So ensure that the column width can accommodate the new value.

AMEXCARD

The AMEXCARD function generates a random American Express (AMEX) credit card number.

Parameters: None

Applies to: Character and Numeric

Example: Generates an AMEX number, such as `345268721090015` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
CUSTOMER	CARD	AMEXCARD			

CHARHASH

The CHARHASH function converts a character hashed value from the input. Parm1 must be set to the method MD2, MD5, SHA-1, SHA-256, SHA-384, or SHA-512.

Parameters

- Parm1
  Specifies the method (algorithm) to use.

Applies to: Character

Example: The value `COUP` in the `DISCOUNT_TYPE_CODE` column is masked to `9ccff020641cc1a68770161075cf33`. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
DISCOUNT_BASE	DISCOUNT_TYPE_CODE	CHARHASH	SHA-512		

CHECKRUT

The CHECKRUT function generates a Chilean Social Security Number (RUT) based on an existing value in another column in the table.

Parameters

- Parm1
  Specifies the name of another column in the same table that contains two or more RUT ardnumbers.

Applies to: Character

Example: The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
CUSTOMER	NUMBER	CHECKRUT	ORIGINAL		

COMBINEVALS

Note: This function appears in the Fast Data Masker UI only when Old Style Mapper is used (**Configuration**, **Old Style Mapper**).

The COMBINEVALS function combines all subsequent masking for the column.

Parameters: Parm1, Parm2, Parm3, and Parm4 (Optional)

Applies to: Character

Example: The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls	DateFor
C_BO_ELCTRNC_ADDR2	ELCTRNC_TXT	HASHLOV	EMAIL PROVIDERS	1			Y	
C_BO_ELCTRNC_ADDR2	ELCTRNC_HIDE_SRC_TXT	COMBIBEVALS					Y	
C_BO_ELCTRNC_ADDR2	ELCTRNC_HIDE_SRC_TXT	HASHLOV	FEMALENAMES	1			Y	
C_BO_ELCTRNC_ADDR2	ELCTRNC_HIDE_SRC_TXT	FIXED	-				Y	
C_BO_ELCTRNC_ADDR2	ELCTRNC_HIDE_SRC_TXT	HASHLOV	MALENAMES	1			Y	

C_BO_ELCTRNC_ADDR2	ELCTRNC_HIDE_SRC_TXT	FIXED	@					
C_BO_ELCTRNC_ADDR2	ELCTRNC_HIDE_SRC_TXT	HASHLOV	EMAIL PROVIDERS	1			Y	

CONCAT

The CONCAT function concatenates values in Parm1 to Parm4 (can be column names or literal values).

Note: Fast Data Masker tests to see if the value entered is a column in the table. If the **Use Masked Values** option is enabled, Fast Data Masker tries to use a masked value. Otherwise, it uses the current value. If the column is not found, it assumes that the value is a literal string value.

Parameters

- Parm1
  Specifies the value or column name.
- Parm2
  Specifies the value or column name.
- Parm3
  Specifies the value or column name.
- Parm4
  Specifies the value or column name.

Applies to: Character

Example: The values in the `FIRST_NAME` and `LAST_NAME` columns are concatenated in the `FULL_NAME` column (as last name, comma, space, first name). For example, `John` is the value in the `FIRST_NAME` column and `Miller` is the value in the `LAST_NAME` column. The Parm1 value is the `LAST_NAME` column, Parm2 value is a comma (`,`), Parm2 value is a space ( ), and Parm4 value is the `FIRST_NAME` column. The `FULL_NAME` column then includes the value as `Miller, John` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
EMPLOYEE	FULL_NAME	CONCAT	LAST_NAME	,		FIRST_NAME	Y

DECRYPT

The DECRYPT function creates a decrypted version of a column based on the key in Parm1.

Parameters

- Parm1
  Specifies the encryption key to use.

Applies to: Character

Example: `e34;;=` could be converted to `ABC` .

DELETE

The DELETE function deletes ALL rows from the table that contains the column on which the function runs. Use an SQL WHERE clause in Parm1 to specify what data to remove.

WARNING

If you do not specify a WHERE clause in Parm1, this function removes ALL data from the table that contains the target column!

Parameters

- (Optional) Parm1
  Specifies the WHERE clause for the delete.

Applies to: Date, Character, and Numeric

Examples:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	Result
ORDERS	CURRENT	DELETE					All entries in all rows in the table `ORDERS` are deleted.
ORDERS	CURRENT	DELETE	WHERE COST > 1000				In the table `ORDERS` , all rows in which the value of column COST is more than 1000 are deleted.

Note on use of DELETE function from TDM Portal

If you assign this function to a column from the Configure Data Masking page of the Portal, and you wish to delete only selected rows, you must use the WHERE clause in Parm1 to specify which rows to delete. Use of the WHERE clause offered by TDM Portal results in deletion of the entire table.

DOB

The DOB function adjusts the date by adding or subtracting the specified number of days from the original value without changing the age.

Parameters

- Parm1
  Specifies the number of days that you want to add or subtract.
- parm2
  (Optional) Specifies the date format used by the column being masked. This parameter is relevant when the column is of type varchar or char.
  For example: yyyy-MM-dd

Applies to: Character and Date

Example: The age `52` (DOB `9/5/1958`) remains `52` , but could be adjusted, for example, by adding `10` days to `19/5/1958` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
PERSONS	DATE_OF_BIRTH	DOB	10	yyyy-MM-dd			

DOD

The DOD function adjusts the date by adding or subtracting the specified number of days from the original value without changing the years since death.

Parameters

- Parm1
  Specifies the number of days that you want to add or subtract.
- parm2
  (Optional) Specifies the date format used by the column being masked. This parameter is relevant when the column is of type varchar or char.
  For example: yyyy-MM-dd

Applies to: Character and Date

Example: The date of death `10/9/2009` remains at `1` year, but could be adjusted by adding `10` days to it; for example, `17/9/2009` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PERSONS	DATE_OF_DEATH	DOD	10	yyyy-MM-dd		

EMAIL

The EMAIL function masks the column with an auto-generated email ID.

Parameters: None

Applies to: Character

Example: The `EMAIL` column in the table `PERSONS` is masked with an auto-generated email ID; for example, `marge756@xyz.com` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
PERSONS	EMAIL	EMAIL					

ENCRYPT

The ENCRYPT function creates an encrypted version of a column based on the key in Parm1.

Parameters

- Parm1
  Specifies the encryption key to use.

Applies to: Character

Example: `ABC` could be converted to `e34;,=`

Note: The encrypted version is longer than the original value. So ensure that the column width can accommodate the new value.

ENCRYPTJAVELIN

The ENCRYPTJAVELIN function takes plain text, or text encrypted with the TDM EncryptionUtil mechanism, and returns text encrypted with Javelin's encryption mechanism.

Parameters

- **Parm1**
  The string to encrypt with Javelin's encryption mechanism.
  – if Parm2 = `FALSE` :
  The string should be **plain text**.
  Example: `JavelinPassword`
  Result: `^_^ZVWelft+/dnJx4TtAtBdWw==^_^`
  – if Parm2 = `TRUE` :
  The string should be **encrypted** with EncryptionUtil.

  NOTE

  If you enter the whole `{cry}` prefix, it is ignored.

  Examples:
  - **Example A:** `{cry}R7fScNVu2SGsZtltywB7MKG8SUJDujzq6FxB3QKkGlAOoLQS6f5Q9L3gUxtmE1jQ`
    Result A: `^_^ZVWelft+/dnJx4TtAtBdWw==^_^`
  - **Example B:** `R7fScNVu2SGsZtltywB7MKG8SUJDujzq6FxB3QKkGlAOoLQS6f5Q9L3gUxtmE1jQ`
    Result B: `^_^ZVWelft+/dnJx4TtAtBdWw==^_^`
- **(Optional) Parm2**
  Boolean. Indicates whether to parse Parm1 as plain or encrypted text. Default: `FALSE`.

ENCRYPTUSSSN

The ENCRYPTUSSSN function consistently encrypts a nine-digit US Social Security Number. This function only encrypts the last seven digits of the number.

Parameters

- None

Applies to: Character, Numeric

Example: `789216362` to `783898720`

FILL

The FILL function fills a column with a string or character defined in Parm1, covering the entire width of the column.

Parameters

- Parm1
  Specifies the fill value to use.

Applies to: Character

Example: All characters in the column `ACCT_NUMBER` in the table `ACCOUNT` are filled with the value provided in Parm1. If `Abc` is provided as the fill value, all values in `ACCT_NUMBER` are replaced as `AbcAbcAbcAbc` , covering the complete column width. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
ACCOUNT	ACCT_NUMBER	FILL	Abc			

Note: C (Character) or N (Numeric) in Parm1 fills relevant values.

FIXED

The FIXED function masks the column values with the fixed values provided in Parm1.

Parameters

- Parm1
  Specifies the fixed value to use.

Applies to: Date, Character, and Numeric

Example: The column `ACCOUNT_NUMBER` is masked with the fixed value `11100022` and the column `PERSON_TYPE_CODE` is masked with the value `CUST` .

Note: If you want to set the value to NULL, enter the string `<NULL>` . If you want to maintain a space, enter the string `<SPACE>` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
PAYMENT_OPTIONS	ACCOUNT_NUMBER	FIXED	11100022		
PAYMENT_OPTIONS	PERSON_TYPE_CODE	FIXED	CUST		

FIXEDDAY

The FIXEDDAY function fixes the day part of the date to the value specified in Parm1.

Parameters

- Parm1
  Specifies the day value to use.

Applies to: Date

Example: The day part of the date in the `ORDER_DATE` column in the table `ORDERS` is fixed with the value `22` . If the date is `1994-01-08` , it becomes `1994-01-22` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
ORDERS	ORDER_DATE	FIXEDDAY	22		

FIXEDUNIQUE

The FIXEDUNIQUE function adds a sequence number to the value in Parm1 to give unique values for all rows.

Parameters

- Parm1
  Specifies the fixed value to which to add the sequence number.

Applies to: Numeric and Character

Example: Parm1 is the fixed part of what is generated (so `999` for numeric or `ABC` for character, for example). A sequence is then added to this to make it unique, up to the column width. Therefore, the masked values might be `99910000` or `ABC1000` , `99910001` or `ABC1001` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
MEMBERS	CARD_NUMBER	FIXEDUNIQUE	999			

MEMBERS	MEMBERSHIP_NO	FIXEDUNIQUE	ABC			

FORMATDECRYPT1

Decrypts a string encrypted using the FORMATENCRYPT1 masking function and the respective parameters used during encryption. The values of Parm1 to Parm4 must be the same as the values that were used for encryption with FORMATENCRYPT1.

Parameters

- Parm1
  (Optional) Specifies the number of start characters to ignore.
- Parm2
  (Optional) Specifies the number of end characters to ignore.
- Parm3
  (Optional) (If Parm1 and Parm2 are not set) Specifies the number of start characters to mask.
- Parm4
  (Optional) (If Parm1 and Parm2 are not set) Specifies the number of end characters to mask.
- Parm5:
  Master Key to apply to the decryption algorithm. The value of the master key must be the same as the value that was provided for encryption with FORMATENCRYPT1. This parameter is mandatory.

  IMPORTANT
  Limitation: Currently, FORMATDECRYPT1 does not support decrypting strings that were encrypted through FORMATENCRYPT1 with values for ignored and excluded characters (param6 and param7 of FORMATENCRYPT1).

FORMATENCRYPT

FORMATENCRYPT consistently masks the given column values with the original format. The function produces unique values as long as the original values are also unique, which makes it ideal for masking key columns.

Parameters

None of the following parameters are mandatory:

- PARM1
  (Optional) Specifies the number of start characters to ignore.
- PARM2
  (Optional) Specifies the number of end characters to ignore.
- PARM3
  (Optional) (If PARM1 and PARM2 are not set) Specifies the number of start characters to mask.
- PARM4
  (Optional) (If PARM1 and PARM2 are not set) Specifies the number of end characters to mask.
- PARM5
  This parameter isn't applicable. If the parameter is provided, it is ignored.
- PARM6
  (Optional) Specifies Ignored Chars, that means, which characters of the original input string will not be masked.
  Example:
  Original input: ABCDE
  Masked output: LIXUV
  Masked output with Ignored chars 'BD': L**B**X**D**V.
- PARM7

(Optional) Specifies "Excluded Chars", that means, which characters will not be present in the output string. As the excluded characters will be not present in the output map of characters, the masking algorithm will produce a completely different result from those where they are not defined.
Example:
Original input: ABCDE
Masked output: LIXUV
Masked output with Excluded chars 'IU': LJAVX

Job parameter:

To use the EXTENDED character map, set the following parameter in the JCL:

```
FORMATENCRYPTEXTENDEDCHARS=Y
```

Applies to:
Characters and Numbers.

Review the following considerations:

- For numeric columns, FORMATENCRYPT ignores the first digit of input values. This is to avoid the generation of a masked value with leading zeroes, which databases typically truncate, and which can then become identical to another value.
  This rule does not apply to character columns, because databases do not truncate character values.
- This function does not mask the first occurrence of a lowercase character. It retains that letter *as is*. For example, aBCd to `aWKj` or `BaB` to `WaJ` . To address this issue, ensure that you enter a lowercase key for the LOWERCASEKEY masking option. For example, `htjugtvffc` . Additionally, verify that the key does not start with the character `a` .

  NOTE
  The Masterkey, Ignored and Excluded chars can be set individually for each Column or Field. This means, that for each column in a table, a different Masterkey can be set, so results can vary upon need. When the Excluded characters feature is used, the masking process uses more CPU, because the output map needs to be redone for each given column referred to in the MAPCSV file.

Extended Characters Map:

À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï	ð	ñ	ò	ó	ô	õ	ö	ø	ù	ú	û	ü	ý	þ	ÿ

FORMATENCRYPT1

Consistently masks the given column values with the original format. The function produces unique values as long as the original values are also unique, which makes it ideal for masking key columns.

This function works pretty much as FORMATENCRYPT, but it has an extended set of 20 group keys against the unique set of keys from FORMATENCRYPT. Each masking group key is split in upper, lower, and numeric keys in a random sequence of characters, giving in total 60 character keys to be used in the masking process. This function also allows the users to set a particular Masterkey to be mixed with those 20 standard keys in other to obtain customized masking results. It also has the ability to work with an EXTENDED character map, listed at the end of this description.

Parameters

None of the following parameters is mandatory:

- (Optional) **Parm1**
  Specifies the number of starting characters to be ignored.
- (Optional) **Parm2**

Specifies the number of ending characters to be ignored.

- (Optional) **Parm3**
  (If PARM1 and PARM2 are not set) Specifies the starting position of the input string to be masked.
- (Optional) **Parm4**
  (If PARM1 and PARM2 are not set) Specifies the ending position of the input string to be masked.
- (Optional) **Parm5**
  A **Master Key** to apply to the encryption algorithm. Defines a user-defined Masterkey to be mixed with the set of the standard 20 masking group keys that are embedded in the product, generating a customized set of 20 masking group keys. If the Masterkey is not defined, the standard set of 20 masking group keys will be used.
  This value must meet the following criteria:
  – Must be between 1 and 20 characters long. 20 characters is the recommended length for this value. Any characters past the 20th character are ignored, and do not affect the encryption algorithm.
  – Must contain ONLY the following characters, in any combination:
    - Numbers (between 0 and 9)
    - Lower case letters (a to z)
    - Upper case letters (A to Z)
  – Examples of **valid** Master Key values: *abcd*, *SAKdsjaSFQ*, *32454GFDss*, *sad987SadSD234dsfsd6*
  – Examples of **invalid** Master Key values: *"!sa£_bcd"*, *'o$VF11'*, *S?F.Q*, *FDss*.+*, *sad987SadSD234dsfsd*^
- PARM6
  (Optional) Defines the Ignored Chars, that means, which characters of the original input string will not be masked.
  Example:
  Original input: ABCDE
  Masked output: LIXUV
  Masked output with Ignored chars 'BD': L**B**X**D**V.
- (Optional) PARM7
  Specifies Excluded Chars, that means, which characters will not be present in the output string. As the excluded characters will be not present in the output map of characters, the masking algorithm will produce a completely different result from those where they are not defined.
  Example:
  Original input: ABCDE
  Masked output: LIXUV
  Masked output with Excluded chars = 'IU': LJAVX

Job parameter:

To use the EXTENDED characters map, set the following parameter in the JCL:

```
FORMATENCRYPTEXTENDEDCHARS=Y
```

Applies to:

Characters and Numbers.

> **NOTE**
> The Masterkey, Ignored and Excluded chars can be defined individually for each Column or Field. This means, that you can set a different Masterkey for each column in a table, so results can vary upon need. Because of this, when the Excluded characters feature is used, the masking process uses more CPU because the output map needs to be redone for each given column referred to in the MAPCSV file.

Extended Characters Map:

À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï	ð	ñ	ò	ó	ô	õ	ö	ø	ù	ú	û	ü	ý	þ	ÿ

FORMATHASH

The FORMATHASH function hashes lowercase letters to lowercase letters, uppercase letters to uppercase letters, and digits to digits. All other characters remain the same. The function ensures the same format and consistent masks for the same original value, but does not ensure uniqueness.

Parameters: None

Applies to: Character

Example: The values in the `PART_ID` column are masked by using the FORMATHASH function; for example, `ABC/123-dur~678` becomes `VRT/529-cas~210` after masking. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
PARTS	PART_ID	FORMATHASH			

FORMATLUHN

The FORMATLUHN function consistently changes the current value, preserving the format (letters to letters, digits to digits). The digits are used to calculate the check digit, which then replaces the last digit in the resulting masked value. The function produces unique values as long as the original values are also unique.

Parameters:

- **Starting From Position**
  (Optional) Defines the initial character position where to start masking, counting from 1. By default it masks all characters up to the last. Can be used together with **Number of Digits to Mask**.
- **Number of Digits to Mask**
  (Optional) Defines the number of digits to mask. By default it starts at the first character. Can be used together with **Starting From Position**.
- **Number of Last Digits to Mask**
  (Optional) Defines the number of digits to mask, counting backwards from the end. The last digit will be the checksum and does not count. If **Start From Position** or **Number of Digits to Mask** are defined, **Number of Last Digits to Mask** is ignored.

Applies to: Number and Character

Example: The values in the `PART_NUMBER` column are masked by using the FORMATLUHN function; for example, `ABC/123-A1` to `VJI/802-E9` , `DEF/456-B1` to `YML/135-F4` , `GHI/123-C3` to `BPO/802-G9` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
PARTS	PART_NUMBER	FORMATLUHN			

Example: I want to format "1234567890" starting from position "3". The function returns "1297432113", where "12" remains unchanged, "3456789" is encrypted to "9743211", and the checksum is calculated as 3 and replaces the last digit. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
STAFF	ID_NUMBER	FORMATLUHN	3		

Example: I want to format "2" digits of "1234567890". The function returns "6834567894", where "3456789" remains unchanged, the first 2 digits "12" are encrypted to 68, and the checksum is calculated as 4 and replaces the last digit. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
STAFF	ID_NUMBER	FORMATLUHN		2	

Example: I want to format the last "2" digits of "1234567890". The function returns "1234567431", where "1234567" remains unchanged, "89" is encrypted to "43", and the checksum is calculated as 1 and replaces the last digit. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
STAFF	ID_NUMBER	FORMATLUHN			2

FORMATMASK

The FORMATMASK function masks a character value, retaining the original format. Only characters A through Z, characters a through z, and digits 0 through 9 are masked.

Parameters

- Parm1
  Specifies the mask key; for example, 345.

Applies to: Character and Numeric

Example: Hash key `345` produces a masked output; for example, `Aa999` becomes `sK110` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2
PURCHASE	ORDER_NUMBER	FORMATMASK	345	

FORMATVIN

This function validates whether the input is a valid Vehicle Identification Number.

- If the input does not have 17 characters, the function leaves it as is.
- If the input has 17 characters and is a valid VIN, the function leaves it as is.
- If the input has 17 characters but is not a valid VIN, it changes the check digit to make it valid.

Parameters: None

Applies to: Characters

Example: The values in the `VEH_ID` column are masked by using FORMATVIN; for example, the valid `JT5RA65K`**`2`**`F4054074` stays `JT5RA65K`**`2`**`F4054074` , and JT5RA65K**2**F4054070 becomes JT5RA65K**5**F4054070 (invalid, check digit corrected).

The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
VEH	VEH_ID	FORMATVIN			

GENCARD

The GENCARD function masks the column values with the 15- and 16-digit credit card numbers, keeping the original format.

Parameters: None

Applies to: Character and Numeric

Example: The `CARD_NUMBER` column masks existing values as `5187230394132622` , `376152764010075` , `4611647914049615` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
CREDIT_CARDS	CARD_NUMBER	GENCARD				

GUID

The GUID function generates a globally unique identifier, which is a 36-character value (including hyphens).

Parameters:

- **Parm1**If this parameter contains the character string 'COLLAPSE', the unique identifier is without hyphens. This parameter is case-sensitive.

Applies to: Character

Example: The `SECTION_ID` column masks the existing value as `94b82ed6-e941-4185-b84c-53cc0f56a006` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	Result
STUDENT	SECTION_ID	GUID					`94b82ed6-e941-4185-b84c-53cc0f56a006`
STUDENT	SECTION_ID	GUID	COLLAPSE				`94b82ed6e9414185b`
STUDENT	SECTION_ID	GUID	Collapse				`94b82ed6-e941-4185-b84c-53cc0f56a006`

HASH

The HASH function returns HASH values for the integer fields.

Parameters

- Parm1
  Specifies the seed value for hash.
- Parm2 (Optional)
  Specifies the maximum number of digits allowed.

Applies to: Number

Example: The `NUMBERS` column in the table `TEST1` is hashed with the seed value of `35` and `5` as the maximum number of digits allowed. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
TEST1	NUMBERS	HASH	35	5		

The following are the original and masked values for this example:

- 101 to 10312
- 102 to 73780
- 103 to 88091

HASHABN

The HASHABN function hashes an Australian Business Number with valid check digits.

Parameters: None

Applies to: Character

Example: The value 51824753556 in the VALID_ABN column is masked to 19503876902 . The following table shows the usage:

Table	Column	Function	Parm1	Parm2
PERSON	VALID_ABN	HASHABN		

HASHACN

The HASHACN function hashes an Australian Company Number with valid check digits.

Applies to: Character

Parameters: None

Example: The values 786362338 , 784862335 , and 789123468 in the VALID_ACN column are masked to 465485787 , 463985784 , and 468246817 respectively. The following table shows the usage:

Table	Column	Function	Parm1
PERSON	VALID_ACN	HASHACN	

HASHCARD

The HASHCARD function retains the first two digits of the original credit card number (which define the credit card type) and hashes the remaining digits, retaining the original length of the number. This function also ensures that the last digit is the correct check digit for a credit card number.

Note: Use this function to mask a number that is part of a key field. If the original number is unique, then it will continue to be unique after the HASHCARD function.

Parameters: None

Applies to: Character and Numeric

Example: The value 5533716111165678 in the CARD_NUMBER column is masked to 5599197827275710 . Note that the first two digits remain the same. The following table shows the usage:

Table	Column	Function	Parm1
EMPLOYEE	CARD_NUMBER	HASHCARD	

HASHCARD1

Hash a credit card number using FORMATENCRYPT1. This function masks digits only. If the data to be masked contains characters and digits, use the start and end position parameters to exclude non-numeric characters from masking. If the input value does not have a valid length for a credit card number, the number will be encrypted and its length will remain invalid. If the input value is null, 0, or 0000000000000000, the value will not be changed.

Applies to: Character and Numeric

- **Parm1 - Card Start Position** (Optional): Position where the credit card number starts in the String. Default value is 1.
  Example: ABC1234567890123456. In this example, Card Start Position is 4.
- **Parm2 - Card End Position** (Optional): Position where the credit card number ends in the String. Default value is 16.
  Example: 1234567890123456ABC. In this example, Card End Position is 17.
- **Parm3 - Mask Start Position** (Optional): Position in the credit card number where to start masking. Default value is 1.
  Example: 1234567890123456. In this example, Mask Start Position is 4.
- **Parm4 - Mask End Position** (Optional): Position in the credit card number where to stop masking. Default value is 16.
  Example: 1234567890123456. In this example, Mask End Position is 14.
- **Parm5 - Master Key** (Optional): Custom user key that will be mixed with the internal key set as it is defined for FORMATENCRYPT1. For more information, see FORMATENCRYPT1.
- **Parm6 - Custom End Digits** (Optional): A sequence of digits in the end of the credit card number that will not be masked. The check digit will not be updated.
  Example: 1234567890123000. In this example, Custom End Digits is 00.

The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	Parm5	Parm6
EMPLOYEE	CARD_NUMBER	HASHCARD1	4	13	3	6		

In this example, we are hashing the alphanumeric string `ABC123456789DEF` . We use Start and End Position parameters to exclude the first 3 characters and the last 3 characters, the numeric substring is now 123456789. Then we use the Start and End Masking Position parameters to mask the substring 3456 only. In the end result, the xxxx positions is replaced by the hashed value and y is replaced by the check digit: `ABC12xxxx78yDEF` . This example does not use the key and custom end digits.

HASHCARD4

The HASHCARD4 function hashes only the last four digits of the original credit card number (which define the credit card type) while retaining the original length of the number. The function does not mask the bad data. Examples of bad data include data containing fewer than thirteen digits or any length of character type data. For example, `1234` , `54365` , `abcdefghijklmnopqrstuvw` , `abcdef` , `teresfdgertygd` .

Parameters: None

Applies to: Character and Numeric

Example: The value `3074067779211613` (before using the function) in the `CARD_NUMBER` column is masked to `3074067779211571` (after using the function). Note that only the last four digits are masked. The following table shows the usage:

Table	Column	Function	Parm1
EMPLOYEE	CARD_NUMBER	HASHCARD4	

HASHDAYS

The HASHDAYS function takes an existing date and consistently changes only the day part of it to a value between 1 and 27. If the date is stored as a string, specify the date format.

Example: The days of dates in the column HIRE_DATE are consistently masked, from 2018-07-24 to 2018-07-01 , and from 1997-12-16 to 1997-12-27 .

Table	Column	Function	Parm1
EMPLOYEE	HIRE_DATE	HASHDAYS	YYYY-MM-DD

HASHDOB

The HASHDOB function hashes a date, keeping the original age.

Parameters: None

Applies to: Date and Character

Example: Date of birth is masked by hashing an existing date of birth. For example, the date in the column BIRTH_DATE is consistently masked (retaining the age) 2016-05-24 to 2016-05-28 , 1999-12-16 to 1999-12-26 . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
EMPLOYEE	BIRTH_DATE	HASHDOB			

HASHFINNISHID

Takes an existing Finnish ID and hashes it to a new value. If the original values are unique, the hashed values are also unique.

Parameters: None

Applies to: Character

Examples:

- 010120-029J to 010120-708E
- 010120-001M to 010120-780S
- 010120-002N to 010120-781T
- 010120-003P to 010120-782U

HASHIBAN

Takes an existing IBAN number and consistently changes it to a new IBAN number. The new number has valid IBAN check digits.
Invalid input numbers, for example those not starting with a valid country code, or having an incorrect length, remain unchanged.

Parameters: None

Applies to: Character

Example: GB82WEST12345698765432

HASHLOV

The HASHLOV function hashes the current value to consistently pick a value from a seed list or table in Parm1. For XML files, you can choose which value of an XML tag to hash on, for more information, see Fast Data Masker Best Practices.

Parameters

- Parm1
  Specifies the seed list or table.
- Parm2
  If using a seed table from a database rather than a file. This is the optional column value from the seed table. So, for example, `3` would return the value for `rd_ref_value3` . If linking columns using seed files, you would use the following naming convention: `address.1.txt` , `address.2.txt` .
- Parm3 (Optional)
  Typically, HASHLOV hashes the value of the column to be masked. If you want the hash to work on a different column, enter that column name here. This would typically be a column with unique values in it to minimize duplication.
- Parm4 (Optional)
  Specifies the seed value for hash.

Note: Set the **MD5HASHLOV** masking option in the **Options** tab to **Y** to use `MD5` hashing with the HASHLOV function. By default, the function uses a Java hash algorithm.

Applies to: Character, Numeric, and Date

Example: The current value in the `FIRST_NAME` column in the table `PERSONS` is hashed to consistently pick a value from the seed list `female_english.txt` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
PERSONS	FIRST_NAME	HASHLOV	female_english.txt		

HASHLOV1

The HASHLOV1 function hashes the current value to consistently pick a value from a database-oriented seed list in Parm3.

Parameters

- Parm1
  Represents the data category in the seed data table.
- Parm2
  Represents the column number in the seed data table that is used for the masking.
- Parm3
  Specifies the column value to define the list. For example, if column name=`CITY` and current value is `New York` , then only `New York` addresses are returned.
- Parm4
  Specifies an integer value describing the maximum length of the value to use from Parm3. So, for example, if Parm3 is a postcode with the current value as `OX29 4TP` and you set Parm4 to `4` , your list then contains all addresses starting with `OX29` .

Masking Option

- **(MS SQL only)** FASTSEEDFETCH
  Uses an improved algorithm to load seed tables.
  Value: Y or N (default N)

Note: Override SQL is used to get the hash integer value rather than the column you are masking. This is important when linking columns to choose a column with the most unique values.

Applies to: Character, Numeric, and Date

Example: This example consistently masks the column `CITY` in the table `Employee_Address` .

It takes the value for the column defined in "Override SQL" (`ADDRESS_ID`) and uses this to get a hashed index. The seed list is from the data category `US CITY STATE ZIP COUNTY` and for the `STATE_PROVINCE` values for the current state (so you have different buckets for `CA` , `OH` , and so on).

You use the value from `STATE_PROVINCE` to get the hashed value rather than the column you are masking to get a better spread of data. Otherwise, for example, all the addresses for the state `CA` would have the same value. As in RANDLOV1, the value for `RD_ REF_VALUE` must match that of the reference column (in this case, `STATE_PROVINCE`).

Notes:

- To provide default values, in this case for postal codes that exist in the table to be masked (`EMPLOYEE_ADDRESS`) but not in the seed data category (`US CITY STATE ZIP COUNTY`), you would need to add a line in the seed data table for that category with default values (`RD_REF_VALUE 'DEFAULT'`).
- Set the **MD5HASHLOV** masking option in the **Options** tab to **Y** to use `MD5` hashing with the HASHLOV1 function. By default, the function uses a Java hash method.

The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls	DateFormat	Cross Reference	Override SQL
EMPLOYEE_ADDRESS	CITY	HASHLOV1	US CITY STATE ZIP COUNTY	2	STATE_PROVINCE		Y			ADDRESS_ID

HASHPASSPORT

The HASHPASSPORT function consistently masks a passport number, keeping the original length.

Parameters: None

Applies to: Character and Numeric

Example: The passport number `359866285` in the `PASSPORT` column is masked to `282777060` . The following tables shows the usage:

Table	Column	Function	Parm1	Parm2
EMPLOYEE	PASSPORT	HASHPASSPORT		

HASHPHONE4

The HASHPHONE4 function masks the last four digits of a phone number. The remaining digits are not changed. The function does not mask the bad data. Examples of bad data include data containing fewer than four digits or characters of any length. For example, `554` , `298` , `avd` , `abcf` , or `adfere` .

Parameters: None

Applies to: Character and Numeric

Example: The phone number `1712441`**`9639`** in the `PHONE` column is masked to `1712441`**`7572`** . Note that only the last four digits are masked. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PEOPLE	PHONE	HASHPHONE4				

HASHRUT

The HASHRUT function takes an existing RUT number (Chilean Social Security number) and hashes the first 8 digits, then adds the appropriate check digit at the end. This is the method that should be used to ensure consistency across tables.

Note: The string length of a RUT is 9, so HASHRUT only works on string columns.

Parameters: None

Applies to: Numeric and Character

Example: The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PEOPLE	RUT_NUMBER	HASHRUT				

HASHSIN

The HASHSIN function masks the given column values with a unique Canadian Social Insurance Number, which is a 9-digit number.

The input string must be a 9-digit numeric value, which can include any number of non-numeric separators. If the value includes non-numeric separators, HASHSIN includes these separators in the output value.

> **WARNING**
>
> If the input entry contains more or fewer than 9 numbers, HASHSIN does not mask the value.

Parameters: None

Applies to: Numeric (input SIN) and Character (separators)

Example: The following table shows examples of HASHSIN's operation:

Input string	Output string
123456789	846812368
123-456-789	846-812-368
123**##**456**RR**789	846**##**812**RR**368
0123456789	0123456789 (no masking, more than 9 numbers)
1234#5678	1234#5678 (no masking, fewer than 9 numbers)

HASHSPANISHID

The HASHSPANISHID function consistently masks Spanish ID numbers (NIF or NIE). You can also use this function to mask Spanish Company IDs (CIF numbers).

Parameters:

- **Parm1**

If parm1 = Y, the function will also mask CIF numbers.

Applies to: Character

HASHTIN

The HASHTIN function masks the given column values with a unique US Tax Identification Number, which is a 9-digit number starting with 9.

Parameters: None

Applies to: Character and Numeric (having integer values)

Example: The following values in the `TIN_ID` column are masked appropriately:

- `12323232435` to `985861557`
- `12323232436` to `944772017`
- `12323232437` to `900772597`

The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
PEOPLE	TIN_ID	HASHTIN			

HASHTURKISHID

The HASHTURKISHID function masks the 11-digit Turkish Identification Number.

Parameters: None

Applies to: Numeric and Character

Example: The value `12345678901` in the column `TURKISH_ID` is masked to `80257913544` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
PEOPLE	TURKISH_ID	HASHTURKISHID			

HASHTURKISHTAXID

The HASHTURKISHTAXID function masks an existing 10-digit Turkish Tax Identification Number.

Parameters: None

Applies to: Numeric, Character

HASHUSSSN

The HASHUSSSN function consistently hashes a US Social Security Number, retaining the original length of the number.

Parameters: None

Applies to: Character and Numeric

Example: The value `198689580` in the `USSSN_ID` column is masked to `228322064` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
PEOPLE	USSSN_ID	HASHUSSSN			

HASHUSSSN4

The HASHUSSSN4 function masks a US Social Security Number's last four digits, retaining the remaining numbers and original length.

Parameters: None

Applies to: Character and number

Example: The value `58019`**`8689`** in the `USSSN_ID` column is masked to `58019`**`1232`** . Note that only the last four digits are masked. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
PEOPLE	USSSN_ID	HASHUSSSN4			

IGNORE

The IGNORE function ignores the mask and retains the value if no cross-reference or default value can be found.

Parameters

- Parm1 (Optional)
  Specifies the cross reference.
- Parm2 (Optional)
  Contains the default value.

Applies to: Character, Numeric, and Date

Example: If Parm1 is absent, IGNORE reverts to using the default value set in Parm2. If Parm2 is also absent, IGNORE masks and uses the existing value. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PERSONS	FIRST_NAME	IGNORE				

INTRANGE

The INTRANGE function masks the column with a random value between Parm1 and Parm2.

Parameters

- Parm1
  Contains the start value of the integer.
- Parm2
  Contains the end value.

Note: The maximum value for Parm2 is `2147483647` . If the database accepts decimal values, you can also use the NUMERICRANGE masking functions.

Applies to: Character and Numeric

Example: The values in the column `NUM_CARDS` are replaced with integer values between `100` and `110` ; for example, `20987 to 103` , `34572 to 105` , and so on. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
CARDS	NUM_CARDS	INTRANGE	100	110	

LUHN

The LUHN function generates a number of a given length, with the correct Luhn algorithm check digit at the end.

> **NOTE**
>
> The Luhn algorithm is designed to detect and prevent accidental errors, and detect valid numbers from random numeric strings. It is not intended as a secure hash.

Parameters

- Parm1
  Specifies the length of the number to generate.

Applies to: Character and Numeric

Example: The `NUMBER` column uses the LUHN function to mask the existing values with new values of length 5; for example, `103` to `98715` , `110` to `91702` , and so on. The following table shows the usage:

Table	Column	Function	Parm1
EMPLOYEE	NUMBER	LUHN	5

MASKBELGIANID

The MASKBELGIANID function masks a Belgian National Identification Number based on parameters provided. If the 'Use Masked Values' check box is selected, FDM uses masked values in the current masking routine.

Parameters:

- Parm1 — Date of Birth
  Specifies date of birth.
  - RANDOM
  - 1999-01-02 — A literal date value. Provide a date format.
  - DOB_COLUMN — The Name of the column containing the date of birth. Provide a date format.
- Parm2 — Gender
  Specifies the gender.
  - RANDOM
  - M
  - F
  - GENDER_COLUMN — Name of the column containing the gender values M or F.

Applies To: Numeric, Character

Example:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PERSONS	BELGIAN_ID_COLUMN	MASKBELGIANID	DOB_COL	G_COL		

MASTERCARD

The MASTERCARD function generates a random Mastercard credit card number.

Parameters: None

Applies to: Character and Numeric

Example: Generates a valid Mastercard number in the CARD column, such as `5532800794260091`, `5185633632025254`, `5148072171231971`, and so on. The following table shows the usage:

Table	Column	Function	Parm1	Parm2
PEOPLE	CARD	MASTERCARD		

MOVETOKEN

Moves a token in a string from first to last position, or from last to first position. Tokens are separated by a delimiter of your choice.

Parameters:

- Parm1
  Specify Y to move the first token to the last position. Specify N to move the last token to the first position.
- Parm2
  Defines the delimiter that separates tokens. Defaults to space. Enter a comma here if the data is in comma-separated format (CSV).

Applies to: Character

Example:

When the delimiter is the space character, and you move the end token to start, then "Micah, Albrecht Harry" becomes "Harry Micah, Albrecht". If you move the start token to end, "Micah, Albrecht Harry" becomes "Albrecht Harry Micah,".

Table	Column	Function	Parm1	Parm2
PEOPLE	NAME	MOVETOKEN	N	

NEXTVAL

The NEXTVAL function finds the next value from an Oracle sequence. If it is the first time the sequence is used, it starts at 1. The value is then incremented by 1 for each subsequent sequence.

Note: You must have an Oracle XREF connection set if you use NEXTVAL.

Parameters

- Parm1
  Specifies the name of the sequence.

Applies to: Numeric

Example: An Oracle sequence "FirstSequence" is called and used to update 20,000 fields in one run. When the sequence is called next time, the run starts from 20,001.

NINO

The NINO function generates a random UK National Insurance Number.

Parameters

- Parm1 (Optional)
  Specifies the separator character.

Applies to: Character

Example: The NUMBER_UK column uses the NINO function to generate a value such as `NB-00-67-21-B` if using (-) as the separator character. The following table shows the usage:

Table	Column	Function	Parm1	Parm2
PEOPLE	NUMBER_UK	NINO		

NUMERICRANGE

The NUMERICRANGE function masks the column with numeric values between Parm1 and Parm2. Use this function to generate values with a decimal, unlike range, which uses whole numbers.

Parameters

- Parm1
  Specifies the start value of the range.
- Parm2
  Specifies the end value of the range.

Applies to: Numeric

Example: The column `ORDER_TOTAL` is replaced with values between `40.01` and `49.99`. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
ORDERS	ORDER_TOTAL	NUMERICRANGE	40.01	49.99			N

NUMHASH

The NUMHASH function hashes a numeric value in a character column as digits.

Parameters: Parm1, Parm2 (Optional), Parm3 (Optional)

Note: Parm2 is the maximum length of the data, Parm3 is the minimum length.

Applies to: Character

Example: Numeric value in a character column is hashed by the seed value to create a numeric string of length between the values in Parm2 and Parm3.

OR

The OR function, in conjunction with WHERE, lets you restrict your mask to certain rows. For example, you can use separate masking rules from credit cards/direct debit expiry dates to invoices based on the PAYMENT_TYPE_CODE column.

Notes

- The ADD function must be used on a separate row to the WHERE clause.
- The OR function is for masking flat files only, as there is no SQL file.

Parameters

- Parm1
  Specifies the SQL WHERE clause.

Applies to: Character, Numeric, and Date

Example: The following table shows the usage:

Table	Column	Function	Parm1	Parm2
PERSONS		WHERE	PAYMENT_TYPE_CODE LIKE 'CC'	
PERSONS		OR	PAYMENT_TYPE_CODE LIKE 'DD'	
PERSONS	EXPIRE_DATE	ADDRANDOMDAYS	-7	21

PARTMASK

The PARTMASK function masks the existing value, replacing only alphabets (if Parm1 is set to "C") or only numerics (if Parm1 is set to "N"). The case of alphabets is retained.

Note: Individual characters are replaced by a randomly selected character.

Parameters

- Parm1
  Specifies whether to replace alphabets (C) or digits (N).

Applies to: Character

Example: Numerics are masked in the `POSTCODE` column of the `ADDRESS` table. For example, the postcode `OX29 4TP` becomes `OX34 8TP` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
ADDRESS	POSTCODE	PARTMASK	N			

PHONE_01

The PHONE_01 function replaces digits 0 through 9 with digits in Parm1 or fixed replacement value.

Parameters

- Parm1 (Optional)
  Specifies the digits to use for replacement.

Applies to: Character and Numeric

Example: Each value in the number `0123456789` is masked with a corresponding value from the number `6721843283` , so 1 = 6, 2 = 7, 3 = 2, and so on. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PERSONS	PHONE_NUMBER	PHONE_01	6721843283			

POSITIONMASK

The POSITIONMASK function masks a value based on positional rules, that you define in Parm1. Separate each rule with a hyphen (no spaces).

Parameters:

- Parm1
  Specifies the rules for each position in the output value, from the following formula:
  - RD*nnn*L: Rando*nnn*m digit at position *nnn* from left.
  - RD*nnn*R: Random digit at position *nnn* from right.
  - RA*nnn*L: Random alphabetic character at position *nnn* from left.
  - RA*nnn*R: Random alphabetic character at position *nnn* from right.
  - RC*nnn*L: Random alphanumeric character at position *nnn* from left.
  - RC*nnn*R: Random alphanumeric character at position *nnn* from right.
  - F#*nnn*L: Fixed digit (#) at position *nnn* from left.
  - F#*nnn*R: Fixed digit (#) at position *nnn* from right.
  - Fa*nnn*L: Fixed alphabetic character (*a*) at position *nnn* from left.
  - F*a nnn*R: Fixed alphabetic character (*a*) at position *nnn* from right.

Notes

- For any position for which you do not provide a rule, the original value remains as the output value.
- If the old value is null or all blanks, the function skips the row.

Applies to: Character

Example: The function masks the first three characters of each value of the `PHONE_NUMBER` column in the table `PEOPLE` , with the fixed value `9`. The rest of the digits remain as their original values. The following table shows the usage:

Table	Column	Function	Parm1
PEOPLE	PHONE_NUMBER	POSITIONMARK	F9001L-F9002L-F9003L

Therefore, the resultant masked value is `999XXXXXX` , where `X` is the existing value.

For example, `1235553283` becomes `9995553283` , and `9238974398` becomes `9998974398` .

RANDEIN

The RANDEIN function masks an existing value with a randomly generated US Employer ID Number.

Parameters:

- Parm1
  Specifies the separator character.

Applies to: Character

Example: The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
COMPANY	EMPLOYER_NUMBER	RANDEIN	-		

RANDHIC

The RANDHIC function masks an existing value with a randomly generated US Health Insurance Claim Number.

Parameter:

- Parm1
  Specifies the separator character.

Applies to: Character

Example: The following table shows the usage:

Table	Column	Function	Parm1	Parm2
EMPLOYEE	NUMBER	RANDHIC	-	

RANDLOV

The RANDLOV function masks the column values with randomly selected values from the seed file.

Note: The data for RANDLOV function can also be drawn from database tables. Select the columns you want to use and place them in order in the mapping file (for example, `RD_REF_VALUE2` , `RD_REF_VALUE3`). Note that columns drawn from the database tables do not contain a .txt suffix.

Parameters

- Parm1
  Specifies the seed file name.

Applies to: Character, Numeric, and Date

Example: The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
ADDRESS	CITY	RANDLOV	uktowns.txt			
ADDRESS	STATE_PROVINCE	RANDLOV	ukpostcode3.txt			
PERSONS	LAST_NAME	RANDLOV	lastnameindian.txt			

RANDLOV1

The RANDLOV1 function generates random addresses, cities, states/provinces, and so on that are valid for the value specified in Parm3 from a seed table.

Parameters

- Parm1

Specifies the seed file name.

- Parm2
  Specifies the position of the column in gtsrc_reference_data.
- Parm3
  Specifies the column in the table from where to get the seed data.
- Parm4 (Optional)
  Specifies the maximum length to test for Parm3. For example, if the postcode is `OX29 4TP` and Parm4 is set as `4` , this function looks only for `OX29` .

Applies to: Character, Numeric, and Date

Example: In the example table, PARM3 is the column `POSTAL_CODE` . This column is used to reference the `rd_ref` values `3` , `5` , and `4` from the `gtrsc_reference` data seed table. The reference is done by using `rd_ref_id` of `US_ADDRESSES` and the postal code stored in `rd_ref` value.

Note: To provide default values, in this case for postal codes that exist in the table to be masked (`ADDRESSES`) but not in the seed data category (`US_ADDRESS`), add a line to the seed data table for that category with default values (`RD_REF_VALUE 'DEFAULT'`).

The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
ADDRESSES	ADDRESS2	RANDLOV1	US_ADDRESS	3	POSTAL_CODE	73 Main St
ADDRESSES	CITY	RANDLOV1	US_ADDRESS	5	POSTAL_CODE	New York
ADDRESSES	STATE_PROVINCE	RANDLOV1	US_ADDRESS	4	POSTAL_CODE	New York

RANDOM

The RANDOM function masks the column with random values between Parm1 and Parm2.

Parameters

- Parm1
  Specifies the start value of the range.
- Parm2
  Specifies the end value of the range.

Note: Provide Parm1 and Parm2 in the format YYYYMMDD for dates.

Applies to: Character, Numeric, and Date

Example: The value in the `ORDER_DATE` column is replaced with a random date between 2001-Sept-18 to 2002-Nov-15. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
ORDERS	ORDER_DATE	RANDOM	20010918	20021115			N

RANDOMBLOB

The RANDOMBLOB function randomly takes BLOB data from a file in the BLOBS sub-directory and loads into this BLOB column. This must contain files of type of data you want to mask for example .pdf or .gif.

Parameters: None

Applies to: Character

Example: The following table shows the usage:

Table	Column	Function	Parm1	Parm2
ORDERS	ORDER_TYPE	RANDOMBLOB		

RANDOMDATE

The RANDOMDATE function replaces an existing date value with a random value between Parm1 and Parm2.

Note: You cannot use RANDOMDATE on a DATE type column.

Parameters

- Parm1
  Specifies the minimum date value.
- Parm2
  Specifies the maximum date value.

Applies to: Character

Example: The date values in the `ORDER_DATE` column are replaced with a random date between 18-Sept-2001 and 15-Nov-2002. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
ORDERS	ORDER_DATE	RANDOMDATE	20010918	20021115			N

RANDOMDAYS

The RANDOMDAYS function changes the day part of a date to a random value.

Parameters: None

Applies to: Date and Character

Example: The day value in the `ORDER_DATE` column is masked to a random value (for example, `2001-01-22` to `2001-01-18`). The following table shows the usage:

Table	Column	Function	Parm1	Parm2
ORDERS	ORDER_DATE	RANDOMDAYS		

RANDOMTXT

The RANDOMTXT function replaces the column with random text.

Parameters

- Parm1
  Specifies the minimum length of the text.
- Parm2
  Specifies the maximum length of the text.
- Parm3
  Specifies that entering U returns all uppercase letters and L returns all lowercase letters.

Applies to: Character

Example: The `FIRST_NAME` column has the value `XrzFF` . The length of the text string is between `3` and `12` ; the case is set to uppercase. In this case, the value changes to `OELQ` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
PERSONS	FIRST_NAME	RANDOMTXT	3	12	U		N

RANDSSN

The RANDSSN function masks the column with a randomly generated US Social Security Number.

Parameters

- Parm1 (Optional)
  Acts a separator for the SSN.

Applies to: Character and Numeric

Example: The value in the column `ID` is replaced with a random US Social Security Number. Entering a separator character into Parm1 (that is, *), as in the example below, generates a social security number like `987*65*4320` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
PERSONS	ID	RANDSSN	*				N

REFLOV

The REFLOV function uses the numeric value from the specified column to consistently pick a value from a seed list or table in Parm1.

Parameters

- Parm1
  Specifies the seed list or table.
- Parm2
  If using a seed table from a database rather than a file. This is the optional column value from the seed table. So, for example, 3 would return the value for rd_ref_value3. If linking columns using seed files, you would use the following naming convention: address.1.txt, address.2.txt.
- Parm3
  The numeric column used for getting value used to get the seed value. If the numeric value is larger than the seed list then integer division is used so that the resulting value is within the seed list size. The masking option USEMASKEDVALUES can be enabled to use the masked value from the numeric column.

Applies to: Character, Numeric, and Date

Example: The numeric value in the NAME_INDEX column in the table PERSONS is used to consistently pick a value from the seed list female_english for the column FIRST_NAME at the same table. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
PERSONS	FIRST_NAME	REFLOV	female_english	NAME_INDEX	

REGEXPREPLACE

The REGEXPREPLACE function searches the column values for the regular expression mentioned in Parm1 and replaces it with the character pattern mentioned in Parm2. The REGEXPREPLACE operation is case-sensitive.

Parameters

- Parm1
  Specifies the regular expression you want to search for in the column.
- Parm2
  Specifies the character pattern you want to use for replacing the expression.

Applies to: Character and numeric

Example: All the first names that match the regular expression `Fir.*` are replaced with a value `John` in the column `FIRST_NAME` . The regular expression `Fir.*` searches for all the names in the `FIRST_NAME` column where the first three characters of the names are `Fir` . For all such first names, the value `John` is used. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
PERSONS	FIRST_NAME	REGEXREPLACE	Fir.*	John			Y

Note: In the case of a flat file, the Table column remains empty.

REGEXPSUBSTR

The REGEXPSUBSTR function returns a sub-string that matches the specified regular expression.

Parameters

- Parm1
  Specifies the regular expression that you want to search for in the column.
- Parm2
  Specifies the start position.
- Parm3
- Parm4

Applies to: Character

Example: The regular expression `[a-j].*` (with the start position as `1`) searches all the names in the `FIRST_NAME` column where the first character includes a letter from `a` through j. The function then extracts the sub-strings from all the names that match the given pattern. For example, if the `FIRST_NAME` column includes a string `Jhonson` , the REGEXPSUBSTR function in this case extracts the sub-string as `honson` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
PERSONS	FIRST_NAME	REGEXSUBSTR	[a-j].*	1			Y

Note: In the case of a flat file, the Table column remains empty.

REPLACE

The REPLACE function searches the column values for the character pattern specified in Parm1 and replaces it with the character pattern specified in Parm2. The replace operation is case-sensitive.

Parameters

- Parm1
  Specifies the character pattern that you want to search for in the column.
- Parm2
  Specifies the character pattern that you want to use for replacing the searched pattern. If Parm2 is absent, Parm1 is the name of a CSV file that contains a list of values to be replaced. Place the CSV file in the same directory as gtfdm.exe.

Applies to: Character

Example 1: Case 1 (When Parm2 is present)

When the pattern `Ab` is found in the column `ADDRESS_1` , it is replaced with `23` . Similarly, if `a` is found in the column `ADDRESS_2` , is also replaced by `23` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PERSONS	ADDRESS_1	REPLACE	Ab	23		
PERSONS	ADDRESS_2	REPLACE	a	23		

Example 2: Case 2 (When Parm2 is absent)

As Parm2 is absent, Parm1 is the name of a CSV file that contains a list of values to be replaced. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
PERSONS	ADDRESS_1	REPLACE	replacelist.csv		

The following snippet shows the example of a CSV file `replacelist.csv` , which must be saved in the `gtfdm.exe` directory.

```
Replace.csv
        Te, AD
        st, CD
           ing, EH
```

Note: Fast Data Masker processes the replace values in the CSV file sequentially from top to bottom, so the unmasked values in the CSV replacement files are order dependent.

RIDCHECKDIGIT

The RID function masks an existing Recipient Identification Number.

Parameters: None

Applies To: Numeric, Character

Example:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PERSONS	RID	RIDCHECKDIGIT				

RJUST

The RJUST function strips blanks from the right of the string and right justifies the column in the string, padding the left with blanks (default) or the value defined in Parm1.

Note: This function is most likely to be used in conjunction with the SUBSTR function.

Parameters

- Parm1 (Optional)
  Specifies the value that you want to use for padding.

Applies to: Character

Example: The example masks `LAST_NAME` (for example, `Smith`) as something like `55555SMITH` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
PEOPLE	LAST_NAME	RJUST	5				N

RUT

The RUT function generates a Chilean Social Security Number.

Parameters: None

Applies to: Character

Example: The `SOCIAL_SECURITY_NUMBER` column is masked with a Chilean RUT value. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PERSONS	SOCIAL_SECURITY_NUMBER	RUT				

SEQCHAR

The SEQCHAR function represents a sequence using BASE62 numbers. The start key is converted to a BASE62 number which becomes the start value of the sequence.

Parameters

- Parm1
  Specifies an alphanumeric key to initialize the sequence.

Applies to: Character

Example: For example, the key aaaaaaaaaa is converted to the BASE62 number 1524750604, which is converted to the shorter BASE62 number 1fBhHg. The CONFIRMED_EMAIL column is now masked with the values 1fBhHg, 1fBhHh, 1fBhHi, and so on. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
PERSONS	CONFIRMED_EMAIL	SEQCHAR	aaaaaaaaaa				N

SEQLOV

The SEQLOV function masks the column values with sequentially selected values from a seed list. This can come from a seed file (.txt) or database-based seed table.

Parameters

- Parm1
  This can be one of two values:
  – The name of the seed file, as a .txt file. You can create multi-column seed file seedlists from multiple seed files (define one seed file for each column).

    NOTE

    By default, the location of seed files is `C:\Program Files\Grid-Tools\FastDataMasker \seedtables` . You can define this within FDM with the dialog at **Settings/Set Default Directories**.
  – The name of a seedlist from a database table (defined by the Masking Option SEEDTABLE). This is in the form of a string, which is present in the RD_REF_ID column (by default, the first column) of the database table, for all the seedlist values.
- Parm2
  Specifies the index of the column in the table you define in Parm1, from which you want to generate a masking value. Default value = 1, i.e. the first column to the right of that which contains the name of the seedlist.
- (Optional) Parm4
  Specifies a sequence identifier. This prevents repetition of values from seedlists used in multiple instances of SEQLOV in a masking job. Instances of SEQLOV with the same seedlist and the same sequence identifier, pick unique values from the seedlist across all instances. Without this identifier, columns masked with the same seedlist may be masked with repeated values from that seedlist.

Applies to: Character, Number, and Date

Example: The following table shows the usage for single-column seedlists.

Table	Column	Function	Parm1	Parm2
ADDRESS	CITY	SEQLOV	uktowns.txt	1
ADDRESS	STATE_PROVINCE	SEQLOV	ukpostcode3.txt	1
PERSONS	LAST_NAME	SEQLOV	lastnameindian.txt	1

SEQLOV1

The SEQLOV1 function generates sequential values from a database-based seed table (that contains for example, addresses, cities, states/provinces, ZIP code, etc) that are valid for one existing value from these columns.

Parameters

- Parm1
  Specifies the name of the seedlist in the table that you define with the Add Seedlists from a database table). This is in the form of a string, which must be present in the Seedlist Name Column (TDM Portal) or in the first of the columns defined by SEEDTABLECOLUMNS (Fast Data Masker), for all entries in the seedlist you want to use. In the provided *scramble* database, this column is RD_REF_ID.
- Parm2
  Specifies the index of the column in the table you define in Parm1, from which you want to generate a masking value. Default value = 1, i.e. the first column to the right of the column that contains the name of the seedlist.
- Parm3
  Specifies the column in the table to be masked, from which to get the existing value that the function matches in the seed table.

NOTE

The column in the seed table against which this column value is matched, is the first of the Seedlist Value Column(s) you define on the Masking Configuration page (TDM Portal), or the second of those defined by the Masking Option SEEDTABLECOLUMNS (Fast Data Masker).

- (Optional) Parm4
  Specifies the first *n* characters from the comparison column defined in Parm3, to compare with the seed table. This filters the seedlist defined by Parm2, from which TDM picks a value for masking.

Applies to: Character, Numeric, and Date

Example: The database table below shows part of a seed table. For each column in your data to mask, FDM generates a seedlist that consists of rows from the seed table, with the value of Parm1 (in this case `US_ADDRESSES`) in the Seedlist Name column (equivalent to RD_REF_ID in the *scramble* database). Parm3 is the column 'ZIP_CODE' - this subdivides the `US_ADDRESSES` seedlist into a sub-list, in which all values of 'REF_COL' (in this case postal codes) match the value of 'ZIP_CODE' in the row to mask (if Parm4 is defined as *n*, this match only applies to the first *n* characters of 'ZIP_CODE'). The value of Parm2 defines from which column of this sub-list, FDM selects a value.

Seed Table 'SEED' (showing part of 'US_ADDRESSES' seedlist)

SEEDLIST_NAME	REF_COL	VALUE_COL_1	VALUE_COL_2	VALUE_COL_3	VALUE_COL_4	VALUE_COL_5
US_ADDRESSES	99627	13	Anderson Rd	Mc grath	Yukon-koyukuk	AK
US_ADDRESSES	99682	86	West Ave	Tyonek	Kenai peninsula	AK
US_ADDRESSES	99929	22	Park Rd	Wrangell	Wrangell-perterburg	AK
US_ADDRESSES	80216	2	Park St	Denver	Denver	CO
US_ADDRESSES	80104	145	Smith St	Castle Rock	Douglas	CO

Examples of use of masking function

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	Source of resultant masking value
ADDRESSES	ADDRESS2	SEQLOV1	US_ADDRESSES	3	ZIP_CODE	4	SEED.VALUE_COL_2, where SEED.REF_COL matches first 4 chars of ADDRESSES.ZIP_CODE
ADDRESSES	CITY	SEQLOV1	US_ADDRESSES	4	ZIP_CODE	3	SEED.VALUE_COL_3, where SEED.REF_COL matches first 3 chars of ADDRESSES.ZIP_CODE
ADDRESSES	STATE_PROVINCE	SEQLOV1	US_ADDRESSES	6	ZIP_CODE	2	SEED.VALUE_COL_4, where SEED.REF_COL matches first 2 chars of ADDRESSES.ZIP_CODE

SEQNUMBER

The SEQNUMBER function updates each row with a user-defined sequence.

Parameters

- Parm1 (Optional)
  Specifies the start value for the sequence. If Parm1 is not provided, the sequence starts at 1.

Applies to: Numeric

Example: If Parm1 is `10` , the first row is updated for this column with a value `10` , the next row with `11` , and so on. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
PERSONS	ID	SEQNUMBER	10				N

SHUFFLE

The SHUFFLE function shuffles the values in the specified column for an entire table. The function creates a seed file by writing the value of each row for the column to a list of values. It then uses the SEQLOV function to overwrite the values in the database.

The SHUFFLE function lets you write the list of values to the following types:

- FILE: Fast Data Masker knows to write to a file if a . (dot) is present in the Parm1 value.
- Database: If Parm1 does not have a . (dot) value, the category name is stored in the database seed table.

Note: Do not run this function in maps with cross-references.

Parameters

- Parm1
  Specifies the category in the seed table in which your list of values is saved. For example, `MY_ADDRESSES` containing `Address` , `City` , and `Postcode` .
- Parm2
  Specifies the column in the seed table where you want to place the value. For example, entering `1` selects `RD_REF_1` , which is `ADDRESS` in the example below.

Masking Options

For this function to run, you need to use the following masking options:

- SEEDTABLECONNECT=`connectscramble.txt`
- SEEDTABLE=`gtsrc_reference_data`
- SEEDTABLECOLUMNS=`RD_REF_ID,RD_REF_VALUE1,RD_REF_VALUE2`

Note: Do not use the name of an existing seed table.txt file as it is overwritten each time the function is run.

Applies to: Character and Number

Example: The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
ADDRESSES	ADDRESS_1	SHUFFLE	MY_ADDRESS	1			N
ADDRESSES	CITY	SHUFFLE	MY_ADDRESS	2			
ADDRESSES	POSTCODE	SHUFFLE	MY_ADDRESS	3			

SQLFUNCTION

The SQLFUNCTION function lets you use a native database function or a user-defined function. You can also use this function to use normal SQL operators to process combinations of other columns. All SQL functions with the exception of aggregate functions must work.

Parameters

- Parm1
  Specifies the SQL function or SQL statement.
- Parm2 (Optional)
  Specifies whether to apply SQLFUNCTION as one SQL update statement at the end of the masking process. This is far more efficient than processing one row at a time. To do so, select this parameter.

Applies to: Character, Numeric, and Date

Example: Parm1=`first_name || ' '|| last_name` . This example concatenates the first name, space, and last name.

Parm1=`mynumberformat(HHNO)` . This example passes HHNO into the database function. The returned value populates the masked column.

The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
C_BO_PRTY2	FST_NM	HASHLOV	FIRSTNAME	1			Y
C_BO_PRTY2	ORIG_FST_NM	HASHLOV	FIRSTNAME	1	PRTY_FST_NM		Y
C_BO_PRTY2	LAST_NM	HASHLOV	LASTNAME	1			Y
C_BO_PRTY2	ORIG_LST_NM	HASHLOV	LASTNAME	1	PRTY_LAST_NM		Y
C_BO_PRTY2	FULL_NM	SQLFUNCTION	ORIG_LST_NM || ',' || ORIG_FST_NM || '' || ORIG_MD_NM				N

Note: You can also use SQLFUNCTION to add extra columns in a mapping CSV. These are as follows:

- UPDATE: If Update=N (default=Y), then the mask for this row is not applied and is put in the memory to be used when required.
- USEMASKEDVALUES: If Use Masked Values=Y (default=N), then columns specified in SQLFUNCTION are tested to see whether they have been masked earlier in the mapping. If so, the masked value is used.

SUBSTR

The SUBSTR function extracts a sub-string from an existing string based on the start position and the length specified for the sub-string.

Parameters

- Parm1
  Specifies the position from where to start the extraction.
- Parm2
  Specifies the number of characters to extract from the string.

Applies to: Character

Example: The SUBSTR function extracts sub-strings from the strings in the `FIRST_NAME` column. The extraction starts from the position one with the number of characters to extract as four. For example, if the original string in the column is `Johnson` , the function in this case extracts the sub-string `John` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
PEOPLE	FIRST_NAME	SUBSTR	1	4			Y

Note: In case of a flat file, the Table column remains empty.

TIN

The TIN function generates a United States Tax Identification Number.

Parameters

- Parm1 (Optional)
  Specifies the separator character that you want to use.

Applies to: Character and Numeric

Example: The `TAXID` column in the table `PEOPLE` is masked with a generated US Tax Identification Number. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PEOPLE	TAXID	TIN				

TRANSLATE

The TRANSLATE function searches the column values for every single character specified in Parm1 and replaces it with the corresponding character (sequentially) specifies in Parm2. TRANSLATE is a character-by-character operation. This function is case-sensitive.

Parameters

- Parm1
  Specifies the characters to be searched in the column.
- Parm2
  Specifies the corresponding characters to be replaced.

Applies to: Character and Numeric

Example: All instances of `a` in the column `FIRST_NAME` are translated to `x` , and all instances of `1` in the column `MEMBERSHIP_ID` are translated to `6` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PERSONS	FIRST_NAME	TRANSLATE	a	x		
PERSONS	MEMBERSHIP_ ID	TRANSLATE	1	6		

TRANSPOSE

The TRANSPOSE function consistently converts one character to another character; for example, a to c, b to d, and so on. Set PARM1 to a number to act as a key.

Parameters

- Parm1
  Specifies the key of the transposition.
- Parm2

Applies to: Character

Example: If `ab` is found in the column value and the Parm1 is set to `4` , the value is converted to `ef` . That is, every `a` character is translated to `e` and every `b` character is translated to `f` based on the key value in Parm1. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
PERSONS	FIRST_NAME	TRANSPOSE	4		

TRIM

The TRIM function removes all the leading and trailing spaces from the specific column. After you use the TRIM function, you can use your other masking functions as required.

Note: For database masking, the TRIM function is applicable only for non-date columns. Also, we recommended that you do not use the TRIM function on the column where the width of the column is explicitly defined.

Parameters: None

Applies to: Character

Example: Leading and trailing spaces in the `LAST_NAME` column of the `PERSONS` table are trimmed. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
PERSONS	LAST_NAME	TRIM					Y

Note: For mainframe, you can use the COMBINEVALS, SETSTR, or ASSIGNSTR functions to get the same functionality that the TRIM function provides. For more information about how to use these mainframe-specific functions, see their corresponding documentation on the mainframe.

TRUNCATE

The TRUNCATE function truncates all the data in the table.

Parameters: None

Applies to: Character, Date, and Numeric

Example: Truncating the data executes a fast delete of all the data in the table. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
PERSONS	LAST_NAME	TRUNCATE					N

UNIQUEBELGIANID

The UNIQUEBELGIANID function generates a unique Belgian National Identification Number sequentially.

Parameters:

- Parm1 — Start Date of Birth
  Specifies a start date of birth in yyyy-MM-dd format.
- Parm2 — End Date of Birth
  Specifies an end date of birth in yyyy-MM-dd format.
- Parm3 — Gender
  Specifies the gender as one the following: U (unisex) or M (male) or F (female).

Applies To: Numeric, Character

Example:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PEOPLE	BELGIAN_ID_COLUMN	UNIQUEBELGIANID	1991-01-01	2001-12-31	U	

UNIQUEBSN

The UNIQUEBSN function consistently masks the given column values with a unique Dutch BSN Number.

Parameters: None

Applies to: Character and Numeric

Example: The following are the examples:

- `123` to `661165544`
- `456` to `384090095`
- `123` to `661165544`
- `456` to `384090095`

UNIQUECPR

The UNIQUECPR function consistently masks the given column values with a unique Danish CPR Number.

Parameters: None

Applies to: Character

Example: The following are the examples:

- `294398775` to `101500314`
- `438425710` to `101500411`
- `294398775` to `101500314`
- `438425710` to `101500411`

UNIQUEFINNISHID

The UNIQUEFINNISHID generates a sequential unique Finnish ID.

Parameters: None

Applies to: Character

Examples:

- 010120-029J
- 010120-001M
- 010120-002N
- 010120-003P
- 010120-004R

UNIQUESPANISHCIF

The UNIQUESPANISHCIF function generates a unique Spanish CIF. A Spanish CIF is a 9-character value (one letter, followed by 7 numbers, followed by one checksum letter/number).

Parameters:

- **Parm1**
  Prefix to CIF. This value forms the start of any CIFs you generate.

  NOTE

  If the prefix is invalid (i.e. does not conform to the CIF format, or is too long), the function ignores all characters from the invalid character onwards.

Applies to: Character

Examples:

Parm1	First Output CIF
<blank>	A0000000G
C86	C8600000K
HG29	H0000000A

UNIQUESPANISHID

The UNIQUESPANISHID function generates a unique Spanish ID.

Parameters:

- **Parm1**
  NIF/NIE
  Specifies whether the Spanish ID is of the type NIF or NIE. Takes argument True (NIF) / False (NIE)

Applies to: Character

UNIQUETURKISHID

The UNIQUETURKISHID function generates a unique 11-digit Turkish Identification Number. If Parm2 and Parm3 are provided, they define the upper and lower bounds; if valid.

Parameters

- Parm1
  Specifies whether you want to generate a sequence. Values: Y or N. Default: `N` .
- Parm2
  Defines the start value of the sequence. Default: `100000000` .
- Parm3
  Defines the end value of the sequence. Default: `999999999` .

Applies to: Numeric

Example 1 : Parm1: Y, Parm2: `100000000` , Parm3: `100000010`

- `10000000078`
- `10000000146`
- `10000000214`
- `10000000382`
- `10000000450`
- `10000000528`
- `10000000696`
- `10000000764`
- `10000000832`
- `10000000900`

Example 2: Parm1: N, Parm2: optional, Parm3: optional.

- `67721221312`
- `48512950116`
- `67358262436`
- `36834234250`
- `49588725660`
- `34986642282`
- `38676152648`
- `49218435488`
- `37194605476`
- `19870583810`

UNIQUETURKISHTAXID

The UNIQUETURKISHTAXID function generates a unique 10-digit Turkish Tax Identification Number.

Applies to: Numeric, Character

Parameters:

- **Parm1**
  Specifies whether you want to generate a sequence. Values: Y or N. Default: N.
- **Parm2**
  Defines the start value of the sequence if Parm1 is Y. Default: 1111111111.
- **Parm3**
  Defines the end value of the sequence if Parm1 is Y. Default: 9999999999.

USPHONE

The USPHONE function masks the column with an auto-generated 7-digit US phone number of the format xxxxxxx.

Parameters: None

Applies to: Character and Numeric

Example: The USPHONE column in the PERSONS table is masked with auto-generated 7-digit US phone numbers. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PERSONS	USPHONE	USPHONE				

USPHONE(10)

The USPHONE(10) function masks the column with an auto-generated 10-digit US phone number of the format xxx-xxx-xxxx.

Parameters: None

Applies to: Character and Numeric

Example: The USPHONE column in the PERSONS table is masked with auto-generated 10-digit US phone numbers. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
PERSONS	USPHONE	USPHONE(10)					

USZIP

The USZIP function masks the columns with an auto-generated 5-digit US ZIP code.

Parameters: None

Applies to: Character and Numeric

Example: The ZIP_CODE column in the PERSONS table is masked with auto-generated 5-digit US ZIP codes. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
PERSONS	ZIP_CODE	USZIP					

USZIP+4

The USZIP+4 function masks the columns with an auto-generated 9-digit US ZIP code (format: xxxxxxxxx).

Parameters: None

Applies to: Character and Numeric

Example: The ZIP_CODE column in the PERSONS table is masked with auto-generated 9-digit US ZIP codes. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
PERSONS	ZIP_CODE	USZIP+4					

VALIDSIN

The VALIDSIN function tests for a valid Canadian Social Insurance Number (SIN). If the data is a valid SIN, then it is replaced with a new value, using an optional Parm1 as a separator for each of the three sets of digits. Otherwise, it leaves the number *as is*.

Some examples of a bad SIN include:

- The SIN is not a number after the separator characters are removed.
- It is not 9 digits in length.
- The number starts with 0 or 8.

Parameters

- Parm1 (Optional)
  Specifies the separator character which must match the existing data or all numbers are flagged as invalid and no changes are made.

Applies to: Character and Numeric

Example: The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4	KeepNulls
PERSON	SIN_NUMBER	VALIDSIN					N

VALIDSSN

The VALIDSSN function identifies whether a column contains a valid SSN (United States Social Security Number). If so, the function masks it with a generated SSN.

Parameters

- Parm1
  Specifies the separator character.

Applies to: Character and Numeric

Example: If the column `ID` in the table `PEOPLE` contains a valid SSN, it is replaced with a random SSN. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PEOPLE	ID	VALIDSSN				

VALIDSSNSUB

The VALIDSSNSUB function identifies whether the first 9 characters of a column contain a valid SSN (United States Social Security Number). If so, the function masks the valid SSN with a generated SSN.

Parameters

- Parm1

Applies to: Character and Numeric

Example: If a valid SSN is found in the first nine characters of the column `ID` in the table `PEOPLE` , it is replaced with a random SSN.

The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PEOPLE	ID	VALIDSSBNSUB				

VALIDTIN

The VALIDTIN function identifies whether the column contains a valid TIN (United States Tax Identification Number). If so, the function masks it with a generated TIN.

Parameters

- Parm1
  Specifies the separator character.

Applies to: Character and Numeric

Example: If the column `ID` in the table `PEOPLE` contains a valid TIN, it is replaced with a random TIN. The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PEOPLE	ID	VALIDTIN				

VARIENCE

The VARIENCE function generates values based on Parm1 (% value) and then adds them to or subtracts them from the column values.

Parameters

- Parm1
  Specifies the percentage variance (1-99).
- Parm2 (Optional)
  Specifies the minimum permitted value.
- Parm3 (Optional)
  Specifies the maximum permitted value.

Applies to: Numeric

Example: If the column value is `100` , then Parm1 applies `60%` variance and a random number is generated between `40` and `160` . However, Parm2 (minimum permitted value) and Parm3 (maximum permitted value) ensure that the generated random value lies in the range `50` through `150` instead of `40` through `160` .

The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3	Parm4
PERSONS	CREDIT_SCORE	VARIENCE	60	50	150	

VISACARD

The VISACARD function generates a random VISA credit card number.

Parameters: None

Applies to: Character and Numeric

Example: Generates a valid VISA credit card number, such as `4012888653017322` . The following table shows the usage:

Table	Column	Function	Parm1	Parm2	Parm3
PERSONS	CREDIT_CARD	VISACRAD			

XMLREPLACE

The XMLREPLACE function replaces values embedded in XML files in the database.

Parameters:

- Parm1
  Specifies the value to replace.
- Parm2
  Specifies the value to use as a replacement for Parm1.

Applies to: Character and Numeric

Example: Replaces the value of XML tag 'first_name' with the value of XML tag 'last_name'.

Table	Column	Function	Parm1	Parm2
PEOPLE	XML_DATA	XMLREPLACE	/TABLES/PEOPLE/first_name	/TABLES/PEOPLE/last_name

Masking Options

The **Options** tab includes additional parameters to control the masking run, audit options, cross-referencing options, and seed table options.

Additionally, as a user or administrator, you can create a global options file to store your custom default values for masking options. Browse to the FDM installation directory and create a file named global_options.txt. The available parameters and format are listed in this article. If you have several sets of options, they become active in the following order:

- Lowest priority: Masking **Options** tab in FDM
- Global options file in installation directory, or in a non-default path specified in the TDM_GLOBAL_OPTIONS_PATH environment variable
- Highest priority: Options file specified on the command line at start-up time

The full list of the available options, what they do, and what values apply is as follows:

Audit

- **AUDIT**=ALL
  All rows are audited.
- **AUDIT**=ROWnnn
  nnn represents the number of rows to be audited. For example, ROW1000 produces an audit of the first 1000.
- **AUDIT**=SAMPLEnnn
  Every nnn rows is displayed. For example, SAMPLE100 produces an audit of every 100th row.
- **AUDITDIR**
  Set the path to the audit file directory.
- **AUDITEPASSWORD**

Set the encrypted password for the audit ZIP file.

- **AUDITFILE**
  The name of the file in which to store the audit information; myaudit.csv.
- **AUDITONLYCOLUMNS**
  Mention the specific list of columns to be audited in the format—table1.column1, table2.column2, table3.column3.
- **AUDITPASSWORD**
  Set the password for the audit ZIP file.
- **AUDITVALUES**
  N — Show only new values, not the old values in the AUDIT file.
  Default: Show both old and new values in the AUDIT file.
- **AUDITZIP**
  Zip and encrypt the audit CSV file. Values are winzip or jzip for the program to use for the zip.

Cross-Reference

- **CASEINSENSITIVEXREF**
  Make comparisons case insensitive (for cross-reference).
- **CROSSREFCONNECT**
  The name of the connection file to read and write cross-reference data; Connectscramble.txt.
- **CROSSREFTABLE**
  The name of the table to read and write cross-reference data to; Gtsrc_xref.
- **ENCRYPTXREF**
  Encrypt the old values in the cross-reference table.
- **TRIMMEDXREF**
  Trim values before comparing (for cross-reference).

Date

- **CDATE**
  Override the date (today) for the purposes of date calculation functions. For example, DOB (format: YYYYMMDD).
- **HIGHDATE**
  Override the highest data that offset date functions process. For example, dates later than 22000101 are ignored.
- **LOWDATE**
  Override the lowest data that offset date functions process. For example, dates earlier than 19000101 are ignored.

Directory

- **BACKUPDIR**
  The directory name for backup files. If this setting is blank, the default from the Fast Data Masker directory is used.
- **ERRORDIR**
  The directory name for error files. If this setting is blank, the default from the Fast Data Masker directory is used.
- **LOGDIR**
  The directory name for log files. If this setting is blank, the default from the Fast Data Masker directory is used.
- **SEEDFILEDIR**
  The directory name where seed data files are stored. The default value is seedtables sub-directory.

Format Encrypt

- **FORMATENCRYPTDELIMITER**
  Defines one or more single-character delimiters, to separate the value to mask into strings. FORMATENCRYPT then masks each string separately.

NOTE

Each delimiter must be either a single character, or the case-insensitive keyword SPACE (to indicate a space, i.e. " "). If you enter more than one consecutive character, FORMATENCRYPT ignores the delimiter.

Separate delimiters with a space, for example " **- , \ space** " to use the characters hyphen, comma, backslash and space as delimiters.

- **FORMATENCRYPT1DELIMITER**
  Defines one or more single-character delimiters, to separate the value to mask into strings. FORMATENCRYPT1 then masks each string separately.

  NOTE

  Delimiter logic for FORMATENCRYPT1DELIMITER is the same as FORMATENCRYPTDELIMITER.
- **FORMATENCRYPT1MAINFRAMECOMP (Y, N)**
  This option ensures that the FDM special character set on the Windows side has the same special characters as on the mainframe side. You can use this option in combination with **FORMATENCRYPTEXTENDEDCHARS = Y** and **OLDEXTENDEDCHARS**.
  Default: N
  – Special character map used when **FORMATENCRYPTEXTENDEDCHARS** is enabled (63 characters):
  `!#$%&'()*+,-./:;<=>?@[\]_`{|}~¡¢£¤¥¦§¨©ª«¬®¯°±²³´µ¶·¸¹º»¼½¾¿×÷"`
  – Special character map used when **OLDEXTENDEDCHARS** is enabled (68 characters):
  `!#$%&'()*+,-./:;<=>?@[\]_`{|}~¡¢£¤¥¦§¨©ª«¬®¯°±²³´µ¶·¸¹º»¼½¾¿Ð×ØÞßø÷ø"`
- **FORMATENCRYPT1EXCLUDESPECIALCHARS (Y, N)**
  Use this option in combination with the **FORMATENCRYPTEXTENDEDCHARS, OLDEXTENDEDCHARS**, or **FORMATENCRYPT1MAINFRAMECOMP** options. When this option is enabled, it will encrypt, following the format encrypt logic, but special characters will be removed from the final result.
  Default: N
  Example:

	FORMATENCRYPT1MAINFRAMECOMP=N		FORMATENCRYPT1MAINFRAMECOMP=Y	
Original	Extended	Extended Old	Extended	Extended Old
GHI/123-C3	ÅNO760O2	ËNO760N2	ÅNO760O2	ËNO760N2

- **FORMATENCRYPT1IGNORESPECIALCHARS (Y, N)**
  Use this options in combination with the **FORMATENCRYPTEXTENDEDCHARS**, **OLDEXTENDEDCHARS**, or **FORMATENCRYPT1MAINFRAMECOMP** options. When this option is enabled, it will encrypt, following the format encrypt logic, but special characters will not be encrypted.
  Default: N
  Example:

	FORMATENCRYPT1MAINFRAMECOMP=N		FORMATENCRYPT1MAINFRAMECOMP=Y	
Original	Extended	Extended Old	Extended	Extended Old
GHI/123-C3	ÅNO/760-O2	ËNO/760-N2	ÅNO/760-O2	ËNO/760-N2

Languages

Fast Data Masker currently supports three languages for masking—English, German, and Spanish.

When Fast Data Masker starts, it verifies the current default locale. If the language is supported (one of the three listed), it processes messages in that language. If the language is not supported, it defaults to English unless set in the options file.

You can override the local language by altering the language option as follows:

- LANGUAGE
  Use one of the three languages—en (English), de (German), or es (Spanish)

Large Tables

- **LARGETABLESPLITENABLED**
  Enables large tables processing. Set this parameter to Y to enable, and to N to disable.
  Default: N
- **LARGETABLESPLITSIZE**
  Defines the minimal number of rows for FastDataMasker to start using large table processing.
  Default: 1000000

Parallelism

- **PARALLEL**=n
  Enables users to attempt to run ‘n’ number of concurrent threads. This is constrained by the number of physical cores and processors available.
  Note: When masking Microsoft SQL Server tables, use the PARALLEL option only when the table has a primary key or unique index. If the table does not have a primary key or unique index and you use the PARALLEL option, masking is either slow or does not work.
  Fast Data Masker assigns a separate thread for each table in a CSV if there is more than one. However, a CSV would have only one table, which can be split using the WHERE clauses. For example, a CSV using the WHERE clauses below would have four splits:
  WHERE, CUSTID<100
  ….
  WHERE, CUSTID BETWEEN 100 AND 200
  ….
  WHERE, CUSTID BETWEEN 200 AND 300
  ….
  WHERE, CUSTID>100

Seed Tables

- **CASEINSENSITIVESEED**
  Makes search for rd_ref_value column, using RANDLOV1 function case insensitive.
- **LOADALLSEEDDATA**
  N (default)—loads all seed data into Java memory for the RANDLOV1 function, irrespective of the distribution of rd_ref_values in the table.
- **SEEDTABLECONNECT**
  The name of the connection file to get seed data from; Connectscramble.txt.
- **SEEDTABLE**
  The name of the table to get the seed data from; Gtsrc_reference_data.
- **SEEDTABLECOLUMNS**
  Comma separated list of the columns in SEEDTABLE.

Shuffle

- **SHUFFLEDISTINCT**
  This option takes Y or N.
  Y selects distinct values for the shuffle creates.
  N is the default and selects all values.
- **SHUFFLELIMIT**

n—only select n values for the shuffle.

- **SHUFFLEONLY**
  This option takes Y or N.
  Y does not update the database. Instead, it just produces the shuffle files or database shuffle values.

Poststep

- **POSTSTEP**
  This is the path to a SQL file to perform post-steps, the SQL should be ANSII standard insert, update, or delete operations. The SQL file is executed after the masking.

Prestep

- **PRESTEP**
  This is the path to a SQL file to perform pre-steps, the SQL should be ANSII standard insert, update, or delete operations. The SQL file is executed prior to masking.

Other

- **BADDATESTRING**
  For DOB/DOD on dates stored in character fields, specify the data to replace the invalid data as YYYY/MM/DD.
- **BATCHSIZE**
  Number of lines to commit to a database at a time.
- **BLANKSASNULLS**
  Set to Y. For character data types, if the column contains blanks to the column width, treat as a null for keepnulls in the masking CSV.
- **CASEINSENSITIVEHASHLOV**
  This option is always used with the HASHLOV function. With this option, you can define whether the HASHLOV function masks the data in a case-sensitive or case-insensitive mode. By default, this option is set to the case-insensitive mode. Enter N as a value to set this option to the case-sensitive mode.
  Example: The firstname and lastname columns include the data in the following format:

firstname	lastname
Jean	Muller
JEAN	MULLER
jean	muller
jEAN	mULER

When the option is set to the case-insensitive mode (default mode) and you use the HASHLOV function, the data is masked as follows:

firstname	lastname
Isabel	Rowland
Isabel	Rowland
Isabel	Rowland
Isabel	Rowland

Now, when you set the option to the case-sensitive mode and use the HASLOV function for masking, the same data is masked as follows:

firstname	lastname
Melany	Maynard
Isabel	Rowland
Tania	Sutton
Daniella	Buchanan

- **CHUNKSIZE** (File masking only)
  Number of lines to write to a file at a time.
- **COMMIT**=nnnn
  Commit after nnn rows for each table to be masked. For example, 1000 forces a commit after 1000 rows for each table.
- **DBUPDATES**
  N—run in simulation mode
  S—create SQL file <table name>_UPDATES.sql
  P—see Prestep and Poststep options
  Note: DBUPDATES=S only available for non-DB2 databases with unique or primary key columns.
- **DB2BATCHUPDATE**
  N (Default). If Y, use fast batched updates rather than standard "update where current of" cursor method (DB2 ONLY).
- **DIAGLEVEL**
  Possible values: 0, 1, 2 or 4. Debug info is generated according to value.
- **DROPRESTART**
  Set the value to N if you do not want to drop the restart column (which Fast Data Masker creates) after masking is complete. Retaining the restart column is helpful in scenarios where you want to use it for audit purposes. The default value is Y. This option is not applicable if you explicitly specify your own restart column.
- **EMPTYASNULL**
  Set to Y. For character data types, if the column contains a blank or spaces, treat as a null for keepnulls in the masking CSV.
- **FASTIGNORE**
  Set the value to Y if you want to use this option. This options is always used with the IGNORE function. When used with the IGNORE function, this option improves the masking performance.
  The IGNORE function inserts as well as retrieves data from the cross-reference table, which is why it is row-by-row processing and slow. However, with the FASTIGNORE option, you update the data in one SQL statement, rather than row-by-row (which is very slow).
  Review the following considerations when using the FASTIGNORE option:
  – Ensure that the cross-reference table and the table to be masked are on the same RDBMS.
  – Ensure that the cross-reference table and the table to be masked are on the same server.
  – Ensure that the cross-reference table must have old and new values pre-populated for the chosen cross-reference identifier.
  – Collation of the table to be masked and gtsrc_xref should be the same.
- **FETCHSIZE**
  Number of lines to read from a database or file at a time.
- **LOWERCASEKEY**
  Specify a lowercase key for masking; for example, `qazwsxedcrfvtgbyhnujmikolp` . Ensure that the key does not start with the character `a` . You can use this option with the FORMATENCRYPT masking function.

For example, if you use this option with the FORMATENCRYPT function, the function starts making the first occurrence of the lowercase character, which the function ignores if this option is not set.

- **MD5HASHLOV**
  Set this value to Y to use an MD5 hashing algorithm with the HASHLOV functions. Leave this value blank to use the default Java hashing algorithm.
- **NUMERICKEY**
  Specify a numeric key for masking; ; for example, `8524569173` . Maximum 15 digits are allowed. If you want to enter more than 15 digits, provide value in quotes; for example, `"9182736450514239687"` .
  You can use this option with the FORMATENCRYPT, FORMATLUHN, FORMATVIN, and HASHTURKISHID masking functions.
- **ORDERBY**
  Y (Default)—you might want to turn it off for RMS(VMS). This value decides whether selected data is ordered by PK column or not.
- **PROCESSCOUNT**
  Process count to limit the number of rows processed per table.
- **RELAXNONINDEXVALIDATION**
  XREF on non-varchar columns of no PK/UK tables.
- **RESETRESTART**
  Reset restart column (de_ident_ind) to null, masking starts from Row 1.
- **RESTART**
  Restart mask from last fail point. Requires varchar column (de_ident_ind) added to table(s) to be masked, or use existing (empty) column in the tables to be masked. Add this column name to the "restart column" column in your masking CSV.
- **TRIMVALUES**
  This option can accept "Y" for yes and "N" for no. Yes implies you want to remove the leading and trailing spaces from all the columns that you have selected for masking. If you do not specify the value, Fast Data Masker uses the default value N.
- **UPPERCASEKEY**
  Specify an uppercase key for masking; for example, `PLOKMIJNUHBYGVTFCRDXESZWAQ` . You can use this option with the FORMATENCRYPT masking function.
- **USERFAASINDEX**
  Use RFA in the WHERE clause of the update SQL (VMS(RMS) DB only).
- **USERRNASINDEX**
  For non-indexed tables, use RRN function to get row identifier, and then combine with DB2BATCHUPDATE to use fast method (DB2400 only).
- **WHEREASSUBSET**
  Y—Flat files are scrambled and subsetted according to the WHERE clause.
  N—Use WHERE as criteria to mask and generate output of all records.
  Default: Y
- **XMLNAMESPACE**
  One or more tags in XML file or data contain xmlnamespace elements.

REST API Reference

This section provides information about the Representational State Transfer (REST) APIs that are available with the CA TDM Portal. The CA TDM Portal offers REST APIs that make the CA TDM data accessible to different development environments. The APIs enable external systems to configure, execute, and monitor various CA TDM operations (for example, registering objects, importing sample data) without having to access the UI. These interfaces provide an HTTP-based integration point to the CA TDM data, allowing read or write access. You can use these APIs with any language that understands how to manage HTTP integration.

NOTE

To execute APIs, it is necessary to submit an authorization token as part of the header. To manage this additional information, we recommend the use of software such as **Postman**.

You can find the complete information about the exposed CA TDM Portal REST APIs at the following location:

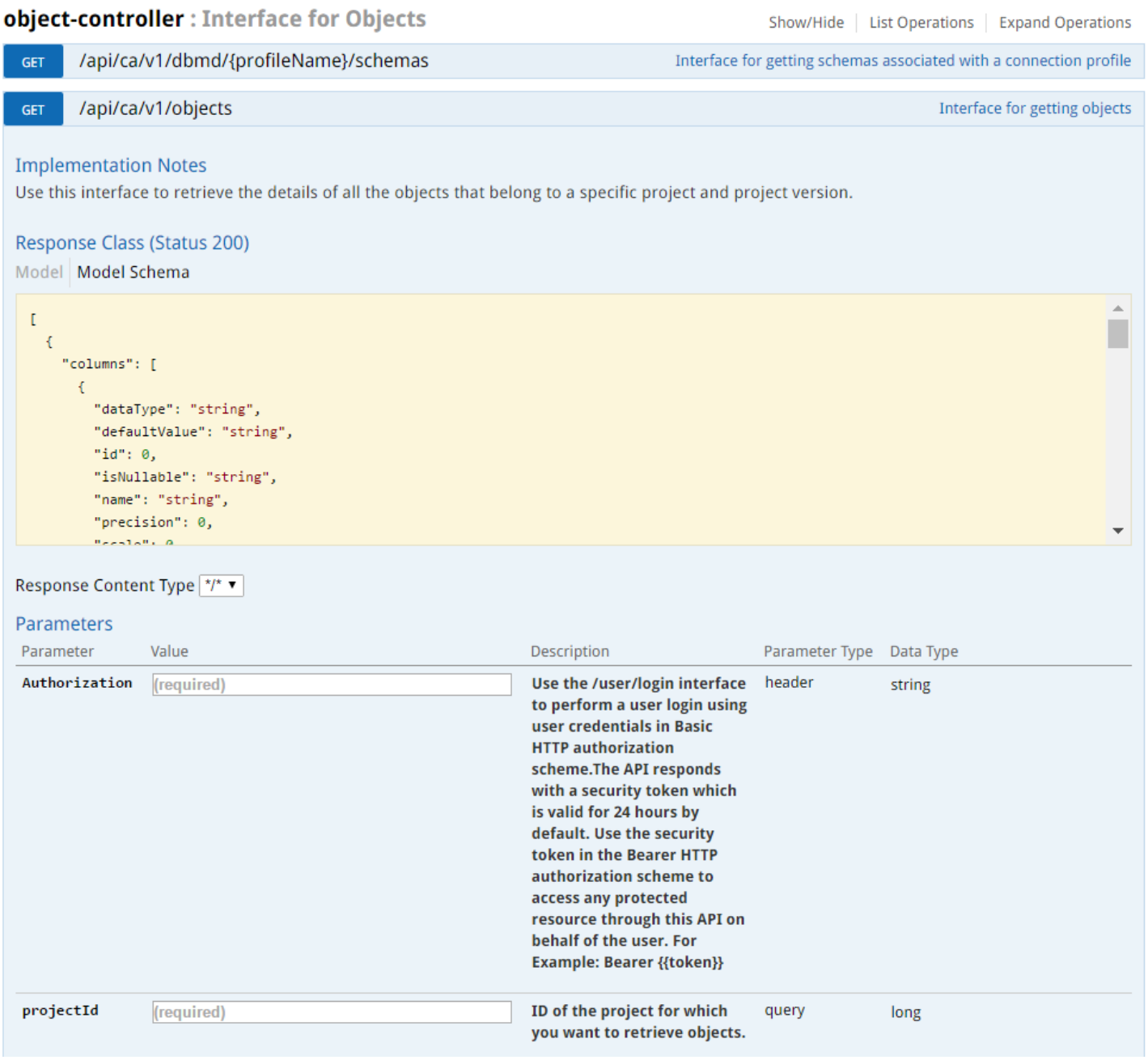
`https://<server>:<port>/<service_name>/swagger-ui.html`

- `<server>` represents the system where the CA TDM Portal instance is available.
- `<port>` represents the port number where the CA TDM Portal instance is listening.
- `<service_name>` represents the name of the service for which you want to access the APIs.

The following services are available:

- TDMConnectionProfileService
- TDMDataFlowService
- TDMDataReservationService
- TDMGeneratorService
- TDMJobService
- TDMLegacyExecuterService
- TDMModelService
- TDMMaskingService
- TDMOrchestrationService
- TDMProjectService
- TDMPublisherService
- TestDataManager

An example URL is `https://TDMserver01:8443/TDMModelService/swagger-ui.html`. When you access this URL, a Swagger UI page is displayed. This page shows all the APIs that are available for TDMModelService. You can specify the appropriate input for your API and test it to review the response. The following screen shot shows how APIs are displayed when you access the required URL:



object-controller : Interface for Objects

Show/Hide | List Operations | Expand Operations

GET /api/ca/v1/dbmd/{profileName}/schemas — Interface for getting schemas associated with a connection profile

GET /api/ca/v1/objects — Interface for getting objects

Implementation Notes

Use this interface to retrieve the details of all the objects that belong to a specific project and project version.

Response Class (Status 200)

Model | Model Schema

```
[
  {
    "columns": [
      {
        "dataType": "string",
        "defaultValue": "string",
        "id": 0,
        "isNullable": "string",
        "name": "string",
        "precision": 0,
```

Response Content Type */*

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization	(required)	Use the /user/login interface to perform a user login using user credentials in Basic HTTP authorization scheme.The API responds with a security token which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}	header	string
projectId	(required)	ID of the project for which you want to retrieve objects.	query	long

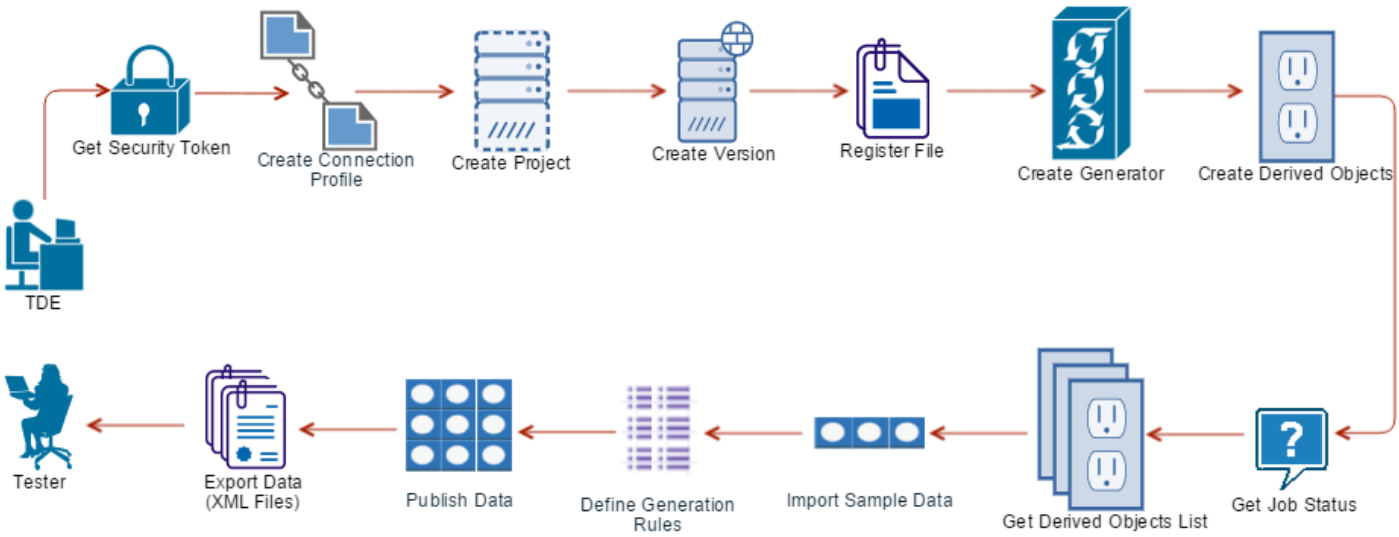
Use APIs to Prepare Test Data for Non-Relational Sources

Organizations want to rigorously test their applications using varied sets of data before each release. Unfortunately, in most of the cases, testers do not get access to the rich, high-quality test data. Because of the non-availability of enough test data, they are not able to comprehensively test their applications. The CA TDM Portal helps organizations address this situation by generating realistic synthetic data that testers can use to test their applications.

This article explains with the help of an example about how Test Data Engineers (TDEs) can use exposed CA TDM Portal APIs to generate synthetic test data for an application that uses data in the form of XML files. The example used in this article uses a sample XSD file to define the relational schema and the associated XML file for the sample data. This example also shows how TDEs can create data generation rules and can export the generated data. Testers can then use the generated XML files to confidently perform different testing scenarios.

The complete process is shown in the following diagram; TDEs perform all these tasks with the help of the exposed APIs:

Figure 58: Use APIs to prepare test data using a flat file



The steps shown in the diagram are as follows:

Note: For information about specific concepts (for example, project, data generator, data painter), see the relevant sections in this documentation. Additionally, the process explained in this article is applicable only for XML, XSD, JSON, WSDL, and RR Pair file types.

This page refers to the following API Services:

- TDMConnectionProfileService
- TDMProjectService
- TDMModelService
- TDMGeneratorService
- TDMJobService

Get a Security Token

The process to get a security token involves two steps:

1. Encode your CA TDM Portal credentials to the Base64 format.
2. Use the encoded value in a POST request to generate a security token.

The security token remains valid for 24 hours.

Encode CA TDM Portal Credentials

Encode your CA TDM Portal credentials to the Base64 format. You can use any application that allows you to do so.

Follow these steps:

1. Access an application that lets you encode your credentials to the Base64 format.
2. Enter your CA TDM Portal login credentials (in the format `<user name>:<password>`) in the source field.
   Note: Ensure that the credentials have appropriate permissions to perform all the required operations.
3. Click the option to encode the credentials. The encoded Base64 format for the example is displayed as follows:

```
ZwRTaX5pc4SxYXSvcjptYXJtaXRl
```

4. Note the encoded value.

Generate the Security Token

After you get the encoded value, use that value in a POST request. When you run the specific API, it generates a security token. Use that token in all the subsequent operations.

Follow these steps:

1. Access the following CA TDM Portal API:

   ```
   POST https://<server>:<host>/TestDataManager/user/login
   ```

   Note: For more information about this API, see the "auth-controller: Auth Controller" section at https://<server>:<port>/TestDataManager/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TestDataManager/swagger-ui.html.
2. Enter the encoded value in the **Authorization** field (`Basic <encoded value>`), which is as follows for the example:

   ```
   Basic YWRtaW5pc3RyYXRvcjptYXJtaXRl
   ```
3. Run the API to get a security token.
4. Note the value of the **token** parameter in the response body, which is as follows for the example:

   ```
   eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojUxm8ENTeRRrdOa-
   RQ5l4Ro
   ```

   You have successfully generated a security token that you can use in all the subsequent operations explained in this article.

The next step is to create a connection profile.

Create a Connection Profile

Create a connection profile to connect to the source or target databases.

Note: For more information about working with connection profiles in the UI, see Create and edit Connection Profiles in the UI section.

1. Access the following CA TDM Portal API:

   ```
   POST https://<server>:<host>/TDMConnectionProfileService/api/ca/v1/connectionProfiles
   ```

   Note: For more information about this API, see the "con-profile-controller: Interface for connection profiles" section at https://<server>:<port>/TDMConnectionProfileService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMConnectionProfileService/swagger-ui.html.
2. Enter the security token in the **Authorization** field as follows:

   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
   RQ5l4Ro
   ```
3. Click the model schema for the **profile** parameter and specify the required connection profile details. For the example used in this article, the following information was entered:

   ```
   {
   "name":"PO_Profile",
   "description":"PO_Profile",
   "dbType":"sql server",
   "server":"abc01-xy001",
   "port":"1433",
   "instance":"",
   "service":"",
   ```

```
"database":"podb",
"schema":"",
"username":"sa",
"password":"abcde@123"
 }
```

4. Run the API.
5. Review the response body and note the connection profile name, which is `PO_Profile` in this case.

The next step is to create a project.

Create a Project

All operations that you perform to prepare test data for non-relational data sources take place in context of a specific CA TDM project.

Note: For more information about working with CA TDM Portal projects in the UI, see Create and Edit Projects in the UI section.

1. Access the following CA TDM Portal API:

   ```
   POST https://<server>:<host>/TDMProjectService/api/ca/v1/projects
   ```

   Note: For more information about this API, see the "project-controller: Interface for projects" section at https://<server>:<port>/TDMProjectService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMProjectService/swagger-ui.html.
2. Enter the security token in the **Authorization** field as follows:

   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
   RQ5l4Ro
   ```

3. Click the model schema for the **projectInfo** parameter and specify the required project details. For this example, the following information was entered:

   ```
   {
   "description": "PO_Project Description",
   "inheritTables": true,
   "name": "PO_Project"
   }
   ```

4. Run the API to create a project.
5. Review the response body to get the project ID, which is `4945` in this case.

The next step is to create a version for this project.

Create a Version

After you create a project, you must create a version for the same project.

Note: For more information about working with CA TDM Portal project versions in the UI, see Manage Project Versions in the UI section.

1. Access the following CA TDM Portal API:

   ```
   POST https://<server>:<host>/TDMProjectService/api/ca/v1/projects/{projectId}/versions
   ```

   Note: For more information about this API, see the "version-controller: Version Controller" section at https://<server>:<port>/TDMProjectService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMProjectService/swagger-ui.html.
2. Enter the security token in the **Authorization** field as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
RQ5l4Ro
```

3. Click the model schema for the **versionInfo** parameter and specify the required version details. For this example, the following information was entered:

```
{
"description": "PO_Project version description",
"name": "PO_Project Version"
}
```

4. Enter the project ID (`4945`) in the **projectId** field.
5. Run the API to create a version.
6. Review the response body to get the version ID, which is `4946` in this case.
7. Note the version ID.
   This version ID is used in all the required operations explained in this article.

The next step is to register an object to the created project and version.

Register a File Object

Register a file object so that you can generate more data for it.

Note: For more information about working with file object registration in the UI, see Register File Objects in the UI section.

1. Access the following CA TDM Portal API:

```
POST https://<server>:<host>/TDMModelService/api/ca/v1/objects
```

   Note: For more information about this API, see the "object-controller: Interface for Objects" section at https://<server>:<port>/TDMModelService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMModelService/swagger-ui.html.

2. Enter the following information to register an object of type XSD for this example:
   - **Authorization**
     For this example, the value is as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
RQ5l4Ro
```

   - **projectId**
     For the example, the value is `4945` .
   - **versionId**
     For the example, the value is `4946` .
   - **body**
     For this example, the value is as follows:

```
{
"objectName": "PO_Object",
"objectType": "XSD",
"fileEncoding": "UTF-8"
}
```

   - **files**
     For this example, the value is `C:\PO_Schema.xsd` (location where the file is present).
     If you are using Swagger, you cannot use this field to upload the file, because file upload does not work in Swagger. Ensure that you use some other client application for this API.

Note: If a specific field is not applicable for your object type, you can ignore that field. For example, the **responseFile** field is not applicable for XSD, JSON, and XML types. Therefore, you can keep it blank for these object types.

3. Run the API to register the object.
4. Review the response body and note the object ID, which is

   `2389`

   in this case.

The next step is to create a data generator.

Create a Data Generator

A data generator lets you create data generation rules and publish data.

Note: For information about working with data generators in the UI, see Create Data Generator in the UI section.

1. Access the following CA TDM Portal API:

   `POST https://<server>:<host>/TDMGeneratorService/api/ca/v1/generators`

   Note: For more information about this API, see the "data-generator-controller: Interface for data generator" section at https://<server>:<port>/TDMGeneratorService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMGeneratorService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**

     For this example, the value is as follows:

     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **generatorInfo**

     For the example, the value is as follows:

     ```
     {
     "description": "PO_Generator",
     "name": "PO_Generator",
     "projectId": 4945,
     "projectName": "PO_Project",
     "versionId": 4946,
     "versionName": "1.0"
     }
     ```
   - **projectId**

     For the example, the value is `4945`.
   - **versionId**

     For the example, the value is `4946`.
3. Run the API to create a data generator.
4. Review the response body and note the generator ID, which is

   `4955`

   in this case.

The next step is to create derived objects and register them.

Create and Register Derived Objects

You create derived objects to convert non-relational data model into a relational model.

Note: For more information about working with derived objects in the UI, see Create and Register Derived Objects in the UI section.

1. Access the following CA TDM Portal API:
   ```
   POST https://<server>:<host>/TDMModelService/api/ca/v1/objects/{objectId}/actions/derive
   ```
   Note: For more information about this API, see the "object-controller: Interface for Objects" section at https://<server>:<port>/TDMModelService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMModelService/swagger-ui.html.
2. Enter information in the following fields for the XSD object type (used for this example):
   - **Authorization**
     For this example, the value is as follows:
     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **objectId**
     For the example, the value is `2389`.
   - **projectId**
     For the example, the value is `4945`.
   - **versionId**
     For the example, the value is `4946`.
   - **async**
     For the example, the value is `true`.
   - **rootElementName**
     For the example, the value is `PO`.
   - **generateForeignKeys**
     For the example, the value is `true`.
   - **cycleRecursionDepth**
     For the example, the value is `2`.
   - **importObjectData**
     For the example, the value is `true`.
   - **profileName**
     For the example, the value is `PO_Profile`.

   Note: Leave the fields that are not applicable for your file object type.
3. Run the API.
4. Review the response body and note the job ID, which is `1223` in this case.

The next step is to get the status of the submitted job.

Get the Status of the Submitted Job (Derived Object)

After you submit a job, get the status of the job to verify that the job is executed without any issue.

1. Access the following CA TDM Portal API:
   ```
   GET https://<server>:<host>/TDMJobService/api/ca/v1/jobs/{jobId}
   ```
   Note: For more information about this API, see the "job-engine-service-controller: Interface for requests" section at https://<server>:<port>/TDMJobService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMJobService/swagger-ui.html.
2. Enter the security token in the **Authorization** field as follows:
   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
   RQ5l4Ro
   ```

3. Enter the job ID (request ID) `1223` in the **jobId** field.
4. Run the API.
5. Review the response body and verify that the status is shown as `Completed`.

The next step is to get the list of derived objects.

Get the List of Derived Objects

Get the list of derived objects to verify that all the tables are created in the database successfully.

1. Access the following CA TDM Portal API:
   `GET https://<server>:<host>/TDMModelService/api/ca/v1/objects/{objectId}/derivedObjects`
   Note: For more information about this API, see the "object-controller: Interface for Objects" section at https://<server>:<port>/TDMModelService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMModelService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the value is as follows:
     `Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ RQ5l4Ro`
   - **objectId**
     For the example, the value is `2389`.
   - **projectId**
     For the example, the value is `4945`.
   - **versionId**
     For the example, the value is `4946`.
3. Run the API.
4. Review the response body to find the list of all derived objects that are related to the specified file object.

The next step is to import sample data into derived objects.

Import the Sample Data

Import the sample data into derived objects so that you can use the same sample data to generate more data.

Note: For more information about working with the import operation in the UI, see Perform Actions on Derived Objects in the UI section.

1. Access the following CA TDM Portal API:
   `POST https://<server>:<host>/TDMModelService/api/ca/v1/objects/{objectId}/actions/import`
   Note: For more information about this API, see the "object-controller: Interface for Objects" section at https://<server>:<port>/TDMModelService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMModelService/swagger-ui.html.
2. Enter information in the following fields for the XML file that is used for this example:
   - **Authorization**
     For this example, the value is as follows:
     `Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ RQ5l4Ro`
   - **objectId**

For the example, the value is `2389` .
- **projectId**
  For the example, the value is `4945` .
- **versionId**
  For the example, the value is `4946` .
- **async**
  For the example, the value is `true` .
- **dataEncoding**
  For the example, the value is `UTF-8` .
- **profileName**
  For the example, the value is `PO_Profile` .
- **files**
  For the example, the value is `C:\PO.xml` .
  If you are using Swagger, you cannot use this field to upload the file, because file upload does not work in Swagger. Ensure that you use some other client application for this API.
- **importToGenerator**
  For the example, the value is `true` .
  Note: We recommend that you enable this option to ensure that the sample data is also imported into the specified data generator. This approach lets you automatically populate the data into the data generator, which otherwise is a manual effort.
- **generatorId**
  For the example, the value is `4955` .

3. Run the API.
4. Review the response body and note the job ID, which is `1224` in this case.
5. Run the API (as explained previously) to get the status of the job based on the job ID `1224` .
6. Verify that the status is shown as `Completed` .
   You have successfully imported the sample data into derived objects.

The next step is write data generation rules.

Write Data Generation Rules

To successfully write data generation rules, you can find the data generation functions and variables that are available for you to use in expressions. You can then write and validate data generation expressions.

Note: For more information about working with the data generation rules in the UI, see Create Data Generation Rules in the UI section.

Get Data Generator Functions

1. Access the following CA TDM Portal API to retrieve the list of all the data generator functions:
   ```
   GET https://<server>:<host>/TDMGeneratorService/api/ca/v1/generatorFunctions
   ```
   Note: For more information about this API, see the "data-painter-controller: Interface for data painter" section at https://<server>:<port>/TDMGeneratorService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMGeneratorService/swagger-ui.html.
2. Enter the security token in the **Authorization** field as follows:
   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
   RQ5l4Ro
   ```
3. Run the API.
4. Review the response body to view the available functions that you can use while creating data generation rules.

Get Data Generator Variables

1. Access the following CA TDM Portal API to retrieve the list of all the data generator variables:
   ```
   GET https://<server>:<host>/TDMGeneratorService/api/ca/v1/generators/{generatorId}/variables
   ```
   Note: For more information about this API, see the "variable-controller : Interface for variables" section at https://<server>:<port>/TDMGeneratorService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMGeneratorService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the value is as follows:
     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **generatorId**
     For this example, the value is `4955` .
   - **projectId**
     For this example, the value is `4945` .
   - **versionId**
     For this example, the value is `4946` .
3. Run the API.
4. Review the response body to view the available variables that you can use while creating data generation rules.

Validate Expressions

1. Access the following CA TDM Portal API to validate the data generation expressions that you write as part of the payload to this API:
   ```
   POST https://<server>:<host>/TDMGeneratorService/api/ca/v1/generators/evaluateExpression
   ```
   Note: For more information about this API, see the "data-painter-controller: Interface for data painter" section at https://<server>:<port>/TDMGeneratorService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMGeneratorService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the value is as follows:
     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojUxm8ENTeRRr
     RQ5l4Ro
     ```
   - **requestBean**
     You enter all expression-related information in this field. You can enter only one expression for a column. For the example used in this article, the following snippet shows the expression that was used for a column in a table:
     ```
     {
     "expression": "@randlov(0,@seedlist(US City)@)@",
      "levelID": 4955,
      "listTildes": [],
      "metaTable": {
      "columns": [
      {
      "columnName": "city",
      "dataType": "nvarchar",
      "length": 0
     ```

```
}
],
"tableName": "billTo"
},
"projectID": 4945,
"versionID": 4946
}
```

When you run this API, the expression generates random US city names depending on the seedlist that is provided in the expression. The `levelID` parameter represents the data generator ID. The API returns only the resolved value; it does not validate the value depending on the column constraints (for example, mandatory column, data type, and so on).

3. Run the API.
4. Review the response body to view that the generated value is valid.

Get the Table IDs and Column Names

Identify the table IDs and corresponding columns where you want to add data generation rules.

1. Access the following CA TDM Portal API:

   `GET https://<server>:<host>/TDMGeneratorService/api/ca/v1/generators/{generatorId}/tables`

   Note: For more information about this API, see the "data-generator-controller: Interface for data generator" section at https://<server>:<port>/TDMGeneratorService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMGeneratorService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the value is as follows:

     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **generatorId**
     For this example, the value is `4955` .
   - **projectId**
     For this example, the value is `4945` .
   - **versionId**
     For this example, the value is `4946` .
3. Run the API.
4. Review the response body to note the table IDs, table names, column IDs, and column names where you want to add data generation rules. You use this information when you add data generation rules to the columns. Each table includes column names and their IDs.

Add Data Generator Definition

You add data generation rules to all the required columns in a specific table by running the following API. Ensure that you run this API separately for each table.

1. Access the following CA TDM Portal API to add data generator definition for a table:

   `POST https://<server>:<host>/TDMGeneratorService/api/ca/v1/generators/{generatorId}/tables/{tableId}/definitionRows`

   Note: For more information about this API, see the "data-generator-controller: Interface for data generator" section at https://<server>:<port>/TDMGeneratorService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMGeneratorService/swagger-ui.html.

2. Enter information in the following fields:
   - **Authorization**
     For this example, the value is as follows:
     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **generatorId**
     For this example, the value is `4955` .
   - **projectId**
     For this example, the value is `4945` .
   - **versionId**
     For this example, the value is `4946` .
   - **tableId**
     For this example, the value for the `PO_tdm_root` table is `2402` .
   - **rowDefinitionDetails**
     For this example, the value for the table ID `2402` is as follows:
     ```
     {
      "definitions": [
     {
     "columnName": "SHRED_ID",
     "columnValue": "@nextval(SHRED_ID_SEQ)@"
     },
     {
     "columnName": "SHRED_GROUP_ID",
     "columnValue": "~NEXT~"
     },
     {
     "columnName": "orderDate",
     "columnValue": "~CDATE~"
     },
     {
     "columnName": "comment",
     "columnValue": "@randlov(0,@seedlist(Random Text)@)@"
     }
      ]
     }
     ```
     This snippet includes all data generations rules that you want to add to the columns in the table `PO_tdm_root` .
3. Run the API.
4. Review the response to view that the success message is displayed.
5. Run the API for other tables as required. For example, in this case, the API is run separately for each remaining table.

The next step is to publish the data.

Publish Data

After you add data generation rules to derived objects, you can publish the data.

Note: For more information about working with data publishing in the UI, see Publishing Data in the UI section.

Submit the Publish Job

1. Access the following CA TDM Portal API to submit the publish job:

   ```
   POST https://<server>:<host>/TDMJobService/api/ca/v1/jobs
   ```

   Note: For more information about this API, see the "job-engine-service-controller: Interface for requests" section at https://<server>:<port>/TDMJobService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMJobService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the value is as follows:

     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **jobInfo**
     For this example, the value is as follows:

     ```
     {
      "name":"Publish_PO_PO_Generator",
      "description":"Publish to PO using PO_Generator",
      "projectId":4945,
      "type":"PUBLISHJOB",
      "origin":"generation",
      "scheduledTime":1464952420271,
      "email":"abcde02@axy.com",
      "jobs":[
     ],
      "parameters":{
      "generatorId":4955,
      "jobType":"PUBLISH",
      "title":"Publish to PO using PO_Generator",
      "publishTo":"TGT",
      "target":"PO_Object_2398",
      "dataTargetProfile":"PO_Profile",
      "repeatCount":1,
      "tables":[
      {
      "tableNo":1,
      "tableName":"PO_tdm_root",
      "status":1
      },
      {
      "tableNo":2,
      "tableName":"billTo",
      "status":1
      },
      {
      "tableNo":3,
      "tableName":"items",
      "status":1
     ```

```
},
{
"tableNo":4,
"tableName":"shipTo",
"status":1
},
{
"tableNo":5,
"tableName":"item",
"status":1
}
],
"email":"abcde02@axy.com"
}
}
```

Note: Ensure that you use the correct format for the scheduledTime parameter as specified in the payload. You can use any available online tool to convert the value to the suggested format.

3. Run the API.
4. Review the response body to note the submitted job ID, which is 1281 in this case. You use this job ID to know the whether publishing is successful.
5. Get the status of the submitted job based on the job ID.

Get the Status of the Submitted Job (Publish)

After you submit the job, get its status to know whether the job is done without any issue.

1. Access the following CA TDM Portal API:
   `GET https://<server>:<host>/TDMJobService/api/ca/v1/jobs/{jobId}`

   Note: For more information about this API, see the "job-engine-service-controller: Interface for requests" section at https://<server>:<port>/TDMJobService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMJobService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the value is as follows:
     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **jobId**
     For this example, the value is `1281` .
3. Run the API.
4. Review the response body to know that the status of the job is shown as `Completed` .

You have successfully published the data. You can query the database to verify that the generated data is available in the tables (derived).

Export Data

Export the data into appropriate file formats. You can then use the exported files in your application environment to perform various test scenarios.

Note: For more information about working with the export data operation in the UI, see Perform Actions on Derived Objects in the UI section.

1. Access the following CA TDM Portal API to export data into appropriate file formats:

   ```
   POST https://<server>:<host>/TDMModelService/api/ca/v1/objects/{objectId}/actions/export
   ```

   Note: For more information about this API, see the "object-controller: Interface for Objects" section at https://<server>:<port>/TDMModelService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMModelService/swagger-ui.html.
2. Enter information in the following fields depending on your object type:
   - **Authorization**
     For this example, the value is as follows:

     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **objectId**
     For this example, the value is `2389` .
   - **projectId**
     For this example, the value is `4945` .
   - **versionId**
     For this example, the value is `4946` .
   - **async**
     For this example, the value is `true` .
   - **dataEncoding**
     For this example, the value is `UTF-8` .
   - **profileName**
     For this example, the value is

     ```
     PO_Profile
     ```

     .
   - **exportIntoMultipleFiles**
     For this example, the value is

     ```
     true
     ```

     .
   - **baseFileName**
     For this example, the value is `CATDMShredder` .
   - **requireDataIndentation**
     For this example, the value is

     ```
     true
     ```

     .
   - **includeXmlDeclaration**
     For this example, the value is

     ```
     true
     ```

     .
   - **includeStandaloneAttribute**
     For this example, the value is

     ```
     false
     ```

     .
   - **honorUnqualifiedForms**
     For this example, the value is

     ```
     true
     ```

.

- **updateVirtualService**
  For this example, the value is
  `false`
  .
- **publishFiles**
  For this example, the value is
  `false`
  .

3. Run the API.
4. Review the response body to find the job ID, which is 1281 in this case.
5. Get the status of the submitted job based on the job ID as mentioned previously.
   You have successfully exported the data.

In this example, Test Data Engineers used the CA TDM Portal APIs to generate synthetic data for an application that uses a non-relational data source.

Use APIs to Create, Manage, and Use Variables

This article explains with the help of an example about how to use exposed CA TDM Portal APIs to create, manage, and use variables in the CA TDM Portal. To demonstrate the usage of variables in context of APIs, this article uses the same example that is explained in Use APIs to Prepare Test Data for Non-Relational Sources. Therefore, appropriate references have been made to the original article wherever steps are the same for the mentioned tasks.

You can create and manage variables at these levels: repository, project, version, and generator. The scope of the variable varies depending on the level at which it is created. The complete process is as follows; you perform all these tasks with the help of the exposed APIs:

Note: For information about specific concepts (for example, project, data generator, data painter), see the relevant sections in this documentation.

This page refers to the following API Services:

- TDMProjectService
- TDMGeneratorService

Get a Security Token

Use the login API to log in and generate a security token. You use your CA TDM Portal login credentials to generate the security token. You can then use the same security token to perform all other operations. The security token remains valid for 24 hours.

For more information about how to get a security token, see the "Get a Security Token" section in Use APIs to Prepare Test Data for Non-Relational Sources.

Create a Connection Profile

A connection profile stores the details about a connection to a database system. Create a connection profile to connect to the source or target databases.

For more information about how to create a connection profile, see the "Create a Connection Profile" section Use APIs to Prepare Test Data for Non-Relational Sources.

Create a Project

All operations that you perform to prepare test data for non-relational data sources take place in context of a specific CA TDM project.

For more information about how to create a project, see the "Create a Project" section in Use APIs to Prepare Test Data for Non-Relational Sources.

Create a Version

Each project that you create in the CA TDM Portal must get associated with at least one version.

For more information about how to create a version for a project, see the "Create a Version" section in Use APIs to Prepare Test Data for Non-Relational Sources.

Register a File Object

In the CA TDM Portal, you register file objects so that you can perform various data manipulation operations (for example, data generation) on them. You register file objects in context of a project and its version.

For more information about how to register a file object, see the "Register a File Object" section in Use APIs to Prepare Test Data for Non-Relational Sources.

Create a Data Generator

A data generator lets you create data generation rules and publish data.

For more information about how to create a data generator, see the "Create a Data Generator" section in Use APIs to Prepare Test Data for Non-Relational Sources.

Create and Register Derived Objects

You create derived objects to convert non-relational data model into a relational model.

For more information about how to create and register derived objects, see the "Create and Register Derived Objects" section in Use APIs to Prepare Test Data for Non-Relational Sources.

Import the Sample Data

Import the sample data into derived objects so that you can use the same sample data to generate more data.

For more information about how to create a project, see the "Create a Project" section in Use APIs to Prepare Test Data for Non-Relational Sources.

Work with Variables

This section includes detailed information about how to use APIs to create variables at different levels. After you create variables, you can use them as part of your data generation rules. Those variables are then resolved at the time of publishing.

Note: This section also includes information about how you can update and delete variables.

Variables at the Repository Level

You can perform the following actions for variables at the repository level:

- Create a repository variable (POST).
- Get all variables available at the repository level (GET).
- Get information about a specific repository variable (GET).
- Update the variable information (PUT).
- Delete a repository variable (DELETE).

Note: For more information about working with variables in the UI, see the Create and Manage Variables in the UI section.

Create a Variable at the Repository Level

A variable created at the repository level is accessible at the repository, project, version, and generator levels.

1. Access the following CA TDM Portal API to create a variable at the repository level:
   ```
   POST  https:/server-po:8443/TestDataManager/api/ca/v1/variables
   ```
   Note: For more information about this API, see the "variable-controller: Variable Controller" section at https://<server>:<port>/TestDataManager/swagger-ui.html. For the example in this article, the URL is https:/server-po:8443/TestDataManager/swagger-ui.html.
2. Enter the security token in the **Authorization** field as follows:
   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
   RQ5l4Ro
   ```
3. Enter the variable information in the **variableInfo** field. For this example, the value is as follows:
   ```
   {
    "defaultValue": "Repository Variable",
    "description": "Variable at the repository level",
    "displayType": "TextBox",
    "helpMessage": "This is a new variable.",
    "isDisplayOnly": true,
    "isOptional": true,
    "name": "Variable_Repository",
    "resolvePriorToPublish": true,
    "type": "string"
   }
   ```
4. Run the API to create a variable at the repository level.
5. Review the response body to verify that the variable `Variable_Repository` is created successfully. The following snippet shows the generated response body:
   ```
   {
    "name": "Variable_Repository",
    "scope": "Repository",
    "description": "Variable at the repository level",
    "defaultValue": "Repository Variable",
    "type": "STRING",
    "resolvePriorToPublish": true,
    "validation": null,
    "isDisplayOnly": true,
    "displayType": "TextBox",
    "helpMessage": "This is a new variable.",
    "listDefinition": null,
   ```

```
  "isOptional": true
 }
```

Note the variable name to use in subsequent operations.

Get All Variables Available in the Repository

You can retrieve all variables that are available in the repository.

1. a. Access the following CA TDM Portal API:

   ```
   GET  https:/server-po:8443/TestDataManager/api/ca/v1/variables
   ```

   Note: For more information about this API, see the "variable-controller: Variable Controller" section at https://<server>:<port>/TestDataManager/swagger-ui.html. For the example in this article, the URL is https:/server-po:8443/TestDataManager/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:

     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **page**
     For this example, this field is left blank.
     However, if you want to view the elements present in a specific page, you can specify the value in the **page** field; `0` represents the first page. For example, if the total number of elements is `30` and you specify the value as `0` , the elements included in the first page are shown, which in this case are the first `20` elements. If you specify the value as `1` , the elements included in the second page are displayed, which in this case, are the remaining `10` elements.
   - **size**
     For this example, this field is left blank. Moreover, this field becomes applicable only if you use the **page** field. However, if you want to view a specific number of elements in each page, you can specify the value in the **size** field. For example, if the total number of elements is 30 and you want to view 7 elements in each page. So, the 30 elements are divided among the required number of pages, ensuring that 7 elements are included in each page. The last page always included whatever is left after equally distributing the elements. In this case, 30 elements are divided among 5 pages. The first 4 pages include 7 elements each, and the last page includes the remaining 2 elements.
   - **searchText**
     For this example, this field is left blank.
     However, if you want to use a string to search for some specific variables and then display only those variables in the result, you can enter the required value in this field. For example, if you use the value as `ver` , only those variables that include `ver` in the variable name, variable description, or default value are displayed in the result.
3. Run the API to get information about all variables available in the repository.
4. Review the response body to verify different variables that are available. The following snippet shows the generated response body:

   ```
   {
     "numberOfElements": 3,
     "totalNumberOfElements": 3,
     "elements": [
       ....
       .....
       {
         "name": "Variable_Repository",
         "scope": "Repository",
   ```

```
        "description": "Variable at the repository level",
        "defaultValue": "Repository Variable",
        "type": "STRING",
        "resolvePriorToPublish": true,
        "validation": null,
        "isDisplayOnly": true,
        "displayType": "TextBox",
        "helpMessage": "This is a new variable.",
        "listDefinition": null,
        "isOptional": true
      }
    ]
}
```

Note that the variable `Variable_Repository` that you created is also present in the response body.

Get Information About a Specific Repository Variable

You can retrieve information about a specific variable in the repository.

1. Access the following CA TDM Portal API:
   `GET https:/server-po:8443/TestDataManager/api/ca/v1/variables/{variableName}`

   Note: For more information about this API, see the "variable-controller: Variable Controller" section at https://<server>:<port>/TestDataManager/swagger-ui.html. For the example in this article, the URL is https:/server-po:8443/TestDataManager/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:
     `Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ RQ5l4Ro`
   - **variableName**
     For this example, the variable for which you want to find the information is `Variable_Repository`.
3. Run the API to get information about the variable `Variable_Repository`.
4. Review the response body to note the required information about the variable. The following snippet shows the generated response body:

```
{
  "name": "Variable_Repository",
  "scope": "Repository",
  "description": "Variable at the repository level",
  "defaultValue": "Repository Variable",
  "type": "STRING",
  "resolvePriorToPublish": true,
  "validation": null,
  "isDisplayOnly": true,
  "displayType": "TextBox",
  "helpMessage": "This is a new variable.",
  "listDefinition": null,
  "isOptional": true
```

```
}
```

Update the Variable Information

If you want to update the information of a variable available in the repository, you can do so.

1. Access the following CA TDM Portal API:
   ```
   PUT  https:/server-po:8443/TestDataManager/api/ca/v1/variables/{variableName}
   ```
   Note: For more information about this API, see the "variable-controller: Variable Controller" section at https://<server>:<port>/TestDataManager/swagger-ui.html. For the example in this article, the URL is https:/server-po:8443/TestDataManager/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:
     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **variableName**
     For this example, the variable that you want to update is `Variable_Repository`.
   - **variableInfo**
     For this example, the updated variable information is as follows:
     ```
     {
      "defaultValue": "Repository Variable Updated",
      "description": "Variable at the repository level updated",
      "displayType": "TextBox",
      "helpMessage": "This is an old variable.",
      "isDisplayOnly": false,
      "isOptional": false,
      "name": "Variable_Repository",
      "resolvePriorToPublish": false,
      "scope": "Repository",
      "type": "string"
     }
     ```
3. Run the API to update the variable information.
4. Review the response body to verify that the updated values are now available. The following snippet shows the generated response body:
   ```
   {
     "name": "Variable_Repository",
     "scope": "Repository",
     "description": "Variable at the repository level updated",
     "defaultValue": "Repository Variable Updated",
     "type": "STRING",
     "resolvePriorToPublish": false,
     "validation": null,
     "isDisplayOnly": false,
     "displayType": "TextBox",
     "helpMessage": "This is an old variable.",
     "listDefinition": null,
   ```

```
    "isOptional": false
  }
```

Note that the parameters now include the updated value.

Delete a Repository Variable

Delete a repository variable if you no longer need it.

1. Access the following CA TDM Portal API:

   `DELETE https:/server-po:8443/TestDataManager/api/ca/v1/variables/{variableName}`

   Note: For more information about this API, see the "variable-controller: Variable Controller" section at https://<server>:<port>/TestDataManager/swagger-ui.html. For the example in this article, the URL is https:/server-po:8443/TestDataManager/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:

     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **variableName**
     For this example, the variable that you want to delete is `Variable_Repository`.
3. Run the API to delete the variable from the repository. The response body in this case does not include any content, so you can use the GET all variables API to find whether the variable is deleted from the project.
4. Use the GET /api/ca/v1/variables API to verify that the `Variable_Repository` variable is no longer available in the repository. The following snippet is generated:

```
{
  "numberOfElements": 2,
  "totalNumberOfElements": 2,
  "elements": [
    {
      "name": "Variable2",
      "scope": "Repository",
      "description": "Variable at the repository level",
      "defaultValue": "Ver",
      "type": "STRING",
      "resolvePriorToPublish": true,
      "validation": null,
      "isDisplayOnly": false,
      "displayType": "TextBox",
      "helpMessage": "This is a repo variable.",
      "listDefinition": null,
      "isOptional": false
    },
    {
      "name": "Variable3",
      "scope": "Repository",
      "description": "Variable at the repository level",
      "defaultValue": "Variable",
```

```
        "type": "STRING",
        "resolvePriorToPublish": true,
        "validation": null,
        "isDisplayOnly": false,
        "displayType": "TextBox",
        "helpMessage": null,
        "listDefinition": null,
        "isOptional": false
      }
    ]
  }
```

Note that

`Variable_Repository`

is no longer available in the generated response. Also, the total number of elements is now 2 instead of the original 3.

Variables at the Project Level

You can perform the following actions for variables at the project level:

- Create a project variable (POST).
- Get all variables available for a project (GET).
- Get information about a specific project variable (GET).
- Update the variable information (PUT).
- Delete a project variable (DELETE).

Note: For more information about working with variables in the UI, see the Create and Manage Variables in the UI section.

Create a Variable at the Project Level

A variable created at the project level is accessible at the project, version, and generator levels.

1. Access the following CA TDM Portal API to create a variable at the project level:

   `POST https://server-po:8443/TDMProjectService/api/ca/v1/projects/{projectId}/variables`

   Note: For more information about this API, see the "variable-controller : Interface for variables" section at https://<server>:<port>/TDMProjectService/swagger-ui.html. For the example in this article, the URL is https:/server-po:8443/TDMProjectService/swagger-ui.html.
2. Enter the security token in the **Authorization** field as follows:

   `Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vRQ5l4Ro`
3. Enter the ID of the project for which you want to create a variable. For this example, the **projectId** value is `5429` .
4. Enter the variable information in the **variableInfo** field. For this example, the value is as follows:

```
{
 "defaultValue": "@nextval(SHRED_ID_SEQ)@",
 "description": "This variable includes an expression that generates the Shred ID
 sequence.",
 "displayType": "TextBox",
 "helpMessage": "Shred ID generation.",
```

```
  "isDisplayOnly": true,
  "isOptional": true,
  "name": "SequenceVar",
  "resolvePriorToPublish": true,
  "scope": "string",
  "type": "string"
 }
```

5. Run the API to create a variable at the project level.
6. Review the response body to verify that the variable SequenceVar is created successfully for the project `5429` . The following snippet shows the generated response body:

```
{
 "name": "SequenceVar",
 "scope": "Project",
 "description": "This variable includes an expression that generates the Shred ID
 sequence.",
 "defaultValue": "@nextval(SHRED_ID_SEQ)@",
 "type": "STRING",
 "resolvePriorToPublish": true,
 "validation": null,
 "isDisplayOnly": true,
 "displayType": "TextBox",
 "helpMessage": "Shred ID generation.",
 "listDefinition": null,
 "isOptional": true
}
```

Note the variable name to use in subsequent operations.

Get All Variables Available for a Project

You can retrieve all variables that are available for a specific project.

1. Access the following CA TDM Portal API:

   ```
   GET https://server-po:8443/TDMProjectService/api/ca/v1/projects/{projectId}/variables
   ```

   Note: For more information about this API, see the "variable-controller : Interface for variables" section at https://<server>:<port>/TDMProjectService/swagger-ui.html. For the example in this article, the URL is https:/server-po:8443/TDMProjectService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:

     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **projectId**
     For this example, the project ID value is `5429` .
   - **page**
     For this example, this field is left blank.
     However, if you want to view the elements present in a specific page, you can specify the value in the **page** field; `0` represents the first page. For example, if the total number of elements is `30` and you specify the value as `0` , the

elements included in the first page are shown, which in this case are the first `20` elements. If you specify the value as `1` , the elements included in the second page are displayed, which in this case, are the remaining `10` elements.

– **size**
For this example, this field is left blank. Moreover, this field becomes applicable only if you use the **page** field. However, if you want to view a specific number of elements in each page, you can specify the value in the **size** field. For example, if the total number of elements is 30 and you want to view 7 elements in each page. So, the 30 elements are divided among the required number of pages, ensuring that 7 elements are included in each page. The last page always included whatever is left after equally distributing the elements. In this case, 30 elements are divided among 5 pages. The first 4 pages include 7 elements each, and the last page includes the remaining 2 elements.

– **searchText**
For this example, this field is left blank.
However, if you want to use a string to search for some specific variables and then display only those variables in the result, you can enter the required value in this field. For example, if you use the value as `ver` , only those variables that include `ver` in the variable name, variable description, or default value are displayed in the result.

3. Run the API to get information about all variables defined for a specific project.
4. Review the response body to verify different variables that are available for the project `5429` . The following snippet shows the generated response body:

```
{
  "numberOfElements": 12,
  "totalNumberOfElements": 12,
  "elements": [
    ....
    ....
    ....
   {
    "name": "SequenceVar",
    "scope": "Project",
    "description": "This variable includes an expression that generates the Shred ID
sequence.",
    "defaultValue": "@nextval(SHRED_ID_SEQ)@",
    "type": "STRING",
    "resolvePriorToPublish": true,
    "validation": null,
    "isDisplayOnly": true,
    "displayType": "TextBox",
    "helpMessage": "Shred ID generation.",
    "listDefinition": null,
    "isOptional": true
   }
  ]
}
```

Note that the variable `SequenceVar` that you created for this project is also present in the response body.

Get Information About a Specific Project Variable

You can retrieve information about a specific variable in a project.

1. Access the following CA TDM Portal API:

```
GET https://server-po:8443/TDMProjectService/api/ca/v1/projects/{projectId}/
variables/{variableName}
```

Note: For more information about this API, see the "variable-controller : Interface for variables" section at https://<server>:<port>/TDMProjectService/swagger-ui.html. For the example in this article, the URL is https:/server-po:8443/TDMProjectService/swagger-ui.html.

2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:
     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **projectId**
     For this example, the project ID value is `5429` .
   - **variableName**
     For this example, the variable for which you want to find the information is `SequenceVar` .
3. Run the API to get information about the variable `SequenceVar` in the project.
4. Review the response body to note the required information about the variable. The following snippet shows the generated response body:

```
{
 "name": "SequenceVar",
 "scope": "Project",
 "description": "This variable includes an expression that generates the Shred ID
 sequence.",
 "defaultValue": "@nextval(SHRED_ID_SEQ)@",
 "type": "STRING",
 "resolvePriorToPublish": true,
 "validation": null,
 "isDisplayOnly": true,
 "displayType": "TextBox",
 "helpMessage": "Shred ID generation.",
 "listDefinition": null,
 "isOptional": true
}
```

Update the Variable Information

If you want to update the variable information, you can do so.

1. Access the following CA TDM Portal API:
   ```
   PUT https://server-po:8443/TDMProjectService/api/ca/v1/projects/{projectId}/
   variables/{variableName}
   ```
   Note: For more information about this API, see the "variable-controller : Interface for variables" section at https://<server>:<port>/TDMProjectService/swagger-ui.html. For the example in this article, the URL is https:/server-po:8443/TDMProjectService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
RQ5l4Ro
```

– **projectId**For this example, the project ID value is `5429` .
– **variableName**
For this example, the variable that you want to update is `SequenceVar` .
– **variableInfo**
For this example, the updated variable information is as follows:

```
{
 "defaultValue": "@nextval(SHRED_ID_SEQ)@",
 "description": "Updated the description",
 "displayType": "TextBox",
 "helpMessage": "Shred ID generation.",
 "isDisplayOnly": true,
 "isOptional": true,
 "name": "SequenceVar",
 "resolvePriorToPublish": false,
 "scope": "string",
 "type": "string"
}
```

3. Run the API to update the variable information.
4. Review the response body to verify that the updated values are now available. The following snippet shows the generated response body:

```
{
 "name": "SequenceVar",
 "scope": "Project",
 "description": "Updated the description",
 "defaultValue": "@nextval(SHRED_ID_SEQ)@",
 "type": "STRING",
 "resolvePriorToPublish": false,
 "validation": null,
 "isDisplayOnly": true,
 "displayType": "TextBox",
 "helpMessage": "Shred ID generation.",
 "listDefinition": null,
 "isOptional": true
}
```

Note that the parameters now include the updated value.

Delete a Project Variable

Delete a project variable if you no longer need it.

1. Access the following CA TDM Portal API:

```
DELETE https://server-po:8443/TDMProjectService/api/ca/v1/projects/{projectId}/
variables/{variableName}
```

Note: For more information about this API, see the "variable-controller : Interface for variables" section at https://<server>:<port>/TDMProjectService/swagger-ui.html. For the example in this article, the URL is https:/server-po:8443/TDMProjectService/swagger-ui.html.

2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:
     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **projectId**
     For this example, the project ID value is `5429` .
   - **variableName**
     For this example, the variable that you want to delete is `SequenceVar` .
3. Run the API to delete the variable from the project `5429` . The response body in this case does not include any content, so you can use the GET all variables API to find whether the variable is deleted from the project.
4. Use the
   ```
   GET https://server-po:8443/TDMProjectService/api/ca/v1/projects/{projectId}/variables
   ```
   API to verify that the
   ```
   SequenceVar
   ```
   variable is no longer available for the project
   ```
   5429
   ```
   . The following snippet is generated:
   ```
   {
    "numberOfElements": 11,
    "totalNumberOfElements": 11,
    "elements": [
       ....
       ....
       ....
       {
         "name": "Test5",
         "scope": "Project",
         "description": "OTP project",
         "defaultValue": "5",
         "type": "STRING",
         "resolvePriorToPublish": false,
         "validation": null,
         "isDisplayOnly": false,
         "displayType": "TextBox",
         "helpMessage": "No help message.",
         "listDefinition": null,
         "isOptional": false
       }
     ]
   }
   ```
   Note that
   ```
   SequenceVar
   ```

is no longer available in the generated response. Also, the total number of elements is now 11 instead of the original 12.

Variables at the Version Level

You can perform the following actions for variables at the version level:

- Create a version variable (POST).
- Get all variables available for a version (GET).
- Get information about a specific version variable (GET).
- Update the variable information (PUT).
- Delete a version variable (DELETE).

Create a Variable at the Version Level

A variable created at the version level is accessible at the version and generator levels.

1. Access the following CA TDM Portal API to create a variable at the version level:
   ```
   POST https://server-po:8443/TDMProjectService/api/ca/v1/projects/{projectId}/
   versions/{versionId}/variables
   ```
   Note: For more information about this API, see the "variable-controller : Interface for variables" section at https://<server>:<port>/TDMProjectService/swagger-ui.html. For the example in this article, the URL is https:/server-po:8443/TDMProjectService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:
     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **projectId**
     For this example, the project ID value is `5429` .
   - **versionId**
     For this example, the version ID value is `5430` .
   - **variableInfo**
     For this example, the variable information is as follows:
     ```
     {
      "defaultValue": "Alian",
      "description": "This is a version variable.",
      "displayType": "DropDownList",
      "helpMessage": "No help message is required for this variable.",
      "listDefinition": "@aslist(@seedlist(FirstName)@)@",
      "isDisplayOnly": false,
      "isOptional": false,
      "name": "VersionVariable",
      "resolvePriorToPublish": false,
      "scope": "string",
      "type": "STRING"
     }
     ```
3. Run the API to create a variable at the version level.

4. Review the response body to verify that the variable `VersionVariable` is created successfully for the version `5430` . The following snippet shows the generated response body:

```
{
 "name": "VersionVariable",
 "scope": "Version",
 "description": "This is a version variable.",
 "defaultValue": "Alian",
 "type": "STRING",
 "resolvePriorToPublish": false,
 "validation": null,
 "isDisplayOnly": false,
 "displayType": "DropDownList",
 "helpMessage": "No help message is required for this variable.",
 "listDefinition": "@aslist(@seedlist(FirstName)@)@",
 "isOptional": false
}
```

Get All Variables Available for a Version

You can retrieve all variables that are available for a specific version.

1. Access the following CA TDM Portal API:

   ```
   GET https://server-po:8443/TDMProjectService/api/ca/v1/projects/{projectId}/versions/
   {versionId}/variables
   ```

   Note: For more information about this API, see the "variable-controller : Interface for variables" section at https://<server>:<port>/TDMProjectService/swagger-ui.html. For the example in this article, the URL is https:/server-po:8443/TDMProjectService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:

     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **projectId**
     For this example, the project ID value is `5429` .
   - **versionId**
     For this example, the version ID value is `5430` .
   - **page**
     For this example, this field is left blank.
     However, if you want to view the elements present in a specific page, you can specify the value in the **page** field; `0` represents the first page. For example, if the total number of elements is `30` and you specify the value as `0` , the elements included in the first page are shown, which in this case are the first `20` elements. If you specify the value as `1` , the elements included in the second page are displayed, which in this case, are the remaining `10` elements.
   - **size**
     For this example, this field is left blank. Moreover, this field becomes applicable only if you use the **page** field. However, if you want to view a specific number of elements in each page, you can specify the value in the **size** field. For example, if the total number of elements is 30 and you want to view 7 elements in each page. So, the 30 elements are divided among the required number of pages, ensuring that 7 elements are included in each page. The last page always included whatever is left after equally distributing the elements. In this case,

30 elements are divided among 5 pages. The first 4 pages include 7 elements each, and the last page includes the remaining 2 elements.

– **searchText**
For this example, this field is left blank.
However, if you want to use a string to search for some specific variables and then display only those variables in the result, you can enter the required value in this field. For example, if you use the value as `ver` , only those variables that include `ver` in the variable name, variable description, or default value are displayed in the result.

3. Run the API to get information about all variables in the version `5430` .
4. Review the response body to verify different variables that are available for the version. The following snippet shows the generated response body:

```
{
  "numberOfElements": 17,
  "totalNumberOfElements": 17,
  "elements": [
   ....
   ....
   {
     "name": "VersionVariable",
     "scope": "Version",
     "description": "This is a version variable.",
     "defaultValue": "Alian",
     "type": "STRING",
     "resolvePriorToPublish": false,
     "validation": null,
     "isDisplayOnly": false,
     "displayType": "DropDownList",
     "helpMessage": "No help message is required for this variable.",
     "listDefinition": "@aslist(@seedlist(FirstName)@)@",
     "isOptional": false
    }
  ]
}
```

Note that the variable
`VersionVariable`
that you created for this version is also present in the response body.

Get Information About a Specific Version Variable

You can retrieve information about a specific variable in a version.

1. Access the following CA TDM Portal API:
   `GET https://server-po:8443/TDMProjectService/api/ca/v1/projects/{projectId}/versions/{versionId}/variables/{variableName}`

   Note: For more information about this API, see the "variable-controller : Interface for variables" section at https://<server>:<port>/TDMProjectService/swagger-ui.html. For the example in this article, the URL is https:/server-po:8443/TDMProjectService/swagger-ui.html.
2. Enter information in the following fields:
   – **Authorization**

For this example, the following value is used:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
RQ5l4Ro
```

– **projectId**
For this example, the project ID value is 5429.
– **versionId**
For this example, the version ID value is 5430.
– **variableName**
For this example, the variable for which you want to find the information is `CommentVar`.

3. Run the API to get information about the variable `VersionVariable` in the version 5430.
4. Review the response body to note the required information about the variable. The following snippet shows the generated response body:

```
{
 "name": "VersionVariable",
 "scope": "Version",
 "description": "This is a version variable.",
 "defaultValue": "Alian",
 "type": "STRING",
 "resolvePriorToPublish": false,
 "validation": null,
 "isDisplayOnly": false,
 "displayType": "DropDownList",
 "helpMessage": "No help message is required for this variable.",
 "listDefinition": "@aslist(@seedlist(FirstName)@)@",
 "isOptional": false
}
```

Update the Variable Information

If you want to update the variable information, you can do so. You cannot update an inherited variable.

Note: You cannot update an inherited variable.

1. Access the following CA TDM Portal API:

```
PUT https://server-po:8443/TDMProjectService/api/ca/v1/projects/{projectId}/versions/
{versionId}/variables/{variableName}
```

Note: For more information about this API, see the "variable-controller : Interface for variables" section at https://<server>:<port>/TDMProjectService/swagger-ui.html. For the example in this article, the URL is https:/server-po:8443/TDMProjectService/swagger-ui.html.

2. Enter information in the following fields:
– **Authorization**
For this example, the following value is used:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
RQ5l4Ro
```

– **projectId**For this example, the project ID value is 5429.
– **versionId**

For this example, the version ID value is 5430.

– **variableName**
For this example, the variable that you want to update is `CommentVar` .

– **variableInfo**
For this example, the updated variable information is as follows:

```
{
 "defaultValue": "Alian",
 "description": "Updating this variable description",
 "displayType": "DropDownList",
 "helpMessage": "No help message is required for this variable.",
 "listDefinition": "@aslist(@seedlist(FirstName)@)@",
 "isDisplayOnly": false,
 "isOptional": false,
 "name": "VersionVariable",
 "resolvePriorToPublish": false,
 "scope": "string",
 "type": "STRING"
}
```

3. Run the API to update the variable information.
4. Review the response body to verify that the updated values are now available. The following snippet shows the generated response body:

```
{
 "name": "VersionVariable",
 "scope": "Version",
 "description": "Updating this variable description",
 "defaultValue": "Alian",
 "type": "STRING",
 "resolvePriorToPublish": false,
 "validation": null,
 "isDisplayOnly": false,
 "displayType": "DropDownList",
 "helpMessage": "No help message is required for this variable.",
 "listDefinition": "@aslist(@seedlist(FirstName)@)@",
 "isOptional": false
}
```

Note the updated values.

Delete a Version Variable

Delete a variable belonging to a specific version if you no longer need it.

Note: You cannot delete an inherited variable.

1. Access the following CA TDM Portal API:

```
DELETE https://server-po:8443/TDMProjectService/api/ca/v1/projects/{projectId}/
versions/{versionId}/variables/{variableName}
```

Note: For more information about this API, see the "variable-controller : Interface for variables" section at https://<server>:<port>/TDMProjectService/swagger-ui.html. For the example in this article, the URL is https:/server-po:8443/TDMProjectService/swagger-ui.html.

2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:
     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **projectId**
     For this example, the project ID value is 5429.
   - **versionId**
     For this example, the version ID value is 5430.
   - **variableName**
     For this example, the variable that you want to delete is `VersionVariable`.
3. Run the API to delete the `VersionVariable` variable from the version `5430`. The response body in this case does not include any content, so use the GET API to view whether the variable is deleted.
4. Use the
   ```
   GET https://server-po:8443
   /TDMProjectService/api/ca/v1/projects/{projectId}/version/{versionId}/variables
   ```
   API to verify that the
   ```
   CommentVar
   ```
   variable is no longer available for the version
   ```
   5430
   ```
   . The following snippet shows the generated response body:
   ```
   {
     "numberOfElements": 16,
     "totalNumberOfElements": 16,
     "elements": [
       .....
       .....
       .....
      {
        "name": "V2",
        "scope": "Version",
        "description": "Not required",
        "defaultValue": "V2",
        "type": "STRING",
        "resolvePriorToPublish": false,
        "validation": null,
        "isDisplayOnly": false,
        "displayType": "TextBox",
        "helpMessage": "Help message.",
        "listDefinition": null,
        "isOptional": false
      }
     ]
   }
   ```

Variables at the Generator Level

You can perform the following actions for variables at the generator level:

- Create a generator variable (POST).
- Get all variables available for a generator (GET).
- Get information about a specific generator variable (GET).
- Update the variable information (PUT).
- Delete a generator variable (DELETE).

Create a Variable at the Generator Level

A variable created at the generator level is accessible only at the generator level.

1. Access the following CA TDM Portal API to create a variable at the generator level:
   ```
   POST https://server-po:8443/TDMGeneratorService/api/ca/v1/generators/{generatorId}/
   variables
   ```
   Note: For more information about this API, see the "data-generator-controller : Interface for data generator" section at https://<server>:<port>/TDMGeneratorService/swagger-ui.html. For the example in this article, the URL is https:/server-po:8443/TDMGeneratorService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:
     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **generatorId**
     For this example, the generator ID is `7979` .
   - **variableInfo**
     For this example, the variable information is as follows:
     ```
     {
      "defaultValue": "2016/08/03",
      "description": "Date, DropDownList",
      "displayType": "DropDownList",
      "helpMessage": "Date, DropDownList",
      "isDisplayOnly": true,
      "isOptional": true,
      "listDefinition": "@aslist(@list(2016/08/01,2016/08/03)@)@",
      "name": "Generator_Variable",
      "resolvePriorToPublish": true,
      "scope": "string",
      "type": "Date",
      "validation": "Min(2016/08/03)"
     }
     ```
   - **projectId**For this example, the project ID value is `5429` .
   - **versionId**
     For this example, the version ID value is `5430` .
3. Run the API to create a variable at the generator level.
4. Review the response body to verify that the variable `Variable_Generator` is created successfully for the generator `7979` . The following snippet shows the generated response body:

```
{
  "name": "Generator_Variable",
  "scope": "Data Pool",
  "description": "Date, DropDownList",
  "defaultValue": "2016/08/03",
  "type": "DATE",
  "resolvePriorToPublish": true,
  "validation": "Min(2016/08/03)",
  "isDisplayOnly": true,
  "displayType": "DropDownList",
  "helpMessage": "Date, DropDownList",
  "listDefinition": "@aslist(@list(2016/08/01,2016/08/03)@)@",
  "isOptional": true
}
```

Get All Variables Available for a Generator

You can retrieve all variables that are available for a specific generator.

1. Access the following CA TDM Portal API:
   `GET https://server-po:8443/TDMGeneratorService/api/ca/v1/generators/{generatorId}/variables`

   Note: For more information about this API, see the "data-generator-controller : Interface for data generator" section at https://<server>:<port>/TDMGeneratorService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMGeneratorService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:
     `Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQRQ5l4Ro`
   - **generatorId**
     For this example, the generator ID is `7979`.
   - **projectId**
     For this example, the project ID value is `5429`.
   - **versionId**
     For this example, the version ID value is `5430`.
   - **fetchSystemVariables**
     For this example, the value is set as `false`.
   - **page**
     For this example, this field is left blank.
     However, if you want to view the elements present in a specific page, you can specify the value in the **page** field; `0` represents the first page. For example, if the total number of elements is `30` and you specify the value as `0`, the elements included in the first page are shown, which in this case are the first `20` elements. If you specify the value as `1`, the elements included in the second page are displayed, which in this case, are the remaining `10` elements.
   - **size**
     For this example, this field is left blank. Moreover, this field becomes applicable only if you use the **page** field.

However, if you want to view a specific number of elements in each page, you can specify the value in the **size** field. For example, if the total number of elements is 30 and you want to view 7 elements in each page. So, the 30 elements are divided among the required number of pages, ensuring that 7 elements are included in each page. The last page always included whatever is left after equally distributing the elements. In this case, 30 elements are divided among 5 pages. The first 4 pages include 7 elements each, and the last page includes the remaining 2 elements.

– **searchText**
For this example, this field is left blank.
However, if you want to use a string to search for some specific variables and then display only those variables in the result, you can enter the required value in this field. For example, if you use the value as `ver` , only those variables that include `ver` in the variable name, variable description, or default value are displayed in the result.

3. Run the API to get information about all variables in a specific generator.
4. Review the response body to verify different variables that are available for the generator `7979` . The following snippet shows the generated response body:

```
{
  "numberOfElements": 18,
  "totalNumberOfElements": 18,
  "elements": [
    ....
    ....
    ....
    {
      "name": "V21",
      "scope": "Generator",
      "description": "This is a new variable",
      "defaultValue": "V21",
      "type": "STRING",
      "resolvePriorToPublish": false
      "validation": null,
      "isDisplayOnly": true,
      "displayType": "TextBox",
      "helpMessage": "This is a new variable",
      "listDefinition": null,
      "isOptional": true
    },
    {
      "name": "Generator_Variable",
      "scope": "Data Pool",
      "description": "Date, DropDownList",
      "defaultValue": "2016/08/03",
      "type": "DATE",
      "resolvePriorToPublish": true,
      "validation": "Min(2016/08/03)",
      "isDisplayOnly": true,
      "displayType": "DropDownList",
      "helpMessage": "Date, DropDownList",
      "listDefinition": "@aslist(@list(2016/08/01,2016/08/03)@)@",
      "isOptional": true
```

```
    },
    {
      "name": "VerVar1",
      "scope": "Generator",
      "description": "VerVar1",
      "defaultValue": "VerVar1",
      "type": "STRING",
      "resolvePriorToPublish": false
      "validation": null,
      "isDisplayOnly": true,
      "displayType": "TextBox",
      "helpMessage": "This is a sample variable",
      "listDefinition": null,
      "isOptional": true
    }
  ]
}
```


Note that the variable `Variable_Generator` that you created for this generator is also present in the response body.

Get Information About a Specific Generator Variable

You can retrieve information about a specific variable in a version.

1. Access the following CA TDM Portal API:
   `GET https://server-po:8443/TDMGeneratorService/api/ca/v1/generators/{generatorId}/variables/{variableName}`

   Note: For more information about this API, see the "data-generator-controller : Interface for data generator" section at https://<server>:<port>/TDMGeneratorService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMGeneratorService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:
     `Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQRQ5l4Ro`
   - **generatorId**
     For this example, the generator ID is `7979`.
   - **variableName**
     For this example, the variable for which you want to find the information is `Variable_Generator`.
   - **projectId**
     For this example, the project ID value is `5429`.
   - **versionId**
     For this example, the version ID value is `5430`.
3. Run the API to get information about the variable.
4. Review the response body to note the required information about the variable. The following snippet shows the generated response body:

```
{
 "name": "Generator_Variable",
 "scope": "Data Pool",
 "description": "Date, DropDownList",
 "defaultValue": "2016/08/03",
 "type": "DATE",
 "resolvePriorToPublish": true,
 "validation": "Min(2016/08/03)",
 "isDisplayOnly": true,
 "displayType": "DropDownList",
 "helpMessage": "Date, DropDownList",
 "listDefinition": "@aslist(@list(2016/08/01,2016/08/03)@)@",
 "isOptional": true
}
```

Update the Variable Information

If you want to update the variable information, you can do so.

Note: You cannot update an inherited variable.

1. Access the following CA TDM Portal API:
   `PUT https://server-po:8443/api/TDMGeneratorService/ca/v1/generators/{generatorId}/variables/{variableName}`

   Note: For more information about this API, see the "data-generator-controller : Interface for data generator" section at https://<server>:<port>/TDMGeneratorService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMGeneratorService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:
     `Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ RQ5l4Ro`
   - **generatorId**
     For this example, the generator ID is `7979`.
   - **variableName**
     For this example, the variable that you want to update is `Variable_Generator`.
   - **variableInfo**
     For this example, the updated variable information is as follows:

```
{
 "defaultValue": "2016/08/03",
 "description": "This is a correct description.",
 "displayType": "DropDownList",
 "helpMessage": "This is a date variable with a drop-down list.",
 "isDisplayOnly": true,
 "isOptional": true,
 "listDefinition": "@aslist(@list(2016/08/01,2016/08/03)@)@",
```

```
  "name": "Generator_Variable",
  "resolvePriorToPublish": true,
  "scope": "string",
  "type": "Date",
  "validation": "Min(2016/08/03)"
 }
```

- projectIdFor this example, the project ID value is `5429`.
- **versionId**
  For this example, the version ID value is `5430`.

3. Run the API to update the variable information.
4. Review the response body to verify that the updated values are now available. The following snippet shows the generated response body:

```
{
 "name": "Generator_Variable",
 "scope": "Data Pool",
 "description": "This is a correct description.",
 "defaultValue": "2016/08/03",
 "type": "DATE",
 "resolvePriorToPublish": true,
 "validation": "Min(2016/08/03)",
 "isDisplayOnly": true,
 "displayType": "DropDownList",
 "helpMessage": "This is a date variable with a drop-down list.",
 "listDefinition": "@aslist(@list(2016/08/01,2016/08/03)@)@",
 "isOptional": true
}
```

Note the updated values.

Delete a Generator Variable

Delete a variable belonging to a specific generator if you no longer need it.

Note: You cannot delete an inherited variable.

1. Access the following CA TDM Portal API:

```
DELETE https://server-po:8443/TDMGeneratorService/api/ca/v1/generators/{generatorId}/
variables/{variableName}
```

Note: For more information about this API, see the "data-generator-controller : Interface for data generator" section at https://<server>:<port>/TDMGeneratorService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMGeneratorService/swagger-ui.html.

2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
RQ5l4Ro
```

   - **generatorId**

For this example, the generator ID is `7979`.

- **variableName**
  For this example, the variable that you want to delete is `Variable_Generator`.
- **projectId**
  For this example, the project ID value is `5429`.
- **versionId**
  For this example, the version ID value is `5430`.

3. Run the API to delete the `Variable_Generator` variable from the version `5430`. The response body does not include any content in this case. Therefore, you can use the GET all variables API to know whether the variable is deleted.
4. Use the GET `https://server-po:8443/TDMGeneratorService/api/ca/v1/generators/{generatorId}/variables` API to verify that the `Variable_Generator` variable is no longer available for the generator `7979`. The following snippet shows the generated response body:

```
{
  "numberOfElements": 17,
  "totalNumberOfElements": 17,
  "elements": [
    ....
    ....
    ....
   {
      "name": "V21",
      "scope": "Generator",
      "description": "This is a new variable",
      "defaultValue": "V21",
      "type": "STRING",
      "resolvePriorToPublish": false
      "validation": null,
      "isDisplayOnly": true,
      "displayType": "TextBox",
      "helpMessage": "This is a new variable",
      "listDefinition": null,
      "isOptional": true
   },
   {
      "name": "VerVar1",
      "scope": "Generator",
      "description": "VerVar1",
      "defaultValue": "VerVar1",
      "type": "STRING",
      "resolvePriorToPublish": false
      "validation": null,
      "isDisplayOnly": true,
```

```
        "displayType": "TextBox",
        "helpMessage": "This is a sample variable",
        "listDefinition": null,
        "isOptional": true
      }
    ]
  }
```

Note that the `Variable_Generator` is removed from the response body. Also, note that the number of elements is 17 instead of the original 18, which shows that one variable has been deleted.

Write Data Generation Rules

You add data generation rules to all the required columns in a specific table by running the following API. Ensure that you run this API separately for each table. This example also uses user-defined variables, which you have already created in the aforementioned sections. For the demonstration purpose, this example uses the variables created at the project (Variable_Generator) and version (CommentVar) levels.

1. Access the following CA TDM Portal API to add data generator definition for a table:

   ```
   POST https://server-po:8443/TDMGeneratorService/api/ca/v1/generators/{generatorId}/tables/{tableId}/definitionRows
   ```

   Note: For more information about this API, see the "data-generator-controller: Interface for data generator" section at https://<server>:<port>/TDMGeneratorService/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TDMGeneratorService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the value is as follows:

     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **generatorId**
     For this example, the value is `4955` .
   - **projectId**
     For this example, the value is `4945` .
   - **versionId**
     For this example, the value is `4946` .
   - **tableId**
     For this example, the value for the `PO_tdm_root` table is `2402` .
   - **rowDefinitionDetails**
     For this example, the value for the table ID `2402` is as follows:

     ```
     {
     "definitions": [
     {
     "columnName": "SHRED_ID",
     "columnValue": "~SequenceVar~"
     },
     {
     "columnName": "SHRED_GROUP_ID",
     "columnValue": "~NEXT~"
     },
     ```

```
{
"columnName": "orderDate",
"columnValue": "~CDATE~"
},
{
"columnName": "custName",
"columnValue": "~VersionVariable~"
}
]
}
```

This snippet includes all data generations rules that you want to add to the columns in the table `PO_tdm_root`. Also, note the use of user-defined variables SequenceVar and VersionVariable.

3. Run the API.
4. Review the response to view that the success message is displayed.
5. Run the API for other tables as required. For example, in this case, the API is run separately for each remaining table (`billTo`, `shipTo`, `items`, and `item`).

Override Default Values of the Used Variables

This section includes detailed information about how to use APIs to override the default values of the variables used in data generation rules to publish data.

You can override the default values of used variables in the following ways:

- Override default variable values using a CSV file
- Override default variable values using a Generator
- Override default variable values using SQL

Override default variable values using a CSV file

1. Access the following CA TDM Portal API:

   ```
   POST  https:/server-po:8443/TDMGeneratorService/api/ca/v1/generators/{generatorId}/
   usedVariables/actions/validateFromFile
   ```

   Note: For more information about this API, see the " data-generator-controller : Interface for data generator " section at https://<server>:<port>/ TDMGeneratorService/swagger-ui.html. For this example, the URL is https:/server-po:8443/ TestDataManager/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:

     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **generatorId**
     For this example, the generator ID is 7979.
   - **project Id**
     For this example, the project ID value is 5429.
   - **version Id**
     For this example, the version ID value is 5430.
   - **File**

Browse and select the CSV file that includes new values for the used variables. For this example, the file name is "employee.csv".

3. Run the API to override the default values of variables used in generation rules.
4. Review the response body to verify the file path on the server which includes the new values of variables. The following snippet shows the generated response body:

```
{
"message": "C:\\ProgramData\\CA\\CA Test Data Manager Portal\\objects\
\projects_17518\\version_17519\\temp\\UsedVariables_1752
}
```

For more information about working overriding default values of variables in the UI, see Publish Data Using the CA TDM Portal in the UI section.

Override default variable values using Generator

1. Access the following CA TDM Portal API:
   POST https:/server-po:8443/TDMGeneratorService//api/ca/v1/generators/{generatorId}/usedVariables/actions/validateFromGenerator
   Note: For more information about this API, see the " data-generator-controller : Interface for data generator " section at https://<server>:<port>/ TDMGeneratorService/swagger-ui.html. For this example, the URL is https:/server-po:8443/TestDataManager/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:
     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **generatorId**
     For this example, the generator ID is 7979.
   - **projectId**
     For this example, the project ID value is 5429.
   - **versionId**
     For this example, the version ID value is 5430.
   - **sourceGeneratorId**
     For this example, the source generator ID is 7449.
3. Run the API to override the default values of variables used in generation rules.
4. Review the response body to verify the new values of variables. The following snippet shows the generated response body:

```
{
"message": "Variables Validated Successfully"
}
```

For more information about working overriding default values of variables in the UI, see Publish Data Using the CA TDM Portal in the UI section.

Override default variable values using SQL

1. Access the following CA TDM Portal API:
   ```
   POST  https:/server-po:8443/TDMGeneratorService/api/ca/v1/generators/{generatorId}/
   usedVariables/actions/validateFromSQL
   ```

Note: For more information about this API, see the " data-generator-controller : Interface for data generator " section at https://<server>:<port>/ TDMGeneratorService/swagger-ui.html. For this example, the URL is https:/server-po:8443/ TestDataManager/swagger-ui.html.

2. Enter information in the following fields:
   - **Authorization**
     For this example, the following value is used:
     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```
   - **generatorId**
     For this example, the generator ID is 7979.
   - **project Id**
     For this example, the project ID value is 5429.
   - **version Id**
     For this example, the version ID value is 5430.
   - **body**
     For this example, the value is as follows:
     ```
     {
       "connectionProfileName": "profileName",
       "globalRepeatCount": "~2~",
       "programId": 1,
       "targetType": "profile"
     }
     ```
3. Run the API to override the default values of variables used in generation rules.
4. Review the response body to verify the new values of variables. The following snippet shows the generated response body:
   ```
   [
     {
       "name": "city",
       "values": [
         "TX"
       ]
     },
     {
       "name": "country_code",
       "values": [
         "USA"
       ]
     },
     {
       "name": "email",
       "values": [
         "john.smith@ca.com"
       ]
     },
     {
       "name": "first_name",
   ```

```
      "values": [
        "john"
      ]
    },
    {
      "name": "last_name",
      "values": [
        "smith"
      ]
    }
  ]
```

Publish the Data

After adding data generation rules, you can publish the data so that you can generate more data and add it to the database. More data is generated based on the data generation rules that you write.

For more information about how to publish the data, see the "Publish the Data" section in Use APIs to Prepare Test Data for Non-Relational Sources.

Export the Data

Finally, you can export the generated data and use it for your testing.

For more information about how to export the data, see the "Export the Data" section in Use APIs to Prepare Test Data for Non-Relational Sources.

Use APIs to Register and Publish CSV Files

This article explains with the help of an example about how to use exposed CA TDM Portal APIs to generate test data for an application that uses data in the form of CSV files.

Consider a scenario where an organization wants to test its application that maintains information about their employees details. The organization wants to rigorously test this application using varied sets of data. However, because of the non-availability of enough sample data, it is not able to perform a comprehensive testing on the application. The CA TDM Portal can help address this situation and can generate enough sample data that the organization can use to test the application.

The example used in this article uses three sample CSV files - employee.csv, employee_address.csv and employee_creaditcard.csv to define the relational schema. This example also helps you understand how you can create data generation rules and generated data.

The complete process is as follows; you perform all these tasks with the help of the exposed APIs:

Note: For information about specific concepts (for example, project, data generator, data painter), see the relevant sections in this documentation.

This page refers to the following API Services:

- TDMModelService
- TDMGeneratorService

Get a Security Token

Use the login API to log in and generate a security token. You use your CA TDM Portal login credentials to generate the security token. You can then use the same security token to perform all other operations. The security token remains valid for 24 hours.

For more information about how to get a security token, see the "Get a Security Token" section in Use APIs to Prepare Test Data for Non-Relational Sources.

Create a Project

All operations that you perform to prepare test data for non-relational data sources take place in context of a specific CA TDM project.

For more information about how to create a project, see the "Create a Project" section in Use APIs to Prepare Test Data for Non-Relational Sources.

Create a Version

Each project that you create in the CA TDM Portal must get associated with at least one version.

For more information about how to create a version for a project, see the "Create a Version" section in Use APIs to Prepare Test Data for Non-Relational Sources.

Create a Data Generator

A data generator lets you create data generation rules and publish data.

For more information about how to create a data generator, see the "Create a Data Generator" section in Use APIs to Prepare Test Data for Non-Relational Sources.

Register CSV File

Register a file object so that you can generate more data for it.

Note: For more information about working with CSV file registration in the UI, see CSV File Type.

1. Access the following CA TDM Portal API:

   ```
   POST https://myserver:8443/TDMModelService/api/ca/v1/objects
   ```

   Note: For more information about this API, see the "object-controller: Interface for Objects" section at https://<server>:<port>/TDMModelService/swagger-ui.html. For the example in this article, the URL is https://myserver:8443/TDMModelService/swagger-ui.html.
2. Enter the following information to register an object of type CSV:
   - **Authorization**
     For this example, the value is as follows:

     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```

   - **projectId**
     For the example, the value is `2974`.
   - **versionId**
     For the example, the value is `2975`.
   - **body**
     For this example, the value is as follows:

```
{
 "objectType":"CSV",
 "headerAt":1,
 "dataStartsAt":2,
 "importData":{
 "employee":true,
 "employee_address":false,
 "employee_creditcard":true
 },
 "generatorId":2978
}
```

– **files**
  Enter the path of the CSV files that you want to register. For this example, the value is: `C:\employee.csv`

Note: If a specific field is not applicable for your object type, you can ignore that field. For example, the **responseFile** and **requestFile** fields are not applicable for CSV types. Therefore, you can keep it blank for these object types.

3. Run the API to register the object.
4. Review the response body for the list of objects. Each object is defined with an object ID. For this example, the following was the response:

```
{
 "objectId":1894,
 "projectId":2974,
 "versionId":2975,
 "objectName":"EMPLOYEE",
 "fileLocation":"C:\\ProgramData\\CA\\CA Test Data Manager Portal\\\\objects\
\projects_2974\\versions_2975\\registerCSV\\uploadedschema",
 "objectType":"DELIM",
 "fileName":"employee.csv",
 "filecount":1,
 "tableOwner":null,
 "tableColumnCount":null,
 "tableIndexCount":null,
 "tableForeignKeyCount":null,
 "tableRegisteredDBMS":null,
 "tablePrimaryKeyIndex":null,
 "tableOrder":-1,
 "parentId":null,
 "fileStatus":0,
 "fileConnectionProfileName":null,
 "fileEncoding":null,
 "rootFilePath":null,
 "jobFailureMessage":null,
 "jobId":-1,
 "programUpdated":null,
 "group":null,
```

```
"schemaLocation":null,
"noNamespaceSchemaLocation":null,
"explicitNamespaces":null,
"columns":null,
"foreignKeys":null,
"relationships":null,
"fileConnProfId":null
},
```

The next step is to write data generation rules.

Write Data Generation Rules

You add data generation rules to all the required columns in a specific table by running the following API. Ensure that you run this API separately for each table.

1. Access the following CA TDM Portal API to add data generator definition for a table:
   `POST https://myserver:8443/TDMGeneratorService/api/ca/v1/generators/{generatorId}/tables/{tableId}/definitionRows`

   Note: For more information about this API, see the "data-generator-controller: Interface for data generator" section at https://<server>:<port>/TDMGeneratorService/swagger-ui.html. For the example in this article, the URL is https://myserver:8443/TDMGeneratorService/swagger-ui.html.
2. Enter information in the following fields:
   - **Authorization**
     For this example, the value is as follows:

     ```
     Bearer
      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQ
     RQ5l4Ro
     ```

   - **generatorId**
     For this example, the value is `2978` .
   - **projectId**
     For this example, the value is `2974` .
   - **versionId**
     For this example, the value is `2975` .
   - **tableId**
     For this example, the value for the `employee` table is `1898` .
   - **rowDefinitionDetails**
     For this example, the value for the table ID `1898` is as follows:

     ```
     {
     "definitions": [
     {
     "columnName": "employee_id",
     "columnValue": "~NEXT~"
     },
     {
     "columnName": "first_name",
     ```

```
"columnValue": "@randlov(0,@seedlist(Name - India and Pakistan First)@)@"
},
{
"columnName": "last_name",
"columnValue": "@randlov(0,@seedlist(Name - Indian Last)@)@"
},
{
"columnName": "email",
"columnValue": "^first_name^.^last_name^@atsign()@ca.com"
}
 ]
}
```

This snippet includes all data generations rules that you want to add to the columns in the table `employee` .

3. Run the API.
4. Review the response to view that the row ID is created. For this example, the following was the response:

```
"rowId": 152584
```

5. Run the API for other tables as required. For example, in this case, the API is run separately for each remaining table (employee_address and employee_creditcard).

The next step is to publish the data.

Publish Data

After adding data generation rules, you can publish the data so that you can generate more data and add it to the database. More data is generated based on the data generation rules that you write.

For more information about how to publish the data, see the "Publish Data" section in Use APIs to Prepare Test Data for Non-Relational Sources.

Use APIs to Design and Consume Automated Test Data Services

This article explains with the help of an example about how to use exposed CA TDM Portal APIs to find and reserve the test data that you can use for your specific test cases.
The complete process covers the following steps.

This page refers to the following API Services:

- TestDataManager
  https://<your-tdm-server>:<your-tdm-port>/TestDataManager/swagger-ui.html
- TDMConnectionProfileService
  https://<your-tdm-server>:<your-tdm-port>/TDMConnectionProfileService/swagger-ui.html
- TDMProjectService
  https://<your-tdm-server>:<your-tdm-port>/TDMProjectService/swagger-ui.html
- TDMDataReservationService
  https://<your-tdm-server>:<your-tdm-port>/TDMDataReservationService/swagger-ui.html
- TDMFindReserveService

https://<your-tdm-server>:<your-tdm-port>/TDMFindReserveService/swagger-ui.html

Tip: For detailed information about any TDM API, see the respective swagger-ui.html subpage on your TDM server.

Get a Security Token

Use the login API to log in and generate a security token. You use your CA TDM Portal login credentials to generate the security token. You can then use the same security token to perform all other operations. The security token remains valid for 24 hours.

Follow these steps:

1. Access an application that allows you to encode your credentials to the Base64 format.
2. Enter your CA TDM Portal login credentials (in the format `username:password`) in the source field.
   Note: Ensure that the credentials have appropriate permissions to perform all the required operations.
3. Click the option to encode the credentials. The encoded Base64 format for the example is displayed as follows:
   ```
   QWRtaW5pc3RyYXRvcjptYXJtaXRl
   ```
4. Copy the encoded value.
5. Access the following CA TDM Portal API:
   ```
   POST https://server:host/TestDataManager/user/login
   ```
6. Enter the encoded value in the **Authorization** field, which is as follows for the example:
   ```
   Basic QWRtaW5pc3RyYXRvcjptYXJtaXRl
   ```
7. Run the API to get a security token for your credentials.
8. Note the value of the **token** parameter in the response body, which is as follows for the example:
   ```
   eyJhbGciOiJIUzI1NiJ9.eyJMT0dJTl9TRVNTSU9OX0lEIjoiNTA5YzA2NTMtMjgzMC00YTIxLThkNjUtNDQ0OTkxMGY5N2NjIiwic3ViIjoiQWRtaW5
   s
   ```
   You have successfully generated a security token that you can use in all the subsequent operations explained in this article.

Create a Connection Profile

Create a connection profile to connect to the source or target databases.

Note: For more information about working with connection profiles in the UI, see Create and edit Connection Profiles in the UI section.

1. Access the following CA TDM Portal API:
   ```
   POST https://server:host/TDMConnectionProfileService/api/ca/v1/connectionProfiles
   ```
2. Enter the security token in the **Authorization** field as follows:
   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJMT0dJTl9TRVNTSU9OX0lEIjoiNTA5YzA2NTMtMjgzMC00YTIxLThkNjUtNDQ0OTkxMGY5N2NjIiwic3ViIjoiQWRtaW
   s
   ```
3. Click the model schema for the **profile** parameter and specify the required connection profile details. For the example used in this article, the following information was entered:
   ```
   {
    "name": "Travel_Src",
    "description": "Travel_Source_Profile",
    "dbType": "sql server",
    "server": "abc01-xy001",
    "port": "1433",
    "instance": "SQLEXPRESS",
    "service": "",
    "database": "Travel",
   ```

```
  "schema": "dbo",
  "username": "sa",
  "password": "abcde@123"
}
```

4. Run the API.
5. Review the response body and note the connection profile name, which is Travel_Src in this case.

```
{
    "name": "Travel_Src",
    "description": "Travel_Source_Profile",
    "dbType": "sql server",
    "server": "abc01-xy001",
    "port": "1433",
    "instance": "SQLEXPRESS",
    "service": "",
    "database": "Travel",
    "schema": "dbo",
    "username": "sa",
    "password": "f-U9A0+i4mCOsDrscvP0aEVIl40Vzd0D3H5bvU+abVYsR-Kb8BNv",
    "datasourceUrl": "jdbc:sqlserver://abc01-xy001\\SQLEXPRESS:1433;database=Travel",
    "datasourceDriver": "com.microsoft.sqlserver.jdbc.SQLServerDriver",
    "created": 1585554983394,
    "modified": 1585554983394,
    "createdBy": 1,
    "additionalConnectionProperties": "",
    "integratedSecurity": false,
    "baseUrl": "jdbc:sqlserver://abc01-xy001\\SQLEXPRESS:1433",
    "connectionProperties": {
        "database": "Travel"
    }
}
```

Create a Project

All operations that you perform to prepare test data for non-relational data sources take place in context of a specific CA TDM project.

Note: For more information about working with CA TDM Portal projects in the UI, see Create and Edit Projects in the UI section.

1. Access the following CA TDM Portal API:

```
POST https://server:host/TDMProjectService/api/ca/v1/projects
```

2. Enter the security token in the **Authorization** field as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJMT0dJTl9TRVNTSU9OX0lEIjoiNTA5YzA2NTMtMjgzMC00YTIxLThkNjUtNDQ0OTkxMGY5N2NjIiwic3ViIjoiQWRtaW
s
```

3. Click the model schema for the **projectInfo** parameter and specify the required project details. For this example, the following information was entered:

```
{
    "name":"CA_Project",
    "description":"CA_Project Description",
    "inheritTables":true
}
```

4. Run the API to create a project.

5. Review the response body to get the project ID, which is 2346 in this case.

```
{
  "id": 2346
  "created": "",
  "updated": "",
  "name": "CA_Project",
  "description": "CA_Project Description",
  "dateOrder": null,
  "inheritTables": "Y",
  "timestampPrecision": 3,
  "type": "DB",
  "levels": null,
  "jobLimit": null
}
```

Create a Version

After you create a project, you must create a version for the same project.

Note: For more information about working with CA TDM Portal project versions in the UI, see Manage Project Versions in the UI section.

1. Access the following CA TDM Portal API:

```
POST https://server:host/TDMProjectService/api/ca/v1/projects/{projectId}/versions
```

2. Enter the security token in the **Authorization** field as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJMT0dJTl9TRVNTSU9OX0lEIjoiNTA5YzA2NTMtMjgzMC00YTIxLThkNjUtNDQ0OTkxMGY5N2NjIiwic3ViIjoiQWRtaW
s
```

3. Click the model schema for the **versionInfo** parameter and specify the required version details. For this example, the following information was entered:

```
{"description": "CA_Project version description","name": "CA_Project Version"}
```

4. Enter the project ID (2346) in the **projectId** field.
5. Run the API to create a version.
6. Review the response body to get the version ID, which is 2347 in this case.

```
{
  "id": 2347,
  "name": "CA_Project Version",
  "created": null,
  "description": "CA_Project version description",
  "projectName": null,
  "levelDetails": null,
  "registeredObjectCount": 0,
  "tablesUsed": null,
  "isGeneric": false
}
```

7. Note the version ID.
   This version ID is used in all the required operations explained in this article.

Share the Connection Profile

After creating a connection profile and project, share that connection profile with a group. The users will get access to the connection profile only after you share the connection profile with the group.

1. Access the following CA TDM Portal API:

   ```
   POST https://server:host/TDMConnectionProfileService/api/ca/v1/connectionProfiles/{profileName}/actions/
   grantAccess
   ```

2. Enter the security token in the Authorization field as follows:

   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJMT0dJTl9TRVNTSU9OX0lEIjoiNTA5YzA2NTMtMjgzMC00YTIxLThkNjUtNDQ0OTkxMGY5N2NjIiwic3ViIjoiQWRta
   s
   ```

3. Enter the connection profile name that you want to share in the profileName field, "Travel_Src" in this case.
4. Click the model schema for the groups parameter and specify list of user groups to whom the access needs to be granted. For the example used in this article, the following information was entered:

   ```
   [
     {
       "groupId":599,
       "groupName":"TDE - CA_Project",
       "description":"Test Data Engineers Group",
       "isAdminGroup":true,
       "projectId":2346,
       "adGroup":"",
       "securityFunctions": {},
       "adminGroup":true
     }
   ]
   ```

5. Run the API and review the response body. The response includes the group name to confirm that the specified connection profile is shared with the respective group.

   ```
   [
     {
       "groupId": 599,
       "groupName": "TDE - CA_Project",
       "description": "Test Data Engineers Group",
       "isAdminGroup": true,
       "projectId": 2346,
       "adGroup": null,
       "securityFunctions": null,
       "adminGroup": true
     }
   ]
   ```

Register Objects

In the CA TDM Portal, you register objects so that you can perform various data manipulation operations (for example, data generation) on them. You register file objects or data tables in context of a project and its version.

To allow testers to find data and reserve it for their use, you can create two kinds of model:

- Standard Model - You no longer need to register objects when using the Standard Model.
- Legacy Find & Reserve Model – You have to register objects when using a Legacy Model.

For information about registering a file object, see the "Register a File Object" section in "Use APIs to Prepare Test Data for Non-Relational Sources".

For information about registering data tables using CSV files, see "Use APIs to Register and Publish CSV Files".

NOTE
Note: In this example, we are using the "Standard Model" of Find and Reserve.

Create Environment

You can create multiple environments within the CA TDM implementation for your enterprise. Within an environment, you must specify connection profiles for each Data Source which are part of the CA TDM implementation.

Follow these steps:

1. Access the following CA TDM Portal API to create environment:
   `POST https://server:host/TDMDataReservationService/api/ca/v1/environments`
2. Enter the security token in the **Authorization** field as follows:
   `Bearer <security token>`
   For the example in this article the following value was entered:
   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJMT0dJTl9TRVNTSU9OX0lEIjoiNTA5YzA2NTMtMjgzMC00YTIxLThkNjUtNDQ0OTkxMGY5N2NjIiwic3ViIjoiQWRtaW
   s
   ```
3. Specify the following parameter values in the request body:
   a. **projectId**
      Specifies the ID of the project that you want to use to create a new environment. For this example, the project ID used is 2346.
   b. **versionId**
      Specifies the ID of the project version that you want to use to create a new environment. For this example, the version ID used is 2347.
   c. **environment**
      Click the model schema for the environment parameter and specify request body for creating an environment.
   d. **datasourcesConnectionProfiles**
      Specifies the list of connection profiles that you want to associate with the environment.
      - **connectionProfileName**
        Specifies the name of the connection profile you want to associate with the environment.
      - **connectionProfileStatus**
        Specifies the current status of the respective connection profile. Following are the valid options:
        - EXISTS
        - NOTEXISTS
        - INVALID
      - **name**
        Specifies the name of the data source that is associated with the respective connection profile.
   e. **description**
      Specifies the brief description for the environment.
   f. **name**
      Specifies the name of the environment that you want to create.

   For the example used in this article, the following information was entered:

```
{
    "datasourcesConnectionProfiles": [
        {
            "connectionProfileName": "Travel_Src",
            "connectionProfileStatus": "EXISTS",
```

```
            "name": "CAdatasource"
        }
    ],
    "description": "CAenvironment",
    "name": "CAenvironment"
}
```

4. Run the API and review the response body. Note down the environment id from the response, 1616 in this case. Creates the environment and returns the response similar to the below example:

```
{
    "id": 1616,
    "name": "CAenvironment",
    "description": "CAenvironment",
    "projectID": 2346,
    "versionID": 2347,
    "createdBy": "Administrator",
    "modifiedBy": "Administrator",
    "creationDate": "2020-03-31 11:30:50.942",
    "modifiedDate": "2020-03-31 11:30:50.942",
    "datasourcesConnectionProfiles": [
        {
            "name": "CAdatasource",
            "connectionProfileName": "Travel_Src",
            "connectionProfileStatus": "EXISTS",
            "dbType": null
        }
    ]
}
```

Note: If the parameter values you entered are not valid, you may receive one of the below errors as response for the corresponding reasons:

- **400:** Bad Request - Specific reason is included in the error message.
- **401:** Unauthorized - Invalid or expired token.
- **403:** Forbidden - User does not have permissions to access the environment.
- **404:** Not Found - Specific reason is included in the error message.
- **409:** Conflict - Environment with the specified name already exists.
- **500:** Internal Server Error - Specific reason is included in the error message.

Pre-Scan the Environment

A Test Data Engineer can pre-scan an environment to collect entity definitions with no Data Model.

Follow these steps:

1. Access the following CA TDM Portal API to pre-scan the environment

```
POST https://server:host/TDMModelService/ca/v1/datamodel/preScan
```

2. Enter the security token in the Authorization field as follows:

```
Bearer <security token>
```

For the example in this article, the following value was entered:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJMT0dJTl9TRVNTSU9OX0lEIjoiNTA5YzA2NTMtMjgzMC00YTIxLThkNjUtNDQ0OTkxMGY5N2NjIiwic3ViIjoiQWRtaW
s
```

3. Specify the following parameter values in the request body:

- projectId Specifies the ID of the project that you want to use to scan an environment. For this example, the project ID used is 2346.
- versionId Specifies the ID of the project version that you want to use to scan environment. For this example, the version ID used is 2347.
- environmentId Specifies the ID of the environment you want to scan. For this example, the environment ID used is 1616.

4. Run the API and review the response body. Submits a job to scan the environment and returns response similar to below example:

```
{
    "jobId": 3,
    "environmentId": 1616,
    "projectId": 2346,
    "projectVersionId": 2347
}
```

Create a Test Data Model

Follow these steps:

1. Access the following CA TDM Portal API to create a Test Data Model:

   `POST https://server:host/TDMDataReservationService/api/ca/v1/testDataModels`

2. Enter the security token in the **Authorization** field as follows:

   Bearer <security token>

   For the example in this article the following value was entered:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJMT0dJTl9TRVNTSU9OX0lEIjoiNTA5YzA2NTMtMjgzMC00YTIxLThkNjUtNDQ0OTkxMGY5N2NjIiwic3ViIjoiQWRtaW
s
```

3. Specify the following parameter values in the request body:
   a. **projectId**
      Specifies the ID of the project that you want to use to create the Data Model. For this example, the project ID used is 2346.
   b. **versionId**
      Specifies the ID of the project version that you want to use to create the Data Model. For this example, the version ID used is 2347.
   c. **testDataModel**
      Specifies the request body for creating a Test Data Model.
      - **dataPrefetch**
        The Data Prefetch feature allows Test Data Engineers to find test data faster. When Data Prefetch is active, data from data sources used by test data models is cached in a TDM database. The Testers' queries are evaluated in the cache, which reduces query time. The following options are valid:
        - ON_DEMAND - data prefetch on demand (data synchronization)
        - PERIODIC - periodic data prefetch (data synchronization)
        - OFF - off (no data synchronization)
      - **modelVersion**
        Specifies the data model version type. The value "new" indicates the Standard Data Model.
      - **name**
        Specifies the name for the Test Data Model you are creating.
      - **description**

Specifies a brief description for the Test Data Model that you are creating. You cannot leave the description empty.

- **Visible**
  Specifies the flag indicating whether the Data Model is visible or not. The value "true" indicates that the Data Model is visible, and the value "false" indicates that the Data Model is not visible. Only the data models that are visible can be used in Find the Test Data API.
  Default: false
- **dataSynchronized**
  Specifies whether data from data sources used by test data models needs to be synchronized. Default: false.
- **reserved**
  Specifies the flag indicating whether the Data Model show reserved rows. The value "true" indicates that the Data Model will display reserved records, and the value "false" indicates that the Data Model will not display reserved records.
  Default: false
- **modelKeys**
  Specifies the list of model keys for the root entity.
- **root**
  Specifies the Root Table object containing the following Root Entity details:
  - **displayName**
    Specifies the display of name of root table.
  - **rootEntity**
    Specifies the Root Entity (Root Table) object containing the following Test Data Model root details:
    - **dataSource**
      Specifies the name of the Data Source for the respective root entity.
    - **Name**
      Specifies the name of the Root Entity (Root Table) that you want to associate.
    - **Schema**
      Specify the Root Entity (Root Table) schema.

For the example used in this article, the following information was entered:

```
{
 "dataPrefetch": "ON_DEMAND",
 "modelVersion": "new",
 "name": "People Data Model",
 "description": "People Data Model",
 "visible": true,
 "dataSynchronized": false,
 "reserved": true,
 "modelKeys": [
   "FIRST_NAME","LAST_NAME","EMAIL", "EMPNO"
],
 "root": {
   "displayName": "People Data Model",
   "rootEntity": {
     "dataSource": "CAdatasource",
     "name": "PEOPLE",
     "schema": "dbo"
   }
}
```

4. Run the API and review the response body. Note down the id returned in the response as "testDataModelId". In this example, testDataModelId is 1620. It creates the Data Model and returns a response similar to the below example:

```
{
```

```
    "id": 1620,
    "name": "People Data Model",
    "description": "People Data Model",
    "visible": true,
    "dataSynchronized": true,
    "reserved": true,
    "modelKeys": [
        "EMPNO",
        "LAST_NAME",
        "EMAIL",
        "FIRST_NAME"
    ],
    "root": {
        "displayName": "People Data Model",
        "rootEntity": {
            "id": 1621,
            "name": "PEOPLE",
            "primaryKeys": [
                "ID"
            ],
            "dataSource": "CAdatasource",
            "schema": "dbo"
        }
    },
    "projectId": 2346,
    "versionId": 2347,
    "creationDate": "2020-04-01 09:47:48.093",
    "modifiedDate": "2020-04-01 09:47:48.093",
    "createdBy": "Administrator",
    "modifiedBy": "Administrator",
    "modelVersion": "new",
    "dataPrefetch": "ON_DEMAND",
    "dataPrefetchErrMsg": null,
    "reservationStorage": false
}
```


Note: If the parameter values you entered are not valid, you may receive one of the below errors as response for the corresponding reasons:

- **400:** Bad Request - Specific reason is included in the error message.
- **401:** Server authentication failed.
- **403:** Forbidden
- **404:** Not Found - Specific reason is included in the error message.
- **409:** Conflict - Data Model with the same name already exists.
- **500:** Internal Server Error - Specific reason is included in the error message.

Create Fields in a Test Data Model

You can create the fields in a Test Data Model which you can use to filter the data based on the criteria you specify while find the test data operation.

Note: You have repeat this API for all the attributes mentioned as Model Keys in the Create Test Data Model API.

Follow these steps:

1. Access the following CA TDM Portal API:

```
POST https://server:host/TDMDataReservationService/api/ca/v1/testDataModels/{testDataModelId}/fields
```

2. Enter the security token in the **Authorization** field as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJMT0dJTl9TRVNTSU9OX0lEIjoiNTA5YzA2NTMtMjgzMC00YTIxLThkNjUtNDQ0OTkxMGY5N2NjIiwic3ViIjoiQWRtaW
s
```

3. Enter information in the following fields as follows:
   - **testDataModelId**
     Specifies the ID of the test data model that you want to use to create a new field. For this example, the Test Data Model ID used is 1620.
   - **projectId**
     Specifies the ID of the project that includes the test data model for which you want to create a new field. For this example, the project ID used is 2346.
   - **versionId**
     Specifies the ID of the project version that includes the test data model for which you want to create a new field. For this example, the version ID used is 2347.
   - **field**
     Specifies the payload that includes the field parameters.Specify the Field parameter values that you want to use to create a field. This payload includes the following parameters:
     - **displayName**
       Specifies the display name of the field that you want to create. For this example, the display name is First Name.
     - **displayOrder**
       Specifies the order in which you want to display this field (in the form) to testers. For this example, the value is 1.
     - **displayType**
       Specifies the display type of the field. For this example, the value is TextBox.
     - **displayValues**
       Specifies the default value for the field. For this example, the value is FIRST_NAME.
     - **isVisible**
       Specifies whether you want to display this field to testers. If yes, set the value to true; otherwise, set the value to false.
     - **name**
       Specifies the name of the field. For this example, the value is FIRST_NAME.

     For this example, the association update payload is as follows.

```
{
  "displayName": "First Name",
  "displayOrder": 1,
  "displayType": "TextBox",
  "displayValues": [
    "FIRST_NAME"
  ],
  "isVisible": true,
  "name": "FIRST_NAME"
}
```

4. Run the API and review the response body. The following example response is generated:

```
{
    "associationId": null,
    "name": "FIRST_NAME",
    "displayName": "First Name",
    "displayOrder": 1,
    "isVisible": true,
```

```
        "displayType": "TextBox",
        "dataType": "varchar",
        "displayValues": [
            "FIRST_NAME"
        ],
        "id": 1622,
        "projectId": 2346,
        "versionId": 2347
    }
```

You have successfully created a field in a test data model.

Note: If the parameter values you entered are not valid, you may receive one of the below errors as response for the corresponding reasons:

- **400:** Bad Request - Specific reason is included in the error message.
- **401:** Server authentication failed.
- **403:** Forbidden
- **404:** Not Found - Specific reason is included in the error message.
- **409:** Conflict - Specific reason is included in the error message.
- **500:** Internal Server Error - Specific reason is included in the error message.

Define Associations in a Test Data Model

Follow these steps:

1. Access the following CA TDM Portal API to create an association:
   `POST https://server:host/TDMDataReservationService/api/ca/v1/testDataModels/{testDataModelId}/associations`
2. Enter the security token in the **Authorization** field as follows:
   `Bearer <security token>`
   For the example in this article the following value was entered:
   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJMT0dJTl9TRVNTSU9OX0lEIjoiNTA5YzA2NTMtMjgzMC00YTIxLThkNjUtNDQ0OTkxMGY5N2NjIiwic3ViIjoiQWRtaW
   s
   ```
3. Specify the following parameter values in the request body:
   a. **projectId**
      Specifies the ID of the project for which you want to create an association. For this example, the project ID used is 2346.
   b. **versionId**
      Specifies the ID of the project version for which you want to create an association. For this example, the version ID used is 2347.
   c. **testDataModelID**
      Specifies the ID of the test data model for which you want to create an association. For this example, the Test Data Model ID used is 1620.
   d. **forceUpdate**
      Specifies whether to create the association or not in case of conflict. Select "true", if you want to forcefully save the new association in case of a conflict.
   e. **association**
      Specifies the request body for defining entities and associations.
      - **associationType**
        Specifies the type of the relationship. Following are the valid values:

- ONE_ONE
- ONE_MANY
- MANY_ONE

- **joinFields**
  - **fieldName**
    Name of the field used in the relationship.
  - **referenceFieldName**
    Name of the reference field used in relationship.
- **name**
  Name of the association that you are creating.
- **sourceEntity**
  - **dataSource**
    Specifies the name of the data source for the entity.
  - **name**
    Specifies the name of the entity or table.
  - **schema**
    Specifies the schema name of the entity or table.
- **targetEntity**
  - **dataSource**
    Specifies the name of the data source for the entity.
  - **name**
    Specifies the name of the entity or table.
  - **schema**
    Specifies the schema name of the entity or table.

For the example used in this article, the following information was entered:

```
{
 "associationType": "ONE_ONE",
 "joinFields": [
   {
     "fieldName": "ID",
     "referenceFieldName": "PEO_ID"
   }
 ],
 "name": "PEOPLETOCREDIT_CARDS",
 "sourceEntity": {
   "dataSource": "CAdatasource",
   "name": "PEOPLE",
   "schema": "dbo"

 },
 "targetEntity": {
   "dataSource": "CAdatasource",
   "name": "CREDIT_CARDS",
   "schema": "dbo"
 }
}
```

4. Run the API and review the response body.
   Note down the associationId from the response, in this example, the associationId is 1624. It creates the Association and returns a response similar to the below example:

```
{
```

```
        "id": 1624,
        "name": "PEOPLETOCREDIT_CARDS",
        "associationType": "ONE_ONE",
        "joinFields": [
            {
                "fieldName": "ID",
                "referenceFieldName": "PEO_ID"
            }
        ],
        "sourceEntity": {
            "id": 1621,
            "name": "PEOPLE",
            "primaryKeys": [
                "ID"
            ],
            "dataSource": "CAdatasource",
            "schema": "dbo"
        },
        "targetEntity": {
            "id": 1623,
            "name": "CREDIT_CARDS",
            "primaryKeys": [
                "CARD_ID"
            ],
            "dataSource": "CAdatasource",
            "schema": "dbo",
            "reservationStorage": false
        }
    }
```


Note: If the parameter values you entered are not valid, you may receive one of the below errors as response for the corresponding reasons:

- **400:** Bad Request - Specific reason is included in the error message.
- **401:** Server authentication failed.
- **403:** Forbidden
- **404:** Not Found - Specific reason is included in the error message.
- **409:** Conflict - Association between the same entities already exists.
- **500:** Internal Server Error - Specific reason is included in the error message.

Add Associated Table Fields to the Data Model

You can add the associated table fields to the data model which you can use to filter the data based on the criteria you specify while find the test data operation. Note: Repeat this API to add additional attributes of the related table.

Follow these steps:

1. Access the following CA TDM Portal API:

   `POST https://server:host/TDMDataReservationService/api/ca/v1/testDataModels/{testDataModelId}/fields`

2. Enter the security token in the **Authorization** field as follows:

   Bearer <security token>

   For the example in this article the following value was entered:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJMT0dJTl9TRVNTSU9OX0lEIjoiNTA5YzA2NTMtMjgzMC00YTIxLThkNjUtNDQ0OTkxMGY5N2NjIiwic3ViIjoiQWRtaW
s
```

3. Enter information in the following fields as follows:
   a. **testDataModelId**
      Specifies the ID of the test data model that you want to use to create a new field. For this example, the Test Data Model ID used is 1620.
   b. **projectId**
      Specifies the ID of the project that includes the test data model for which you want to create a new field. For this example, the project ID used is 2346.
   c. **versionId**
      Specifies the ID of the project version that includes the test data model for which you want to create a new field. For this example, the version ID used is 2347.
   d. **field**
      Specifies the payload that includes the field parameters. Specify the Field parameter values that you want to use to create a field. This payload includes the following parameters:
      - **associationId**
        Specifies the ID of the association that is related to the field you want to create. For this example, the association ID used is 1624.
      - **displayName**
        Specifies the display name of the field that you want to create. For this example, the display name is CARD_NUMBER.
      - **displayOrder**
        Specifies the order in which you want to display this field (in the form) to testers. For this example, the value is 5.
      - **displayType**
        Specifies the display type of the field. For this example, the value is TextBox.
      - **displayValues**
        Specifies the default value for the field. For this example, the value is CARD_NUMBER.
      - **isVisible**
        Specifies whether you want to display this field to testers. If yes, set the value to true; otherwise, set the value to false.
      - **name**
        Specifies the name of the field. For this example, the value is CARD_NUMBER.

      For this example, the association update payload is as follows.

```
{
  "associationId": 1624,
  "name": "CARD_NUMBER",
  "displayName": "CARD_NUMBER",
  "displayOrder": 5,
  "isVisible": true,
  "displayType": " TextBox",
  "displayValues": [
    "CARD_NUMBER"
  ]
}
```

4. Run the API and review the response body. The following example response is generated:

```
{
    "associationId": 1624,
    "name": "CARD_NUMBER",
```

```
    "displayName": "CARD_NUMBER",
    "displayOrder": 5,
    "isVisible": true,
    "displayType": "TextBox",
    "dataType": "varchar",
    "displayValues": [
        "CARD_NUMBER"
    ],
    "id": 1625,
    "projectId": 2346,
    "versionId": 2347
}
```


You have successfully created a field in a test data model.

Note: If the parameter values you entered are not valid, you may receive one of the below errors as response for the corresponding reasons:

- **400:** Bad Request - Specific reason is included in the error message.
- **401:** Server authentication failed.
- **403:** Forbidden.
- **404:** Not Found - Specific reason is included in the error message.
- **409:** Conflict - Specific reason is included in the error message.
- **500:** Internal Server Error - Specific reason is included in the error message.

Find the Test Data

After creating and adding the attributes for filter the Test Data Model, you can Find the Test Data that matches your specific test criteria. Attributes from multiple Test Data Model entities can be selected or used for filtering. Custom order can be defined instead of default order by primary key.

Follow these steps:

1. Access the following CA TDM Portal API:
   `https://server:host/TDMFindReserveService/api/ca/v1/testDataModels/{testDataModelId}/actions/find`
2. Enter the security token in the **Authorization** field as follows:
   Bearer <security token>
   For the example in this article the following value was entered:

   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJMT0dJTl9TRVNTSU9OX0lEIjoiNTA5YzA2NTMtMjgzMC00YTIxLThkNjUtNDQ0OTkxMGY5N2NjIiwic3ViIjoiQWRtaW
   s
   ```

3. Enter information in the following fields as follows:
   a. **testDataModelId**
      Specifies the ID of the test data model that you want to use to create a new field. For this example, the Test Data Model ID used is 1620.
   b. **projectId**
      Specifies the ID of the project that includes the test data model for which you want to create a new field. For this example, the project ID used is 2346.
   c. **versionId**
      Specifies the ID of the project version that includes the test data model for which you want to create a new field. For this example, the version ID used is 2347.
   d. **requestBody**
      Specifies the request body for finding the test data.
      - **attributes**

Specifies the attribute for which you want to get the data
- **attributeName**
  Specifies the attribute name. For this example, FIRST_NAME is used.
- **dataSource**
  Specifies the name of the data source for the entity.
- **entityName**
  Specifies the name of the entity or table.
- **schema**
  Specifies the schema of the entity or table.

- **environmentId**
  Specifies the ID of the Environment that is associated with the corresponding test data model. In this example, it is 1616
- **filters**
  Specifies the filters to find the right data.
  - **attributeName**
    Specifies the attribute name to filter the data.
  - **dataSource**
    Specifies the name of the data source for the entity.
  - **entityName**
    Specifies the name of the entity or table.
  - **operator**
    Specifies the logical operator allowed for the corresponding filter. The following are the logical operators you can use:
    - "EQUALS"
    - "NOT_EQUAL"
    - "LESS_THAN"
    - "LESS_THAN_OR_EQUAL_TO"
    - "GREATER_THAN"
    - "GREATER_THAN_OR_EQUAL_TO"
    - "CONTAINS"
    - "BETWEEN"
    - "IN_VALUES"
    - "NOT_IN_VALUES"
    - "STARTS_WITH"
    - "ENDS_WITH"
  - **schema**
    Specifies the schema of the entity or table.
  - **values**
    Specifies the list of allowed values for the corresponding filter.
- **orderBys**
  Specifies the order in which test data should be displayed.
  - **attributeName**
    Specifies the attribute name to on which data should be sorted.
  - **dataSource**
    Specifies name of the data source of the entity or table.
  - **direction**
    Specifies the sort order. Possible values are ASC and DESC.
  - **entityName**

Specifies the name of the entity or table.

- **schema**
  Specifies the schema of the entity or table.

- **page**
  Specifies the page number that you want to retrieve from Find data results that span across multiple pages. Default: 1
- **showReservedRecords**
  Specifies the flag indicating whether to include or exclude the already reserved records in the find test data results. The value "true" indicates to include the reserved records in the results, and the value "false" indicates to exclude. Default: false
- **size**
  Specifies the number of records that you want to retrieve from Find data results to show on each page. Default: 1

For the example used in this article, the following information was entered:

```
{
 "attributes": [
   {
     "attributeName": "FIRST_NAME",
     "dataSource": "CAdatasource",
     "entityName": "PEOPLE",
     "schema": "dbo"
   },
   {
     "attributeName": "CARD_NUMBER",
     "dataSource": "CAdatasource",
     "entityName": "CREDIT_CARDS",
     "schema": "dbo"
   }
 ],
 "environmentId": 1616,
 "filters": [
   {
     "attributeName": "FIRST_NAME",
     "dataSource": "CAdatasource",
     "entityName": "PEOPLE",
     "operator": "EQUALS",
     "schema": "dbo",
     "values": [
       "ERIK"
     ]
   },
  {
     "attributeName": "TYPE",
     "dataSource": "CAdatasource",
     "entityName": "CREDIT_CARDS",
     "operator": "EQUALS",
     "schema": "dbo",
     "values": [
       "AX"
     ]
   }
 ],
```

```
  "orderBys": [
    {
      "attributeName": "FIRST_NAME",
      "dataSource": "CAdatasource",
      "direction": "ASC",
      "entityName": "PEOPLE",
      "schema": "dbo"
    }
  ],
  "page": 1,
  "showReservedRecords": false,
  "size": 5
}
```

4. Run the API and review the response body.
   Finds the data and returns the response similar to the below example:

```
{
   "totalCount": 2,
   "records": [
       {
           "attributes": [
               {
                   "attributeName": "FIRST_NAME",
                   "entityName": "PEOPLE",
                   "schema": "dbo",
                   "dataSource": "CAdatasource",
                   "value": "ERIK"
               },
               {
                   "attributeName": "CARD_NUMBER",
                   "entityName": "CREDIT_CARDS",
                   "schema": "dbo",
                   "dataSource": "CAdatasource",
                   "value": "3221-4055-0843-004"
               }
           ],
           "modelKeys": {
               "EMPNO": "3",
               "LAST_NAME": "CARDENAS",
               "EMAIL": "ECARDENA@us.broadcom.com",
               "FIRST_NAME": "ERIK"
           }
       },
       {
           "attributes": [
               {
                   "attributeName": "FIRST_NAME",
                   "entityName": "PEOPLE",
                   "schema": "dbo",
                   "dataSource": "CAdatasource",
                   "value": "ERIK"
               },
               {
```

```
                    "attributeName": "CARD_NUMBER",
                    "entityName": "CREDIT_CARDS",
                    "schema": "dbo",
                    "dataSource": "CAdatasource",
                    "value": "4343-4055-4843-814"
                }
            ],
            "modelKeys": {
                "EMPNO": "2168",
                "LAST_NAME": "FLEISCHMAN",
                "EMAIL": "EFLEISCH@us.broadcom.com",
                "FIRST_NAME": "ERIK"
            }
        }
    ],
    "page": 1,
    "size": 5
}
```


Note: If the parameter values you entered are not valid, you may receive one of the below errors as response for the corresponding reasons:

- **400:** Bad Request - Specific reason is included in the error message.
- **401:** Server authentication failed.
- **403:** Forbidden.
- **404:** Not Found - Specific reason is included in the error message.
- **500:** Internal Server Error - Specific reason is included in the error message.

Reserve the Test Data

After finding the data that matches your criteria, you can reserve the required data for your specific test cases.

Follow these steps:

1. Access the following CA TDM Portal API to reserve the test data:
   `POST https://server:host/TDMDataReservationService/api/ca/v1/reservations`
2. Enter the security token in the **Authorization** field as follows:
   Bearer <security token>
   For the example in this article the following value was entered:

   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJMT0dJTl9TRVNTSU9OX0lEIjoiNTA5YzA2NTMtMjgzMC00YTIxLThkNjUtNDQ0OTkxMGY5N2NjIiwic3ViIjoiQWRtaW
   s
   ```

3. Specify the following parameter values in the request body:
   - **projectId**
     Specifies the ID of the project where you want to perform the data reservation. For this example, the project ID used is 2346.
   - **versionId**
     Specifies the ID of the project version where you want to perform the data reservation. For this example, the version ID used is 2347.
   - **reservationInfo**
     Specifies the request body for data reservation.
     - **dataModelId**

Specifies the ID of the Data Model that associates the reservation. For this example, the data model ID used is 1620.

- **environmentId**
  Specifies the ID of the Environment to use for the reservation. For this example, the environment ID used is 1616.
- **modelVersion**
  Specifies the data model type, i.e. Standard or Legacy. For this example, the model version is "new" (Standard).
- **resErrorMessage**
  Specifies the user message that displays, if the reservation fails. Do not provide any input for this parameter.
- **reservationId**
  Specifies the ID of the reservation that is auto generated after the reservation is performed. Do not provide any input for this parameter.
- **reservationName**
  Specifies the name of the reservation.
- **reservationState**
  Specifies the state of the reservation. Default value "UNDEFINED". Do not change this value.
- **resources**
  Specifies the Reservation Resources details object containing the key value pairs of the respective Reservation.
  - **dataModelId**
    Specifies the ID of the Data Model for the corresponding resource. For this example, the data model ID used is 1620.
  - **modelKeys**
    Specifies the map of the entity key associated. For example, {"EMPNO": "2168", "LAST_NAME": "FLEISCHMAN", "EMAIL": "EFLEISCH@us.broadcom.com", "FIRST_NAME": "ERIK" }

For the example used in this article, the following information was entered:

```
{
  "dataModelId": 1620,
  "environmentId": 1616,
  "modelVersion": "new",
  "resErrorMessage": "string",
  "reservationId": 0,
  "reservationName": "CAtestdatareservation",
  "reservationState": "UNDEFINED",
  "resources": [
    {
      "dataModelId": 1620,
      "modelKeys": {"EMPNO": "2168",
                "LAST_NAME": "FLEISCHMAN",
                "EMAIL": "EFLEISCH@us.broadcom.com",
                "FIRST_NAME": "ERIK" }
    }
  ]
}
```

4. Run the API and review the response body.
   Reserved the data and returns the response similar to the below example:

```
{
    "reservationId": 403
}
```

Note: If the parameter values you entered are not valid, you may receive one of the below errors as response for the corresponding reasons:

- **202:** Reservation request has been accepted but the resources have not been reserved yet.
- **400:** Bad Request - Specific reason is included in the error message.
- **401:** Server authentication failed.
- **403:** Forbidden - User does not have permissions to perform the reservation.
- **404:** Not Found - Specific reason is included in the error message.
- **409:** Conflict - Specific reason is included in the error message.
- **500:** Internal Server Error - Specific reason is included in the error message.

Review the Reservation Status

After reserving the data, if you want to review the reservation details you can run the get reservation status API to fetch the reservation status.

Follow these steps:

1. Access the following CA TDM Portal API to get the test reservation:
   ```
   GET https://server:host/TDMDataReservationService/api/ca/v1/reservations/{reservationId}
   ```
2. Enter the security token in the **Authorization** field as follows:
   ```
   Bearer <security token>
   ```
   For the example in this article the following value was entered:
   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJMT0dJTl9TRVNTSU9OX0lEIjoiNTA5YzA2NTMtMjgzMC00YTIxLThkNjUtNDQ0OTkxMGY5N2NjIiwic3ViIjoiQWRtaW
   s
   ```
3. Specify the following parameter values in the request body:
   - **projectId**
     Specifies the ID of the project that associates the reservation you want to review. For this example, the project ID used is 2346.
   - **versionId**
     Specifies the ID of the project version that associates the reservation you want to review. For this example, the version ID used is 2347.
   - **reservationId**
     Specifies the ID of the reservation that you want to review. For this example the reservation ID used is 403.
4. Run the API and review the response body.
   Gets the reservation details and returns the response similar to the below example:

   ```
   {
      "id": "307b643a-c0fd-40b2-82a5-a1ed0712d89b",
      "name": "CAtestdatareservation",
      "state": "SUCCESS",
      "projectId": 2346,
      "versionId": 2347,
      "legacyModelId": 1620,
      "legacyEnvironmentId": 1616,
      "legacyId": 403,
      "reservedBy": 1,
      "resErrorMessage": null,
      "scheduledDate": "2020-04-02T04:27:23.150+0000",
      "expiryDate": "2120-04-02T04:27:23.150+0000",
      "releaseDate": null,
      "resources": [
          {
              "dataViewInstanceId": "ffc77ac8-bfba-49a0-8ddb-acaaa715f218",
   ```


```
            "modelKeys": [
                {
                    "EMPNO": "2168",
                    "LAST_NAME": "FLEISCHMAN",
                    "EMAIL": "EFLEISCH@us.broadcom.com",
                    "FIRST_NAME": "ERIK"
                }
            ]
        }
    ]
}
```

Note: If the parameter values you entered are not valid, you may receive one of the below errors as response for the corresponding reasons:

- **400:** Bad Request - Specific reason is included in the error message.
- **401:** Server authentication failed - Invalid or expired token.
- **403:** Forbidden - User does not have permissions to delete the reservation.
- **404:** Not Found - Reservation with the specific ID is not found.
- **500:** Internal Server Error - Specific reason is included in the error message.

Export Reserved Records as CSV

After reserving the data, you can export reserved records as CSV.

Follow these steps:

1. Access the following CA TDM Portal API to export reserved records:
   ```
   GET https://server:host/ TDMFindReserveService/api/ca/v1/reservations/{reservationId}/reservedData/
   actions/export
   ```
2. Enter the security token in the **Authorization** field as follows:
   ```
   Bearer <security token>
   ```
   For the example in this article the following value was entered:
   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJMT0dJTl9TRVNTSU9OX0lEIjoiNTA5YzA2NTMtMjgzMC00YTIxLThkNjUtNDQ0OTkxMGY5N2NjIiwic3ViIjoiQWRtaW
   s
   ```
3. Specify the following parameter values in the request body:
   - **projectId**
     Specifies the ID of the project that associates the reservations you want to export. For this example, the project ID used is 2346.
   - **versionId**
     Specifies the ID of the project version that associates the reservations you want to export. For this example, the version ID used is 2347.
   - **withRelatedTables**
     Specifies whether data from related tables should be included in export. Default: false
   - **reservationId**
     Specifies the ID of the reservation that you want to export. For this example the reservation ID used is 403.
4. 4. Run the API and download the reserved records as CSV.
   Note: If the parameter values you entered are not valid, you may receive one of the below errors as response for the corresponding reasons:

- **400:** Bad Request - Specific reason is included in the error message.
- **401:** Server authentication failed - Invalid or expired token.
- **403:** Forbidden - User does not have permissions to delete the reservation.
- **404:** Not Found - Reservation with the specific ID is not found.
- **500:** Internal Server Error - Specific reason is included in the error message.

(Optional) Release the Reservation

After using the reserved data for your testing and when you do not need the reservation any more, you can release the reservation.

Note: The released reservations are moved to "purged" state and will be permanently deleted after 30 days. This delete process runs once in every 12 hours to identify, if there are any purged reservations that are older than 30 days. You can configure these default values of delete process running interval and the number of days to keep the reservations in purged or failed state. For more information, see Configure CA TDM Portal for Deleting the Purged Reservations.

Follow these steps:

1. Access the following CA TDM Portal API to delete the test reservation:
   ```
   DELETE https://server:host/TDMDataReservationService/api/ca/v1/reservations/{reservationId}
   ```
2. Enter the security token in the **Authorization** field as follows:
   ```
   Bearer <security token>
   ```
   For the example in this article the following value was entered:
   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJMT0dJTl9TRVNTSU9OX0lEIjoiNTA5YzA2NTMtMjgzMC00YTIxLThkNjUtNDQ0OTkxMGY5N2NjIiwic3ViIjoiQWRtaW
   s
   ```
3. Specify the following parameter values in the request body:
   - **projectId**
     Specifies the ID of the project that associates the reservation you want to delete. For this example, the project ID used is 2346.
   - **versionId**
     Specifies the ID of the project version that associates the reservation you want to delete. For this example, the version ID used is 2347.
   - **reservationId**
     Specifies the ID of the reservation that you want to delete. For this example the reservation ID used is 403.
4. Run the API and review the response body.
   Reserved the data and returns the response similar to the below example:
   ```
   {"response": "Release operation completed successfully"}
   ```
   Note: If the parameter values you entered are not valid, you may receive one of the below errors as response for the corresponding reasons:
   - **204:** No content.
   - **400:** Bad Request - Specific reason is included in the error message.
   - **401:** Server authentication failed - Invalid or expired token.
   - **403:** Forbidden - User does not have permissions to delete the reservation.
   - **404:** Not Found - Reservation with the specific ID is not found.
   - **500:** Internal Server Error - Specific reason is included in the error message.

Use APIs to Manage Environments

This article explains with the help of an example about how test data engineers (TDEs) can use exposed CA TDM Portal APIs to manage environments after they create them.

This article covers the following tasks. You perform all these tasks by using the APIs. You get the information about the available environments for a specific project and version. You then identify the environment that you want to update and delete.

Note: For more information about environment concepts, prerequisites, assumptions, and considerations, see Use APIs to Design and Consume Automated Test Data Services.

This page refers to the following API Services:

- TestDataManager
- TDMProjectService
- TDMDataReservationService

Update an Environment

The process to update an environment is as follows:

1. Get the Security Token
2. Get the Project ID
3. Get the Version ID
4. Get the Environments for the Identified Project and Version
5. Get the details of the Identified Environment
   a. Get Data Sources that Include a Specific Table
6. Update the Identified Environment

You can update the following properties of an environment:

- Name of the environment
- Description of the environment
- Add Data Sources
- Change the Connection Profile Associated with Data Source

You cannot update the Data Source Name or you cannot delete the Data Source that is already added to the environment.

Get the Security Token

Use the login API to log in and generate a security token. You use your CA TDM Portal login credentials to generate the security token. You can then use the same security token to perform all other operations. The security token remains valid for 24 hours.
For more information about how to get a security token, see the "Get a Security Token" section in "Use APIs to Design and Consume Automated Test Data Services".

Get the Project ID

Get the project ID that includes the required environment. Note the project ID, because you will be using it in all the subsequent operations.

Follow these steps:

1. Access the following CA TDM Portal API:
   `GET https://<server>:<host>/TDMProjectService/api/ca/v1/projects`
2. Enter the security token in the **Authorization** field as follows:

   ```
   Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyE
   RQ5l4Ro
   ```

3. Run the API and review the response body. The following example response is generated:


```
[
{
"name": "CA_Project",
"description": "CA_Project Description",
"dateOrder": "YMD",
"id": 2716,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": 1,
"grantedFunctions": []
},
{
"name": "StoreFront - Example Project - Oracle",
"description": "StoreFront - Oracle",
"dateOrder": "YMD",
"id": 1760,
"inheritTables": true,
"timestampPrecision": 6,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "StoreFront - Example Project - SQL Server",
"description": "StoreFront - Example Project - SQL Server",
"dateOrder": "YMD",
"id": 2234,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
```

```
"name": "TDMPublish_Centrica",
"description": "TDMPublish_Centrica",
"dateOrder": "YMD",
"id": 7739,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
}
]
```


4. Identify the project and note the project ID. For this example, the project is 2716.

Get the Version ID

After you get the project ID, you must get the associated version ID. Note the version ID, because you will be using it in all the subsequent operations.

Follow these steps:

1. Access the following CA TDM Portal API:
   `GET https://<server>:<host>/TDMProjectService/api/ca/v1/projects/{projectId}/versions`
2. Enter the security token in the **Authorization** field as follows:


   ```
   Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyF
   RQ5l4Ro
   ```


3. Enter the project ID as 2716 in the **projectId** field.
4. Run the API and review the response body. The following example response is generated:


```
[
{
"id": 2717,
"name": "CA_Project Version",
"created": "2017-03-07T07:28:09+0000",
"description": "CA_Project version description",
"projectName": CA_Project,
"levelDetails": null,
"registeredObjectCount": 0,
"tablesUsed": null,
"isGeneric": false
}
]
```


5. Note the version ID, which is 2717 in this example.

Get Environments for the Identified Project and Version

Get the list of environments for your specific project and version. After you get the list, identify the environment that you want to update.

Follow these steps:

1. Access the following CA TDM Portal API:
   ```
   GET https://<server>:<host>/TDMDataReservationService/api/ca/v1/environments
   ```
2. Enter the security token in the **Authorization** field as follows:

   ```
   Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyE
   RQ5l4Ro
   ```

3. Enter information in the following fields:
   - **projectId**
     Specifies the ID of the project for which you want to retrieve the environments. For this example, the value is 2716.
   - **versionId**
     Specifies the ID of the project version for which you want to retrieve the environments. For this example, the value is 2717.
   - **page**
     Page number that you want to retrieve in a paginated result. Defaults to 1 if page size is specified. Returns all environments if page and size are empty.
   - **size**
     Page size of each page with which you want to retrieve the paginated result. Defaults to 25 if page number is specified. Returns all environments if page and size are empty.
   - **searchText**
     Search text that you want to use to perform the search on the environment name and description to get the list of environments.
   - **sortDir**
     Sorting order that you want to use to sort the paginated environments result. Valid values are ASC and DESC.
4. Run the API and review the response body. The following example response is generated:


```
[

  {

 "createdBy": "Administrator",

 "creationDate": "2017-02-09 04:09:38.081",

 "description": "CAenvironment",

 "id": 4,

 "modifiedBy": "Administrator",

 "modifiedDate": "2017-02-09 04:09:38.081",

 "name": "CAenvironment",
```

```
 "projectID": 2716,

  "versionID": 2717,

  }

{

 "createdBy": "Administrator",

 "creationDate": "2017-02-09 04:09:38.084",

 "description": "PO Environment",

 "id": 9,

 "modifiedBy": "Administrator",

 "modifiedDate": "2017-02-09 04:09:38.084",

 "name": "POenvironment",

 "projectID": 2716,

  "versionID": 2717,

  }

]
```


5. Note that the specified project and version include the environment " CAenvironment" with the ID "4".

Get Details of the Identified Environment

After you note the environment ID that you want to update, you can retrieve its details to review the information and note the properties that you want to update.

Follow these steps:

1. Access the following CA TDM Portal API:
   `GET https://<server>:<host>/TDMDataReservationService/api/ca/v1/environments/{environmentId}`
2. Enter the security token in the **Authorization** field as follows:

   ```
   Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyF
   RQ5l4Ro
   ```

3. Enter information in the following fields as follows:
   - **projectId**
     Specifies the ID of the project for which you want to retrieve the environments. For this example, the value is 2716.
   - **versionId**

Specifies the ID of the project version for which you want to retrieve the environments. For this example, the value is 2717.

- **environmentID**
  Specifies the ID of the environment for which you want to get the details. For this example, the value is 4.

4. Run the API and review the response body. The following example response is generated:


```
{
  "createdBy": "Administrator",
  "creationDate": "2017-02-09 04:09:38.081",
  "datasourcesConnectionProfiles": [
    {
      "connectionProfileName": "CAconprof",
      "connectionProfileStatus": "INVALID",
      "name": "CAdatasource"
    }
    {
      "connectionProfileName": "CAconprof1",
      "connectionProfileStatus": "INVALID",
      "name": "CAdatasource1"
    }
    {
      "connectionProfileName": "CAconprof2",
      "connectionProfileStatus": "INVALID",
      "name": "CAdatasource2"
    }
  ],
  "description": "CAenvironment",
  "id": 4,
```

```
    "modifiedBy": "Administrator",

    "modifiedDate": "2017-02-09 04:09:38.081",

    "name": "CAenvironment",

    "projectID": 2716,

    "versionID": 2717

  }
```


5. Review the properties that you want to update. For this example, the following properties are identified to change:
   - The environment description "CAenvironment" is identified to be changed to "CA Environment".
   - The connection profile of the data source which includes the table "Order" is to be changed. To get the details of the data sources (within an environment) which inlcude a specific table, see Get Data Sources that Include a Specific Table.

Get Data Sources that Include a Specific Table

When you retrieve the environment details, you get the details of all the data sources related to that environment. If you want to retrieve only those data sources which include a specific table, you must run the API: "GET https://server-po:8443/TDMDataReservationService/api/ca/v1/environments/{environmentId}".

Follow these steps:

1. Access the following CA TDM Portal API:
   `GET https://<server>:<host>/TDMDataReservationService/api/ca/v1/environments/{environmentId}`
2. Enter the security token in the **Authorization** field as follows:


```
Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyF
RQ5l4Ro
```


3. Enter information in the following fields as follows:
   - **projectId**
     Specifies the ID of the project for which you want to retrieve the environments. For this example, the value is 2716.
   - **versionId**
     Specifies the ID of the project version for which you want to retrieve the environments. For this example, the value is 2717.
   - **environmentID**
     Specifies the ID of the environment for which you want to get the details. For this example, the value is 4.
   - **tableName**
     Table name that you want to use to find the related data sources where the table exists. For this example, the value is "Orders"
4. Run the API and review the response body. The following example response is generated:


```
{

  "datasources": [
```

```
        "CAdatasource",

        "CAdatasource2"

      ]

    }
```

5. Review the properties that you want to update. For this example, the connection profile of the data source "CAdatasource" is identified to change to "POcProfile".

Update the Identified Environment

After you identify the properties that you want to update for the environment and the related data sources, you can use the update environment API to do the update.

Follow these steps:

1. Access the following CA TDM Portal API:
   `PUT https://<server>:<host>/TDMDataReservationService/api/ca/v1/environments/{environmentId}`
2. Enter the security token in the **Authorization** field as follows:

   ```
   Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyF
   RQ5l4Ro
   ```

3. Enter information in the following fields as follows:
   - **projectId**
     Specifies the ID of the project that is related to the environment for which you want to update the details. For this example, the value is 2716.
   - **versionId**
     Specifies the ID of the project version that is related to the environment for which you want to update the details. For this example, the value is 2717.
   - **environmentID**
     Specifies the ID of the environment that you want to update. For this example, the value is 4.
   - **environmentUpdate**
     Specifies the request body for updating an environment. For this example, the following properties are identified to change:
     - The environment description is to be changed to "CA Environment"
     - The connection profile of "CAdatasource" to be changed to "POcProfile"

     ```
     {

       "datasourcesConnectionProfiles": [

         {

           "connectionProfileName": "POcProfile",

           "connectionProfileStatus": "INVALID",

           "name": "CAdatasource"
     ```

```
        }
      ],
      "description": "CAenvironment",
      "name": "CA Environment"
    }
```

4. Run the API and review the response body. The following example response is generated for the environment (4):

```
{
  "createdBy": "Administrator",
  "creationDate": "2017-02-09 05:07:26.048",
  "datasourcesConnectionProfiles": [
    {
      "connectionProfileName": "POcProfile",
      "connectionProfileStatus": "INVALID",
      "name": "CAdatasource"
    }
    {
      "connectionProfileName": "CAconprof1",
      "connectionProfileStatus": "INVALID",
      "name": "CAdatasource1"
    }
    {
      "connectionProfileName": "CAconprof2",
      "connectionProfileStatus": "INVALID",
      "name": "CAdatasource2"
```

```
        }
      ],
      "description": "CA Environment",
      "id": 4,
      "modifiedBy": "Administrator",
      "modifiedDate": "2017-02-09 05:07:26.048",
      "name": "CAenvironment",
      "projectID": 2716,
      "versionID": 2717
    }
```


5. Review that the response includes the updated property. In this case, the description for the environment is changed to "CA Environment", and the connection profile for CAdatasource is changed to "POcProfile".

You have successfully updated an environment.

Delete an Environment

The process to delete an environment is as follows:

1. Get the Security Token
2. Get the Project ID
3. Get the Version ID
4. Get the Environment for the Identified Project and Version
5. Get Details of the Identified Environment
6. Delete the Identified Environment
7. Verify the Deletion

Get the Security Token

Use the login API to log in and generate a security token. You use your CA TDM Portal login credentials to generate the security token. You can then use the same security token to perform all other operations. The security token remains valid for 24 hours. To get the security token and log into the CA TDM Portal, follow the instructions in the corresponding section in Update an Environment.

Get the Project ID

Get the project ID that includes the required environment. Note the project ID, because you will be using it in all the subsequent operations. To get the project ID, follow the detailed instructions in the corresponding section in Update an Environment.

Summary of the example value used in this API is as follows:

- **Authorization:**Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro

The following response is generated, note the project ID (2716):


```
[
{
"name": "CA_Project",
"description": "CA_Project Description",
"dateOrder": "YMD",
"id": 2716,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": 1,
"grantedFunctions": []
},
{
"name": "StoreFront - Example Project - Oracle",
"description": "StoreFront - Oracle",
"dateOrder": "YMD",
"id": 1760,
"inheritTables": true,
"timestampPrecision": 6,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "StoreFront - Example Project - SQL Server",
"description": "StoreFront - Example Project - SQL Server",
"dateOrder": "YMD",
"id": 2234,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
```

```
"grantedFunctions": []
},
{
"name": "TDMPublish_Centrica",
"description": "TDMPublish_Centrica",
"dateOrder": "YMD",
"id": 7739,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
}
]
```


Get the Version ID

After you get the project ID, you must get the associated version ID. Note the version ID, because you will be using it in all the subsequent operations. To get the version ID associated with the retrieved project ID, follow the detailed instructions in the corresponding section in Update an Environment.

Summary of the example values used in this API is as follows:

- **Authorization:**Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **projectId:** 2716

The following example response is generated; note the version ID (2716):


```
[
{
"id": 2717,
"name": "CA_Project Version",
"created": "2017-03-07T07:28:09+0000",
"description": "CA_Project version description",
"projectName": CA_Project,
"levelDetails": null,
"registeredObjectCount": 0,
"tablesUsed": null,
"isGeneric": false
}
]
```

Get Environment for the Identified Project and Version

Get the list of environments for your specific project and version. After you get the list, identify the environment that you want to delete. To get all the environments for the retrieved project and version, follow the detailed instructions in the corresponding section in Update an Environment.

Summary of the example values used in this API is as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **projectId:** 2616
- **versionID:** 2617

The following response is generated:


```
[
  {
 "createdBy": "Administrator",
 "creationDate": "2017-02-09 04:09:38.081",
 "description": "CA Environment",
 "id": 4,
 "modifiedBy": "Administrator",
 "modifiedDate": "2017-02-09 04:09:38.081",
 "name": "CAenvironment",
 "projectID": 2716,
  "versionID": 2717,
  }
{
 "createdBy": "Administrator",
 "creationDate": "2017-02-09 04:09:38.084",
 "description": "PO Environment",
 "id": 9,
 "modifiedBy": "Administrator",
```

```
  "modifiedDate": "2017-02-09 04:09:38.084",

  "name": "POenvironment",

  "projectID": 2716,

   "versionID": 2717,

   }

]
```


Identify the environment that you want to delete and note its ID. For this example, the environment with the name "CAenvironment" and ID "4" is chosen for the deletion.

Get Details of the Identified Environment

After you note the environment ID that you want to delete, you can retrieve its details to review the information in more detail. To get details of a specific environment, follow the instructions in the corresponding section in Update an Environment.

Summary of the example values used in this API is as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojU RQ5l4Ro
- **projectId:** 2616
- **versionId:** 2617
- **environmentId:** 4

The following example response is generated for the environment, "CAenvironment" (4):


```
{

  "createdBy": "Administrator",

  "creationDate": "2017-02-09 05:07:26.048",

  "datasourcesConnectionProfiles": [

    {

      "connectionProfileName": "POcProfile",

      "connectionProfileStatus": "INVALID",

      "name": "CAdatasource"

    }

    {
```

```
      "connectionProfileName": "CAconprof1",
      "connectionProfileStatus": "INVALID",
      "name": "CAdatasource1"
    }
    {
      "connectionProfileName": "CAconprof2",
      "connectionProfileStatus": "INVALID",
      "name": "CAdatasource2"
    }
  ],
  "description": "CA Environment",
  "id": 4,
  "modifiedBy": "Administrator",
  "modifiedDate": "2017-02-09 05:07:26.048",
  "name": "CAenvironment",
  "projectID": 2716,
  "versionID": 2717
}
```


Review the properties to confirm that you want to delete this environment.

Delete the Identified Environment

After you identify and confirm the appropriate environment (4 in this case), you can go ahead and delete it.

Follows these steps:

1. Access the following CA TDM Portal API:
   `DELETE https://<server>:<host>/TDMDataReservationService/api/ca/v1/environments/{environmentId}`
2. Enter the security token in the **Authorization** field as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
RQ5l4Ro
```

3. Enter information in the following fields as follows:
   - **projectId**
     Specifies the ID of the project related to the environment that you want to delete. For this example, the value of the project ID is 2616.
   - **versionId**
     Specifies the ID of the project version related to the environment that you want to delete. For this example, the value of the version ID is 2617.
   - **environmentId**
     Specifies the ID of the environment that you want to delete. For this example, the value of the environment ID is 4.
4. Run the API and review the response body:

```
{
"message": "Environment is deleted successfully."
}
```

5. Review that the response includes a message that states that the Environment (4) is deleted successfully.

Verify the Deletion

After you run the delete API to delete the environment, you can verify whether the environment is appearing in the project version.

Follow these steps:

1. Access the following CA TDM Portal API:
   `GET https://<server>:<host>/TDMDataReservationService/api/ca/v1/environments`
2. Enter the security token in the **Authorization** field as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
RQ5l4Ro
```

3. Enter information in the following fields:
   - **projectId**
     Specifies the ID of the project for which you want to retrieve the environments. For this example, the value is 2616.
   - **versionId**
     Specifies the ID of the project version for which you want to retrieve the environments. For this example, the value is 2617.
4. Run the API and review the response body. The following example response is generated:

```
[

{

 "createdBy": "Administrator",
```

```
    "creationDate": "2017-02-09 04:09:38.084",

    "description": "PO Environment",

    "id": 9,

    "modifiedBy": "Administrator",

    "modifiedDate": "2017-02-09 04:09:38.084",

    "name": "POenvironment",

    "projectID": 2716,

     "versionID": 2717,

     }

  ]
```


5. Note that the specified project version now does not include the environment with the name "CAenvironment" and the ID "4", which is correct.

You have successfully deleted an environment.

Use APIs to Manage Test Data Models

This article explains with the help of an example about how test data engineers (TDEs) can use exposed CA TDM Portal APIs to manage test data models after they create them.

This article covers the following tasks. You perform all these tasks by using the APIs. You get the information about the available test data models for a specific project and version. You then identify the test data model that you want to update and delete.

Note: For more information about test data model concepts, prerequisites, assumptions, and considerations, see Use APIs to Design and Consume Automated Test Data Services.

This page refers to the following API Services:

- TestDataManager
- TDMProjectService
- TDMDataReservationService

Update a Test Data Model

The process to update a test data model is as follows:

1. Get the Security Token.
2. Get the Project ID.
3. Get the Version ID.
4. Get Test Data Models for the Identified Project and Version.
5. Get Details of the Identified Test Data Model.
6. Update the Identified Test Data Model.

Note: These examples use the sample Northwind database that is available for Microsoft SQL Server. Refer the Microsoft website to download the Northwind database.

You can update the following properties of a test data model irrespective of whether the visible parameter is set to true or false. The visible parameter is used only to decide whether you want to display the test data model to testers.

- Name of the test data model.
- Description of the test data model.
- Visibility of the test data model.
- Display name of the root entity.

You cannot update the following properties:

- Name of the root entity.
- Model keys and root entity.
- Data source of the root entity.

Get the Security Token

Use the login API to log in and generate a security token. You use your CA TDM Portal login credentials to generate the security token. You can then use the same security token to perform all other operations. The security token remains valid for 24 hours.

Follow these steps:

1. Access an application that allows you to encode your credentials to the Base64 format.
2. Enter your CA TDM Portal login credentials (in the format `<user name>:<password>`) in the source field.
   Note: Ensure that the credentials have appropriate permissions to perform all the required operations.
3. Click the option to encode the credentials. The encoded Base64 format for the example is displayed as follows:
   ```
   ZwRTaX5pc4SxYXSvcjptYXJtaXRl
   ```
4. Copy the encoded value.
5. Access the following CA TDM Portal API:
   ```
   POST https://<server>:<host>/TestDataManager/user/login
   ```
6. Enter the encoded value in the **Authorization** field, which is as follows for the example:
   ```
   Basic YWRtaW5pc3RyYXRvcjptYXJtaXRl
   ```
7. Run the API to get a security token for your credentials.
8. Note the value of the **token** parameter in the response body, which is as follows for the example:
   ```
   eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojUxm8ENTeRRrdOa-
   RQ5l4Ro
   ```
   You have successfully generated a security token that you can use in all the subsequent operations explained in this article.

Get the Project ID

Get the project ID that includes the required test data model. Note the project ID, because you will be using it in all the subsequent operations.

Follow these steps:

1. Access the following CA TDM Portal API:
   ```
   GET https://<server>:<host>/TDMProjectService/api/ca/v1/projects
   ```
2. Enter the security token in the **Authorization** field as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
RQ5l4Ro
```

3. Run the API and review the response body. The following example response is generated:


```
[
 {
 "name": "Order",
 "description": "This is Order Management project.",
 "dateOrder": "YMD",
 "id": 141357,
 "inheritTables": true,
 "timestampPrecision": 3,
 "type": "DB",
 "levels": [],
 "created": null,
 "updated": null,
 "versions": [],
 "grantedFunctions": []
 },
 {
 "name": "StoreFront - Example Project - Oracle",
 "description": "StoreFront - Oracle",
 "dateOrder": "YMD",
 "id": 1760,
 "inheritTables": true,
 "timestampPrecision": 6,
 "type": "DB",
 "levels": [],
 "created": null,
 "updated": null,
 "versions": [],
 "grantedFunctions": []
 },
 {
 "name": "StoreFront - Example Project - SQL Server",
 "description": "StoreFront - Example Project - SQL Server",
 "dateOrder": "YMD",
 "id": 2234,
 "inheritTables": true,
 "timestampPrecision": 3,
 "type": "DB",
 "levels": [],
 "created": null,
 "updated": null,
 "versions": [],
```

```
 "grantedFunctions": []
 },
 {
 "name": "TDMPublish_Centrica",
 "description": "TDMPublish_Centrica",
 "dateOrder": "YMD",
 "id": 7739,
 "inheritTables": true,
 "timestampPrecision": 3,
 "type": "DB",
 "levels": [],
 "created": null,
 "updated": null,
 "versions": [],
 "grantedFunctions": []
 }
]
```


4. Identify the project and note the project ID. For this example, the project is Order with the ID 141357.

Get the Version ID

After you get the project ID, you must get the associated version ID. Note the version ID, because you will be using it in all the subsequent operations.

Follow these steps:

1. Access the following CA TDM Portal API:
   `GET https://<server>:<host>/TDMProjectService/api/ca/v1/projects/{projectId}/versions`
2. Enter the security token in the **Authorization** field as follows:


```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
RQ5l4Ro
```


3. Enter the project ID as 141357 in the **projectId** field.
4. Run the API and review the response body. The following example response is generated:


```
[
 {
 "id": 141358,
 "name": "1.0",
 "created": "2017-03-07T07:28:09+0000",
 "description": "This is Order Management version 1.0.",
 "projectName": null,
 "levelDetails": null,
 "registeredObjectCount": 0,
 "tablesUsed": null,
 "isGeneric": false
```

```
  }
 ]
```

5. Note the version ID, which is 141358 in this example.

Get Test Data Models for the Identified Project and Version

Get the list of test data models for your specific project and version. After you get the list, identify the test data model that you want to update.

Follow these steps:

1. Access the following CA TDM Portal API:
   `GET https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels`
2. Enter the security token in the **Authorization** field as follows:

   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
   RQ5l4Ro
   ```

3. Enter information in the following fields:
   - **projectId**
     Specifies the ID of the project for which you want to retrieve test data models. For this example, the value is 141357.
   - **versionId**
     Specifies the ID of the project version for which you want to retrieve test data models. For this example, the value is 141358.
4. Run the API and review the response body. The following example response is generated:


```
{
 "numberOfTestDataModels": 2,
 "totalNumberOfTestDataModels": 2,
 "testDataModelsList": [
 {
 "id": 386,
 "name": "Orders",
 "description": "This test data model is for Orders Management application.",
 "visible": true,
 "projectId": 141357,
 "versionId": 141358
 },
 {
 "id": 410,
 "name": "Product_Purchase",
 "description": "This test data model is for the Purchase application.",
 "visible": true,
 "projectId": 141357,
 "versionId": 141358
 }
```

```
 ]
}
```

Note that the specified project and version include two test data models: Orders and Product_Purchase.

5. Identify the test data model that you want to update and note its ID. For this example, Product_Purchase with the ID 410 is chosen for the update.

Get Details of the Identified Test Data Model

After you note the test data model ID that you want to update, you can retrieve its details to review the information and note the properties that you want to update.

Follow these steps:

1. Access the following CA TDM Portal API:
   ```
   GET https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels/{testDataModelId}
   ```
2. Enter the security token in the **Authorization** field as follows:

   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
   RQ5l4Ro
   ```

3. Enter information in the following fields as follows:
   - **testDataModelId**
     Specifies the ID of the test data model for which you want to get the details. For this example, the value of the test data model ID is 410.
   - **projectId**
     Specifies the ID of the project for which you want to retrieve test data models. For this example, the value is 141357.
   - **versionId**
     Specifies the ID of the project version for which you want to retrieve test data models. For this example, the value is 141358.
4. Run the API and review the response body. The following example response is generated for the Product_Purchase (410) test data model:

   ```
   {
    "id": 410,
    "name": "Product_Purchase",
    "description": "This test data model is for the Purchase application.",
    "visible": true,
    "modelKeys": [
    "ProductID"
    ],
    "root": {
    "displayName": "Products",
    "rootEntity": {
    "id": 393,
    "name": "Products",
    "primaryKeys": [
    "ProductID"
   ```

```
],
"dataSource": "Orders_DS"
}
},
"projectId": 141357,
"versionId": 141358,
"creationDate": "2017-03-10 11:10:01.159",
"modifiedDate": "2017-03-10 11:10:01.159",
"createdBy": "John",
"modifiedBy": "John"
}
```

5. Review the properties that you want to update. For this example, the display name Products is identified to be changed to Supplier_Product.

Update the Identified Test Data Model

After you identify the test data model and the properties that you want to update, you can use the update test data model API to do the update.

Follow these steps:

1. Access the following CA TDM Portal API:
   `PUT https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels/{testDataModelId}`
2. Enter the security token in the **Authorization** field as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
RQ5l4Ro
```

3. Enter information in the following fields as follows:
   - **testDataModelId**
     Specifies the ID of the test data model that you want to update. For this example, the value is 410.
   - **projectId**
     Specifies the ID of the project that is related to the test data model for which you want to update the details. For this example, the value is 141357.
   - **versionId**
     Specifies the ID of the project version that is related to the test data model for which you want to update the details. For this example, the value is 141358.
   - **testDataModel**
     Specifies the payload that includes the test data model parameters. Specify the parameter values that you want to update. This payload includes the following parameters:
     - **description**
       Specifies the description of the test data model that you are creating. For this example, the value of the description is "This test data model is for the Purchase application."
     - **modelKeys**
       Specifies the list of model keys for the root. For this example, the value of the model key is ProductID, which is the primary key in the Products entity.
     - **name**

Specifies the name of the test data model that you are updating. For this example, the value of the test data model name is Product_Purchase.

- **root**
  Specifies the test data model root details.
- **visible**
  Specifies whether you want this test data model to be visible to testers. For this example, the value is set to true.
- **displayName**
  Specifies the display name of the model key. For this example, the value of the display name of the model key is Supplier_Product.
- **rootEntity**
  Specifies the root entity details (entity data source and entity name).
- **dataSource**
  Specifies the data source of the entity. For this example, the value is Orders_DS.
- **name**
  Specifies the name of the entity. For this example, the value is Products.

For this example, the display name (Products) for the model key is chosen for the update (Supplier_Product).

```
{
"description": "This test data model is for the Purchase application.",
"modelKeys": [
"ProductID"
],
"name": "Product_Purchase",
"root": {
"displayName": "Supplier_Product",
"rootEntity": {
"dataSource": "Orders_DS",
"name": "Products"
}
},
"visible": true
}
```

4. Run the API and review the response body. The following example response is generated for the test data model ID (410):

```
{
 "id": 410,
 "name": "Product_Purchase",
 "description": "This test data model is for the Purchase application.",
 "visible": true,
 "modelKeys": [
 "ProductID"
 ],
 "root": {
 "displayName": "Supplier_Product",
 "rootEntity": {
 "id": 393,
```

```
"name": "Products",
"primaryKeys": [
"ProductID"
],
"dataSource": "Orders_DS"
}
},
"projectId": 141357,
"versionId": 141358,
"creationDate": "2017-03-10 11:10:01.159",
"modifiedDate": "2017-03-10 11:43:12.703",
"createdBy": "John",
"modifiedBy": "John"
}
```


5. Review that the response includes the updated property. In this case, the display name for the model key is changed to Supplier_Product.

You have successfully updated a test data model.

Delete a Test Data Model

The process to delete a test data model is as follows:

1. Get the Security Token.
2. Get the Project ID.
3. Get the Version ID.
4. Get Test Data Models for the Identified Project and Version.
5. Get Details of the Identified Test Data Model.
6. Delete the Identified Test Data Model.
7. Verify the Deletion.

Note: These examples use the sample Northwind database that is available for Microsoft SQL Server. Refer the Microsoft website to download the Northwind database.

Get the Security Token

Use the login API to log in and generate a security token. You use your CA TDM Portal login credentials to generate the security token. You can then use the same security token to perform all other operations. The security token remains valid for 24 hours. To get the security token and log into the CA TDM Portal, follow the instructions in the corresponding section in Update a Test Data Model.

Get the Project ID

Get the project ID that includes the required test data model. Note the project ID, because you will be using it in all the subsequent operations. To get the project ID, follow the detailed instructions in the corresponding section in Update a Test Data Model.

Summary of the example value used in this API is as follows:

- **Authorization:**Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro

The following response is generated, note the project ID (141357):


```
[
{
"name": "Order",
"description": "This is Order Management project.",
"dateOrder": "YMD",
"id": 141357,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "StoreFront - Example Project - Oracle",
"description": "StoreFront - Oracle",
"dateOrder": "YMD",
"id": 1760,
"inheritTables": true,
"timestampPrecision": 6,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "StoreFront - Example Project - SQL Server",
"description": "StoreFront - Example Project - SQL Server",
"dateOrder": "YMD",
"id": 2234,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
```

```
"name": "TDMPublish_Centrica",
"description": "TDMPublish_Centrica",
"dateOrder": "YMD",
"id": 7739,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
}
]
```


Get the Version ID

After you get the project ID, you must get the associated version ID. Note the version ID, because you will be using it in all the subsequent operations. To get the version ID associated with the retrieved project ID, follow the detailed instructions in the corresponding section in Update a Test Data Model.

Summary of the example values used in this API is as follows:

- **Authorization:**Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **projectId:** 141357

The following example response is generated; note the version ID (141358):


```
[
{
"id": 141358,
"name": "1.0",
"created": "2017-03-07T07:28:09+0000",
"description": "This is Order Management version 1.0.",
"projectName": null,
"levelDetails": null,
"registeredObjectCount": 0,
"tablesUsed": null,
"isGeneric": false
}
]
```

Get Test Data Models for the Identified Project and Version

Get the list of test data models for your specific project and version. After you get the list, identify the test data model that you want to update. To get all the test data models for the retrieved project and version, follow the detailed instructions in the corresponding section in Update a Test Data Model.

Summary of the example values used in this API is as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojU RQ5l4Ro
- **projectId:** 141357
- **versionID:** 141358

The following response is generated:


```
{
"numberOfTestDataModels": 2,
"totalNumberOfTestDataModels": 2,
"testDataModelsList": [
{
"id": 386,
"name": "Orders",
"description": "This test data model is for Orders Management application.",
"visible": true,
"projectId": 141357,
"versionId": 141358
},
{
"id": 410,
"name": "Product_Purchase",
"description": "This test data model is for the Purchase application.",
"visible": true,
"projectId": 141357,
"versionId": 141358
}
]
}
```


Identify the test data model that you want to delete and note its ID. For this example, Product_Purchase with the ID 410 is chosen for the deletion.

Get Details of the Identified Test Data Model

After you note the test data model ID that you want to delete, you can retrieve its details to review the information in more detail. To get details of a specific test data model, follow the instructions in the corresponding section in Update a Test Data Model.

Summary of the example values used in this API is as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **testDataModelId:** 410
- **projectId:** 141357
- **versionId:** 141358

The following example response is generated for the Product_Purchase (410) test data model:

```
{
"id": 410,
"name": "Product_Purchase",
"description": "This test data model is for the Purchase application.",
"visible": true,
"modelKeys": [
"ProductID"
],
"root": {
"displayName": "Products",
"rootEntity": {
"id": 393,
"name": "Products",
"primaryKeys": [
"ProductID"
],
"dataSource": "Orders_DS"
}
},
"projectId": 141357,
"versionId": 141358,
"creationDate": "2017-03-10 11:10:01.159",
"modifiedDate": "2017-03-10 11:10:01.159",
"createdBy": "John",
"modifiedBy": "John"
}
```

Review the properties to confirm that you want to delete this test data model.

Delete the Identified Test Data Model

After you identify and confirm the appropriate test data model (410 in this case), you can go ahead and delete it.

Follows these steps:

1. Access the following CA TDM Portal API:
   `DELETE https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels/{testDataModelId}`
2. Enter the security token in the **Authorization** field as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
RQ5l4Ro
```

3. Enter information in the following fields as follows:
   - **testDataModelId**
     Specifies the ID of the test data model that you want to delete. For this example, the value of the test data model ID is 410.
   - **projectId**
     Specifies the ID of the project related to the test data model that you want to delete. For this example, the value of the project ID is 141357.
   - **versionId**
     Specifies the ID of the project version related to the test data model that you want to delete. For this example, the value of the version ID is 141358.
4. Run the API and review the response body:

```
{
 "message": "Test Data Model is deleted successfully."
}
```

5. Review that the response includes a message that states that the test data model (410) is deleted successfully.

Verify the Deletion

After you run the delete API to delete the test data model, you can verify whether the test data model is appearing in the project version.

Follow these steps:

1. Access the following CA TDM Portal API:
   `GET https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels`
2. Enter the security token in the **Authorization** field as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
RQ5l4Ro
```

3. Enter information in the following fields:
   - **projectId**
     Specifies the ID of the project for which you want to retrieve test data models. For this example, the value is 141357.
   - **versionId**
     Specifies the ID of the project version for which you want to retrieve test data models. For this example, the value is 141358.
4. Run the API and review the response body. The following example response is generated:

```
{
 "numberOfTestDataModels": 1,
 "totalNumberOfTestDataModels": 1,
 "testDataModelsList": [
```

```
{
"id": 386,
"name": "Orders",
"description": "This test data model is for Orders Management application.",
"visible": false,
"projectId": 141357,
"versionId": 141358
}
]
}
```


5. Note that the specified project version now does not include the two test data models. The Product_Purchase test data model is no longer available, which is correct.

You have successfully deleted a test data model.

Use APIs to Manage Associations in a Test Data Model

This article explains with the help of an example about how test data engineers (TDEs) can use exposed CA TDM Portal APIs to manage associations in test data models after they create them.

This article covers the following tasks. You perform all these tasks by using the APIs. You get the information about the available associations in a test data model. You then identify the association that you want to update and delete.

Note: For more information about test data model concepts, persona-based tasks, prerequisites, assumptions, and considerations, see Use APIs to Design and Consume Automated Test Data Services.

This page refers to the following API Services:

- TestDataManager
- TDMProjectService
- TDMDataReservationService

Update an Association in a Test Data Model

The process to update an association in a test data model is as follows:

1. Get the Security Token.
2. Get the Project ID.
3. Get the Version ID.
4. Get the Test Data Model ID.
5. Get All Associations in the Identified Test Data Model.
6. Get Details of the Identified Association.
7. Update the Identified Association.

You can update the following association-related properties:

- Name of the association.
- Source entity of the association.
- Join fields (if forceUpdate is true).
- Association type (if forceUpdate is true).

Get the Security Token

Use the login API to log in and generate a security token. You use your CA TDM Portal login credentials to generate the security token. You can then use the same security token to perform all other operations. The security token remains valid for 24 hours.

Follow these steps:

1. Access an application that allows you to encode your credentials to the Base64 format.
2. Enter your CA TDM Portal login credentials (in the format `<user name>:<password>`) in the source field.
   Note: Ensure that the credentials have appropriate permissions to perform all the required operations.
3. Click the option to encode the credentials. The encoded Base64 format for the example is displayed as follows:
   ```
   ZwRTaX5pc4SxYXSvcjptYXJtaXRl
   ```
4. Copy the encoded value.
5. Access the following CA TDM Portal API:
   ```
   POST https://<server>:<host>/TestDataManager/user/login
   ```
   Note: For more information about this API, see the "auth-controller: Auth Controller" section at https://<server>:<port>/TestDataManager/swagger-ui.html. For the example in this article, the URL is https://server-po:8443/TestDataManager/swagger-ui.html.
6. Enter the encoded value in the **Authorization** field, which is as follows for the example:
   ```
   Basic YWRtaW5pc3RyYXRvcjptYXJtaXRl
   ```
7. Run the API to get a security token for your credentials.
8. Note the value of the **token** parameter in the response body, which is as follows for the example:
   ```
   eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojUxm8ENTeRRrdOa-
   RQ5l4Ro
   ```
   You have successfully generated a security token that you can use in all the subsequent operations explained in this article.

Get the Project ID

Get the project ID that includes the required test data model. Note the project ID, because you will be using it in all the subsequent operations.

Follow these steps:

1. Access the following CA TDM Portal API:
   ```
   GET https://<server>:<host>/TDMProjectService/api/ca/v1/projects
   ```
2. Enter the security token in the **Authorization** field as follows:

   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
   RQ5l4Ro
   ```

3. Run the API and review the response body. The following example response is generated:

   ```
   [
   {
   "name": "Order",
   "description": "This is Order Management project.",
   "dateOrder": "YMD",
   "id": 141357,
   "inheritTables": true,
   ```

```
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "StoreFront - Example Project - Oracle",
"description": "StoreFront - Oracle",
"dateOrder": "YMD",
"id": 1760,
"inheritTables": true,
"timestampPrecision": 6,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "StoreFront - Example Project - SQL Server",
"description": "StoreFront - Example Project - SQL Server",
"dateOrder": "YMD",
"id": 2234,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "TDMPublish_Centrica",
"description": "TDMPublish_Centrica",
"dateOrder": "YMD",
"id": 7739,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
```

```
"updated": null,
"versions": [],
"grantedFunctions": []
}
]
```


4. Identify the project and note the project ID. For this example, the project is Order with the ID 141357.

Get the Version ID

After you get the project ID, you must get the associated version ID. Note the version ID, because you will be using it in all the subsequent operations.

Follow these steps:

1. Access the following CA TDM Portal API:
   `GET https://<server>:<host>/TDMProjectService/api/ca/v1/projects/{projectId}/versions`
2. Enter the security token in the **Authorization** field as follows:


```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
RQ5l4Ro
```


3. Enter the project ID as 141357 in the **projectId** field.
4. Run the API and review the response body. The following example response is generated:


```
[
{
"id": 141358,
"name": "1.0",
"created": "2017-03-07T07:28:09+0000",
"description": "This is Order Management version 1.0.",
"projectName": null,
"levelDetails": null,
"registeredObjectCount": 0,
"tablesUsed": null,
"isGeneric": false
}
]
```


5. Note the version ID, which is 141358 in this example.

Get the Test Data Model ID

Get the list of test data models for your specific project and version. After you get the list, identify the test data model that includes the association that you want to update. Note the ID of the test data model.

Follow these steps:

1. Access the following CA TDM Portal API:
   `GET https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels`

2. Enter the security token in the **Authorization** field as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
RQ5l4Ro
```

3. Enter information in the following fields:
   - **projectId**
     Specifies the ID of the project for which you want to retrieve test data models. For this example, the value is 141357.
   - **versionId**
     Specifies the ID of the project version for which you want to retrieve test data models. For this example, the value is 141358.
4. Run the API and review the response body. The following example response is generated:

```
{
"numberOfTestDataModels": 2,
"totalNumberOfTestDataModels": 2,
"testDataModelsList": [
{
"id": 386,
"name": "Orders",
"description": "This test data model is for Orders Management application.",
"visible": true,
"projectId": 141357,
"versionId": 141358
},
{
"id": 410,
"name": "Product_Purchase",
"description": "This test data model is for the Purchase application.",
"visible": true,
"projectId": 141357,
"versionId": 141358
}
]
}
```

   Note that the specified project and version include two test data models: Orders and Product_Purchase.
5. Identify the test data model that includes the association; note the test data model ID. For this example, the test data model ID 386 includes the association.

Get All Associations in the Identified Test Data Model

Get all the associations that are related to a specific test data model. After you get the list, identify the association that you want to update.

Follow these steps:

1. Access the following CA TDM Portal API:

```
GET https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels/{testDataModelId}/
associations
```

2. Enter the security token in the **Authorization** field as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
RQ5l4Ro
```

3. Enter information in the following fields:
   - **projectId**
     Specifies the ID of the project that includes the test data model for which you want to get the associations. For this example, the value is 141357.
   - **versionId**
     Specifies the ID of the project version that includes the test data model for which you want to get the associations. For this example, the value is 141358.
   - **testDataModelId**
     Specifies the ID of the test data model for which you want to get the associations. For this example, the value is 386.
4. Run the API and review the response body. The following example response that includes all the associations for the selected test data model is generated:


```
[
 {
 "id": 391,
 "name": "Order Details",
 "associationType": "ONE_MANY",
 "joinFields": [
 {
 "fieldName": "OrderID",
 "referenceFieldName": "OrderID"
 }
 ],
 "sourceEntity": {
 "id": 387,
 "name": "Orders",
 "primaryKeys": [
 "OrderID"
 ],
 "dataSource": "Orders_DS"
 },
 "targetEntity": {
 "id": 390,
 "name": "Order Details",
 "primaryKeys": [
 "ProductID",
 "OrderID"
 ],
 "dataSource": "Orders_DS"
```

```
 }
 },
 {
 "id": 394,
 "name": "Products",
 "associationType": "MANY_ONE",
 "joinFields": [
 {
 "fieldName": "ProductID",
 "referenceFieldName": "ProductID"
 }
 ],
 "sourceEntity": {
 "id": 390,
 "name": "Order Details",
 "primaryKeys": [
 "ProductID",
 "OrderID"
 ],
 "dataSource": "Orders_DS"
 },
 "targetEntity": {
 "id": 393,
 "name": "Products",
 "primaryKeys": [
 "ProductID"
 ],
 "dataSource": "Orders_DS"
 }
 }
]
```


Note that the response includes two associations with IDs 391 and 394 for the Orders test data model (386).

5. Identify the association that you want to update and note its ID. For this example, association with the ID 394 is chosen for the update.

Get Details of the Identified Association

After you note the association ID that you want to update, you can retrieve its details to review the information in more detail.

Follow these steps:

1. Access the following CA TDM Portal API:

   ```
   GET https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels/{testDataModelId}/
   associations/{associationId}
   ```

2. Enter the security token in the **Authorization** field as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
RQ5l4Ro
```

3. Enter information in the following fields as follows:
   - **projectId**
     Specifies the ID of the project related to the test data model that includes the association for which you want to get the details. For this example, the project ID value is 141357.
   - **versionId**
     Specifies the ID of the project version related to the test data model that includes the association for which you want to get the details. For this example, the value of the version ID is 141358.
   - **testDataModelId**
     Specifies the ID of the test data model that includes the association for which you want to get the details. For this example, the value of the test data model ID is 386 (Orders).
   - **associationId**
     Specifies the ID of the association for which you want to get the details. For this example, the value of the association ID is 394.
4. Run the API and review the response body. The following example response is generated for the association ID 394:


```
{
 "id": 394,
 "name": "Products",
 "associationType": "MANY_ONE",
 "joinFields": [
 {
 "fieldName": "ProductID",
 "referenceFieldName": "ProductID"
 }
 ],
 "sourceEntity": {
 "id": 390,
 "name": "Order Details",
 "primaryKeys": [
 "ProductID",
 "OrderID"
 ],
 "dataSource": "Orders_DS"
 },
 "targetEntity": {
 "id": 393,
 "name": "Products",
 "primaryKeys": [
 "ProductID"
 ],
 "dataSource": "Orders_DS"
 }
}
```

5. Review the properties that you want to update. For this example, the association type is identified to be changed to ONE_ONE.

Update the Identified Association

After you review and identify the association that you want to update, you can use the update association API to do the update.

Follow these steps:

1. Access the following CA TDM Portal API:
   ```
   PUT https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels/{testDataModelId}/
   associations/{associationId}
   ```
2. Enter the security token in the **Authorization** field as follows:

   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
   RQ5l4Ro
   ```

3. Enter information in the following fields as follows:
   - **projectId**
   - Specifies the ID of the project related to the test data model that includes the association you want to update. For this example, the value is 141357.
   - **versionId**
     Specifies the ID of the project version related to the test data model that includes the association you want to update. For this example, the value is 141358.
   - **testDataModelId**
     Specifies the ID of the test data model that includes the association you want to update. For this example, the value is 386.
   - **associationId**
     Specifies the ID of the association that you want to update. For this example, the value of the association ID is 394.
   - **forceUpdate**
     Specifies whether you want to forcefully save the association in case of a conflict. To do so, set the value to true; otherwise, select false. For this example, the value is true.
   - **association**
     Specifies the payload that includes the association parameters. Specify the parameter values that you want to update. This payload includes the following parameters:
     - **associationType**
       Specifies the type of the association. Applicable values are: 'ONE_ONE', 'ONE_MANY', 'MANY_ONE'. For this example, the value is set it ONE_ONE.
     - **joinFields**
       Specifies the field details that are used for establishing the join (association) between the source and the target entities.
     - **name**
       Specifies the name of the association that you are updating. For this example, the value of the association name is Products.
     - **sourceEntity**
       Specifies the parent data entity details.
     - **targetEntity**
       Specifies the child data entity details.
     - **fieldName**

Specifies the name of the field used in the association. For this example, the value of the field name is ProductID, which comes from the Order Details entity (source).

- **referenceFieldName**
  Specifies the name of the reference field used in the association. For this example, the value of the reference field name is ProductID, which comes from the Products entity (target).
- **dataSource**
  Specifies the data source of the entities. For this example, the value of the data source is Orders_DS.
- **name**
  Specifies the name of the source and target entities. For this example, the value of the source entity name is Order Details, and the value of the target entity name is Products.

For this example, the association update payload is as follows. Note that the association type is chosen for the update:

```
{
 "associationType": "ONE_ONE",
 "joinFields": [
 {
 "fieldName": "ProductID",
 "referenceFieldName": "ProductID"
 }
 ],
 "name": "Products",
 "sourceEntity": {
 "dataSource": "Orders_DS",
 "name": "Order Details"
 },
 "targetEntity": {
 "dataSource": "Orders_DS",
 "name": "Products"
 }
}
```

4. Run the API and review the response body. The following example response is generated:

```
{
 "id": 394,
 "name": "Products",
 "associationType": "ONE_ONE",
 "joinFields": [
 {
 "fieldName": "ProductID",
 "referenceFieldName": "ProductID"
 }
 ],
 "sourceEntity": {
 "id": 390,
 "name": "Order Details",
 "primaryKeys": [
```

```
"ProductID",
"OrderID"
],
"dataSource": "Orders_DS"
},
"targetEntity": {
"id": 393,
"name": "Products",
"primaryKeys": [
"ProductID"
],
"dataSource": "Orders_DS"
}
}
```

5. Review that the response includes the updated property. In this case, the association type is changed to ONE_ONE.

You have successfully updated an association in a test data model.

Delete an Association in a Test Data Model

The process to delete an association in a test data model is as follows:

1. Get the Security Token.
2. Get the Project ID.
3. Get the Version ID.
4. Get the Test Data Model ID.
5. Get All Associations in the Identified Test Data Model.
6. Get Details of the Identified Association.
7. Delete the Identified Association.
8. Verify the Deletion.

Get the Security Token

Use the login API to log in and generate a security token. You use your CA TDM Portal login credentials to generate the security token. You can then use the same security token to perform all other operations. The security token remains valid for 24 hours. To get the security token and log into the CA TDM Portal, follow the instructions in the corresponding section in Update a Test Data Model.

Get the Project ID

Get the project ID that includes the required test data model. Note the project ID, because you will be using it in all the subsequent operations. To get the project ID, follow the detailed instructions in the corresponding section in Update an Association in a Test Data Model.

Summary of the example value used in this API is as follows:

- **Authorization:**Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro

The following response is generated, note the project ID (141357):

```
[
{
"name": "Order",
"description": "This is Order Management project.",
"dateOrder": "YMD",
"id": 141357,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "StoreFront - Example Project - Oracle",
"description": "StoreFront - Oracle",
"dateOrder": "YMD",
"id": 1760,
"inheritTables": true,
"timestampPrecision": 6,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "StoreFront - Example Project - SQL Server",
"description": "StoreFront - Example Project - SQL Server",
"dateOrder": "YMD",
"id": 2234,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "TDMPublish_Centrica",
"description": "TDMPublish_Centrica",
```

```
"dateOrder": "YMD",
"id": 7739,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
}
]
```


Get the Version ID

After you get the project ID, you must get the associated version ID. Note the version ID, because you will be using it in all the subsequent operations. To get the version ID associated with the retrieved project ID, follow the detailed instructions in the corresponding section in Update an Association in a Test Data Model.

Summary of the example values used in this API is as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojU
RQ5l4Ro
- **projectId:** 141357

The following example response is generated; note the version ID (141358):


```
[
{
"id": 141358,
"name": "1.0",
"created": "2017-03-07T07:28:09+0000",
"description": "This is Order Management version 1.0.",
"projectName": null,
"levelDetails": null,
"registeredObjectCount": 0,
"tablesUsed": null,
"isGeneric": false
}
]
```


Get the Test Data Model ID

Get the list of test data models for your specific project and version. After you get the list, identify the test data model that includes the association that you want to delete. Note the ID of the test data model. To get the ID, follow the detailed instructions in the corresponding section in Update an Association in a Test Data Model.

Summary of the example values used in this API is as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojU RQ5l4Ro
- **projectId:** 141357
- **versionId:** 141358

The following example response is generated. Note that the specified project and version include two test data models: Orders and Product_Purchase. Identify the test data model that includes the association and note the test data model ID. For this example, the test data model ID 386 includes the association.


```
{
"numberOfTestDataModels": 2,
"totalNumberOfTestDataModels": 2,
"testDataModelsList": [
{
"id": 386,
"name": "Orders",
"description": "This test data model is for Orders Management application.",
"visible": true,
"projectId": 141357,
"versionId": 141358
},
{
"id": 410,
"name": "Product_Purchase",
"description": "This test data model is for the Purchase application.",
"visible": true,
"projectId": 141357,
"versionId": 141358
}
]
}
```


Get All Associations in the Identified Test Data Model

Get all the associations that are related to a specific test data model. After you get the list, identify the association that you want to delete. To get the list of associations in a test data model, follow the detailed instructions in the corresponding section in Update an Association in a Test Data Model.

Summary of the example values used in this API are as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojU RQ5l4Ro
- **projectId:** 141357
- **versionId:** 141358
- **testDataModelId:** 386

The following example response that includes all the associations for the selected test data model is generated. Note that the response includes two associations with IDs 391 and 394 for the Orders test data model (386). Identify the association that you want to delete and note its ID. For this example, association ID 394 is chosen for the delete:


```
[
{
"id": 391,
"name": "Order Details",
"associationType": "ONE_MANY",
"joinFields": [
{
"fieldName": "OrderID",
"referenceFieldName": "OrderID"
}
],
"sourceEntity": {
"id": 387,
"name": "Orders",
"primaryKeys": [
"OrderID"
],
"dataSource": "Orders_DS"
},
"targetEntity": {
"id": 390,
"name": "Order Details",
"primaryKeys": [
"ProductID",
"OrderID"
],
"dataSource": "Orders_DS"
}
},
{
"id": 394,
"name": "Products",
"associationType": "MANY_ONE",
"joinFields": [
{
"fieldName": "ProductID",
"referenceFieldName": "ProductID"
}
],
"sourceEntity": {
"id": 390,
"name": "Order Details",
```

```
"primaryKeys": [
"ProductID",
"OrderID"
],
"dataSource": "Orders_DS"
},
"targetEntity": {
"id": 393,
"name": "Products",
"primaryKeys": [
"ProductID"
],
"dataSource": "Orders_DS"
}
}
]
```

Get Details of the Identified Association

After you note the association ID that you want to delete, you can retrieve its details to review the information in more detail. To get details of a specific association, follow the detailed instructions in the corresponding section in Update an Association in a Test Data Model.

Summary of the example values used in this API are as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **projectId:** 141357
- **versionId:** 141358
- **testDataModelId:** 386
- **associationId:** 394

The following example response is generated for the association ID 394. Review the properties to confirm that you want to delete this association:

```
{
"id": 394,
"name": "Products",
"associationType": "MANY_ONE",
"joinFields": [
{
"fieldName": "ProductID",
"referenceFieldName": "ProductID"
}
],
"sourceEntity": {
"id": 390,
```

```
"name": "Order Details",
"primaryKeys": [
"ProductID",
"OrderID"
],
"dataSource": "Orders_DS"
},
"targetEntity": {
"id": 393,
"name": "Products",
"primaryKeys": [
"ProductID"
],
"dataSource": "Orders_DS"
}
}
```


Delete the Identified Association

After you identify and confirm the appropriate association in a test data model, you can go ahead and delete it.

Follows these steps:

1. Access the following CA TDM Portal API:
   ```
   DELETE https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels/{testDataModelId}/
   associations/{associationId}
   ```
2. Enter the security token in the **Authorization** field as follows:

   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
   RQ5l4Ro
   ```

3. Enter information in the following fields as follows:
   - **projectId**
     Specifies the ID of the project that includes the test data model from which you want to delete the association. For this example, the value of the project ID is 141357.
   - **versionId**
     Specifies the ID of the project version that includes the test data model from which you want to delete the association. For this example, the value of the version ID is 141358.
   - **testDataModelId**
     Specifies the ID of the test data model from which you want to delete the association. For this example, the value of the test data model ID is 386.
   - **associationId**
     Specifies the ID of the association you want to delete. For this example, the value of the association ID is 394.
4. Run the API and review the response body:

   ```
   {
    "message": "Association is deleted successfully."
   }
   ```

5. Review that the response includes a message that states that the association has been deleted successfully.

Verify the Deletion

After you run the delete association API to delete the association, you can verify whether the association is appearing in the test data model.

Follow these steps:

1. Access the following CA TDM Portal API:
   ```
   GET https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels/{testDataModelId}/associations
   ```
2. Enter the security token in the **Authorization** field as follows:

   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
   RQ5l4Ro
   ```

3. Enter information in the following fields:
   - **projectId**
     Specifies the ID of the project that includes the test data model for which you want to get the associations. For this example, the value is 141357.
   - **versionId**
     Specifies the ID of the project version that includes the test data model for which you want to get the associations. For this example, the value is 141358.
   - **testDataModelId**
     Specifies the ID of the test data model for which you want to get the associations. For this example, the value is 386.
4. Run the API and review the response body. The following example response that includes all the associations for the selected test data model is generated:

   ```
   [
   {
   "id": 391,
   "name": "Order Details",
   "associationType": "ONE_MANY",
   "joinFields": [
   {
   "fieldName": "OrderID",
   "referenceFieldName": "OrderID"
   }
   ],
   "sourceEntity": {
   "id": 387,
   "name": "Orders",
   "primaryKeys": [
   "OrderID"
   ],
   "dataSource": "Orders_DS"
   ```

```
},
"targetEntity": {
"id": 390,
"name": "Order Details",
"primaryKeys": [
"ProductID",
"OrderID"
],
"dataSource": "Orders_DS"
}
}
]
```


5. Note that the response now does not include the association ID 394 for the Orders test data model (386), which is correct.

You have successfully deleted an association related to a test data model.

Use APIs to Manage Fields in a Test Data Model

This article explains with the help of an example about how test data engineers (TDEs) can use exposed CA TDM Portal APIs to manage fields in test data models after they create them.

This article covers the following tasks. You perform all these tasks by using the APIs. You get the information about the available fields in a test data model. You then identify the fields that you want to update and delete.

Note: For more information about test data model concepts, persona-based tasks, prerequisites, assumptions, and considerations, see Use APIs to Design and Consume Automated Test Data Services.

This page refers to the following API Services:

- TestDataManager
- TDMProjectService
- TDMDataReservationService

Update a Field in a Test Data Model

The process to update a field in a test data model is as follows:

1. Get the Security Token.
2. Get the Project ID.
3. Get the Version ID.
4. Get the Test Data Model ID.
5. Get All Fields in the Identified Test Data Model.
6. Get Details of the Identified Field.
7. Update the Identified Field.

You can update the following properties:

- Display name of the field
- Display type of the field
- Display order of the field
- Display values of the field
- Whether to display the field

Get the Security Token

Use the login API to log in and generate a security token. You use your CA TDM Portal login credentials to generate the security token. You can then use the same security token to perform all other operations. The security token remains valid for 24 hours.

Follow these steps:

1. Access an application that allows you to encode your credentials to the Base64 format.
2. Enter your CA TDM Portal login credentials (in the format `<user name>:<password>`) in the source field.
   Note: Ensure that the credentials have appropriate permissions to perform all the required operations.
3. Click the option to encode the credentials. The encoded Base64 format for the example is displayed as follows:
   ```
   ZwRTaX5pc4SxYXSvcjptYXJtaXRl
   ```
4. Copy the encoded value.
5. Access the following CA TDM Portal API:
   ```
   POST https://<server>:<host>/TestDataManager/user/login
   ```
6. Enter the encoded value in the **Authorization** field, which is as follows for the example:
   ```
   Basic YWRtaW5pc3RyYXRvcjptYXJtaXRl
   ```
7. Run the API to get a security token for your credentials.
8. Note the value of the **token** parameter in the response body, which is as follows for the example:
   ```
   eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojUxm8ENTeRRrdOa-
   RQ5l4Ro
   ```
   You have successfully generated a security token that you can use in all the subsequent operations explained in this article.

Get the Project ID

Get the project ID that includes the required test data model. Note the project ID, because you will be using it in all the subsequent operations.

Follow these steps:

1. Access the following CA TDM Portal API:
   ```
   GET https://<server>:<host>/TDMProjectService/api/ca/v1/projects
   ```
2. Enter the security token in the **Authorization** field as follows:

   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
   RQ5l4Ro
   ```

3. Run the API and review the response body. The following example response is generated:

   ```
   [
   {
   "name": "Order",
   ```

```
"description": "This is Order Management project.",
"dateOrder": "YMD",
"id": 141357,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "StoreFront - Example Project - Oracle",
"description": "StoreFront - Oracle",
"dateOrder": "YMD",
"id": 1760,
"inheritTables": true,
"timestampPrecision": 6,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "StoreFront - Example Project - SQL Server",
"description": "StoreFront - Example Project - SQL Server",
"dateOrder": "YMD",
"id": 2234,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "TDMPublish_Centrica",
"description": "TDMPublish_Centrica",
"dateOrder": "YMD",
"id": 7739,
"inheritTables": true,
```

```
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
}
]
```


4. Identify the project and note the project ID. For this example, the project is Order with the ID 141357.

Get the Version ID

After you get the project ID, you must get the associated version ID. Note the version ID, because you will be using it in all the subsequent operations.

Follow these steps:

1. Access the following CA TDM Portal API:
   `GET https://<server>:<host>/TDMProjectService/api/ca/v1/projects/{projectId}/versions`
2. Enter the security token in the **Authorization** field as follows:


```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
RQ5l4Ro
```


3. Enter the project ID as 141357 in the **projectId** field.
4. Run the API and review the response body. The following example response is generated:


```
[
{
"id": 141358,
"name": "1.0",
"created": "2017-03-07T07:28:09+0000",
"description": "This is Order Management version 1.0.",
"projectName": null,
"levelDetails": null,
"registeredObjectCount": 0,
"tablesUsed": null,
"isGeneric": false
}
]
```


5. Note the version ID, which is 141358 in this example.

Get the Test Data Model ID

Get the list of test data models for your specific project and version. After you get the list, identify the test data model that includes the field that you want to update. Note the ID of the test data model.

Follow these steps:

1. Access the following CA TDM Portal API:
   ```
   GET https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels
   ```
2. Enter the security token in the **Authorization** field as follows:

   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
   RQ5l4Ro
   ```

3. Enter information in the following fields:
   - **projectId**
     Specifies the ID of the project for which you want to retrieve test data models. For this example, the value is 141357.
   - **versionId**
     Specifies the ID of the project version for which you want to retrieve test data models. For this example, the value is 141358.
4. Run the API and review the response body. The following example response is generated:

   ```
   {
   "numberOfTestDataModels": 2,
   "totalNumberOfTestDataModels": 2,
   "testDataModelsList": [
   {
   "id": 2587,
   "name": "Orders",
   "description": "This test data model is for Orders Management application.",
   "visible": true,
   "projectId": 141357,
   "versionId": 141358
   },
   {
   "id": 410,
   "name": "Product_Purchase",
   "description": "This test data model is for the Purchase application.",
   "visible": true,
   "projectId": 141357,
   "versionId": 141358
   }
   ]
   }
   ```

   Note that the specified project and version include two test data models: Orders and Product_Purchase.
5. Identify the test data model that includes the field; note the test data model ID. For this example, the test data model ID 2587 includes the field.

Get All Fields in the Identified Test Data Model

Get all the fields that are related to a specific test data model. After you get the list, identify the field that you want to update.

Follow these steps:

1. Access the following CA TDM Portal API:
   ```
   GET https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels/{testDataModelId}/fields
   ```
2. Enter the security token in the **Authorization** field as follows:

   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
   RQ5l4Ro
   ```

3. Enter information in the following fields:
   - **projectId**
     Specifies the ID of the project that includes the test data model for which you want to get the fields. For this example, the value is 141357.
   - **versionId**
     Specifies the ID of the project version that includes the test data model for which you want to get the fields. For this example, the value is 141358.
   - **testDataModelId**
     Specifies the ID of the test data model for which you want to get the associations. For this example, the value is 2587.
4. Run the API and review the response body. The following example response that includes all the fields for the selected test data model is generated:

   ```
   {
   "totalNoOfFields": 3,
   "fields": [
   {
   "associationId": null,
   "name": "OrderID",
   "displayName": "OrderID_MK_Display",
   "displayOrder": 1,
   "isVisible": true,
   "displayType": "TextBox",
   "dataType": "int",
   "displayValues": [],
   "id": 2589,
   "projectId": 141357,
   "versionId": 141358

   },
   {
   "associationId": 2594,
   "name": "UnitPrice",
   "displayName": "UnitPrice_DE_Display",
   "displayOrder": 1,
   ```

```
"isVisible": true,
"displayType": "TextBox",
"dataType": "money",
"displayValues": [],
"id": 2595,
"projectId": 141357,
"versionId": 141358

},
{
"associationId": 2591,
"name": "Quantity",
"displayName": "Quantity_DE_Display",
"displayOrder": 1,
"isVisible": true,
"displayType": "TextBox",
"dataType": "smallint",
"displayValues": [],
"id": 2592,
"projectId": 141357,
"versionId": 141358

}
],
"noOfFields": 3
}
```


Note that the response includes three fields with IDs 2589, 2595, and 2592 for the test data model (2587).

5. Identify the field that you want to update and note its ID. For this example, the filed with the ID 2592 is chosen for the update.

Get Details of the Identified Field

After you note the field ID that you want to update, you can retrieve its details to review the information in more detail.

Follow these steps:

1. Access the following CA TDM Portal API:
   `GET https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels/{testDataModelId}/fields/{fieldId}`
2. Enter the security token in the **Authorization** field as follows:


   ```
   Bearer
    eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
   RQ5l4Ro
   ```


3. Enter information in the following fields as follows:
   – **projectId**

Specifies the ID of the project related to the test data model that includes the field for which you want to get the details. For this example, the project ID value is 141357.

- **versionId**
  Specifies the ID of the project version related to the test data model that includes the field for which you want to get the details. For this example, the value of the version ID is 141358.
- **testDataModelId**
  Specifies the ID of the test data model that includes the field for which you want to get the details. For this example, the value of the test data model ID is 2587.
- **field**
  Specifies the ID of the field for which you want to get the details. For this example, the value of the association ID is 2592.

4. Run the API and review the response body. The following example response is generated for the field ID 2592:

```
{
"associationId": 2591,
"name": "Quantity",
"displayName": "Quantity_DE_Display",
"displayOrder": 1,
"isVisible": true,
"displayType": "TextBox",
"dataType": "smallint",
"displayValues": [],
"id": 2592,
"projectId": 141357,
"versionId": 141358

}
```

5. Review the properties that you want to update. For this example, the display name (Quantity_DE_Display) of the field is selected to be changed to Quantity_Display.

Update the Identified Field

After you review and identify the field property that you want to update, you can use the update field API to do the update.

Follow these steps:

1. Access the following CA TDM Portal API:
   `PUT https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels/{testDataModelId}/fields/{fieldId}`
2. Enter the security token in the **Authorization** field as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
RQ5l4Ro
```

3. Enter information in the following fields as follows:
   - **projectId**

Specifies the ID of the project related to the test data model that includes the field you want to update. For this example, the value is 141357.

- **versionId**
  Specifies the ID of the project version related to the test data model that includes the field you want to update. For this example, the value is 141358.
- **testDataModelId**
  Specifies the ID of the test data model that includes the field you want to update. For this example, the value is 386.
- **fieldId**
  Specifies the ID of the field that you want to update. For this example, the value of the association ID is 2592.
- **field**
  Specifies the payload that includes the field parameters. Specify the parameter values that you want to update. This payload includes the following parameters:
  - **associationId**
    Specifies the ID of the association that is related to the field you want to update. For this example, the value is 2591.
  - **displayName**
    Specifies the display name of the field that you want to update. For this example, the value is Quantity_DE_Display.
  - **displayOrder**
    Specifies the order in which you want to display this field (in the form) to testers. For this example, the value is 1.
  - **displayType**
    Specifies the display type of the field. For this example, the value is TextBox.
  - **displayValues**
    Specifies the default value for the field. For this example, no display values are used.
  - **isVisible**
    Specifies whether you want to display this field to testers. If yes, set the value to true; otherwise, set the value to false. For this example, the value is set to true.
  - **name**
    Specifies the name of the field. For this example, the value is Quantity.

  For this example, the association update payload is as follows. Note that the display name of the field is selected for the update:

```
{
"associationId": 2591,
"displayName": "Quantity_Display",
"displayOrder": 1,
"displayType": "TextBox",

"displayValues": [],
"isVisible": true,
"name": "Quantity"
}
```

4. Run the API and review the response body. The following example response is generated:

```
{
"associationId": 2591,
"name": "Quantity",
"displayName": "Quantity_Display",
```

```
"displayOrder": 1,
"isVisible": true,
"displayType": "TextBox",
"dataType": "smallint",
"displayValues": [],
"id": 2592,
"projectId": 141357,
"versionId": 141358

}
```


5. Review that the response includes the updated property. In this case, the display name of the field is changed to Quantity_Display.

You have successfully updated a field in a test data model.

Delete a Field in a Test Data Model

The process to delete a field in a test data model is as follows:

1. Get the Security Token.
2. Get the Project ID.
3. Get the Version ID.
4. Get the Test Data Model ID.
5. Get All Fields in the Identified Test Data Model.
6. Get Details of the Identified Field.
7. Delete the Identified Field.
8. Verify the Deletion.

Get the Security Token

Use the login API to log in and generate a security token. You use your CA TDM Portal login credentials to generate the security token. You can then use the same security token to perform all other operations. The security token remains valid for 24 hours. To get the security token and log into the CA TDM Portal, follow the instructions in the corresponding section in Update a Test Data Model.

Get the Project ID

Get the project ID that includes the required test data model. Note the project ID, because you will be using it in all the subsequent operations. To get the project ID, follow the detailed instructions in the corresponding section in Update a Field in a Test Data Model.

Summary of the example value used in this API is as follows:

- **Authorization:**Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro

The following response is generated, note the project ID (141357):


```
[
{
"name": "Order",
```

```
"description": "This is Order Management project.",
"dateOrder": "YMD",
"id": 141357,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "StoreFront - Example Project - Oracle",
"description": "StoreFront - Oracle",
"dateOrder": "YMD",
"id": 1760,
"inheritTables": true,
"timestampPrecision": 6,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "StoreFront - Example Project - SQL Server",
"description": "StoreFront - Example Project - SQL Server",
"dateOrder": "YMD",
"id": 2234,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "TDMPublish_Centrica",
"description": "TDMPublish_Centrica",
"dateOrder": "YMD",
"id": 7739,
"inheritTables": true,
```

```
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
}
]
```


Get the Version ID

After you get the project ID, you must get the associated version ID. Note the version ID, because you will be using it in all the subsequent operations. To get the version ID associated with the retrieved project ID, follow the detailed instructions in the corresponding section in Update a Field in a Test Data Model.

Summary of the example values used in this API is as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **projectId:** 141357

The following example response is generated; note the version ID (141358):


```
[
{
"id": 141358,
"name": "1.0",
"created": "2017-03-07T07:28:09+0000",
"description": "This is Order Management version 1.0.",
"projectName": null,
"levelDetails": null,
"registeredObjectCount": 0,
"tablesUsed": null,
"isGeneric": false
}
]
```


Get the Test Data Model ID

Get the list of test data models for your specific project and version. After you get the list, identify the test data model that includes the field that you want to delete. Note the ID of the test data model. To get the ID, follow the detailed instructions in the corresponding section in Update a Field in a Test Data Model.

Summary of the example values used in this API is as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojU RQ5l4Ro
- **projectId:** 141357
- **versionId:** 141358

The following example response is generated. Note that the specified project and version include two test data models: Orders and Product_Purchase. Identify the test data model that includes the field; note the test data model ID. For this example, the test data model ID 2587 includes the field.


```
{
"numberOfTestDataModels": 2,
"totalNumberOfTestDataModels": 2,
"testDataModelsList": [
{
"id": 2587,
"name": "Orders",
"description": "This test data model is for Orders Management application.",
"visible": true,
"projectId": 141357,
"versionId": 141358
},
{
"id": 410,
"name": "Product_Purchase",
"description": "This test data model is for the Purchase application.",
"visible": true,
"projectId": 141357,
"versionId": 141358
}
]
}
```


Get All Fields in the Identified Test Data Model

Get all the fields that are related to the selected test data model. After you get the list, identify the field that you want to delete. To get list of fields in a test data model, follow the detailed instructions in the corresponding section in Update a Field in a Test Data Model.

Summary of the example values used in this API are as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojU RQ5l4Ro
- **projectId:** 141357
- **versionId:** 141358
- **testDataModelId:** 2587

The following example response is generated. Note that the response includes three fields with IDs 2589, 2595, and 2592 for the test data model (2587). Also, note that the field 2592 now shows the updated display name Quantity_Display,

which you updated in the previous section. Identify the field that you want to delete and note its ID. For this example, the filed with the ID 2592 is chosen for the delete operation:


```
{
"totalNoOfFields": 3,
"fields": [
{
"associationId": null,
"name": "OrderID",
"displayName": "OrderID_MK_Display",
"displayOrder": 1,
"isVisible": true,
"displayType": "TextBox",
"dataType": "int",
"displayValues": [],
"id": 2589,
"projectId": 141357,
"versionId": 141358

},
{
"associationId": 2594,
"name": "UnitPrice",
"displayName": "UnitPrice_DE_Display",
"displayOrder": 1,
"isVisible": true,
"displayType": "TextBox",
"dataType": "money",
"displayValues": [],
"id": 2595,
"projectId": 141357,
"versionId": 141358

},
{
"associationId": 2591,
"name": "Quantity",
"displayName": "Quantity_Display",
"displayOrder": 1,
"isVisible": true,
"displayType": "TextBox",
"dataType": "smallint",
"displayValues": [],
"id": 2592,
"projectId": 141357,
"versionId": 141358
```

```
}
],
"noOfFields": 3
}
```

Get Details of the Identified Field

After you note the field ID that you want to delete, you can retrieve its details to review the information in more detail. To get details of a specific field, follow the detailed instructions in the corresponding section in Update a Field in a Test Data Model.

Summary of the example values used in this API are as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **projectId:** 141357
- **versionId:** 141358
- **testDataModelId:** 2587
- **fieldId:** 2592

The following example response is generated for the field ID 2592. Review the properties to verify that you want to delete this field:

```
{
"associationId": 2591,
"name": "Quantity",
"displayName": "Quantity_Display",
"displayOrder": 1,
"isVisible": true,
"displayType": "TextBox",
"dataType": "smallint",
"displayValues": [],
"id": 2592,
"projectId": 141357,
"versionId": 141358

}
```

Delete the Identified Field

After you identify and confirm the appropriate field in a test data model, you can go ahead and delete it.

Follows these steps:

1. Access the following CA TDM Portal API:
   ```
   DELETE https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels/{testDataModelId}/fields/{fieldId}
   ```
2. Enter the security token in the **Authorization** field as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
RQ5l4Ro
```

3. Enter information in the following fields as follows:
   - **projectId**
     Specifies the ID of the project that includes the test data model from which you want to delete the field. For this example, the value of the project ID is 141357.
   - **versionId**
     Specifies the ID of the project version that includes the test data model from which you want to delete the field. For this example, the value of the version ID is 141358.
   - **testDataModelId**
     Specifies the ID of the test data model from which you want to delete the field. For this example, the value of the test data model ID is 2587.
   - **fieldId**
     Specifies the ID of the field you want to delete. For this example, the value of the field ID is 2592.
4. Run the API and review the response body:

```
{
"message": "Field is deleted successfully."
}
```

5. Review that the response includes a message that states that the field has been deleted successfully.

Verify the Deletion

After you run the delete field API to delete the field, you can verify whether the field is appearing in the test data model.

Follow these steps:

1. Access the following CA TDM Portal API:
   `GET https://<server>:<host>/TDMDataReservationService/api/ca/v1/testDataModels/{testDataModelId}/fields`
2. Enter the security token in the **Authorization** field as follows:

```
Bearer
 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0v
RQ5l4Ro
```

3. Enter information in the following fields:
   - **projectId**
     Specifies the ID of the project that includes the test data model for which you want to get the fields. For this example, the value is 141357.
   - **versionId**
     Specifies the ID of the project version that includes the test data model for which you want to get the fields. For this example, the value is 141358.
   - **testDataModelId**
     Specifies the ID of the test data model for which you want to get the associations. For this example, the value is 2587.
4. Run the API and review the response body. The following example response that includes all the fields for the selected test data model is generated:

```
{
"totalNoOfFields": 2,
"fields": [
{
"associationId": null,
"name": "OrderID",
"displayName": "OrderID_MK_Display",
"displayOrder": 1,
"isVisible": true,
"displayType": "TextBox",
"dataType": "int",
"displayValues": [],
"id": 2589,
"projectId": 141357,
"versionId": 141358

},
{
"associationId": 2594,
"name": "UnitPrice",
"displayName": "UnitPrice_DE_Display",
"displayOrder": 1,
"isVisible": true,
"displayType": "TextBox",
"dataType": "money",
"displayValues": [],
"id": 2595,
"projectId": 141357,
"versionId": 141358

}
],
"noOfFields": 2
}
```


5. Note that the response no longer includes the field 2592. The response now includes only two remaining fields with IDs 2589 and 2595 for the test data model 2587.

You have successfully deleted a field in a test data model.

Additional API Usage Examples

This section includes information about some additional API usage examples. All these examples use the sample Northwind database that is available for Microsoft SQL Server. Refer the Microsoft website to download the Northwind database.

Use APIs to Verify Concurrency During Data Reservation

The Portal maintains concurrency during the data reservation request. If two users try to reserve the same data at the same time, the Portal creates reservation requests for both the users. However, the request of the first user succeeds and that of the second user fails. The second user is not able to reserve the data, because the resources are already blocked by the other user.

Note: For more information about test data model concepts, prerequisites, assumptions, and considerations, see Use APIs to Design and Consume Automated Test Data Services.

Follow these steps to verify concurrency during the data reservation process:

1. Get the Security Token.
2. Get the Project ID.
3. Get the Version ID.
4. Get the Test Data Model ID.
5. Get the Environment ID.
6. Get the Field ID.
7. Find the Data.
8. Reserve the Data (User 1 and User 2).
9. Get the Reservation Status (User 1 and User 2).

Note: This example uses the Northwind sample database that is available for Microsoft SQL Server. Refer the Microsoft website to download the Northwind sample database.

Get the Security Token

Use the login API to log in and generate a security token. You use your CA TDM Portal login credentials to generate the security token. You can then use the same security token to perform all other operations. The security token remains valid for 24 hours. To get the security token and log into the CA TDM Portal, follow the instructions in the Use APIs to Manage Test Data Models.

Get the Project ID

Get the project ID that includes the required test data model. Note the project ID, because you will be using it in all the subsequent operations. To get the project ID, follow the detailed instructions in the corresponding section in Use APIs to Manage Test Data Models.

Summary of the example value used in this API is as follows:

- **Authorization:**Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojU
RQ5l4Ro

The following response is generated, note the project ID (141357):


```
[
{
"name": "Order",
"description": "This is Order Management project.",
"dateOrder": "YMD",
"id": 141357,
"inheritTables": true,
"timestampPrecision": 3,
```

```
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "StoreFront - Example Project - Oracle",
"description": "StoreFront - Oracle",
"dateOrder": "YMD",
"id": 1760,
"inheritTables": true,
"timestampPrecision": 6,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "StoreFront - Example Project - SQL Server",
"description": "StoreFront - Example Project - SQL Server",
"dateOrder": "YMD",
"id": 2234,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "TDMPublish_Centrica",
"description": "TDMPublish_Centrica",
"dateOrder": "YMD",
"id": 7739,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
```

```
"versions": [],
"grantedFunctions": []
}
]
```


Get the Version ID

After you get the project ID, you must get the associated version ID. Note the version ID, because you will be using it in all the subsequent operations. To get the version ID associated with the retrieved project ID, follow the detailed instructions in the corresponding section in Use APIs to Manage Test Data Models.

Summary of the example values used in this API is as follows:

- **Authorization:**Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **projectId:** 141357

The following example response is generated; note the version ID (141358):


```
[
{
"id": 141358,
"name": "1.0",
"created": "2017-03-07T07:28:09+0000",
"description": "This is Order Management version 1.0.",
"projectName": null,
"levelDetails": null,
"registeredObjectCount": 0,
"tablesUsed": null,
"isGeneric": false
}
]
```


Get the Test Data Model ID

Get the list of test data models for your specific project and version. After you get the list, identify the test data model that is related to the data that you want to find and reserve. To get all the test data models for the retrieved project and version, follow the detailed instructions in the corresponding section in Use APIs to Manage Test Data Models.

Summary of the example values used in this API is as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **projectId:** 141357
- **versionID:** 141358

The following response is generated:

```
{
"numberOfTestDataModels": 2,
"totalNumberOfTestDataModels": 2,
"testDataModelsList": [
{
"id": 3213,
"name": "Employee_Orders",
"description": "This is an employee order test data model.",
"visible": true,
"projectId": 141357,
"versionId": 141358
},
{
"id": 5209,
"name": "Orders",
"description": "This is Orders test data model.",
"visible": true,
"projectId": 141357,
"versionId": 141358
}
]
}
```


For this example, the test data model Orders with the ID 5209 is chosen for the subsequent operations.

Get the Environment ID

Get the list of environments for your specific project and version. After you get the list, identify the environment that you want to use. To get all the environments for the retrieved project and version, follow the detailed instructions in the corresponding section in Use APIs to Manage Environments.

Summary of the example values used in this API are as follows:

- **Authorization:**Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojLRQ5l4Ro
- **projectId:** 141357
- **versionID:** 141358

The following response is generated:


```
{
"numberOfElements": 1,
"totalNumberOfElements": 1,
"elements": [
{
"id": 2757,
"name": "Order_Environment",
"description": "This environment is for orders.",
```

```
"projectID": 141357,
"versionID": 141358,
"createdBy": "John",
"modifiedBy": "John",
"creationDate": "2017-03-21 11:44:48.303",
"modifiedDate": "2017-03-21 11:44:48.303"
}
]
}
```


For this example, the environment Order_Environment with the ID 2757 is chosen for the subsequent operations.

Get the Field ID

Get all the fields that are related to a specific test data model. After you get the list, identify the field that you want to use. To get all the fields for the retrieved project and version, follow the detailed instructions in the corresponding section in Use APIs to Manage Fields in a Test Data Model.

Summary of the example values used in this API are as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **projectId:** 141357
- **versionId:** 141358
- **testDataModelId:** 5209

The following response is generated. Note that the total number of fields in this test data model is 7. The field ShipCity with the ID 5737 is identified to be used for the relevant operation:


```
{
"totalNoOfFields": 7,
"fields": [
{
"associationId": null,
"name": "OrderID",
"displayName": "OrderID",
"displayOrder": 1,
"isVisible": true,
"displayType": "TextBox",
"dataType": "int",
"displayValues": [],
"id": 5210,
"projectId": 141357,
"versionId": 141358

},
{
"associationId": null,
```

```
"name": "ShipCity",
"displayName": "ShipCity",
"displayOrder": 2,
"isVisible": true,
"displayType": "TextBox",
"dataType": "nvarchar",
"displayValues": [],
"id": 5737,
"projectId": 141357,
"versionId": 141358

},
{
"associationId": null,
"name": "ShipRegion",
"displayName": "ShipRegion",
"displayOrder": 3,
"isVisible": true,
"displayType": "TextBox",
"dataType": "nvarchar",
"displayValues": [],
"id": 5738,
"projectId": 141357,
"versionId": 141358

},
{
"associationId": null,
"name": "ShipPostalCode",
"displayName": "ShipPostalCode",
"displayOrder": 4,
"isVisible": true,
"displayType": "TextBox",
"dataType": "nvarchar",
"displayValues": [],
"id": 5749,
"projectId": 141357,
"versionId": 141358

},
{
"associationId": 5781,
"name": "Discount",
"displayName": "Discount",
"displayOrder": 1,
"isVisible": true,
"displayType": "TextBox",
```

```
"dataType": "real",
"displayValues": [],
"id": 5782,
"projectId": 141357,
"versionId": 141358

},
{
"associationId": 5781,
"name": "Quantity",
"displayName": "Quantity",
"displayOrder": 2,
"isVisible": true,
"displayType": "TextBox",
"dataType": "smallint",
"displayValues": [],
"id": 5801,
"projectId": 141357,
"versionId": 141358

},
{
"associationId": 5841,
"name": "UnitPrice",
"displayName": "UnitPrice",
"displayOrder": 1,
"isVisible": true,
"displayType": "TextBox",
"dataType": "money",
"displayValues": [],
"id": 5842,
"projectId": 141357,
"versionId": 141358

}
],
"noOfFields": 7
}
```


Find the Data

To be able to successfully reserve the data, you must first find the relevant data. To find the data, follow the detailed instructions in the corresponding section in Use APIs to Find and Reserve Test Data.

Summary of the example values used in this API are as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **projectId:** 141357
- **versionId:** 141358
- **testDataModelId:** 5209
- **requestBody:**

```
{
"environmentId": 2757,
"filters": [
{
"fieldId": 5737,
"operator": "CONTAINS",
"values": [
"CHEN"
]
}
],
"includeReservedRecords": true,
"startAfterValues": {}
}
```

The following response is generated when CHEN is used as a value for ShipCity field. Three records with the order ID values 84, 85, and 91 are identified for the reservation:

```
{
"startAfterValues": {
"OrderID": 98
},
"records": [
{
"recordId": {
"keys": {
"OrderID": "2"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "24129",
"OrderID": 2,
"ShipRegion": "TS"
}
},
{
"recordId": {
```

```
"keys": {
"OrderID": "3"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "50947",
"OrderID": 3,
"ShipRegion": "TN"
}
},
{
"recordId": {
"keys": {
"OrderID": "4"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "20997",
"OrderID": 4,
"ShipRegion": "AP"
}
},
{
"recordId": {
"keys": {
"OrderID": "9"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "79091",
"OrderID": 9,
"ShipRegion": "KS"
}
},
{
"recordId": {
"keys": {
"OrderID": "20"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "60667",
```

```
"OrderID": 20,
"ShipRegion": "TN"
}
},
{
"recordId": {
"keys": {
"OrderID": "28"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "47832",
"OrderID": 28,
"ShipRegion": "TN"
}
},
{
"recordId": {
"keys": {
"OrderID": "34"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "78280",
"OrderID": 34,
"ShipRegion": "TS"
}
},
{
"recordId": {
"keys": {
"OrderID": "39"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "15368",
"OrderID": 39,
"ShipRegion": "KS"
}
},
{
"recordId": {
"keys": {
```

```
"OrderID": "42"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "30446",
"OrderID": 42,
"ShipRegion": "TS"
}
},
{
"recordId": {
"keys": {
"OrderID": "43"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "74558",
"OrderID": 43,
"ShipRegion": "KS"
}
},
{
"recordId": {
"keys": {
"OrderID": "49"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "41052",
"OrderID": 49,
"ShipRegion": "TS"
}
},
{
"recordId": {
"keys": {
"OrderID": "54"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "61016",
"OrderID": 54,
```

```
"ShipRegion": "TN"
}
},
{
"recordId": {
"keys": {
"OrderID": "63"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "52565",
"OrderID": 63,
"ShipRegion": "TN"
}
},
{
"recordId": {
"keys": {
"OrderID": "66"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "67214",
"OrderID": 66,
"ShipRegion": "TS"
}
},
{
"recordId": {
"keys": {
"OrderID": "68"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "22576",
"OrderID": 68,
"ShipRegion": "TS"
}
},
{
"recordId": {
"keys": {
"OrderID": "78"
```

```
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "55952",
"OrderID": 78,
"ShipRegion": "TN"
}
},
{
"recordId": {
"keys": {
"OrderID": "82"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "75250",
"OrderID": 82,
"ShipRegion": "AP"
}
},
{
"recordId": {
"keys": {
"OrderID": "84"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "49626",
"OrderID": 84,
"ShipRegion": "TN"
}
},
{
"recordId": {
"keys": {
"OrderID": "85"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "47445",
"OrderID": 85,
"ShipRegion": "KS"
```

```
}
},
{
"recordId": {
"keys": {
"OrderID": "91"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "18711",
"OrderID": 91,
"ShipRegion": "TS"
}
},
{
"recordId": {
"keys": {
"OrderID": "95"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "17244",
"OrderID": 95,
"ShipRegion": "KS"
}
},
{
"recordId": {
"keys": {
"OrderID": "97"
}
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "36855",
"OrderID": 97,
"ShipRegion": "KS"
}
},
{
"recordId": {
"keys": {
"OrderID": "98"
}
```

```
},
"columnValues": {
"ShipCity": "CHEN",
"ShipPostalCode": "62984",
"OrderID": 98,
"ShipRegion": "TS"
}
}
]
}
```

Reserve the Data

After finding the data, both the users send the reservation request for the three identified records: 84, 85, and 91. To reserve the data, follow the detailed instructions in the corresponding section in Use APIs to Find and Reserve Test Data.

Summary of the example values used in this API are as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **projectId:** 141357
- **versionId:** 141358
- **reservationInfo:**

```
{
"dataModelId": 5209,
"dataModelName": "Orders",
"environmentId": 2757,
"environmentName": "Order_Environment",
"resErrorMessage": "Reservation",
"reservationId": 0,
"reservationName": "Order_Reservation",
"reservationState": "UNDEFINED",
"resources": [
{
"dataModelId": 5209,
"modelKeys": {"OrderID":"84"}
},
{
"dataModelId": 5209,
"modelKeys": {"OrderID":"85"}
},
{
"dataModelId": 5209,
"modelKeys": {"OrderID":"91"}
}
]
```

```
}
```

The following responses are generated for the two users who are trying to reserve the same data at the same time:

First User

```
{
"reservationId": 664
}
```

Second User

```
{
"reservationId": 665
}
```

Get the Reservation Status for Both the Requests

After both the users submit their data reservation requests and note the reservation IDs, they can get the status of their requests. In this case, the request of the first user succeeds while that of the second user fails. To get the reservation status, follow the detailed instructions in the corresponding section in Use APIs to Find and Reserve Test Data.

Summary of the example values used in this API are as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **projectId:** 141357
- **versionId:** 141358
- **reservationId:** 664 for first user and 665 for the second user

The following response is generated:

First User

Note that the reservation status for the first user is marked as SUCCESS with all the three records 84, 85, and 91 reserved succesfully.

```
{
"reservationId": 664,
"reservationName": "Orders_Employee_Order",
"reservationState": "SUCCESS",
"dataModelId": 5209,
"dataModelName": "Orders",
"environmentId": 2757,
"environmentName": "Order_Environment",
"projectId": 141357,
"versionId": 141358,
"reservedBy": 5084,
```

```
"scheduledDate": "2017-03-21T09:51:22.948Z",
"expiryDate": "2117-03-21T09:51:22.948Z",
"resources": [
{
"dataModelId": 5209,
"reservationId": 664,
"projectId": 141357,
"versionId": 141358,
"modelKeys": {
"OrderID": "84"
}
},
{
"dataModelId": 5209,
"reservationId": 664,
"projectId": 141357,
"versionId": 141358,
"modelKeys": {
"OrderID": "85"
}
},
{
"dataModelId": 5209,
"reservationId": 664,
"projectId": 141357,
"versionId": 141358,
"modelKeys": {
"OrderID": "91"
}
}
],
"resErrorMessage": null,
"jobPayload": null,
"releaseDate": null
}
```


Second User

Note that the reservation status for the second user is marked as FAILED. Also note the message that states that the resources are already blocked for another user.


```
{
"reservationId": 665,
"reservationName": "Orders_Employee_Order",
"reservationState": "FAILED",
"dataModelId": 5209,
```

```
"dataModelName": "Orders",
"environmentId": 2757,
"environmentName": "Order_Environment",
"projectId": 141357,
"versionId": 141358,
"reservedBy": 6075,
"scheduledDate": "2017-03-21T09:51:23.261Z",
"expiryDate": "2117-03-21T09:51:23.261Z",
"resources": null,
"resErrorMessage": "Resources are already blocked for another reservation",
"jobPayload": null,
"releaseDate": "2017-03-21T09:51:23.290Z"
}
```


Use APIs to Filter the Find Data Results

The CA TDM Portal APIs allow to use various operators in the fields to filter the data during the find test data process. Each field based on its data type, allows only specific operators and the values in specific format.

Follow these steps to filter the data using various operators in the fields of specific data type:

1. Get the Security Token
2. Get the Project ID
3. Get the Version ID
4. Get the Test Data Model ID
5. Get the Environment ID
6. Get the Field ID
7. Find the Data
   a. Find Data filtered by Field type of nvarchar
   b. Find Data filtered by Field type of int
   c. Find Data filtered by Field type of datetime

Get the Security Token

Use the login API to log in and generate a security token. You use your CA TDM Portal login credentials to generate the security token. You can then use the same security token to perform all other operations. The security token remains valid for 24 hours. To get the security token and log into the CA TDM Portal, follow the instructions in the Use APIs to Manage Test Data Models.

Get the Project ID

Get the project ID that includes the required test data model. Note the project ID, because you will be using it in all the subsequent operations. To get the project ID, follow the detailed instructions in the corresponding section in Use APIs to Manage Test Data Models.

Summary of the example value used in this API is as follows:

- **Authorization:**Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlzdHJhdG9yIiwiYXVkIjoiQUxMIiwiUFdEX0hBU0hfQ0xBSU

The following response is generated, note the project ID (167733):

```
[
{
"name": "Order",
"description": "This is Order Management project.",
"dateOrder": "YMD",
"id": 141357,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},

{
 "name": "MyOrders",
 "description": "MyOrders",
 "dateOrder": "YMD",
 "id":
 167733,
 "inheritTables": true,
 "timestampPrecision": 3,
 "type": "DB",
 "levels": [],
 "created": null,
 "updated": null,
 "versions": [],
 "grantedFunctions": []
 },



{
"name": "StoreFront - Example Project - Oracle",
"description": "StoreFront - Oracle",
"dateOrder": "YMD",
"id": 1760,
"inheritTables": true,
"timestampPrecision": 6,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
```

```
},
{
"name": "StoreFront - Example Project - SQL Server",
"description": "StoreFront - Example Project - SQL Server",
"dateOrder": "YMD",
"id": 2234,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
},
{
"name": "TDMPublish_Centrica",
"description": "TDMPublish_Centrica",
"dateOrder": "YMD",
"id": 7739,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [],
"created": null,
"updated": null,
"versions": [],
"grantedFunctions": []
}
]
```


Get the Version ID

After you get the project ID, you must get the associated version ID. Note the version ID, because you will be using it in all the subsequent operations. To get the version ID associated with the retrieved project ID, follow the detailed instructions in the corresponding section in Use APIs to Manage Test Data Models.

Summary of the example values used in this API is as follows:

- **Authorization:**Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojU
RQ5l4Ro
- **projectId:** 167733

The following example response is generated; note the version ID (167734):


```
[
 {
```

```
  "id": 167734,
  "name": "1.0",
  "created": "2017-03-24T09:05:33+0000",
  "description": "1.0",
  "projectName": null,
  "levelDetails": null,
  "registeredObjectCount": 0,
  "tablesUsed": null,
  "isGeneric": false
  }
 ]
```


Get the Test Data Model ID

Get the list of test data models for your specific project and version. After you get the list, identify the test data model that is related to the data that you want to find and reserve. To get all the test data models for the retrieved project and version, follow the detailed instructions in the corresponding section in Use APIs to Manage Test Data Models.

Summary of the example values used in this API is as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlzdHJhdG9yIiwiYXVkIjoiQUxMIiwiUFdEX0hBU0hfQ0xBSU0iOiI4MzkzMTQyODM
- **projectId:** 167733
- **versionID:** 167734

The following response is generated:


```
{
 "numberOfTestDataModels": 1,
 "totalNumberOfTestDataModels": 1,
 "testDataModelsList": [
 {
 "id": 2030,
 "name": "MyOrdersDataModel",
 "description": "MyOrdersDataModel",
 "visible": true,
 "projectId": 167733,
 "versionId": 167734
 }
 ]
}
```


For this example, the test data model Orders with the ID 5209 is chosen for the subsequent operations.

Get the Environment ID

Get the list of environments for your specific project and version. After you get the list, identify the environment that you want to use. To get all the environments for the retrieved project and version, follow the detailed instructions in the corresponding section in Use APIs to Manage Environments.

Summary of the example values used in this API are as follows:

- **Authorization:**Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlzdHJhdG9yIiwiYXVkIjoiQUxMIiwiUFdEX0hBU0hfQ0xBSU0iOiI4MzkzMTQyODM
- **projectId:** 167733
- **versionID:** 167734

The following response is generated:


```
{
 "numberOfElements": 1,
 "totalNumberOfElements": 1,
 "elements": [
 {
 "id": 2026,
 "name": "MyOrdersEnvironment",
 "description": "MyOrdersEnvironment",
 "projectID": 167733,
 "versionID": 167734,
 "createdBy": "Administrator",
 "modifiedBy": "Administrator",
 "creationDate": "2017-03-24 14:37:54.241",
 "modifiedDate": "2017-03-24 15:15:14.240"
 }
 ]
}
```


For this example, the environment Order_Environment with the ID 2026 is chosen for the subsequent operations.

Get the Field ID

Get all the fields that are related to a specific test data model. After you get the list, identify the field that you want to use. To get all the fields for the retrieved project and version, follow the detailed instructions in the corresponding section in Use APIs to Manage Fields in a Test Data Model.

Summary of the example values used in this API are as follows:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlzdHJhdG9yIiwiYXVkIjoiQUxMIiwiUFdEX0hBU0hfQ0xBSU0iOiI4MzkzMTQyODM
- **projectId:** 167733
- **versionId:** 167734
- **testDataModelId:** 2030

The following response is generated. Note that the total number of fields in this test data model are 11. Each of these fields are of different data type.


```
{
 "totalNoOfFields": 6,
 "fields": [
 {
```

```
 "associationId": null,
 "name": "FirstName",
 "displayName": "FirstName",
 "displayOrder": 4,
 "isVisible": true,
 "displayType": "TextBox",
 "dataType": "nvarchar",
 "displayValues": [],
 "id": 2033,
 "projectId": 167733,
 "versionId": 167734

},
 {
 "associationId": null,
 "name": "HireDate",
 "displayName": "HireDate",
 "displayOrder": 2,
 "isVisible": true,
 "displayType": "TextBox",
 "dataType": "datetime",
 "displayValues": [],
 "id": 2035,
 "projectId": 167733,
 "versionId": 167734

},
 {
 "associationId": 2038,
 "name": "RegionID",
 "displayName": "RegionID",
 "displayOrder": 1,
 "isVisible": true,
 "displayType": "TextBox",
 "dataType": "int",
 "displayValues": [],
 "id": 2039,
 "projectId": 167733,
 "versionId": 167734

},
 {
 "associationId": null,
 "name": "EmployeeID",
 "displayName": "EmployeeID",
 "displayOrder": 1,
 "isVisible": true,
```

```
 "displayType": "TextBox",
 "dataType": "int",
 "displayValues": [],
 "id": 2032,
 "projectId": 167733,
 "versionId": 167734

},
 {
 "associationId": null,
 "name": "Title",
 "displayName": "Title",
 "displayOrder": 3,
 "isVisible": true,
 "displayType": "TextBox",
 "dataType": "nvarchar",
 "displayValues": [],
 "id": 2034,
 "projectId": 167733,
 "versionId": 167734

},
 {
 "associationId": null,
 "name": "LastName",
 "displayName": "LastName",
 "displayOrder": 5,
 "isVisible": true,
 "displayType": "TextBox",
 "dataType": "nvarchar",
 "displayValues": [],
 "id": 2036,
 "projectId": 167733,
 "versionId": 167734

}
 ],
 "noOfFields": 6
}
```


Find Data

You can find only the relevant data by using filters specifying an operator and a value for each field to filter the find data results. You must specify the values in a specific format for each field based on its data type. To find the data, follow the detailed instructions in the corresponding section in Use APIs to Find and Reserve Test Data.

Find Data filtered by Field type of *nvarchar*

Summary of the example values used in this API are as follows:

- **Authorization:** Bearer Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlzdHJhdG9yIiwiYXVkIjoiQUxMIiwiUFdEX0hBU0hfQ0xBSU0iOiI4MzkzMTQyODM
- **projectId:** 167733
- **versionId:** 167734
- **testDataModelId:** 2030
- **requestBody:**

```
{
 "environmentId": 2026,
 "filters": [
 {
 "fieldId": 2033,
 "operator": "EQUALS",
 "values": [
 "chen"
 ]
 }
 ],
 "includeReservedRecords": true,
 "startAfterValues": {}
}
```

The following response is generated when "chen" is used as a value for first name field. One matching record with the first name "Pchenitchn" is returned:

```
{
  "startAfterValues": {
    "FirstName": "Conrad",
    "HireDate": null,
    "Title": null,
    "LastName": "Daniel",
    "EmployeeID": 60100
  },
  "records": [
{
 "recordId": {
```

```
  "keys": {
  "FirstName": "Pchenitchn",
  "HireDate": null,
  "Title": null,
  "LastName": "Ross",
  "EmployeeID": "60764"
  }
  },
  "columnValues": {
  "FirstName": "Pchenitchn",
  "HireDate": null,
  "Title": null,
  "LastName": "Ross",
  "EmployeeID": 60764
  }
  }
  ]
 }
```


Find Data filtered by Field type of *int*

Summary of the example values used in this API are as follows:

- **Authorization:** Bearer Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlzdHJhdG9yIiwiYXVkIjoiQUxMIiwiUFdEX0hBU0hfQ0xBSU0iOiI4MzkzMTQyODM
- **projectId:** 167733
- **versionId:** 167734
- **testDataModelId:** 2030
- **requestBody:**


```
  {
  "environmentId": 2026,
  "filters": [
  {
  "fieldId": 2032,
  "operator": "EQUALS",
  "values": [
  "60004"
  ]
  }
  ],
  "includeReservedRecords": true,
  "startAfterValues": {}
  }
```

The following response is generated with one matching record:


```
{
  "startAfterValues": {
    "FirstName": "Arijune",
    "HireDate": null,
    "Title": null,
    "LastName": "Krishna",
    "EmployeeID": 60001
  },
  "records": [
 {
      "recordId": {
        "keys": {
          "FirstName": "Petitpas",
          "HireDate": null,
          "Title": null,
          "LastName": "Vick",
          "EmployeeID": "60004"
        }
      },
      "columnValues": {
        "FirstName": "Petitpas",
        "HireDate": null,
        "Title": null,
        "LastName": "Vick",
```

```
        "EmployeeID": 60004

      }

  ]

}
```

Find Data filtered by Field type of *datetime*

Summary of the example values used in this API are as follows:

- **Authorization:** Bearer Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlzdHJhdG9yIiwiYXVkIjoiQUxMIiwiUFdEX0hBU0hfQ0xBSU0iOiI4MzkzMTQyODM
- **projectId:** 167733
- **versionId:** 167734
- **testDataModelId:** 2030
- **requestBody:**

```
{
"environmentId": 2026,
"filters": [
{
"fieldId": 2035,
"operator": "GREATER_THAN",
"values": [
"2017-03-22 22:32:28.012"
]
}
],
"includeReservedRecords": true,
"startAfterValues": {}
}
```

The following response is generated with two matching records:

```
{

  "startAfterValues": {

    "FirstName": "Conrad",

    "HireDate": null,

    "Title": null,
```

```
    "LastName": "Daniel",
    "EmployeeID": 60100
  },
  "records": [

{
      "recordId": {
        "keys": {
          "FirstName": "Johannsen",
          "HireDate": null,
          "Title": null,
          "LastName": "Hang",
          "EmployeeID": "60087"
        }
      },
      "columnValues": {
        "FirstName": "Johannsen",
        "HireDate": null,
        "Title": null,
        "LastName": "Hang",
        "EmployeeID": 60087
      }
    },
    {
      "recordId": {
```

```
          "keys": {
            "FirstName": "Rosindale",
            "HireDate": null,
            "Title": null,
            "LastName": "Arnold",
            "EmployeeID": "60088"
          }
        },
        "columnValues": {
          "FirstName": "Rosindale",
          "HireDate": null,
          "Title": null,
          "LastName": "Arnold",
          "EmployeeID": 60088
        }
      }
  }
  ]
  }
```

Use APIs to Define Associations with Self Reference

This article explains with the help of an example about how test data engineers (TDEs) can use exposed CA TDM Portal APIs to define self referencing associations.

Note: For more information about dynamic test data reservation concepts, prerequisites, assumptions, and considerations, see Use APIs to Design and Consume Automated Test Data Services.

The complete process for defining associations with self reference is as follows:

The example in this article uses the Northwind sample database that is available for Microsoft SQL Server. Refer the Microsoft website to download the Northwind sample database.

Get a Security Token

Use the login API to log in and generate a security token. You use your CA TDM Portal login credentials to generate the security token. You can then use the same security token to perform all other operations. The security token remains valid for 24 hours. To get the security token and log into the CA TDM Portal, follow the instructions in the corresponding section in Use APIs to Design and Consume Automated Test Data Services.

The security token generated for authorization in this example is as below:

```
eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcB
RQ5l4Ro
```

Create a Connection Profile

For detailed information about how to create a connection profile, see the corresponding section in Use APIs to Design and Consume Automated Test Data Services.

The following values are used in this example:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojU RQ5l4Ro
- **profile:**

```
{
 "name": "Employee_SQLServer",

"description": "Employee_SQLServer",
 "dbType": "sql server",
 "server": "matlo01-IP036",
 "port": "1443",
 "instance": "",
 "service": "",
 "database": "employee",
 "schema": "dbo",
 "username": "sa",
 "password": "NsH0JPgmqEsrcvpO46UW6F7wEjOqNWNr1RRK+zewswiGXNS2d+S8DbFOM3t4",
 "datasourceUrl": "jdbc:sqlserver://matlo01-IP036;database=employee",
 "datasourceDriver": "com.microsoft.sqlserver.jdbc.SQLServerDriver",
 "created": 1457000111753,
 "modified": 1487585989363,
 "createdBy": 1,
 "additionalConnectionProperties": ""
 }
```

The following response is returned. Note the connection profile "**Employee_SQLServer**" is created in this example.

```
{
```

```
    "name": "Employee_SQLServer",

    "description": "Employee_SQLServer",

    "dbType": "sql server",

    "server": "matlo01-IP036",

    "port": "1443",

    "instance": "",

    "service": "",

    "database": "employee",

    "schema": "dbo",

    "username": "sa",

    "password": "NsH0JPgmqEsrcvpO46UW6F7wEjOqNWNr1RRK+zewswiGXNS2d+S8DbFOM3t4",

    "datasourceUrl": "jdbc:sqlserver://matlo01-IP036;database=employee",

    "datasourceDriver": "com.microsoft.sqlserver.jdbc.SQLServerDriver",

    "created": 1457000111753,

    "modified": 1487585989363,

    "createdBy": 1,

    "additionalConnectionProperties": ""

  },
```


Create a Project

For detailed information about how to create a project, see the corresponding section in Use APIs to Design and Consume Automated Test Data Services.

The following values are used in this example::

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **projectInfo:**


```
  {
```

```
"name": "Order",
"description": "This is Order Management project.",
"dateOrder": "YMD",
"id": 141357,
"inheritTables": true,
"timestampPrecision": 3,
"type": "DB",
"levels": [
  {
    "level": 1,
    "created": "",
    "updated": "",
    "hasData": "0",
    "keyOrder": "NAME",
    "name": "Data Group",
    "publish": "0"
  },
  {
    "level": 2,
    "created": "",
    "updated": "",
    "hasData": "0",
    "keyOrder": "NAME",
    "name": "Data Set",
    "publish": "0"
```

```
      },
      {
        "level": 3,
        "created": "",
        "updated": "",
        "hasData": "1",
        "keyOrder": "NAME",
        "name": "Data Pool",
        "publish": "1"
      }
    ],
    "created": null,
    "updated": null,
    "versions": [],
    "grantedFunctions": []
  }
```


The following response is returned. Note the project ID 141357 is created in this example.


```
{
  "name": "Order",
  "description": "This is Order Management project.",
  "dateOrder": "YMD",
  "id": 141357,
  "inheritTables": true,
  "timestampPrecision": 3,
```

```
"type": "DB",
"levels": [
  {
    "level": 1,
    "created": "",
    "updated": "",
    "hasData": "0",
    "keyOrder": "NAME",
    "name": "Data Group",
    "publish": "0"
  },
  {
    "level": 2,
    "created": "",
    "updated": "",
    "hasData": "0",
    "keyOrder": "NAME",
    "name": "Data Set",
    "publish": "0"
  },
  {
    "level": 3,
    "created": "",
    "updated": "",
    "hasData": "1",
```

```
      "keyOrder": "NAME",
      "name": "Data Pool",
      "publish": "1"
    }
  ],
  "created": null,
  "updated": null,
  "versions": [],
  "grantedFunctions": []
}
```

Create a Version

For detailed information about how to create a project, see the corresponding section in Use APIs to Design and Consume Automated Test Data Services.

The following values are used in this example::

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojLRQ5l4Ro
- **projectId:** 141357
- **versionInfo:**

```
  {
    "description": "This is Order Management version 1.0.",
    "name": "1.0",
    "projectName": "Order",
  }
```

The following response is returned. Note the version ID 141358 is created in this example.

```
{
   "id": 141358,
```

```
    "name": "1.0",
    "created": "2017-03-07T07:28:09+0000",
    "description": "This is Order Management version 1.0.",
    "projectName": Order,
    "levelDetails": null,
    "registeredObjectCount": 0,
    "tablesUsed": null,
    "isGeneric": false
  }
```

Share the Connection Profile

For detailed information about how to share a connection profile, see the corresponding section in Use APIs to Design and Consume Automated Test Data Services.

The following values are used in this example::

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **profileName:** Employee_SQLServer
- **groups:**

```
  {
      "adGroup": "TDE",
      "description": "Group includes TDEs",
      "groupName": "TDE",
      "isAdminGroup": true,
      "projectId": 141357,
      "securityFunctions": {}
    }
```

The following response is returned. Note the connection profile Employee_SQLServer is shared to the TDE group in this example.

```
{

 TDE

  }
```

Register Objects

For detailed information about how to register relational data, see Use APIs to Prepare Test Data for Non-Relational Sources.

The following values are used in this example:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **projectId:** 141357
- **versionId:** 141358
- **files:** employee.csv
  Below is the content of the CSV file used in this example:

```
 {
  "objectType": "TABLE",
  "schema": "dbo",
  "connectionProfileName": "Employee_SQLServer",

 "tableNames": [
 "ORDERS",
  "EMPLOYEES"]
 }
```

The following response is returned. Note that the objects with the ID 230275 and 230276 are registered in this example.

```
[
 {
 "objectId": 230275,
 "projectId": 141357,
 "versionId": 141358,
 "objectName": "Employees",
 "fileLocation": null,
 "objectType": "TABLE",
 "fileName": null,
 "filecount": null,
 "tableOwner": "dbo",
 "tableColumnCount": "18",
 "tableIndexCount": "2",
 "tableForeignKeyCount": "4",
```

```
"tableRegisteredDBMS": "sql server",
"tablePrimaryKeyIndex": "PK_Employees",
"tableOrder": null,
"parentId": null,
"fileStatus": 0,
"fileConnectionProfileName": null,
"fileEncoding": null,
"rootFilePath": null,
"jobFailureMessage": null,
"jobId": -1,
"programUpdated": "TDMApi",
"group": "TABLE",
"schemaLocation": null,
"noNamespaceSchemaLocation": null,
"explicitNamespaces": null,
"columns": [
{
"id": 869414,
"name": "Country",
"sequence": 12,
"dataType": "nvarchar",
"precision": 15,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869405,
"name": "FirstName",
"sequence": 3,
"dataType": "nvarchar",
"precision": 10,
"scale": 0,
"isNullable": "0",
"defaultValue": null
},
{
"id": 869409,
"name": "HireDate",
"sequence": 7,
"dataType": "datetime",
"precision": 23,
"scale": 3,
"isNullable": "1",
"defaultValue": null
},
```

```
{
"id": 869417,
"name": "Photo",
"sequence": 15,
"dataType": "image",
"precision": 0,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869407,
"name": "TitleOfCourtesy",
"sequence": 5,
"dataType": "nvarchar",
"precision": 25,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869418,
"name": "Notes",
"sequence": 16,
"dataType": "ntext",
"precision": 1073741823,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869415,
"name": "HomePhone",
"sequence": 13,
"dataType": "nvarchar",
"precision": 24,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869406,
"name": "Title",
"sequence": 4,
"dataType": "nvarchar",
"precision": 30,
```

```
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869413,
"name": "PostalCode",
"sequence": 11,
"dataType": "nvarchar",
"precision": 10,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869420,
"name": "PhotoPath",
"sequence": 18,
"dataType": "nvarchar",
"precision": 255,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869408,
"name": "BirthDate",
"sequence": 6,
"dataType": "datetime",
"precision": 23,
"scale": 3,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869403,
"name": "EmployeeID",
"sequence": 1,
"dataType": "int",
"precision": 10,
"scale": 0,
"isNullable": "2",
"defaultValue": null
},
{
"id": 869412,
```

```
"name": "Region",
"sequence": 10,
"dataType": "nvarchar",
"precision": 15,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869410,
"name": "Address",
"sequence": 8,
"dataType": "nvarchar",
"precision": 60,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869411,
"name": "City",
"sequence": 9,
"dataType": "nvarchar",
"precision": 15,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869416,
"name": "Extension",
"sequence": 14,
"dataType": "nvarchar",
"precision": 4,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869404,
"name": "LastName",
"sequence": 2,
"dataType": "nvarchar",
"precision": 20,
"scale": 0,
"isNullable": "0",
```

```
"defaultValue": null
},
{
"id": 869419,
"name": "ReportsTo",
"sequence": 17,
"dataType": "int",
"precision": 10,
"scale": 0,
"isNullable": "1",
"defaultValue": null
}
],
"foreignKeys": null,
"relationships": null,
"fileConnProfId": null
},
{
"objectId": 230276,
"projectId": 141357,
"versionId": 141358,
"objectName": "Orders",
"fileLocation": null,
"objectType": "TABLE",
"fileName": null,
"filecount": null,
"tableOwner": "dbo",
"tableColumnCount": "14",
"tableIndexCount": "8",
"tableForeignKeyCount": "4",
"tableRegisteredDBMS": "sql server",
"tablePrimaryKeyIndex": "PK_Orders",
"tableOrder": null,
"parentId": null,
"fileStatus": 0,
"fileConnectionProfileName": null,
"fileEncoding": null,
"rootFilePath": null,
"jobFailureMessage": null,
"jobId": -1,
"programUpdated": "TDMApi",
"group": "TABLE",
"schemaLocation": null,
"noNamespaceSchemaLocation": null,
"explicitNamespaces": null,
"columns": [
```

```
{
"id": 869451,
"name": "ShipPostalCode",
"sequence": 13,
"dataType": "nvarchar",
"precision": 10,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869445,
"name": "ShipVia",
"sequence": 7,
"dataType": "int",
"precision": 10,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869448,
"name": "ShipAddress",
"sequence": 10,
"dataType": "nvarchar",
"precision": 60,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869443,
"name": "RequiredDate",
"sequence": 5,
"dataType": "datetime",
"precision": 23,
"scale": 3,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869440,
"name": "CustomerID",
"sequence": 2,
"dataType": "nchar",
"precision": 5,
```

```
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869444,
"name": "ShippedDate",
"sequence": 6,
"dataType": "datetime",
"precision": 23,
"scale": 3,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869441,
"name": "EmployeeID",
"sequence": 3,
"dataType": "int",
"precision": 10,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869439,
"name": "OrderID",
"sequence": 1,
"dataType": "int",
"precision": 10,
"scale": 0,
"isNullable": "2",
"defaultValue": null
},
{
"id": 869442,
"name": "OrderDate",
"sequence": 4,
"dataType": "datetime",
"precision": 23,
"scale": 3,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869450,
```

```
"name": "ShipRegion",
"sequence": 12,
"dataType": "nvarchar",
"precision": 15,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869449,
"name": "ShipCity",
"sequence": 11,
"dataType": "nvarchar",
"precision": 15,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869452,
"name": "ShipCountry",
"sequence": 14,
"dataType": "nvarchar",
"precision": 15,
"scale": 0,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869446,
"name": "Freight",
"sequence": 8,
"dataType": "money",
"precision": 19,
"scale": 4,
"isNullable": "1",
"defaultValue": null
},
{
"id": 869447,
"name": "ShipName",
"sequence": 9,
"dataType": "nvarchar",
"precision": 40,
"scale": 0,
"isNullable": "1",
```

```
 "defaultValue": null
 }
 ],
 "foreignKeys": null,
 "relationships": null,
 "fileConnProfId": null
 }
]
```


Create Environment

For detailed information about how to create an environment, see the corresponding section in Use APIs to Design and Consume Automated Test Data Services.

The following values are used in this example:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojLRQ5l4Ro
- **projectId:** 141357
- **versionId:** 141358
- **environment:**


```
 {

 "datasourcesConnectionProfiles": [
 {
 "connectionProfileName": "SQL",
 "connectionProfileStatus": "EXISTS",
 "name": "SQL Server"
 }
 ],
 "description": "This is the staging environment.",
 "name": "Staging"
 }
```


The following response is returned. Note environment ID 1804 is created in this example.


```
{
 "numberOfElements": 1,
 "totalNumberOfElements": 1,
 "elements": [
 {
 "id": 1804,
 "name": "Staging",
 "description": "This is the staging environment.",
 "projectID": 141357,
```

```
  "versionID": 141358,
  "createdBy": "Administrator",
  "modifiedBy": "Administrator",
  "creationDate": "2017-03-30 11:17:29.085",
  "modifiedDate": "2017-03-30 11:17:29.085"
  }
  ]
}
```

Create a Data Model

For detailed information about how to create a project, see the corresponding section in Use APIs to Design and Consume Automated Test Data Services.

The following values are used in this example:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojLRQ5l4Ro
- **projectId:** 141357
- **versionId:** 141358
- **testDataModel:**

```
 {
  "description": "This test data model includes self-referencing association.",
  "modelKeys": [
  "EmployeeID"
  ],
  "name": "Employee_Datamodel",
  "root": {
  "displayName": "Employee Data Model",
  "rootEntity": {
  "dataSource": "SQL Server",
  "name": "Employees"
  },
  "visible": true
 }
 }
```

The following response is returned. Note data model with ID 1899 is created in this example. EmployeeID is used as the model key.

```
{

  "id": 1899,

  "name": "Employee_Datamodel",
```

```
  "description": "This test data model includes self-referencing association.",
  "visible": false,
  "modelKeys": [
    "EmployeeID"
  ],
  "root": {
    "displayName": "Employee Data Model",
    "rootEntity": {
      "id": 1900,
      "name": "Employees",
      "primaryKeys": [
        "EmployeeID"
      ],
      "dataSource": "SQL Server"
    }
  },
  "projectId": 141357,
  "versionId": 141358,
  "creationDate": "2017-03-30 19:47:49.619",
  "modifiedDate": "2017-03-30 19:47:49.619",
  "createdBy": "Administrator",
  "modifiedBy": "Administrator"
}
```

Define Association with Self-Reference

You can define the self reference at any level of field in an association. In this example, self reference is defined at root entity Employee level.

Employee entity has fields - Employee ID and Reports To. The field Employee ID includes the values of both the employee and manager to whom the employee is reporting to. As the manager's employee ID is available in both the fields Employee ID and Report To, the self reference association is formed for the field Employee ID.

The following values are used in this example:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojLRQ5l4Ro
- **projectId:** 141357
- **versionId:** 141358
- **testDataModelId:** 1899
- **association:**

```
{
  "associationType": "MANY_ONE",
  "joinFields": [
    {
      "fieldName": "EmployeeID",
      "referenceFieldName": "ReportsTo"
    }
  ],
  "name": "Manager",
  "sourceEntity": {
    "dataSource": "SQL Server",
    "name": "Employees"
  },
  "targetEntity": {
    "dataSource": "SQL Server",
    "name": "Employees"
  }
```

```
}
```

The following response is returned. Note the association with name Manager (ID-1901) is created with self reference.


```
{
  "id": 1901,
  "name": "Manager",
  "associationType": "MANY_ONE",
  "joinFields": [
    {
      "fieldName": "EmployeeID",
      "referenceFieldName": "ReportsTo"
    }
  ],
  "sourceEntity": {
    "id": 1900,
    "name": "Employees",
    "primaryKeys": [
      "EmployeeID"
    ],
    "dataSource": "SQL Server"
  },
  "targetEntity": {
    "id": 1900,
    "name": "Employees",
    "primaryKeys": [
```

```
        "EmployeeID"

      ],

      "dataSource": "SQL Server"

    }

}
```

Review the Associations Defined for the Data Model

After defining the associations, if you want to review the association details you can run the get association for a test data model to fetch the association details.

The following values are used in this example:

- **Authorization:** Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlU0VSX0lEBTExfUFJPSkVDVFNcIjpbMTAwXX0ifQ.7T1CyH_xQK0vQcBB7dLojURQ5l4Ro
- **projectId:** 141357
- **versionId:** 141358
- **testDataModelId:** 1899

The following response is returned with the information about available associations.

```
{

    "id": 1901,

    "name": "Manager",

    "associationType": "MANY_ONE",

    "joinFields": [

      {

        "fieldName": "EmployeeID",

        "referenceFieldName": "ReportsTo"

      }

    ],

    "sourceEntity": {

      "id": 1900,
```

```
      "name": "Employees",
      "primaryKeys": [
        "EmployeeID"
      ],
      "dataSource": "SQL Server"
    },
    "targetEntity": {
      "id": 1900,
      "name": "Employees",
      "primaryKeys": [
        "EmployeeID"
      ],
      "dataSource": "SQL Server"
    }
  }
```

Use APIs to Manage and Consume vTDM Clones

Test Data Engineers and Testers use vTDM either through the CA TDM Portal, or you write scripts using the vTDM REST API. For more information on the vTDM workflow, see Virtual Test Data Management (vTDM).

This page refers to the following API Services:

- TestDataManager
- TDMvDataService

Follow these steps:

Get a Security Token

You as Test Data Engineer or Tester need to authenticate and get a token to use in the subsequent APIs. In the following, *host* is the hostname of the CA TDM Portal, and *token* is your authentication token.

```
POST https://<server>:<host>/TestDataManager/user/login
```

- **HEADER**

Authorization	Basic YWRtaW5pc3RyYXRvcjptYXJtaXRl

- **RESPONSE**

```
{
"token": "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJBZG1pbmlzdHJhdG9yIiwiYXVkIjoiQUJlA2zKHI-
lHAiLd6VPn8P4",
"userName": "administrator",
"emailId": null,
"accessPermissions": [
{
"project": "ALL_PROJECTS",
"accessFunctions": [
"Admin"
]
}
],
"previewFeaturesEnabled": true,
"ldapAuthentication": false,
"targetProfile": null,
"sourceProfile": null,
"id": 1
}
```

Use the *token* from this response in the Authorization headers for all following API requests.

Register the Appliance (TDE)

You as Test Data Engineer can use the REST API to register the vTDM Appliance that houses the data.

Use the API access password for the `vtdmadmin` account. An administrator may have changed the password from the default value shown here.

```
POST https://<server>:<host>/TDMvDataService/api/ca/v1/appliances
```

- **HEADER**

Authorization	Bearer *token*
Content-Type	application/json

- **BODY**

```
{

"hostname" : "my.vtdm.host",
```

```
"password" : "vtdmadmin"

}
```

If the Appliance is not reachable, or the password is incorrect, the interface returns an appropriate error. For example:

- **RESPONSE**

```
{
"status": 401,
"errorCode": "",
"errorMsg": "Authentication failed. Bad username/password",
"errorDetail": "",
"timestamp": "2017-04-11T10:08+0000"
}
```

Manage Gold-copies (TDE)

You can use the vTDM REST API to manage Gold-copies from external automation scripts. Use the Rest API to automate creation of a Filesystem to copy your data into. You can then attach this Filesystem to your Windows machine hosting the SQL Server database.

Create a Filesystem

```
POST https://<server>:<host>/TDMvDataService/api/ca/v1/filesystems
```

- **HEADER**

Authorization	Bearer *token*
Content-Type	application/json

- **BODY**

```
{

"applianceHostname" : "my.vtdm.host",

"name" : "gold-copy",

"description" : "My SQL Gold-copy database"

}
```

Provide the hostname, Filesystem name, and a short description.**Format:** The **name** field can only contain the characters and numbers (a-z, A-Z, 0-9), do not use spaces or other special characters. This name forms part of the name used to connect the Filesystem to the Windows machine.

The Filesystem is ready for the data to be copied.

Checkpoint the Gold-copy Data

After you have copied the database to the Filesystem, you freeze the data in time using a 'checkpoint'. You use this checkpoint to make Clones of the data from that point in time. If you update to the Gold-copy data now, it does not affect the Clones, and you can continue to make identical Clones from that checkpoint.

For example, to create a checkpoint called 'initial' for the Filesystem called 'gold-copy', perform this operation:

```
POST https://<server>:<host>/TDMvDataService/api/ca/v1/checkpoints
```

- **HEADER**

Authorization	Bearer *token*
Content-Type	application/json

- **BODY**

```
{
"applianceHostname" : "my.vtdm.host",
"filesystem" : "gold-copy",
"checkpoint" : "initial"
"description" : "Gold-copy available for cloning!"
}
```

You have set up the appliance for the Tester to use.

Consume Gold-Copy Clones (Tester)

You as tester can use the API to automate creation of a Clone from the provided filesystem at a specific Checkpoint.

```
POST https://<server>:<host>/TDMvDataService/api/ca/v1/clones
```

- **HEADER**

Authorization	Bearer *token*
Content-Type	application/json

- **BODY**

```
{
"applianceHostname" : "my.vtdm.host",
"origin_filesystem" : "gold-copy",
```

```
"origin_checkpoint" : "initial",
"name" : "testing",
"description" : "Use this for testing"
}
```

A new share is now available under "*host*\database_gold-copy_testing ". Mount this share onto a new SQL Server host.

For more information, see vTDM Troubleshooting.

Use APIs to Create and Manage a Data Model

You as a Test Data Engineer, use a Data Model either through the CA TDM Portal, or you write scripts using the Data Model REST API. For more information on the Data Model workflow, see The Data Model in CA TDM Portal.

This page refers to the following API Services:

- TDMModelService

Follow these steps:

- Pre-scan an Environment
- Discover Entity Relationships in an Environment
- Retrieve All Entity Details in an Environment
- Retrieve Details for a Specific Entity
- Retrieve Data Discovery Scan Job Status
- Cancel a Data Discovery Scan
- Delete Data for a Failed Data Discovery Scan
- Delete Data Discovery Scan Details for a Deleted Environment
- Retrieve Entity Relationships for a Data Source
- Profile PII Data in a Data Model

NOTE

In this release, the term entity refers to a table and the term attribute refers to a Column.

NOTE

For all REST APIs with page and size parameters:

- Use page and size parameters to control enumeration of entities
- Default page size is 10
- Page numbers start at 0
- Divide size by total elements to determine the number of pages

Pre-scan an Environment

A Test Data Engineer can pre-scan an environment to collect entity definitions with no Data Model. Use this REST API to load all the entity definitions into your data source and use the Data Model APIs to enumerate those entity definitions in the CA TDM Portal.

```
POST https://<server>:<host>/TDMModelService/ca/v1/datamodel/preScan?
environmentId=163&projectId=7366&versionId=7367
```

- **RESPONSE:**

200 OK

- **RESPONSE BODY:**

```
{
  "jobId":765,
  "environmentId":163,
  "environmentName":"env_aaa",
  "projectId":7366,
  "projectVersionId":7367,
  "jobState":"CREATED",
  "startedBy":"Administrator",
  "jobName":"PRE_SCAN_7366_7367_163",
  "jobRunning":true
}
```

Discover Entity Relationships in an Environment

A Test Data Engineer can perform a Data Model scan on an environment to collect entity definitions along with entity relationships.

```
POST https://<server>:<host>/TDMModelService/ca/v1/datamodel/discoverRelationships?
environmentId=163&projectId=7366&versionId=7367
```

- **RESPONSE:**

200 OK

- **RESPONSE BODY:**

```
{
  "jobId":765,
  "environmentId":163,
  "environmentName":"env_aaa",
  "projectId":7366,
  "projectVersionId":7367,
  "jobState":"CREATED",
  "startedBy":"Administrator",
  "jobName":"PRE_SCAN_7366_7367_163",
  "jobRunning":true
}
```

Retrieve All Entity Details in an Environment

A Test Data Engineer can retrieve a list of all entities in an environment including attribute and entity relationship details.

```
GET https://<server>:<host>/TDMModelService/api/ca/v1/datamodel/entities/api/ca/v1/
datamodel/entities?
projectId=7366&versionId=7428&includeRelationships=true&includeRelatedEntities=true&includeAttr
&q=<query string>
```

Parameters:

- **projectId**Specifies the project ID.
- **projectVersionId**Specifies the project version.
- **includeRelationships**(Optional) Includes entity relationships for all entities.

Values: true or false (default)

- **includeRelatedEntities**(Optional) Includes all related entities.
  Values: true or false (default)
- **includeAttributes**(Optional) Includes all entity attributes.
  Values: true or false (default)
- **includeHierarchy**(Optional) Includes the fully qualified path of an entity in a hierarchical format.
  Values: true or false (default)
- **q**(Optional) Queries the entities to identify a specific entity name, attribute name, schema name, data source name, or database name. If no key field is specified, all entities, attributes, schema names, data sources, and databases are queried. The string value in this parameter is case insensitive.
  This parameter supports the basic wild card characters such as * (used to match one or more characters) and ? (used to match a single character). All supplied search terms are 'ANDed' together. For example, when you specify "q=attribute=address+database=travel" the response includes all entities that have either a travel database and an address attribute.
  You can perform a search on the relevant data sources based on one or more of the following key fields:
  - **entity=**
    Matches an entity name. For example, entity=cust* matches all entities starting with "cust" search term, such as CUSTOMER, custs, customers, and so on.
  - **attribute=**
    Matches an attribute name.
  - **schema=**
    Matches a schema name.
  - **datasource=**
    Matches a data source name.
  - **database=**
    Matches a database name.

  Example: The following REST API performs a search for entities that match 'customer', attributes that start with 'addr', datasource that matches 'northwnd', database that matches 'travel', and schema that matches 'dbo'.

```
GET https://<server>:<host>/TDMModelService/api/ca/v1/datamodel/entities?
projectId=7366&versionId=7367&includeRelationships=true&includeRelatedEntities=true&includeAt
+attribute=addr*+datasource=northwnd+databsase=travel+schema=dbo
```

Response:

200 OK

Response Body:

```
{
  "elements": [
    {
      "entityId": 50505,
      "entityName": "PEOPLE10",
      "dataSourceName": "piitest",
      "dataSourceType": "sql server",
      "databaseName": "piitest",
      "schemaName": "dbo",
      "entityOwner": "dbo",
      "fullyQualifiedPath": "piitest#piitest#dbo",
      "attributes": [
        {
```

```
  "attributeId": 1264524,
  "attributeName": "ID",
  "dataType": "numeric"
},
{
  "attributeId": 1264525,
  "attributeName": "DESIGNATION",
  "dataType": "varchar"
},
{
  "attributeId": 1264526,
  "attributeName": "FIRST_NAME",
  "dataType": "varchar"
},
{
  "attributeId": 1264527,
  "attributeName": "LAST_NAME",
  "dataType": "varchar"
},
{
  "attributeId": 1264528,
  "attributeName": "JOB_TITLE",
  "dataType": "varchar"
},
{
  "attributeId": 1264529,
  "attributeName": "LOB",
  "dataType": "varchar"
},
{
  "attributeId": 1264530,
  "attributeName": "EMAIL",
  "dataType": "varchar"
},
{
  "attributeId": 1264531,
  "attributeName": "CONTACT_PHONE",
  "dataType": "varchar"
},
{
  "attributeId": 1264532,
  "attributeName": "HOME_PHONE",
  "dataType": "varchar"
},
{
  "attributeId": 1264533,
```

```
    "attributeName": "MOBILE_PHONE",
    "dataType": "varchar"
  },
  {
    "attributeId": 1264534,
    "attributeName": "ADDRESS",
    "dataType": "varchar"
  },
  {
    "attributeId": 1264535,
    "attributeName": "START_DATE",
    "dataType": "datetime"
  },
  {
    "attributeId": 1264536,
    "attributeName": "TERMINATION_DATE",
    "dataType": "datetime"
  },
  {
    "attributeId": 1264537,
    "attributeName": "NATIONALITY_ID",
    "dataType": "numeric"
  },
  {
    "attributeId": 1264538,
    "attributeName": "RESIDENT_ID",
    "dataType": "numeric"
  },
  {
    "attributeId": 1264539,
    "attributeName": "COST_CENTRE",
    "dataType": "varchar"
  },
  {
    "attributeId": 1264540,
    "attributeName": "PHOTO_FILENAME",
    "dataType": "varchar"
  },
  {
    "attributeId": 1264541,
    "attributeName": "AUTHORISATION_ID",
    "dataType": "numeric"
  },
  {
    "attributeId": 1264542,
    "attributeName": "EMPNO",
```

```
      "dataType": "numeric"
    }
  ],
  "relatedEntities": [
    {
      "entityId": 50506,
      "entityName": "PEOPLE10A",
      "dataSourceName": "piitest",
      "dataSourceType": "sql server",
      "databaseName": "piitest",
      "schemaName": "dbo",
      "entityOwner": "dbo",
      "fullyQualifiedPath": "piitest#piitest#dbo",
      "attributes": [
        {
          "attributeId": 1264562,
          "attributeName": "ID",
          "dataType": "numeric"
        },
        {
          "attributeId": 1264563,
          "attributeName": "DESIGNATION",
          "dataType": "varchar"
        },
        {
          "attributeId": 1264564,
          "attributeName": "FIRST_NAME",
          "dataType": "varchar"
        },
        {
          "attributeId": 1264565,
          "attributeName": "LAST_NAME",
          "dataType": "varchar"
        },
        {
          "attributeId": 1264566,
          "attributeName": "JOB_TITLE",
          "dataType": "varchar"
        },
        {
          "attributeId": 1264567,
          "attributeName": "LOB",
          "dataType": "varchar"
        },
        {
          "attributeId": 1264568,
```

```
    "attributeName": "EMAIL",
    "dataType": "varchar"
  },
  {
    "attributeId": 1264569,
    "attributeName": "CONTACT_PHONE",
    "dataType": "varchar"
  },
  {
    "attributeId": 1264570,
    "attributeName": "HOME_PHONE",
    "dataType": "varchar"
  },
  {
    "attributeId": 1264571,
    "attributeName": "MOBILE_PHONE",
    "dataType": "varchar"
  },
  {
    "attributeId": 1264572,
    "attributeName": "ADDRESS",
    "dataType": "varchar"
  },
  {
    "attributeId": 1264573,
    "attributeName": "START_DATE",
    "dataType": "datetime"
  },
  {
    "attributeId": 1264574,
    "attributeName": "TERMINATION_DATE",
    "dataType": "datetime"
  },
  {
    "attributeId": 1264575,
    "attributeName": "NATIONALITY_ID",
    "dataType": "numeric"
  },
  {
    "attributeId": 1264576,
    "attributeName": "RESIDENT_ID",
    "dataType": "numeric"
  },
  {
    "attributeId": 1264577,
    "attributeName": "COST_CENTRE",
```

```
          "dataType": "varchar"
        },
        {
          "attributeId": 1264578,
          "attributeName": "PHOTO_FILENAME",
          "dataType": "varchar"
        },
        {
          "attributeId": 1264579,
          "attributeName": "AUTHORISATION_ID",
          "dataType": "numeric"
        },
        {
          "attributeId": 1264580,
          "attributeName": "EMPNO",
          "dataType": "numeric"
        }
      ]
    }
  ],
  "relationshipDetails": [
    {
      "id": 455919,
      "parentEntityId": 50505,
      "parentEntityName": "PEOPLE10",
      "childEntityId": 50506,
      "childEntityName": "PEOPLE10A",
      "relationshipAttributes": [
        {
          "parentAttributeId": 1264524,
          "parentAttributeName": "ID",
          "childAttributeId": 1264562,
          "childAttributeName": "ID",
          "sequence": 1
        }
      ],
      "relationshipMatcher": "attribute name"
    },
    {
      "id": 455920,
      "parentEntityId": 50505,
      "parentEntityName": "PEOPLE10",
      "childEntityId": 50506,
      "childEntityName": "PEOPLE10A",
      "relationshipAttributes": [
        {
```

```
        "parentAttributeId": 1264525,
        "parentAttributeName": "DESIGNATION",
        "childAttributeId": 1264563,
        "childAttributeName": "DESIGNATION",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455921,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264526,
        "parentAttributeName": "FIRST_NAME",
        "childAttributeId": 1264564,
        "childAttributeName": "FIRST_NAME",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455922,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264527,
        "parentAttributeName": "LAST_NAME",
        "childAttributeId": 1264565,
        "childAttributeName": "LAST_NAME",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455923,
    "parentEntityId": 50505,
```

```
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264528,
        "parentAttributeName": "JOB_TITLE",
        "childAttributeId": 1264566,
        "childAttributeName": "JOB_TITLE",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455924,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264529,
        "parentAttributeName": "LOB",
        "childAttributeId": 1264567,
        "childAttributeName": "LOB",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455925,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264530,
        "parentAttributeName": "EMAIL",
        "childAttributeId": 1264568,
        "childAttributeName": "EMAIL",
        "sequence": 1
      }
    ],
```

```
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455926,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264531,
        "parentAttributeName": "CONTACT_PHONE",
        "childAttributeId": 1264569,
        "childAttributeName": "CONTACT_PHONE",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455927,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264532,
        "parentAttributeName": "HOME_PHONE",
        "childAttributeId": 1264570,
        "childAttributeName": "HOME_PHONE",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455928,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264533,
        "parentAttributeName": "MOBILE_PHONE",
```

```
        "childAttributeId": 1264571,
        "childAttributeName": "MOBILE_PHONE",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455929,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264534,
        "parentAttributeName": "ADDRESS",
        "childAttributeId": 1264572,
        "childAttributeName": "ADDRESS",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455930,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264539,
        "parentAttributeName": "COST_CENTRE",
        "childAttributeId": 1264577,
        "childAttributeName": "COST_CENTRE",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455931,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
```

```
      "childEntityName": "PEOPLE10A",
      "relationshipAttributes": [
        {
          "parentAttributeId": 1264540,
          "parentAttributeName": "PHOTO_FILENAME",
          "childAttributeId": 1264578,
          "childAttributeName": "PHOTO_FILENAME",
          "sequence": 1
        }
      ],
      "relationshipMatcher": "attribute name"
    },
    {
      "id": 455932,
      "parentEntityId": 50505,
      "parentEntityName": "PEOPLE10",
      "childEntityId": 50506,
      "childEntityName": "PEOPLE10A",
      "relationshipAttributes": [
        {
          "parentAttributeId": 1264542,
          "parentAttributeName": "EMPNO",
          "childAttributeId": 1264580,
          "childAttributeName": "EMPNO",
          "sequence": 1
        }
      ],
      "relationshipMatcher": "attribute name"
    }
  ]
},
{
  "entityId": 50506,
  "entityName": "PEOPLE10A",
  "dataSourceName": "piitest",
  "dataSourceType": "sql server",
  "databaseName": "piitest",
  "schemaName": "dbo",
  "entityOwner": "dbo",
  "fullyQualifiedPath": "piitest#piitest#dbo",
  "attributes": [
    {
      "attributeId": 1264562,
      "attributeName": "ID",
      "dataType": "numeric"
    },
```

```
{
  "attributeId": 1264563,
  "attributeName": "DESIGNATION",
  "dataType": "varchar"
},
{
  "attributeId": 1264564,
  "attributeName": "FIRST_NAME",
  "dataType": "varchar"
},
{
  "attributeId": 1264565,
  "attributeName": "LAST_NAME",
  "dataType": "varchar"
},
{
  "attributeId": 1264566,
  "attributeName": "JOB_TITLE",
  "dataType": "varchar"
},
{
  "attributeId": 1264567,
  "attributeName": "LOB",
  "dataType": "varchar"
},
{
  "attributeId": 1264568,
  "attributeName": "EMAIL",
  "dataType": "varchar"
},
{
  "attributeId": 1264569,
  "attributeName": "CONTACT_PHONE",
  "dataType": "varchar"
},
{
  "attributeId": 1264570,
  "attributeName": "HOME_PHONE",
  "dataType": "varchar"
},
{
  "attributeId": 1264571,
  "attributeName": "MOBILE_PHONE",
  "dataType": "varchar"
},
{
```

```
      "attributeId": 1264572,
      "attributeName": "ADDRESS",
      "dataType": "varchar"
    },
    {
      "attributeId": 1264573,
      "attributeName": "START_DATE",
      "dataType": "datetime"
    },
    {
      "attributeId": 1264574,
      "attributeName": "TERMINATION_DATE",
      "dataType": "datetime"
    },
    {
      "attributeId": 1264575,
      "attributeName": "NATIONALITY_ID",
      "dataType": "numeric"
    },
    {
      "attributeId": 1264576,
      "attributeName": "RESIDENT_ID",
      "dataType": "numeric"
    },
    {
      "attributeId": 1264577,
      "attributeName": "COST_CENTRE",
      "dataType": "varchar"
    },
    {
      "attributeId": 1264578,
      "attributeName": "PHOTO_FILENAME",
      "dataType": "varchar"
    },
    {
      "attributeId": 1264579,
      "attributeName": "AUTHORISATION_ID",
      "dataType": "numeric"
    },
    {
      "attributeId": 1264580,
      "attributeName": "EMPNO",
      "dataType": "numeric"
    }
  ],
  "relatedEntities": [
```

```
{
  "entityId": 50505,
  "entityName": "PEOPLE10",
  "dataSourceName": "piitest",
  "dataSourceType": "sql server",
  "databaseName": "piitest",
  "schemaName": "dbo",
  "entityOwner": "dbo",
  "fullyQualifiedPath": "piitest#piitest#dbo",
  "attributes": [
    {
      "attributeId": 1264524,
      "attributeName": "ID",
      "dataType": "numeric"
    },
    {
      "attributeId": 1264525,
      "attributeName": "DESIGNATION",
      "dataType": "varchar"
    },
    {
      "attributeId": 1264526,
      "attributeName": "FIRST_NAME",
      "dataType": "varchar"
    },
    {
      "attributeId": 1264527,
      "attributeName": "LAST_NAME",
      "dataType": "varchar"
    },
    {
      "attributeId": 1264528,
      "attributeName": "JOB_TITLE",
      "dataType": "varchar"
    },
    {
      "attributeId": 1264529,
      "attributeName": "LOB",
      "dataType": "varchar"
    },
    {
      "attributeId": 1264530,
      "attributeName": "EMAIL",
      "dataType": "varchar"
    },
    {
```

```
    "attributeId": 1264531,
    "attributeName": "CONTACT_PHONE",
    "dataType": "varchar"
  },
  {
    "attributeId": 1264532,
    "attributeName": "HOME_PHONE",
    "dataType": "varchar"
  },
  {
    "attributeId": 1264533,
    "attributeName": "MOBILE_PHONE",
    "dataType": "varchar"
  },
  {
    "attributeId": 1264534,
    "attributeName": "ADDRESS",
    "dataType": "varchar"
  },
  {
    "attributeId": 1264535,
    "attributeName": "START_DATE",
    "dataType": "datetime"
  },
  {
    "attributeId": 1264536,
    "attributeName": "TERMINATION_DATE",
    "dataType": "datetime"
  },
  {
    "attributeId": 1264537,
    "attributeName": "NATIONALITY_ID",
    "dataType": "numeric"
  },
  {
    "attributeId": 1264538,
    "attributeName": "RESIDENT_ID",
    "dataType": "numeric"
  },
  {
    "attributeId": 1264539,
    "attributeName": "COST_CENTRE",
    "dataType": "varchar"
  },
  {
    "attributeId": 1264540,
```

```
          "attributeName": "PHOTO_FILENAME",
          "dataType": "varchar"
        },
        {
          "attributeId": 1264541,
          "attributeName": "AUTHORISATION_ID",
          "dataType": "numeric"
        },
        {
          "attributeId": 1264542,
          "attributeName": "EMPNO",
          "dataType": "numeric"
        }
      ]
    }
  ],
  "relationshipDetails": [
    {
      "id": 455919,
      "parentEntityId": 50505,
      "parentEntityName": "PEOPLE10",
      "childEntityId": 50506,
      "childEntityName": "PEOPLE10A",
      "relationshipAttributes": [
        {
          "parentAttributeId": 1264524,
          "parentAttributeName": "ID",
          "childAttributeId": 1264562,
          "childAttributeName": "ID",
          "sequence": 1
        }
      ],
      "relationshipMatcher": "attribute name"
    },
    {
      "id": 455920,
      "parentEntityId": 50505,
      "parentEntityName": "PEOPLE10",
      "childEntityId": 50506,
      "childEntityName": "PEOPLE10A",
      "relationshipAttributes": [
        {
          "parentAttributeId": 1264525,
          "parentAttributeName": "DESIGNATION",
          "childAttributeId": 1264563,
          "childAttributeName": "DESIGNATION",
```

```
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455921,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264526,
        "parentAttributeName": "FIRST_NAME",
        "childAttributeId": 1264564,
        "childAttributeName": "FIRST_NAME",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455922,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264527,
        "parentAttributeName": "LAST_NAME",
        "childAttributeId": 1264565,
        "childAttributeName": "LAST_NAME",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455923,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
```

```
      {
        "parentAttributeId": 1264528,
        "parentAttributeName": "JOB_TITLE",
        "childAttributeId": 1264566,
        "childAttributeName": "JOB_TITLE",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455924,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264529,
        "parentAttributeName": "LOB",
        "childAttributeId": 1264567,
        "childAttributeName": "LOB",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455925,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264530,
        "parentAttributeName": "EMAIL",
        "childAttributeId": 1264568,
        "childAttributeName": "EMAIL",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455926,
```

```
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264531,
        "parentAttributeName": "CONTACT_PHONE",
        "childAttributeId": 1264569,
        "childAttributeName": "CONTACT_PHONE",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455927,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264532,
        "parentAttributeName": "HOME_PHONE",
        "childAttributeId": 1264570,
        "childAttributeName": "HOME_PHONE",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455928,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264533,
        "parentAttributeName": "MOBILE_PHONE",
        "childAttributeId": 1264571,
        "childAttributeName": "MOBILE_PHONE",
        "sequence": 1
      }
```

```
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455929,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264534,
        "parentAttributeName": "ADDRESS",
        "childAttributeId": 1264572,
        "childAttributeName": "ADDRESS",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455930,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264539,
        "parentAttributeName": "COST_CENTRE",
        "childAttributeId": 1264577,
        "childAttributeName": "COST_CENTRE",
        "sequence": 1
      }
    ],
    "relationshipMatcher": "attribute name"
  },
  {
    "id": 455931,
    "parentEntityId": 50505,
    "parentEntityName": "PEOPLE10",
    "childEntityId": 50506,
    "childEntityName": "PEOPLE10A",
    "relationshipAttributes": [
      {
        "parentAttributeId": 1264540,
```

```
            "parentAttributeName": "PHOTO_FILENAME",
            "childAttributeId": 1264578,
            "childAttributeName": "PHOTO_FILENAME",
            "sequence": 1
          }
        ],
        "relationshipMatcher": "attribute name"
      },
      {
        "id": 455932,
        "parentEntityId": 50505,
        "parentEntityName": "PEOPLE10",
        "childEntityId": 50506,
        "childEntityName": "PEOPLE10A",
        "relationshipAttributes": [
          {
            "parentAttributeId": 1264542,
            "parentAttributeName": "EMPNO",
            "childAttributeId": 1264580,
            "childAttributeName": "EMPNO",
            "sequence": 1
          }
        ],
        "relationshipMatcher": "attribute name"
      }
    ]
  },
  {
    "entityId": 50507,
    "entityName": "trace_xe_action_map",
    "dataSourceName": "piitest",
    "dataSourceType": "sql server",
    "databaseName": "piitest",
    "schemaName": "sys",
    "entityOwner": "sys",
    "fullyQualifiedPath": "piitest#piitest#sys",
    "attributes": [
      {
        "attributeId": 1264584,
        "attributeName": "trace_column_id",
        "dataType": "smallint"
      },
      {
        "attributeId": 1264585,
        "attributeName": "package_name",
        "dataType": "nvarchar"
```

```
    },
    {
      "attributeId": 1264586,
      "attributeName": "xe_action_name",
      "dataType": "nvarchar"
    }
  ],
  "relatedEntities": [
    {
      "entityId": 50508,
      "entityName": "trace_xe_event_map",
      "dataSourceName": "piitest",
      "dataSourceType": "sql server",
      "databaseName": "piitest",
      "schemaName": "sys",
      "entityOwner": "sys",
      "fullyQualifiedPath": "piitest#piitest#sys",
      "attributes": [
        {
          "attributeId": 1264590,
          "attributeName": "trace_event_id",
          "dataType": "smallint"
        },
        {
          "attributeId": 1264591,
          "attributeName": "package_name",
          "dataType": "nvarchar"
        },
        {
          "attributeId": 1264592,
          "attributeName": "xe_event_name",
          "dataType": "nvarchar"
        }
      ]
    }
  ],
  "relationshipDetails": [
    {
      "id": 455933,
      "parentEntityId": 50507,
      "parentEntityName": "trace_xe_action_map",
      "childEntityId": 50508,
      "childEntityName": "trace_xe_event_map",
      "relationshipAttributes": [
        {
          "parentAttributeId": 1264585,
```

```
            "parentAttributeName": "package_name",
            "childAttributeId": 1264591,
            "childAttributeName": "package_name",
            "sequence": 1
          }
        ],
        "relationshipMatcher": "attribute name"
      }
    ]
  },
  {
    "entityId": 50508,
    "entityName": "trace_xe_event_map",
    "dataSourceName": "piitest",
    "dataSourceType": "sql server",
    "databaseName": "piitest",
    "schemaName": "sys",
    "entityOwner": "sys",
    "fullyQualifiedPath": "piitest#piitest#sys",
    "attributes": [
      {
        "attributeId": 1264590,
        "attributeName": "trace_event_id",
        "dataType": "smallint"
      },
      {
        "attributeId": 1264591,
        "attributeName": "package_name",
        "dataType": "nvarchar"
      },
      {
        "attributeId": 1264592,
        "attributeName": "xe_event_name",
        "dataType": "nvarchar"
      }
    ],
    "relatedEntities": [
      {
        "entityId": 50507,
        "entityName": "trace_xe_action_map",
        "dataSourceName": "piitest",
        "dataSourceType": "sql server",
        "databaseName": "piitest",
        "schemaName": "sys",
        "entityOwner": "sys",
        "fullyQualifiedPath": "piitest#piitest#sys",
```

```
          "attributes": [
            {
              "attributeId": 1264584,
              "attributeName": "trace_column_id",
              "dataType": "smallint"
            },
            {
              "attributeId": 1264585,
              "attributeName": "package_name",
              "dataType": "nvarchar"
            },
            {
              "attributeId": 1264586,
              "attributeName": "xe_action_name",
              "dataType": "nvarchar"
            }
          ]
        }
      ],
      "relationshipDetails": [
        {
          "id": 455933,
          "parentEntityId": 50507,
          "parentEntityName": "trace_xe_action_map",
          "childEntityId": 50508,
          "childEntityName": "trace_xe_event_map",
          "relationshipAttributes": [
            {
              "parentAttributeId": 1264585,
              "parentAttributeName": "package_name",
              "childAttributeId": 1264591,
              "childAttributeName": "package_name",
              "sequence": 1
            }
          ],
          "relationshipMatcher": "attribute name"
        }
      ]
    }
  ],
  "numberOfElements": 4,
  "totalElements": 4
}
```

Retrieve Details for a Specific Entity

A Test Data Engineer can retrieve details for a specific entity in an environment. This REST API includes all attribute and entity relationship details for the specific entity.

```
GET https://<server>:<host>/TDMModelService/api/ca/v1/datamodel/entities/{entityId}?
projectId=7366&versionId=7428&includeRelationships=true&includeRelatedEntities=true&includeAttr
```

Parameters:

- **projectId**Specifies the project ID.
- **projectVersionId**Specifies the project version.
- **entityId**Specifies the entity ID.
- **includeRelationships**(Optional) Includes all entity relationships for a specific entity.
  Values: true or false (default)
- **includeRelatedEntities**(Optional) Includes all related entities.
  Values: true or false (default)
- **includeAttributes**(Optional) Includes all entity attributes.
  Values: true or false (default)
- **includeHierarchy**(Optional) Includes the fully qualified path of an entity in a hierarchical format.
  Values: true or false (default)

Response:

200 OK

Response Body:

The following response code includes all entity relationships for the specific entity, all related entities, and all related entity attributes.

```
{
 "entityId": 50505,
 "entityName": "PEOPLE10",
 "dataSourceName": "piitest",
 "dataSourceType": "sql server",
 "databaseName": "piitest",
 "schemaName": "dbo",
 "entityOwner": "dbo",
 "fullyQualifiedPath": "piitest#piitest#dbo",
 "attributes": [
 {
 "attributeId": 1264524,
 "attributeName": "ID",
 "dataType": "numeric"
 },
 {
 "attributeId": 1264525,
 "attributeName": "DESIGNATION",
 "dataType": "varchar"
 },
 {
 "attributeId": 1264526,
```

```
"attributeName": "FIRST_NAME",
"dataType": "varchar"
},
{
"attributeId": 1264527,
"attributeName": "LAST_NAME",
"dataType": "varchar"
},
{
"attributeId": 1264528,
"attributeName": "JOB_TITLE",
"dataType": "varchar"
},
{
"attributeId": 1264529,
"attributeName": "LOB",
"dataType": "varchar"
},
{
"attributeId": 1264530,
"attributeName": "EMAIL",
"dataType": "varchar"
},
{
"attributeId": 1264531,
"attributeName": "CONTACT_PHONE",
"dataType": "varchar"
},
{
"attributeId": 1264532,
"attributeName": "HOME_PHONE",
"dataType": "varchar"
},
{
"attributeId": 1264533,
"attributeName": "MOBILE_PHONE",
"dataType": "varchar"
},
{
"attributeId": 1264534,
"attributeName": "ADDRESS",
"dataType": "varchar"
},
{
"attributeId": 1264535,
"attributeName": "START_DATE",
```

```
"dataType": "datetime"
},
{
"attributeId": 1264536,
"attributeName": "TERMINATION_DATE",
"dataType": "datetime"
},
{
"attributeId": 1264537,
"attributeName": "NATIONALITY_ID",
"dataType": "numeric"
},
{
"attributeId": 1264538,
"attributeName": "RESIDENT_ID",
"dataType": "numeric"
},
{
"attributeId": 1264539,
"attributeName": "COST_CENTRE",
"dataType": "varchar"
},
{
"attributeId": 1264540,
"attributeName": "PHOTO_FILENAME",
"dataType": "varchar"
},
{
"attributeId": 1264541,
"attributeName": "AUTHORISATION_ID",
"dataType": "numeric"
},
{
"attributeId": 1264542,
"attributeName": "EMPNO",
"dataType": "numeric"
}
],
"relatedEntities": [
{
"entityId": 50506,
"entityName": "PEOPLE10A",
"dataSourceName": "piitest",
"dataSourceType": "sql server",
"databaseName": "piitest",
"schemaName": "dbo",
```

```
"entityOwner": "dbo",
"fullyQualifiedPath": "piitest#piitest#dbo",
"attributes": [
{
"attributeId": 1264562,
"attributeName": "ID",
"dataType": "numeric"
},
{
"attributeId": 1264563,
"attributeName": "DESIGNATION",
"dataType": "varchar"
},
{
"attributeId": 1264564,
"attributeName": "FIRST_NAME",
"dataType": "varchar"
},
{
"attributeId": 1264565,
"attributeName": "LAST_NAME",
"dataType": "varchar"
},
{
"attributeId": 1264566,
"attributeName": "JOB_TITLE",
"dataType": "varchar"
},
{
"attributeId": 1264567,
"attributeName": "LOB",
"dataType": "varchar"
},
{
"attributeId": 1264568,
"attributeName": "EMAIL",
"dataType": "varchar"
},
{
"attributeId": 1264569,
"attributeName": "CONTACT_PHONE",
"dataType": "varchar"
},
{
"attributeId": 1264570,
"attributeName": "HOME_PHONE",
```

```
"dataType": "varchar"
},
{
"attributeId": 1264571,
"attributeName": "MOBILE_PHONE",
"dataType": "varchar"
},
{
"attributeId": 1264572,
"attributeName": "ADDRESS",
"dataType": "varchar"
},
{
"attributeId": 1264573,
"attributeName": "START_DATE",
"dataType": "datetime"
},
{
"attributeId": 1264574,
"attributeName": "TERMINATION_DATE",
"dataType": "datetime"
},
{
"attributeId": 1264575,
"attributeName": "NATIONALITY_ID",
"dataType": "numeric"
},
{
"attributeId": 1264576,
"attributeName": "RESIDENT_ID",
"dataType": "numeric"
},
{
"attributeId": 1264577,
"attributeName": "COST_CENTRE",
"dataType": "varchar"
},
{
"attributeId": 1264578,
"attributeName": "PHOTO_FILENAME",
"dataType": "varchar"
},
{
"attributeId": 1264579,
"attributeName": "AUTHORISATION_ID",
"dataType": "numeric"
```

```
},
{
"attributeId": 1264580,
"attributeName": "EMPNO",
"dataType": "numeric"
}
]
}
],
"relationshipDetails": [
{
"id": 455919,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264524,
"parentAttributeName": "ID",
"childAttributeId": 1264562,
"childAttributeName": "ID",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455920,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264525,
"parentAttributeName": "DESIGNATION",
"childAttributeId": 1264563,
"childAttributeName": "DESIGNATION",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455921,
```

```
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264526,
"parentAttributeName": "FIRST_NAME",
"childAttributeId": 1264564,
"childAttributeName": "FIRST_NAME",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455922,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264527,
"parentAttributeName": "LAST_NAME",
"childAttributeId": 1264565,
"childAttributeName": "LAST_NAME",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455923,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264528,
"parentAttributeName": "JOB_TITLE",
"childAttributeId": 1264566,
"childAttributeName": "JOB_TITLE",
"sequence": 1
}
```

```
],
"relationshipMatcher": "attribute name"
},
{
"id": 455924,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264529,
"parentAttributeName": "LOB",
"childAttributeId": 1264567,
"childAttributeName": "LOB",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455925,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264530,
"parentAttributeName": "EMAIL",
"childAttributeId": 1264568,
"childAttributeName": "EMAIL",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455926,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264531,
```

```
"parentAttributeName": "CONTACT_PHONE",
"childAttributeId": 1264569,
"childAttributeName": "CONTACT_PHONE",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455927,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264532,
"parentAttributeName": "HOME_PHONE",
"childAttributeId": 1264570,
"childAttributeName": "HOME_PHONE",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455928,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264533,
"parentAttributeName": "MOBILE_PHONE",
"childAttributeId": 1264571,
"childAttributeName": "MOBILE_PHONE",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455929,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
```

```
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264534,
"parentAttributeName": "ADDRESS",
"childAttributeId": 1264572,
"childAttributeName": "ADDRESS",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455930,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264539,
"parentAttributeName": "COST_CENTRE",
"childAttributeId": 1264577,
"childAttributeName": "COST_CENTRE",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455931,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264540,
"parentAttributeName": "PHOTO_FILENAME",
"childAttributeId": 1264578,
"childAttributeName": "PHOTO_FILENAME",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
```

```
  },
  {
  "id": 455932,
  "parentEntityId": 50505,
  "parentEntityName": "PEOPLE10",
  "childEntityId": 50506,
  "childEntityName": "PEOPLE10A",
  "relationshipAttributes": [
  {
  "parentAttributeId": 1264542,
  "parentAttributeName": "EMPNO",
  "childAttributeId": 1264580,
  "childAttributeName": "EMPNO",
  "sequence": 1
  }
  ],
  "relationshipMatcher": "attribute name"
  }
  ]
}
```

Retrieve Data Model Scan Job Status

After performing a Data Model scan, a Test Data Engineer can retrieve information related to the Data Model scan job. The "jobState" parameter indicates the status of a Data Model scan.

```
GET https://<server>:<host>/TDMModelService/api/ca/v1/datamodel/information?
projectId=7366&versionId=7367
```

- **RESPONSE**200 OK
- **RESPONSE BODY**

```
[
  {
    "jobId": 5502,
    "environmentId": 1001,
    "environmentName": "ev1",
    "projectId": 7366,
    "projectVersionId":
 7367,
    "jobState": "COMPLETED",
    "jobStatus": "",
    "startedBy": "Administrator",
    "jobName": "DISCOVER_7366_7367_1001",
    "jobRunning": false
  }
]
```

Cancel a Data Model Scan

A Test Data Engineer can cancel a running Data Model scan and delete all discovery related data from your data source. For example, if a Data Model scan takes a long time to complete, use the following REST API to cancel the Data Model scan:

```
POST https://<server>:<host>/TDMModelService/api/ca/v1/datamodel/cancel?
environmentId=163&projectId=7366&versionId=7367
```

- **RESPONSE:**
  200 OK
- **RESPONSE BODY:**

```
true
```

Delete Data for a Failed Data Model Scan

For the most recent failed Data Model scan, a Test Data Engineer can delete all discovery related data from the data source. For example, if an environment fails to connect to a data source that fails the Data Model scan. Use the following REST API to delete the failed Data Model scan and all discovery related data:

```
POST https://<server>:<host>/TDMModelService/api/ca/v1/datamodel/confirm?
projectId=7366&versionId=7367
```

- **RESPONSE:**
  200 OK
- **RESPONSE BODY:**

```
true
```

Delete Data Model Scan Details for a Deleted Environment

If an environment that includes a Data Model scan is deleted, a Test Data Engineer can delete all discovery related data for that specific environment.

```
DELETE https://<server>:<host>/TDMModelService/api/ca/v1/datamodel/environments/163
```

- **RESPONSE:**
  200 OK
- **RESPONSE BODY:**

```
true
```

Retrieve Entity Relationships for a Data Source

A Test Data Engineer can retrieve entity relationships for the selected Data Source. Retrieving entity relationships for a Data Source provides an understanding on how different entities are linked. Two entities which are related (linked entities), indicate two database entities, where a relationship exists. It is common for an attribute in one entity to reference an attribute in another entity. For example, an entity named Person includes an individual's details in attributes such as first name, last name, and person ID. This person ID attribute can be referenced in one or more entities.

To retrieve entity relationships in a Data Source use the following REST API:

```
GET https://<server>:<host>/TDMModelService/api/ca/v1/datamodel/relationships?
projectId=7528&versionId=7529&page=0&size=20&q=<query string>
```

Parameters:

- **projectId**

Specifies the project ID.

- **versionId**
  Specifies the project version.
- **q**
  (Optional) Queries the entities to identify values for a specific parameter returned in this APIs response code. For example, you can filter results based on parentEntityID, parentEntityName, parentAttributeName, parentAttributeName, childAttributeName, relationshipName, and so on. The string value for the object field is case-sensitive. The query value depends on the data-type being searched. **This parameter supports the basic wild card characters such as * (used to match one or more characters) and ? (used to match a single character). All supplied search terms can be ANDed with '+' or ORed with '|'.**
  Examples:
  - The following API searches for all relationships where "parentEntityName=Employees" or "childEntityName=Employees"

    ```
    GET https://<server>:<host>/TDMModelService/api/ca/v1/datamodel/relationships?
    projectId=7528&versionId=7529&q=(parentEntityName=Employees)|(childEntityName=Employees)
    ```
  - The following API searches for all relationships where the "relationshipName=1_2346_2347_604"

    ```
    GET https://<server>:<host>/TDMModelService/api/ca/v1/datamodel/relationships?
    projectId=7528&versionId=7529&q=(relationshipName=1_2346_2347_604*)
    ```

RESPONSE: OK

RESPONSE BODY:

Note: The following example code is only a subset of the original Response Body.


```
{
 "elements": [
 {
 "id": 455919,
 "parentEntityId": 50505,
 "parentEntityName": "PEOPLE10",
 "childEntityId": 50506,
 "childEntityName": "PEOPLE10A",
 "relationshipAttributes": [
 {
 "parentAttributeId": 1264524,
 "parentAttributeName": "ID",
 "childAttributeId": 1264562,
 "childAttributeName": "ID",
 "sequence": 1
 }
 ],
 "relationshipMatcher": "attribute name"
 },
 {
 "id": 455920,
 "parentEntityId": 50505,
 "parentEntityName": "PEOPLE10",
 "childEntityId": 50506,
 "childEntityName": "PEOPLE10A",
```

```
"relationshipAttributes": [
{
"parentAttributeId": 1264525,
"parentAttributeName": "DESIGNATION",
"childAttributeId": 1264563,
"childAttributeName": "DESIGNATION",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455921,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264526,
"parentAttributeName": "FIRST_NAME",
"childAttributeId": 1264564,
"childAttributeName": "FIRST_NAME",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455922,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264527,
"parentAttributeName": "LAST_NAME",
"childAttributeId": 1264565,
"childAttributeName": "LAST_NAME",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
```

```
"id": 455923,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264528,
"parentAttributeName": "JOB_TITLE",
"childAttributeId": 1264566,
"childAttributeName": "JOB_TITLE",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455924,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264529,
"parentAttributeName": "LOB",
"childAttributeId": 1264567,
"childAttributeName": "LOB",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455925,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264530,
"parentAttributeName": "EMAIL",
"childAttributeId": 1264568,
"childAttributeName": "EMAIL",
"sequence": 1
```

```
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455926,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264531,
"parentAttributeName": "CONTACT_PHONE",
"childAttributeId": 1264569,
"childAttributeName": "CONTACT_PHONE",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455927,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264532,
"parentAttributeName": "HOME_PHONE",
"childAttributeId": 1264570,
"childAttributeName": "HOME_PHONE",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455928,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
```

```
"parentAttributeId": 1264533,
"parentAttributeName": "MOBILE_PHONE",
"childAttributeId": 1264571,
"childAttributeName": "MOBILE_PHONE",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455929,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264534,
"parentAttributeName": "ADDRESS",
"childAttributeId": 1264572,
"childAttributeName": "ADDRESS",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455930,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264539,
"parentAttributeName": "COST_CENTRE",
"childAttributeId": 1264577,
"childAttributeName": "COST_CENTRE",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455931,
"parentEntityId": 50505,
```

```
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264540,
"parentAttributeName": "PHOTO_FILENAME",
"childAttributeId": 1264578,
"childAttributeName": "PHOTO_FILENAME",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455932,
"parentEntityId": 50505,
"parentEntityName": "PEOPLE10",
"childEntityId": 50506,
"childEntityName": "PEOPLE10A",
"relationshipAttributes": [
{
"parentAttributeId": 1264542,
"parentAttributeName": "EMPNO",
"childAttributeId": 1264580,
"childAttributeName": "EMPNO",
"sequence": 1
}
],
"relationshipMatcher": "attribute name"
},
{
"id": 455933,
"parentEntityId": 50507,
"parentEntityName": "trace_xe_action_map",
"childEntityId": 50508,
"childEntityName": "trace_xe_event_map",
"relationshipAttributes": [
{
"parentAttributeId": 1264585,
"parentAttributeName": "package_name",
"childAttributeId": 1264591,
"childAttributeName": "package_name",
"sequence": 1
}
],
```

```
  "relationshipMatcher": "attribute name"
  }
  ],
  "numberOfElements": 15,
  "totalElements": 15
 }
```

Profile PII Data in a Data Model

A Test Data Engineer (TDE) can perform a Data Profiling scan job on an existing Data Model against a set of Classifier Packs. A Data Profiling scan job helps a TDE to identify any Personally Identifiable Information (PII) data across multiple data sources in an environment. In the request body you can specify the job configuration, filters, scan level, and store samples.

You can obtain the projectId from the Discover Entity Relationships in an Environment REST API.

```
POST https://<server>:<host>/TDMModelService/api/ca/v1/datamodel/profile?
projectId=7366&versionId=7367
```

- **REQUEST BODY:**

```
{
  "scheduledTime":946684800000,
  "parameters":{
                "connProfiles":["creditcard","travel"],
                "environment":"env1",
                "environmentId":10104,
                "classifierPacks":[15,323,10079,10089,10119,19961],
                "scanLevel":0,
                "storeSamples":false,
                "scanLevelSet":true,
                "storeSamplesSet":true,
                "dataSources":["cc","dd"],
                "isIncludeFilter":false,
                "filters":null,
                "refreshToken":null,
                "RefreshToken":null
              }
}
```

- **RESPONSE:**200 OK
- **RESPONSE BODY:**

```
{
  "jobId":{jobId}
}
```

Now, you have started a Data Profiling scan job. You can also obtain the Job ID from the above response to check the job status. For more information on how to check the job status, see Check Job Status.

Retrieve Results for Data Profiling in a Data Model

A Test Data Engineer can retrieve results for a Data Profiling scan job on an existing Data Model. This REST API helps you to identify the specific table names, column names, schema names, connection profile names, and tags. Depending on the content of the tables you can mark the table as Not PII.

```
GET https://<server>:<host>/TDMModelService/api/ca/v1/datamodel/profile/piidata?
projectId=&versionId=&page=1&size=5&q=string>
```

- **Example:** Use the following API to search for tables with tags equal to Surname, or tags containing the word "Code", such as Post Code, Zip Code.
  ```
  GET https://<server>:<host>/TDMModelService/api/ca/v1/datamodel/profile/piidata?
  projectId=1&versionId=1&page=1&size=5&q="tag=Surname tag=*Code"
  ```
- **Example:** Use the following API to search for table names, column names, schema names, connection profile names, and tags matching the word "account".
  ```
  GET https://<server>:<host>/TDMModelService/api/ca/v1/datamodel/profile/piidata?
  projectId=1&versionId=1&page=1&size=5&q="account"
  ```

  NOTE

  All details including response code of this REST API is same as Query Through the Tables REST API for Data Profiling.

Mark Profiled Tables as Not PII in a Data Model

After retrieveing results for a Data Profiling scan in a Data Model, a Test Data Engineer can identify if a table can be marked as Not PII.

```
PATCH https://<server>:<host>/TDMModelService/api/ca/v1/datamodel/profile/piidata/604?
projectId=<id>&versionId=<id>
```

NOTE

All details including response code of this REST API is same as Mark Empty/Known/Unmatched Tables as Not PII REST API for Data Profiling.

Use APIs to Audit and Mask PII Data

Test Data Engineers may Audit and Mask data sources either though the CA TDM Portal or via the auditing and masking APIs. For more information on the Audit PII workflow, see PII Audit Using CA TDM Portal.

This page refers to the following API Services:

- TestDataManager
- TDMModelService
- TDMMaskingService
- TDMConnectionProfileService
- TDMJobService

TIP

We recommend that for each application you want to discover and profile Personally Identifiable Information (PII) data in is part of a separate project.

Common parameters

The following parameters are common to all APIs:

- `<server>`
  The system where the CA TDM Portal instance is available.
- **<port>**
  The address of the port through which you access your CA TDM installation.

Pagination parameters

APIs that incorporate pagination can take the following parameters:

- **size=<size>**
  For enumeration of long results. `size` defines the number of elements you receive per page of results. The default page size is 10.
- **page=<page>**
  For enumeration of long results. Defines the page of your results that you want to receive in the body of your response. Page numbers start at 0.
  Divide the total number of elements by `size` to determine the number of pages your results contain.

APIs for audit and masking

Import Classifiers

As a Test Data Engineer, you can customize Classifiers as per your requirement and import classifiers into the CA TDM Portal. For more information about Classifiers, see Manage Data Classifiers.

NOTE

During installation, CA TDM imports a standard set of Classifiers and SeedLists into CA TDM Portal. SeedLists are part of Classifier Packs.

Syntax

```
POST https://<server>:<port>/TDMModelService/api/ca/v1/profiler/classifiers
```

Parameters

- **definitionFile** (formData)
  Zip file containing classifier hierarchy definition to be imported.
- **parentId** (query, long)
  Container where the definitions should be imported.
- **onduplicate** (query, string)
  Defines behaviour during import, if a duplicate resource is found what to do. Possible values:
  - **IGNORE**
  - **ABORT**
  - **OVERWRITE**

Example

```
POST https://host:8443/TDMModelService/api/ca/v1/profiler/classifiers?
onduplicate=OVERWRITE
```

- **FORM DATA**

Multipart/form-data	Includes a file named 'definitionsFile' in a Zip file containing the Classifier Pack

- **RESPONSE BODY**

```
{
  "classifiersCreated": 0,
  "classifiersUpdated": 2,
  "containersCreated": 0,
  "containersUpdated": 0,
  "seedListCreated": 0,
  "seedListUpdated": 1,
  "duplicateClassifiers": 2,
  "duplicateSeedlist": 1,
  "duplicateContainers": 3,
  "duplicateOption": "OVERWRITE"
}
```

You have imported the required Classifiers and now you can continue to set up Audit PII.

Setup PII Data Scan

You as Test Data Engineer can setup PII Data scan by performing the following activities:

1. Enumerate Data Sources:
   You as Test Data Engineer can perform PII Data scan on one or more Connection Profiles or an Environment.
   a. enumerate_connection_profilesAs a Test Data Engineer you can determine the Connection Profiles on which you want to perform a PII Data scan job.
   b. Enumerate an Environment
   As a Test Data Engineer you can determine the Environment on which you want to perform a PII Data scan job.
2. Enumerate Classifiers
   You as Test Data Engineer determine the Classifiers against which you want to perform a PII Data scan job.

NOTE

If the required Connection Profiles and Data Classifiers are predefined in the CA TDM Portal, continue to Initiate?a Data Profiling Scan Job.

Enumerate Connection Profiles

Enumerating Connection Profiles enables you to identify the Connection Profiles that are defined for your CA TDM Portal instance. You can review this list of Connection Profiles to identify the Connection Profiles on which you want to Audit PII. If a Connection Profile does not exist to the required database, the Connection Profile will have to be defined through the CA TDM Portal or the Connection Profiles REST API.

For more information about how to create a Connection Profile using REST API, see Create a Connection Profile.

TIP

We recommend that you make a note of the value of the `name` parameter value returned in the response. The `name` parameter value is used as an input for the Audit PII scan job.

Syntax

```
GET https://<server>:<port>/TDMConnectionProfileService/api/ca/v1/connectionProfiles
```

Parameters

- (Optional) Filter criteria for results:
  - **groupOnly** (query, Boolean)

Example

```
GET https://host:8443/TDMConnectionProfileService/api/ca/v1/connectionProfiles?
groupOnly=true
```

- **RESPONSE BODY**


```
[
  {
    "name": "Travel_E",
    "description": "none",
    "dbType": "sql server",
    "server": "host",
    "port": "",
    "instance": "",
    "service": "",
    "database": "Travel_E",
    "schema": "",
    "username": "sa",
    "password": "nwoUjTtlSBWRSbl9Z1i9p7ZP3iv42jgNimHwoNq+RNUNfnfO+DV-E55F",
    "datasourceUrl": "jdbc:sqlserver://host;database=Travel_E",
    "datasourceDriver": "com.microsoft.sqlserver.jdbc.SQLServerDriver",
    "created": 1513012472677,
    "modified": 1513012472677,
    "createdBy": 1,
    "additionalConnectionProperties": ""
  },
  {
    "name": "TravelDB",
    "description": "",
    "dbType": "sql server",
    "server": "host",
    "port": "",
    "instance": "",
    "service": "",
    "database": "travel",
    "schema": "",
    "username": "sa",
    "password": "taPAa1cuko63c9NCX32Y8DkKS3RkES8aWMKczIlgFFlq5wYyPk-yE24V",
    "datasourceUrl": "jdbc:sqlserver://host;database=travel",
    "datasourceDriver": "com.microsoft.sqlserver.jdbc.SQLServerDriver",
    "created": 1513063321730,
    "modified": 1513063321730,
    "createdBy": 1,
    "additionalConnectionProperties": ""
```

```
    }
  ]
```

You have enumerated the Connection Profiles and now you can continue to enumerate Classifiers.

Enumerate Classifiers

NOTE

If you want to use all Classifier Packs for an Audit PII scan, you do not need to select Classifiers. Set `classifierPacks` parameter to 0 to initiate a Audit PII scan using all Classifier Packs.

Enumerating Classifiers returns a list of Classifiers and enables you to identify the specific Classifier against which you want to perform a Audit PII scan. You can use multiple Classifier Packs during Audit PII.

You can use one of the following REST APIs to enumerate Classifiers:

- Enumerate Classifiers in a Container
- Enumerate Classifiers and Containers which are children of a container

Enumerate Classifiers in a Container (Classifier Pack)

Syntax

```
GET https://<server>:<port>/TDMModelService/api/ca/v1/profiler/classifiers/containers/
<containerId>
```

Parameters

- **containerId** (path, long)
  Container of which to enumerate contents.
- (Optional) Filter criteria for results:
  - **recursive** (query, long)
  - **classifierType** (query, string)
  - **classifierClass** (query, string)
  - **classifierOrigin** (query, string)
  - **tags** (query, string)

Example

```
GET https://host:8443/TDMModelService/api/ca/v1/profiler/classifiers/containers/0
```

The response for this REST API includes the Classifier Pack **name** and each Classifier Pack includes a specific **id** that you can use in the request to initiate a Audit PII scan.

For example, to use the Common Classifier Pack, find the Classifier Pack name ("name":"**Common**") in the response and use the associated id parameter ("id":"**19668**") when you perform an Audit PII scan.

- **RESPONSE**
  200 OK
- **RESPONSE BODY**

```
{
  "name": "ROOT",
  "names": [
            {
              "id": 1,
              "lang": "en",
```

```
              "value": "ROOT"
            }
          ],
 "description": "The root classifier container",
 "descriptions": [
                  {
                    "id": 2,
                    "lang": "en",
                    "value": "The root classifier container"
                  }
                ],
 "root": true,
 "created": 1513012309553,
 "updated": null,
 "createdBy": "SYSTEM",
 "updatedBy": null,
 "containedContainers": [

{
                          "name": "Common",
                          "names": [

{
                                      "id": 19668,
                                      "lang": "en",
                                      "value": "Common"
                                    }
                                  ],
                          "description": null,
                          "descriptions": [],
                          "root": null,
                          "created": 1513012385037,
                          "updated": null,
                          "createdBy": "SYSTEM",
                          "updatedBy": null,
                          "containedContainers": [],
                          "containedClassifiers": [],
                          "id": 19667
                        },

{
                          "name": "Germany",
                          "names": [

{
                                      "id": 19950,
                                      "lang": "en",
```

```
                                    "value": "Germany"
                                }
                            ],
                    "description": null,
                    "descriptions": [],
                    "root": null,
                    "created": 1513012387117,
                    "updated": null,
                    "createdBy": "SYSTEM",
                    "updatedBy": null,
                    "containedContainers": [],
                    "containedClassifiers": [],
                    "id": 19949
                },

    {
                    "name": "Japan",
                    "names": [

    {
                                    "id": 29690,
                                    "lang": "en",
                                    "value": "Japan"
                                }
                            ],
                    "description": null,
                    "descriptions": [],
                    "root": null,
                    "created": 1513012413520,
                    "updated": null,
                    "createdBy": "SYSTEM",
                    "updatedBy": null,
                    "containedContainers": [],
                    "containedClassifiers": [],
                    "id": 29689
                },

    {
                    "name": "Sweden",
                    "names": [

    {
                                    "id": 29698,
                                    "lang": "en",
                                    "value": "Sweden"
                                }
                            ],
```

```
                "description": null,
                "descriptions": [],
                "root": null,
                "created": 1513012414340,
                "updated": null,
                "createdBy": "SYSTEM",
                "updatedBy": null,
                "containedContainers": [],
                "containedClassifiers": [],
                "id": 29697
              },

    {
                "name": "UK",
                "names": [

    {
                            "id": 29722,
                            "lang": "en",
                            "value": "UK"
                          }
                        ],
                "description": null,
                "descriptions": [],
                "root": null,
                "created": 1513012415370,
                "updated": null,
                "createdBy": "SYSTEM",
                "updatedBy": null,
                "containedContainers": [],
                "containedClassifiers": [],
                "id": 29721
              },

    {
                "name": "USA",
                "names": [

    {
                            "id": 39536,
                            "lang": "en",
                            "value": "USA"
                          }
                        ],
                "description": null,
                "descriptions": [],
                "root": null,
```

```
                                "created": 1513012443263,
                                "updated": null,
                                "createdBy": "SYSTEM",
                                "updatedBy": null,
                                "containedContainers": [],
                                "containedClassifiers": [],
                                "id": 39535
                            }
                        ],
    "containedClassifiers": [],
    "parentId": -1,
    "id": 0
}
```

Enumerate classifiers and containers (classifier packs) which are children of a container (classifier pack)

Syntax

```
GET https://<server>:<port>/TDMModelService/api/ca/v1/profiler/classifiers/classifiers/
{classifierId}
```

Parameters

- **classifierId** (path, long)
  Container (classifier pack) of which to enumerate contents.

Example

```
GET https://host:8443/TDMModelService/api/ca/v1/profiler/classifiers/classifiers/0
```

The response for this REST API includes a list of Classifiers and Classifier Pack names for a specific Classifier id. The response code also includes the masking function associated with the tag of each Classifier.

- **RESPONSE:** `200 OK`
- **RESPONSE BODY:**

```
{
"classifierClass": "string",
"classifierOrigin": "string",
"classifierType": "string",
"config":
[
{
"id": 0,
"name": "string",
"value": "string"
}
],
"created": "2018-07-02T14:11:00.070Z",
"createdBy": "string",
"description": "string",
"descriptions":
```

```
[
{
"id": 0,
"lang": "string",
"value": "string"
}
],
"id": 0,
"name": "string",
"names":
[
{
"id": 0,
"lang": "string",
"value": "string"
}
],
"parentId": 0,
"tags": "string",
"updated": "2018-07-02T14:11:00.070Z",
"updatedBy": "string"
}
```


You have enumerated the required Classifiers and now you can continue to initiate a Audit PII scan job.

Initiate an Audit PII Scan Job

You as a Test Data Engineer can start a Audit PII scan job using the Connection profiles and Classifier Packs. In the request body you can specify the classifier packs and connection profiles you want to use.

Syntax

```
POST https://<server>:<port>/TDMJobService/api/ca/v1/jobs
```

Parameters

- **Request body** (JSON, body)

Example

```
POST https://host:8443/TDMJobService/api/ca/v1/jobs
```

- **REQUEST**


```
{
  "name": "Profiling Scan for PII [ 1.0 ]",
  "description": "Profiling Scan - project id: 2352",
  "projectId": 2352,
  "versionId": 2353,
  "type": "PIISCAN",
  "origin": "profiling",
  "scheduledTime": 1513063791088,
  "parameters": {
```

```
                    "connProfiles": [
                                      "TravelDB"
                                    ],
                    "classifierPacks": [
                                         19667,
                                         19949,
                                         39535
                                        ],
                    "scanType": "FAST",
                    "refreshToken": null
                  }
}
```

- **RESPONSE:** 201 Created
- **RESPONSE :**

```
{
  "message": "Job successfully submitted with name: Profiling Scan for PII [ 1.0 ],
 id: 2",
  "name":
 null,
  "jobId": 2,
  "description": null,
  "projectName": null,
  "projectId": 0,
  "versionId": null,
  "createdBy": null,
  "email": null,
  "scheduledTime": null,
  "startTime": null,
  "endTime": null,
  "status": null,
  "type": null,
  "parentId": 0,
  "jobs": null,
  "duration": null,
  "artifactLocation": null,
  "origin": null,
  "parameters": null,
  "statusMessage": null,
  "runningStatus": null,
  "created": null
}
```

You have started a Audit PII scan job and now you can continue to check the job status with the value of the **jobId** attribute.

Check Job Status

You as a Test Data Engineer can obtain the status of a specific Job to identify if the scan results are ready for review. You can obtain the Job ID from the response provided in Initiate a Audit PII Scan Job.

You can use one of the following REST APIs to obtain the status of a specific Job:

1. **Syntax**

   ```
   GET https://<server>:<port>/TDMJobService/api/ca/v1/jobs/{jobId}
   ```

   Parameters

   – **jobId** (path, long)

   Example

   ```
   GET https://host:8443/TDMJobService/api/ca/v1/jobs/2
   ```

   – **RESPONSE:**
   200 OK

   – **RESPONSE :**The `runningStatus` parameter in the response identifies the job status.

   ```
   {
     "name": "Profiling Scan for PII-Example
    [ 1.0 ]",
     "jobId": 2,
     "description": "Profiling Scan - project id: 2362",
     "projectName": "PII-Example",
     "projectId": 0,
     "versionId": null,
     "createdBy": "Administrator",
     "email": null,
     "scheduledTime": "2017-12-13T09:36:53Z",
     "startTime": "2017-12-13T09:41:19Z",
     "endTime": "2017-12-13T09:41:36Z",
     "status": "Completed",
     "type": "PIISCAN",
     "parentId": 0,
     "jobs": [],
     "duration": 16630,
     "artifactLocation": null,
     "origin": "profiling",
     "parameters": {},
     "statusMessage":
    "",
     "runningStatus": "Completed",
     "created": null
   }
   ```

2. Use this REST API to get more detailed status of a job. The response includes the number of tables and columns scanned for PII data.**Syntax**

   ```
   GET https://<server>:<port>/TDMModelService/api/ca/v1/profiler/jobs/{jobId}
   ```

   Parameters

   – **jobId**

   Example

```
GET https://host:8443/TDMModelService/api/ca/v1/profiler/jobs/2
```

The `state` parameter in the response identifies the job status and also provides more information about the Profiling job progress.

- – **RESPONSE :**

```
{
    "jobID": 2,
    "jobName": "Profiling Scan for PII-Example [ 1.0 ]",
    "projectID": 1180,
    "projectVersionID": 1181,
    "projectName": "QTP - Example Project",
    "state": "SCAN_COMPLETE",
    "startDate": 1513412020743,
    "stopDate": 1513412055037,
    "completeDate": null,
    "signOffRequestedDate": null,
    "setup": null,
    "submittedBy": "Administrator",
    "approvedBy": null,
    "approved": null,
    "reason": null,
    "severity": 0.0,
    "totalPii": 57,
    "contentClassifierHash": -1693587654,
    "contentSeedlistHash": 1009518500,
    "columnClassifierHash": -1072720395,
    "columnSeedlistHash": null,
    "totalTables": 40,
    "totalColumns": 298,
    "columnsClassified": 46,
    "tablesClassified": 23,
    "tablesScanned": 40,
    "columnsScanned": 298,
    "tablesReviewed": 0,
    "totalReviewers": 0,
    "totalApprovers": 0,
    "listApprovers": [],
    "listReviewers": [],
    "warnings": ""
}
```

After the job status changes to Completed or SCAN_COMPLETE state, you can continue to query through the tables to obtain table details.

Query Through the Tables

As a Test Data Engineer, you can query the tables to identify the specific table names, column names, schema names, connection profile names, or tags. Depending on the content of the tables you can mark the table as Not PII.

Syntax

```
GET https://<server>:<host>/TDMModelService/api/ca/v1/profiler/jobs/{jobId}/piidata?
```

Parameters

- **jobId** (JSON, body)
  ID of the PII Profiling job
- **hasTags** (boolean)
  True = Only include columns with PII tag(s)
- **history** (boolean)
  True = Include tag history
- Pagination parameters
- **q** (query, string)
  Filter query results. You can perform a search on the relevant data source based on one or more of the following key fields:
  - **table=**
    Matches a table name.
  - **column=**
    Matches a column name.
  - **schema=**
    Matches a schema name.
  - **profile=**
    Matches a connection profile name.
  - **tag=**
    Matches a tag name.

    NOTE

    If no key field is specified then all tables, columns, schema names, connection profiles, and tags are searched.

  This REST API supports the basic wild card characters such as * (used to match one or more characters) and ? (used to match a single character). Supplied search terms are treated with ANY logic (i.e. condition A=true OR condition B=true). For example, when you perform a search for "tag=Surname tag=Title", the response includes all tables that have either a Surname tag or a Title tag.

ExampleUse the following API call to search for tables with tags equal to Surname, or tags that end with the word "Code", such as Post Code, Zip Code.

```
GET https://host:8443/TDMModelService/api/ca/v1/profiler/jobs/2/piidata?
page=1&size=5&q="tag=Surname tag=*Code"
```

- **RESPONSE**

```
{
 "elements": [
              {
                "databaseName": "travel",
                "schemaName": "dbo",
                "tableName": "ACCESS_CONTROLS",
                "profileName": "travel",
                "tableId": 601,
```

```
  "rowCount": 418,
  "columnCount": 4,
  "tagHistory": null,
  "piiTags": [
              "Surname"
            ],
  "confirmed": false,
  "reason": null,
  "reviewer": null,
  "severity": 2,
  "matchedSamples": 3,
  "dateReviewed": null,
  "notPII": false
},
{
  "databaseName": "travel",
  "schemaName": "dbo",
  "tableName": "ACCOUNT_PERIODS",
  "profileName": "travel",
  "tableId": 602,
  "rowCount": 789,
  "columnCount": 18,
  "tagHistory": null,
  "piiTags": [
              "Surname"
            ],
  "confirmed": false,
  "reason": null,
  "reviewer": null,
  "severity": 2,
  "matchedSamples": 2,
  "dateReviewed": null,
  "notPII": false
},
{
  "databaseName": "travel",
  "schemaName": "dbo",
  "tableName": "ADDRESS",
  "profileName": "travel",
  "tableId": 603,
  "rowCount": 10,
  "columnCount": 9,
  "tagHistory": null,
  "piiTags": [
              "Given Name",
              "Post Code",
```

```
                "Surname",
                "Towns"
            ],
  "confirmed": false,
  "reason": null,
  "reviewer": null,
  "severity": 2,
  "matchedSamples": 21,
  "dateReviewed": null,
  "notPII": false
},
{
  "databaseName": "travel",
  "schemaName": "dbo",
  "tableName": "AIRCRAFT_TYPES",
  "profileName": "travel",
  "tableId": 605,
  "rowCount": 11,
  "columnCount": 4,
  "tagHistory": null,
  "piiTags": [
                "Post Code"
            ],
  "confirmed": false,
  "reason": null,
  "reviewer": null,
  "severity": 2,
  "matchedSamples": 2,
  "dateReviewed": null,
  "notPII": false
},
{
  "databaseName": "travel",
  "schemaName": "dbo",
  "tableName": "AIRCRAFT_LAYOUTS",
  "profileName": "travel",
  "tableId": 604,
  "rowCount": 1307,
  "columnCount": 12,
  "tagHistory": null,
  "piiTags": [
                "ZIP Code",
                "Account Number"
            ],
  "confirmed": false,
  "reason": null,
```

```
                "reviewer": null,
                "severity": 1,
                "matchedSamples": 0,
                "dateReviewed": null,
                "notPII": false
              }
            ],
  "numberOfElements": 5,
  "totalElements": 40
}
```


You have queried through all the tables and now you can identify and mark tables as Not PII.

Mark Empty/Known/Unmatched Tables as Not PII

After querying through the tables and retrieving relationships for a table, as a Test Data Engineer you can identify if a table can be marked as Not PII. For example, you can mark a table as Not PII based on the following conditions:

- Table is empty
  where, `rowCount` parameter is 0
- Table does not match any PII tags
  where, `piiTags` parameter is empty
- Known table that does not contain any PII data

Syntax

```
PATCH https://<server>:<host>/TDMModelService/api/ca/v1/profiler/jobs/{jobId}/piidata/
{tableId}
```

Parameters

- **jobId** (JSON, body)
  ID of the PII Profiling job
- **tableId** (JSON, body)
  ID of the table on which to change tag data
- **history** (boolean)
  True = Include tag history
- **patch** (JSON, body)
  Properties to update within the table object.

Example

Use the following REST API to mark a table as Not PII, and to confirm that a table is not using any PII data.

```
PATCH https://host:8443/TDMModelService/api/ca/v1/profiler/jobs/2/piidata/604
```

- **REQUEST BODY**:


```
{
  "confirmed": true,
  "notPII": true,
  "reason": "Not PII Data"
}
```


- **RESPONSE**:

200 OK

- **RESPONSE BODY:**


```
{
  "databaseName": "travel",
  "schemaName": "dbo",
  "tableName": "ACCESS_CONTROLS",
  "profileName":
 "TravelDB",
  "tableId": 604,
  "rowCount": 418,
  "columnCount": 4,
  "tagHistory": null,
  "piiTags":
 [],
  "confirmed":
 true,
  "reason": "Not PII Data",
  "reviewer": "Administrator",
  "severity": 0.0,
  "matchedSamples": 0,
  "dateReviewed":
 null,
  "notPII": true
}
```


NOTE

After marking empty/known/unmatched tables as Not PII, login to the CA TDM Portal and review the remaining tables and create a report. For more information, see End-to-End Scenario for PII Audit.

Audit Log Extraction

Use this REST API to get an audit on all operations that have been performed on a Audit PII scan job. The response of this REST API includes details about who initiated a scan, who reviewed and confirmed the tables and so on. You can filter results with the query. See available filter fields below.

Syntax

```
GET https://<server>:<host>/TestDataManager/api/ca/v1/auditlogs
```

Parameters

- **q** (query)
  Filter query results. You can filter the audit logs you receive with the following key fields (use wildcard % to match zero, one or multiple characters in this position):
  – **format** (string)

Format in which results returned. JSON (default) or ZIP-CSV.

- **origin** (string)
- **description** (string)
- **type** (string)
- **status** (string)
- **user_name** (string)
- **link_id** (string)
  This corresponds with the jobId.
- **proj_id** (string)
- **proj_version_id** (string)
- **timestamp_start** (string)
- **timestamp_end** (string)
- **sort** (string)
- **order** (string)
- **Pagination parameters** (integers)

  NOTE

  parameter **pagesize** replaces size**.**
- **cachesize** (integer in range 1 to 1,000,000)
  Performance tuning parameter. Default = 1,000.

Example

Filter by Job ID (`link_id` parameter)

```
GET https://host:8443/TestDataManager/api/ca/v1/auditlogs?link_id=2
```

- **RESPONSE BODY:**


```
{"elements":[
{
"id":11,
"user_name":"Administrator",
"link_id":2,
"origin":"PII Data Scan",
"type":"MODELPIISCAN",
"status":"CREATED",
"description":"STARTED",
"timestamp":"2018-07-27 08:48:26.83",
"proj_id":2346,
"proj_version_id":2347
},
{
"id":12,
"user_name":"Administrator",
"link_id":2,
"origin":"PII Data Scan",
"type":"MODELPIISCAN",
"status":"STARTED",
"description":"STARTED",
"timestamp":"2018-07-27 08:48:27.027",
```

```
"proj_id":2346,
"proj_version_id":2347
},
{
"id":13,
"user_name":"Administrator",
"link_id":2,
"origin":"PII Data Scan",
"type":"MODELPIISCAN",
"status":"COMPLETED",
"description":"COMPLETED",
"timestamp":"2018-07-27 08:48:56.107",
"proj_id":2346,
"proj_version_id":2347
},
{"id":14,
"user_name":"Administrator",
"link_id":2,
"origin":"PII Data Scan",
"type":"MODELPIISCAN",
"status":"COMPLETED",
"description":"COMPLETED",
"timestamp":"2018-07-27 08:48:56.357",
"proj_id":2346,
"proj_version_id":2347}
],
"numberOfElements":4,
"totalElements":4,
"totalPages":1}
```


You have obtained the audit log and identified all operations that have been performed on a Audit PII scan job.

Retrieve a List of Tags

A Test Data Engineer (TDE) can enumerate a list of tags that are available in the CA TDM Portal. The search criteria follows the RSQL format and allows the query to be filtered on any of the resource's field values, such as 'name', 'id', 'whoCreated', etc. For more information about the RSQL format, see https://github.com/jirutka/rsql-parser.

This REST API supports the wild card characters * (used to match one or more characters) and ? (used to match a single character). Search terms are treated with ANY logic (i.e. condition A=true OR condition B=true). For example, when you perform a search for "q=name==co*", the response includes all tags with Country or County names. The query parameter in this REST API is case insensitive.

NOTE

If no search criteria is included in this REST API, a list of all tags is returned.

Syntax

```
GET https://<server>:<host>/TDMModelService/api/ca/v1/profiler/tags
```

Parameters

- Pagination parameters
- **q** (query)
  Filter criteria.

Example:

The following example only returns tags whose name starts 'co' (not case-sensitive). It returns the first page of results, with up to 20 entries per page.

```
GET https://host:8443/TDMModelService/api/ca/v1/profiler/tags?page=0&size=20&q=name==co*
```

- **RESPONSE:**
  200 OK
- **RESPONSE BODY:**

```
{
  "elements": [
                {
                  "id": 1271674,
                  "name": "Country",
                  "whoCreated": "Administrator",
                  "programCreated": "TDMApi"
                },
                {
                  "id": 1281739,
                  "name": "County",
                  "whoCreated": "Administrator",
                  "programCreated": "TDMApi"
                }
              ],
  "numberOfElements": 2,
  "totalElements": 2
}
```

Retrieve Tag Details for a Specific Tag

A Test Data Engineer (TDE) can retrieve details about a specific tag based on a tagId. This helps a TDE to understand how the data identified by a tag can be masked.

Syntax

```
GET https://<server>:<host>/TDMModelService/api/ca/v1/profiler/tags/{tagId}
```

Parameters

- **tagId** (integer, long)
  ID of the tag for which to return details.

Example:

```
GET https://host:8443/TDMModelService/api/ca/v1/profiler/tags/1271674
```

- **RESPONSE:**
  200 OK
- **RESPONSE BODY:**

```
{
```

```
    "id": 1271674,
    "name": "Country",
    "dateCreated": 1524242459710,
    "whoCreated": "Administrator",
    "programCreated": "TDMApi"
  }
```

Retrieve a list of Matched Classifiers for all Columns in a Table

As a Test Data Engineer (TDE), you can retrieve a list of counts of classifiers that matched a column in a table. These results are organized by column. This helps a TDE to understand how the data identified by a tag can be masked.

In the response code:

- parameter **clsMatches** provides a list of all classifier IDs that have matched to the column together with the count of matched values from the column.
- parameter **clsMaxMatch** provides the Classifier Id with the highest match count.

Syntax

```
GET https://<server>:<host>/TDMModelService/api/ca/v1/profiler/jobs/{jobId}/piidata/
{tableId}/columns
```

Parameters

- **tagId** (integer, long)
  ID of the tag for which to return details.
- **tableId** (integer, long)
  ID of the tag for which to return details.
- Pagination parameters
- **hasTags** (boolean)
  True = Only include columns with PII tag(s)
- **history** (boolean)
  True = Include tag history

Example

```
GET https://host:8443/TDMModelService/api/ca/v1/profiler/jobs/18/piidata/1644/columns
```

- **RESPONSE:**
  `200 OK`
- **RESPONSE BODY:**

```
{"elements":
          [
            {
              "columnName":"ID",
              "columnId":39207,
              "dataType":"numeric",
              "piiTags":[],
              "clsMatches":{},
              "clsMaxMatch":null,
              "tagsSet":false,
              "reason":null,
```

```
              "dateReviewed":null,
              "reviewer":null,
              "severity":null,
              "tagHistory":[]
            },
            {
              "columnName":"NAME",
              "columnId":39208,
              "dataType":"varchar",
              "piiTags":["Country","County","Ethnicity","Gender","Given
Name","Towns"],
              "clsMatches":{
                            "108064":1,
                            "108992":2,
                            "98326":4,
                            "108969":1,
                            "30009":1,
                            "108120":17,
                            "98042":7,
                            "109002":9
                          },
              "clsMaxMatch":108120,
              "tagsSet":true,
              "reason":null,
              "dateReviewed":null,
              "reviewer":null,
              "severity":null,
              "tagHistory":[]
            },
            {
              "columnName":"STATE_PROVINCE",
              "columnId":39209,
              "dataType":"varchar",
              "piiTags":["Country","Ethnicity","Given Name","Surname","Title"],
              "clsMatches":{
                            "109842":2,
                            "117845":2,
                            "98681":1,
                            "108969":6,
                            "98042":10,
                            "109002":1
                          },
               "clsMaxMatch":98042,
               "tagsSet":true,
               "reason":null,
               "dateReviewed":null,
```

```
  "reviewer":null,
  "severity":null,
  "tagHistory":[]
},
{
  "columnName":"TIME_ZONE_CITY",
  "columnId":39210,
  "dataType":"varchar",
  "piiTags":[],
  "clsMatches":{},
  "clsMaxMatch":null,
  "tagsSet":false,
  "reason":null,
  "dateReviewed":null,
  "reviewer":null,
  "severity":null,
  "tagHistory":[]
},
{
  "columnName":"TIME_ZONE_TO_GMT",
  "columnId":39211,
  "dataType":"numeric",
  "piiTags":[],
  "clsMatches":{},
  "clsMaxMatch":null,
  "tagsSet":false,
  "reason":null,
  "dateReviewed":null,
  "reviewer":null,
  "severity":null,
  "tagHistory":[]
},
{
  "columnName":"CITY_MAP_LEVEL",
  "columnId":39212,
  "dataType":"numeric",
  "piiTags":[],
  "clsMatches":{},
  "clsMaxMatch":null,
  "tagsSet":false,
  "reason":null,
  "dateReviewed":null,
  "reviewer":null,
  "severity":null,
  "tagHistory":[]
},
```

```
            ],
    "numberOfElements":6,
    "totalElements":6
}
```

download fast data masker configuration

A Test Data Engineer (TDE) can download a PII data scan job in the Fast Data Masker (FDM) configuration format. You can import this FDM configuration file into FDM to mask PII data, or run .bat file provided in the zip file to mask PII data.

The FDM configuration is downloaded as a zip file and includes the following files for each database connection:

- **Text file**
  Specifies the connection details for FDM to connect to the database.
- **CSV file**
  Specifies the masking rules for tables and columns for a particular database.
- **.BAT file**
  The script file can be used on a Windows machine with FDM installed to run FDM in batch mode to perform the masking operation.

Syntax

```
GET https://<server>:<host>/TDMModelService/api/ca/v1/datamodel/profiler/fdm
```

Parameters

- **projectId**Specifies the project ID.
- **versionId**Specifies the version ID.
- **environmentId**
  Specifies the environment ID to mask.
- (Optional) **confirmedOnly**
  Includes all confirmed tables.
  Values: true or false (default)
- (Optional) **fileName**Specifies the name of the downloaded FDM configuration file.
- (Optional) **excNotPii**Excludes all tables marked as Not PII.Values: true or false (default)
- (Optional) **dataSources**Filter for data sources to mask.
- (Optional) **options**Filter for masking options.

Example

```
GET https://host:8443/TDMModelService/api/ca/v1/datamodel/profiler/fdm?
projectId=1180&versionId=1181&confirmedOnly=true&excNotPii=true&environmentId=6
```

- **RESPONSE:**
  200 OK

You have downloaded the FDM configuration and now you can use this with FDM to mask PII data. You can run the appropriate .bat file to perform the masking directly. Alternatively, to import a masking configuration into FDM, copy the required text file(s) into your Connection Files directory, and select the appropriate connection in FDM. Choose **Open Saved Mask** and open a .csv file from the zip and perform masking using FDM. For more information about how to perform masking using the FDM configuration file in FDM, see Mask Profiled Data in a Data Model.

Initiate Masking for a Data Model

A Test Data Engineer (TDE) can initiate a masking job for an entire Data Model. This helps the user to mask any sensitive data that should not be used for purposes such as development and testing.

Syntax

```
POST http://<server>:<host>/TDMMaskingService/api/ca/v1/masking/jobs/start
```

Parameters

- **Masking job** (JSON, body)

Example

```
POST http://host:8443/TDMMaskingService/api/ca/v1/masking/jobs/start
```

- **REQUEST BODY:**

```
{
  "projId" : 2378,
  "pverId" : 2379,
  "environmentId" : 704
}
```

- **RESPONSE:**
  200 OK
- **RESPONSE BODY:**

```
{
  "jobId": 507,
  "projId": 2378,
  "pverId": 2379,
  "environmentId": 704
}
```

The masking job is initiated. You can query the masking job ID to retrieve status of the masking job.

Initiate a custom masking task from a file

As a Test Data Engineer, you can initiate a masking task with a custom configuration file that you download from Fast Data Masker. With this endpoint, you can execute masking tasks you create in FDM, with TDM Portal's scalable masking architecture.

> **NOTE**
>
> Files that you download from FDM are in plain text. You can encode plain text to BASE64 with a number of text editors, for example Notepad++.

Syntax

```
POST http://<server>:<host>/TDMMaskingService/api/ca/v1/masking/jobs/startCustom
```

Parameters

- **projId**
  Specifies the project ID on which to execute the masking task.
- **pverId**
  Specifies the project version ID on which to execute the masking task.
- **customConnectionFile**Connection file, encoded to BASE64. For more information on Connection Files, see Use, Create, and Manage Connection Files. You can download this file with the download_fdm_config endpoint.
  Sample customConnectionFile

```
datasource=10.0.2.5
 username=sa
```

```
epassword=XnKS5U=
DBMS=SQLSERVER
defaultschema=dbo
database=fdm
port=1433
host=host_name
forcedencryption=Y
```

- **customConfigFile**Masking configuration CSV file, encoded to BASE64. You can download this file with the download_fdm_configendpoint.
  Sample customConfigFile

```
Table,Column,Function,Parm1,Parm2,Parm3,Parm4,Keep Nulls,Date Format,Cross
 Reference,Override SQL,Unique Columns,XPath Element,Substr start,Substr
 length,Notes,Preformat,Update,Use Masked Values,Restart Column,From Occurance,To
 Occurance,Parm5
 address,street,HASHLOV,VEGETABLE PRODUCE,1,,,Y,,,,,,,,,,,,,,
```

- **customSeedConnectionFile (optional)**Seedtable connection file, encoded to BASE64.
  Sample SeedConnectionFile

```
datasource=win10-sql14
username=sa
epassword=Z+Lcfuij=
DBMS=SQLSERVER
defaultschema=dbo
database=Scramble
port=1433
forcedencryption=Y
```

- **customOptionsFile (optional)**Options file, encoded to BASE64.
  Sample customOptionsFile

```
SEEDTABLE=gtsrc_reference_lov1
SEEDTABLECOLUMNS=rl_ref_id,rl_ref_value,rl_ref_value2,rl_ref_value3,rl_ref_value4,rl_ref_valu
```

Example

```
POST http://host:8443/TDMMaskingService/api/ca/v1/masking/jobs/startCustom
```

- **REQUEST BODY:**

```
{
  "projId" : 2346,
  "pverId" : 2347,
  "customConnectionFile": "<customConnectionFile encoded into BASE64>",
  "customConfigFile": "<customConfigFile encoded into BASE64>",
  "customSeedConnectionFile": "<customSeedConnectionFile encoded into BASE64>",
  "customOptionsFile":"<customOptionsFile encoded into BASE64>"
}
```

- **RESPONSE:**
  200 OK
- **RESPONSE BODY:**

```
{
  "jobId": 229,
  "projId": 2346,
  "pverId": 2347,
  "jobName": "Custom masking for FDM [ 1.2 ]",
  "environmentId": 0,
  "previewMode": false,
  "storePreSamples": false,
  "autoHandleConstraints": false,
  "confirmedOnly": false,
  "excNotPii": false,
  "customConnectionFile": "<customConnectionFile encoded into BASE64>",
  "customConfigFile": "<customConfigFile encoded into BASE64>",
  "customSeedConnectionFile": "<customSeedConnectionFile encoded into BASE64>",
  "customOptionsFile":"<customOptionsFile encoded into BASE64>",
  "customMasking": true
}
```

Retrieve Status of a Job

After initiating a job, a Test Data Engineer (TDE) can query the job ID to retrieve status of the job. The "status" parameter indicates the status of a job.

Syntax

```
GET http://<server>:<host>/TDMMaskingService/api/ca/v1/masking/jobs/{jobId}
```

Parameters

- **jobId** (path, long)

Example

```
GET http://host:8443/TDMMaskingService/api/ca/v1/masking/jobs/507
```

- **RESPONSE:**
  200 OK
- **RESPONSE BODY:**

```
{
  "name": "PII Data Scan",
  "jobId": 507,
  "description": "Masksing Project: 2378 Version: 2379",
  "projectName": "fresh",
  "projectId": 0,
  "versionId": null,
  "createdBy": "Administrator",
  "email": null,
  "scheduledTime": "2018-05-09T15:07:57Z",
  "startTime": "2018-05-09T15:07:58Z",
  "endTime": null,
  "status": "Running",
```

```
    "type": "PIIMASK",
    "parentId": 0,
    "jobs": [],
    "duration": null,
    "artifactLocation": null,
    "origin": "masking",
    "parameters": {},
    "statusMessage": null,
    "runningStatus": "Running",
    "created": null
  }
```

APIs related to Mask Function Groups

Retrieve a List of Mask Function Groups for a Project Version

As a TDE, you can retrieve a list of all mask function groups in a project version.

Syntax

```
GET http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/maskConfigurations
```

Parameters

- **projectId** (query, long)Project ID.
- **versionId** (query, long)Version ID.
- (Optional) **attributeId** (query, long)Attribute ID.If attributeId is provided, the API only returns mask function groups that have a tag the same as the attribute's primary tag.
- (Optional) **q** (query, long)RSQL format (see https://github.com/jirutka/rsql-parser). Filter the query on 'tagName', 'maskGroupId', 'maskGroupLabel', 'maskGroupShared', 'classifierBased'.

Example

```
GET http://host:8443/TDMModelService/api/ca/v1/datamodel/maskConfigurations?
projectId=1&versionId=1
```

- **RESPONSE BODY:**

```
{ "elements":
[
{
"maskGroupId": 20720235,
"maskGroupLabel": "Bank Account Number (Germany)",
"tagName": "Bank Account Number",
"maskGroupShared": true,
"classifierBased": true
},
{
"maskGroupId": 20729988,
"maskGroupLabel": "Bank Account Number (Sweden)",
"tagName": "Bank Account Number",
```

```
"maskGroupShared": true,
"classifierBased": true
},
{
"maskGroupId": 20730908,
"maskGroupLabel": "Sort Code (UK)",
"tagName": "Bank Sort Code",
"maskGroupShared": true,
"classifierBased": true
}
],
"numberOfElements": 3,
"totalElements": 3
}
```


Add Mask Function Group to a Project Version

As a TDE, you can add a mask function group to a project version.

Syntax

```
POST http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/maskConfigurations
```

Parameters

- **projectId** (query, long)Project ID.
- **versionId** (query, long)Version ID.
- **maskFunctionGroup** (body, maskFunctionGroup)Mask Function Group to add.

Example

```
POST http://host:8443/TDMModelService/api/ca/v1/datamodel/maskConfigurations?
projectId=1&versionId=1&maskFunctionGroup={"maskGroupLabel":
 "My Surname Mask Function Group","maskGroupShared": false,"tagName":
"Surname","configuration": [ {"maskFunctionName":
"HASHLOV","displayName": "Surname UK","maskFunctionParams": [{"pos":
"1","value": "uknames.txt"}]}]}
```

- **RESPONSE BODY:**


```
{
"maskGroupId": 20755001,
"maskGroupLabel": "My Surname Mask Function Group",
"tagName": "Surname", "maskGroupShared": false,
"classifierBased": false,
"configuration": [
{
"maskFunctionId": 20755002,
"maskFunctionLabel": "HASHLOV (uknames.txt)",
"maskFunctionName": "HASHLOV",
"displayName": "Surname UK",
"maskFunctionParams": [
```

```
{
"pos": 1,
"value": "uknames.txt"
}
]
}
]
}
```

Get Details of a Mask Function Group

As a TDE, you can retrieve details of a mask function group.

Syntax

```
GET http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/maskConfigurations/
{maskFunctionGroupId}
```

Parameters

- **maskFunctionGroupId** (path, long)Id of Mask Function Group for which to get details.

Example

```
GET http://host:8443/TDMModelService/api/ca/v1/datamodel/maskConfigurations/20755001
```

- **RESPONSE BODY:**

```
{
"maskGroupId": 20755001,
"maskGroupLabel": "My Surname Mask Function Group",
"tagName": "Surname",
"maskGroupShared": false,
"classifierBased": false,
"configuration": [
{
"maskFunctionId": 20755002
"maskFunctionLabel": "HASHLOV (uknames.txt)",
"maskFunctionName": "HASHLOV",
"displayName": "Surname UK",
"maskFunctionParams": [
{
"pos": 1,
"value": "uknames.txt"
}
]
}
]
}
```

Update Details of a Mask Function Group

As a TDE, you can update details of a mask function group. You submit a new maskFunctionGroup object as the parameter.

Syntax

```
PATCH http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/maskConfigurations/
{maskFunctionGroupId}
```

Parameters

- **maskFunctionGroupId** (path, long)Id of Mask Function Group for which to get details.
- **maskFunctionGroup** (body, maskFunctionGroup)Mask Function Group to update to maskFunctionGroupId.

Example

```
PATCH http://host:8443/TDMModelService/api/ca/v1/datamodel/maskConfigurations/20755001?
maskFunctionGroup={"maskGroupLabel":
 "My Updated Surname Mask Function Group","maskGroupShared":
true,"tagName": "Surname","configuration": [ {"maskFunctionName":
"HASHLOV","displayName": "Surname France","maskFunctionParams": [{"pos":
 "1","value": "frenchnames.txt"}]}]}
```

- **RESPONSE BODY:**

```
{
"maskGroupId": 20755001,
"maskGroupLabel": "My Updated Surname Mask Function Group",
"tagName": "Surname",
"maskGroupShared": true,
"classifierBased": false,
"configuration": [
{
"maskFunctionId": 20755004,
"maskFunctionLabel": "HASHLOV (frenchnames.txt)",
"maskFunctionName": "HASHLOV",
"displayName": "Surname France",
"maskFunctionParams": [
{
"pos": 1,
"value": "frenchnames.txt"
}
]
}
]
}
```

Delete a Mask Function Group

As a TDE, you can delete a user-defined mask function group. If you try to delete a mask function group from a classifier, the API returns a 'Forbidden' error.

Syntax

```
DELETE http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/maskConfigurations/
{maskFunctionGroupId}
```

Parameters

- **maskFunctionGroupId** (path, long)Id of Mask Function Group to delete.

Example

```
DELETE http://host:8443/TDMModelService/api/ca/v1/datamodel/maskConfigurations/20755001
```

- **RESPONSE:**

```
true
```

Masking configurations by entities (tables) and attributes (columns)

Get list of attributes linked to a masking configuration in a project version

As a TDE, you can retrieve a list of attributes (columns) in a project version, that use the same masking configuration.

Syntax

```
GET  http://host:8443/TDMModelService/api/ca/v1/datamodel/maskConfigurations/
{maskFunctionGroupId}/attributes
```

Parameters

- **maskFunctionGroupId** (path, long)Id of Mask Function Group for which to get list of linked attributes.
- **projectId** (query, long)Project ID.
- **versionId** (query, long)Version ID.

Example

```
GET http://host:8443/TDMModelService/api/ca/v1/datamodel/maskConfigurations/20755001/
attributes?projectId=1&versionId=1
```

- **RESPONSE BODY:**


```
[
{
"attributeId": 91280,
"attributeName": "WEB_USER_PASSWORD",
"primaryTag": "Surname",
"entityName": "ACCESS_CONTROLS",
"dataSource": "ds2",
"databaseName": "travel2",
"schemaName": "dbo"
},
{
"attributeId": 91353,
"attributeName": "MONTH_NAME",
"primaryTag": "Surname",
"entityName": "ACCOUNT_PERIODS",
"dataSource": "ds2",
"databaseName": "travel2",
```

```
"schemaName": "dbo"
},
{
"attributeId": 91460,
"attributeName": "DESCRIPTION",
"primaryTag": "Surname",
"entityName": "AIRPORTS",
"dataSource": "ds2",
"databaseName": "travel2",
"schemaName": "dbo"
}
]
```


Get a list of masking configurations for tables that contain PII, for a project version

As a TDE, you can retrieve a list of masking configurations for all tables that contain PII (entities) in a project version.

Syntax

```
GET http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/entities/
maskConfigurations
```

Parameters

- **projectId** (query, long)Project ID.
- **versionId** (query, long)Version ID.

Example

```
GET http://host:8443/TDMModelService/api/ca/v1/datamodel/entities/maskConfigurations?
projectId=1&versionId=
```

- **RESPONSE BODY:**


```
[
{
"attributeId": 91748,
"attributeName": "NAME",
"primaryTag": "Country",
"numOtherTags": 1,
"maskGroupId": 20719929,
"maskGroupLabel": "Country",
"hasWhereClause": false,
"maskGroupShared": true,
"classifierBased": true,
"dataType": "char"
},
{
"attributeId": 91753,
"attributeName": "CURRENCY_DESCRIPTION",
"primaryTag": "Country",
"numOtherTags": 2,
```

```
"maskGroupId": 20719929,
"maskGroupLabel": "Country",
"hasWhereClause": false,
"maskGroupShared": true,
"classifierBased": true,
"dataType": "char"
},
{
"attributeId": 91752,
"attributeName": "FLAG_PHOTO_FILENAME",
"primaryTag": "Country",
"numOtherTags": 2,
"maskGroupId": 20719929,
"maskGroupLabel": "Country",
"hasWhereClause": false,
"maskGroupShared": true,
"classifierBased": true,
"dataType": "char"
}
]
```


Get a list of attributes and their associated masking configurations, for tables that contain PII

As a TDE, you can retrieve a list of attributes (columns) that contain PII, for a table that contains PII (an entity).

Syntax

```
GET http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/entities/{entityId}/
maskConfigurations
```

Parameters

- **entityId** (path, long)
- **projectId** (query, long)
- **versionId** (query, long)

Example

```
GET http://host:8443/TDMModelService/api/ca/v1/datamodel/entities/1234/
maskConfigurations?projectId=1&versionId=1
```

- **RESPONSE BODY:**


```
[
{
"attributeId": 91748,
"attributeName": "NAME",
"primaryTag": "Country",
"numOtherTags": 1,
"maskGroupId": 20719929,
"maskGroupLabel": "Country",
"hasWhereClause": false,
```

```
"maskGroupShared": true,
"classifierBased": true,
"dataType": "char"
},
{
"attributeId": 91753,
"attributeName": "CURRENCY_DESCRIPTION",
"primaryTag": "Country",
"numOtherTags": 2,
"maskGroupId": 20719929,
"maskGroupLabel": "Country",
"hasWhereClause": false,
"maskGroupShared": true,
"classifierBased": true,
"dataType": "char"
},
{
"attributeId": 91752,
"attributeName": "FLAG_PHOTO_FILENAME",
"primaryTag": "Country",
"numOtherTags": 2,
"maskGroupId": 20719929,
"maskGroupLabel": "Country",
"hasWhereClause": false,
"maskGroupShared": true,
"classifierBased": true,
"dataType": "char"
}
]
```


Get the current masking configuration for an attribute, for a project version

As a TDE, you can retrieve the masking configuration for an atttribute (column), in a project version.

Syntax

```
GET http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/entities/{entityId}/
attributes/{attributeId}/maskConfigurations
```

Parameters

- **attributeId** (path, long)
- **entityId** (path, long)
- **projectId** (query, long)
- **versionId** (query, long)

Example

```
GET http://host:8443/TDMModelService/api/ca/v1/datamodel/entities/1234/attributes/91748/
maskConfigurations?projectId=1&versionId=1
```

- **RESPONSE BODY:**

```
{
"maskGroupId": 20719929,
"maskGroupLabel": "Country",
"tagName": "Country",
"maskGroupShared": true,
"classifierBased": true,
"notes": "Created from classifier import",
"configuration":
[
{
"maskFunctionId": 20719930,
"maskFunctionLabel": "HASHLOV (country.txt)",
"maskFunctionName": "HASHLOV",
"displayName": "Countries",
"maskFunctionParams":
[
{
"pos": 1,
"value": "country.txt"
}
],
"notes": "Countries derived from a hashed index into a lookup-table"
}
]
}
```

Set the mask function group of an attribute for a project version

As a TDE, you can set the masking function group for an attribute (column), in a project version.

Syntax

```
POST http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/entities/{entityId}/
attributes/{attributeId}/maskConfigurations
```

Parameters

- **maskFunctionGroupId** (body, JSON)Id of Mask Function Group to assign to an attribute (attributeId).
- **attributeId** (path, long)
- **entityId** (path, long)
- **projectId** (query, long)
- **versionId** (query, long)

Example

```
POST http://host:8443/TDMModelService/api/ca/v1/datamodel/entities/1234/
attributes/91748/maskConfigurations?projectId=1&versionId=1
```

- **BODY TEXT:**

```
{"maskGroupId" : 20719929}
```

- **RESPONSE BODY:**

```
{
"maskGroupId": 20719929,
"maskGroupLabel": "Country",
"tagName": "Country",
"maskGroupShared": true,
"classifierBased": true,
"notes": "Created from classifier import",
"configuration":
[
{
"maskFunctionId": 20719930,
"maskFunctionLabel": "HASHLOV (country.txt)",
"maskFunctionName": "HASHLOV",
"displayName": "Countries",
"maskFunctionParams":
[
{
"pos": 1,
"value": "country.txt"
}
],
"notes": "Countries derived from a hashed index into a lookup-table"
}
]
}
```

Update the mask function group of an attribute, for a project version

As a TDE, you can update the mask function group of an attribute (column), in a project version.

NOTE

If the previous mask function group of the attribute is not from a classifier, not marked as 'shared', and not linked to any other attributes, this API deletes it.

Syntax

```
PATCH http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/entities/{entityId}/
attributes/{attributeId}/maskConfigurations
```

Parameters

- **maskFunctionGroupId** (body, JSON)Id of Mask Function Group with which to update attribute (attributeId).
- **attributeId** (path, long)
- **entityId** (path, long)
- **projectId** (query, long)
- **versionId** (query, long)

Example

```
PATCH http://host:8443/TDMModelService/api/ca/v1/datamodel/entities/1234/
attributes/91748/maskConfigurations?projectId=1&versionId=1
```

- **BODY TEXT:**

```
{"maskGroupId" : 20719929}
```

- **RESPONSE BODY:**

```
{
"maskGroupId": 20719929,
"maskGroupLabel": "Country",
"tagName": "Country",
"maskGroupShared": true,
"classifierBased": true,
"notes": "Created from classifier import",
"configuration":
[
{
"maskFunctionId": 20719930,
"maskFunctionLabel": "HASHLOV (country.txt)",
"maskFunctionName": "HASHLOV",
"displayName": "Countries",
"maskFunctionParams":
[
{
"pos": 1,
"value": "country.txt"
}
],
"notes": "Countries derived from a hashed index into a lookup-table"
}
]
}
```

Remove current mask function group from an attribute, for a project version

As a TDE, you can remove the mask function group that an attribute (column) uses, in a project version.

> **NOTE**
>
> This action has the same effect as changing the mask function group to 'Do Not Mask' for a column in the Configure Data Masking section of the Portal UI.

Syntax

```
DELETE http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/entities/{entityId}/
attributes/{attributeId}/maskConfigurations
```

Parameters

- **attributeId** (path, long)
- **entityId** (path, long)
- **projectId** (query, long)
- **versionId** (query, long)

Example

```
DELETE http://host:8443/TDMModelService/api/ca/v1/datamodel/entities/1234/
attributes/91748/maskConfigurations
```

- **RESPONSE**:

```
true
```

Masking configurations by tag

Get a list of mask function groups and attributes

As a TDE, you can get a hierarchical view on the current masking function groups sorted by tag. Each tag that is in the discovered PII is returned in the top-level object. Nested in this object is the current masking group list and their associated attributes, a list of attributes with this tag assigned but not configured to be masked and a list of all mask function groups associated with this tag.

If all mask function groups for the contained masking group list are the same, the effective mask group name and ID are populated into the top-level object.

Syntax

```
GET http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/tags/maskConfigurations
```

Parameters

- **projectId** (query, long)
- **versionId** (query, long)

Example

```
GET http://host:8443/TDMModelService/api/ca/v1/datamodel/tags/maskConfigurations?
projectId=1&versionId=1
```

- **RESPONSE BODY:**

For brevity, the full response body is not shown.

```
{
"elements":
[
{
"tagId": 5021,
 "tagName": "Country",
 "effectiveMaskGroup": "Country",
 "effectiveMaskGroupID": 5022,
 "maskingGroupCount": 2,
 "currentMaskingGroupList": [ ],
 "unmaskedAttributes": [],
 "knownMaskingGroupsList": []
```

```
},
<further tags listed with the same format>
]
}
```

Get a list of mask function groups and attributes for a tag

As a TDE, you can get a hierarchical view on the current masking function groups for a single tag.

Syntax

```
GET http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/tags/{tagId}/
maskConfigurations
```

Parameters

- **tagId** (path, long)
- **projectId** (query, long)
- **versionId** (query, long)

Example

```
GET http://host:8443/TDMModelService/api/ca/v1/datamodel/tags/5021/maskConfigurations?
projectId=1&versionId=1
```

- **RESPONSE BODY:**

```
{
"tagId": 5021,
"tagName": "Country",
"effectiveMaskGroup": "Country",
"effectiveMaskGroupID": 5022,
"maskingGroupCount": 2,
"currentMaskingGroupList": [ ],
"unmaskedAttributes": [],
"knownMaskingGroupsList": []
}
```

Remove mask function groups for all attributes with a tag

As a TDE, you can remove all masking functions associated with a tag.

Syntax

```
DELETE http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/tags/{tagId}/
maskConfigurations
```

Parameters

- **tagId** (path, long)
- **projectId** (query, long)
- **versionId** (query, long)

Example

```
DELETE http://host:8443/TDMModelService/api/ca/v1/datamodel/tags/5021/
maskConfigurations?projectId=1&versionId=1
```

- **RESPONSE BODY:**

`true` (Response Code 204)

Add existing mask function group to a tag

As a TDE, you can set an existing mask function group to all attributes associated with a tag. This replaces any existing mask function group associated with any of these attributes.

Syntax

```
POST http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/tags/{tagId}/
maskConfigurations
```

Parameters

- **maskFunctionGroupId** (body, maskFunctionGroup)Id of Mask Function Group with which to update tag (tagId), as maskFunctionGroup object.
- **tagId** (path, long)
- **projectId** (query, long)
- **versionId** (query, long)

Example

```
POST http://host:8443/TDMModelService/api/ca/v1/datamodel/tags/5021/maskConfigurations?
projectId=1&versionId=1
```

- **Body text:**

```
{"maskGroupId" : 20719929}
```

- **RESPONSE BODY:**

`true` (Response Code 204)

Add new mask function group to a tag

As a TDE, you can create a new mask function group and assign it to all attributes associated with a tag.

Syntax

```
PUT http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/tags/{tagId}/
maskConfigurations
```

Parameters

- **maskFunctionGroup** (body, maskFunctionGroup)
  Mask Function Group with which to update tag (tagId), as a maskFunctionGroup pbject.
- **tagId** (path, long)
- **projectId** (query, long)
- **versionId** (query, long)

Example

```
PUT http://host:8443/TDMModelService/api/ca/v1/datamodel/tags/5021/maskConfigurations?
projectId=1&versionId=1
```

- **Body text:**

```
{
"maskGroupLabel":"test label",
 "tagName":"Country",
 "maskGroupShared":true,
 "classifierBased":false,
 "configuration":
[
{
"maskFunctionName":"HASHLOV",
 "displayName":"TestMaskFunction",
 "maskFunctionParams":
[
{
"pos":1,
 "value":"uktowns.txt"
}
]
}
]
}
```

- **RESPONSE BODY:**

```
{
"maskGroupId": 20719955,
"maskGroupLabel": "test label",
"tagName": "Country",
"maskGroupShared": true,
"classifierBased": false,
"notes": "",
"configuration":
[
{
"maskFunctionId": 20719930,
"maskFunctionLabel": "HASHLOV (uktowns.txt)",
"maskFunctionName": "HASHLOV",
"displayName": "TestMaskFunction",
"maskFunctionParams":
[
{
"pos": 1,
"value": "uktowns.txt"
}
],
"notes": ""
}
```

```
]
}
```

Remove specific mask function groups from attributes with a specific tag

As a TDE, you can remove specific mask function groups from a tag. Use this API to remove masking for all attributes with the specified tag ID that are masked with the specified mask function group.

Syntax

```
DELETE  http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/tags/{tagId}/
maskConfigurations/{maskFunctionGroupId}
```

Parameters

- **maskFunctionGroupId** (path, long)
  Id of Mask Function Group to remove from attributes associated with tagId.
- **tagId** (path, long)
- **projectId** (query, long)
- **versionId** (query, long)

Example

```
DELETE http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/tags/5021/
maskConfigurations/25001?projectVersion=1&versionId=1
```

- **RESPONSE:**

`true` (Response Code 200)

Get a mask function group

As a TDE, you can get the description of a specific mask function group associated with a tag.

Syntax

```
GET http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/tags/{tagId}/
maskConfigurations/{maskFunctionGroupId}
```

Parameters

- **maskFunctionGroupId** (path, long)
  Id of Mask Function Group for which to retrieve details.
- **tagId** (path, long)
- **projectId** (query, long)
- **versionId** (query, long)

Example

```
GET http://host:8443/TDMModelService/api/ca/v1/datamodel/tags/5021/
maskConfigurations/25001?projectVersion=1&versionId=1
```

- **RESPONSE BODY:**

```
{
"maskGroupId": 25001,
"maskGroupLabel": "test label",
"tagName": "Country",
```

```
"maskGroupShared": true,
"classifierBased": false,
"notes": "",
"configuration":
[
{
"maskFunctionId": 20719930,
"maskFunctionLabel": "HASHLOV (uktowns.txt)",
"maskFunctionName": "HASHLOV",
"displayName": "TestMaskFunction",
"maskFunctionParams":
[
{
"pos": 1,
"value": "uktowns.txt"
}
],
"notes": ""
}
]
}
```

Make changes to a mask function group

As a TDE, you can update a mask function group associated with a tag.

Syntax

```
PATCH http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/tags/{tagId}/
maskConfigurations/{maskFunctionGroupId}
```

Parameters

- **maskFunctionGroupId** (path, long)
  Id of Mask Function Group to which to make changes based on maskFunctionGroup object.
- **maskFunctionGroup**(body, object)
  A maskFunctionGroup object (can be incomplete). CA TDM amends attributes of the Mask Function Group defined by maskFunctionGroupId, with whichever attributes you supply in this object.
- **tagId** (path, long)
- **projectId** (query, long)
- **versionId** (query, long)

Example

```
PATCH http://host:8443/TDMModelService/api/ca/v1/datamodel/tags/5021/
maskConfigurations/25001?projectVersion=1&versionId=1
```

- **BODY TEXT:**

```
{"maskGroupLabel" : "new label for function"}
```

- **RESPONSE BODY:**

```
{
"maskGroupId": 25001,
"maskGroupLabel": "new label for function",
"tagName": "Country",
"maskGroupShared": true,
"classifierBased": false,
"notes": "",
"configuration":
[
{
"maskFunctionId": 20719930,
"maskFunctionLabel": "HASHLOV (uktowns.txt)",
"maskFunctionName": "HASHLOV",
"displayName": "TestMaskFunction",
"maskFunctionParams":
[
{
"pos": 1,
"value": "uktowns.txt"
}
],
"notes": ""
}
]
}
```

- **RESPONSE:**

`true` (Response Code 200)

Replace the mask function group assigned with an existing group

As a TDE, you can replace the mask function group to all attributes associated with a tag and an existing group ID.

Syntax

```
PUT http://<server>:<host>/TDMModelService/api/ca/v1/datamodel/tags/{tagId}/
maskConfigurations/{groupId}
```

Parameters

- **projectId** (query, long)
- **versionId** (query, long)
- **tagId** (path, long)
- **groupId** (path,long)
  Mask function group ID
- **maskFunctionGroup** (body, object)
  The new mask function group object

Example:

```
PUT http://host:8443/TDMModelService/api/ca/v1/datamodel/tags/5021/
maskConfigurations/25001?projectVersion=1&versionId=1
```

- **BODY TEXT:**

```
{
"maskGroupLabel":"new group label",
 "tagName":"Country",
 "maskGroupShared":true,
 "classifierBased":false,
 "configuration":
[
{
"maskFunctionName":"HASHLOV",
 "displayName":"hashlov uk towns",
 "maskFunctionParams":
[
{
"pos":1,
 "value":"uktowns.txt"
}
]
}
]
}
```

- **RESPONSE BODY:**

```
{
"maskGroupId": 25001,
"maskGroupLabel": "new group label",
"tagName": "Country",
"maskGroupShared": true,
"classifierBased": false,
"notes": "",
"configuration":
[
{
"maskFunctionId": 20719930,
"maskFunctionLabel": "HASHLOV (uktowns.txt)",
"maskFunctionName": "HASHLOV",
"displayName": "hashlov uk towns",
"maskFunctionParams":
[
{
"pos": 1,
"value": "uktowns.txt"
}
],
"notes": ""
}
```

```
]
}
```

Retrieve a List of Masking Functions

A Test Data Engineer (TDE) can retrieve a list of available masking functions provided by Fast Data Masker.

Syntax

```
GET https://<server>:<host>/TDMMaskingService/api/ca/v1/masking/functions
```

Parameters

- (Optional) **dataType**
  Filters the masking functions based on data type.
  Values: char, number, date, char_date
- (Optional) **functionName**
  Filters the masking functions based on full or partial function names.
  For example, 'ACCT' will match function name 'ACCT_01'
- Pagination parameters.

Example

```
GET https://host:8443/TDMMaskingService/api/ca/v1/masking/functions?page=0&size=2
```

- **RESPONSE:**
  200 OK
- **RESPONSE BODY:**

```
{
  "elements":{
              "ACCT_01":{
                         "id":"1",
                         "name":"ACCT_01",
                         "description":"ACCT_01 - replace digits 0..9 in original",
                         "parm1":"Digits to replace",
                         "parm2":"",
                         "parm3":"",
                         "parm4":"",
                         "char":"true",
                         "number":"true",
                         "date":"false",
                         "char_date":"false"
                        },
               "ADD":{
                        "id":"2",
                        "name":"ADD",
                        "description":"ADD - Add a fixed value",
                        "parm1":"Fixed Value",
                        "parm2":"",
                        "parm3":"",
                        "parm4":"",
```

```
                        "char":"true",
                        "number":"true",
                        "date":"true",
                        "char_date":"true"
                      }
            },
    "numberOfElements":2,
    "totalElements":101
  }
```

Retrieve Details of a Masking Function

A Test Data Engineer (TDE) can retrieve details about a specific masking function.

Syntax

```
GET https://<server>:<host>/TDMMaskingService/api/ca/v1/masking/functions/{functionId}
```

Parameters

- **functionId**

Example

```
GET https://host:8443/TDMMaskingService/api/ca/v1/masking/functions/1
```

- **RESPONSE:**
  200 OK
- **RESPONSE BODY:**

```
 "ACCT_01":{
              "id":"1",
              "name":"ACCT_01",
              "description":"ACCT_01 - replace digits 0..9 in original",
              "parm1":"Digits to replace",
              "parm2":"",
              "parm3":"",
              "parm4":"",
              "char":"true",
              "number":"true",
              "date":"false",
              "char_date":"false"
            },
```

Retrieve a List of Masking Seed Lists

As a Test Data Engineer (TDE), you can retrieve a list of masking seed lists available. This helps you in choosing the appropriate seed list for the masking function to be used.

Syntax

```
GET https://<server>:<host>/TDMMaskingService/api/ca/v1/masking/seedlists
```

Parameters

- Pagination parameters

Example

```
GET https://host:8443/TDMMaskingService/api/ca/v1/masking/seedlists?size=2&page=6
```

- **RESPONSE:**
  200 OK
- **RESPONSE BODY:**

```
{
    "elements": [
        {
            "id": 13,
            "name": "firstnamemaleamerican.txt",
            "description": "American Male First Name"
        },
        {
           "id": 14,
           "name": "firstnames.txt",
           "description": "First Names"
        }
    ],
    "numberOfElements": 2,
    "totalElements": 66
}
```

Retrieve Details of a Seed List

As a Test Data Engineer (TDE), you can retrieve details about a single seed list.

Syntax

```
GET https://<server>:<host>/TDMMaskingService/api/ca/v1/masking/seedlists/{seedlistId}
```

Parameters

- **seedlistId**
  ID of the seedlist for which to return details
- Pagination parameters

Example

```
GET https://host:8443/TDMMaskingService/api/ca/v1/masking/seedlists/13
```

- **RESPONSE:**
  200 OK
- **RESPONSE BODY:**

```
{
    "id": 13,
    "name": "firstnamemaleamerican.txt",
    "description": "American Male First Name"
}
```

Download Masking Audit Files

A Test Data Engineer (TDE) can download masking audit details to see what data has been masked and view samples of the masked data. The masking audit file is downloaded as a zip file and includes the project version, environment Id, when the masking job was started, when the masking job was completed and so on.

Syntax

```
GET https://<server>:<host>/TDMMaskingService/api/ca/v1/masking/jobs/{jobId}/audit
```

Parameters

- **jobId**
  ID of the job for which to return masking audit details

Example

```
GET https://host:8443/TDMMaskingService/api/ca/v1/masking/jobs/1/audit
```

- **RESPONSE:**
  200 OK

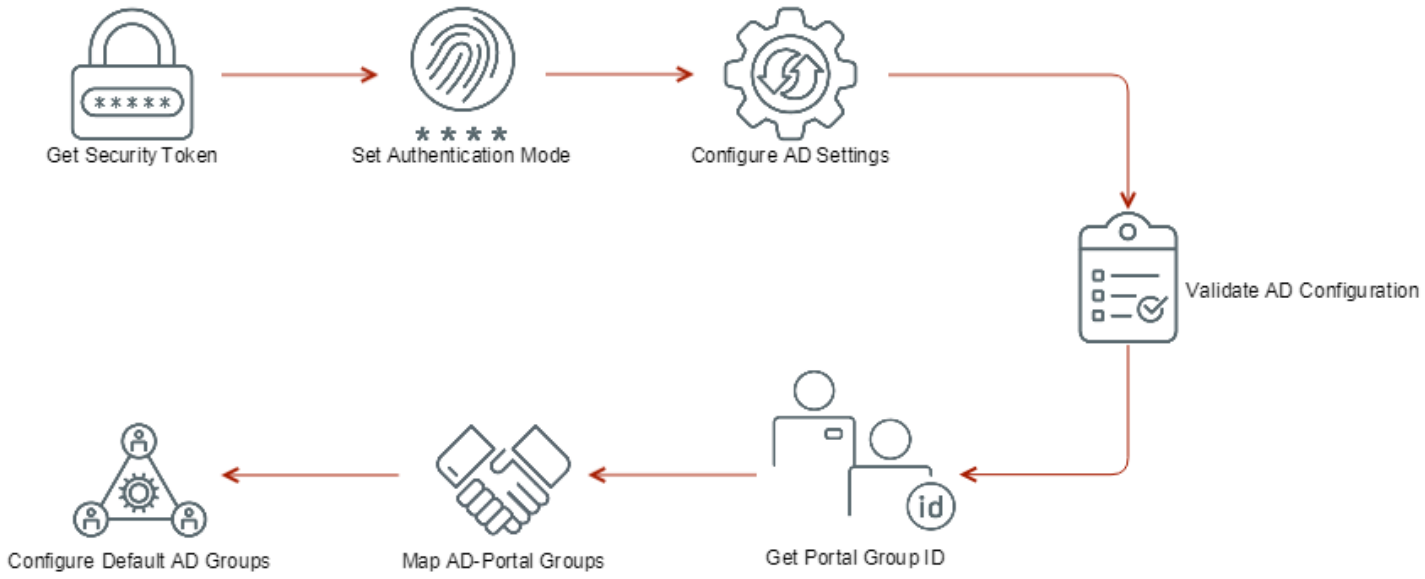
You have downloaded the masking audit file. You can use this to understand the data that is masked and view samples of the masked data.

Use APIs to Integrate Active Directory/LDAP with the CA TDM Portal

This article provides information about how administrators can use APIs to integrate Active Directory (AD)/LDAP with the CA TDM Portal. This page refers to the TestDataManager API Service.

The following diagram shows the overall process:

Figure 59: Active Directory Integration Using APIs



Perform the following steps:

NOTE

For more information about how to work with AD/LDAP integration in the UI, see LDAP Integration with the CA TDM Portal.

Get a Security Token

To understand how to get a security token, see Use APIs to Prepare Test Data for Non-Relational Sources. After you generate the token, use the same token in all the subsequent operations in this article.

Set the Authentication Mode

To integrate Active Directory with the CA TDM Portal, you must set the authentication mode as LDAP.

Note: For more information about working with this configuration in the UI, see Active Directory Integration with the CA ? TDM Portal.

Follow these steps:

1. Access the following CA TDM Portal API:

   ```
   PUT https://<server>:<host>/TestDataManager/api/ca/v1/settings/security
   ```

   Note: For more information about this API, see the "settings-controller : Settings Controller" section at `https://<server>:<port>/TestDataManager/swagger-ui.html` .
2. Example security token in the **Authorization** field as follows:

   ```
   Bearer eyJhbGciOiJIUzI1NiJ9.eyXX0ifQ.7T1CyH_xQK0vQcBB7dLojUxm8ENTeRRrdOa-RQ5l4Ro
   ```

3. Example information in the **securitySettingsConfiguration** field is as follows:

   ```
   {
    "authenticationMode": "AD/LDAP"
   }
   ```

4. Run the API.
5. Example response body is as follows:

   ```
   {
    "message": "Security settings are configured successfully."
   }
   ```

   The authentication mode is set to LDAP.

Configure the Active Directory Settings

After you set the authentication mode to LDAP, configure the required Active Directory settings in the CA TDM Portal.

Note: For more information about working with this configuration in the UI, see Active Directory Integration with the CA ? TDM Portal.

Follow these steps:

1. Access the following CA TDM Portal API:

   ```
   POST https://<server>:<host>/TestDataManager/api/ca/v1/settings/security/authorities
   ```

Note: For more information about this API, see the "settings-controller : Settings Controller" section at `https://<server>:<port>/TestDataManager/swagger-ui.html`.

2. Example security token in the **Authorization** field as follows:

```
Bearer eyJhbGciOiJIUzI1NiJ9.eyXX0ifQ.7T1CyH_xQK0vQcBB7dLojUxm8ENTeRRrdOa-RQ5l4Ro
```

3. Example information in the **ldapServerProperties** field is as follows:

```
{
 "authorityName": "Default",
 "baseDN": "DC=Server02ad1,DC=ca,DC=com",
 "globalTDMGroup": null,
 "groupAttributes": {
 "groupIdAttribute": "cn",
 "groupMemberAttribute": "member",
 "groupObjectClass": "group",
 "groupOrganization": "cn=Users"
 },
 "hostName": "Server02ad1",
 "ldapAdvanceConfiguration": {
 "referralStrategy": "FOLLOW"
 },
 "password": "Abc@123",
 "port": "389",
 "tlsAttributes": {
 "useTLS": "false"
 },
 "userAttributes": {
 "userIdAttribute": "cn",
 "userObjectClass": "person",
 "userOrganization": "cn=Users"
 },
 "userDN": "CN=administrator,CN=Users,DC=Server02ad1,DC=ca,DC=com"
}
```

Note: Ensure that the value of the `authorityName` parameter is set to `Default`.

4. Run the API.
5. Example response is as follows:

```
{
 "authorityName": "Default",
 "hostName": "Server02ad1",
 "port": "389",
 "userDN": "CN=administrator,CN=Users,DC=Server02ad1,DC=ca,DC=com",
 "password": null,
 "baseDN": "DC=Server02ad1,DC=ca,DC=com",
 "globalTDMGroup": null,
```

```
 "tlsAttributes": null,
 "userAttributes": {
 "userObjectClass": "person",
 "userIdAttribute": "cn",
 "userOrganization": "CN=Users"
 },
 "groupAttributes": {
 "groupObjectClass": "group",
 "groupIdAttribute": "cn",
 "groupOrganization": "CN=Users",
 "groupMemberAttribute": "member"
 },
 "ldapAdvanceConfiguration": {"referralStrategy": "FOLLOW"},
 "message": "LDAP server settings are configured successfully.",
 "updtTime": 1501761545477
}
```


The Active Directory settings are configured.

Validate the Active Directory Configuration

After you configure the Active Directory settings, validate whether the configurations are correct and work without any issue.

Note: For more information about working with this configuration in the UI, see Active Directory Integration with the CA ? TDM Portal.

Follow these steps:

1. Access the following CA TDM Portal API:

   ```
   POST https://<server>:<host>/TestDataManager/api/ca/v1/settings/security/authorities/{authorityName}/validate
   ```

   Note: For more information about this API, see the "settings-controller : Settings Controller" section at `https://<server>:<port>/TestDataManager/swagger-ui.html` .
2. Example security token in the **Authorization** field as follows:

   ```
   Bearer eyJhbGciOiJIUzI1NiJ9.eyXX0ifQ.7T1CyH_xQK0vQcBB7dLojUxm8ENTeRRrdOa-RQ5l4Ro
   ```

3. Example information in the **ldapServerProperties** field is as follows:

   ```
   {
    "authorityName": "Default",
    "hostName": "Server02ad1",
    "port": "389",
    "userDN": "CN=administrator,CN=Users,DC=Server02ad1,DC=ca,DC=com",
    "password": null,
    "baseDN": "DC=Server02ad1,DC=ca,DC=com",
    "globalTDMGroup": null,
    "tlsAttributes": null,
    "userAttributes": {
   ```

```
 "userObjectClass": "person",
 "userIdAttribute": "cn",
 "userOrganization": "CN=Users"
 },
 "groupAttributes": {
 "groupObjectClass": "group",
 "groupIdAttribute": "cn",
 "groupOrganization": "CN=Users",
 "groupMemberAttribute": "member"
 },
 "ldapAdvanceConfiguration": {
 "referralStrategy": "FOLLOW"
 },
 "message": null,
 "updtTime": 1501761545477
}
```


Note: Ensure that the value of the `authorityName` parameter is set to `Default`.

4. Run the API.
5. Example response is as follows:


```
{
 "authorityName": "Default",
 "hostName": "Server02ad1",
 "port": "389",
 "userDN": "CN=administrator,CN=Users,DC=Server02ad1,DC=ca,DC=com",
 "password": null,
 "baseDN": "DC=Server02ad1,DC=ca,DC=com",
 "globalTDMGroup": null,
 "tlsAttributes": null,
 "userAttributes": {
 "userObjectClass": "person",
 "userIdAttribute": "cn",
 "userOrganization": "CN=Users"
 },
 "groupAttributes": {
 "groupObjectClass": "group",
 "groupIdAttribute": "cn",
 "groupOrganization": "CN=Users",
 "groupMemberAttribute": "member"
 },
 "ldapAdvanceConfiguration": {
 "referralStrategy": "FOLLOW"
 },
 "message": null,
 "updtTime": 1501761545477
}
```

The Active Directory configuration is validated successfully.

Get the CA TDM Portal User Group ID

To map Active Directory groups to the CA TDM Portal user group, you must identify the group ID of the CA TDM Portal user group. Note this ID so that you can use it during the mapping step.

Note: For more information about working with the CA TDM Portal user groups in the UI, see User and Group Management.

Follow these steps:

1. Access the following CA TDM Portal API:
   ```
   GET https://<server>:<host>/TestDataManager/api/ca/v1/groups
   ```
   Note: For more information about this API, see the "security-controller : Interface for Users and Groups Management" section at `https://<server>:<port>/TestDataManager/swagger-ui.html` .
2. Example security token in the **Authorization** field as follows:

   ```
   Bearer eyJhbGciOiJIUzI1NiJ9.eyXX0ifQ.7T1CyH_xQK0vQcBB7dLojUxm8ENTeRRrdOa-RQ5l4Ro
   ```

3. Run the API.
4. Example response is as follows:


```
{
 "numberOfGroups": 15,
 "totalNumberOfGroups": 44,
 "groups": [
 {
 "groupId": 1,
 "groupName": "ADMIN",
 "description": "ADMIN",
 "isAdminGroup": true,
 "projectId": null,
 "adGroup": null,
 "securityFunctions": null,
 "adminGroup": true
 },

...

...

{
 "groupId": 22,
 "groupName": "Admin - StoreFront - Example Project - SQL Server",
 "description": "Administration - StoreFront - Example Project - SQL Server",
 "isAdminGroup": true,
 "projectId": 2234,
 "adGroup": null,
```

```
 "securityFunctions": null,
 "adminGroup": true
 },

...

...

{
 "groupId": 72350,
 "groupName": "Grp_FN",
 "description": "This group is for Finance.",
 "isAdminGroup": false,
 "projectId": 36135,
 "adGroup": null,
 "securityFunctions": null,
 "adminGroup": false
 }

]
}
```

The CA TDM Portal user group (group ID 72350) to which you want to map the Active Directory group is available.

Map Active Directory Groups

After you validate that your Active Directory configuration is valid and working correctly, you can map the required Active Directory group to the CA TDM Portal user group. This mapping allows users who are members of the mapped Active Directory group to log into the CA TDM Portal and get access to all the resources that other users of the mapped CA TDM Portal user group have.

Note: For more information about working with this configuration in the UI, see Create and Edit Projects.

Follow these steps:

1. Access the following CA TDM Portal API:
   `POST https://<server>:<host>/TestDataManager/api/ca/v1/groups/{groupId}/actions/mapExternalGroups`
   Note: For more information about this API, see the "security-controller : Interface for Users and Groups Management" section at `https://<server>:<port>/TestDataManager/swagger-ui.html` .
2. Example security token in the **Authorization** field as follows:

   ```
   Bearer eyJhbGciOiJIUzI1NiJ9.eyXX0ifQ.7T1CyH_xQK0vQcBB7dLojUxm8ENTeRRrdOa-RQ5l4Ro
   ```

3. Example information in the **groupId** field is 72350.
4. Example information in the **externalGroups** field is as follows:

   ```
   [
    {
    "authorityName": "Default",
    "name": "GRP2_AD2"
   ```

```
  }
 ]
```

5. Run the API.
6. Example response is as follows:

```
{"response": "Successfully mapped external groups."}
```

The Active Directory group (GRP2_AD2) is mapped to the CA TDM Portal user group (group ID 72350).

Configure Default Active Directory Groups

You can also configure default Active Directory groups by mapping them to the default CA TDM Portal user groups (ADMIN and TESTER). With this mapping, whenever a new project is created in the CA TDM Portal, default Active Directory groups are also created in addition to the usual default CA TDM Portal user groups.

Note: For more information about working with this configuration in the UI, see Active Directory Integration with the CA ? TDM Portal.

Follow these steps:

1. Access the following CA TDM Portal API:

   ```
   POST https://<server>:<host>/TestDataManager/api/ca/v1/settings/security/actions/mapDefaultExternalGroups
   ```

   Note: For more information about this API, see the "settings-controller : Settings Controller" section at `https://<server>:<port>/TestDataManager/swagger-ui.html` .
2. Example security token in the **Authorization** field as follows:

   ```
   Bearer eyJhbGciOiJIUzI1NiJ9.eyXX0ifQ.7T1CyH_xQK0vQcBB7dLojUxm8ENTeRRrdOa-RQ5l4Ro
   ```

3. Example information in the **groupId** field is 72350.
4. Example information in the **externalGroups** field is as follows:

```
{
 "adminGroups": [
 {
 "authorityName": "Default",
 "name": "GRP2_AD3"
 }
 ],
 "testerGroups": [
 {
 "authorityName": "Default",
 "name": "GRP2_AD4"
 }
 ]
}
```

5. Run the API.
6. Example response is as follows:

```
{

"message": "Successfully mapped default external (LDAP) groups."

}
```


The default Active Directory groups are mapped to the default CA TDM Portal user groups (ADMIN and TESTER).

You have successfully integrated Active Directory with the CA TDM Portal by using APIs.

API Services reference

Swagger documentation for the following API services are available:

- TDMConnectionProfileService
- TDMDataReservationService
- TDMGeneratorService
- TDMJobService
- TDMMaskingService
- TDMModelService
- TDMProjectService
- TDMvDataService
- TestDataManager

TDMConnectionProfileService

alpha


```
{"swagger":"2.0","info":{"description":"This is an API to allow management of CA TDM connection
 profiles. Connection Profiles define connections available to external data sources and destinations,
 which can be used by CA Test Data Manager. ","version":"1.0","title":"CA TDM Connection Profile
 API","termsOfService":"http://ca.com","contact":{"name":"CA Technologies"},"license":{"name":"The
 CA License Version 2.0","url":"https://ca.com/LICENSE"}},"host":"vtdm-dev-demo:8443","basePath":"/
TDMConnectionProfileService","tags":[{"name":"con-profile-controller","description":"Interface for connection
 profiles"},{"name":"database-metadata-controller","description":"Interface for Database Metadata"}],"paths":
{"/api/ca/v1/connectionProfiles":{"get":{"tags":["con-profile-controller"],"summary":"Interface
 for getting all connection profiles","description":"Use this interface to retrieve the details of
 all connection profiles.","operationId":"getAuthUserProfilesUsingGET","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"groupOnly","in":"query","description":"Set this parameter to true to return group connection
 profiles only","required":false,"type":"boolean"}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/ProfileSet"}},"401":{"description":"Server authentication failed.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["con-profile-controller"],"summary":"Interface
 for creating a new connection profile","description":"Use this interface to create a new connection
 profile.","operationId":"createAuthUserProfileUsingPOST","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
```

authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"profile","description":"Connection profile details using which you want to create a new connection profile.","required":true,"schema":{"$ref":"#/definitions/ConnectionProfile"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/ConnectionProfile"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/connectionProfiles/{profileName}":{"get":{"tags":["con-profile-controller"],"summary":"Interface for getting connection profile details","description":"Use this interface to retrieve the details of a connection profile.","operationId":"getConnectionProfileUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"profileName","in":"path","description":"Name of the connection profile for which you want to use to get the details.","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/ConnectionProfile"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":{"tags":["con-profile-controller"],"summary":"Interface for updating a connection profile","description":"Use this interface to update the details of a connection profile.","operationId":"updateAuthUserProfileUsingPUT","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"profileName","in":"path","description":"Name of the connection profile that you want to update.","required":true,"type":"string"},{"in":"body","name":"profile","description":"Connection profile details using which you want to update an existing connection profile.","required":true,"schema":{"$ref":"#/definitions/ConnectionProfile"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/ConnectionProfile"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["con-profile-controller"],"summary":"Interface for deleting a connection profile","description":"Use this interface to delete a connection profile.","operationId":"deleteAuthUserProfileUsingDELETE","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"profileName","in":"path","description":"Name of the connection profile that you want to delete.","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":{"type":"object"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/

definitions/ErrorResponse"}}}}},"/api/ca/v1/connectionProfiles/{profileName}/actions/getGrantedUserGroups":
{"get":{"tags":["con-profile-controller"],"summary":"Interface to get user groups which were grant access
 to the connection profile.","description":"Use this interface to get the user groups which has been granted
 access to use an existing connection profile.","operationId":"getGrantedUserGroupsUsingGET","consumes":
["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"profileName","in":"path","description":"Name of the connection profile that you want to
 validate.","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":
{"type":"object"}},"400":{"description":"Bad Request - Specific reason is included in the error
 message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication
 failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific
 reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/
api/ca/v1/connectionProfiles/{profileName}/actions/grantAccess":{"post":{"tags":["con-profile-
controller"],"summary":"Interface to grant access to the connection profile to the specified user
 groups.","description":"Use this interface to grant access to use an existing connection profile to
 the specified user groups.","operationId":"grantAccessToProfileUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"profileName","in":"path","description":"Name of the connection profile that you want to
 validate.","required":true,"type":"string"},{"in":"body","name":"groups","description":"List of user groups
 to whom the access needs to be granted.","required":true,"schema":{"type":"array","items":{"$ref":"#/
definitions/GroupDTO"}}}],"responses":{"200":{"description":"Success.","schema":{"type":"object"}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the
 error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":{"tags":["con-profile-
controller"],"summary":"Interface to grant access to the connection profile to the specified user
 groups.","description":"Use this interface to grant access to use an existing connection profile to
 the specified user groups.","operationId":"grantAccessToProfileUsingPUT","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"profileName","in":"path","description":"Name of the connection profile that you want to
 validate.","required":true,"type":"string"},{"in":"body","name":"groups","description":"List of user
 groups to whom the access needs to be granted.","required":true,"schema":{"type":"array","items":
{"$ref":"#/definitions/GroupDTO"}}}],"responses":{"200":{"description":"Success.","schema":
{"type":"object"}},"400":{"description":"Bad Request - Specific reason is included in the error
 message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication
 failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific
 reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/
api/ca/v1/connectionProfiles/{profileName}/actions/revokeAccess":{"post":{"tags":["con-profile-
controller"],"summary":"Interface to revoke access to the connection profile from the specified user
 groups","description":"Use this interface to revoke access to an existing connection profile to the
 specified user groups.","operationId":"revokeAccessToProfileUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /

user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"profileName","in":"path","description":"Name of the connection profile that you want to
 validate.","required":true,"type":"string"},{"in":"body","name":"groups","description":"List of user groups
 to whom the access needs to be revoked.","required":true,"schema":{"type":"array","items":{"$ref":"#/
definitions/GroupDTO"}}}],"responses":{"200":{"description":"Success.","schema":{"type":"object"}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error
 message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/connectionProfiles/{profileName}/
actions/validate":{"post":{"tags":["con-profile-controller"],"summary":"Interface to validate the details
 of an existing connection profile","description":"Use this interface to validate the details of an existing
 connection profile.","operationId":"validateConnectionUsingPOST","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"profileName","in":"path","description":"Name
 of the connection profile that you want to validate.","required":true,"type":"string"}],"responses":{"200":
{"description":"Success.","schema":{"type":"object"}},"400":{"description":"Bad Request - Specific reason is
 included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server
 authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found -
 Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":
{"description":"Conflict - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}},"put":{"tags":["con-profile-controller"],"summary":"Interface to validate the details of
 an existing connection profile","description":"Use this interface to validate the details of an existing
 connection profile.","operationId":"validateConnectionUsingPUT","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"profileName","in":"path","description":"Name
 of the connection profile that you want to validate.","required":true,"type":"string"}],"responses":
{"200":{"description":"Success.","schema":{"type":"object"}},"400":{"description":"Bad Request - Specific
 reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"409":{"description":"Conflict - Specific reason is included in the
 error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/connectionProfiles/
{profileName}/schemas":{"get":{"tags":["database-metadata-controller"],"summary":"Interface for getting
 schemas associated with a connection profile","description":"Use this interface to retrieve the list of
 schemas for a given connection profile.","operationId":"getSchemasUsingGET","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"profileName","in":"path","description":"Name of the connection profile for which you want to
 retrieve schemas.","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":
{"type":"array","items":{"type":"string"}}},"401":{"description":"Server authentication
 failed.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}}},"definitions":{"ConnectionProfile":

{"type":"object","required":["dbType","name","password","server","username"],"properties":
{"additionalConnectionProperties":{"type":"string","description":"JDBC connection string properties.
Applicable only for database type db2/400 sql"},"created":{"type":"string","format":"date-
time","description":"Creation date"},"createdBy":{"type":"integer","format":"int64","description":"Created
by"},"database":{"type":"string","description":"Database name"},"datasourceDriver":
{"type":"string","description":"DataSource Driver"},"datasourceUrl":{"type":"string","description":"DataSource
URL"},"dbType":{"type":"string","description":"Type of database","enum":["sql
server","oracle","mysql","sybase","teradata","db2","db2/400 sql"]},"description":
{"type":"string","description":"Descriptive text"},"instance":{"type":"string","description":"Sql
server instance name"},"integratedSecurity":{"type":"boolean","example":false,"description":"Use
Integrated Security for authentication. Applicable only for database type SQL
Server"},"modified":{"type":"string","format":"date-time","description":"Last modified
date"},"name":{"type":"string","description":"Name of the connection profile"},"password":
{"type":"string","description":"Password"},"port":{"type":"string","description":"Database
server port"},"schema":{"type":"string","description":"Sql server schema name"},"server":
{"type":"string","description":"Database server hostname"},"service":{"type":"string","description":"Oracle
service name"},"username":{"type":"string","description":"Username"}}},"ErrorResponse":
{"type":"object","properties":{"errorCode":{"type":"string"},"errorDetail":{"type":"string"},"errorMsg":
{"type":"string"},"status":{"type":"integer","format":"int32"},"timestamp":{"type":"string"}}},"GroupDTO":
{"type":"object","properties":{"adGroup":{"type":"string","description":"Name of the AD
group associated with this group"},"description":{"type":"string","description":"Description
of the Group"},"groupId":{"type":"integer","format":"int64","description":"ID of the
group","readOnly":true},"groupName":{"type":"string","description":"Name of the Group"},"isAdminGroup":
{"type":"boolean","example":false,"description":"Flage to identify whether this group is and admin
group or not"},"projectId":{"type":"integer","format":"int64","description":"Id of the project
associated with group"},"securityFunctions":{"type":"object","description":"Map of security
functions available and their flags whether they are enabled or not","additionalProperties":
{"type":"boolean"}}}},"ProfileSet":{"type":"object","properties":{"profiles":{"type":"array","items":
{"$ref":"#/definitions/ConnectionProfile"}}}}}}

TDMFindReserveService

none

{"swagger":"2.0","info":{"description":"This section includes the APIs that perform various operations for
data reservation.It also provides the REST API URL for the respective operation along with sample request and
response body content.","version":"1.0","title":"CA TDM Find Reserve Service API","termsOfService":"http://
ca.com","contact":{"name":"CA Technologies"},"license":{"name":"The CA License Version 2.0","url":"https://
ca.com/LICENSE"}},"host":"far-demo.dhcp.broadcom.net:8443","basePath":"/TDMFindReserveService","tags":
[{"name":"data-reservation-controller","description":"Data Reservation Controller"},{"name":"test-data-
model-controller","description":"Test Data Model Controller"},{"name":"find-controller","description":"Find
Controller"},{"name":"data-view-controller","description":"Data View Controller"},{"name":"data-
view-instance-controller","description":"Data View Instance Controller"},{"name":"reservation-table-
controller","description":"Reservation Table Controller"}],"paths":{"/api/ca/v1/dataViewInstances/syncTasks/
actions/clearInitialDelays":{"post":{"tags":["data-view-instance-controller"],"summary":"Interface
for clearing initial synchronization delay","description":"Use this interface for clearing Data View
Instance initial synchronization delay and triggering immediate synchronization. The initial delay is
configured during Data View import.","operationId":"clearInitialDelaysUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through

this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"projectId","in":"query","description":"projectId","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"versionId","required":true,"type":"integer","format":"int64"}],"respons
{"200":{"description":"Success."},"201":{"description":"Created"},"401":{"description":"Server authentication
failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":
{"description":"Not Found"},"500":{"description":"Internal Server Error - Specific reason is included in
the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/dataViewInstances/
syncTasks/actions/startSync":{"post":{"tags":["data-view-instance-controller"],"summary":"Interface
for triggering synchronization","description":"Use this interface to trigger immediate Data
View Instance synchronization.","operationId":"startSyncUsingPOST_1","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"projectId","in":"query","description":"projectId","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"versionId","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"startSyncRequestDto","description":"Request body that includes parameters to trigger
synchronization.","required":true,"schema":{"$ref":"#/definitions/StartSyncRequestDto"}}],"responses":
{"200":{"description":"Success.","schema":{"$ref":"#/definitions/StartSyncResponseDto"}},"201":
{"description":"Created"},"400":{"description":"Bad Request - Specific reason is included in the error
message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication
failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":
{"description":"Not Found"},"500":{"description":"Internal Server Error - Specific reason is included
in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/dataViews":
{"get":{"tags":["data-view-controller"],"summary":"Interface for finding Data Views","description":"Use
this interface to find Data Views. Data Views represent synchronized tables with their attributes
regardless of environment","operationId":"findDataViewsUsingGET","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"projectId","in":"query","description":"projectId","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"versionId","required":true,"type":"integer","format":"int64"},
{"name":"profileName","in":"query","description":"profileName","required":true,"type":"string"}],"responses":
{"200":{"description":"Success.","schema":{"type":"array","items":{"$ref":"#/definitions/
DataViewDto"}}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not
Found"},"500":{"description":"Internal Server Error - Specific reason is included in the error
message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/reservationTables":
{"post":{"tags":["reservation-table-controller"],"summary":"Interface for creating reservation
table","description":"Use this interface to create reservation table necessary for data prefetch
OFF","operationId":"createReservationTableUsingPOST","consumes":["application/json"],"produces":["*/
*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to
perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
authorization scheme to access any protected resource through this API on behalf of the user. For Example:
Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Id
of the project that you want to use to create table","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Id of the project version that you want
to use to use to create table.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"createReservationTableDto","description":"createReservationTableDto","required":true,"schema":

{"$ref":"#/definitions/CreateReservationTableDto"}}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/ReservedRecordsResult"}},"201":{"description":"Created"},"400":{"description":"Bad
 Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found"},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/reservationTables/actions/downloadSql":{"get":
{"tags":["reservation-table-controller"],"summary":"Interface for downloading create reservation
 table SQL","description":"Use this interface to download SQL file with DDL for creating reservation
 table.","operationId":"downloadReservationTableSqlUsingGET","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Id
 of the project that you want to use to generate SQL","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Id of the project version that you
 want to use to generate SQL.","required":true,"type":"integer","format":"int64"},
{"name":"environmentId","in":"query","description":"Id of environment for which
 SQL should be generated.","required":true,"type":"integer","format":"int64"},
{"name":"dataSource","in":"query","description":"Data source where root entity is
 located.","required":true,"type":"string"},{"name":"schema","in":"query","description":"Schema where root
 entity is located.","required":true,"type":"string"},{"name":"entity","in":"query","description":"Name of
 root entity.","required":true,"type":"string"},{"name":"reservationSchema","in":"query","description":"Schema
 where reservation entity will be located.","required":true,"type":"string"},
{"name":"reservationEntity","in":"query","description":"Name of reservation
 entity.","required":true,"type":"string"},{"name":"preview","in":"query","description":"Preview mode (true/
false). When true, the sql is for preview only, not for
 execution.","required":true,"type":"boolean","default":false}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/ReservedRecordsResult"}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not
 Found"},"500":{"description":"Internal Server Error - Specific reason is included in the error
 message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/reservationTables/actions/
validate":{"post":{"tags":["reservation-table-controller"],"summary":"Interface for validating
 reservation table","description":"Use this interface to validate existence of reservation table in
 environment","operationId":"validateReservationTableUsingPOST","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Id of
 the project that you want to use to perform validation","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Id of the project version that you want
 to use to use to perform validation.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"validateReservationTableDto","description":"validateReservationTableDto","required":true,"schema":
{"$ref":"#/definitions/ValidateReservationTableDto"}}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/ReservedRecordsResult"}},"201":{"description":"Created"},"400":{"description":"Bad
 Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not
 Found"},"500":{"description":"Internal Server Error - Specific reason is included in the

```
 error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/reservations/
{reservationId}":{"get":{"tags":["data-reservation-controller"],"summary":"Interface for getting
 a reservation","description":"Use this interface to get a specific reservation and its associated
 resources.","operationId":"getReservationUsingGET","consumes":["application/json"],"produces":["*/
*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to
 perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Id of
 the project that associates the reservation to get.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Id of the project version that
 associates the reservation to get.","required":true,"type":"integer","format":"int64"},
{"name":"reservationId","in":"path","description":"Id of the reservation for which the details are to
 get.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/TestDataReservationDto"}},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/reservations/{reservationId}/
reservedData/actions/export":{"get":{"tags":["data-reservation-controller"],"summary":"Interface
 for exporting reserved records","description":"Use this interface to export reserved records as
 CSV.","operationId":"exportReservedDataUsingGET","consumes":["application/json"],"produces":["*/
*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to
 perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Id of
 the project that you want to use to export the data.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Id of the project version that you
 want to use to export the data.","required":true,"type":"integer","format":"int64"},
{"name":"withRelatedTables","in":"query","description":"Whether data from related tables
 should be included in export.","required":true,"type":"boolean","default":false},
{"name":"reservationId","in":"path","description":"Id of the reservation for which the details
 are to be exported.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/ReservedRecordsResult"}},"400":{"description":"Bad
 Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific
 reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/reservations/{reservationId}/reservedData/actions/
fetch":{"post":{"tags":["data-reservation-controller"],"summary":"Interface for fetching reserved
 records","description":"Use this interface to fetch reserved records. Attributes from multiple Test
 Data Model entities can be selected. Custom order can be defined instead of default order by primary
 key.","operationId":"fetchReservedDataUsingPOST","consumes":["application/json"],"produces":["*/
*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to
 perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Id of
 the project that you want to use to find the data.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Id of the project version that you
 want to use to find the data.","required":true,"type":"integer","format":"int64"},
{"name":"reservationId","in":"path","description":"Id of the reservation for which
 the details are to get.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"requestBody","description":"Request body that includes parameters to select attributes.
```

For more information about parameters in Model Schema, click Model.","required":true,"schema":{"$ref":"#/definitions/FetchReservedRecordsRequest"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/ReservedRecordsResult"}},"201":{"description":"Created"},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/testDataModels/{testDataModelId}/actions/find":{"post":{"tags":["find-controller"],"summary":"Interface for finding test data","description":"Use this interface to find data. Attributes from multiple Test Data Model entities can be selected or used for filtering. Custom order can be defined instead of default order by primary key.","operationId":"findTestDataUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"testDataModelId","in":"path","description":"Id of the test data model that you want to use to find the data.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"Id of the project that you want to use to find the data.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"Id of the project version that you want to use to find the data.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"requestBody","description":"Request body that includes parameters to find the test data. For more information about parameters in Model Schema, click Model.","required":true,"schema":{"$ref":"#/definitions/FindTestDataRequest"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/FindTestDataResult"}},"201":{"description":"Created"},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/testDataModels/{testDataModelId}/actions/findAttributeValues":{"post":{"tags":["find-controller"],"summary":"Interface for finding attribute values","description":"Use this interface to find values for attribute","operationId":"findAttributeValuesUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"testDataModelId","in":"path","description":"Id of the test data model that you want to use to find attribute values.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"Id of the project that you want to use to find attribute values.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"Id of the project version that you want to use to find attribute values.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"requestBody","description":"Request body that includes parameters to find attribute values.","required":true,"schema":{"$ref":"#/definitions/FindAttributeValuesRequest"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/FindAttributeValuesResult"}},"201":{"description":"Created"},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":

{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error -
Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}}},"definitions":{"CreateReservationTableDto":{"type":"object","required":
["dataSource","entity","environmentId","reservationEntity","reservationSchema","schema"],"properties":
{"dataSource":{"type":"string","description":"Data source where root entity is
located."},"entity":{"type":"string","description":"Name of root entity."},"environmentId":
{"type":"integer","format":"int64","description":"Id of environment where reservation
table should be created."},"reservationEntity":{"type":"string","description":"Name of
reservation entity."},"reservationSchema":{"type":"string","description":"Schema where
reservation entity will be located."},"schema":{"type":"string","description":"Schema
where root entity is located."}}},"DataViewDto":{"type":"object","properties":{"createdAt":
{"type":"string","example":"yyyy-MM-dd'T'HH:mm:ss.SSSZ"},"dataSource":{"type":"string"},"id":
{"type":"string"},"name":{"type":"string"},"profileName":{"type":"string"},"projectId":
{"type":"integer","format":"int64"},"properties":{"type":"array","items":{"$ref":"#/
definitions/DataViewPropertyDto"}},"schema":{"type":"string"},"sourceTable":
{"type":"string"},"updatedAt":{"type":"string","example":"yyyy-MM-dd'T'HH:mm:ss.SSSZ"},"versionId":
{"type":"integer","format":"int64"}}},"DataViewInstanceDto":{"type":"object","properties":
{"createdAt":{"type":"string","example":"yyyy-MM-dd'T'HH:mm:ss.SSSZ"},"dataPrefetch":
{"type":"string","enum":["OFF","ON_DEMAND","PERIODIC"]},"dataTableName":{"type":"string"},"dataViewId":
{"type":"string"},"id":{"type":"string"},"pendingDelete":{"type":"boolean"},"profileName":
{"type":"string"},"schema":{"type":"string"},"updatedAt":{"type":"string","example":"yyyy-
MM-dd'T'HH:mm:ss.SSSZ"}}},"DataViewPropertyDto":{"type":"object","properties":
{"createdAt":{"type":"string","example":"yyyy-MM-dd'T'HH:mm:ss.SSSZ"},"dataViewId":
{"type":"string"},"id":{"type":"string"},"keySeq":{"type":"integer","format":"int32"},"name":
{"type":"string"},"primaryKey":{"type":"boolean"},"propertyType":{"type":"string","enum":
["BOOLEAN","STRING","BINARY","NUMBER","IEEE_754_NUMBER","DATE","TIME","TIMESTAMP","TIMESTAMP_TZ"]},"sourceColumnName":
{"type":"string"},"sourceColumnType":{"type":"string"},"sourcePrecision":
{"type":"integer","format":"int32"},"sourceScale":{"type":"integer","format":"int32"},"targetColumnName":
{"type":"string"},"targetPrecision":{"type":"integer","format":"int32"},"targetScale":
{"type":"integer","format":"int32"},"updatedAt":{"type":"string","example":"yyyy-MM-
dd'T'HH:mm:ss.SSSZ"}}},"DeleteDataViewInstanceResultDto":{"type":"object","properties":
{"errorMsg":{"type":"string"},"id":{"type":"string"},"status":{"type":"string","enum":
["SUCCESS","ERROR"]}}},"DeleteDataViewInstancesRequest":{"type":"object","properties":{"ids":
{"type":"array","items":{"type":"string"}}}},"DeleteDataViewInstancesResultDto":{"type":"object","properties":
{"items":{"type":"array","items":{"$ref":"#/definitions/DeleteDataViewInstanceResultDto"}}}},"ErrorResponse":
{"type":"object","properties":{"errorCode":{"type":"string"},"errorDetail":{"type":"string"},"errorMsg":
{"type":"string"},"status":{"type":"integer","format":"int32"},"timestamp":{"type":"string","example":"yyyy-
MM-dd'T'HH:mm:ss.SSSZ"}}},"FetchReservedRecordsRequest":{"type":"object","properties":{"attributes":
{"type":"array","description":"List of selected attributes to return. Model keys are always returned.
If not provided, all attributes from root entity will be returned.","items":{"$ref":"#/definitions/
FindTestDataAttribute"}},"orderBys":{"type":"array","description":"List of order by definitions.","items":
{"$ref":"#/definitions/OrderByAttribute"}},"page":{"type":"integer","format":"int32"},"size":
{"type":"integer","format":"int32"}}},"FindAttributeValuesRequest":{"type":"object","required":
["attribute"],"properties":{"attribute":{"description":"Attribute to return.","$ref":"#/
definitions/FindTestDataAttribute"},"direction":{"type":"string","description":"Order by
direction (ASC, DESC). When not provided, ASC is assumed.","enum":["ASC","DESC"]},"environment":
{"type":"string","description":"Name of the environment. Either environmentId or environment must be
present"},"environmentId":{"type":"integer","format":"int64","description":"Id of the environment.
Either environmentId or environment must be present"},"filters":{"type":"array","description":"List
of filters for finding the data.","items":{"$ref":"#/definitions/FindTestDataFilter"}},"page":
{"type":"integer","format":"int32"},"size":{"type":"integer","format":"int32"}}},"FindAttributeValuesResult":
{"type":"object","required":["page","records","size","totalCount"],"properties":
{"page":{"type":"integer","format":"int32","description":"Number of page that is

```
 being returned."},"records":{"type":"array","description":"Found records.","items":
{"type":"string"}},"size":{"type":"integer","format":"int32","description":"Requested page
 size."},"totalCount":{"type":"integer","format":"int64","description":"Total count of
 records."}}},"FindTestDataAttribute":{"type":"object","required":["attributeName"],"properties":
{"attributeName":{"type":"string","description":"Name of the selected attribute."},"dataSource":
{"type":"string","description":"Data source name of entity. Must be provided when entityName and
 schema is not unique in model."},"entityName":{"type":"string","description":"Name of entity
 to filter. Must be provided if attribute name is not unique and there are multiple entities in
 model."},"schema":{"type":"string","description":"Schema name of entity. Must be provided when
 entityName is not unique in model."}}},"FindTestDataAttributeValue":{"type":"object","properties":
{"attributeName":{"type":"string"},"dataSource":{"type":"string"},"entityName":{"type":"string"},"schema":
{"type":"string"},"value":{"type":"string"}}},"FindTestDataFilter":{"type":"object","required":
["attributeName","operator"],"properties":{"attributeName":{"type":"string","description":"Name of the filter
 attribute."},"dataSource":{"type":"string","description":"Data source name of entity. Must be provided
 when entityName and schema is not unique in model."},"entityName":{"type":"string","description":"Name
 of entity to filter. Must be provided if attribute name is not unique and there are multiple entities
 in model."},"operator":{"type":"string","description":"Operator allowed for this filter","enum":
["EQUALS","NOT_EQUAL","LESS_THAN","LESS_THAN_OR_EQUAL_TO","GREATER_THAN","GREATER_THAN_OR_EQUAL_TO","CONTAINS","BETWEEN
{"type":"string","description":"Schema name of entity. Must be provided when entityName is not unique
 in model."},"values":{"type":"array","description":"List of allowed values for the filter. Required if
 operator requires a value.","items":{"type":"string"}}}},"FindTestDataRecord":{"type":"object","required":
["attributes","modelKeys"],"properties":{"attributes":{"type":"array","description":"Requested attributes
 with their values. Attributes can come from any entity in the Find Reserve model.","items":{"$ref":"#/
definitions/FindTestDataAttributeValue"}},"modelKeys":{"type":"object","description":"Model keys are values
 used for the purpose of creating reservation. They are defined on root entity during Test Data Model
 creation","additionalProperties":{"type":"string"}},"reservationId":{"type":"string","description":"Id
 of reservation if this row is reserved. This value is present only if 'showReservedRecords' is
 true."},"reservedByFullName":{"type":"string","description":"Full name of user who reserved this
 row. May be null if user has no full name. This value is present only if 'showReservedRecords' is
 true."},"reservedByUsername":{"type":"string","description":"User name of user who reserved this row. This
 value is present only if 'showReservedRecords' is true."},"reservedDate":{"type":"string","description":"Date
 of reservation in format yyyy-MM-dd'T'HH:mm:ss.SSSZ. This value is present only if
 'showReservedRecords' is true."}}},"FindTestDataRequest":{"type":"object","properties":{"attributes":
{"type":"array","description":"List of selected attributes to return. If not provided, all attributes from
 root entity will be returned.","items":{"$ref":"#/definitions/FindTestDataAttribute"}},"environment":
{"type":"string","description":"Name of the environment. Either environmentId or environment must be
 present"},"environmentId":{"type":"integer","format":"int64","description":"Id of the environment.
 Either environmentId or environment must be present"},"filters":{"type":"array","description":"List
 of filters for finding the data.","items":{"$ref":"#/definitions/FindTestDataFilter"}},"orderBys":
{"type":"array","description":"List of order by definitions.","items":{"$ref":"#/definitions/
OrderByAttribute"}},"page":{"type":"integer","format":"int32"},"showReservedRecords":
{"type":"boolean","example":false,"description":"Show reserved records in find result. When false, reserved
 records will not be returned."},"size":{"type":"integer","format":"int32"}}},"FindTestDataResult":
{"type":"object","required":["page","records","size","totalCount"],"properties":
{"page":{"type":"integer","format":"int32","description":"Number of page that is being
 returned."},"records":{"type":"array","description":"Found records.","items":{"$ref":"#/
definitions/FindTestDataRecord"}},"size":{"type":"integer","format":"int32","description":"Requested
 page size."},"totalCount":{"type":"integer","format":"int64","description":"Total count of
 records."}}},"ImportDataViewRequest":{"type":"object","properties":{"columnNames":{"type":"array","items":
{"type":"string"}},"dataPrefetch":{"type":"string","enum":["OFF","ON_DEMAND","PERIODIC"]},"dataSource":
{"type":"string"},"modelKeys":{"type":"array","items":{"type":"string"}},"profileName":
{"type":"string"},"reservationStorage":{"type":"boolean"},"root":{"type":"boolean"},"schema":
{"type":"string"},"tableName":{"type":"string"}}},"ImportDataViewResultDto":{"type":"object","properties":
```

{"dataView":{"$ref":"#/definitions/DataViewDto"},"dataViewInstance":{"$ref":"#/definitions/
DataViewInstanceDto"},"errorMsg":{"type":"string"},"profileName":{"type":"string"},"schema":
{"type":"string"},"status":{"type":"string","enum":["CREATED","UPDATED","UNCHANGED","ERROR"]},"tableName":
{"type":"string"}}},"ImportDataViewsRequest":{"type":"object","properties":{"dataPrefetch":
{"type":"string","enum":["OFF","ON_DEMAND","PERIODIC"]},"initialSyncDelaySec":
{"type":"integer","format":"int64"},"items":{"type":"array","items":{"$ref":"#/definitions/
ImportDataViewRequest"}},"legacyModelId":{"type":"integer","format":"int64"},"modelCreatedAt":
{"type":"string","example":"yyyy-MM-dd'T'HH:mm:ss.SSSZ"},"modelDescription":{"type":"string"},"modelName":
{"type":"string"},"modelVersion":{"type":"string"}}},"ImportDataViewsResultDto":
{"type":"object","properties":{"items":{"type":"array","items":{"$ref":"#/definitions/
ImportDataViewResultDto"}},"testDataModelId":{"type":"string"}}},"Map«string,string»":
{"type":"object","additionalProperties":{"type":"string"}},"OrderByAttribute":{"type":"object","required":
["attributeName"],"properties":{"attributeName":{"type":"string","description":"Name of
 the order by attribute."},"dataSource":{"type":"string","description":"Data source name of
 entity. Must be provided when entityName and schema is not unique in model."},"direction":
{"type":"string","description":"Order by direction (ASC, DESC). When not provided, ASC is
 assumed.","enum":["ASC","DESC"]},"entityName":{"type":"string","description":"Name of entity. Must
 be provided if attribute name is not unique and there are multiple entities in model."},"schema":
{"type":"string","description":"Schema name of entity. Must be provided when entityName is not unique in
 model."}}},"ReservationSyncItemDto":{"type":"object","properties":{"dviId":{"type":"string"},"environmentId":
{"type":"integer","format":"int64"},"projectId":{"type":"integer","format":"int64"},"reservationId":
{"type":"integer","format":"int64"},"testDataModelId":{"type":"integer","format":"int64"},"versionId":
{"type":"integer","format":"int64"}}},"ReservationSyncRequestDto":{"type":"object","properties":
{"items":{"type":"array","items":{"$ref":"#/definitions/ReservationSyncItemDto"}}}},"ReservedRecord":
{"type":"object","required":["attributes"],"properties":{"attributes":{"type":"array","description":"Requested
 attributes with their values. Attributes can come from any entity in the Find Reserve
 model.","items":{"$ref":"#/definitions/FindTestDataAttributeValue"}}}},"ReservedRecordsResult":
{"type":"object","required":["page","records","size","totalCount"],"properties":{"page":
{"type":"integer","format":"int32","description":"Number of page that is being returned."},"records":
{"type":"array","description":"Found records.","items":{"$ref":"#/definitions/ReservedRecord"}},"size":
{"type":"integer","format":"int32","description":"Requested page size."},"totalCount":
{"type":"integer","format":"int64","description":"Total count of records."}}},"StartSyncItemResultDto":
{"type":"object","required":["id"],"properties":{"errorMsg":{"type":"string","description":"Error
 message in case of error status"},"id":{"type":"string","description":"Data View Instance id"},"status":
{"type":"string","description":"Result status","enum":["SUCCESS","ERROR"]}},"description":"Represents
 result of synchronization trigger"},"StartSyncRequestDto":{"type":"object","required":
["ids"],"properties":{"ids":{"type":"array","description":"List of Data View Instance ids","items":
{"type":"string"}}}},"StartSyncResponseDto":{"type":"object","required":["items"],"properties":
{"items":{"type":"array","description":"List synchronization trigger results","items":{"$ref":"#/
definitions/StartSyncItemResultDto"}}}},"TestDataModelDto":{"type":"object","properties":
{"createdAt":{"type":"string","example":"yyyy-MM-dd'T'HH:mm:ss.SSSZ"},"description":
{"type":"string"},"id":{"type":"string"},"name":{"type":"string"},"projectId":
{"type":"integer","format":"int64"},"updatedAt":{"type":"string","example":"yyyy-MM-
dd'T'HH:mm:ss.SSSZ"},"versionId":{"type":"integer","format":"int64"}}},"TestDataReservationDto":
{"type":"object","properties":{"expiryDate":{"type":"string","example":"yyyy-MM-dd'T'HH:mm:ss.SSSZ"},"id":
{"type":"string"},"legacyEnvironmentId":{"type":"integer","format":"int64"},"legacyId":
{"type":"integer","format":"int64"},"legacyModelId":{"type":"integer","format":"int64"},"name":
{"type":"string"},"projectId":{"type":"integer","format":"int64"},"releaseDate":
{"type":"string","example":"yyyy-MM-dd'T'HH:mm:ss.SSSZ"},"resErrorMessage":
{"type":"string"},"reservedBy":{"type":"integer","format":"int64"},"resources":
{"type":"array","items":{"$ref":"#/definitions/TestDataReservationResourceDto"}},"scheduledDate":
{"type":"string","example":"yyyy-MM-dd'T'HH:mm:ss.SSSZ"},"state":{"type":"string","enum":
["UNDEFINED","CREATED","STARTED","SUCCESS","FAILED","EXPIRED","INVALID","PURGED"]},"versionId":

{"type":"integer","format":"int64"}}},"TestDataReservationResourceDto":{"type":"object","properties":
{"dataViewInstanceId":{"type":"string"},"modelKeys":{"type":"array","items":{"$ref":"#/
definitions/Map«string,string»"}}}},"ValidateReservationTableDto":{"type":"object","required":
["dataSource","entity","environmentId","schema"],"properties":{"dataSource":
{"type":"string","description":"Data source where reservation entity is located."},"entity":
{"type":"string","description":"Name of reservation entity."},"environmentId":
{"type":"integer","format":"int64","description":"Id of environment where reservation table must
 exist."},"schema":{"type":"string","description":"Schema where reservation entity is located."}}}}}

TDMDataReservationService

alpha

{"swagger":"2.0","info":{"description":"This section includes the APIs that perform various operations
 for data reservation.It also provides the REST API URL for the respective operation along with
 sample request and response body content.","version":"1.0","title":"CA TDM Data Reservation Service
 API","termsOfService":"http://ca.com","contact":{"name":"CA Technologies"},"license":{"name":"The
 CA License Version 2.0","url":"https://ca.com/LICENSE"}},"host":"vtdm-dev-demo:8443","basePath":"/
TDMDataReservationService","tags":[{"name":"field-controller","description":"Interface for
 defining fields in a test data model"},{"name":"find-controller","description":"Interface for
 Find"},{"name":"test-data-model-controller","description":"Interface for Test Data Models"},
{"name":"environment-controller","description":"Interface for environments"},{"name":"data-
reservation-controller","description":"Interface for reservations"},{"name":"association-
controller","description":"Interface for defining associations in a test data model"}],"paths":{"/api/ca/
v1/copyEnvironmentsToVersion":{"post":{"tags":["environment-controller"],"summary":"Interface for copying
 environments between project\\versions.","description":"Use this interface to copy an environment to a
 new project\\version.","operationId":"copyEnvironmentsFromVersionUsingPOST","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"request","description":"request","required":true,"schema":{"$ref":"#/definitions/
CopyEnvironmentsFromRequest"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/
definitions/DataSourceSet"}},"201":{"description":"Created"},"400":{"description":"Bad Request - Request
 does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Resource
 not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error -
 Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/
api/ca/v1/environments":{"get":{"tags":["environment-controller"],"summary":"Interface for getting
 all environments","description":"Use this interface to retrieve the details of all the environments
 related to a project and version.","operationId":"getAllEnvironmentsUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"projectId","in":"query","description":"ID of the project that includes the version for
 which you want to get all the environments.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version for which you
 want to get all the environments.","required":true,"type":"integer","format":"int64"},
{"name":"page","in":"query","description":"Page number that you want to retrieve in a

paginated result. Defaults to 1 if page size is specified. Returns all environments if page and size are empty.","required":false,"type":"integer","format":"int32"},{"name":"size","in":"query","description":"Page size of each page with which you want to retrieve the paginated result. Defaults to 25 if page number is specified. Returns all environments if page and size are empty.","required":false,"type":"integer","format":"int32"},{"name":"searchText","in":"query","description":"Search text that you want to use to perform the search on the environment name and description to get the list of environments.","required":false,"type":"string"},{"name":"sortDir","in":"query","description":"Sorting order that you want to use to sort the paginated environments result. Valid values are ASC and DESC.","required":false,"type":"string"}],"responses":{"200":{"description":"Success.","schema":{"type":"array","items":{"$ref":"#/definitions/EnvironmentResponse"}}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["environment-controller"],"summary":"Interface to create a new environment","description":"Use this interface to create a new environment.","operationId":"createEnvironmentUsingPOST","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the project that includes the version for which you want to create a new environment.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version for which you want to create a new environment.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"environment","description":"Request body for creating an environment. For more information about the request parameters, click Model and Model Schema.","required":true,"schema":{"$ref":"#/definitions/Environment"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/EnvironmentDetailsResponse"}},"201":{"description":"Created.","schema":{"$ref":"#/definitions/EnvironmentDetailsResponse"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the environment.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - An environment with the specified name already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["environment-controller"],"summary":"Interface to delete all environments on a project versiona","description":"Use this interface to delete all environments on a project version.","operationId":"deleteAllEnvironmentUsingDELETE","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the project that includes the version from which you want delete an environment.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version from which you want to delete an environment.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success","schema":{"type":"object","additionalProperties":{"type":"object"}}},"204":{"description":"No Content"},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid

or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the environment.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Environment with the specific ID is not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/environments/{environmentId}":{"get":{"tags":["environment-controller"],"summary":"Interface for getting environment details","description":"Use this interface to retrieve the details of an environment.","operationId":"getEnvironmentUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the project that includes the version related to the environment for which you want to get the details.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version related to the environment for which you want to get the details.","required":true,"type":"integer","format":"int64"},{"name":"environmentId","in":"path","description":"ID of the environment for which you want to get the details.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/EnvironmentDetailsResponse"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":{"tags":["environment-controller"],"summary":"Interface for updating an environment","description":"Use this interface to update the name, description, and data sources of an environment.","operationId":"updateEnvironmentUsingPUT","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the project that includes the version related to the environment that you want to update.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version related to the environment that you want to update.","required":true,"type":"integer","format":"int64"},{"name":"environmentId","in":"path","description":"ID of the environment that you want to update.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"environmentUpdate","description":"Request body for updating an environment. For more information about the parameters, click Model and Model Schema.","required":false,"schema":{"$ref":"#/definitions/EnvironmentUpdate"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/EnvironmentDetailsResponse"}},"201":{"description":"Created"},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the environment.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found or environment with the specific ID is not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - An environment with the specified name already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["environment-controller"],"summary":"Interface to delete an environment","description":"Use this interface to delete

an environment.","operationId":"deleteEnvironmentUsingDELETE","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"projectId","in":"query","description":"ID of the project that includes the version
from which you want delete an environment.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version from which
you want to delete an environment.","required":true,"type":"integer","format":"int64"},
{"name":"environmentId","in":"path","description":"ID of the environment that you want
to delete.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success","schema":{"type":"object","additionalProperties":{"type":"object"}}},"204":
{"description":"No Content"},"400":{"description":"Bad Request - Specific reason is included in the error
message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or
expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does
not have permissions to access the environment.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Environment with the specific ID is not found.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error
message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/environments/{environmentId}/
actions/findDatasources":{"get":{"tags":["environment-controller"],"summary":"Interface for getting data
sources where a specific table exists.","description":"Use this interface to get the data sources where a
specific table exists.","operationId":"findDataSourcesOfTableinEnvironmentUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"projectId","in":"query","description":"ID of the project related to the environment for
which you want to get the data sources.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version related to the environment
for which you want to get the data sources.","required":true,"type":"integer","format":"int64"},
{"name":"environmentId","in":"path","description":"ID of the environment for which you
want to get the data sources.","required":true,"type":"integer","format":"int64"},
{"name":"tableName","in":"query","description":"Table name that you want to use to find the related
data sources where the table exists.","required":true,"type":"string"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/DataSourceSet"}},"400":{"description":"Bad
Request - Request does not have a valid format or has missing required parameters.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired
token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/reservations":{"get":{"tags":["data-reservation-
controller"],"summary":"Interface to get all reservations for a user.","description":"Use this interface
to get all reservations for a user. Supports paginated response with filtering by optional search token.
Optionally, Project ID and Version ID can also be passed to further filter the records but they must be
provided together.","operationId":"getReservationsUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"projectId","in":"query","description":"ID of the project for which you want to

retrieve the reservations. Optional.","required":false,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the version under the given project for which
you want to retrieve the reservations. Optional.","required":false,"type":"integer","format":"int64"},
{"name":"page","in":"query","description":"Page number that you want to retrieve in a
paginated result. Defaults to 1 if page size is specified. Returns all reservations
if page and size are empty.","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"Number of reservations you want to retrieve per
page in a paginated reservations result. Defaults to 25 if page is specified. Returns all
reservations if page and size are empty.","required":false,"type":"integer","format":"int32"},
{"name":"searchText","in":"query","description":"Search text that you want to use to filter the reservations.
Optional.","required":false,"type":"string"}],"responses":{"200":{"description":"Success.","schema":
{"type":"array","items":{"$ref":"#/definitions/PaginatedReservationDTO"}}},"400":{"description":"Bad
Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions
to get the reservations.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found
- Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}},"post":{"tags":["data-reservation-controller"],"summary":"Interface for creating a new
reservation entry in the reservation registry","description":"Use this interface to create a new reservation
entry.","operationId":"createReservationUsingPOST","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
authorization scheme to access any protected resource through this API on behalf of the user. For Example:
Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of
the project where the reservation has to be performed.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version where the
reservation has to be performed.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"reservationInfo","description":"Request body for creating a
reservation.","required":true,"schema":{"$ref":"#/definitions/ReservationEntity"}}],"responses":
{"200":{"description":"OK","schema":{"$ref":"#/definitions/ReservationCreateResult"}},"201":
{"description":"Created"},"202":{"description":"Reservation request has been accepted but the resources
have not been reserved yet.","schema":{"$ref":"#/definitions/ReservationCreateResult"}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to perform
the reservation.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found -
Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":
{"description":"Conflict - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included
in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/reservations/
{reservationId}":{"get":{"tags":["data-reservation-controller"],"summary":"Interface for getting
a reservation","description":"Use this interface to get a specific reservation and its associated
resources.","operationId":"getReservationUsingGET","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
authorization scheme to access any protected resource through this API on behalf of the user. For Example:
Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of
the project that associates the reservation to get.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version that
associates the reservation to get.","required":true,"type":"integer","format":"int64"},

{"name":"reservationId","in":"path","description":". ID of the reservation for which the
details are to get.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/ReservationEntity"}},"400":{"description":"Bad
Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Server authentication failed - Invalid or expired
token","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden -
User does not have permissions to delete the reservation.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Reservation with the specific ID is not
found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server
Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}},"delete":{"tags":["data-reservation-controller"],"summary":"Interface for deleting
reservation","description":"Use this interface to delete a specific reservation and its associated
resources.","operationId":"deleteReservationUsingDELETE","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the
project where the reservation has to be deleted from.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the version where the reservation
has to be deleted from.","required":true,"type":"integer","format":"int64"},
{"name":"reservationId","in":"path","description":"ID of the reservation that has to
be deleted.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/StringResponse"}},"204":{"description":"No
Content"},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed - Invalid
or expired token","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
- User does not have permissions to delete the reservation.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Reservation with the specific ID is not found.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific
reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/
api/ca/v1/testDataModels":{"get":{"tags":["test-data-model-controller"],"summary":"Interface for
getting the list of test data models","description":"Use this interface to get the list of test
data models.","operationId":"getDataModelsUsingGET","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login
interface to perform a user login using user credentials in the Basic HTTP authorization scheme.
The API responds with a security token, which is valid for 24 hours by default. Use the security
token in the Bearer HTTP authorization scheme to access any protected resource through this
API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"projectId","in":"query","description":"ID of the project for which you want
to retrieve test data models.","required":false,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version for which you
want to retrieve test data models.","required":false,"type":"integer","format":"int64"},
{"name":"page","in":"query","description":"Page number that you want to retrieve in the
paginated result. Default value is 1.","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"Page size of each page that you want to retrieve in
the paginated result. Default value is 15.","required":false,"type":"integer","format":"int32"},
{"name":"sortDir","in":"query","description":"Sorting order that you want to use to sort
the paginated result. Valid values are ASC and DESC.","required":false,"type":"string"},
{"name":"sortField","in":"query","description":"Field on which you want to apply the
sorting.","required":false,"type":"string"},{"name":"searchText","in":"query","description":"Search
text that you want to use to filter the test data model list.","required":false,"type":"string"},

{"name":"searchFields","in":"query","description":"List of fields on which you want to perform the search.","required":false,"type":"array","items":{"type":"string"},"collectionFormat":"multi"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/PaginatedTestDataModelsDTO"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["test-data-model-controller"],"summary":"Interface for creating a new test data model","description":"Use this interface to create a new test data model.","operationId":"createTestDataModelUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the project that you want to use to create a new test data model.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version that you want to use to create a new test data model.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"testDataModel","description":"Test data model details that you want to use to create a test data model. For more information about parameters in Model Schema, click Model.","required":true,"schema":{"$ref":"#/definitions/TestDataModelDetails"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/TestDataModelDetails"}},"201":{"description":"Created.","schema":{"$ref":"#/definitions/TestDataModelDetails"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/testDataModels/{testDataModelId}":{"get":{"tags":["test-data-model-controller"],"summary":"Interface for getting test data model details","description":"Use this interface to retrieve the details of a specific test data model.","operationId":"getDataModelUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"testDataModelId","in":"path","description":"ID of the test data model for which you want to get the details.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project that is related to the test data model for which you want to get the details.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version that is related to the test data model for which you want to get the details.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/TestDataModelDetails"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Specific reason is included in the error

message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":{"tags":["test-data-model-controller"],"summary":"Interface for modifying test data model attributes","description":"Use this interface to modify the attributes of a test data model.","operationId":"updateDataModelUsingPUT","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"testDataModelId","in":"path","description":"ID of the test data model that you want to update.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project that is related to the test data model for which you want to update the details.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version that is related to the test data model for which you want to update the details.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"testDataModel","description":"Test data model details that you want to update. For more information about parameters, click Model.","required":true,"schema":{"$ref":"#/definitions/TestDataModelDetails"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/TestDataModelDetails"}},"201":{"description":"Created"},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["test-data-model-controller"],"summary":"Interface for deleting a test data model","description":"Use this interface to delete a specific test data model.","operationId":"deleteDataModelUsingDELETE","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"testDataModelId","in":"path","description":"ID of the test data model that you want to delete.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project related to the test data model that you want to delete.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version related to the test data model that you want to delete.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"type":"object","additionalProperties":{"type":"object"}}},"204":{"description":"No Content"},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/testDataModels/{testDataModelId}/actions/find":{"post":{"tags":["find-controller"],"summary":"Interface for finding the data","description":"Use this interface to find the data.","operationId":"findTestDataUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer

{{token}}","required":true,"type":"string"},{"name":"testDataModelId","in":"path","description":"ID of the test data model that you want to use to find the data.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project that you want to use to find the data.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version that you want to use to find the data.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"requestBody","description":"Request body that includes parameters to find the test data.For more information about parameters in Model Schema, click Model.","required":true,"schema":{"$ref":"#/definitions/FindTestDataModel"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/FindResult"}},"201":{"description":"Created"},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/testDataModels/{testDataModelId}/associations":{"get":{"tags":["association-controller"],"summary":"Interface for getting associations","description":"Use this interface to get associations in a test data model.","operationId":"getAssociationsUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the / user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the project that includes the test data model for which you want to get the associations.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version that includes the test data model for which you want to get the associations.","required":true,"type":"integer","format":"int64"},{"name":"testDataModelId","in":"path","description":"ID of the test data model for which you want to get the associations.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/Association"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["association-controller"],"summary":"Interface for creating a new association","description":"Use this interface to create a new association in a test data model.","operationId":"createAssociationUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the / user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the project that includes the test data model for which you want to create an association.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version that includes the test data model for which you want to create an association.","required":true,"type":"integer","format":"int64"},{"name":"testDataModelId","in":"path","description":"ID of the test data model for which you want to create an association.","required":true,"type":"integer","format":"int64"},{"name":"forceUpdate","in":"query","description":"Set it to true if you want to forcefully save the new association in case of a conflict.","required":true,"type":"boolean"},{"in":"body","name":"association","description":"Association details that you want to add to the data model. For more information about parameters in Model Schema, click Model.","required":true,"schema":

{"$ref":"#/definitions/Association"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/Association"}},"201":{"description":"Created"},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Assocaition between the same entities already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/testDataModels/{testDataModelId}/associations/{associationId}":{"get":{"tags":["association-controller"],"summary":"Interface for getting details of an association","description":"Use this interface to get the details of a specific association in a test data model.","operationId":"getAssociationUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the project related to the test data model that includes the association for which you want to get the details.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version related to the test data model that includes the association for which you want to get the details.","required":true,"type":"integer","format":"int64"},{"name":"testDataModelId","in":"path","description":"ID of the test data model that includes the association for which you want to get the details.","required":true,"type":"integer","format":"int64"},{"name":"associationId","in":"path","description":"ID of the association for which you want to get the details.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/Association"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Assocaition between the same entities already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":{"tags":["association-controller"],"summary":"Interface for updating an association","description":"Use this interface to update an association in a test data model.","operationId":"updateAssociationUsingPUT","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the project related to the test data model that includes the association you want to update.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version related to the test data model that includes the association you want to update.","required":true,"type":"integer","format":"int64"},{"name":"testDataModelId","in":"path","description":"ID of the test data model that includes the association you want to update.","required":true,"type":"integer","format":"int64"},{"name":"associationId","in":"path","description":"ID of the association that you want to update.","required":true,"type":"integer","format":"int64"},{"name":"forceUpdate","in":"query","description":"Set it to true if you want to forcefully save the association in case of a conflict.","required":true,"type":"boolean"},{"in":"body","name":"association","description":"Association details that you want to update. For more information about parameters in Model Schema, click Model.","required":true,"schema":{"$ref":"#/

definitions/Association"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/Association"}},"201":{"description":"Created"},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Assocaition between the same entities already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["association-controller"],"summary":"Interface for deleting an association","description":"Use this interface to delete an association in a test data model.","operationId":"deleteAssociationUsingDELETE","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the project that includes the test data model from which you want to delete the association.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version that includes the test data model from which you want to delete the association.","required":true,"type":"integer","format":"int64"},{"name":"testDataModelId","in":"path","description":"ID of the test data model from which you want to delete the association.","required":true,"type":"integer","format":"int64"},{"name":"associationId","in":"path","description":"ID of the association you want to delete.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/Association"}},"204":{"description":"No Content"},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Assocaition between the same entities already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/testDataModels/{testDataModelId}/fields":{"get":{"tags":["field-controller"],"summary":"Interface for listing fields in a test data model","description":"Use this interface to list the fields in a test data model.","operationId":"listFieldsUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"testDataModelId","in":"path","description":"ID of the test data model for which you want to get the associated fields.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project that includes the test data model for which you want to get the associated fields.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version that includes the test data model for which you want to get the associated fields.","required":true,"type":"integer","format":"int64"},{"name":"page","in":"query","description":"Page number that you want to retrieve in the paginated result. Default value is 1.","required":false,"type":"integer","format":"int32"},{"name":"size","in":"query","description":"Page size of each page that you want to retrieve in the paginated result. Default value is 15.","required":false,"type":"integer","format":"int32"},{"name":"sortDir","in":"query","description":"Sorting order that you want to use to sort the paginated result. Valid values are ASC and DESC.","required":false,"type":"string"},{"name":"sortField","in":"query","description":"Field on which you want to apply the

```
 sorting.","required":false,"type":"string"},{"name":"searchText","in":"query","description":"Search
 text that you want to use to filter the field list.","required":false,"type":"string"},
{"name":"searchFields","in":"query","description":"List of fields on which you want to perform the
 search.","required":false,"type":"array","items":{"type":"string"},"collectionFormat":"multi"}],"responses":
{"200":{"description":"Success.","schema":{"$ref":"#/definitions/PaginatedFieldsListDTO"}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not
 Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in
 the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["field-
controller"],"summary":"Interface for creating a new field","description":"Use this interface to create
 a new field in a test data model.","operationId":"createFieldUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"testDataModelId","in":"path","description":"ID of the test data model that you
 want to use to create a new field.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project that includes the test data model
 for which you want to create a new field.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version that includes the test data
 model for which you want to create a new field.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"field","description":"Field details that you want to use to create a field. For
 more information about parameters in Model Schema, click Model.","required":true,"schema":{"$ref":"#/
definitions/FieldRequest"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/
FieldResponse"}},"201":{"description":"Created.","schema":{"$ref":"#/definitions/FieldResponse"}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific
 reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":
{"description":"Conflict - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included
 in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/testDataModels/
{testDataModelId}/fields/{fieldId}":{"get":{"tags":["field-controller"],"summary":"Interface for
 getting field details","description":"Use this interface to get the details of a field in a test data
 model.","operationId":"getFieldUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":
[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user
 login using user credentials in the Basic HTTP authorization scheme. The API responds with a security
 token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization
 scheme to access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"testDataModelId","in":"path","description":"ID
 of the test data model that includes the field for which you want to get the
 details.","required":true,"type":"integer","format":"int64"},{"name":"fieldId","in":"path","description":"ID
 of the field for which you want to get the details.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project related to the test data model that includes
 the field for which you want to get the details.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version related
 to the test data model that includes the field for which you want to get the
 details.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/FieldResponse"}},"400":{"description":"Bad Request
 - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
```

{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":
{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error
 message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error -
 Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":
{"tags":["field-controller"],"summary":"Interface for updating field properties","description":"Use this
 interface to update the properties of a field.","operationId":"updateFieldUsingPUT","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"testDataModelId","in":"path","description":"ID of the test data model that includes the field
 for which you want to update the properties.","required":true,"type":"integer","format":"int64"},
{"name":"fieldId","in":"path","description":"ID of the field for which you want
 to update the properties.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project related to the test data model that includes
 the field for which you want to update the properties.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version
 related to the test data model that includes the field for which you want to
 update the properties.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"field","description":"Field property details that you want to update. For
 more information about parameters in Model Schema, click Model.","required":true,"schema":
{"$ref":"#/definitions/FieldRequest"}}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/FieldResponse"}},"201":{"description":"Created"},"400":{"description":"Bad
 Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific
 reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["field-controller"],"summary":"Interface
 for deleting a field","description":"Use this interface to delete a field in a test data
 model.","operationId":"deleteFieldUsingDELETE","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login
 interface to perform a user login using user credentials in the Basic HTTP authorization scheme.
 The API responds with a security token, which is valid for 24 hours by default. Use the security
 token in the Bearer HTTP authorization scheme to access any protected resource through this
 API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"testDataModelId","in":"path","description":"ID of the test data model that
 includes the field you want to delete.","required":true,"type":"integer","format":"int64"},
{"name":"fieldId","in":"path","description":"ID of the field that you
 want to delete.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project related to the test data model
 that contains the field you want to delete.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version related to the test data model that
 contains the field you want to delete.","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"Success.","schema":{"type":"object","additionalProperties":{"type":"object"}}},"204":
{"description":"No Content"},"400":{"description":"Bad Request - Specific reason is included in the error
 message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication
 failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":
{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is
 included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/
v1/testDataModels/{testDataModelId}/fields/{fieldId}/actions/findValues":{"post":{"tags":["field-

controller"],"summary":"Interface for finding field values","description":"Use this interface
 to find values for field","operationId":"findFieldValuesUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"testDataModelId","in":"path","description":"ID of the test data model that you
 want to use to find field values.","required":true,"type":"integer","format":"int64"},
{"name":"fieldId","in":"path","description":"ID of the field for which you
 want to find values","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project that you want
 to use to find field values.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version that you
 want to use to find field values.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"requestBody","description":"Request body that includes parameters to find field
 values. For more information about parameters in Model Schema, click Model.","required":true,"schema":
{"$ref":"#/definitions/FindFieldValuesRequest"}}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/FindFieldValuesResult"}},"201":{"description":"Created"},"400":{"description":"Bad
 Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific
 reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/testDataModels/{testDataModelId}/filters":
{"get":{"tags":["test-data-model-controller"],"summary":"Interface for getting the list of filters
 of a test data model.","description":"Use this interface to get the list of filters of a test data
 model.","operationId":"getDataModelFiltersUsingGET","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login
 interface to perform a user login using user credentials in the Basic HTTP authorization scheme.
 The API responds with a security token, which is valid for 24 hours by default. Use the security
 token in the Bearer HTTP authorization scheme to access any protected resource through this
 API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"testDataModelId","in":"path","description":"ID of the test data model for which you
 want to list the associated filters.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project related to the test data model for
 which you want to list the associated filters.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version
 related to the test data model for which you want to list the associated
 filters.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success","schema":{"$ref":"#/definitions/ResourceClassFilterSet"}},"400":{"description":"Bad
 Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found
 - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}},"/api/ca/v1/testDataModels/{testDataModelId}/syncTasks/actions/startSync":{"post":
{"tags":["test-data-model-controller"],"summary":"Interface for starting synchronization of data from
 remote db to local cache","description":"Use this interface start data synchronization. Test Data
 Model must have dataPrefetch enabled","operationId":"startSyncUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the

security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"testDataModelId","in":"path","description":"ID of the test data model that you want to start synchronization for.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project that is related to the test data model.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version that is related to the test data model.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/TestDataModelDetails"}},"201":{"description":"Created"},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v2/testDataModels/{testDataModelId}":{"get":{"tags":["test-data-model-controller"],"summary":"Interface for getting New (V 2.0) test data model details","description":"Use this interface to retrieve the details of a specific test data model.","operationId":"getNewDataModelUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"testDataModelId","in":"path","description":"ID of the test data model for which you want to get the details.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project that is related to the test data model for which you want to get the details.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version that is related to the test data model for which you want to get the details.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/TestDataModelDetails"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}}},"definitions":{"Association":{"type":"object","required":["associationType","joinFields","name","sourceEntity","targetEntity"],"properties":{"associationType":{"type":"string","description":"Type of the relationship.","enum":["ONE_ONE","ONE_MANY","MANY_ONE"]},"joinFields":{"type":"array","description":"Join Fields.","items":{"$ref":"#/definitions/JoinFieldDetails"}},"name":{"type":"string","description":"Name of the association that you are creating"},"sourceEntity":{"description":"Parent Data Entity","$ref":"#/definitions/DataEntity"},"targetEntity":{"description":"Child Data Entity.","$ref":"#/definitions/DataEntity"}}},"CopyEnvironmentsFromRequest":{"type":"object","properties":{"projectId":{"type":"integer","format":"int64"},"sourceVersionId":{"type":"integer","format":"int64"},"targetVersionId":{"type":"integer","format":"int64"}}},"DataEntity":{"type":"object","required":["dataSource","name","schema"],"properties":{"dataSource":{"type":"string","description":"Data source of the entity"},"name":{"type":"string","description":"Name of the Entity or table."},"schema":{"type":"string","description":"Schema of the entity"}}},"DataRecord":{"type":"object","properties":{"columnValues":{"type":"object"},"recordId":{"$ref":"#/definitions/RecordIdentifier"},"reserved":{"type":"boolean"},"reservedBy":{"type":"string"},"reservedDate":

```
{"type":"string"}}},"DataSourceSet":{"type":"object","properties":{"datasources":
{"type":"array","items":{"type":"string"}}}},"DatasourceConnProfile":{"type":"object","required":
["name"],"properties":{"connectionProfileName":{"type":"string","description":"Name of the
 connection profile."},"connectionProfileStatus":{"type":"string","description":"Existence
 of the connection profile.","readOnly":true,"enum":["INVALID","EXISTS","NOTEXISTS"]},"name":
{"type":"string","description":"Name of the data source."}}},"Environment":{"type":"object","required":
["name"],"properties":{"datasourcesConnectionProfiles":{"type":"array","description":"List
 of data source and the corresponding connection profile names to associate with the
 environment","items":{"$ref":"#/definitions/DatasourceConnProfile"}},"description":
{"type":"string","description":"Description of the environment"},"name":{"type":"string","description":"Name
 of the environment"}}},"EnvironmentDetailsResponse":{"type":"object","required":
["datasourcesConnectionProfiles","description","id","name","projectID","versionID"],"properties":
{"createdBy":{"type":"string","description":"Name of the user who created this environment
 template"},"creationDate":{"type":"string","description":"Creation date of the environment
 template"},"datasourcesConnectionProfiles":{"type":"array","description":"List of datasource
 and corresponding connection profile names associated with the environment","items":{"$ref":"#/
definitions/DatasourceConnProfile"}},"description":{"type":"string","description":"Description
 of the environment"},"id":{"type":"integer","format":"int64","description":"ID of the
 environment"},"modifiedBy":{"type":"string","description":"Name of the user who modified this
 environment template"},"modifiedDate":{"type":"string","description":"Last modified date of the
 environment template"},"name":{"type":"string","description":"Name of the environment"},"projectID":
{"type":"integer","format":"int64","description":"Project with which the environment is
 associated"},"versionID":{"type":"integer","format":"int64","description":"Version with
 which the environment is associated"}}},"EnvironmentResponse":{"type":"object","required":
["description","id","name","projectID","versionID"],"properties":{"createdBy":
{"type":"string","description":"Name of the user who created this environment
 template"},"creationDate":{"type":"string","description":"Creation date of the environment
 template"},"description":{"type":"string","description":"Description of the environment"},"id":
{"type":"integer","format":"int64","description":"ID of the environment"},"modifiedBy":
{"type":"string","description":"Name of the user who modified this environment
 template"},"modifiedDate":{"type":"string","description":"Last modified date of the environment
 template"},"name":{"type":"string","description":"Name of the environment"},"projectID":
{"type":"integer","format":"int64","description":"Project with which the environment is
 associated"},"versionID":{"type":"integer","format":"int64","description":"Version with
 which the environment is associated"}}},"EnvironmentUpdate":{"type":"object","properties":
{"datasourcesConnectionProfiles":{"type":"array","description":"List of data source and
 corresponding connection profile names to update the environment.","items":{"$ref":"#/definitions/
DatasourceConnProfile"}},"description":{"type":"string","description":"Description of the
 environment."},"name":{"type":"string","description":"Name of the environment."}}},"ErrorResponse":
{"type":"object","properties":{"errorCode":{"type":"string"},"errorDetail":{"type":"string"},"errorMsg":
{"type":"string"},"status":{"type":"integer","format":"int32"},"timestamp":{"type":"string"}}},"FieldRequest":
{"type":"object","required":["associationId","displayType","name"],"properties":{"associationId":
{"type":"integer","format":"int64","description":"ID of the association for which you want to create
 the field."},"name":{"type":"string","description":"Name of the column  for which you are creating
 the field."},"displayName":{"type":"string","description":"Display name of the field"},"displayOrder":
{"type":"integer","format":"int32","description":"The order in which this field is to be displayed in
 the Tester self service"},"isVisible":{"type":"boolean","example":false,"description":"Flag to indicate
 whether field is visible or not. Defaults to true"},"displayType":{"type":"string","description":"Display
 type of the field"},"displayValues":{"type":"array","description":"Default values for the
 field","items":{"type":"string"}}}},"FieldResponse":{"type":"object","properties":{"associationId":
{"type":"integer","format":"int64","description":"ID of the association on which this field is
 created."},"name":{"type":"string","description":"Name of the column  for which this field is
 created"},"displayName":{"type":"string","description":"Display name of the field"},"displayOrder":
```

{"type":"integer","format":"int32","description":"The order in which this field is to be displayed
in the Tester self service"},"isVisible":{"type":"boolean","example":false,"description":"Flag to
indicate whether field is visible or not"},"displayType":{"type":"string","description":"Display
type of the field"},"dataType":{"type":"string","description":"Data type of the
field"},"displayValues":{"type":"array","description":"Default values for the field","items":
{"type":"string"}},"id":{"type":"integer","format":"int64","description":"ID of the field"},"projectId":
{"type":"integer","format":"int64","description":"ID of project on which the field is
created"},"versionId":{"type":"integer","format":"int64","description":"ID of project on which the field
is created"}}},"FindFieldValuesFilter":{"type":"object","required":["operator","values"],"properties":
{"operator":{"type":"string","description":"Operator allowed for this filter","enum":
["EQUALS","NOT_EQUAL","LESS_THAN","LESS_THAN_OR_EQUAL_TO","GREATER_THAN","GREATER_THAN_OR_EQUAL_TO","CONTAINS","BETWEEN
{"type":"array","description":"List of allowed values for the filter.","items":
{"type":"string"}}}},"FindFieldValuesRequest":{"type":"object","required":["environmentId"],"properties":
{"count":{"type":"integer","format":"int32","description":"Number of records to fetch."},"environmentId":
{"type":"integer","format":"int64","description":"ID of the environment."},"filters":
{"type":"array","description":"List of filters to apply","items":{"$ref":"#/definitions/
FindFieldValuesFilter"}}}},"FindFieldValuesResult":{"type":"object","properties":{"values":
{"type":"array","items":{"type":"string"}}}},"FindResult":{"type":"object","properties":
{"records":{"type":"array","items":{"$ref":"#/definitions/DataRecord"}},"startAfterValues":
{"type":"object"}}},"FindTestDataModel":{"type":"object","required":["environmentId"],"properties":
{"environmentId":{"type":"integer","format":"int64","description":"ID of the
environment."},"filters":{"type":"array","description":"List of filters for finding the
data.","items":{"$ref":"#/definitions/FindTestDataModelFilter"}},"includeReservedRecords":
{"type":"boolean","example":false,"description":"Include reserved records in find
result."},"showReservedRecords":{"type":"boolean","example":false,"description":"Show reserved
records in find result."},"startAfterValues":{"type":"object","description":"Start After Values
required to find the data for remaining pages. Provide Map of model key and value which is the
result of current page find result."}}},"FindTestDataModelFilter":{"type":"object","required":
["fieldId","operator","values"],"properties":{"fieldId":{"type":"integer","format":"int64","description":"ID
of the filter field."},"operator":{"type":"string","description":"Operator allowed for this filter","enum":
["EQUALS","NOT_EQUAL","LESS_THAN","LESS_THAN_OR_EQUAL_TO","GREATER_THAN","GREATER_THAN_OR_EQUAL_TO","CONTAINS","BETWEEN
{"type":"array","description":"List of allowed values for the filter.","items":
{"type":"string"}}}},"JoinFieldDetails":{"type":"object","required":
["fieldName","referenceFieldName"],"properties":{"fieldName":{"type":"string","description":"Name
of the field used in the relationship."},"referenceFieldName":{"type":"string","description":"Name
of the reference field used in relationship"}}},"PaginatedFieldsListDTO":
{"type":"object","properties":{"fields":{"type":"array","items":{"$ref":"#/definitions/
FieldResponse"}},"noOfFields":{"type":"integer","format":"int32"},"totalNoOfFields":
{"type":"integer","format":"int64"}}},"PaginatedReservationDTO":{"type":"object","properties":
{"numberOfReservations":{"type":"integer","format":"int32","description":"Number of reservations
retrieved"},"reservations":{"type":"array","description":"List of actual reservations
retrieved","items":{"$ref":"#/definitions/ReservationEntity"}},"totalNumberOfReservations":
{"type":"integer","format":"int64","description":"Total number of reservations
available"}}},"PaginatedTestDataModelsDTO":{"type":"object","properties":
{"numberOfTestDataModels":{"type":"integer","format":"int32"},"testDataModelsList":
{"type":"array","items":{"$ref":"#/definitions/TestDataModel"}},"totalNumberOfTestDataModels":
{"type":"integer","format":"int64"}}},"RecordIdentifier":{"type":"object","properties":
{"keys":{"type":"object","additionalProperties":{"type":"string"}}}},"ReservationCreateResult":
{"type":"object","required":["reservationId"],"properties":{"reservationId":
{"type":"integer","format":"int64","description":"ID of the reservation"}}},"ReservationEntity":
{"type":"object","required":["dataModelId","environmentId","reservationName","resources"],"properties":
{"dataModelId":{"type":"integer","format":"int64","description":"ID of the Data Model associated with
the reservation"},"dataModelName":{"type":"string","description":"Name of the Data Model associated

with the reservation"},"environmentId":{"type":"integer","format":"int64","description":"Environment ID against which reservation is made"},"environmentName":{"type":"string","description":"Name of the Environment against which reservation is made"},"resErrorMessage":{"type":"string","description":"Job Failure error Message"},"reservationId":{"type":"integer","format":"int64","description":"ID of the data reservation. This is generated internally."},"reservationName":{"type":"string","description":"Name of the reservation."},"reservationState":{"type":"string","description":"Reservation State.","enum":["UNDEFINED","CREATED","STARTED","SUCCESS","FAILED","EXPIRED","INVALID","PURGED"]},"resources":{"type":"array","description":"List of reserved entities","items":{"$ref":"#/definitions/ReservationResource"}}}},"ReservationResource":{"type":"object","required":["dataModelId","modelKeys"],"properties":{"dataModelId":{"type":"integer","format":"int64","description":"ID of the Data class for the resource"},"modelKeys":{"type":"object","description":"Map of the model key associated. E.g {\"OrderID\" : \"O1\",\"OrderName\":\"Order1\"}","additionalProperties":{"type":"string"}}}},"ResourceClassFilterSet":{"type":"object","properties":{"filters":{"type":"array","items":{"$ref":"#/definitions/TestDataModelFilter"}}}},"Root":{"type":"object","properties":{"displayName":{"type":"string","description":"Test data model Root Details"},"rootEntity":{"description":"Root entity details","$ref":"#/definitions/DataEntity"}}},"StringResponse":{"type":"object","properties":{"response":{"type":"string"}}},"TestDataModel":{"type":"object","properties":{"dataPrefetch":{"type":"string"},"dataPrefetchErrMsg":{"type":"string"},"dataSynchronized":{"type":"boolean"},"description":{"type":"string"},"id":{"type":"integer","format":"int64"},"modelVersion":{"type":"string"},"name":{"type":"string"},"projectId":{"type":"integer","format":"int64"},"reserved":{"type":"boolean"},"versionId":{"type":"integer","format":"int64"},"visible":{"type":"boolean"}}},"TestDataModelDetails":{"type":"object","required":["modelKeys","name","root"],"properties":{"dataPrefetch":{"type":"string","description":"Data Prefetch settings of this test data model. Possible values: OFF, ON_DEMAND, PERIODIC. If left empty data prefetch is OFF."},"modelVersion":{"type":"string","description":"Serves to distinguish new and legacy data models"},"name":{"type":"string","description":"Name of the test data model that you are creating."},"description":{"type":"string","description":"Description of the test data model that you are creating."},"visible":{"type":"boolean","example":false,"description":"Flag indicating if test data model is active. Defaults to false"},"dataSynchronized":{"type":"boolean","example":false,"description":"Flag indicating if data views should be synchronized. Defaults to false"},"reserved":{"type":"boolean","example":false,"description":"Flag indicating if test data model can show already reserved data. Defaults to false"},"modelKeys":{"type":"array","description":"List of model keys for the root.","items":{"type":"string"}},"root":{"description":"Test data model Root Details","$ref":"#/definitions/Root"}}},"TestDataModelFilter":{"type":"object","properties":{"defaultValues":{"type":"array","description":"Default value of the field.","readOnly":true,"items":{"type":"string"}},"displayDataType":{"type":"string","description":"Data type of the field.","readOnly":true},"displayName":{"type":"string","description":"Display name of the field.","readOnly":true},"displayType":{"type":"string","description":"Display type of the field","readOnly":true},"fieldId":{"type":"integer","format":"int64","description":"ID of the field","readOnly":true}}}}}

TDMGeneratorService

alpha

{"swagger":"2.0","info":{"description":"This section includes the APIs that perform various operations for modeling the projects. It also provides the REST API URL for the respective operation along with sample request and response body content.","version":"1.0","title":"CA TDM Generator Service API","termsOfService":"http://ca.com","contact":{"name":"CA Technologies"},"license":{"name":"The CA License Version 2.0","url":"https://ca.com/LICENSE"}},"host":"vtdm-dev-demo:8443","basePath":"/TDMGeneratorService","tags":[{"name":"data-generator-controller","description":"Interface for data generator"},{"name":"data-painter-controller","description":"Interface for data painter"},{"name":"seed-data-controller","description":"Interface for SeedData"},{"name":"data-

generator-action-controller","description":"Interface for publish actions"},{"name":"publish-
configuration-controller","description":"Interface for publish configurations"},{"name":"variable-
controller","description":"Interface for variables"}],"paths":{"/api/ca/v1/generatorFunctions":{"get":
{"tags":["data-painter-controller"],"summary":"Interface for getting list of all data generator
 functions","description":"Use this interface to retrieve the list of all the data generator
 functions.","operationId":"getFunctionsUsingGET","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login
 interface to perform a user login using user credentials in the Basic HTTP authorization scheme.
 The API responds with a security token, which is valid for 24 hours by default. Use the security
 token in the Bearer HTTP authorization scheme to access any protected resource through this
 API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"q","in":"query","description":"List of properties on the resource based on which you
 want to search or filter. The property that you can filter on is dataType. All the properties
 must be sent as value for 'q' query parameter, for example q=(datatype=Numeric) or q=(dataType!
=Numeric)","required":false,"type":"string"}],"responses":{"200":{"description":"Success.","schema":
{"type":"array","items":{"$ref":"#/definitions/GeneratorFunctionsContainerModel"}}},"401":
{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the
 project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error -
 Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/
api/ca/v1/generatorSeedCategories":{"get":{"tags":["seed-data-controller"],"summary":"Interface for
 getting list of unique seed data categories","description":"Use this interface to retrieve the list of
 unique seed data categories.","operationId":"getSeedNameAndColumnsUsingGET","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"getColumnDetails","in":"query","description":"Set this attribute to true if you want to retrieve
 the seed column details.","required":true,"type":"boolean","default":false}],"responses":{"200":
{"description":"Success","schema":{"type":"array","items":{"$ref":"#/definitions/SeedData"}}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions
 to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found
 - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}},"/api/ca/v1/generators":{"get":{"tags":["data-generator-controller"],"summary":"Interface
 to get all the available data generators for a given user. Supports paginated response with filtering
 by search token and project and version Ids.","description":"Use this interface to get all the data
 generators for a user. Supports paginated response with filtering by search token which are optional.
 Project Id and Version Id can also be mentioned to further filter the records but they must be provided
 together.","operationId":"getGeneratorsByUserUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"page","in":"query","description":"Page number which you want to retrieve in a paginated data
 generators result. Indexed with 0. Optional.","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"Page size of each page with which
 you want to retrieve in a paginated data generators result. Default value
 is 100. Optional.","required":false,"type":"integer","format":"int32"},

{"name":"projectId","in":"query","description":"Id of the project for which generators
have to be retrieved. Optional.","required":false,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Id of the version under a given project for which generators
have to be retrieved. Optional.","required":false,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"Success.","schema":{"$ref":"#/definitions/PaginatedGeneratorsResult"}},"400":
{"description":"Bad Request - Request does not have a valid format or has missing required
parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid
or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
- User does not have permissions to access the project.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the
error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["data-generator-
controller"],"summary":"Interface to create a new data generator where you can write data generation
definitions","description":"Use this interface to create a new data generator where you can write
data generation definitions.","operationId":"createDataGeneratorUsingPOST","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"generatorInfo","description":"GeneratorInfo object containing the key value
pairs of the properties with which you want to create a new generator.","required":true,"schema":
{"$ref":"#/definitions/GeneratorInfo"}},{"name":"projectId","in":"query","description":"ID of the
project where the generator is to be created.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the version where the generator
is to be created.","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"OK","schema":{"$ref":"#/definitions/GeneratorResult"}},"201":
{"description":"Created.","schema":{"$ref":"#/definitions/GeneratorResult"}},"400":{"description":"Bad
Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access
the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource
not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - A generator
with the specified name already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/generators/evaluateExpression":{"post":{"tags":["data-
painter-controller"],"summary":"Interface for validating the data generation rule","description":"Use this
interface to validate the data generation rule.","operationId":"resolveExpressionUsingPOST","consumes":
["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"requestBean","description":"Expression validator request information which you want to
validate.","required":true,"schema":{"$ref":"#/definitions/ExpressionValidatorRequestBean"}}],"responses":
{"200":{"description":"Success.","schema":{"type":"string"}},"400":{"description":"Bad Request - Specific
reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the
project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server
Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}},"/api/ca/v1/generators/generator/{generatorId}":{"put":{"tags":["data-generator-
controller"],"summary":"Interface to update a Generator.","description":"Use this interface to update

a Generator.","operationId":"updateGeneratorUsingPUT","consumes":["application/json"],"produces":["*/
*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to
perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
authorization scheme to access any protected resource through this API on behalf of the user. For Example:
Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID
of the project of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version
of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"generatorId","in":"path","description":"Id of the generator for which
variable has to be updated.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"generatorInfo","description":"GeneratorInfo object containing the key value pairs
of the properties with which you want to update a new generator.","required":true,"schema":{"$ref":"#/
definitions/GeneratorInfo"}}],"responses":{"200":{"description":"Success.","schema":{"type":"boolean"}},"400":
{"description":"Bad Request - Request does not have a valid format or has missing required
parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid
or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User
does not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/generators/systemVariables":{"get":{"tags":
["variable-controller"],"summary":"getSystemVariables","operationId":"getSystemVariablesUsingGET","consumes":
["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"page","in":"query","description":"page","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"size","required":false,"type":"integer","format":"int32"}],"responses":
{"200":{"description":"OK","schema":{"$ref":"#/definitions/PaginatedVariableBean"}}}}},"/api/ca/
v1/generators/validateExpression":{"post":{"tags":["data-painter-controller"],"summary":"Interface
for validating the data generation rule","description":"Use this interface to validate the
data generation rule.","operationId":"validateExpressionUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"in":"body","name":"requestBean","description":"Expression
validator request information which you want to validate.","required":true,"schema":{"$ref":"#/
definitions/ExpressionValidatorRequestBean"}},{"name":"columnId","in":"query","description":"ID
of the column to be validated.","required":true,"type":"integer","format":"int64"},
{"name":"isExpression","in":"query","description":"boolean to identify whether you need to
evaluate an expression or not.","required":false,"type":"boolean"}],"responses":{"200":
{"description":"Success.","schema":{"type":"string"}},"400":{"description":"Bad Request - Specific
reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the
project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server
Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}":{"delete":{"tags":["data-generator-
controller"],"summary":"Interface to delete a data generator where you have defined your data generator
definitions","description":"Use this interface to delete a data generator where you have defined your

data generator definitions.","operationId":"deleteGeneratorUsingDELETE","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"ID
of the generator that you want to delete.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project to which the
generator is associated.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the version to which the generator
is associated.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"type":"object"}},"400":{"description":"Bad Request - Request
does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the
data generator.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Generator
not found or may have already been deleted.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Check logs for more information.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/actions/defaultParentReferences":{"post":{"tags":
["data-generator-controller"],"summary":"Interface for updating columns in parent tables that have
default values","description":"Use this interface to update columns in parent tables that have default
values.","operationId":"defaultParentReferencesUsingPOST","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"generatorId","in":"path","description":"ID of the data generator where the table definitions are
saved","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID
of the project where the table is registered","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the version where the table is
registered","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/ObjectEntriesEffected"}},"400":{"description":"Bad
Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions
to access the resource.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found
- Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/actions/deriveChildReferences":{"post":{"tags":
["data-generator-controller"],"summary":"Interface for converting values in child columns to references to the
parent column","description":"Use this interface to populate default data generation rules for all the child
table columns referencing the appropriate columns in the parent table. Also, the data must match in both
the columns (child and parent).","operationId":"deriveChildReferencesUsingPOST","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"generatorId","in":"path","description":"ID of the data generator where the parent
and child table definitions are saved.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project where the parent

```
 and child tables are registered.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the version where the parent and child
 tables are registered.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/ObjectEntriesEffected"}},"400":{"description":"Bad
 Request - Request does not have a valid format or has missing required parameters.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired
 token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does
 not have permissions to access the resource.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/actions/updatePublishInfo":
{"put":{"tags":["data-generator-controller"],"summary":"Interface to update the publish information
 for a generator","description":"Use this interface to update the publish information for a
 generator","operationId":"savePublishConfigurationUsingPUT","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"generatorId","in":"path","description":"ID of the data generator where the publish
 configuration should be updated","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project to which the
 generator is associated.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the version to which the
 generator is associated.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"info","description":"PublishConfiguration object containing the publish information
 to be saved ","required":true,"schema":{"$ref":"#/definitions/PublishConfiguration"}}],"responses":
{"200":{"description":"Success.","schema":{"type":"object"}},"400":{"description":"Bad Request - Request
 does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access
 the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found -
 Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/publishActions":{"get":{"tags":["data-
generator-action-controller"],"summary":"Interface for getting the details of all actions for a
 generator","description":"Use this interface to get the details of all actions associated with a
 generator","operationId":"getAllPublishActionsUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"ID
 of the data generator","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project of the
 generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version of the
 generator.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"type":"array","items":{"$ref":"#/definitions/
GtrepPublishAction"}}},"400":{"description":"Bad Request - Request does not have a valid format
 or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/
```

```
ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the
 resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found -
 Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}},"post":{"tags":["data-generator-action-controller"],"summary":"Interface for creation
 a new action for a generator","description":"Use this interface for creation a new action for a
 generator","operationId":"saveActionUsingPOST","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"ID
 of the data generator","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project of the
 generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version
 of the generator.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"action","description":"action","required":true,"schema":{"$ref":"#/definitions/
GtrepPublishAction"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/
GtrepPublishAction"}},"400":{"description":"Bad Request - Request does not have a valid format
 or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the
 resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not
 found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error -
 Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/
api/ca/v1/generators/{generatorId}/publishActions/actions/updateSequences":{"post":{"tags":["data-
generator-action-controller"],"summary":"Interface for updating the sequence number of all actions for
 a given generator","description":"Use this interface for updating the sequence number of all actions
 for a given generator","operationId":"updateSequencesForGeneratorUsingPOST","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"ID
 of the data generator","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project of the
 generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version
 of the generator.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"listOfUpdatedActions","description":"List of actions with changed
 sequence numbers","required":true,"schema":{"type":"array","items":{"$ref":"#/definitions/
GtrepPublishActionOrdering"}}}],"responses":{"200":{"description":"Success.","schema":{"type":"array","items":
{"$ref":"#/definitions/GtrepPublishAction"}}},"400":{"description":"Bad Request - Request does
 not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access
 the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found -
 Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/publishActions/{actionName}":{"get":{"tags":
["data-generator-action-controller"],"summary":"Interface for getting the details of single action for
```

a given generator","description":"Use this interface to get the details of single action for a given generator","operationId":"getActionDetailsUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"ID of the data generator","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project of the generator.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project's version of the generator.","required":true,"type":"integer","format":"int64"},{"name":"actionId","in":"query","description":"ID of the action if action name is not known.","required":false,"type":"integer","format":"int64"},{"name":"actionName","in":"path","description":"actionName","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/GtrepPublishAction"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":{"tags":["data-generator-action-controller"],"summary":"Interface for updating the details of an action for a given generator","description":"Use this interface for updating the details of an action for a given generator","operationId":"updateActionUsingPUT","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"ID of the data generator","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project of the generator.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project's version of the generator.","required":true,"type":"integer","format":"int64"},{"name":"actionName","in":"path","description":"actionName","required":true,"type":"string"},{"in":"body","name":"action","description":"action","required":true,"schema":{"$ref":"#/definitions/GtrepPublishAction"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/GtrepPublishAction"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["data-generator-action-controller"],"summary":"Interface for deleting action with given name for a generator","description":"Use this interface for deleting action with given name for a generator","operationId":"deletActionUsingDELETE","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example:

```
 Bearer {{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"ID
 of the data generator","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project of the
 generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version
 of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"actionName","in":"path","description":"actionName","required":true,"type":"string"}],"responses":
{"200":{"description":"Success.","schema":{"type":"array","items":{"$ref":"#/definitions/
GtrepPublishAction"}}},"400":{"description":"Bad Request - Request does not have a valid format or has missing
 required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized -
 Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
 - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in
 the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/generators/
{generatorId}/publishActions/{actionName}/actions/execute":{"post":{"tags":["data-generator-action-
controller"],"summary":"API to execute a publish action","description":"Use this API to execute publish
 action","operationId":"executeActionUsingPOST","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"ID of
 the data generator where you want to get the details.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project of the
 generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version
 of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"timeout","in":"query","description":"Timeout value (in seconds) to wait for completion
 of actions. The action will be terminated after waiting for the mentioned timeout period for
 completion and will be counted as failure. This is applicable only for HOST type of actions
 and is ignored for other action types.","required":false,"type":"integer","format":"int64"},
{"name":"actionName","in":"path","description":"ID of the action to be
 executed.","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/PublishActionResult"}},"400":{"description":"Bad Request - Request does
 not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have
 permissions to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/publishConfigurations":
{"get":{"tags":["publish-configuration-controller"],"summary":"Interface for getting the list of all
 configurations for a generator","description":"Use this interface to get all configurations for a
 generator","operationId":"getAllConfigurationsUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the
 project","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the
```

```
 version","required":true,"type":"integer","format":"int64"},
{"name":"generatorId","in":"path","description":"ID of the data generator where you want
 to get the details.","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"Success.","schema":{"type":"array","items":{"$ref":"#/definitions/
GtrepPublishConfig"}}},"400":{"description":"Bad Request - Request does not have a valid format or has missing
 required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized -
 Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
 - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error
 message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["publish-configuration-
controller"],"summary":"Interface to create a new publish configuration which can be used for
 publishing data","description":"Interface to create a new publish configuration which can be used
 for publishing data","operationId":"createPublishConfigurationUsingPOST","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Id of the
 project where publish configuration is to be created.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Id of the version where publish
 configuration is to be created.","required":true,"type":"integer","format":"int64"},
{"name":"generatorId","in":"path","description":"ID of the data generator where publish
 configuration is to be created.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"publishConfig","description":"GtrepPublishConfig object containing the properties
 with which you want to create a new publish configuration.","required":true,"schema":{"$ref":"#/
definitions/GtrepPublishConfig"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/
definitions/GtrepPublishConfig"}},"201":{"description":"Created.","schema":{"$ref":"#/definitions/
GtrepPublishConfig"}},"400":{"description":"Bad Request - Specific reason is included in the error
 message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid
 or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
 - User does not have permissions to access the project.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"409":{"description":"Conflict - A publish configuration with the specified name
 already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/publishConfigurations/{configurationId}":
{"get":{"tags":["publish-configuration-controller"],"summary":"Interface for getting the
 configuration for a generator","description":"Use this interface for getting the configuration for a
 generator","operationId":"getConfigurationUsingGET","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the
 project","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the
 version","required":true,"type":"integer","format":"int64"},
{"name":"generatorId","in":"path","description":"ID of the data generator where
 you want to get the details.","required":true,"type":"integer","format":"int64"},
{"name":"configurationId","in":"path","description":"ID of the publish configuration to
 be retrieved.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
```

{"description":"Success.","schema":{"$ref":"#/definitions/GtrepPublishConfig"}},"400":{"description":"Bad
Request - Request does not have a valid format or has missing required parameters.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired
token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
- User does not have permissions to access the resource","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error
message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":{"tags":["publish-configuration-
controller"],"summary":"Interface to update publish configuration","description":"Interface to update
publish configuration","operationId":"updatePublishConfigurationUsingPUT","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Id of the
project where publish configuration is to be updated.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Id of the version where publish
configuration is to be updated.","required":true,"type":"integer","format":"int64"},
{"name":"generatorId","in":"path","description":"ID of the data generator where publish
configuration is to be updated.","required":true,"type":"integer","format":"int64"},
{"name":"configurationId","in":"path","description":"ID of the publish
configuration to be updated.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"publishConfig","description":"publishConfig object containing publish config Object
to be updated.","required":true,"schema":{"$ref":"#/definitions/GtrepPublishConfig"}}],"responses":
{"200":{"description":"Success.","schema":{"$ref":"#/definitions/GtrepPublishConfig"}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired
token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
- User does not have permissions to access the project.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error
message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["publish-configuration-
controller"],"summary":"Interface for deleting configuration","description":"Use this interface
to delete configuration","operationId":"deleteConfigurationUsingDELETE","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID
of the project of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version
of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"generatorId","in":"path","description":"ID of the data
generator","required":true,"type":"integer","format":"int64"},
{"name":"configurationId","in":"path","description":"ID of the
configuration","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/GtrepPublishConfigResponseWrapper"}},"400":
{"description":"Bad Request - Request does not have a valid format or has missing required
parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid
or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User
does not have permissions to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":

{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/publishConfigurations/
{configurationId}/params":{"get":{"tags":["publish-configuration-controller"],"summary":"Interface for
 getting the list of all Params for a configuration","description":"Use this interface to get all Params
 for a configuration","operationId":"getParamsForConfigurationsUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID
 of the project of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version
 of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"generatorId","in":"path","description":"ID of the data generator where
 you want to get the details.","required":true,"type":"integer","format":"int64"},
{"name":"configurationId","in":"path","description":"ID of the configuration where you
 want to get the details.","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"Success.","schema":{"type":"array","items":{"$ref":"#/definitions/
GtrepPublishConfigParam"}}},"400":{"description":"Bad Request - Request does not have a valid
 format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the
 resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found -
 Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}},"post":{"tags":["publish-configuration-controller"],"summary":"Interface
 for creation a new params","description":"Use this interface for creation a new
 params","operationId":"createParamsUsingPOST","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID
 of the project of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version
 of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"generatorId","in":"path","description":"ID of the data
 generator","required":true,"type":"integer","format":"int64"},
{"name":"configurationId","in":"path","description":"ID of the configuration where
 you want to get the details.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"listRequestContainer","description":"listRequestContainer","required":true,"schema":
{"$ref":"#/definitions/ListRequestContainer«GtrepPublishConfigParam»"}}],"responses":{"200":
{"description":"OK","schema":{"$ref":"#/definitions/GtrepPublishConfigResponseWrapper"}},"201":
{"description":"Created.","schema":{"$ref":"#/definitions/GtrepPublishConfigResponseWrapper"}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have
 permissions to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":
{"description":"Conflict - A param with the specified name already exists.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included

in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":{"tags":["publish-configuration-controller"],"summary":"Interface to Update params","description":"Use this interface to Update params","operationId":"updateParamsUsingPUT","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the project of the generator.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project's version of the generator.","required":true,"type":"integer","format":"int64"},{"name":"generatorId","in":"path","description":"ID of the data generator","required":true,"type":"integer","format":"int64"},{"name":"configurationId","in":"path","description":"ID of the configuration where you want to get the details.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"listRequestContainer","description":"listRequestContainer","required":true,"schema":{"$ref":"#/definitions/ListRequestContainer«GtrepPublishConfigParam»"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/GtrepPublishConfigResponseWrapper"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["publish-configuration-controller"],"summary":"Interface for deleting params with given name for a configuration","description":"Use this interface for deleting params with given name for a configuration","operationId":"deleteParamsUsingDELETE","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the project of the generator.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project's version of the generator.","required":true,"type":"integer","format":"int64"},{"name":"generatorId","in":"path","description":"ID of the data generator","required":true,"type":"integer","format":"int64"},{"name":"configurationId","in":"path","description":"ID of the configuration","required":true,"type":"integer","format":"int64"},{"in":"body","name":"listRequestContainer","description":"listRequestContainer","required":true,"schema":{"$ref":"#/definitions/ListRequestContainer«string»"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/GtrepPublishConfigResponseWrapper"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/publishConfigurations/{configurationId}/tables":{"get":{"tags":["publish-configuration-controller"],"summary":"Interface for getting the list of all tables for a configuration","description":"Use this interface to get all tables for a

configuration","operationId":"getTableConfigurationsUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID
of the project of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version
of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"generatorId","in":"path","description":"ID of the data generator where
you want to get the details.","required":true,"type":"integer","format":"int64"},
{"name":"configurationId","in":"path","description":"ID of the configuration where you want to get the
details.","required":true,"type":"integer","format":"int64"},{"name":"page","in":"query","description":"Page
number which you want to retrieve in a paginated Table.","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"Page size of each page with which you want to retrieve in a
paginated table result. Default value is 100. Optional.","required":false,"type":"integer","format":"int32"},
{"name":"sort","in":"query","description":"Name of the field on which you want to sort the paginated
table result.","required":false,"type":"string"},{"name":"order","in":"query","description":"Sorting
direction on which you want to sort the paginated table result. Defaults to Ascending Order.
Optional.","required":false,"type":"string"},{"name":"searchText","in":"query","description":"Search
text you want to use to search on publish configurations table name and schema to get list of
publish configuration tables. Optional.","required":false,"type":"string"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/GtrepPublishConfigTableResponse"}},"400":
{"description":"Bad Request - Request does not have a valid format or has missing required
parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid
or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
- User does not have permissions to access the resource","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error
message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["publish-configuration-
controller"],"summary":"Interface for creation a new tables","description":"Use this interface for creation a
new Tables","operationId":"createTablesUsingPOST","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
authorization scheme to access any protected resource through this API on behalf of the user. For Example:
Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID
of the project of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version
of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"generatorId","in":"path","description":"ID of the data
generator","required":true,"type":"integer","format":"int64"},
{"name":"configurationId","in":"path","description":"ID of the configuration where
you want to get the details.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"publishConfigtables","description":"publishConfigtables","required":true,"schema":
{"$ref":"#/definitions/ListRequestContainer«GtrepPublishConfigTable»"}}],"responses":{"200":
{"description":"OK","schema":{"$ref":"#/definitions/GtrepPublishConfigResponseWrapper"}},"201":
{"description":"Created.","schema":{"$ref":"#/definitions/GtrepPublishConfigResponseWrapper"}},"400":
{"description":"Bad Request - Request does not have a valid format or has missing required
parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid
or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
- User does not have permissions to access the resource","schema":{"$ref":"#/definitions/

ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error
 message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":{"tags":["publish-configuration-
controller"],"summary":"Interface for creation a new tables","description":"Use this interface for creation
 a new Tables","operationId":"updateTablesUsingPUT","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID
 of the project of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version
 of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"generatorId","in":"path","description":"ID of the data
 generator","required":true,"type":"integer","format":"int64"},
{"name":"configurationId","in":"path","description":"ID of the configuration where
 you want to get the details.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"publishConfigTables","description":"publishConfigTables","required":true,"schema":
{"$ref":"#/definitions/ListRequestContainer«GtrepPublishConfigTable»"}}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/GtrepPublishConfigResponseWrapper"}},"400":
{"description":"Bad Request - Request does not have a valid format or has missing required
 parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid
 or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
 - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is
 included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":
{"tags":["publish-configuration-controller"],"summary":"Interface for deleting tables with given
 name for a configuration","description":"Use this interface for deleting tables with given
 configuration","operationId":"deleteTablesUsingDELETE","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID
 of the project of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version
 of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"generatorId","in":"path","description":"ID of the data
 generator","required":true,"type":"integer","format":"int64"},
{"name":"configurationId","in":"path","description":"ID of the
 configuration","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"publishConfigTableIds","description":"publishConfigTableIds","required":true,"schema":
{"$ref":"#/definitions/ListRequestContainer«long»"}}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/GtrepPublishConfigResponseWrapper"}},"400":{"description":"Bad Request - Request
 does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access
 the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found -
 Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/publishConfigurations/{configurationId}/

variables":{"get":{"tags":["publish-configuration-controller"],"summary":"Interface for getting the
 list of all variables for a configuration","description":"Use this interface to get all variables for
 a configuration","operationId":"getAllVariablesForConfigurationsUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID
 of the project of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version
 of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"generatorId","in":"path","description":"ID of the data generator where
 you want to get the details.","required":true,"type":"integer","format":"int64"},
{"name":"configurationId","in":"path","description":"ID of the configuration where
 you want to get the details.","required":true,"type":"integer","format":"int64"},
{"name":"page","in":"query","description":"Page number which you want to retrieve
 in a paginated Variables.","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"Page size of each page with
 which you want to retrieve in a paginated variable result. Default value
 is 100. Optional.","required":false,"type":"integer","format":"int32"},
{"name":"sort","in":"query","description":"Name of the field on which you want to sort the paginated
 variable result.","required":false,"type":"string"},{"name":"order","in":"query","description":"Sorting
 direction on which you want to sort the paginated variable result. Defaults to Ascending Order.
 Optional.","required":false,"type":"string"}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/GtrepPublishConfigVariableResponse"}},"400":{"description":"Bad Request - Request
 does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access
 the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found -
 Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}},"post":{"tags":["publish-configuration-controller"],"summary":"Interface
 for creation a new variables","description":"Use this interface for creation a new
 Variables","operationId":"createVariablesUsingPOST","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID
 of the project of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version
 of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"generatorId","in":"path","description":"ID of the data
 generator","required":true,"type":"integer","format":"int64"},
{"name":"configurationId","in":"path","description":"ID of the configuration where
 you want to get the details.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"listRequestContainer","description":"listRequestContainer","required":true,"schema":
{"$ref":"#/definitions/ListRequestContainer«GtrepPublishConfigVar»"}}],"responses":{"200":
{"description":"OK","schema":{"$ref":"#/definitions/GtrepPublishConfigResponseWrapper"}},"201":
{"description":"Created.","schema":{"$ref":"#/definitions/GtrepPublishConfigResponseWrapper"}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":

{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have
permissions to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":
{"description":"Conflict - A variable with the specified name already exists.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the
error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":{"tags":["publish-configuration-
controller"],"summary":"Interface for updating variables","description":"Use this interface for Updating
Variables","operationId":"updateVariablesUsingPUT","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
authorization scheme to access any protected resource through this API on behalf of the user. For Example:
Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID
of the project of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version
of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"generatorId","in":"path","description":"ID of the data
generator","required":true,"type":"integer","format":"int64"},
{"name":"configurationId","in":"path","description":"ID of the configuration where
you want to get the details.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"listRequestContainer","description":"listRequestContainer","required":true,"schema":
{"$ref":"#/definitions/ListRequestContainer«GtrepPublishConfigVar»"}}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/GtrepPublishConfigResponseWrapper"}},"400":
{"description":"Bad Request - Request does not have a valid format or has missing required
parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid
or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
- User does not have permissions to access the resource","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is
included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":
["publish-configuration-controller"],"summary":"Interface for deleting variable with given name
for a configuration","description":"Use this interface for deleting variable with given name for a
configuration","operationId":"deleteVariablesUsingDELETE","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID
of the project of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project's version
of the generator.","required":true,"type":"integer","format":"int64"},
{"name":"generatorId","in":"path","description":"ID of the data
generator","required":true,"type":"integer","format":"int64"},
{"name":"configurationId","in":"path","description":"ID of the
configuration","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"listRequestContainer","description":"listRequestContainer","required":true,"schema":
{"$ref":"#/definitions/ListRequestContainer«string»"}}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/GtrepPublishConfigResponseWrapper"}},"400":{"description":"Bad Request - Request
does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access
the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource

not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/publishInfo":{"get":{"tags":["data-generator-controller"],"summary":"Interface to get the publish information for a generator","description":"Use this interface to get the publish information for a generator","operationId":"getPublishTableInfoUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"ID of the data generator for which you want to get the publish information","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project to which the generator is associated.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the version to which the generator is associated.","required":true,"type":"integer","format":"int64"},{"name":"page","in":"query","description":"Page number for which you want to retrieve paginated tables result. Indexed with 0. Optional.","required":false,"type":"integer","format":"int32"},{"name":"size","in":"query","description":"Page size of each page with which you want to retrieve in a paginated tables result. Default value is 100. Optional.","required":false,"type":"integer","format":"int32"}],"responses":{"200":{"description":"Success.","schema":{"type":"array","items":{"$ref":"#/definitions/PublishTable"}}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/tables":{"get":{"tags":["data-generator-controller"],"summary":"Interface for getting the details of all registered tables","description":"Use this interface to get the details of all registered tables associated with a given project and version.","operationId":"getTablesUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"ID of the data generator where you want to get the details.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project where the table is registered.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the version where the table is registered.","required":true,"type":"integer","format":"int64"},{"name":"page","in":"query","description":"Page number for which you want to retrieve paginated tables result. Indexed with 0. Optional.","required":false,"type":"integer","format":"int32"},{"name":"size","in":"query","description":"Page size of each page with which you want to retrieve in a paginated tables result. Default value is 100. Optional.","required":false,"type":"integer","format":"int32"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/PaginatedTablesResult"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":

{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/tables/{tableId}":
{"get":{"tags":["data-generator-controller"],"summary":"Interface for getting the details of a specified
 table","description":"Use this interface to get the details of a specified table associated with
 a data generator.","operationId":"getTableUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"generatorId","in":"path","description":"ID of the data generator where you want to get the
 details.","required":true,"type":"integer","format":"int64"},{"name":"tableId","in":"path","description":"ID
 of the table for which you want to get the details.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project where the
 table is registered","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the version where the table is
 registered","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/TableDetails"}},"400":{"description":"Bad
 Request - Request does not have a valid format or has missing required parameters.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired
 token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does
 not have permissions to access the resource.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/tables/{tableId}/
definitionRows":{"get":{"tags":["data-generator-controller"],"summary":"Interface for getting the
 data generator definitions for the table","description":"Use this interface to get the data generator
 definitions for the table.","operationId":"getGeneratorDefinitionsUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"ID of the
 data generator where the table definitions are saved.","required":true,"type":"integer","format":"int64"},
{"name":"tableId","in":"path","description":"ID of the table for which you want
 to get the data definitions.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project where the
 table is registered.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the version where the table is
 registered.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success","schema":{"type":"array","items":{"$ref":"#/definitions/
RowDefinitionDetails"}}},"400":{"description":"Bad Request - Request does not have a valid format
 or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the
 resource.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found -
 Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}},"post":{"tags":["data-generator-controller"],"summary":"Interface for adding data
 generator definitions for a table","description":"Use this interface to add data generator definitions
 for a table.","operationId":"addDefinitionRowsUsingPOST","consumes":["application/json"],"produces":

["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"generatorId","in":"path","description":"ID of the data generator where the data generator
 definitions are to be added for a table.","required":true,"type":"integer","format":"int64"},
{"name":"tableId","in":"path","description":"ID of the table for which the definitions are to be
 added.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID
 of the project where the table is registered.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the version where the
 table is registered.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"rowDefinitionDetails","description":"RowDefinitionDetails object containing
 the array of definitions to be added","required":false,"schema":{"$ref":"#/definitions/
RowDefinitionDetails"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/
RowResult"}},"201":{"description":"Created.","schema":{"$ref":"#/definitions/RowResult"}},"400":
{"description":"Bad Request - Request does not have a valid format or has missing required
 parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid
 or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
 - User does not have permissions to access the resource.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is
 included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/
ca/v1/generators/{generatorId}/tables/{tableId}/definitionRows/{rowId}":{"put":{"tags":["data-
generator-controller"],"summary":"Interface for updating the data generator definitions for a
 table","description":"Use this interface to update the data generator definitions for a given row in
 a table.","operationId":"updateDefinitionRowsUsingPUT","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"generatorId","in":"path","description":"ID of the data generator where the table
 definitions are to be updated.","required":true,"type":"integer","format":"int64"},
{"name":"tableId","in":"path","description":"ID of the table for which you want to update
 the data generator definitions.","required":true,"type":"integer","format":"int64"},
{"name":"rowId","in":"path","description":"ID of the row in the table that
 you want to update.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project where the
 table is registered.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the version where the
 table is registered.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"rowDefinitionDetails","description":"RowDefinitionDetails object containing
 the array of definitions to be updated","required":false,"schema":{"$ref":"#/definitions/
RowDefinitionDetails"}}],"responses":{"200":{"description":"Success.","schema":{"type":"object"}},"400":
{"description":"Bad Request - Request does not have a valid format or has missing required
 parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid
 or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
 - User does not have permissions to access the resource.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is
 included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":
["data-generator-controller"],"summary":"Interface for deleting the data generator definitions of a

table","description":"Use this interface to delete the data generator definitions for a given row of a table.","operationId":"deleteDefinitionRowsUsingDELETE","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"ID of the data generator where the table definitions are saved.","required":true,"type":"integer","format":"int64"},{"name":"tableId","in":"path","description":"ID of the table for which you want to delete the definitions.","required":true,"type":"integer","format":"int64"},{"name":"rowId","in":"path","description":"ID of the row in the table that you want to delete.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project where the table is registered.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the version where the table is registered.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success","schema":{"type":"object"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/usedVariables":{"delete":{"tags":["data-generator-controller"],"summary":"Interface to delete a variable file from the server","description":"Use this interface to delete a variable file which has preciously be validated on the server","operationId":"deleteVariablesFileUsingDELETE","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"ID of the generator for which you want to validate the used variables.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project that is associated to the generator for which you want to validate the used variables.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version that is associated to the generator for which you want to validate the used variables","required":true,"type":"integer","format":"int64"},{"name":"file","in":"query","description":"Variable file to be deleted","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/ProjectResult"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/usedVariables/actions/save":{"post":{"tags":["data-generator-controller"],"summary":"Interface to save used variables to a file.","description":"Use this interface to save the used variables in a generator to file","operationId":"saveUsedVariablesUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},

{"name":"generatorId","in":"path","description":"ID of the generator for which you want to save
the list of data generator used variables.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project that is
associated to the generator for which you want to save the list of data
generator used variables.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version that is
associated to the generator for which you want to save the list of data generator used
variables.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/InputStreamResource"}},"401":
{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the
project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server
Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/usedVariables/actions/validateFromFile":
{"post":{"tags":["data-generator-controller"],"summary":"Interface to validate the variables in
a file ","description":"Use this interface to validate the variables that are mentioned in a
file","operationId":"validateVariablesFromFileUsingPOST","consumes":["multipart/form-data"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login
interface to perform a user login using user credentials in the Basic HTTP authorization scheme.
The API responds with a security token, which is valid for 24 hours by default. Use the security
token in the Bearer HTTP authorization scheme to access any protected resource through this
API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"generatorId","in":"path","description":"ID of the generator for which you want
to validate the used variables.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project that is associated to the generator
for which you want to validate the used variables.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version that is associated to the generator
for which you want to validate the used variables","required":true,"type":"integer","format":"int64"},
{"name":"file","in":"formData","description":"File to be
validated","required":true,"type":"file"}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/ProjectResult"}},"401":{"description":"Unauthorized - Invalid or expired
token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does
not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/usedVariables/actions/
validateFromGenerator":{"post":{"tags":["data-generator-controller"],"summary":"Interface to validate
the variables in a generator ","description":"Use this interface to validate the variables that are
available in a generator","operationId":"validateVariablesFromGeneratorUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"generatorId","in":"path","description":"ID of the generator for which you want
to validate the used variables.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project that is associated to the generator
for which you want to validate the used variables.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version that is associated to the generator
for which you want to validate the used variables","required":true,"type":"integer","format":"int64"},
{"name":"sourceGeneratorId","in":"query","description":"ID of the generator from
which the variable values will be picked (This generator acts as a Variable
Container)","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/ProjectResult"}},"401":{"description":"Unauthorized

```
 - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
 - User does not have permissions to access the project.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error
 message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/generators/{generatorId}/
usedVariables/actions/validateFromSQL":{"post":{"tags":["data-generator-controller"],"summary":"Interface
 to validate the variables using SQL","description":"Use this interface to validate the variables in the
 SQL result set, using the SQL on the target","operationId":"validateVariablesUsingSQLUsingPOST","consumes":
["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"generatorId","in":"path","description":"ID of the generator for which you want
 to validate the used variables.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project that is associated to the generator
 for which you want to validate the used variables.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version that is associated to the generator
 for which you want to validate the used variables","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"body","description":"SQL object information to validate","required":true,"schema":
{"$ref":"#/definitions/ValidateSQLDTO"}}],"responses":{"200":{"description":"Success.","schema":
{"type":"array","items":{"$ref":"#/definitions/VariableFromSourceBean"}}},"400":{"description":"Bad
 Request - Request does not have a valid format or has missing required parameters.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired
 token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does
 not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/VariablesValidationErrorReport"}}}}},"/api/ca/v1/generators/{generatorId}/
variables":{"get":{"tags":["variable-controller"],"summary":"Interface for getting list of all
 generator variables","description":"Use this interface to retrieve the list of all the generator
 variables.","operationId":"getGeneratorVariablesUsingGET","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login
 interface to perform a user login using user credentials in the Basic HTTP authorization scheme.
 The API responds with a security token, which is valid for 24 hours by default. Use the security
 token in the Bearer HTTP authorization scheme to access any protected resource through this
 API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"generatorId","in":"path","description":"ID of the generator for which you want to get
 the list of data generator variables.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project that is
 associated to the generator for which you want to get the list of data
 generator variables.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version that
 is associated to the generator for which you want to get the list of data
 generator variables.","required":true,"type":"integer","format":"int64"},
{"name":"page","in":"query","description":"Page number to retrieve in a paginated variables
 result. Indexed with 0. Optional.","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"Page size of each page with which
 you want to retrieve in a paginated variables result. Default value is 20.
 Optional.","required":false,"type":"integer","format":"int32"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/PaginatedVariableBean"}},"400":{"description":"Bad
 Request - Request does not have a valid format or has missing required parameters.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired
 token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does
```

not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["variable-controller"],"summary":"Interface
to create a new generator variable.","description":"Use this interface to create a new generator
variable.","operationId":"createDataGeneratorVariableUsingPOST","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"Id of the
generator for which variable has to be created.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project where
the generator exists.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the version where
the generator exists.","required":true,"type":"integer","format":"int64"},
{"name":"validate","in":"query","description":"Set this parameter to true to validate the expressions used in
the variable","required":false,"type":"boolean"},{"in":"body","name":"variableInfo","description":"Request
body for creating a variable in a project.\n Mandatory parameters are: \n name: Specify variable
name, accepts strings; \ndescription: Specify variable description, accepts strings; \n defaultValue:
Specify default value for the variable, accepts strings; \nresolvePriorToPublish: Specify if variable
should be resolved prior to publishing, accepts true or false","required":true,"schema":{"$ref":"#/
definitions/VariableBean"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/
VariableBean"}},"201":{"description":"Created.","schema":{"$ref":"#/definitions/VariableBean"}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have
permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":
{"description":"Conflict - A variable with the specified name already exists.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included
in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/generators/
{generatorId}/variables/{variableName}":{"get":{"tags":["variable-controller"],"summary":"Interface
to get details of a variable in a Generator.","description":"Use this interface to get details of a
variable in a Generator.","operationId":"getGeneratorVariableDetailsUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"Id of the
generator for which variable has to be retrieved.","required":true,"type":"integer","format":"int64"},
{"name":"variableName","in":"path","description":"Name of the variable whose details have to be
retrieved.","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID
of the project where the generator exists.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the version where the generator
exists.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/VariableBean"}},"400":{"description":"Bad
Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions
to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found

```
 - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}},"put":{"tags":["variable-controller"],"summary":"Interface to update details
 of a variable in a Generator.","description":"Use this interface to update details of a variable
 in a Generator.","operationId":"updateGeneratorVariableDetailsUsingPUT","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"Id of the
 generator for which variable has to be updated.","required":true,"type":"integer","format":"int64"},
{"name":"variableName","in":"path","description":"Name of the variable whose details have to be
 updated.","required":true,"type":"string"},{"in":"body","name":"variableInfo","description":"Request
 body for creating a variable in a project.\n Mandatory parameters are: \n name: Specify variable
 name, accepts strings; \ndescription: Specify variable description, accepts strings; \n defaultValue:
 Specify default value for the variable, accepts strings; \nresolvePriorToPublish: Specify if
 variable should be resolved prior to publishing, accepts true or false","required":true,"schema":
{"$ref":"#/definitions/VariableBean"}},{"name":"projectId","in":"query","description":"ID of
 the project where the generator exists.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the version where
 the generator exists.","required":true,"type":"integer","format":"int64"},
{"name":"validate","in":"query","description":"Set this parameter to true to validate the
 expressions used in the variable","required":false,"type":"boolean"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/VariableBean"}},"400":{"description":"Bad
 Request - Request does not have a valid format or has missing required parameters.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired
 token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does
 not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["variable-controller"],"summary":"Interface
 to delete generator variables","description":"Use this interface to delete variables in a
 generator","operationId":"deleteGeneratorVariableUsingDELETE","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"generatorId","in":"path","description":"Id of the
 generator for which the variable has to be deleted.","required":true,"type":"integer","format":"int64"},
{"name":"variableName","in":"path","description":"Name of the variable to be
 deleted.","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Id of the
 project for which the variable has to be deleted.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Id of the version for which the variable
 has to be deleted.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"type":"object"}},"400":{"description":"Bad Request - Specific
 reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access
 the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not
 Found - Variable not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
```

{"$ref":"#/definitions/ErrorResponse"}}}}}},"definitions":{"ColumnDefinitionDetails":
{"type":"object","properties":{"columnName":{"type":"string","description":"Name of the
column"},"columnValue":{"type":"string","description":"Value of the column"}}},"ColumnDetails":
{"type":"object","properties":{"columnId":{"type":"integer","format":"int64","description":"ID of the
column","readOnly":true},"columnRules":{"type":"array","description":"List of Rules of the column","items":
{"$ref":"#/definitions/ModelingColumnRuleDetails"}},"dataType":{"type":"string","description":"Datatype
of the column"},"globalDefault":{"type":"string","description":"Global Default of the
column"},"isNullable":{"type":"string","description":"Nullable column"},"localDefault":
{"type":"string","description":"Local Default of the column"},"name":{"type":"string","description":"Name
of the column"},"precision":{"type":"integer","format":"int64","description":"Precision
for a column"},"scale":{"type":"integer","format":"int64","description":"Scale for a
column"},"sequence":{"type":"integer","format":"int64","description":"Sequential number of the
column"}}},"ErrorResponse":{"type":"object","properties":{"errorCode":{"type":"string"},"errorDetail":
{"type":"string"},"errorMsg":{"type":"string"},"status":{"type":"integer","format":"int32"},"timestamp":
{"type":"string"}}},"ExpressionValidatorRequestBean":{"type":"object","properties":{"expression":
{"type":"string"},"levelID":{"type":"integer","format":"int64"},"listTildes":{"type":"array","items":
{"$ref":"#/definitions/TildeValue"}},"metaTable":{"$ref":"#/definitions/MetaTable"},"projectID":
{"type":"integer","format":"int64"},"sourceProfile":{"type":"string"},"targetProfile":
{"type":"string"},"versionID":{"type":"integer","format":"int64"}}},"File":{"type":"object","properties":
{"absolute":{"type":"boolean"},"absoluteFile":{"$ref":"#/definitions/File"},"absolutePath":
{"type":"string"},"canonicalFile":{"$ref":"#/definitions/File"},"canonicalPath":{"type":"string"},"directory":
{"type":"boolean"},"file":{"type":"boolean"},"freeSpace":{"type":"integer","format":"int64"},"hidden":
{"type":"boolean"},"name":{"type":"string"},"parent":{"type":"string"},"parentFile":{"$ref":"#/
definitions/File"},"path":{"type":"string"},"totalSpace":{"type":"integer","format":"int64"},"usableSpace":
{"type":"integer","format":"int64"}}},"ForeignKeyDetails":{"type":"object","properties":
{"referenceTableColumnName":{"type":"string","description":"Foreign key reference Column
name"},"referenceTableName":{"type":"string","description":"Foreign key reference table
name"},"sequence":{"type":"integer","format":"int64","description":"Sequential number of the foreign
Key"},"tableColumnName":{"type":"string","description":"Foreign key table Column name"},"tableName":
{"type":"string","description":"Foreign key table name"}}},"GeneratorFunctionsContainerModel":
{"type":"object","properties":{"dataType":{"type":"string","description":"Datatype of the
evaluated value","readOnly":true},"functions":{"type":"array","description":"List of functions
whose evaluated value is of current datatype","readOnly":true,"items":{"$ref":"#/definitions/
GeneratorFunctionsModel"}}}},"GeneratorFunctionsModel":{"type":"object","properties":{"isDeprecated":
{"type":"boolean","example":false,"description":"Function is deprecated","readOnly":true},"name":
{"type":"string","description":"Name of the function","readOnly":true},"requiresPriorPublishKeyList":
{"type":"boolean","example":false,"description":"Function requires priorpublishkeylist
data","readOnly":true},"requiresSeedList":{"type":"boolean","example":false,"description":"Function requires
seedlist data","readOnly":true},"requiresSqlList":{"type":"boolean","example":false,"description":"Function
requires a connection profile","readOnly":true}}},"GeneratorInfo":{"type":"object","properties":
{"comment":{"type":"string","description":"Comment assigned by the user to the generator"},"created":
{"type":"string","description":"Timestamp of the generator creation"},"description":
{"type":"string","description":"Description of the generator"},"generatorId":
{"type":"number","description":"Id of the generator"},"name":{"type":"string","description":"Name of
the generator"},"onDemand":{"type":"string","description":"Indicates if the generator is available
on demand as service"},"parentId":{"type":"number","description":"Id of the parent"},"projectId":
{"type":"integer","format":"int64","description":"Id of the project to which the generator
belongs to"},"projectName":{"type":"string","description":"Name of the project to which the
generator belongs to"},"type":{"type":"string","description":"Type of the generator"},"updated":
{"type":"string","description":"Timestamp of the last updation of the generator"},"versionId":
{"type":"integer","format":"int64","description":"Id of the version under which the generator is
created"},"versionName":{"type":"string","description":"Name of the version to which the generator belongs
to"}}},"GeneratorResult":{"type":"object","properties":{"comment":{"type":"string","description":"Comment

```
 assigned by the user to the generator"},"created":{"type":"string","description":"Timestamp of the generator
 creation"},"description":{"type":"string","description":"Description of the generator"},"generatorId":
{"type":"number","description":"Id of the generator"},"name":{"type":"string","description":"Name of
 the generator"},"onDemand":{"type":"string","description":"Indicates if the generator is available
 on demand as service"},"parentId":{"type":"number","description":"Id of the parent"},"projectId":
{"type":"integer","format":"int64","description":"Id of the project to which the generator
 belongs to"},"projectName":{"type":"string","description":"Name of the project to which the
 generator belongs to"},"type":{"type":"string","description":"Type of the generator"},"updated":
{"type":"string","description":"Timestamp of the last updation of the generator"},"versionId":
{"type":"integer","format":"int64","description":"Id of the version under which the generator is
 created"},"versionName":{"type":"string","description":"Name of the version to which the generator
 belongs to"}}},"GtrepPublishAction":{"type":"object","properties":{"actionType":{"type":"string"},"code":
{"type":"string"},"codeType":{"type":"string"},"dbConnectionName":{"type":"string"},"description":
{"type":"string"},"name":{"type":"string"},"parameters":{"type":"string"},"programId":
{"type":"integer","format":"int64"},"sequenceId":{"type":"integer","format":"int64"},"successCriterion":
{"type":"string"},"successRequired":{"type":"string"},"tableName":{"type":"string"},"timeout":
{"type":"integer","format":"int64"}}},"GtrepPublishActionOrdering":{"type":"object","properties":
{"name":{"type":"string","description":"Name of the action"},"sequenceId":
{"type":"integer","format":"int64"}}},"GtrepPublishConfig":{"type":"object","properties":
{"configActive":{"type":"string"},"configDescription":{"type":"string"},"configId":
{"type":"integer","format":"int64"},"configName":{"type":"string"}}},"GtrepPublishConfigParam":
{"type":"object","properties":{"paramName":{"type":"string"},"paramValue":
{"type":"string"}}},"GtrepPublishConfigResponseWrapper":{"type":"object","properties":
{"result":{"type":"boolean"}}},"GtrepPublishConfigTable":{"type":"object","properties":
{"actionOnDuplicateInTarget":{"type":"string"},"locationProfile":{"type":"string"},"locationSchema":
{"type":"string"},"locationTable":{"type":"string"},"order":{"type":"integer","format":"int64"},"rowCount":
{"type":"integer","format":"int64"},"schema":{"type":"string"},"tableId":
{"type":"integer","format":"int64"},"tableIncluded":{"type":"string"},"tableName":
{"type":"string"},"tableRepeatCount":{"type":"string"},"tableWhereClause":
{"type":"string"}}},"GtrepPublishConfigTableResponse":{"type":"object","properties":{"elements":
{"type":"array","items":{"$ref":"#/definitions/GtrepPublishConfigTable"}},"numberOfElements":
{"type":"integer","format":"int32"},"totalNumberOfElements":
{"type":"integer","format":"int64"}}},"GtrepPublishConfigVar":{"type":"object","properties":
{"variableDefaultValue":{"type":"string"},"variableGroupId":
{"type":"integer","format":"int64"},"variableListDefinition":{"type":"string"},"variableName":
{"type":"string"},"variableType":{"type":"string"},"variableValue":
{"type":"string"}}},"GtrepPublishConfigVariableResponse":{"type":"object","properties":{"elements":
{"type":"array","items":{"$ref":"#/definitions/GtrepPublishConfigVar"}},"numberOfElements":
{"type":"integer","format":"int32"},"totalNumberOfElements":
{"type":"integer","format":"int64"}}},"InputStream":{"type":"object"},"InputStreamResource":
{"type":"object","properties":{"description":{"type":"string"},"file":{"$ref":"#/
definitions/File"},"filename":{"type":"string"},"inputStream":{"$ref":"#/definitions/
InputStream"},"open":{"type":"boolean"},"readable":{"type":"boolean"},"uri":{"$ref":"#/definitions/
URI"},"url":{"$ref":"#/definitions/URL"}}},"ListRequestContainer«GtrepPublishConfigParam»":
{"type":"object","properties":{"payloadArray":{"type":"array","items":{"$ref":"#/
definitions/GtrepPublishConfigParam"}}}},"ListRequestContainer«GtrepPublishConfigTable»":
{"type":"object","properties":{"payloadArray":{"type":"array","items":{"$ref":"#/
definitions/GtrepPublishConfigTable"}}}},"ListRequestContainer«GtrepPublishConfigVar»":
{"type":"object","properties":{"payloadArray":{"type":"array","items":
{"$ref":"#/definitions/GtrepPublishConfigVar"}}}},"ListRequestContainer«long»":
{"type":"object","properties":{"payloadArray":{"type":"array","items":
{"type":"integer","format":"int64"}}}},"ListRequestContainer«string»":{"type":"object","properties":
{"payloadArray":{"type":"array","items":{"type":"string"}}}},"Map«string,string»":
```

{"type":"object","additionalProperties":{"type":"string"}},"MetaColumn":
{"type":"object","properties":{"columnName":{"type":"string"},"dataType":{"type":"string"},"length":
{"type":"integer","format":"int32"},"scale":{"type":"integer","format":"int32"}}},"MetaTable":
{"type":"object","properties":{"columns":{"type":"array","items":{"$ref":"#/
definitions/MetaColumn"}},"tableName":{"type":"string"}}},"ModelingColumnRuleDetails":
{"type":"object","properties":{"rule":{"type":"string","description":"Rule
 definition"},"ruleId":{"type":"integer","format":"int64","description":"ID of
 the Column rule","readOnly":true},"ruleType":{"type":"string","description":"Rule
 Type"}}},"ObjectEntriesEffected":{"type":"object","properties":{"code":{"type":"string","enum":
["100","101","102","103","200","201","202","203","204","205","206","207","208","226","300","301","302","303","304","305
{"type":"integer","format":"int64"},"objectId":{"type":"integer","format":"int64"},"objectsEffected":
{"type":"integer","format":"int32"},"successMsg":{"type":"string"}}},"PaginatedGeneratorsResult":
{"type":"object","properties":{"elements":{"type":"array","description":"List of actual
 Generator objects","items":{"$ref":"#/definitions/GeneratorInfo"}},"numberOfElements":
{"type":"integer","format":"int64","description":"Number of Generator objects
 retrieved"},"totalElements":{"type":"integer","format":"int64","description":"Total
 number of available Generator objects"}}},"PaginatedResponse«GtrepPublishConfigTable»":
{"type":"object","properties":{"elements":{"type":"array","items":{"$ref":"#/definitions/
GtrepPublishConfigTable"}},"numberOfElements":{"type":"integer","format":"int32"},"totalNumberOfElements":
{"type":"integer","format":"int64"}}},"PaginatedResponse«GtrepPublishConfigVar»":
{"type":"object","properties":{"elements":{"type":"array","items":{"$ref":"#/definitions/
GtrepPublishConfigVar"}},"numberOfElements":{"type":"integer","format":"int32"},"totalNumberOfElements":
{"type":"integer","format":"int64"}}},"PaginatedTablesResult":{"type":"object","properties":
{"numberOfTables":{"type":"integer","format":"int64","description":"Number of tables
 retrieved"},"tables":{"type":"array","description":"List of actual tables retrieved","items":
{"$ref":"#/definitions/Tables"}},"totalTables":{"type":"integer","format":"int64","description":"Total
 number of tables available"}}},"PaginatedVariableBean":{"type":"object","properties":
{"elements":{"type":"array","items":{"$ref":"#/definitions/VariableBean"}},"numberOfElements":
{"type":"integer","format":"int32"},"totalNumberOfElements":
{"type":"integer","format":"int64"}}},"ProjectResult":{"type":"object","properties":
{"message":{"type":"string","description":"Success when API call is successful, error
 otherwise"}}},"PublishActionResult":{"type":"object","properties":{"actionCount":
{"type":"integer","format":"int32"},"actionSucceeded":{"type":"boolean"},"criterionMet":
{"type":"boolean"},"criterionTested":{"type":"boolean"},"warningMsg":
{"type":"string"}}},"PublishConfiguration":{"type":"object","properties":{"actionOnDuplicateInTarget":
{"type":"string","description":"Table level attribute to specify the action to be taken if duplicate data
 found in target"},"locationProfile":{"type":"string","description":"Location profile name"},"locationSchema":
{"type":"string","description":"Location owner"},"locationTable":{"type":"string","description":"Location
 table name"},"storedColumns":{"type":"array","description":"List of stored column
 names","items":{"type":"string"}},"tableId":{"type":"integer","format":"int64","description":"ID
 of the table for which the publish configuration should be updated"},"tableName":
{"type":"string","description":"The name of the table"},"tableRepeatCount":{"type":"string","description":"The
 value of table repeat count"}}},"PublishTable":{"type":"object","properties":
{"actionOnDuplicateInTarget":{"type":"string","description":"Action on duplicate found
 in table"},"fileId":{"type":"integer","format":"int64","description":"File Id of the
 table"},"locationProfile":{"type":"string","description":"location profile name"},"locationSchema":
{"type":"string","description":"location owner"},"locationTable":{"type":"string","description":"location
 table name"},"order":{"type":"integer","format":"int64","description":"Sequence number of the
 table"},"rowCount":{"type":"integer","format":"int64","description":"Number of rows in the
 table"},"storedColumns":{"type":"array","description":"List of stored column names","items":
{"type":"string"}},"tableId":{"type":"integer","format":"int64","description":"Id of the
 table"},"tableName":{"type":"string","description":"Name of the table"},"tableRepeatCount":
{"type":"string","description":"Value of the table repeat count for publish"}}},"RelationshipColumnDetails":

{"type":"object","properties":{"childColumn":{"type":"string","description":"Relationship child column
name"},"parentColumn":{"type":"string","description":"Relationship parent column name"},"sequence":
{"type":"integer","format":"int64","description":"Relationship column sequence"}}},"RelationshipDetails":
{"type":"object","properties":{"childCardinality":{"type":"string","description":"Relationship
child Cardinality"},"childTableName":{"type":"string","description":"Relationship child
table name"},"parentCardinality":{"type":"string","description":"Relationship parent
Cardinality"},"parentTableName":{"type":"string","description":"Relationship parent table
name"},"relationshipColumns":{"type":"array","description":"Relationship Columns","items":{"$ref":"#/
definitions/RelationshipColumnDetails"}}}},"ResponseWrapper«boolean»":{"type":"object","properties":
{"result":{"type":"boolean"}}},"RowDefinitionDetails":{"type":"object","properties":
{"definitions":{"type":"array","description":"List of Column Definitions","items":{"$ref":"#/
definitions/ColumnDefinitionDetails"}},"id":{"type":"integer","format":"int64"},"rowNumber":
{"type":"integer","format":"int64","description":"Sequential number of the row"}}},"RowResult":
{"type":"object","properties":{"rowId":{"type":"integer","format":"int64","description":"Success when
API call is successful, error otherwise"}}},"SeedData":{"type":"object","properties":{"columnDetails":
{"type":"array","description":"Additional column details of the corresponding seed data category. Set if
getColumnDetails is true","items":{"type":"string"}},"columns":{"type":"number","description":"Number of
columns associated with the corresponding seed data category"},"name":{"type":"string","description":"Name
of the seed data category"},"rows":{"type":"number","description":"Number of rows of seed data for
the corresponding seed data category"}}},"TableDetails":{"type":"object","properties":{"columns":
{"type":"array","description":"List of columns of the table","items":{"$ref":"#/definitions/
ColumnDetails"}},"fileId":{"type":"integer","format":"int64","description":"ID of the Parent File under which
this table exists","readOnly":true},"foreignKeys":{"type":"array","description":"List of foreign Keys of
the table","items":{"$ref":"#/definitions/ForeignKeyDetails"}},"name":{"type":"string","description":"Name
of the table"},"order":{"type":"integer","format":"int64","description":"Order of the
table"},"relationships":{"type":"array","description":"List of Relationships of the table","items":
{"$ref":"#/definitions/RelationshipDetails"}},"rowCount":{"type":"integer","format":"int64","description":"No
of rows in the table"},"schema":{"type":"string","description":"Location of the table
(schema)"},"tableId":{"type":"integer","format":"int64","description":"ID of the
table","readOnly":true},"tableProjVersion":{"type":"integer","format":"int64","description":"Version the
table belongs to"}}},"Tables":{"type":"object","properties":{"name":{"type":"string","description":"Name
of the table"},"order":{"type":"integer","format":"int64","description":"Order of the
table"},"rowCount":{"type":"integer","format":"int64","description":"No of rows in the
table"},"schema":{"type":"string","description":"Location of the table (schema)"},"tableId":
{"type":"integer","format":"int64","description":"ID of the table","readOnly":true}}},"TildeValue":
{"type":"object","properties":{"name":{"type":"string"},"preResolveError":{"type":"string"},"preResolveValue":
{"type":"string"},"value":{"type":"string"}}},"URI":{"type":"object","properties":{"absolute":
{"type":"boolean"},"authority":{"type":"string"},"fragment":{"type":"string"},"host":
{"type":"string"},"opaque":{"type":"boolean"},"path":{"type":"string"},"port":
{"type":"integer","format":"int32"},"query":{"type":"string"},"rawAuthority":{"type":"string"},"rawFragment":
{"type":"string"},"rawPath":{"type":"string"},"rawQuery":{"type":"string"},"rawSchemeSpecificPart":
{"type":"string"},"rawUserInfo":{"type":"string"},"scheme":{"type":"string"},"schemeSpecificPart":
{"type":"string"},"userInfo":{"type":"string"}}},"URL":{"type":"object","properties":{"authority":
{"type":"string"},"content":{"type":"object"},"defaultPort":{"type":"integer","format":"int32"},"file":
{"type":"string"},"host":{"type":"string"},"path":{"type":"string"},"port":
{"type":"integer","format":"int32"},"protocol":{"type":"string"},"query":{"type":"string"},"ref":
{"type":"string"},"userInfo":{"type":"string"}}},"ValidateSQLDTO":{"type":"object","required":
["programId","targetType"],"properties":{"targetType":{"type":"string","description":"Type of the
target on which SQL should run on.","enum":["profile","repository"]},"connectionProfileName":
{"type":"string","description":"Name of the connection profile on which you want to run the query on.
Mandatory if 'targetType' is profile"},"programId":{"type":"integer","format":"int64","description":"Saved
SQL program id"},"globalRepeatCount":{"type":"string","description":"Global repeat count expression,
need this value to consider the variables used in the expression as used variables"}}},"VariableBean":

{"type":"object","required":["defaultValue","description","name","type"],"properties":
{"defaultValue":{"type":"string","description":"Default Value of the variable"},"description":
{"type":"string","description":"Description of the variable"},"displayType":
{"type":"string","description":"Display type","enum":
["TextBox","CheckBox","DropDownList","MultiSelectList","RadioButton","DatePicker"]},"helpMessage":
{"type":"string","description":"Help message"},"isDisplayOnly":
{"type":"boolean","example":false,"description":"Variable value is read-only at runtime"},"isOptional":
{"type":"boolean","example":false,"description":"Is Optional"},"listDefinition":
{"type":"string","description":"Definition for list values"},"name":{"type":"string","description":"Name
 of the variable"},"resolvePriorToPublish":{"type":"boolean","example":false,"description":"Resolve
 this variable prior to publish"},"scope":{"type":"string","description":"Scope of the
 variable","readOnly":true},"type":{"type":"string","description":"Type of the variable","enum":
["String","Number","Date","Boolean"]},"validation":{"type":"string","description":"Validation
 rule for the variable value"}}},"VariableFromSourceBean":{"type":"object","properties":{"name":
{"type":"string","description":"Name of the variable"},"values":{"type":"array","description":"List
 of variable values","items":{"type":"string"}}}},"VariablesValidationErrorReport":
{"type":"object","properties":{"errorCode":{"type":"string"},"errorDetail":{"type":"string"},"errorMsg":
{"type":"string"},"report":{"type":"object","description":"Report containing validation error
 details","additionalProperties":{"type":"object","additionalProperties":{"type":"string"}}},"status":
{"type":"integer","format":"int32"},"timestamp":{"type":"string"}}}}}

TDMJobService

alpha

{"swagger":"2.0","info":{"description":"This section includes the APIs that perform various operations
 for scheduling and processing requests.It also provides the REST API URL for the submitting
 requests and getting the status of request.","version":"1.0","title":"CA TDM Job Engine Service
 API","termsOfService":"http://ca.com","contact":{"name":"CA Technologies"},"license":{"name":"The CA License
 Version 2.0","url":"https://ca.com/LICENSE"}},"host":"vtdm-dev-demo:8443","basePath":"/TDMJobService","tags":
[{"name":"job-engine-service-controller","description":"Interface for requests"}],"paths":{"/api/
ca/v1/job/{jobId}":{"get":{"tags":["job-engine-service-controller"],"summary":"Interface getting
 the job information for a single job","description":"Use this interface to retrieve details of
 a single job.","operationId":"getJobDetailsUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"jobId","in":"path","description":"Id of the request for which you want to get details
 of all child requests.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/Job"}},"400":{"description":"Bad
 Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to
 access the Job.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"When a
 job with requested id is not available.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Check logs for more information.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}},"/api/ca/v1/jobs":{"get":{"tags":["job-engine-service-controller"],"summary":"Interface
 for getting all requests","description":"Use this interface to retrieve details of all the
 requests.","operationId":"getAllJobsUsingGET","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds

with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"q","in":"query","description":"List of properties on the resource based on which you want to search or filter. The properties that you can filter on are - origin and jobtype individually. All the properties must be sent as value for 'q' query parameter. Ex: q=(origin='generation') or q=(type='deletion')","required":false,"type":"string"},{"name":"page","in":"query","description":"Page number for which you want to retrieve paginated jobs result. Indexed with 0. Optional.","required":false,"type":"integer","format":"int32"},{"name":"size","in":"query","description":"Page size of each page with which you want to retrieve in a paginated jobs result. Default value is 100. Optional.","required":false,"type":"integer","format":"int32"},{"name":"sort","in":"query","description":"Name of the field on which you want to sort the paginated jobs result. Defaults to Job Id. Optional.","required":false,"type":"string"},{"name":"order","in":"query","description":"Sorting direction on which you want to sort the paginated jobs result. Defaults to Ascending Order. Optional.","required":false,"type":"string"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/PaginatedJobsResult"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"When this REST end point is down or not accessible.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Check logs for more information.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["job-engine-service-controller"],"summary":"Interface for submitting a new request","description":"Use this interface to submit a new request.","operationId":"submitJobUsingPOST","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"jobInfo","description":"Request information you want to use to submit a new request.","required":true,"schema":{"$ref":"#/definitions/Job"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/JobResponse"}},"201":{"description":"Created.","schema":{"$ref":"#/definitions/JobResponse"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"When this REST end point is down or not accessible.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Check logs for more information.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/jobs/{jobId}":{"get":{"tags":["job-engine-service-controller"],"summary":"Interface for getting all child requests","description":"Use this interface to retrieve details of all the child requests of a request.","operationId":"getAllSubJobsUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"jobId","in":"path","description":"Id of the request for which you want to get details of all child requests.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/Job"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the Job.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"When a job with requested id is not available.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Check logs for more information.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/jobs/{jobId}/actions/cancelJob":{"post":{"tags":["job-engine-service-controller"],"summary":"Interface

to cancel a job","description":"Use this interface to cancel jobs which are in a scheduled state. Active jobs cannot be canceled.","operationId":"cancelJobUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"jobId","in":"path","description":"Id of the job for which you want to cancel","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"Id of the project","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"type":"string"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Check logs for more information.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/jobs/{jobId}/actions/downloadArtifact":{"post":{"tags":["job-engine-service-controller"],"summary":"Interface for download an artifact belonging to a request","description":"Use this interface to download an artifact belonging to a request.","operationId":"downloadArtifactsUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"jobId","in":"path","description":"Id of the request/job for which you want to download the artifact.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/InputStreamResource"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the Job.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Available - Requested artifcat is not available.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Check logs for more information.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/jobs/{jobId}/actions/resumeJob":{"post":{"tags":["job-engine-service-controller"],"summary":"Interface to resume a cancelled a job","description":"Use this interface to resume jobs which were previously cancelled.","operationId":"resumeJobUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"jobId","in":"path","description":"Id of the job for which you want resume","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"Id of the project","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"type":"string"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Check logs for more information.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}}},"definitions":{"ErrorResponse":{"type":"object","properties":{"errorCode":{"type":"string"},"errorDetail":{"type":"string"},"errorMsg":{"type":"string"},"status":{"type":"integer","format":"int32"},"timestamp":{"type":"string"}}},"File":{"type":"object","properties":{"absolute":{"type":"boolean"},"absoluteFile":{"$ref":"#/definitions/File"},"absolutePath":{"type":"string"},"canonicalFile":{"$ref":"#/definitions/File"},"canonicalPath":{"type":"string"},"directory":{"type":"boolean"},"file":{"type":"boolean"},"freeSpace":{"type":"integer","format":"int64"},"hidden":

{"type":"boolean"},"name":{"type":"string"},"parent":{"type":"string"},"parentFile":{"$ref":"#/definitions/File"},"path":{"type":"string"},"totalSpace":{"type":"integer","format":"int64"},"usableSpace":{"type":"integer","format":"int64"}}},"InputStream":{"type":"object"},"InputStreamResource":{"type":"object","properties":{"description":{"type":"string"},"file":{"$ref":"#/definitions/File"},"filename":{"type":"string"},"inputStream":{"$ref":"#/definitions/InputStream"},"open":{"type":"boolean"},"readable":{"type":"boolean"},"uri":{"$ref":"#/definitions/URI"},"url":{"$ref":"#/definitions/URL"}}},"Job":{"type":"object","properties":{"artifactLocation":{"type":"string","description":"Location of the artifact related to the Job"},"created":{"type":"string","format":"date-time","description":"Time at which Job was created in the system"},"createdBy":{"type":"string","description":"Name of the user who submitted the Job"},"description":{"type":"string","description":"Description of the Job"},"duration":{"type":"integer","format":"int64","description":"Amount of time taken for completion of the Job"},"email":{"type":"string","description":"Email address to which job execution report is sent"},"endTime":{"type":"string","format":"date-time","description":"Time at which Job execution has completed"},"jobId":{"type":"integer","format":"int64","description":"Id of the Job"},"jobs":{"type":"array","description":"List of child jobs","items":{"$ref":"#/definitions/Job"}},"name":{"type":"string","description":"Name of the Job"},"origin":{"type":"string","description":"Name of the module that has submitted the Job"},"parameters":{"type":"object","description":"Job related parameters"},"parentId":{"type":"integer","format":"int64","description":"Id of the parent Job if the Job is a child job. Zero otherwise"},"projectId":{"type":"integer","format":"int64","description":"Id of the project for which Job is submitted"},"projectName":{"type":"string","description":"Name of the project for which Job is submitted"},"runningStatus":{"type":"string","description":"Indicates the running status of the Job"},"scheduledTime":{"type":"string","format":"date-time","description":"Time for which Job execution is scheduled"},"startTime":{"type":"string","format":"date-time","description":"Time at which Job execution has begun"},"status":{"type":"string","description":"Status of Job execution"},"statusMessage":{"type":"string","description":"Job's status message"},"type":{"type":"string","description":"Type of the Job"},"versionId":{"type":"integer","format":"int64","description":"Version Id of the project for which Job is submitted"}}},"JobResponse":{"type":"object","properties":{"artifactLocation":{"type":"string","description":"Location of the artifact related to the Job"},"created":{"type":"string","format":"date-time","description":"Time at which Job was created in the system"},"createdBy":{"type":"string","description":"Name of the user who submitted the Job"},"description":{"type":"string","description":"Description of the Job"},"duration":{"type":"integer","format":"int64","description":"Amount of time taken for completion of the Job"},"email":{"type":"string","description":"Email address to which job execution report is sent"},"endTime":{"type":"string","format":"date-time","description":"Time at which Job execution has completed"},"jobId":{"type":"integer","format":"int64","description":"Id of the Job"},"jobs":{"type":"array","description":"List of child jobs","items":{"$ref":"#/definitions/Job"}},"name":{"type":"string","description":"Name of the Job"},"origin":{"type":"string","description":"Name of the module that has submitted the Job"},"parameters":{"type":"object","description":"Job related parameters"},"parentId":{"type":"integer","format":"int64","description":"Id of the parent Job if the Job is a child job. Zero otherwise"},"projectId":{"type":"integer","format":"int64","description":"Id of the project for which Job is submitted"},"projectName":{"type":"string","description":"Name of the project for which Job is submitted"},"runningStatus":{"type":"string","description":"Indicates the running status of the Job"},"scheduledTime":{"type":"string","format":"date-time","description":"Time for which Job execution is scheduled"},"startTime":{"type":"string","format":"date-time","description":"Time at which Job execution has begun"},"status":{"type":"string","description":"Status of Job execution"},"statusMessage":{"type":"string","description":"Job's status message"},"type":{"type":"string","description":"Type of the Job"},"versionId":{"type":"integer","format":"int64","description":"Version Id of the project for which Job is submitted"}}},"PaginatedJobsResult":{"type":"object","properties":{"elements":{"type":"array","description":"List of actual jobs retrieved","items":{"$ref":"#/definitions/Job"}},"numberOfElements":{"type":"integer","format":"int64","description":"Number of jobs retrieved"},"totalElements":{"type":"integer","format":"int64","description":"Total number of jobs available"},"totalPages":{"type":"integer","format":"int64"}}},"URI":

{"type":"object","properties":{"absolute":{"type":"boolean"},"authority":{"type":"string"},"fragment":
{"type":"string"},"host":{"type":"string"},"opaque":{"type":"boolean"},"path":
{"type":"string"},"port":{"type":"integer","format":"int32"},"query":{"type":"string"},"rawAuthority":
{"type":"string"},"rawFragment":{"type":"string"},"rawPath":{"type":"string"},"rawQuery":
{"type":"string"},"rawSchemeSpecificPart":{"type":"string"},"rawUserInfo":{"type":"string"},"scheme":
{"type":"string"},"schemeSpecificPart":{"type":"string"},"userInfo":{"type":"string"}}},"URL":
{"type":"object","properties":{"authority":{"type":"string"},"content":{"type":"object"},"defaultPort":
{"type":"integer","format":"int32"},"file":{"type":"string"},"host":{"type":"string"},"path":
{"type":"string"},"port":{"type":"integer","format":"int32"},"protocol":{"type":"string"},"query":
{"type":"string"},"ref":{"type":"string"},"userInfo":{"type":"string"}}}}}

TDMMaskingService

none


```
{
    "swagger": "2.0",
    "info": {
        "description": "This section includes the APIs that perform various operations for masking data.It
 also provides the REST API URL for the respective operation along with sample request and response body
 content.",
        "version": "1.0",
        "title": "CA TDM Masking Service API",
        "termsOfService": "http://ca.com",
        "contact": {
            "name": "CA Technologies"
        },
        "license": {
            "name": "The CA License Version 2.0",
            "url": "https://ca.com/LICENSE"
        }
    },
    "host": "192.168.56.101:8443",
    "basePath": "/TDMMaskingService",
    "tags": [{
            "name": "fdm-controller",
            "description": "FDM Controller"
        }
    ],
    "paths": {
        "/api/ca/v1/masking/functions": {
            "get": {
                "tags": ["fdm-controller"],
                "summary": "Interface to fetch all masking functions from the local instance of the Fast Data
 Masker",
                "operationId": "getFunctionsUsingGET",
                "consumes": ["application/json"],
                "produces": ["*/*"],
                "parameters": [{
                        "name": "Authorization",
                        "in": "header",
```

```
                    "description": "Use the /user/login interface to perform a user login using user
credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid
for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any
protected resource through this API on behalf of the user. For Example: Bearer {{token}}",
                    "required": true,
                    "type": "string"
                }, {
                    "name": "dataType",
                    "in": "query",
                    "description": "The 'dataType' to filter on, can be 'char', 'number', 'date' or
'char_date'.",
                    "required": false,
                    "type": "string"
                }, {
                    "name": "functionName",
                    "in": "query",
                    "description": "The 'functionName' to filter on, can be matched anywhere (i.e. a
'contains' search).",
                    "required": false,
                    "type": "string"
                }, {
                    "name": "page",
                    "in": "query",
                    "description": "The page of data to request, starting from 0.",
                    "required": false,
                    "type": "integer",
                    "default": 0,
                    "format": "int32"
                }, {
                    "name": "size",
                    "in": "query",
                    "description": "The size of the page of data to request.",
                    "required": false,
                    "type": "integer",
                    "default": 50,
                    "format": "int32"
                }
            ],
            "responses": {
                "200": {
                    "description": "OK",
                    "schema": {
                        "$ref": "#/definitions/PagedListResult?Map?string,string??"
                    }
                },
                "401": {
                    "description": "Unauthorized"
                },
                "403": {
                    "description": "Forbidden"
                },
                "404": {
                    "description": "Not Found"
```

```
                }
            }
        }
    },
    "/api/ca/v1/masking/functions/{id}": {
        "get": {
            "tags": ["fdm-controller"],
            "summary": "Interface to fetch details of a single masking function",
            "operationId": "getFunctionInfoUsingGET",
            "consumes": ["application/json"],
            "produces": ["*/*"],
            "parameters": [{
                    "name": "Authorization",
                    "in": "header",
                    "description": "Use the /user/login interface to perform a user login using user
credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid
for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any
protected resource through this API on behalf of the user. For Example: Bearer {{token}}",
                    "required": true,
                    "type": "string"
                }, {
                    "name": "id",
                    "in": "path",
                    "description": "The ID of the masking function to fetch",
                    "required": true,
                    "type": "integer",
                    "format": "int64"
                }
            ],
            "responses": {
                "200": {
                    "description": "OK",
                    "schema": {
                        "type": "object",
                        "additionalProperties": {
                            "type": "string"
                        }
                    }
                },
                "401": {
                    "description": "Unauthorized"
                },
                "403": {
                    "description": "Forbidden"
                },
                "404": {
                    "description": "Not Found"
                }
            }
        }
    },
    "/api/ca/v1/masking/jobs": {
        "delete": {
```

```
                "tags": ["fdm-controller"],
                "summary": "Interface to delete all masking job data for specified project/version",
                "operationId": "deleteDataUsingDELETE",
                "consumes": ["application/json"],
                "produces": ["*/*"],
                "parameters": [{
                        "name": "Authorization",
                        "in": "header",
                        "description": "Use the /user/login interface to perform a user login using user
credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid
for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any
protected resource through this API on behalf of the user. For Example: Bearer {{token}}",
                        "required": true,
                        "type": "string"
                    }, {
                        "name": "projectId",
                        "in": "query",
                        "description": "Project ID.",
                        "required": true,
                        "type": "integer",
                        "format": "int64"
                    }, {
                        "name": "versionId",
                        "in": "query",
                        "description": "Project version ID.",
                        "required": true,
                        "type": "integer",
                        "format": "int64"
                    }
                ],
                "responses": {
                    "200": {
                        "description": "OK",
                        "schema": {
                            "type": "object"
                        }
                    },
                    "204": {
                        "description": "No Content"
                    },
                    "401": {
                        "description": "Unauthorized"
                    },
                    "403": {
                        "description": "Forbidden"
                    }
                }
            }
        },
        "/api/ca/v1/masking/jobs/start": {
            "post": {
                "tags": ["fdm-controller"],
                "summary": "Interface to start a masking job given an environment to mask",
```

```
                "operationId": "startJobUsingPOST",
                "consumes": ["application/json"],
                "produces": ["*/*"],
                "parameters": [{
                        "name": "Authorization",
                        "in": "header",
                        "description": "Use the /user/login interface to perform a user login using user
credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid
for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any
protected resource through this API on behalf of the user. For Example: Bearer {{token}}",
                        "required": true,
                        "type": "string"
                    }, {
                        "in": "body",
                        "name": "params",
                        "description": "params",
                        "required": true,
                        "schema": {
                            "$ref": "#/definitions/PIIMaskingParameters"
                        }
                    }
                ],
                "responses": {
                    "200": {
                        "description": "OK",
                        "schema": {
                            "$ref": "#/definitions/PIIMaskingParameters"
                        }
                    },
                    "201": {
                        "description": "Created"
                    },
                    "401": {
                        "description": "Unauthorized"
                    },
                    "403": {
                        "description": "Forbidden"
                    },
                    "404": {
                        "description": "Not Found"
                    }
                }
            }
        },
        "/api/ca/v1/masking/jobs/startCustom": {
            "post": {
                "tags": ["fdm-controller"],
                "summary": "Interface to start custom masking job given an masking file and connection
profiles",
                "operationId": "startCustomJobUsingPOST",
                "consumes": ["application/json"],
                "produces": ["application/json"],
                "parameters": [{
```

```
                        "name": "Authorization",
                        "in": "header",
                        "description": "Use the /user/login interface to perform a user login using user
credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid
for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any
protected resource through this API on behalf of the user. For Example: Bearer {{token}}",
                        "required": true,
                        "type": "string"
                    }, {
                        "in": "body",
                        "name": "params",
                        "description": "Masking parameters",
                        "required": true,
                        "schema": {
                            "$ref": "#/definitions/MaskingParametersCustom"
                        }
                    }
                ],
                "responses": {
                    "200": {
                        "description": "OK",
                        "schema": {
                            "$ref": "#/definitions/PIIMaskingParameters"
                        }
                    },
                    "201": {
                        "description": "Created"
                    },
                    "401": {
                        "description": "Unauthorized"
                    },
                    "403": {
                        "description": "Forbidden"
                    },
                    "404": {
                        "description": "Not Found"
                    }
                }
            }
        },
        "/api/ca/v1/masking/jobs/{jobId}": {
            "get": {
                "tags": ["fdm-controller"],
                "summary": "Interface to fetch the status of a masking job",
                "operationId": "getStatusUsingGET",
                "consumes": ["application/json"],
                "produces": ["application/json"],
                "parameters": [{
                        "name": "Authorization",
                        "in": "header",
```

```
                        "description": "Use the /user/login interface to perform a user login using user
credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid
for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any
protected resource through this API on behalf of the user. For Example: Bearer {{token}}",
                        "required": true,
                        "type": "string"
                    }, {
                        "name": "jobId",
                        "in": "path",
                        "description": "jobId",
                        "required": true,
                        "type": "integer",
                        "format": "int64"
                    }
                ],
                "responses": {
                    "200": {
                        "description": "OK",
                        "schema": {
                            "$ref": "#/definitions/DBMaskJob"
                        }
                    },
                    "401": {
                        "description": "Unauthorized"
                    },
                    "403": {
                        "description": "Forbidden"
                    },
                    "404": {
                        "description": "Not Found"
                    }
                }
            }
        },
        "/api/ca/v1/masking/jobs/{jobId}/audit": {
            "get": {
                "tags": ["fdm-controller"],
                "summary": "Interface to fetch zipfile containing audit log of the given masking job",
                "operationId": "getAuditZipFileUsingGET",
                "consumes": ["application/json"],
                "produces": ["*/*"],
                "parameters": [{
                        "name": "Authorization",
                        "in": "header",
                        "description": "Use the /user/login interface to perform a user login using user
credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid
for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any
protected resource through this API on behalf of the user. For Example: Bearer {{token}}",
                        "required": true,
                        "type": "string"
                    }, {
                        "name": "jobId",
                        "in": "path",
```

```
                        "description": "jobId",
                        "required": true,
                        "type": "integer",
                        "format": "int64"
                    }
                ],
                "responses": {
                    "200": {
                        "description": "OK"
                    },
                    "401": {
                        "description": "Unauthorized"
                    },
                    "403": {
                        "description": "Forbidden"
                    },
                    "404": {
                        "description": "Not Found"
                    }
                }
            }
        },
        "/api/ca/v1/masking/jobs/{jobId}/preSamples": {
            "delete": {
                "tags": ["fdm-controller"],
                "summary": "Interface to delete all pre-masked sample data for the specified masking job",
                "operationId": "deletePreSamplesUsingDELETE",
                "consumes": ["application/json"],
                "produces": ["*/*"],
                "parameters": [{
                        "name": "Authorization",
                        "in": "header",
                        "description": "Use the /user/login interface to perform a user login using user
credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid
for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any
protected resource through this API on behalf of the user. For Example: Bearer {{token}}",
                        "required": true,
                        "type": "string"
                    }, {
                        "name": "jobId",
                        "in": "path",
                        "description": "jobId",
                        "required": true,
                        "type": "integer",
                        "format": "int64"
                    }
                ],
                "responses": {
                    "200": {
                        "description": "OK",
                        "schema": {
                            "type": "object"
                        }
```

```
                    },
                    "204": {
                        "description": "No Content"
                    },
                    "401": {
                        "description": "Unauthorized"
                    },
                    "403": {
                        "description": "Forbidden"
                    }
                }
            }
        },
        "/api/ca/v1/masking/preSamples": {
            "delete": {
                "tags": ["fdm-controller"],
                "summary": "Interface to delete pre-masked sample data for the specified project/version. If
no project/version is specified then all pre-masked sample data will be deleted.",
                "operationId": "deleteAllPreSamplesUsingDELETE",
                "consumes": ["application/json"],
                "produces": ["*/*"],
                "parameters": [{
                        "name": "Authorization",
                        "in": "header",
                        "description": "Use the /user/login interface to perform a user login using user
credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid
for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any
protected resource through this API on behalf of the user. For Example: Bearer {{token}}",
                        "required": true,
                        "type": "string"
                    }, {
                        "name": "projectId",
                        "in": "query",
                        "description": "Project ID.",
                        "required": true,
                        "type": "integer",
                        "format": "int64"
                    }, {
                        "name": "versionId",
                        "in": "query",
                        "description": "Project version ID.",
                        "required": true,
                        "type": "integer",
                        "format": "int64"
                    }
                ],
                "responses": {
                    "200": {
                        "description": "OK",
                        "schema": {
                            "type": "object"
                        }
                    },
```

```
                    "204": {
                        "description": "No Content"
                    },
                    "401": {
                        "description": "Unauthorized"
                    },
                    "403": {
                        "description": "Forbidden"
                    }
                }
            }
        },
        "/api/ca/v1/masking/scripts": {
            "get": {
                "tags": ["fdm-controller"],
                "summary": "Interface to get pre and post scripts for enabling and disabling triggers and
constraints",
                "operationId": "getScriptsUsingGET",
                "consumes": ["application/json"],
                "produces": ["*/*"],
                "parameters": [{
                        "name": "Authorization",
                        "in": "header",
                        "description": "Use the /user/login interface to perform a user login using user
credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid
for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any
protected resource through this API on behalf of the user. For Example: Bearer {{token}}",
                        "required": true,
                        "type": "string"
                    }, {
                        "name": "projectId",
                        "in": "query",
                        "description": "projectId",
                        "required": true,
                        "type": "integer",
                        "format": "int64"
                    }, {
                        "name": "versionId",
                        "in": "query",
                        "description": "versionId",
                        "required": true,
                        "type": "integer",
                        "format": "int64"
                    }, {
                        "name": "environmentId",
                        "in": "query",
                        "description": "environmentId",
                        "required": true,
                        "type": "integer",
                        "format": "int64"
                    }, {
                        "name": "format",
                        "in": "query",
```

```
                        "description": "format",
                        "required": false,
                        "type": "string"
                    }
                ],
                "responses": {
                    "200": {
                        "description": "OK",
                        "schema": {
                            "type": "array",
                            "items": {
                                "$ref": "#/definitions/PrePostScripts"
                            }
                        }
                    },
                    "401": {
                        "description": "Unauthorized"
                    },
                    "403": {
                        "description": "Forbidden"
                    },
                    "404": {
                        "description": "Not Found"
                    }
                }
            }
        },
        "/api/ca/v1/masking/seedlists": {
            "get": {
                "tags": ["fdm-controller"],
                "summary": "Interface to fetch all seedlists from the local instance of the Fast Data Masker",
                "operationId": "getSeedlistsUsingGET",
                "consumes": ["application/json"],
                "produces": ["*/*"],
                "parameters": [{
                        "name": "Authorization",
                        "in": "header",
                        "description": "Use the /user/login interface to perform a user login using user
credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid
for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any
protected resource through this API on behalf of the user. For Example: Bearer {{token}}",
                        "required": true,
                        "type": "string"
                    }, {
                        "name": "page",
                        "in": "query",
                        "description": "The page of data to request, starting from 0.",
                        "required": false,
                        "type": "integer",
                        "default": 0,
                        "format": "int32"
                    }, {
                        "name": "size",
```

```
                        "in": "query",
                        "description": "The size of the page of data to request.",
                        "required": false,
                        "type": "integer",
                        "default": 50,
                        "format": "int32"
                    }
                ],
                "responses": {
                    "200": {
                        "description": "OK",
                        "schema": {
                            "$ref": "#/definitions/PagedListResult?FDMSeedlistDTO?"
                        }
                    },
                    "401": {
                        "description": "Unauthorized"
                    },
                    "403": {
                        "description": "Forbidden"
                    },
                    "404": {
                        "description": "Not Found"
                    }
                }
            }
        },
        "/api/ca/v1/masking/seedlists/{id}": {
            "get": {
                "tags": ["fdm-controller"],
                "summary": "Interface to fetch details of a single seedlist",
                "operationId": "getSeedlistInfoUsingGET",
                "consumes": ["application/json"],
                "produces": ["*/*"],
                "parameters": [{
                        "name": "Authorization",
                        "in": "header",
                        "description": "Use the /user/login interface to perform a user login using user
credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid
for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any
protected resource through this API on behalf of the user. For Example: Bearer {{token}}",
                        "required": true,
                        "type": "string"
                    }, {
                        "name": "id",
                        "in": "path",
                        "description": "The ID of the seedlist to fetch",
                        "required": true,
                        "type": "integer",
                        "format": "int64"
                    }
                ],
                "responses": {
```

```
                    "200": {
                        "description": "OK",
                        "schema": {
                            "$ref": "#/definitions/FDMSeedlistDTO"
                        }
                    },
                    "401": {
                        "description": "Unauthorized"
                    },
                    "403": {
                        "description": "Forbidden"
                    },
                    "404": {
                        "description": "Not Found"
                    }
                }
            }
        },
        "/api/ca/v1/masking/setup": {
            "get": {
                "tags": ["fdm-controller"],
                "summary": "Interface for getting masking setup for the specified project and version",
                "description": "Returns a default (empty) setup if no masking setup exists for the specified
project id and version id.",
                "operationId": "getMaskingSetupUsingGET",
                "consumes": ["application/json"],
                "produces": ["application/json"],
                "parameters": [{
                        "name": "Authorization",
                        "in": "header",
                        "description": "Use the /user/login interface to perform a user login using user
credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid
for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any
protected resource through this API on behalf of the user. For Example: Bearer {{token}}",
                        "required": true,
                        "type": "string"
                    }, {
                        "name": "projectId",
                        "in": "query",
                        "description": "projectId",
                        "required": true,
                        "type": "integer",
                        "format": "int64"
                    }, {
                        "name": "versionId",
                        "in": "query",
                        "description": "versionId",
                        "required": true,
                        "type": "integer",
                        "format": "int64"
                    }
                ],
                "responses": {
```

```
                    "200": {
                        "description": "OK",
                        "schema": {
                            "$ref": "#/definitions/DBMaskSetup"
                        }
                    },
                    "401": {
                        "description": "Unauthorized"
                    },
                    "403": {
                        "description": "Forbidden"
                    },
                    "404": {
                        "description": "Not Found"
                    }
                }
            },
            "patch": {
                "tags": ["fdm-controller"],
                "summary": "Interface to update masking setup for the specified project and version",
                "description": "The profiler setup will be created if one doesn't already exist for the
specified project id and version id.",
                "operationId": "updateSetupUsingPATCH",
                "consumes": ["application/json"],
                "produces": ["application/json"],
                "parameters": [{
                        "name": "Authorization",
                        "in": "header",
                        "description": "Use the /user/login interface to perform a user login using user
credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid
for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any
protected resource through this API on behalf of the user. For Example: Bearer {{token}}",
                        "required": true,
                        "type": "string"
                    }, {
                        "name": "projectId",
                        "in": "query",
                        "description": "projectId",
                        "required": true,
                        "type": "integer",
                        "format": "int64"
                    }, {
                        "name": "versionId",
                        "in": "query",
                        "description": "versionId",
                        "required": true,
                        "type": "integer",
                        "format": "int64"
                    }, {
                        "in": "body",
                        "name": "setup",
                        "description": "setup",
                        "required": true,
```

```
                    "schema": {
                        "$ref": "#/definitions/DBMaskSetup"
                    }
                }
            ],
            "responses": {
                "200": {
                    "description": "OK",
                    "schema": {
                        "$ref": "#/definitions/DBMaskSetup"
                    }
                },
                "204": {
                    "description": "No Content"
                },
                "401": {
                    "description": "Unauthorized"
                },
                "403": {
                    "description": "Forbidden"
                }
            }
        }
    },
    "/api/ca/v1/masking/status": {
        "get": {
            "tags": ["fdm-controller"],
            "summary": "Interface to get the masking service status (local or remote)",
            "operationId": "statusUsingGET",
            "consumes": ["application/json"],
            "produces": ["*/*"],
            "parameters": [{
                    "name": "Authorization",
                    "in": "header",
                    "description": "Use the /user/login interface to perform a user login using user
credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid
for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any
protected resource through this API on behalf of the user. For Example: Bearer {{token}}",
                    "required": true,
                    "type": "string"
                }
            ],
            "responses": {
                "200": {
                    "description": "OK",
                    "schema": {
                        "$ref": "#/definitions/Status"
                    }
                },
                "401": {
                    "description": "Unauthorized"
                },
                "403": {
```

```
                    "description": "Forbidden"
                },
                "404": {
                    "description": "Not Found"
                }
            }
        }
    }
},
"definitions": {
    "DBMaskJob": {
        "type": "object",
        "properties": {
            "config": {
                "type": "string"
            },
            "environmentId": {
                "type": "integer",
                "format": "int64"
            },
            "jobId": {
                "type": "integer",
                "format": "int64"
            },
            "jobState": {
                "type": "string"
            },
            "previewMode": {
                "type": "boolean"
            },
            "progress": {
                "type": "string"
            },
            "projectId": {
                "type": "integer",
                "format": "int64"
            },
            "startDate": {
                "type": "string",
                "format": "date-time"
            },
            "stopDate": {
                "type": "string",
                "format": "date-time"
            },
            "storePreSamples": {
                "type": "boolean"
            },
            "versionId": {
                "type": "integer",
                "format": "int64"
            }
        }
```

```
},
"DBMaskSetup": {
    "type": "object",
    "properties": {
        "confirmedOnly": {
            "type": "boolean"
        },
        "dataSources": {
            "type": "string"
        },
        "excludeNotPii": {
            "type": "boolean"
        },
        "previewMode": {
            "type": "boolean"
        },
        "projectId": {
            "type": "integer",
            "format": "int64"
        },
        "storePreSamples": {
            "type": "boolean"
        },
        "targetEnv": {
            "type": "integer",
            "format": "int64"
        },
        "versionId": {
            "type": "integer",
            "format": "int64"
        },
        "wholeEnv": {
            "type": "boolean"
        }
    }
},
"FDMSeedlistDTO": {
    "type": "object",
    "properties": {
        "description": {
            "type": "string"
        },
        "extendedDescription": {
            "type": "string"
        },
        "group": {
            "type": "string"
        },
        "id": {
            "type": "integer",
            "format": "int64"
        },
        "name": {
```

```
            "type": "string"
        }
    }
},
"Map?string,string?": {
    "type": "object",
    "additionalProperties": {
        "type": "string"
    }
},
"MaskingParametersCustom": {
    "type": "object",
    "properties": {
        "customConfigFile": {
            "type": "string"
        },
        "customConnectionFile": {
            "type": "string"
        },
        "customOptionsFile": {
            "type": "string"
        },
        "customSeedConnectionFile": {
            "type": "string"
        },
        "jobName": {
            "type": "string"
        },
        "projId": {
            "type": "integer",
            "format": "int64"
        },
        "pverId": {
            "type": "integer",
            "format": "int64"
        }
    }
},
"PIIMaskingParameters": {
    "type": "object",
    "properties": {
        "autoHandleConstraints": {
            "type": "boolean"
        },
        "confirmedOnly": {
            "type": "boolean"
        },
        "connectionProfile": {
            "type": "string"
        },
        "connectionProfiles": {
            "type": "object",
            "additionalProperties": {
```

```
            "type": "string"
        }
    },
    "customConfigFile": {
        "type": "string"
    },
    "customConnectionFile": {
        "type": "string"
    },
    "customMasking": {
        "type": "boolean"
    },
    "customOptionsFile": {
        "type": "string"
    },
    "customSeedConnectionFile": {
        "type": "string"
    },
    "dataSources": {
        "type": "array",
        "items": {
            "type": "string"
        }
    },
    "environmentId": {
        "type": "integer",
        "format": "int64"
    },
    "excNotPii": {
        "type": "boolean"
    },
    "jobId": {
        "type": "integer",
        "format": "int64"
    },
    "jobName": {
        "type": "string"
    },
    "mappings": {
        "type": "array",
        "items": {
            "$ref": "#/definitions/Map?string,string?"
        }
    },
    "previewMode": {
        "type": "boolean"
    },
    "projId": {
        "type": "integer",
        "format": "int64"
    },
    "pverId": {
        "type": "integer",
```

```
            "format": "int64"
        },
        "scheduledTime": {
            "type": "string",
            "format": "date-time"
        },
        "storePreSamples": {
            "type": "boolean"
        },
        "userName": {
            "type": "string"
        }
    }
},
"PagedListResult?FDMSeedlistDTO?": {
    "type": "object",
    "properties": {
        "elements": {
            "type": "array",
            "items": {
                "$ref": "#/definitions/FDMSeedlistDTO"
            }
        },
        "numberOfElements": {
            "type": "integer",
            "format": "int32"
        },
        "totalElements": {
            "type": "integer",
            "format": "int64"
        }
    }
},
"PagedListResult?Map?string,string??": {
    "type": "object",
    "properties": {
        "elements": {
            "type": "array",
            "items": {
                "$ref": "#/definitions/Map?string,string?"
            }
        },
        "numberOfElements": {
            "type": "integer",
            "format": "int32"
        },
        "totalElements": {
            "type": "integer",
            "format": "int64"
        }
    }
},
"PrePostScripts": {
```

```
            "type": "object",
            "properties": {
                "database": {
                    "type": "string"
                },
                "datasourceName": {
                    "type": "string"
                },
                "post": {
                    "type": "string"
                },
                "pre": {
                    "type": "string"
                },
                "profileName": {
                    "type": "string"
                },
                "serverName": {
                    "type": "string"
                }
            }
        },
        "Status": {
            "type": "object",
            "properties": {
                "agentCount": {
                    "type": "integer",
                    "format": "int32"
                },
                "error": {
                    "type": "string"
                },
                "installPath": {
                    "type": "string"
                },
                "installed": {
                    "type": "boolean"
                },
                "minimumVersion": {
                    "type": "string"
                },
                "supported": {
                    "type": "boolean"
                },
                "version": {
                    "type": "string"
                }
            }
        }
    }
}
```

TDMModelService

none

{"swagger":"2.0","info":{"description":"This section includes the APIs that perform various operations for modeling the projects.It also provides the REST API URL for the respective operation along with sample request and response body content.","version":"1.0","title":"CA TDM Modeling Service API","termsOfService":"http://ca.com","contact":{"name":"CA Technologies"},"license":{"name":"The CA License Version 2.0","url":"https://ca.com/LICENSE"}},"host":"vtdm-dev-demo:8443","basePath":"/TDMModelService","tags":[{"name":"results-controller-data-model","description":"Results Controller Data Model"},{"name":"mask-settings-controller","description":"Mask Settings Controller"},{"name":"object-controller","description":"Interface for Objects"},{"name":"profiler-controller","description":"Profiler Controller"},{"name":"results-controller","description":"Results Controller"},{"name":"classifier-controller","description":"Classifier Controller"},{"name":"data-model-controller","description":"Data Model Controller"},{"name":"mask-function-group-controller","description":"Mask Function Group Controller"},{"name":"tags-controller","description":"Tags Controller"},{"name":"job-controller","description":"Job Controller"},{"name":"mask-config-by-tag-controller","description":"Mask Config By Tag Controller"},{"name":"where-clause-controller","description":"Where Clause Controller"}],"paths":{"/api/ca/v1/datamodel":{"delete":{"tags":["data-model-controller"],"summary":"Interface to delete all related data discovery data after a project version has been deleted","operationId":"onProjectVersionDeleteUsingDELETE","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project ID.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"Project version ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":{"type":"boolean"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/api/ca/v1/datamodel/attributes/{attributeId}":{"patch":{"tags":["data-model-controller"],"summary":"Interface to update Data Model Attribute of an Entity. Currently supports only updating of Alias.","operationId":"patchEntityAttributeUsingPATCH","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"attributeId","in":"path","description":"Entity Attribute ID.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"entityAttributeInfo","description":"Entity Attribute data that shall be updated.","required":true,"schema":{"$ref":"#/definitions/DataModelEntityAttributeInfo"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/DataModelEntityAttributeInfo"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/api/ca/v1/datamodel/cancel":{"post":{"tags":["data-model-controller"],"summary":"Interface to cancel a running data discovery scan and clean up all related discovered data","operationId":"cancelDataDiscoveryScanUsingPOST","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer

{{token}}","required":true,"type":"string"},{"name":"environmentId","in":"query","description":"Environment ID.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"Project ID.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"Project version ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":{"type":"boolean"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/datamodel/confirm":{"post":{"tags":["data-model-controller"],"summary":"Interface to delete the most recent failed data discovery scan and related discovered data","operationId":"deleteLastDataDiscoveryScanUsingPOST","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"environmentId","in":"query","description":"Environment ID.","required":false,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"Project ID.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"Project version ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":{"type":"boolean"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/datamodel/copyToVersion":{"post":{"tags":["data-model-controller"],"summary":"Interface to copy a datamodel between versions of a project.","operationId":"copyModelToVersionUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"request","description":"request object contains id of new project","required":false,"schema":{"type":"object","additionalProperties":{"type":"string"}}}],"responses":{"200":{"description":"OK","schema":{"type":"boolean"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/datamodel/discoverRelationships":{"post":{"tags":["data-model-controller"],"summary":"Interface to scan on an environment to collect entity and attribute metadata as well as discover relationships between the entities","operationId":"discoverRelationshipsUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"environmentId","in":"query","description":"Environment ID.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"Project ID.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"Project version ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/DataDiscoveryJobDTO"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/datamodel/entities":{"get":{"tags":["data-model-controller"],"summary":"Interface to get entities and their properties, such as attributes and relationships","operationId":"getAllEntitiesUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use

the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version ID.","required":true,"type":"integer","format":"int64"},
{"name":"includeRelationships","in":"query","description":"Include relationships in response","required":false,"type":"boolean","default":false},
{"name":"includeRelatedEntities","in":"query","description":"Include related entities in response","required":false,"type":"boolean","default":false},
{"name":"includeAttributes","in":"query","description":"Include attributes in response","required":false,"type":"boolean","default":false},
{"name":"includeHierarchy","in":"query","description":"Include hierarchy in response","required":false,"type":"boolean","default":false},
{"name":"includeUniqueKeys","in":"query","description":"Include unique keys in response","required":false,"type":"boolean","default":false},{"name":"page","in":"query","description":"The page of data to request, starting from 0.","required":false,"type":"integer","default":0,"format":"int32"},
{"name":"size","in":"query","description":"The size of the page of data to request.","required":false,"type":"integer","default":20,"format":"int32"},
{"name":"q","in":"query","description":"Query parameter to search. e.g. entity=<value>+attribute=<value>+schema=<value>+database=<value>datasource=<value>","required":false,"type":"string"}],"responses":{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/DataModelEntityInfo"}}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/datamodel/entities/maskConfigurations":{"get":{"tags":["mask-function-group-controller"],"summary":"Interface to get mask configurations of entities","description":"Use this interface to get the mask configurations of entities which are tagged as having PII.","operationId":"getEntitiesMaskInfoUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version ID.","required":true,"type":"integer","format":"int64"},{"name":"page","in":"query","description":"The page of data to request, starting from 0.","required":false,"type":"integer","default":0,"format":"int32"},
{"name":"size","in":"query","description":"The size of the page of data to request.","required":false,"type":"integer","default":20,"format":"int32"},
{"name":"q","in":"query","description":"Search criteria. RSQL format (see https://github.com/jirutka/rsql-parser).Allows the query to be filtered e.g. on 'entityName'","required":false,"type":"string"}],"responses":{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/EntityMaskInfo"}}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/datamodel/entities/{entityId}":{"get":{"tags":["data-model-controller"],"summary":"Interface to get entity and its properties, such as attributes and relationships","operationId":"getEntityUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project ID.","required":true,"type":"integer","format":"int64"},

{"name":"versionId","in":"query","description":"Project version
ID.","required":true,"type":"integer","format":"int64"},{"name":"entityId","in":"path","description":"Entity
ID","required":true,"type":"integer","format":"int64"},
{"name":"includeRelationships","in":"query","description":"Include relationships in
response","required":false,"type":"boolean","default":false},
{"name":"includeRelatedEntities","in":"query","description":"Include related
entities in response","required":false,"type":"boolean","default":false},
{"name":"includeAttributes","in":"query","description":"Include attributes in
response","required":false,"type":"boolean","default":false},
{"name":"includeHierarchy","in":"query","description":"Include hierarchy in
response","required":false,"type":"boolean","default":false},
{"name":"includeUniqueKeys","in":"query","description":"Include unique keys in
response","required":false,"type":"boolean","default":false}],"responses":{"200":
{"description":"OK","schema":{"$ref":"#/definitions/DataModelEntityInfo"}},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}},"patch":
{"tags":["data-model-controller"],"summary":"Interface to update Data Model Entity. Currently
supports only updating of Alias.","operationId":"patchEntityUsingPATCH","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
ID.","required":true,"type":"integer","format":"int64"},{"name":"entityId","in":"path","description":"Entity
ID","required":true,"type":"integer","format":"int64"},{"in":"body","name":"entityInfo","description":"Entity
data that shall be updated.","required":true,"schema":{"$ref":"#/definitions/
DataModelEntityInfo"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/
DataModelEntityInfo"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"}}}},"/api/ca/v1/datamodel/entities/{entityId}/attributes/{attributeId}/
maskConfigurations":{"get":{"tags":["mask-function-group-controller"],"summary":"Interface to get the
current mask configuration of an attribute","description":"Use this interface to get the current mask
configuration of an attribute","operationId":"getAttribMaskFunctionGroupUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
ID.","required":true,"type":"integer","format":"int64"},
{"name":"entityId","in":"path","description":"entityId","required":true,"type":"integer","format":"int64"},
{"name":"attributeId","in":"path","description":"attributeId","required":true,"type":"integer","format":"int64"}],"resp
{"200":{"description":"OK","schema":{"$ref":"#/definitions/MaskFunctionGroup"}},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}},"post":
{"tags":["mask-function-group-controller"],"summary":"Interface to set the mask function group of an
attribute for a project version","description":"Use this interface to set the mask function group of an
attribute for a project version","operationId":"postAttrMaskFunctionGroupUsingPOST","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for

```
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"entityId","in":"path","description":"entityId","required":true,"type":"integer","format":"int64"},
{"name":"attributeId","in":"path","description":"attributeId","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"maskFunctionGroupId","description":"maskFunctionGroupId","required":true,"schema":
{"$ref":"#/definitions/MaskFunctionGroupId"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/
definitions/MaskFunctionGroup"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}},"delete":{"tags":["mask-
function-group-controller"],"summary":"Interface to remove current mask function group from
 an attribute","description":"Use this interface to remove current mask function group from an
 attribute,","operationId":"deleteAttribMaskFunctionGroupUsingDELETE","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"entityId","in":"path","description":"entityId","required":true,"type":"integer","format":"int64"},
{"name":"attributeId","in":"path","description":"attributeId","required":true,"type":"integer","format":"int64"}],"resp
{"200":{"description":"OK","schema":{"type":"boolean"}},"204":{"description":"No Content"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"}}},"patch":{"tags":["mask-function-
group-controller"],"summary":"Interface to update the mask function group of an attribute for a project
 version","description":"Use this interface to update the mask function group of an attribute for
 a project version","operationId":"patchAttrMaskFunctionGroupUsingPATCH","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"entityId","in":"path","description":"entityId","required":true,"type":"integer","format":"int64"},
{"name":"attributeId","in":"path","description":"attributeId","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"maskFunctionGroupId","description":"maskFunctionGroupId","required":true,"schema":
{"$ref":"#/definitions/MaskFunctionGroupId"}}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/MaskFunctionGroup"}},"204":{"description":"No Content"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/api/ca/v1/datamodel/entities/{entityId}/
maskConfigurations":{"get":{"tags":["mask-function-group-controller"],"summary":"Interface to get the
 list of attributes of an entity and their associated current masking configuration","description":"Use
 this interface to get the list of attributes of an entity and their associated current masking
 configuration","operationId":"getEntityMaskFunctionGroupUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
```

```
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"entityId","in":"path","description":"entityId","required":true,"type":"integer","format":"int64"}],"responses"
{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/
AttribMaskFunctionGroup"}}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":
{"description":"Not Found"}}}},"/api/ca/v1/datamodel/entityExclusions":{"get":{"tags":
["data-model-controller"],"summary":"Interface to get entity exclusions for a project
 version","operationId":"getEntityExclusionsUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"OK","schema":{"$ref":"#/definitions/EntityExclusionInfo"}},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not
 Found"}}},"post":{"tags":["data-model-controller"],"summary":"Interface to add entity
 exclusions","operationId":"addEntityExclusionsUsingPOST","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"exclusions","description":"Exclusions to add.","required":true,"schema":
{"type":"array","items":{"$ref":"#/definitions/EntityExclusion"}}}],"responses":{"200":
{"description":"OK","schema":{"$ref":"#/definitions/EntityExclusionInfo"}},"201":
{"description":"Created"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":
{"description":"Not Found"}}},"patch":{"tags":["data-model-controller"],"summary":"Interface to
 patch entity exclusion","operationId":"onPatchEntityExclusionsUsingPATCH","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"updates","description":"Exclusions to add.","required":true,"schema":
{"type":"array","items":{"$ref":"#/definitions/EntityExclusion"}}}],"responses":{"200":
{"description":"OK","schema":{"$ref":"#/definitions/EntityExclusionInfo"}},"204":
{"description":"No Content"},"401":{"description":"Unauthorized"},"403":
```

{"description":"Forbidden"}}}},"/api/ca/v1/datamodel/entityExclusions/{entityExclusionId}":
{"delete":{"tags":["data-model-controller"],"summary":"Interface to delete a single entity
 exclusion","operationId":"onDeleteEntityExclusionUsingDELETE","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"entityExclusionId","in":"path","description":"Entity Exclusion
 ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"type":"boolean"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"}}}},"/api/ca/v1/datamodel/environments/{environmentId}":{"delete":{"tags":
["data-model-controller"],"summary":"Interface to delete all related data discovery data after an
 environment has been deleted","operationId":"onPostEnvironmentDeleteUsingDELETE","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"environmentId","in":"path","description":"Environment
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"type":"boolean"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"}}}},"/api/ca/v1/datamodel/information":{"get":{"tags":["data-
model-controller"],"summary":"Interface to get some data discovery information for a project
 version","operationId":"getDataDiscoveryInfoUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/DataDiscoveryJobDTO"}}},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not
 Found"}}}},"/api/ca/v1/datamodel/maskConfigurations":{"get":{"tags":["mask-function-group-
controller"],"summary":"Interface to get list of mask function groups","description":"Use this
 interface to get list of mask function groups which originate either from classifiers or from the
 specified project version.","operationId":"getMaskFunctionGroupsUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer

{{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project ID.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"Project version ID.","required":true,"type":"integer","format":"int64"},{"name":"attributeId","in":"query","description":"Attribute ID.","required":false,"type":"integer","format":"int64"},{"name":"page","in":"query","description":"The page of data to request, starting from 0.","required":false,"type":"integer","default":0,"format":"int32"},{"name":"size","in":"query","description":"The size of the page of data to request.","required":false,"type":"integer","default":20,"format":"int32"},{"name":"q","in":"query","description":"Search criteria. RSQL format (see https://github.com/jirutka/rsql-parser).Allows the query to be filtered e.g. on 'tagName', 'maskGroupShared' and 'attributeId'","required":false,"type":"string"}],"responses":{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/MaskFunctionGroup"}}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}},"post":{"tags":["mask-function-group-controller"],"summary":"Interface to add a new mask function group to a project version","description":"Use this interface to add a new mask function group to a project version.","operationId":"postMaskFunctionGroupUsingPOST","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project ID.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"Project version ID.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"maskFunctionGroup","description":"maskFunctionGroup","required":true,"schema":{"$ref":"#/definitions/MaskFunctionGroup"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/MaskFunctionGroup"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/datamodel/maskConfigurations/{maskFunctionGroupId}":{"get":{"tags":["mask-function-group-controller"],"summary":"Interface to get the details of a mask function group","description":"Use this interface to get the details of a mask function group","operationId":"getMaskFunctionGroupUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"maskFunctionGroupId","in":"path","description":"maskFunctionGroupId","required":true,"type":"integer","format"{"200":{"description":"OK","schema":{"$ref":"#/definitions/MaskFunctionGroup"}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}},"delete":{"tags":["mask-function-group-controller"],"summary":"Interface to delete a mask function group","description":"Use this interface to delete a mask function group","operationId":"deleteMaskFunctionGroupUsingDELETE","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"maskFunctionGroupId","in":"path","description":"maskFunctionGroupId","required":true,"type":"integer","format"{"200":{"description":"OK","schema":{"type":"boolean"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"}}},"patch":{"tags":["mask-function-group-controller"],"summary":"Interface to update the details of a mask

```
 function group","description":"Use this interface to update the details of a mask function
 group","operationId":"patchMaskFunctionGroupUsingPATCH","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"maskFunctionGroupId","in":"path","description":"maskFunctionGroupId","required":true,"type":"integer","format"
{"in":"body","name":"maskFunctionGroup","description":"maskFunctionGroup","required":true,"schema":{"$ref":"#/
definitions/MaskFunctionGroup"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/
MaskFunctionGroup"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"}}}},"/api/ca/v1/datamodel/maskConfigurations/{maskFunctionGroupId}/
attributes":{"get":{"tags":["mask-function-group-controller"],"summary":"Interface to get
 the list of attributes linked to the specified masking function group","description":"Use
 this interface to get the list of attributes linked to the specified masking function
 group","operationId":"getAttribsMaskFunctionGroupUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"maskFunctionGroupId","in":"path","description":"maskFunctionGroupId","required":true,"type":"integer","format"
{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/
AttribMaskFunctionGroup"}}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":
{"description":"Not Found"}}}},"/api/ca/v1/datamodel/maskSettings":{"get":{"tags":["mask-settings-
controller"],"summary":"Interface to get the user masking settings for specified current project/
version","operationId":"getSettingsUsingGET_1","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"onlyUserSettings","in":"query","description":"Only return settings that the user
 had edited/updated","required":false,"type":"boolean","default":false}],"responses":{"200":
{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/MaskSetting"}}},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not
 Found"}}}},"/api/ca/v1/datamodel/maskSettings/{id}":{"get":{"tags":["mask-settings-
controller"],"summary":"Interface to get the specific user masking setting for specified current project/
version","operationId":"getSettingsUsingGET","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
```

```
 ID.","required":true,"type":"integer","format":"int64"},{"name":"id","in":"path","description":"Id
 of the masking settings to fetch.","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/MaskSetting"}}},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}},"patch":
{"tags":["mask-settings-controller"],"summary":"Interface to set the specific user masking setting for
 specified current project/version","operationId":"postSettingsUsingPATCH","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},{"name":"id","in":"path","description":"Id
 of the masking settings to set.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"setting","description":"The new value of the settings.","required":true,"schema":
{"$ref":"#/definitions/MaskSetting"}}],"responses":{"200":{"description":"OK","schema":
{"type":"array","items":{"$ref":"#/definitions/MaskSetting"}}},"204":{"description":"No Content"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/api/ca/v1/datamodel/preScan":{"post":
{"tags":["data-model-controller"],"summary":"Interface to pre-scan on an environment to collect entity and
 attribute metadata but with no data discovery","operationId":"preScanEnvironmentUsingPOST","consumes":
["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"environmentId","in":"query","description":"Environment
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/DataDiscoveryJobDTO"}},"201":{"description":"Created"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/
api/ca/v1/datamodel/profile":{"post":{"tags":["data-model-controller"],"summary":"Interface to
 perform a profile scan on a data model.","operationId":"profileUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"profileJobRequest","description":"job details.","required":true,"schema":
{"$ref":"#/definitions/Job"}},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/DataDiscoveryJobDTO"}},"201":{"description":"Created"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not
 Found"}}},"delete":{"tags":["data-model-controller"],"summary":"Interface to delete a profile scan on
 a data model.","operationId":"deleteProfileUsingDELETE","consumes":["application/json"],"produces":["*/
*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to
 perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
```

```
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/DataDiscoveryJobDTO"}},"204":{"description":"No Content"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/api/ca/v1/datamodel/profile/actions/
cancelJob":{"post":{"tags":["data-model-controller"],"summary":"Interface to cancel a profile scan job on
 the data model.","operationId":"cancelProfileJobUsingPOST","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"type":"boolean"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/datamodel/profile/
job":{"get":{"tags":["results-controller-data-model"],"summary":"Interface to get a single
 job","operationId":"getAJobUsingGET_1","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/PiiJob"}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":
{"description":"Not Found"}}}},"/api/ca/v1/datamodel/profile/piidata":{"get":{"tags":["results-controller-
data-model"],"summary":"Interface to get the PII data for a job","description":"Returns potential PII
 data for a job a page at a time.","operationId":"getAJobsPiiDataUsingGET_1","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},{"name":"page","in":"query","description":"Page
 number if fetch, starting at 0","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"Page size to
 fetch","required":false,"type":"integer","format":"int32"},
{"name":"hasTags","in":"query","description":"Include columns with PII
 only ?","required":false,"type":"boolean"},{"name":"history","in":"query","description":"Include
 tag history in output, default is false","required":false,"type":"boolean"},
{"name":"q","in":"query","description":"Query parameter to search tags:, tables:, columns:, schema:, profile:
 or all of those","required":false,"type":"string"}],"responses":{"200":{"description":"OK","schema":
{"type":"array","items":{"$ref":"#/definitions/PiiData"}}},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/datamodel/profile/piidata/
```

{table}":{"get":{"tags":["results-controller-data-model"],"summary":"Interface to get the PII data
 for a single table","operationId":"getAJobsPiiDataForOneTableUsingGET_1","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"},
{"name":"history","in":"query","description":"Include tag history in output, default is
 false","required":false,"type":"boolean"}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/
definitions/PiiData"}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":
{"description":"Not Found"}}},"patch":{"tags":["results-controller-data-model"],"summary":"Interface
 to patch the PII data for a table","description":"Use this interface to either accept, reject or
 alter the PII data for a table","operationId":"patchAJobPiiDataUsingPATCH_1","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"},
{"name":"history","in":"query","description":"Include tag history in output, default is
 false","required":false,"type":"boolean"},
{"in":"body","name":"patch","description":"patch","required":true,"schema":{"$ref":"#/
definitions/PiiData"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/
PiiData"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"}}}},"/api/ca/v1/datamodel/profile/piidata/{table}/columns":{"get":{"tags":
["results-controller-data-model"],"summary":"Interface to get the PII data for all columms in a
 table","operationId":"getAJobsPiiDataForOneTableAndColumnsUsingGET_1","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"},
{"name":"page","in":"query","description":"The page of data to request, starting from
 0.","required":false,"type":"integer","format":"int32"},{"name":"size","in":"query","description":"The
 size of the page of data to request.","required":false,"type":"integer","format":"int32"},
{"name":"hasTags","in":"query","description":"Include columns with PII
 only ?","required":false,"type":"boolean"},{"name":"history","in":"query","description":"Include
 tag history in output, default is false","required":false,"type":"boolean"}],"responses":{"200":
{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/PiiDataColumn"}}},"401":

{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}},"put":
{"tags":["results-controller-data-model"],"summary":"Interface to put or patch the PII data for several
columns in a table","description":"Use this interface to either add or remove PII tags for several columns
in a table. Use an Action of REMOVE to remove tags on a PUT call. An action of ADD is implicit on a
PATCH.","operationId":"patchAJobsPiiDataForOneTableAndManyColumnsUsingPUT_1","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
ID.","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"},
{"name":"history","in":"query","description":"Include tag history in output, default is
false","required":false,"type":"boolean"},
{"in":"body","name":"patch","description":"patch","required":true,"schema":{"type":"array","items":
{"$ref":"#/definitions/PiiDataColumn"}}}],"responses":{"200":{"description":"OK","schema":
{"type":"array","items":{"$ref":"#/definitions/PiiDataColumn"}}},"201":{"description":"Created"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}},"patch":
{"tags":["results-controller-data-model"],"summary":"Interface to put or patch the PII data for several
columns in a table","description":"Use this interface to either add or remove PII tags for several columns
in a table. Use an Action of REMOVE to remove tags on a PUT call. An action of ADD is implicit on a
PATCH.","operationId":"patchAJobsPiiDataForOneTableAndManyColumnsUsingPATCH_1","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
ID.","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"},
{"name":"history","in":"query","description":"Include tag history in output, default is
false","required":false,"type":"boolean"},
{"in":"body","name":"patch","description":"patch","required":true,"schema":{"type":"array","items":
{"$ref":"#/definitions/PiiDataColumn"}}}],"responses":{"200":{"description":"OK","schema":
{"type":"array","items":{"$ref":"#/definitions/PiiDataColumn"}}},"204":{"description":"No
Content"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/api/ca/
v1/datamodel/profile/piidata/{table}/columns/{column}":{"put":{"tags":["results-controller-
data-model"],"summary":"Interface to put or patch the PII data for a single columm in a
table","description":"Use this interface to either add or remove PII tags for a single column in
a table. Use an Action of REMOVE to remove tags on a PUT call. An action of ADD is implicit on a
PATCH.","operationId":"patchAJobsPiiDataForOneTableAndColumnsUsingPUT_1","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
ID.","required":true,"type":"integer","format":"int64"},

{"name":"versionId","in":"query","description":"Project version
ID.","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"},
{"name":"column","in":"path","description":"column","required":true,"type":"integer","format":"int64"},
{"name":"history","in":"query","description":"Include tag history in output, default is
false","required":false,"type":"boolean"},
{"in":"body","name":"patch","description":"patch","required":true,"schema":{"$ref":"#/definitions/
PiiDataColumn"}}],"responses":{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/
definitions/PiiDataColumn"}}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}},"patch":{"tags":["results-
controller-data-model"],"summary":"Interface to put or patch the PII data for a single columm in a
table","description":"Use this interface to either add or remove PII tags for a single column in
a table. Use an Action of REMOVE to remove tags on a PUT call. An action of ADD is implicit on a
PATCH.","operationId":"patchAJobsPiiDataForOneTableAndColumnsUsingPATCH_1","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
ID.","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"},
{"name":"column","in":"path","description":"column","required":true,"type":"integer","format":"int64"},
{"name":"history","in":"query","description":"Include tag history in output, default is
false","required":false,"type":"boolean"},
{"in":"body","name":"patch","description":"patch","required":true,"schema":{"$ref":"#/definitions/
PiiDataColumn"}}],"responses":{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/
definitions/PiiDataColumn"}}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"}}}},"/api/ca/v1/datamodel/profile/report":{"get":{"tags":["results-controller-
data-model"],"summary":"Interface to download a PII Job report in CSV format","description":"For report
class DRAFT, the report is generated on the fly. For TDE, AUDITOR or MANAGEMENT, the report is retrieved
from the repository.","operationId":"getAJobAsCsvUsingGET_1","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
authorization scheme to access any protected resource through this API on behalf of the user. For Example:
Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
ID.","required":true,"type":"integer","format":"int64"},{"name":"fileName","in":"query","description":"File
name to be returned in content-disposition","required":false,"type":"string"},
{"name":"format","in":"query","description":"Format of the report
document","required":true,"type":"string","default":"csv","enum":["csv"]}],"responses":
{"200":{"description":"Success.","schema":{"$ref":"#/definitions/InputStreamResource"}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have
permission to access the report.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"When this REST end point is down or not accessible.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Check logs for more
information.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/datamodel/profile/

samples":{"get":{"tags":["results-controller-data-model"],"summary":"Interface to return total
samples stored for ","description":"Samples collected in a job can be fetched for a specified list of
columms.","operationId":"getJobSamplesUsingGET_3","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
authorization scheme to access any protected resource through this API on behalf of the user. For Example:
Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"type":"object","additionalProperties":{"type":"object"}}},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}},"delete":{"tags":
["results-controller-data-model"],"summary":"Interface to delete stored samples from a
job","operationId":"deleteJobSamplesUsingDELETE_1","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
authorization scheme to access any protected resource through this API on behalf of the user. For Example:
Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"type":"boolean"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"}}}},"/api/ca/v1/datamodel/profile/search":{"get":{"tags":
["results-controller-data-model"],"summary":"Interface to search for data within a data
model.","description":"Used this interface to search for tables, columns, schema etc within a data
model.","operationId":"searchUsingGET_1","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
authorization scheme to access any protected resource through this API on behalf of the user. For Example:
Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
ID.","required":true,"type":"integer","format":"int64"},{"name":"q","in":"query","description":"Query
parameter to search for","required":false,"type":"string"},{"name":"max","in":"query","description":"Maximum
number of results to return for each type, default
10","required":false,"type":"integer","format":"int32"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/PiiReviewer"}},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/datamodel/profile/
tables/{table}/rows":{"get":{"tags":["results-controller-data-model"],"summary":"Interface to get raw
data samples from a job","description":"A random set of 10 rows are returned for all columns from a
table.","operationId":"getJobSamplesUsingGET_4","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
authorization scheme to access any protected resource through this API on behalf of the user. For Example:
Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
ID.","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"}],"responses":

{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/
PiiSample"}}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":
{"description":"Not Found"}}}},"/api/ca/v1/datamodel/profile/tables/{table}/samples":{"get":
{"tags":["results-controller-data-model"],"summary":"Interface to get raw data samples from
 a job","description":"Samples collected in a job can be fetched for a specified list of
 columms.","operationId":"getJobSamplesUsingGET_5","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"},
{"name":"columnIds","in":"query","description":"A comma separated list of
 column IDs to fetch sample data for.","required":false,"type":"array","items":
{"type":"integer","format":"int64"},"collectionFormat":"multi"},
{"name":"page","in":"query","description":"Page number if fetch, starting at
 0","required":false,"type":"integer","format":"int32"},{"name":"size","in":"query","description":"Page size
 to fetch","required":false,"type":"integer","format":"int32"},
{"name":"includeTags","in":"query","description":"includeTags","required":false,"type":"boolean"}],"responses":
{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/PiiSample"}}},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/
api/ca/v1/datamodel/profile/tags":{"get":{"tags":["results-controller-data-model"],"summary":"Interface
 to get all PII tags","description":"Use this interface to get a list of all PII tags found or added to
 a data model","operationId":"getTagsUsingGET","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/PiiReviewer"}},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/datamodel/profiler/fdm":
{"get":{"tags":["results-controller-data-model"],"summary":"Interface to download a PII Job in FDM
 configuration format","description":"Fast Data Masker is a masking application, available for Windows
 and Linux. This API downloads the PII job in a format that is suitable to be imported into FDM to mask
 PII data.","operationId":"getAJobAsFDMConfigUsingGET_1","consumes":["application/json"],"produces":["*/
*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to
 perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"confirmedOnly","in":"query","description":"Include confirmed tables
 only","required":false,"type":"boolean"},{"name":"fileName","in":"query","description":"File
 name to be returned in content-disposition","required":false,"type":"string"},
{"name":"excNotPii","in":"query","description":"Exclude tables marked as Not

Pii","required":false,"type":"boolean"},{"name":"environmentId","in":"query","description":"Environment to be masked","required":true,"type":"integer","format":"int64"},
{"name":"dataSources","in":"query","description":"Data sources to be masked","required":false,"type":"array","items":{"type":"string"},"collectionFormat":"multi"},
{"name":"options","in":"query","description":"Options override","required":false,"type":"array","items":{"type":"string"},"collectionFormat":"multi"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/InputStreamResource"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permission to access the report.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"When this REST end point is down or not accessible.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Check logs for more information.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/datamodel/relationships":{"get":{"tags":["data-model-controller"],"summary":"Interface to get relationships for entities in a project version","operationId":"getRelationshipsUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version ID.","required":true,"type":"integer","format":"int64"},{"name":"q","in":"query","description":"Search criteria. TDM and RSQL format (see https://github.com/jirutka/rsql-parser).Allows the query to be filtered on any of the resource's field values, such as 'parentEntityName' or 'childAttributName', etc.","required":false,"type":"string"},{"name":"page","in":"query","description":"The page of data to request, starting from 0.","required":false,"type":"integer","default":0,"format":"int32"},
{"name":"size","in":"query","description":"The size of the page of data to request.","required":false,"type":"integer","default":20,"format":"int32"}],"responses":{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/EntityRelationshipDetails"}}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}},"post":{"tags":["data-model-controller"],"summary":"Interface to add a relationships for entities and attributes","operationId":"createRelationshipUsingPOST","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version ID.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"relationship","description":"Relationship details.","required":true,"schema":{"$ref":"#/definitions/EntityRelationshipDetails"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/EntityRelationshipDetails"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/datamodel/relationships/{relationshipId}":{"get":{"tags":["data-model-controller"],"summary":"Interface to get details for a single relationship in a project version","operationId":"getRelationshipUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP

```
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"relationshipId","in":"path","description":"Relationship
 ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/EntityRelationshipDetails"}},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}},"put":{"tags":["data-
model-controller"],"summary":"Interface to post a relationships for entities and
 attributes","operationId":"updateRelationshipUsingPUT","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"relationshipId","in":"path","description":"Relationship
 ID.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"relationship","description":"Relationship details.","required":true,"schema":
{"$ref":"#/definitions/EntityRelationshipDetails"}}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/EntityRelationshipDetails"}},"201":{"description":"Created"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}},"delete":
{"tags":["data-model-controller"],"summary":"Interface to delete a relationship for entities and
 attributes","operationId":"deleteRelationshipUsingDELETE","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"relationshipId","in":"path","description":"Relationship
 ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/EntityRelationshipDetails"}},"204":{"description":"No Content"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"}}},"patch":{"tags":
["data-model-controller"],"summary":"Interface to patch a relationships for entities and
 attributes","operationId":"patchRelationshipUsingPATCH","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"relationshipId","in":"path","description":"Relationship
```

ID.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"relationship","description":"Relationship details.","required":true,"schema":
{"$ref":"#/definitions/EntityRelationshipDetails"}}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/EntityRelationshipDetails"}},"204":{"description":"No Content"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/api/ca/v1/datamodel/tables":
{"get":{"tags":["object-controller"],"summary":"Interface for getting the tables to register from
the current data model","description":"Use this interface to get the tables to register from
the current data model","operationId":"getDataModelTablesUsingGET","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"projectId","in":"query","description":"ID of the project under which you want to get the
tables. Need this for computing the differences","required":false,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version under which you want to get
tables. Need this for computing the differences","required":false,"type":"integer","format":"int64"},
{"name":"searchText","in":"query","description":"Search Text.","required":false,"type":"string"},
{"name":"page","in":"query","description":"Page number which you want to retrieve. default is
1","required":false,"type":"integer","format":"int32"},{"name":"size","in":"query","description":"Page
size of each page. default is 25","required":false,"type":"integer","format":"int32"}],"responses":
{"200":{"description":"Success.","schema":{"$ref":"#/definitions/TablesInfo"}},"400":{"description":"Bad
Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific
reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":
{"description":"Conflict - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included
in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["object-
controller"],"summary":"Interface for exporting tables from the new data model to the legacy data
model.","description":"Use this interface to copy tables and associated data from the new data model
to the legacy data model","operationId":"postDataModelTablesUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"entityIds","description":"entityIds","required":true,"schema":{"type":"array","items":
{"$ref":"#/definitions/ModelTableInfo"}}},{"name":"projectId","in":"query","description":"ID
of the project under which you want to get the tables. Need this for computing
the differences","required":false,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version under which you want to get tables.
Need this for computing the differences","required":false,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"Success.","schema":{"$ref":"#/definitions/TablesInfo"}},"201":
{"description":"Created"},"400":{"description":"Bad Request - Specific reason is included in the error
message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication
failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":
{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"409":{"description":"Conflict - Specific reason is included in the error
message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error -
Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/
ca/v1/datamodel/tags/maskConfigurations":{"get":{"tags":["mask-config-by-tag-controller"],"summary":"Interface
to get mask information from tags","description":"Use this interface to get the mask information

from all PII tags.","operationId":"getAllTagMaskConfigGroupsUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
ID.","required":true,"type":"integer","format":"int64"},{"name":"page","in":"query","description":"The
page of data to request, starting from 0.","required":false,"type":"integer","default":0,"format":"int32"},
{"name":"size","in":"query","description":"The size of the page of data to
request.","required":false,"type":"integer","default":20,"format":"int32"},
{"name":"q","in":"query","description":"Search criteria. RSQL format (see https://github.com/jirutka/rsql-
parser).Allows the query to be filtered e.g. on 'tagName'","required":false,"type":"string"}],"responses":
{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/
MaskConfigGroupsByTag"}}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":
{"description":"Not Found"}}}},"/api/ca/v1/datamodel/tags/{tagId}/attributes/maskConfigurations":
{"get":{"tags":["mask-config-by-tag-controller"],"summary":"Interface to get masking
group configuration for all attributes of a single tag","description":"Use this
interface to get the masking group configuration for all attributes associated with a PII
tag.","operationId":"getTagsAttributesMaskConfigurationsUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
ID.","required":true,"type":"integer","format":"int64"},{"name":"tagId","in":"path","description":"Tag
ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/MaskConfigGroupsByTag"}},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/datamodel/tags/{tagId}/
maskConfigurations":{"get":{"tags":["mask-config-by-tag-controller"],"summary":"Interface to get mask
information from a single tag","description":"Use this interface to get the mask information from
a single PII tag.","operationId":"getSingleTagMaskConfigGroupsUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
ID.","required":true,"type":"integer","format":"int64"},{"name":"tagId","in":"path","description":"Tag
ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/MaskConfigGroupsByTag"}},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}},"post":{"tags":["mask-config-by-tag-
controller"],"summary":"Interface to add a new masking configuration to current mask configurations
associated with a PII tag for a project version","description":"Use this interface to add a new
masking configuration to current mask configurations associated with a PII tag tag for a project
version.","operationId":"postMaskConfigGroupToTagIdUsingPOST","consumes":["application/json"],"produces":

["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},{"name":"tagId","in":"path","description":"Tag
 ID.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"maskFunctionGroup","description":"maskFunctionGroup","required":true,"schema":{"$ref":"#/
definitions/MaskFunctionGroup"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/
MaskConfigGroupsByTag"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}},"put":{"tags":["mask-config-
by-tag-controller"],"summary":"Interface to get masking group configuration from a single
 tag","description":"Use this interface to get the masking group configuration from a PII
 tag.","operationId":"putSingleTagMaskConfigGroupsUsingPUT","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},{"name":"tagId","in":"path","description":"Tag
 ID.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"maskFunctionGroup","description":"maskFunctionGroup","required":true,"schema":
{"$ref":"#/definitions/MaskFunctionGroup"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/
definitions/MaskFunctionGroup"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}},"delete":{"tags":["mask-config-by-tag-
controller"],"summary":"Interface to delete the mask group configuration for a tag","description":"Use
 this interface to delete the mask group configuration for all attributes assocaited with a PII
 tag.","operationId":"deleteSingleTagMaskConfigGroupsUsingDELETE","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},{"name":"tagId","in":"path","description":"Tag
 ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/MaskConfigGroupsByTag"}},"204":{"description":"No Content"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/api/ca/v1/
datamodel/tags/{tagId}/maskConfigurations/{groupId}":{"get":{"tags":["mask-config-by-tag-
controller"],"summary":"Interface to get masking group configuration from a single tag and
 group","description":"Use this interface to get the specific masking group configuration from a PII
 tag.","operationId":"getSingleTagMaskConfigGroupsByTagIdAndGroupIdUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for

```
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},{"name":"tagId","in":"path","description":"Tag
 ID.","required":true,"type":"integer","format":"int64"},{"name":"groupId","in":"path","description":"Group
 ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/MaskConfigGroupsByTag"}},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}},"put":{"tags":["mask-config-by-
tag-controller"],"summary":"Interface to put a new masking configuration group onto a single
 tag","description":"Use this interface to add a new masking configuration group to a PII
 tag.","operationId":"putSingleTagMaskConfigGroupsByTagIdAndGroupIdUsingPUT","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},{"name":"tagId","in":"path","description":"Tag
 ID.","required":true,"type":"integer","format":"int64"},{"name":"groupId","in":"path","description":"Group
 ID.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"maskFunctionGroup","description":"maskFunctionGroup","required":true,"schema":
{"$ref":"#/definitions/MaskFunctionGroup"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/
definitions/MaskFunctionGroup"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}},"delete":{"tags":["mask-
config-by-tag-controller"],"summary":"Interface to delete a masking configuration on a
 tag","description":"Use this interface to delete a masking configuration group from a PII
 tag.","operationId":"deleteSingleTagMaskConfigGroupsByTagIdAndGroupIdUsingDELETE","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},{"name":"tagId","in":"path","description":"Tag
 ID.","required":true,"type":"integer","format":"int64"},{"name":"groupId","in":"path","description":"Group
 ID.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/MaskConfigGroupsByTag"}},"204":{"description":"No Content"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"}}},"patch":{"tags":["mask-config-
by-tag-controller"],"summary":"Interface to patch the masking function group information on a
 tag","description":"Use this interface to patch the specific masking group configuration on a PII
 tag.","operationId":"patchSingleTagMaskConfigGroupsByTagIdAndGroupIdUsingPATCH","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
```

```
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},{"name":"tagId","in":"path","description":"Tag
 ID.","required":true,"type":"integer","format":"int64"},{"name":"groupId","in":"path","description":"Group
 ID.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"maskFunctionGroup","description":"maskFunctionGroup","required":true,"schema":{"$ref":"#/
definitions/MaskFunctionGroup"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/
MaskFunctionGroup"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"}}}},"/api/ca/v1/datamodel/whereClauses":{"get":{"tags":["where-
clause-controller"],"summary":"Interface to get a set of where clauses","description":"Use
 this interface to get set of where clauses for an attribute, mask function, or mask function
 group.","operationId":"getWhereClausesUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"attributeId","in":"query","description":"Attribute
 ID","required":false,"type":"integer","format":"int64"},
{"name":"maskFunctionId","in":"query","description":"Mask Function
 ID.","required":false,"type":"integer","format":"int64"},
{"name":"maskGroupId","in":"query","description":"Mask Function Group
 ID.","required":false,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/WhereClauseInfo"}},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}},"post":{"tags":["where-clause-
controller"],"summary":"Interface to add a new where clause","description":"Use this interface
 to add a new where clause.","operationId":"postWhereClauseUsingPOST","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"whereClause","description":"whereClause","required":true,"schema":{"$ref":"#/
definitions/WhereClauseInfo"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/
WhereClauseInfo"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}},"patch":{"tags":["where-clause-
controller"],"summary":"Interface to add a new where clause","description":"Use this interface
 to add a new where clause.","operationId":"postWhereClauseUsingPATCH","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project
 ID.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"whereClause","description":"whereClause","required":true,"schema":{"$ref":"#/
definitions/WhereClauseInfo"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/
```

WhereClauseInfo"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"}}}},"/api/ca/v1/datamodel/whereClauses/{id}":{"get":{"tags":["where-clause-
controller"],"summary":"Interface to get a specific where clause","description":"Use this interface
 to get a specific where clause.","operationId":"getWhereClauseUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"id","in":"path","description":"id","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"Project
 ID.","required":false,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":false,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/WhereClauseInfo"}},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}},"delete":{"tags":["where-clause-
controller"],"summary":"Interface to get a specific where clause","description":"Use this interface to
 get a specific where clause.","operationId":"deleteWhereClauseUsingDELETE","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"id","in":"path","description":"id","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"Project
 ID.","required":false,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"Project version
 ID.","required":false,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/WhereClauseInfo"}},"204":{"description":"No Content"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/api/ca/v1/dbmd/{profileName}/
schemas":{"get":{"tags":["object-controller"],"summary":"Interface for getting schemas associated
 with a connection profile","description":"Use this interface to retrieve the list of schemas
 for a given connection profile.","operationId":"getSchemasUsingGET","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"profileName","in":"path","description":"Name of the connection profile for which you want to
 retrieve schemas.","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":
{"type":"array","items":{"type":"string"}}},"401":{"description":"Server authentication failed.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not
 Found"}}}},"/api/ca/v1/jobs/{jobId}/start":{"post":{"tags":["job-controller"],"summary":"Interface to start
 a modeling service job from the Job Engine","operationId":"startJobUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"jobId","in":"path","description":"jobId","required":true,"type":"string"},
{"in":"body","name":"requestBody","description":"requestBody","required":true,"schema":
{"$ref":"#/definitions/DBModellingParameters"}}],"responses":{"200":{"description":"OK","schema":
{"type":"boolean"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":

{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/objects":{"get":
{"tags":["object-controller"],"summary":"Interface for getting objects","description":"Use this
 interface to retrieve the details of all the objects that belong to a specific project and project
 version.","operationId":"getObjectsUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":
[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user
 login using user credentials in the Basic HTTP authorization scheme. The API responds with a security
 token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization
 scheme to access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the
 project for which you want to retrieve objects.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version for which you want
 to retrieve objects.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"type":"array","items":{"$ref":"#/definitions/ObjectDTO"}}},"401":
{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}},"post":{"tags":["object-
controller"],"summary":"Interface for registering a new object","description":"Use this interface to
 register a new object. The following types of objects are supported: XML,XSD,WSDL,RRPAIR,JSON,TABLE
 and CSV.","operationId":"createObjectUsingPOST","consumes":["multipart/form-data"],"produces":["*/
*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to
 perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the
 project that you want to use to create a new object.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version that you want
 to use to create a new object.","required":true,"type":"integer","format":"int64"},
{"name":"body","in":"formData","description":"body","required":true,"type":"ref"},
{"name":"files","in":"formData","description":"List of files to be associate with the object you
 are creating.","required":false,"type":"array","items":{"type":"file"},"collectionFormat":"multi"},
{"name":"requestFile","in":"formData","description":"Request file that you want to associate to the object
 that you are creating. This parameter is valid only for the RRPAIR type.","required":false,"type":"file"},
{"name":"responseFile","in":"formData","description":"Response file that you want to
 associate to the object that you are creating. This parameter is valid only for the RRPAIR
 type.","required":false,"type":"file"}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/ObjectDTO"}},"201":{"description":"Created"},"400":{"description":"Bad
 Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific
 reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":
{"description":"Conflict - Object with object name already exists.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in
 the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["object-
controller"],"summary":"Interface for deleting objects in a version","description":"Use this interface to
 delete objects in a version.","operationId":"deleteObjectsInVersionUsingDELETE","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"projectId","in":"query","description":"ID of the project with the object that
 you want to delete is associated.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version with the object
 that you want to delete is associated.","required":true,"type":"integer","format":"int64"},

{"name":"async","in":"query","description":"Set this attribute to true if you want to perform deleteObject operation asynchronously.","required":false,"type":"boolean"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/ObjectDTO"}},"204":{"description":"No Content - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/objects/actions/delete":{"post":{"tags":["object-controller"],"summary":"Interface for deleting multiple objects","description":"Use this interface to delete multiple objects.","operationId":"deleteObjectsUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"objectIds","description":"IDs of the objects that you want to delete.","required":true,"schema":{"$ref":"#/definitions/ObjectList"}},{"name":"projectId","in":"query","description":"ID of the project with the object that you want to delete is associated.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version with the object that you want to delete is associated.","required":true,"type":"integer","format":"int64"},{"name":"async","in":"query","description":"Set this attribute to true if you want to perform deleteObject operation asynchronously.","required":false,"type":"boolean"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/ObjectDTO"}},"201":{"description":"Created"},"204":{"description":"No Content - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/objects/{objectId}":{"get":{"tags":["object-controller"],"summary":"Interface for getting registered object details","description":"Use this interface to retrieve the details of a specific registered object.","operationId":"getObjectUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"objectId","in":"path","description":"ID of the object that you want to use to get object details.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project that is associated to the object for which you want to get details.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version that is associated to the object for which you want to get details.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/ObjectDTO"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Object with ID not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":{"tags":["object-controller"],"summary":"Interface for modifying object attributes","description":"Use this interface to modify the attributes of an object which is already created.","operationId":"updateObjectUsingPUT","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use

the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"objectId","in":"path","description":"ID of the object to be updated.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project under which the object is already created.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the version of the project under which the object is already created.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"object","description":"Object details.","required":true,"schema":{"$ref":"#/definitions/ObjectRequest"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/ObjectDTO"}},"201":{"description":"Created"},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Object with object Name already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["object-controller"],"summary":"Interface for deleting objects","description":"Use this interface to delete a specific object.","operationId":"deleteObjectUsingDELETE","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"objectId","in":"path","description":"ID of the object that you want to delete.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project with the object that you want to delete is associated.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version with the object that you want to delete is associated.","required":true,"type":"integer","format":"int64"},{"name":"async","in":"query","description":"Set this attribute to true if you want to perform deleteObject operation asynchronously.","required":false,"type":"boolean"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/ObjectDTO"}},"204":{"description":"No Content - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/objects/{objectId}/actions/deleteData":{"post":{"tags":["object-controller"],"summary":"Interface for deleting data in derived objects","description":"Use this interface to delete the data from derived objects. Derived objects are not deleted; only data is deleted.","operationId":"deleteDataUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the project with which the object is registered.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version with which the object is registered.","required":true,"type":"integer","format":"int64"},

{"name":"objectId","in":"path","description":"ID of the registered object for which you want
to delete data in its derived objects.","required":true,"type":"integer","format":"int64"},
{"name":"async","in":"query","description":"Set this attribute to true if you want to perform this operation
asynchronously.","required":false,"type":"boolean"},{"name":"profileName","in":"query","description":"Name
of the connection profile where the derived object from which you want to delete data is
available.","required":false,"type":"string"}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/ObjectEntriesEffected"}},"201":{"description":"Created"},"204":{"description":"No
Content - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific
reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/objects/{objectId}/actions/derive":{"post":{"tags":
["object-controller"],"summary":"Interface for creating and registering tables associated with a registered
object","description":"Use this interface to derive and register tables associated with a registered
object.","operationId":"createAndRegisterTablesUsingPOST","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
authorization scheme to access any protected resource through this API on behalf of the user. For Example:
Bearer {{token}}","required":true,"type":"string"},{"name":"objectId","in":"path","description":"ID of the
object for which you want to create and register tables.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project under which you want
to create and register tables.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version under which you
want to create and register tables.","required":true,"type":"integer","format":"int64"},
{"name":"async","in":"query","description":"Set this attribute to true if you want
to perform this operation asynchronously.","required":false,"type":"boolean"},
{"name":"rootElementName","in":"query","description":"Root element of the object
that is used to create and register tables.","required":false,"type":"string"},
{"name":"generateForiegnKeys","in":"query","description":"Set this attribute to true if you want to
create foreign key constraints on the derived tables of the object.","required":false,"type":"boolean"},
{"name":"reconcile","in":"query","description":"Set this attribute to true if you want to reconcile
table conflicts created from the derived tables of the object.","required":false,"type":"boolean"},
{"name":"tablePrefix","in":"query","description":"Prefix that you want to add to the names of
the derived tables of the object. No default prefix value.","required":false,"type":"string"},
{"name":"duplicateTableSuffix","in":"query","description":"Sufix that you want to add to the names
of the duplicate tables of the object. Default sufix value.","required":false,"type":"string"},
{"name":"cycleRecursionDepth","in":"query","description":"Level up to which you want to create
tables in case of a cyclic references in the schema that the object creates. Its default
value is 2 and maximum value is 32.","required":false,"type":"integer","format":"int32"},
{"name":"wsdlOperation","in":"query","description":"Operation name for which tables have to be created if
the object type is WSDL. It is mandatory if the object type is WSDL.","required":false,"type":"string"},
{"name":"wsdlPortType","in":"query","description":"Port type for which you want to create tables
if the object type is WSDL. If there are multiple operations of same name, port type has to be
provided.","required":false,"type":"string"},{"name":"wsdlPortBinding","in":"query","description":"Port
binding for which you want to create tables if the object type is WSDL. If there are multiple
operations of same name, port binding has to be provided.","required":false,"type":"string"},
{"name":"wsdlPortBindingNameSpace","in":"query","description":"Port binding namespace for which
you want to create tables if the object type is WSDL. If there are conflicting port bindings
of same name, port binding namespace has to be provided.","required":false,"type":"string"},
{"name":"importObjectData","in":"query","description":"Set this attribute to true if you want to
import the data after tables are created and registered. Valid only for XML, JSON, and RRPAIR object

types.","required":false,"type":"boolean"},{"name":"documentGroupId","in":"query","description":"Document group ID of the imported records. This parameter is considered only if the importObjectData attribute is set to true.","required":false,"type":"string"},{"name":"rrPairLinkId","in":"query","description":"ID to map the request and response files of a request-response pair. Valid only for the RRPAIR object type.","required":false,"type":"string"},{"name":"allowComments","in":"query","description":"Set this attribute to true to allow the use of Java or C++ style comments (both '/'+'*' and '//' varieties) within the parsed content for a JSON object. Valid only for the JSON object type.","required":false,"type":"boolean"},{"name":"allowNonNumericValues","in":"query","description":"Set this attribute to true to recognize a set of Not-a-Number (NaN) tokens as valid floating number values for a JSON object. Valid only for the JSON object type.","required":false,"type":"boolean"},{"name":"allowNumericLeadingZeros","in":"query","description":"Set this attribute to true to allow JSON integral numbers to start with additional (ignorable) zeroes (for example, 005). Valid only for the JSON object type.","required":false,"type":"boolean"},{"name":"allowBackSlashEscaping","in":"query","description":"Set this attribute to true to allow the use of backslash to escape any character in JSON content. Valid for object of type JSON.Valid only for the JSON object type.","required":false,"type":"boolean"},{"name":"allowSingleQoutes","in":"query","description":"Set this attribute to true to allow the use of single quotes (apostrophe, character ''') for quoting strings (names and values). Valid only for the JSON object type.","required":false,"type":"boolean"},{"name":"allowUnqoutedControlChars","in":"query","description":"Set this attribute to true to allow JSON strings to contain unquoted control characters (ASCII characters with value less than 32, including tab and line feed characters). Valid only for the JSON object type.","required":false,"type":"boolean"},{"name":"allowUnqoutedFieldNames","in":"query","description":"Set this attribute to true to allow the use of unquoted field names (which is allowed by JavaScript, but not by JSON specification). Valid only for the JSON object type.","required":false,"type":"boolean"},{"name":"profileName","in":"query","description":"Name of the connection profile that identifies the database that you want to use to create derived tables.","required":false,"type":"string"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/ObjectDTO"}},"201":{"description":"Created"},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/objects/{objectId}/actions/export":{"post":{"tags":["object-controller"],"summary":"Interface to export data present in derived objects of a registered object in file formats supported by the registered object. This interface also updates the data to a virtual service","description":"Use this interface to export the data from derived objects (in the database) into supported file formats (XML and JSON). You can also use the interface to export the data into a virtual service for the RRPAIR type.","operationId":"exportDataFromDerivedObjectsUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"objectId","in":"path","description":"ID of the object that you want to use for the export operation.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project associated with the object that you want to use for the export operation.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version associated with the object that you want to use for the export operation.","required":true,"type":"integer","format":"int64"},{"name":"async","in":"query","description":"Set this attribute to true if you want to perform export operation asynchronously.","required":false,"type":"boolean"},{"name":"dataEncoding","in":"query","description":"Encoding format in which the files have to be exported.","required":false,"type":"string"},{"name":"documentGroupId","in":"query","description":"Document

group ID used to perform export operation.","required":false,"type":"string"},
{"name":"rrPairLinkId","in":"query","description":"RRPair link ID used to perform export
operation.","required":false,"type":"string"},{"name":"profileName","in":"query","description":"Name of
the connection profile that you want to use for the export operation.","required":false,"type":"string"},
{"name":"schemaName","in":"query","description":"Name of the schema in the connection
profile that you want to use for the import operation.","required":false,"type":"string"},
{"name":"exportIntoMultipleFiles","in":"query","description":"Set this attribute to true if you want
to export the data into multiple files or else set it to false if you want to export the data into
a single file. By default, it is exported to multiple files.","required":false,"type":"boolean"},
{"name":"baseFileName","in":"query","description":"Base file name of the files that are generated
as a result of the export operation. Default value is CATDM.","required":false,"type":"string"},
{"name":"elementNameForSuffix","in":"query","description":"Specifies the metadata that is
used in the name of the exported XML document. This value must be a name of an element in the
exported XML. For example, If you want to use the first name (<firstname>John</firstname>)
as a suffix, provide firstname without quotes. The exported document then includes John in
its name as <baseFileName>_<documentGroupId>_John.xml.","required":false,"type":"string"},
{"name":"requireDataIndentation","in":"query","description":"Set this attribute to false if you do not
want to indent the XML data while exporting. By default it is true.","required":false,"type":"boolean"},
{"name":"includeXmlDeclaration","in":"query","description":"Set this attribute to false if you do not
want to include the XML declaration in the exported XML files.","required":false,"type":"boolean"},
{"name":"includeStandaloneAttribute","in":"query","description":"Set this attribute to
true to set the standalone attribute to true in the exported XML files. It will take
affect only if includeXmlDeclaration is set to true.","required":false,"type":"boolean"},
{"name":"honorUnqualifiedForms","in":"query","description":"Set this attribute to false if you
do not want to honor the unqualified form for elements in exported XML files. By default, it is
true.","required":false,"type":"boolean"},{"name":"escapeNonASCII","in":"query","description":"Set
this attribute to true to specify that all characters beyond 7-bit ASCII range (that is, code points
of 128 and above) must be exported using format-specific escapes. Valid only for the JSON object
type.","required":false,"type":"boolean"},{"name":"quoteFieldNames","in":"query","description":"Set
this attribute to false if you do not want to quote JSON field names using double quotes,
as specified by JSON specification. By default, it is true. Valid only for the JSON object
type.","required":false,"type":"boolean"},{"name":"quoteNonNumerics","in":"query","description":"Set
this attribute to false if you do not want to output exceptional (not real number) float/double
values as strings using double quotes. By default, it is true. Valid only for the JSON object
type.","required":false,"type":"boolean"},{"name":"writeNumbersAsStrings","in":"query","description":"Set
this attribute to true to write all Java numbers as JSON strings. By default, it
is false. Valid for object of type JSON only.","required":false,"type":"boolean"},
{"name":"prettyPrintJSON","in":"query","description":"Set this attribute to false if
you do not want to format the exported JSON files for better readability. By default,
it is true. Valid for object of type JSON only.","required":false,"type":"boolean"},
{"name":"updateVirtualService","in":"query","description":"Set this attribute to true in order to
update virtual service by importing the exported request and response documents into CA Service
Virtualization. Valid for object of types WSDL and RRPAIR only.","required":false,"type":"boolean"},
{"name":"virtualServiceEnvironment","in":"query","description":"Virtual service
environment that contains the virtual service.","required":false,"type":"string"},
{"name":"virtualService","in":"query","description":"Virtual service to
update.","required":false,"type":"string"},{"name":"publishFiles","in":"query","description":"Set
this attribute to true to perform the publish operation before the export
operation.","required":false,"type":"boolean"},{"name":"generatorId","in":"query","description":"Generator
ID of the data generator that you want to use for the publish operation. This parameter is
applicable only when the publishFiles parameter is set to true.","required":false,"type":"string"},
{"name":"noOfFiles","in":"query","description":"Number of times you want to repeat the publish operation. This
parameter is applicable when publishFiles is set to true.","required":false,"type":"string"}],"responses":

{"200":{"description":"Success.","schema":{"type":"object"}},"201":{"description":"Created"},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific
 reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/objects/{objectId}/actions/getRowCount":{"post":
{"tags":["object-controller"],"summary":"Interface for getting the the data row count of data for
 tables associated with a registered object","description":"Use this interface to get the data row
 count of data of a registered object.","operationId":"getRowCountUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"projectId","in":"query","description":"ID of the project associated with the registered
 object for which you want to find the row count.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version associated with the registered
 object for which you want to find the row count.","required":true,"type":"integer","format":"int64"},
{"name":"schemaName","in":"query","description":"Name of the schema in the connection
 profile that you want to use for the import operation.","required":false,"type":"string"},
{"name":"objectId","in":"path","description":"ID of the object for which you
 want to find the row count.","required":true,"type":"integer","format":"int64"},
{"name":"profileName","in":"query","description":"Connection profile name where the
 tables are present for which you want to determine the row count for a registered
 object.","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/ObjectEntriesEffected"}},"201":{"description":"Created"},"401":
{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":
{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the
 error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}}}}},"/api/ca/v1/objects/{objectId}/actions/getroots":{"post":
{"tags":["object-controller"],"summary":"Interface to retrieve roots defined in registered
 objects","description":"Use this interface to retrieve the root elements defined in a registered
 object.","operationId":"getRootElementsUsingPOST","consumes":["application/json"],"produces":["*/
*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to
 perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"objectId","in":"path","description":"ID of the
 object for which you want to find the root elements.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project associated with the object for
 which you want to find the root elements.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version associated with the object for
 which you want to find the root elements.","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"Success.","schema":{"$ref":"#/definitions/RootElementBean"}},"201":
{"description":"Created"},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific
 reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/objects/{objectId}/actions/import":{"post":
{"tags":["object-controller"],"summary":"Interface for performing import data action on registered
 objects","description":"Use this interface to import the data into derived objects of a registered

object.","operationId":"importDataUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":
[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user
login using user credentials in the Basic HTTP authorization scheme. The API responds with a security
token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization
scheme to access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"objectId","in":"path","description":"ID of the object
for which you want to perform the import operation.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project associated with the object for
which you want to perform the import operation.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version associated with the object for
which you want to perform the import operation.","required":true,"type":"integer","format":"int64"},
{"name":"async","in":"query","description":"Set this attribute to true if you want to
perform import data operation asynchronously.","required":false,"type":"boolean"},
{"name":"dataEncoding","in":"query","description":"Encoding format of the file using which you
want to perform the import operation. Standard character sets include US-ASCII, ISO-8859-1,
UTF-8, UTF-16BE, UTF-16LE, UTF-16. Default value is UTF-8.","required":false,"type":"string"},
{"name":"documentGroupId","in":"query","description":"Document group ID of the imported
documents.","required":false,"type":"string"},{"name":"rrPairLinkId","in":"query","description":"ID
to map the request and response files of a request-response pair. Valid only for the RRPAIR object
type.","required":false,"type":"string"},{"name":"schemaName","in":"query","description":"Name of the schema
in the connection profile that you want to use for the import operation.","required":false,"type":"string"},
{"name":"profileName","in":"query","description":"Name of the connection profile that you want to use for the
import operation.","required":false,"type":"string"},{"name":"files","in":"formData","description":"List
of XML and JSON files to be imported for objects of type XSD,XML and JSON types
respectively.","required":false,"type":"array","items":{"type":"file"},"collectionFormat":"multi"},
{"name":"requestFiles","in":"formData","description":"List of request XML or JSON files to be
imported. Only valid for RRPAIR and WSDL type of object.","required":false,"type":"array","items":
{"type":"file"},"collectionFormat":"multi"},{"name":"responseFiles","in":"formData","description":"List
of response XML or JSON files to be imported. Only valid for RRPAIR and WSDL type of
object.","required":false,"type":"array","items":{"type":"file"},"collectionFormat":"multi"},
{"name":"importToGenerator","in":"query","description":"Set this attribute to true if you
also want to import the data into the data generator.","required":false,"type":"boolean"},
{"name":"generatorId","in":"query","description":"ID of the data generator into which you
want to import the data. This parameter is applicable only when you set importToGenerator
to true.","required":false,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/ObjectDTO"}},"201":{"description":"Created"},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific
reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/objects/{objectId}/actions/unRegister":{"post":
{"tags":["object-controller"],"summary":"Interface for unregistering and dropping derived objects associated
with a registered object","description":"Use this interface to unregister and drop derived objects
associated with a registered object.","operationId":"unRegisterUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"projectId","in":"query","description":"ID of the project with which the derived object
that you want to drop is registered.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version with which the derived

object that you want to drop is registered.","required":true,"type":"integer","format":"int64"},
{"name":"objectId","in":"path","description":"ID of the registered object for which
you want to drop derived objects.","required":true,"type":"integer","format":"int64"},
{"name":"dropTables","in":"query","description":"Set the attribute to true if you
want to drop tables after unregistering them.","required":false,"type":"boolean"},
{"name":"profileName","in":"query","description":"Name of the connection profile
from where you want to drop the derived objects.","required":false,"type":"string"},
{"name":"async","in":"query","description":"Set this attribute to true if you want to perform this operation
asynchronously.","required":false,"type":"boolean"}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/ObjectDTO"}},"201":{"description":"Created"},"400":{"description":"Bad
Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific
reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/objects/{objectId}/derivedObjects":{"get":
{"tags":["object-controller"],"summary":"Interface for getting details of derived objects associated
with a registered object","description":"Use this interface to retrieve the list of derived objects
associated with a registered object.","operationId":"getDerivedObjectsUsingGET","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"objectId","in":"path","description":"ID of the registered object for which you want to
get the list of associated derived objects.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project associated with the registered object
for which derived objects are already created.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version
associated with the registered object for which derived objects are already
created.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/ObjectDTO"}},"401":{"description":"Server
authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/objects/{objectId}/
derivedObjects/{derivedObjectId}":{"get":{"tags":["object-controller"],"summary":"Interface for getting
details of a derived object","description":"Use this interface to retrieve the details of a derived object
associated with a registered object.","operationId":"getDerivedObjectUsingGET","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"objectId","in":"path","description":"ID of the registered object that is associated with the
derived object for which you want to get the details.","required":true,"type":"integer","format":"int64"},
{"name":"projectId","in":"query","description":"ID of the project associated
with the registered object that is related to the derived object for which you
want to get the details.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version associated
with the registered object that is related to the derived object for which you
want to get the details.","required":true,"type":"integer","format":"int64"},
{"name":"derivedObjectId","in":"path","description":"ID of the derived object for which you
want to get the details.","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"Success.","schema":{"$ref":"#/definitions/ObjectDTO"}},"401":

{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":
{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in
 the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/profiler/
classifiers":{"post":{"tags":["classifier-controller"],"summary":"Interface to import classifier
 definitions","description":"Use this interface to import classifier definitions. Definitions should
 be packaged in zip file.","operationId":"importClassifiersUsingPOST","consumes":["multipart/form-
data"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"definitionsFile","in":"formData","description":"Zip file containing classifier hierarchy definition
 to be imported.","required":false,"type":"file"},{"name":"parentId","in":"query","description":"container
 where the definitions should be imoprted.","required":false,"type":"integer","format":"int64"},
{"name":"onduplicate","in":"query","description":"defines behaviour during import, if a duplicate resource
 is found what to do. (ignore, abort, overwrite)","required":false,"type":"string"},{"name":"Accept-
Language","in":"header","description":"Accept-Language","required":false,"type":"string"}],"responses":
{"200":{"description":"OK","schema":{"$ref":"#/definitions/ImportClassifierResponse"}},"201":
{"description":"Created"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":
{"description":"Not Found"}}}},"/api/ca/v1/profiler/classifiers/classifiers":
{"post":{"tags":["classifier-controller"],"summary":"Interface for creating a
 classifier","operationId":"createClassifierUsingPOST","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"classifier","description":"classifier","required":true,"schema":{"$ref":"#/
definitions/Classifier"}},{"name":"Accept-Language","in":"header","description":"Accept-
Language","required":false,"type":"string"}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/
definitions/Classifier"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/profiler/classifiers/
classifiers/{classifierId}":{"get":{"tags":["classifier-controller"],"summary":"Interface for
 getting the list of classifiers and containers which are children of a container specified by
 classifierId.","operationId":"getClassifierUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"classifierId","in":"path","description":"classifierId","required":true,"type":"integer","format":"int64"},
{"name":"Accept-Language","in":"header","description":"Accept-
Language","required":false,"type":"string"}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/
definitions/Classifier"}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":
{"description":"Not Found"}}},"put":{"tags":["classifier-controller"],"summary":"Interface for modifying
 a classifier","operationId":"modifyClassifierUsingPUT","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"classifier","description":"classifier","required":true,"schema":{"$ref":"#/definitions/
Classifier"}},

{"name":"classifierId","in":"path","description":"classifierId","required":true,"type":"integer","format":"int64"}],"re
{"200":{"description":"OK","schema":{"$ref":"#/definitions/Classifier"}},"201":
{"description":"Created"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":
{"description":"Not Found"}}},"delete":{"tags":["classifier-controller"],"summary":"Interface for deleting
 a classifier","operationId":"deleteClassifierUsingDELETE","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"classifierId","in":"path","description":"classifierId","required":true,"type":"integer","format":"int64"}],"re
{"200":{"description":"OK","schema":{"type":"string"}},"204":{"description":"No Content"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/api/ca/v1/profiler/classifiers/
containers/{containerId}":{"get":{"tags":["classifier-controller"],"summary":"Interface for getting the
 container details","operationId":"getContainerUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"containerId","in":"path","description":"containerId","required":true,"type":"integer","format":"int64"},
{"name":"recursive","in":"query","description":"recursive","required":false,"type":"integer","format":"int64"},
{"name":"classifierType","in":"query","description":"classifierType","required":false,"type":"string"},
{"name":"classifierClass","in":"query","description":"classifierClass","required":false,"type":"string"},
{"name":"classifierOrigin","in":"query","description":"classifierOrigin","required":false,"type":"string"},
{"name":"tags","in":"query","description":"tags","required":false,"type":"string"},{"name":"Accept-
Language","in":"header","description":"Accept-Language","required":false,"type":"string"}],"responses":
{"200":{"description":"OK","schema":{"$ref":"#/definitions/ClassifierContainer"}},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not
 Found"}}},"put":{"tags":["classifier-controller"],"summary":"Interface for updating a
 container","operationId":"modifyContainerUsingPUT","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"container","description":"container","required":true,"schema":{"$ref":"#/definitions/
DBClassifierContainer"}},
{"name":"containerId","in":"path","description":"containerId","required":true,"type":"integer","format":"int64"}],"resp
{"200":{"description":"OK","schema":{"$ref":"#/definitions/ClassifierContainer"}},"201":
{"description":"Created"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":
{"description":"Not Found"}}},"delete":{"tags":["classifier-controller"],"summary":"Interface for deleting
 a container","operationId":"deleteContainerUsingDELETE","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"containerId","in":"path","description":"containerId","required":true,"type":"integer","format":"int64"}],"resp
{"200":{"description":"OK","schema":{"type":"string"}},"204":{"description":"No Content"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/api/ca/v1/profiler/classifiers/
containers/{parentId}":{"post":{"tags":["classifier-controller"],"summary":"Interface for creating
 a container","operationId":"createContainerUsingPOST","consumes":["application/json"],"produces":

["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"container","description":"container","required":true,"schema":{"$ref":"#/definitions/
DBClassifierContainer"}},
{"name":"parentId","in":"path","description":"parentId","required":true,"type":"integer","format":"int64"},
{"name":"Accept-Language","in":"query","description":"Accept-
Language","required":false,"type":"string"}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/
definitions/ClassifierContainer"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/profiler/classifiers/
containersclassifiers":{"delete":{"tags":["classifier-controller"],"summary":"Interface for deleting a
 container","operationId":"deleteContainersClassifiersUsingDELETE","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"ids","in":"query","description":"a list of containers and classifiers to be deleted.
 CXXX,MXXX,...","required":false,"type":"string"}],"responses":{"200":{"description":"OK","schema":
{"type":"string"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"}}}},"/api/ca/v1/profiler/classifiers/snapshot/classifiers/{snapshotHash}":{"get":
{"tags":["classifier-controller"],"summary":"Interface for getting the snapshot classifier definition
 by hash code","operationId":"getClassifierSnapshotUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"snapshotHash","in":"path","description":"snapshotHash","required":true,"type":"integer","format":"int32"},
{"name":"Accept-Language","in":"header","description":"Accept-
Language","required":false,"type":"string"}],"responses":{"200":{"description":"OK","schema":
{"type":"array","items":{"$ref":"#/definitions/DBClassifier"}}},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/
profiler/classifiers/snapshot/seedlists/{snapshotHash}":{"get":{"tags":["classifier-
controller"],"summary":"Interface for getting the snapshot seedlist definitions by hash
 code","operationId":"getSeedlistSnapshotUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"snapshotHash","in":"path","description":"snapshotHash","required":true,"type":"integer","format":"int32"},
{"name":"Accept-Language","in":"header","description":"Accept-
Language","required":false,"type":"string"}],"responses":{"200":{"description":"OK","schema":
{"type":"array","items":{"$ref":"#/definitions/SeedListSnapshot"}}},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not
 Found"}}}},"/api/ca/v1/profiler/classifiers/status":{"get":{"tags":["classifier-
controller"],"summary":"getClassifierStatus","operationId":"getClassifierStatusUsingGET","consumes":
["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security

token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/ClassifierStatusResponse"}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/profiler/jobs":{"get":{"tags":["results-controller"],"summary":"Interface to get the profiling jobs","operationId":"getJobsUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"projectId","required":false,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"versionId","required":false,"type":"integer","format":"int64"},{"name":"state","in":"query","description":"state","required":false,"type":"array","items":{"type":"string","enum":["CREATED","STARTED","CANCELLING","CANCELLED","SCAN_COMPLETE","APPROVAL_REQUIRED","APPROVED","APPROVAL_REJECTED","FAILE
["CREATED","STARTED","CANCELLING","CANCELLED","SCAN_COMPLETE","APPROVAL_REQUIRED","APPROVED","APPROVAL_REJECTED","FAILE
{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/PiiJob"}}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/profiler/jobs/":{"delete":{"tags":["results-controller"],"summary":"Interface to delete all profiling jobs for a particular project and version","operationId":"deleteJobsUsingDELETE","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"Project ID to delete","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"Project version ID to delete","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":{"type":"boolean"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/api/ca/v1/profiler/jobs/{jobId}/start":{"post":{"tags":["profiler-controller"],"summary":"Interface to start a PII profiling job from the Job Engine","operationId":"startProfilingJobUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"jobId","in":"path","description":"jobId","required":true,"type":"string"},{"in":"body","name":"requestBody","description":"requestBody","required":true,"schema":{"$ref":"#/definitions/PIIScanParameters"}},{"name":"Authorization","in":"header","description":"Authorization","required":true,"type":"string"}],"responses":{"200":{"description":"OK","schema":{"type":"boolean"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/profiler/jobs/{job}":{"get":{"tags":["results-controller"],"summary":"Interface to get a single job","operationId":"getAJobUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/PiiJob"}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}},"delete":{"tags":["results-controller"],"summary":"Interface to delete a profiling job","operationId":"deleteJobUsingDELETE","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login

interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/PiiJob"}}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"}}},"patch":{"tags":["results-controller"],"summary":"Interface to set the review state for a job","operationId":"patchAJobUsingPATCH","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},{"in":"body","name":"patch","description":"patch","required":true,"schema":{"$ref":"#/definitions/PiiJob"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/PiiJob"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/api/ca/v1/profiler/jobs/{job}/fdm":{"get":{"tags":["results-controller"],"summary":"Interface to download a PII Job in FDM configuration format","description":"Fast Data Masker is a masking application, available for Windows and Linux. This API downloads the PII job in a format that is suitable to be imported into FDM to mask PII data.","operationId":"getAJobAsFDMConfigUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},{"name":"confirmedOnly","in":"query","description":"Include confirmed tables only","required":false,"type":"boolean"},{"name":"fileName","in":"query","description":"File name to be returned in content-disposition","required":false,"type":"string"},{"name":"excNotPii","in":"query","description":"Exclude tables marked as Not Pii","required":false,"type":"boolean"},{"name":"environmentId","in":"query","description":"Environment to be masked","required":true,"type":"integer","format":"int64"},{"name":"dataSources","in":"query","description":"Data sources to be masked","required":false,"type":"array","items":{"type":"string"},"collectionFormat":"multi"},{"name":"options","in":"query","description":"Options override","required":false,"type":"array","items":{"type":"string"},"collectionFormat":"multi"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/InputStreamResource"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permission to access the configuration.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"When this REST end point is down or not accessible.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Check logs for more information.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/profiler/jobs/{job}/piidata":{"get":{"tags":["results-controller"],"summary":"Interface to get the PII data for a job","description":"Returns potential PII data for a job a page at a time.","operationId":"getAJobsPiiDataUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},

{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"page","in":"query","description":"Page number if fetch, starting at
 0","required":false,"type":"integer","format":"int32"},{"name":"size","in":"query","description":"Page size
 to fetch","required":false,"type":"integer","format":"int32"},
{"name":"hasTags","in":"query","description":"Include columns with PII
 only ?","required":false,"type":"boolean"},{"name":"history","in":"query","description":"Include
 tag history in output, default is false","required":false,"type":"boolean"},
{"name":"q","in":"query","description":"Query parameter to search tags:, tables:, columns:, schema:, profile:
 or all of those","required":false,"type":"string"}],"responses":{"200":{"description":"OK","schema":
{"type":"array","items":{"$ref":"#/definitions/PiiData"}}},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/profiler/jobs/{job}/piidata/
{table}":{"get":{"tags":["results-controller"],"summary":"Interface to get the PII data for a single
 table","operationId":"getAJobsPiiDataForOneTableUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"},
{"name":"history","in":"query","description":"Include tag history in output, default is
 false","required":false,"type":"boolean"}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/
definitions/PiiData"}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":
{"description":"Not Found"}}},"patch":{"tags":["results-controller"],"summary":"Interface to patch the
 PII data for a table","description":"Use this interface to either accept, reject or alter the PII data
 for a table","operationId":"patchAJobPiiDataUsingPATCH","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"},
{"name":"history","in":"query","description":"Include tag history in output, default is
 false","required":false,"type":"boolean"},
{"in":"body","name":"patch","description":"patch","required":true,"schema":{"$ref":"#/
definitions/PiiData"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/
PiiData"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"}}}},"/api/ca/v1/profiler/jobs/{job}/piidata/{table}/columns":{"get":
{"tags":["results-controller"],"summary":"Interface to get the PII data for all columms in a
 table","operationId":"getAJobsPiiDataForOneTableAndColumnsUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"},
{"name":"page","in":"query","description":"The page of data to request, starting from
 0.","required":false,"type":"integer","format":"int32"},{"name":"size","in":"query","description":"The
 size of the page of data to request.","required":false,"type":"integer","format":"int32"},
{"name":"hasTags","in":"query","description":"Include columns with PII
 only ?","required":false,"type":"boolean"},{"name":"history","in":"query","description":"Include

```
 tag history in output, default is false","required":false,"type":"boolean"}],"responses":{"200":
{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/PiiDataColumn"}}},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}},"put":
{"tags":["results-controller"],"summary":"Interface to put or patch the PII data for several columns
 in a table","description":"Use this interface to either add or remove PII tags for several columns
 in a table. Use an Action of REMOVE to remove tags on a PUT call. An action of ADD is implicit on a
 PATCH.","operationId":"patchAJobsPiiDataForOneTableAndManyColumnsUsingPUT","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"},
{"name":"history","in":"query","description":"Include tag history in output, default is
 false","required":false,"type":"boolean"},
{"in":"body","name":"patch","description":"patch","required":true,"schema":{"type":"array","items":
{"$ref":"#/definitions/PiiDataColumn"}}}],"responses":{"200":{"description":"OK","schema":
{"type":"array","items":{"$ref":"#/definitions/PiiDataColumn"}}},"201":{"description":"Created"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}},"patch":
{"tags":["results-controller"],"summary":"Interface to put or patch the PII data for several columns
 in a table","description":"Use this interface to either add or remove PII tags for several columns
 in a table. Use an Action of REMOVE to remove tags on a PUT call. An action of ADD is implicit on a
 PATCH.","operationId":"patchAJobsPiiDataForOneTableAndManyColumnsUsingPATCH","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"},
{"name":"history","in":"query","description":"Include tag history in output, default is
 false","required":false,"type":"boolean"},
{"in":"body","name":"patch","description":"patch","required":true,"schema":{"type":"array","items":
{"$ref":"#/definitions/PiiDataColumn"}}}],"responses":{"200":{"description":"OK","schema":
{"type":"array","items":{"$ref":"#/definitions/PiiDataColumn"}}},"204":{"description":"No Content"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/api/ca/v1/profiler/jobs/{job}/piidata/
{table}/columns/{column}":{"put":{"tags":["results-controller"],"summary":"Interface to put or patch the PII
 data for a single columm in a table","description":"Use this interface to either add or remove PII tags for a
 single column in a table. Use an Action of REMOVE to remove tags on a PUT call. An action of ADD is implicit
 on a PATCH.","operationId":"patchAJobsPiiDataForOneTableAndColumnsUsingPUT","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"},
{"name":"column","in":"path","description":"column","required":true,"type":"integer","format":"int64"},
{"name":"history","in":"query","description":"Include tag history in output, default is
 false","required":false,"type":"boolean"},
{"in":"body","name":"patch","description":"patch","required":true,"schema":{"$ref":"#/definitions/
PiiDataColumn"}}],"responses":{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/
```

definitions/PiiDataColumn"}}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}},"patch":{"tags":["results-
controller"],"summary":"Interface to put or patch the PII data for a single columm in a
 table","description":"Use this interface to either add or remove PII tags for a single column in
 a table. Use an Action of REMOVE to remove tags on a PUT call. An action of ADD is implicit on a
 PATCH.","operationId":"patchAJobsPiiDataForOneTableAndColumnsUsingPATCH","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"},
{"name":"column","in":"path","description":"column","required":true,"type":"integer","format":"int64"},
{"name":"history","in":"query","description":"Include tag history in output, default is
 false","required":false,"type":"boolean"},
{"in":"body","name":"patch","description":"patch","required":true,"schema":{"$ref":"#/definitions/
PiiDataColumn"}}],"responses":{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/
definitions/PiiDataColumn"}}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"}}}},"/api/ca/v1/profiler/jobs/{job}/profiles":{"get":{"tags":["results-
controller"],"summary":"Interface to get the connection profiles","description":"Returns the connection
 profiles used in this profiling job","operationId":"getAJobsProfilesUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/
ConnectionProfile"}}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":
{"description":"Not Found"}}}},"/api/ca/v1/profiler/jobs/{job}/report":{"get":{"tags":["results-
controller"],"summary":"Interface to download a PII Job report in PDF format","description":"For
 report class DRAFT, the report is generated on the fly. For TDE, AUDITOR or MANAGEMENT, the report
 is retrieved from the repository.","operationId":"getAJobAsPdfUsingGET","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"fileName","in":"query","description":"File name to be returned in content-
disposition","required":false,"type":"string"},{"name":"reportClass","in":"query","description":"Report
 class: DRAFT, TDE, AUDITOR or MANAGEMENT","required":true,"type":"string"},
{"name":"format","in":"query","description":"Format of the report
 document","required":true,"type":"string","default":"pdf","enum":["pdf"]}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/InputStreamResource"}},"400":{"description":"Bad
 Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden - User does not have permission to access the
 report.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"When this REST
 end point is down or not accessible.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Check logs for more information.","schema":{"$ref":"#/
definitions/ErrorResponse"}}}}},"/api/ca/v1/profiler/jobs/{job}/reportGen":{"post":{"tags":["results-

controller"],"summary":"Interface to create and store PII Job reports in the repository","description":"If
the report class is not specified, all three of TDE, MANAGEMENT and AUDITOR reports will be
generated.","operationId":"reportCreateUsingPOST","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"reportClass","in":"query","description":"Report class: TDE, AUDITOR or
MANAGEMENT","required":false,"type":"string"}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/InputStreamResource"}},"201":{"description":"Created"},"400":{"description":"Bad
Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden - User does not have permission to access the
report.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"When this REST end point
is down or not accessible.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
Server Error - Check logs for more information.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/
api/ca/v1/profiler/jobs/{job}/reportInfo":{"get":{"tags":["results-controller"],"summary":"Interface
to retrieve information about a PII report","description":"Information retrieved excludes the
PDF report.","operationId":"getReportInfoUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"reportClass","in":"query","description":"Report class: DRAFT, TDE, AUDITOR or
MANAGEMENT","required":false,"type":"string"}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/DBPiiReport"}},"400":{"description":"Bad Request - Specific reason is included
in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server
authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
- User does not have permission to access the report.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"When this REST end point is down or not accessible.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Check logs for
more information.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/profiler/jobs/
{job}/reviewers":{"get":{"tags":["results-controller"],"summary":"Interface to get the reviewers
of a project","operationId":"getJobReviewersUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/PiiReviewer"}}},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}},"post":
{"tags":["results-controller"],"summary":"Interface to add a reviewer to a project.","description":"The
user ID and user name are required.","operationId":"postjobReviewerUsingPOST","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},

{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"post","description":"post","required":true,"schema":{"$ref":"#/definitions/
PiiReviewer"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/
PiiReviewer"}},"201":{"description":"Created"},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/profiler/jobs/{job}/
reviewers/{id}":{"get":{"tags":["results-controller"],"summary":"Interface to get a specific reviewer
 for a project","operationId":"getAJobReviewerUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"id","in":"path","description":"id","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"OK","schema":{"$ref":"#/definitions/PiiReviewer"}},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/
ca/v1/profiler/jobs/{job}/reviewers/{reviewer}":{"delete":{"tags":["results-controller"],"summary":"Interface
 to remove a reviewer from a job","operationId":"deleteJobReviewerUsingDELETE","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"reviewer","in":"path","description":"reviewer","required":true,"type":"integer","format":"int64"}],"responses"
{"200":{"description":"OK","schema":{"type":"boolean"}},"204":{"description":"No
 Content"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"}}},"patch":
{"tags":["results-controller"],"summary":"Interface to update a reviewers input for a
 job","description":"Used this interface to accept or reject the PII classification of a
 job.","operationId":"patchjobReviewerUsingPATCH","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"reviewer","in":"path","description":"reviewer","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"patch","description":"patch","required":true,"schema":{"$ref":"#/definitions/
PiiReviewer"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/PiiReviewer"}},"204":
{"description":"No Content"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/
api/ca/v1/profiler/jobs/{job}/samples":{"get":{"tags":["results-controller"],"summary":"Interface to
 return total samples stored for ","description":"Samples collected in a job can be fetched for a specified
 list of columms.","operationId":"getJobSamplesUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"OK","schema":{"type":"object","additionalProperties":{"type":"object"}}},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not
 Found"}}},"delete":{"tags":["results-controller"],"summary":"Interface to delete stored samples
 from a job","operationId":"deleteJobSamplesUsingDELETE","consumes":["application/json"],"produces":

["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"OK","schema":{"type":"boolean"}},"204":{"description":"No Content"},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/api/ca/v1/profiler/jobs/{job}/
tables":{"get":{"tags":["results-controller"],"summary":"Interface to get the tables discovered
 in this job","operationId":"getJobTablesUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"page","in":"query","description":"The page of data to request,
 starting from 0.","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"The size of the page of data to
 request.","required":false,"type":"integer","format":"int32"},{"name":"q","in":"query","description":"Query
 parameter to search for","required":false,"type":"string"}],"responses":{"200":{"description":"OK","schema":
{"type":"array","items":{"$ref":"#/definitions/PiiTable"}}},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/profiler/jobs/{job}/
tables/{table}":{"get":{"tags":["results-controller"],"summary":"Interface to get details of a single
 table in a job","operationId":"getJobOneTableUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"OK","schema":{"$ref":"#/definitions/PiiTable"}},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/
ca/v1/profiler/jobs/{job}/tables/{table}/rows":{"get":{"tags":["results-controller"],"summary":"Interface
 to get raw data samples from a job","description":"A random set of 10 rows are returned for all columns
 from a table.","operationId":"getJobSamplesUsingGET_1","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/PiiSample"}}},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/
ca/v1/profiler/jobs/{job}/tables/{table}/samples":{"get":{"tags":["results-controller"],"summary":"Interface
 to get raw data samples from a job","description":"Samples collected in a job can be fetched for a specified
 list of columms.","operationId":"getJobSamplesUsingGET_2","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through

this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"table","in":"path","description":"table","required":true,"type":"integer","format":"int64"},
{"name":"columnIds","in":"query","description":"A comma separated list of
column IDs to fetch sample data for.","required":false,"type":"array","items":
{"type":"integer","format":"int64"},"collectionFormat":"multi"},
{"name":"page","in":"query","description":"Page number if fetch, starting at
0","required":false,"type":"integer","format":"int32"},{"name":"size","in":"query","description":"Page size
to fetch","required":false,"type":"integer","format":"int32"},
{"name":"includeTags","in":"query","description":"includeTags","required":false,"type":"boolean"}],"responses":
{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/PiiSample"}}},"401":
{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/
api/ca/v1/profiler/search/{job}":{"get":{"tags":["results-controller"],"summary":"Interface to
update a reviewers input for a job","description":"Used this interface to accept or reject the PII
classification of a job.","operationId":"searchUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"job","in":"path","description":"job","required":true,"type":"integer","format":"int64"},
{"name":"q","in":"query","description":"Query parameter to search for","required":false,"type":"string"},
{"name":"max","in":"query","description":"Maximum number of results to return for each
type, default 10","required":false,"type":"integer","format":"int32"}],"responses":{"200":
{"description":"OK","schema":{"$ref":"#/definitions/PiiReviewer"}},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/profiler/setup":{"get":{"tags":
["profiler-controller"],"summary":"Interface to get the profiler setup for the specified project and
version","description":"Returns a default (empty) setup if no profiler setup exists for the specified project
id and version id.","operationId":"getProfilerSetupUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"projectId","in":"query","description":"projectId","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"versionId","required":true,"type":"integer","format":"int64"},
{"name":"origin","in":"query","description":"origin","required":false,"type":"string"}],"responses":{"200":
{"description":"OK","schema":{"$ref":"#/definitions/PiiSetup"}},"401":{"description":"Unauthorized"},"403":
{"description":"Forbidden"},"404":{"description":"Not Found"}}},"delete":{"tags":["profiler-
controller"],"summary":"Interface to delete the profiler setup for the specified project and
version","description":"Returns 404 Not Found if the record does not exist for the specified project id
and version id.","operationId":"deleteProfilerSetupUsingDELETE","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"projectId","in":"query","description":"projectId","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"versionId","required":true,"type":"integer","format":"int64"},
{"name":"origin","in":"query","description":"origin","required":false,"type":"string"}],"responses":
{"200":{"description":"OK","schema":{"$ref":"#/definitions/PiiSetup"}},"204":{"description":"No
Content"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"}}},"patch":{"tags":
["profiler-controller"],"summary":"Interface to update the list of connection profile names for a

profiler setup","description":"The profiler setup will be created if one doesn't already exist for the specified project id and version id.","operationId":"updateSetupUsingPATCH","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"projectId","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"versionId","required":true,"type":"integer","format":"int64"},{"name":"origin","in":"query","description":"origin","required":false,"type":"string"},{"in":"body","name":"setup","description":"setup","required":true,"schema":{"$ref":"#/definitions/PiiSetup"}}],"responses":{"200":{"description":"OK","schema":{"type":"boolean"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/api/ca/v1/profiler/tags":{"get":{"tags":["tags-controller"],"summary":"Interface to get list of tags","operationId":"getPagedTagsUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"page","in":"query","description":"The page of data to request, starting from 0.","required":false,"type":"integer","default":0,"format":"int32"},{"name":"size","in":"query","description":"The size of the page of data to request.","required":false,"type":"integer","default":20,"format":"int32"},{"name":"q","in":"query","description":"Search criteria. RSQL format (see https://github.com/jirutka/rsql-parser).Allows the query to be filtered on any of the resource's field values, such as 'id' or 'name', etc.","required":false,"type":"string"}],"responses":{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/DBPiiTag"}}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/profiler/tags/{tagId}":{"get":{"tags":["tags-controller"],"summary":"Interface to get tag information for a given tag id","operationId":"getTagUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"tagId","in":"path","description":"tagId","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/DBPiiTag"}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/projects/{projectId}/versions/{versionId}/actions/calculateTableOrder":{"post":{"tags":["object-controller"],"summary":"Interface for Creating Table Order","description":"Use this interface for Creating Table Order.","operationId":"calculateTableOrderUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"ID of the project you want to calculate table order.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"path","description":"ID of the project version associated with the registered table for which you want to retrieve the columns.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"tableOrderRequest","description":"Table Order Additional options.","required":false,"schema":{"$ref":"#/definitions/TableOrderRequest"}}],"responses":{"200":{"description":"Success.","schema":{"type":"array","items":{"$ref":"#/definitions/ObjectDTO"}}},"201":{"description":"Created"},"202":{"description":"Cycles are found while calculating Table Order

select any one or more of the relations which need to be ignored.","schema":{"$ref":"#/definitions/CyclicRelations"}},"204":{"description":"No Content - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/projects/{projectId}/versions/{versionId}/dataGenerators":{"get":{"tags":["object-controller"],"summary":"Interface for getting data generators for a given user, project, and project version","description":"Use this interface to retrieve the list of data generators for a given user, project, and project version.","operationId":"getDataGeneratorsUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"ID of the project for which you want to retrieve data generators.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"path","description":"ID of the project version for which you want to retrieve data generators.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"type":"array","items":{"$ref":"#/definitions/GeneratorInfo"}}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/api/ca/v1/registeredTables":{"get":{"tags":["object-controller"],"summary":"Interface for getting all the tables registered with a project version","description":"Use this interface to retrieve the list of registered tables for a project version.","operationId":"getRegisteredTablesUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the project for which you want to retrieve registered tables.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the version for which you want to retrieve registered tables.","required":true,"type":"integer","format":"int64"},{"name":"pageNum","in":"query","description":"Page number that you want to retrieve in the paginated result. Default value is 1.","required":false,"type":"integer","format":"int32"},{"name":"pageSize","in":"query","description":"Page size of each page that you want to retrieve in the paginated result. Default value is 1000000.","required":false,"type":"integer","format":"int32"},{"name":"searchText","in":"query","description":"Search text that you want to use to filter the registered tables.","required":false,"type":"string"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/PaginatedResult"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/registeredTables/{tableId}":{"get":{"tags":["object-controller"],"summary":"Interface for getting the registered table details","description":"Use this interface to retrieve the registered table details.","operationId":"getRegisteredTableDetailsUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer

HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"tableId","in":"path","description":"ID of the table for which you want to retrieve the details.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"ID of the project to which the registered table belongs.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the version to which the registered table belongs.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/TableDetails"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/tables":{"get":{"tags":["object-controller"],"summary":"Interface for getting the tables under a schema","description":"Use this interface to get the tables under the schema","operationId":"getTablesUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the project under which you want to get the tables. Need this for computing the differences","required":false,"type":"integer","format":"int64"},{"name":"versionId","in":"query","description":"ID of the project version under which you want to get tables. Need this for computing the differences","required":false,"type":"integer","format":"int64"},{"name":"schema","in":"query","description":"Name of the table location from which you want to get tables.","required":true,"type":"string"},{"name":"profileName","in":"query","description":"Name of the connection profile that identifies the database from where you register the tables.","required":false,"type":"string"},{"name":"searchText","in":"query","description":"Search Text.","required":false,"type":"string"},{"name":"page","in":"query","description":"Page number which you want to retrieve. default is 1","required":false,"type":"integer","format":"int32"},{"name":"size","in":"query","description":"Page size of each page. default is 25","required":false,"type":"integer","format":"int32"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/TablesInfo"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden"},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}}},"definitions":{"AttribMaskFunctionGroup":{"type":"object","properties":{"attributeId":{"type":"integer","format":"int64"},"attributeName":{"type":"string"},"classifierBased":{"type":"boolean"},"dataSource":{"type":"string"},"dataType":{"type":"string"},"databaseName":{"type":"string"},"entityName":{"type":"string"},"hasWhereClause":{"type":"boolean"},"maskGroupId":{"type":"integer","format":"int64"},"maskGroupLabel":{"type":"string"},"maskGroupShared":{"type":"boolean"},"numOtherTags":{"type":"integer","format":"int64"},"primaryTag":{"type":"string"},"schemaName":{"type":"string"}}},"Classifier":{"type":"object","properties":{"classifierClass":{"type":"string"},"classifierOrigin":{"type":"string"},"classifierType":{"type":"string"},"config":{"type":"array","items":{"$ref":"#/definitions/DBClassifierConfig"}},"created":{"type":"string","format":"date-time"},"createdBy":{"type":"string"},"description":{"type":"string"},"descriptions":{"type":"array","items":{"$ref":"#/definitions/DBClassifierString"}},"id":

{"type":"integer","format":"int64"},"maskFunction":{"type":"array","items":{"$ref":"#/definitions/DBMaskFunction"}},"maskFunctionGroup":{"type":"array","items":{"$ref":"#/definitions/DBMaskFunctionGroup"}},"name":{"type":"string"},"names":{"type":"array","items":{"$ref":"#/definitions/DBClassifierString"}},"parentId":{"type":"integer","format":"int64"},"tagId":{"type":"integer","format":"int64"},"tags":{"type":"string"},"updated":{"type":"string","format":"date-time"},"updatedBy":{"type":"string"}}},"ClassifierContainer":{"type":"object","properties":{"containedClassifiers":{"type":"array","items":{"$ref":"#/definitions/DBClassifier"}},"containedContainers":{"type":"array","items":{"$ref":"#/definitions/DBClassifierContainer"}},"created":{"type":"string","format":"date-time"},"createdBy":{"type":"string"},"description":{"type":"string"},"descriptions":{"type":"array","items":{"$ref":"#/definitions/DBClassifierString"}},"id":{"type":"integer","format":"int64"},"name":{"type":"string"},"names":{"type":"array","items":{"$ref":"#/definitions/DBClassifierString"}},"parentId":{"type":"integer","format":"int64"},"root":{"type":"boolean"},"updated":{"type":"string","format":"date-time"},"updatedBy":{"type":"string"}}},"ClassifierStatusResponse":{"type":"object","properties":{"importing":{"type":"boolean"}}},"ColumnDetails":{"type":"object","properties":{"dataType":{"type":"string","description":"Datatype of the column"},"defaultValue":{"type":"string"},"id":{"type":"integer","format":"int64","description":"ID of the column","readOnly":true},"isNullable":{"type":"string","description":"Nullable column"},"name":{"type":"string","description":"Name of the column"},"precision":{"type":"integer","format":"int64","description":"Precision for a column"},"scale":{"type":"integer","format":"int64","description":"Scale for a column"},"sequence":{"type":"integer","format":"int64","description":"Sequential number of the column"}}},"ConnectionProfile":{"type":"object","required":["dbType","name","password","server","username"],"properties":{"additionalConnectionProperties":{"type":"string","description":"JDBC connection string properties. Applicable only for database type db2/400 sql"},"created":{"type":"string","format":"date-time","description":"Creation date"},"createdBy":{"type":"integer","format":"int64","description":"Created by"},"database":{"type":"string","description":"Database name"},"datasourceDriver":{"type":"string","description":"DataSource Driver"},"datasourceUrl":{"type":"string","description":"DataSource URL"},"dbType":{"type":"string","description":"Type of database","enum":["sql server","oracle","mysql","sybase","teradata","db2","db2/400 sql"]},"description":{"type":"string","description":"Descriptive text"},"instance":{"type":"string","description":"Sql server instance name"},"integratedSecurity":{"type":"boolean","example":false,"description":"Use Integrated Security for authentication. Applicable only for database type SQL Server"},"modified":{"type":"string","format":"date-time","description":"Last modified date"},"name":{"type":"string","description":"Name of the connection profile"},"password":{"type":"string","description":"Password"},"port":{"type":"string","description":"Database server port"},"schema":{"type":"string","description":"Sql server schema name"},"server":{"type":"string","description":"Database server hostname"},"service":{"type":"string","description":"Oracle service name"},"username":{"type":"string","description":"Username"}}},"CyclicRelations":{"type":"object","properties":{"foreignKeys":{"type":"array","items":{"$ref":"#/definitions/ForeignKeyDetails"}},"message":{"type":"string"},"relationships":{"type":"array","items":{"$ref":"#/definitions/RelationshipDetails"}}}},"DBClassifier":{"type":"object","properties":{"classifierClass":{"type":"string"},"classifierOrigin":{"type":"string"},"classifierType":{"type":"string"},"config":{"type":"array","items":{"$ref":"#/definitions/DBClassifierConfig"}},"created":{"type":"string","format":"date-time"},"createdBy":{"type":"string"},"description":{"type":"string"},"descriptions":{"type":"array","items":{"$ref":"#/definitions/DBClassifierString"}},"id":{"type":"integer","format":"int64"},"maskFunction":{"type":"array","items":{"$ref":"#/definitions/DBMaskFunction"}},"maskFunctionGroup":{"type":"array","items":{"$ref":"#/definitions/DBMaskFunctionGroup"}},"name":{"type":"string"},"names":{"type":"array","items":{"$ref":"#/definitions/DBClassifierString"}},"tagId":{"type":"integer","format":"int64"},"tags":{"type":"string"},"updated":{"type":"string","format":"date-time"},"updatedBy":{"type":"string"}}},"DBClassifierConfig":{"type":"object","properties":{"id":{"type":"integer","format":"int64"},"name":{"type":"string"},"value":{"type":"string"}}},"DBClassifierContainer":{"type":"object","properties":{"containedClassifiers":{"type":"array","items":{"$ref":"#/definitions/DBClassifier"}},"containedContainers":{"type":"array","items":

{"$ref":"#/definitions/DBClassifierContainer"}},"created":{"type":"string","format":"date-time"},"createdBy":
{"type":"string"},"description":{"type":"string"},"descriptions":{"type":"array","items":{"$ref":"#/
definitions/DBClassifierString"}},"id":{"type":"integer","format":"int64"},"name":{"type":"string"},"names":
{"type":"array","items":{"$ref":"#/definitions/DBClassifierString"}},"root":{"type":"boolean"},"updated":
{"type":"string","format":"date-time"},"updatedBy":{"type":"string"}}},"DBClassifierString":
{"type":"object","properties":{"id":{"type":"integer","format":"int64"},"lang":
{"type":"string"},"value":{"type":"string"}}},"DBDataDiscoveryEntityAttrDef":
{"type":"object","properties":{"attributeAlias":{"type":"string"},"attributeId":
{"type":"integer","format":"int64"},"attributeName":{"type":"string"},"attributeSeq":
{"type":"integer","format":"int64"},"computed":{"type":"string"},"createFields":{"type":"string"},"datatype":
{"type":"string"},"datatypeId":{"type":"integer","format":"int64"},"datatypeOwner":
{"type":"string"},"dateCreated":{"type":"string","format":"date-time"},"dateUpdated":
{"type":"string","format":"date-time"},"dbDefault":{"type":"string"},"defaultValue":
{"type":"string"},"direction":{"type":"string"},"entExclude":{"type":"string"},"entIndex":
{"type":"boolean"},"entityDefinition":{"$ref":"#/definitions/DBDataDiscoveryEntityDef"},"entityFormat":
{"type":"string"},"entityId":{"type":"integer","format":"int64"},"entityPrecision":
{"type":"integer","format":"int64"},"expValFor":{"type":"string"},"fileFrom":
{"type":"integer","format":"int64"},"fileTo":{"type":"integer","format":"int64"},"fileValueEnd":
{"type":"string"},"fileValueStart":{"type":"string"},"format":{"type":"string"},"isComputed":
{"type":"string"},"jobId":{"type":"integer","format":"int64"},"nullable":
{"type":"string"},"precision":{"type":"integer","format":"int64"},"primaryTag":
{"type":"string"},"programCreated":{"type":"string"},"programUpdated":{"type":"string"},"projId":
{"type":"integer","format":"int64"},"pvId":{"type":"integer","format":"int64"},"rbt":
{"type":"string"},"remark2":{"type":"string"},"remark3":{"type":"string"},"remarks":
{"type":"string"},"scale":{"type":"integer","format":"int64"},"updateFields":{"type":"string"},"validation":
{"type":"string"},"whoCreated":{"type":"string"},"whoUpdated":{"type":"string"},"xpath":
{"type":"string"}}},"DBDataDiscoveryEntityDef":{"type":"object","properties":{"aliasOfId":
{"type":"integer","format":"int64"},"attributeCount":{"type":"integer","format":"int64"},"attributeList":
{"type":"array","items":{"$ref":"#/definitions/DBDataDiscoveryEntityAttrDef"}},"clobId1":
{"type":"integer","format":"int64"},"clobId2":{"type":"integer","format":"int64"},"comments":
{"type":"string"},"confirmed":{"type":"boolean"},"dataSourceName":{"type":"string"},"databaseName":
{"type":"string"},"dateReviewed":{"type":"string","format":"date-time"},"description":
{"type":"string"},"entityAlias":{"type":"string"},"entityId":{"type":"integer","format":"int64"},"entityName":
{"type":"string"},"entityType":{"type":"string"},"excluded":{"type":"boolean"},"fdId":
{"type":"integer","format":"int64"},"fdLocation":{"type":"string"},"fkList":
{"type":"array","items":{"$ref":"#/definitions/TableForeignKeyDefinition"}},"foreignKeyCount":
{"type":"integer","format":"int64"},"indexCount":{"type":"integer","format":"int64"},"isPII":
{"type":"boolean"},"jobId":{"type":"integer","format":"int64"},"ownerName":
{"type":"string"},"primaryKeyDescriptor":{"type":"string"},"primaryKeyIndex":{"type":"string"},"profileJobId":
{"type":"integer","format":"int64"},"profileName":{"type":"string"},"profileSuId":
{"type":"integer","format":"int64"},"projectId":{"type":"integer","format":"int64"},"projectVersionId":
{"type":"integer","format":"int64"},"reason":{"type":"string"},"refName":{"type":"string"},"refOwner":
{"type":"string"},"refType":{"type":"string"},"registeredDBMS":{"type":"string"},"registeredName":
{"type":"string"},"restPassword":{"type":"string"},"restUrl":{"type":"string"},"restUsername":
{"type":"string"},"reviewer":{"type":"string"},"rowCount":{"type":"integer","format":"int64"},"schemaName":
{"type":"string"},"serverName":{"type":"string"},"tablespace":{"type":"string"},"uniqueKeys":
{"type":"array","items":{"$ref":"#/definitions/DBDataDiscoveryUniqueKey"}},"whereClause":
{"type":"string"}}},"DBDataDiscoveryUniqueKey":{"type":"object","properties":{"columnName":
{"type":"string"},"columnPos":{"type":"integer","format":"int64"},"createFields":
{"type":"string"},"dateCreated":{"type":"string","format":"date-time"},"dateUpdated":
{"type":"string","format":"date-time"},"entityId":{"type":"integer","format":"int64"},"indexName":
{"type":"string"},"indexOwner":{"type":"string"},"keyName":{"type":"string"},"keyType":
{"type":"string"},"programCreated":{"type":"string"},"programUpdated":{"type":"string"},"projectId":

{"type":"integer","format":"int64"},"projectVersionId":{"type":"integer","format":"int64"},"updateFields":
{"type":"string"},"whoCreated":{"type":"string"},"whoUpdated":{"type":"string"}}},"DBMaskFunction":
{"type":"object","properties":{"classifierId":{"type":"integer","format":"int64"},"crossReference":
{"type":"string"},"dateFormat":{"type":"string"},"displayName":{"type":"string"},"fromOccurrence":
{"type":"string"},"funcId":{"type":"integer","format":"int64"},"functionName":{"type":"string"},"keepNulls":
{"type":"boolean"},"maskParams":{"type":"array","items":{"$ref":"#/definitions/DBMaskParams"}},"notes":
{"type":"string"},"origFrom":{"type":"integer","format":"int64"},"overrideSql":{"type":"string"},"preformat":
{"type":"string"},"projectId":{"type":"integer","format":"int64"},"projectVersionId":
{"type":"integer","format":"int64"},"restartColumn":{"type":"string"},"substrLength":
{"type":"integer","format":"int32"},"substrStart":{"type":"integer","format":"int32"},"tagId":
{"type":"integer","format":"int64"},"tagName":{"type":"string"},"tagPrimaryMf":
{"$ref":"#/definitions/DBTagPrimaryMf"},"toOccurrence":{"type":"string"},"uniqueColumns":
{"type":"string"},"updateDb":{"type":"boolean"},"useMaskedValues":{"type":"boolean"},"xpathElement":
{"type":"string"}}},"DBMaskFunctionGroup":{"type":"object","properties":{"classifierBased":
{"type":"boolean"},"classifierId":{"type":"integer","format":"int64"},"displayName":
{"type":"string"},"groupId":{"type":"integer","format":"int64"},"groupName":
{"type":"string"},"groupNotes":{"type":"string"},"isGlobal":{"type":"boolean"},"maskFunction":
{"type":"array","items":{"$ref":"#/definitions/DBMaskFunction"}},"projectId":
{"type":"integer","format":"int64"},"projectVersionId":{"type":"integer","format":"int64"},"tagId":
{"type":"integer","format":"int64"},"tagName":{"type":"string"}}},"DBMaskParams":
{"type":"object","properties":{"paramId":{"type":"integer","format":"int64"},"paramPosition":
{"type":"integer","format":"int32"},"paramValue":{"type":"string"}}},"DBModellingParameters":
{"type":"object","properties":{"RefreshToken":{"type":"string"},"classifierPacks":
{"type":"array","items":{"type":"integer","format":"int64"}},"connProfiles":{"type":"array","items":
{"type":"string"}},"connectionProfiles":{"type":"array","items":{"type":"string"}},"dataSourceNames":
{"type":"array","items":{"type":"string"}},"environment":{"type":"string"},"environmentId":
{"type":"integer","format":"int64"},"environmentName":{"type":"string"},"filters":
{"type":"array","items":{"$ref":"#/definitions/PIIScanFilter"}},"isIncludeFilter":
{"type":"boolean"},"jobId":{"type":"integer","format":"int64"},"jobName":
{"type":"string"},"jobType":{"type":"string"},"projId":{"type":"integer","format":"int64"},"pverId":
{"type":"integer","format":"int64"},"scanLevel":{"type":"integer","format":"int32"},"scanNumericKeys":
{"type":"boolean"},"scanStringKeys":{"type":"boolean"},"scanType":{"type":"string","enum":
["PRE_SCAN_ONLY","PRE_SCAN_AND_DISCOVER_RELATIONS"]},"storeSamples":{"type":"boolean"},"userName":
{"type":"string"}}},"DBPiiApprover":{"type":"object","properties":{"approved":{"type":"boolean"},"comments":
{"type":"string"},"dateApproved":{"type":"string","format":"date-time"},"jobId":
{"type":"integer","format":"int64"},"reviewerId":{"type":"integer","format":"int64"}}},"DBPiiReport":
{"type":"object","properties":{"createdBy":{"type":"string"},"hash":{"type":"string"},"jobId":
{"type":"integer","format":"int64"},"pdf":{"type":"string","format":"byte"},"reportClass":
{"type":"string","enum":["TDE","AUDITOR","MANAGEMENT"]},"reportId":
{"type":"integer","format":"int64"},"timeCreated":{"$ref":"#/definitions/
Timestamp"}}},"DBPiiReviewer":{"type":"object","properties":{"jobId":
{"type":"integer","format":"int64"},"reviewerId":{"type":"integer","format":"int64"},"reviewerName":
{"type":"string"},"role":{"type":"string","enum":["REVIEWER","APPROVER"]}}},"DBPiiTag":
{"type":"object","properties":{"id":{"type":"integer","format":"int64"},"name":
{"type":"string"},"programCreated":{"type":"string"},"programUpdated":{"type":"string"},"projectId":
{"type":"integer","format":"int64"},"versionId":{"type":"integer","format":"int64"},"whoCreated":
{"type":"string"},"whoUpdated":{"type":"string"}}},"DBTagPrimaryMf":{"type":"object","properties":
{"funcId":{"type":"integer","format":"int64"},"notes":{"type":"string"},"pmfId":
{"type":"integer","format":"int64"},"projectId":{"type":"integer","format":"int64"},"projectVersionId":
{"type":"integer","format":"int64"},"tagId":{"type":"integer","format":"int64"}}},"DataDiscoveryJobDTO":
{"type":"object","properties":{"environmentId":{"type":"integer","format":"int64"},"environmentName":
{"type":"string"},"jobId":{"type":"integer","format":"int64"},"jobName":
{"type":"string"},"jobRunning":{"type":"boolean"},"jobState":{"type":"string"},"jobStatus":

```
{"type":"string"},"profileJobId":{"type":"integer","format":"int64"},"profileJobState":
{"type":"string"},"profileJobStatus":{"type":"string"},"profileStartDate":
{"type":"string","format":"date-time"},"projectId":{"type":"integer","format":"int64"},"projectVersionId":
{"type":"integer","format":"int64"},"scannedDataSources":{"type":"array","items":{"$ref":"#/
definitions/HashMap«string,string»"}},"startDate":{"type":"string","format":"date-
time"},"startedBy":{"type":"string"},"stopDate":{"type":"string","format":"date-
time"}}},"DataModelEntityAttributeInfo":{"type":"object","properties":{"alias":
{"type":"string"},"attributeId":{"type":"integer","format":"int64"},"attributeName":
{"type":"string"},"dataType":{"type":"string"},"datatypeId":{"type":"integer","format":"int64"},"precision":
{"type":"integer","format":"int64"},"primaryTag":{"type":"string"},"scale":
{"type":"integer","format":"int64"}}},"DataModelEntityInfo":{"type":"object","properties":{"alias":
{"type":"string"},"attributes":{"$ref":"#/definitions/
Iterable«DataModelEntityAttributeInfo»"},"dataSourceName":{"type":"string"},"dataSourceType":
{"type":"string"},"databaseName":{"type":"string"},"entityId":
{"type":"integer","format":"int64"},"entityName":{"type":"string"},"entityOwner":
{"type":"string"},"fromEntityDef":{"$ref":"#/definitions/DBDataDiscoveryEntityDef"},"fullyQualifiedPath":
{"type":"string"},"hierarchy":{"$ref":"#/definitions/HierarchyDTO"},"relatedEntities":{"$ref":"#/definitions/
Iterable«DataModelEntityInfo»"},"relationshipDetails":{"type":"array","items":{"$ref":"#/definitions/
EntityRelationshipDetails"}},"schemaName":{"type":"string"},"uniqueKeys":{"type":"array","items":{"$ref":"#/
definitions/DataModelUniqueKeyInfo"}}}},"DataModelExclusionsDetails":{"type":"object","properties":{"active":
{"type":"boolean"},"name":{"type":"string"}}},"DataModelExclusionsInfo":{"type":"object","properties":
{"attributeDetails":{"type":"array","items":{"$ref":"#/definitions/DataModelExclusionsDetails"}},"jobId":
{"type":"integer","format":"int64"},"projectId":{"type":"integer","format":"int64"},"versionId":
{"type":"integer","format":"int64"}}},"DataModelUniqueKeyInfo":{"type":"object","properties":
{"columnNames":{"type":"array","items":{"type":"string"}},"indexName":{"type":"string"},"indexOwner":
{"type":"string"},"keyName":{"type":"string"},"keyType":{"type":"string"}}},"EntityAttributeRelDetails":
{"type":"object","properties":{"childAttributeId":{"type":"integer","format":"int64","description":"Child
 Attribute ID"},"childAttributeName":{"type":"string","description":"Child Attribute
 name"},"parentAttributeId":{"type":"integer","format":"int64","description":"Parent Attribute
 ID"},"parentAttributeName":{"type":"string","description":"Parent Attribute name"},"sequence":
{"type":"integer","format":"int64","description":"Attribute sequence"}}},"EntityExclusion":
{"type":"object","properties":{"id":{"type":"integer","format":"int64"},"type":
{"type":"string"},"value":{"type":"string"}}},"EntityExclusionInfo":{"type":"object","properties":
{"exclusions":{"type":"array","items":{"$ref":"#/definitions/EntityExclusion"}},"projectId":
{"type":"integer","format":"int64"},"versionId":{"type":"integer","format":"int64"}}},"EntityMaskInfo":
{"type":"object","properties":{"dataSource":{"type":"string"},"databaseName":{"type":"string"},"entityId":
{"type":"integer","format":"int64"},"entityName":{"type":"string"},"notes":{"type":"string"},"schemaName":
{"type":"string"},"taggedAttributes":{"type":"integer","format":"int64"}}},"EntityRelationshipDetails":
{"type":"object","properties":{"childEntityId":{"type":"integer","format":"int64","description":"Child
 entity ID"},"childEntityName":{"type":"string","description":"Child entity name"},"id":
{"type":"integer","format":"int64","description":"ID of the relationship","readOnly":true},"parentEntityId":
{"type":"integer","format":"int64","description":"Parent entity ID"},"parentEntityName":
{"type":"string","description":"Parent entity name"},"relationshipAttributes":
{"type":"array","description":"Relationship Attributes","items":{"$ref":"#/definitions/
EntityAttributeRelDetails"}},"relationshipMatcher":{"type":"string","description":"Relationship
 Matcher"},"relationshipType":{"type":"integer","format":"int64","description":"Relationship
 type"}}},"ErrorResponse":{"type":"object","properties":{"errorCode":{"type":"string"},"errorDetail":
{"type":"string"},"errorMsg":{"type":"string"},"status":{"type":"integer","format":"int32"},"timestamp":
{"type":"string"}}},"File":{"type":"object","properties":{"absolute":{"type":"boolean"},"absoluteFile":
{"$ref":"#/definitions/File"},"absolutePath":{"type":"string"},"canonicalFile":{"$ref":"#/definitions/
File"},"canonicalPath":{"type":"string"},"directory":{"type":"boolean"},"file":{"type":"boolean"},"freeSpace":
{"type":"integer","format":"int64"},"hidden":{"type":"boolean"},"name":{"type":"string"},"parent":
{"type":"string"},"parentFile":{"$ref":"#/definitions/File"},"path":{"type":"string"},"totalSpace":
```

{"type":"integer","format":"int64"},"usableSpace":{"type":"integer","format":"int64"}}},"ForeignKeyDetails":
{"type":"object","properties":{"columnPosition":{"type":"integer","format":"int64","description":"Column
Position of the foreign Key"},"foreignKeyName":{"type":"string","description":"Name of
the Foreign Key"},"referenceTableColumnName":{"type":"string","description":"Foreign key
reference Column name"},"referenceTableName":{"type":"string","description":"Foreign key
reference table name"},"sequence":{"type":"integer","format":"int64","description":"Sequential
number of the foreign Key"},"tableColumnName":{"type":"string","description":"Foreign
key table Column name"},"tableId":{"type":"integer","format":"int64","description":"ID of
the table","readOnly":true},"tableName":{"type":"string","description":"Foreign key table
name"}}},"GeneratorInfo":{"type":"object","properties":{"comment":{"type":"string","description":"Comment
assigned by the user to the generator"},"created":{"type":"string","description":"Timestamp
of the generator creation"},"description":{"type":"string","description":"Description of
the generator"},"generatorId":{"type":"number","description":"Id of the generator"},"name":
{"type":"string","description":"Name of the generator"},"onDemand":{"type":"string","description":"Indicates
if the generator is available on demand as service"},"parentId":{"type":"number","description":"Id
of the parent"},"projectId":{"type":"integer","format":"int64","description":"Id of the project
to which the generator belongs to"},"projectName":{"type":"string","description":"Name of the
project to which the generator belongs to"},"type":{"type":"string","description":"Type of the
generator"},"updated":{"type":"string","description":"Timestamp of the last updation of the
generator"},"versionId":{"type":"integer","format":"int64","description":"Id of the version under
which the generator is created"},"versionName":{"type":"string","description":"Name of the version to
which the generator belongs to"}}},"HashMap«string,string»":{"type":"object","additionalProperties":
{"type":"string"}},"HierarchyDTO":{"type":"object","properties":{"childrenNames":{"type":"string"},"dbType":
{"type":"string"},"entityId":{"type":"integer","format":"int64"},"fullyQualifiedPath":
{"type":"string"},"jobId":{"type":"integer","format":"int64"},"node":{"type":"array","items":
{"$ref":"#/definitions/HierarchyDTO"}},"nodeId":{"type":"integer","format":"int64"},"nodeName":
{"type":"string"},"nodePath":{"type":"array","items":{"type":"string"}},"nodeType":
{"type":"string"},"numChildren":{"type":"integer","format":"int64"},"parentNodeId":
{"type":"integer","format":"int64"},"projectId":{"type":"integer","format":"int64"},"projectVersionId":
{"type":"integer","format":"int64"}}},"ImportClassifierResponse":{"type":"object","properties":
{"classifiersCreated":{"type":"integer","format":"int32"},"classifiersUpdated":
{"type":"integer","format":"int32"},"containersCreated":
{"type":"integer","format":"int32"},"containersUpdated":
{"type":"integer","format":"int32"},"duplicateClassifiers":
{"type":"integer","format":"int32"},"duplicateContainers":
{"type":"integer","format":"int32"},"duplicateOption":{"type":"string"},"duplicateSeedlist":
{"type":"integer","format":"int32"},"seedListCreated":{"type":"integer","format":"int32"},"seedListUpdated":
{"type":"integer","format":"int32"}}},"InputStream":{"type":"object"},"InputStreamResource":
{"type":"object","properties":{"description":{"type":"string"},"file":{"$ref":"#/
definitions/File"},"filename":{"type":"string"},"inputStream":{"$ref":"#/definitions/
InputStream"},"open":{"type":"boolean"},"readable":{"type":"boolean"},"uri":{"$ref":"#/
definitions/URI"},"url":{"$ref":"#/definitions/URL"}}},"Iterable«DataModelEntityAttributeInfo»":
{"type":"object"},"Iterable«DataModelEntityInfo»":{"type":"object"},"Iterable«PiiColumn»":
{"type":"object"},"Job":{"type":"object","properties":{"artifactLocation":
{"type":"string","description":"Location of the artifact related to the Job"},"created":
{"type":"string","format":"date-time","description":"Time at which Job was created in the
system"},"createdBy":{"type":"string","description":"Name of the user who submitted the
Job"},"description":{"type":"string","description":"Description of the Job"},"duration":
{"type":"integer","format":"int64","description":"Amount of time taken for completion of the
Job"},"email":{"type":"string","description":"Email address to which job execution report is
sent"},"endTime":{"type":"string","format":"date-time","description":"Time at which Job execution
has completed"},"jobId":{"type":"integer","format":"int64","description":"Id of the Job"},"jobs":
{"type":"array","description":"List of child jobs","items":{"$ref":"#/definitions/Job"}},"name":

{"type":"string","description":"Name of the Job"},"origin":{"type":"string","description":"Name of
the module that has submitted the Job"},"parameters":{"type":"object","description":"Job related
parameters"},"parentId":{"type":"integer","format":"int64","description":"Id of the parent Job if the Job
is a child job. Zero otherwise"},"projectId":{"type":"integer","format":"int64","description":"Id of the
project for which Job is submitted"},"projectName":{"type":"string","description":"Name of the project
for which Job is submitted"},"runningStatus":{"type":"string","description":"Indicates the running status
of the Job"},"scheduledTime":{"type":"string","format":"date-time","description":"Time for which Job
execution is scheduled"},"startTime":{"type":"string","format":"date-time","description":"Time at which Job
execution has begun"},"status":{"type":"string","description":"Status of Job execution"},"statusMessage":
{"type":"string","description":"Job's status message"},"type":{"type":"string","description":"Type
of the Job"},"versionId":{"type":"integer","format":"int64","description":"Version Id of the project
for which Job is submitted"}}},"JobProgress":{"type":"object","properties":{"columnsClassified":
{"type":"integer","format":"int64"},"columnsScanned":{"type":"integer","format":"int64"},"listApprovers":
{"type":"array","items":{"$ref":"#/definitions/DBPiiApprover"}},"listReviewers":
{"type":"array","items":{"$ref":"#/definitions/DBPiiReviewer"}},"tablesClassified":
{"type":"integer","format":"int64"},"tablesReviewed":{"type":"integer","format":"int64"},"tablesScanned":
{"type":"integer","format":"int64"},"totalApprovers":{"type":"integer","format":"int64"},"totalColumns":
{"type":"integer","format":"int64"},"totalReviewers":{"type":"integer","format":"int64"},"totalTables":
{"type":"integer","format":"int64"}}},"MaskConfigGroupByTagWithPath":
{"type":"object","properties":{"attributeId":{"type":"integer","format":"int64"},"attributeName":
{"type":"string"},"dataSource":{"type":"string"},"databaseName":{"type":"string"},"entityId":
{"type":"integer","format":"int64"},"entityName":{"type":"string"},"group":{"$ref":"#/
definitions/MaskFunctionGroup"},"maskCrossRef":{"type":"integer","format":"int64"},"schemaName":
{"type":"string"},"tagName":{"type":"string"}}},"MaskConfigGroupsByTag":
{"type":"object","properties":{"currentMaskingGroupList":{"type":"array","items":
{"$ref":"#/definitions/MaskConfigGroupByTagWithPath"}},"defaultTagMaskingGroup":
{"type":"integer","format":"int64"},"effectiveMaskGroup":{"type":"string"},"effectiveMaskGroupId":
{"type":"integer","format":"int64"},"hasWhereClause":{"type":"boolean"},"knownMaskingGroupsList":
{"type":"array","items":{"$ref":"#/definitions/MaskFunctionGroup"}},"maskingGroupCount":
{"type":"integer","format":"int64"},"tagId":{"type":"integer","format":"int64"},"tagName":
{"type":"string"},"unmaskedAttributes":{"type":"array","items":{"$ref":"#/definitions/
MaskConfigGroupByTagWithPath"}}}},"MaskFunction":{"type":"object","properties":
{"crossReference":{"type":"string"},"dateFormat":{"type":"string"},"displayName":
{"type":"string"},"fromOccurrence":{"type":"string"},"keepNulls":{"type":"boolean"},"maskFunctionId":
{"type":"integer","format":"int64"},"maskFunctionLabel":{"type":"string"},"maskFunctionName":
{"type":"string"},"maskFunctionParams":{"type":"array","items":{"$ref":"#/
definitions/MaskFunctionParams"}},"notes":{"type":"string"},"overrideSql":
{"type":"string"},"preformat":{"type":"string"},"restartColumn":{"type":"string"},"substrLength":
{"type":"integer","format":"int32"},"substrStart":{"type":"integer","format":"int32"},"toOccurrence":
{"type":"string"},"uniqueColumns":{"type":"string"},"updateDb":{"type":"boolean"},"useMaskedValues":
{"type":"boolean"},"whereClause":{"type":"string"},"xpathElement":{"type":"string"}}},"MaskFunctionGroup":
{"type":"object","properties":{"attributeCount":{"type":"integer","format":"int32"},"classifierBased":
{"type":"boolean"},"configuration":{"type":"array","items":{"$ref":"#/definitions/
MaskFunction"}},"maskGroupId":{"type":"integer","format":"int64"},"maskGroupLabel":
{"type":"string"},"maskGroupShared":{"type":"boolean"},"notes":{"type":"string"},"tagName":
{"type":"string"}}},"MaskFunctionGroupId":{"type":"object","properties":{"groupId":
{"type":"integer","format":"int64"}}},"MaskFunctionParams":{"type":"object","properties":{"pos":
{"type":"integer","format":"int32"},"value":{"type":"string"}}},"MaskSetting":{"type":"object","properties":
{"allowedValues":{"type":"array","items":{"type":"string"}},"description":{"type":"string"},"hasUserSetting":
{"type":"boolean"},"hiddenFromUser":{"type":"boolean"},"id":{"type":"integer","format":"int64"},"isBoolean":
{"type":"string"},"isChar":{"type":"string"},"name":{"type":"string"},"type":{"type":"string"},"value":
{"type":"string"}}},"ModelTableInfo":{"type":"object","properties":{"databaseName":
{"type":"string"},"differences":{"type":"string"},"diffrenceReport":{"$ref":"#/definitions/

```
TableReconcileReport"},"entityId":{"type":"integer","format":"int64"},"isUnique":
{"type":"boolean"},"profileName":{"type":"string"},"schemaName":{"type":"string"},"status":
{"type":"string"},"tableName":{"type":"string"}}},"ObjectDTO":{"type":"object","properties":
{"columns":{"type":"array","items":{"$ref":"#/definitions/ColumnDetails"}},"explicitNamespaces":
{"type":"string"},"fileConnProfId":{"type":"integer","format":"int64"},"fileConnectionProfileName":
{"type":"string"},"fileEncoding":{"type":"string"},"fileLocation":{"type":"string"},"fileName":
{"type":"string"},"fileStatus":{"type":"integer","format":"int64"},"filecount":
{"type":"integer","format":"int64"},"foreignKeys":{"type":"array","items":{"$ref":"#/definitions/
ForeignKeyDetails"}},"group":{"type":"string"},"jobFailureMessage":{"type":"string"},"jobId":
{"type":"integer","format":"int64"},"noNamespaceSchemaLocation":{"type":"string"},"objectId":
{"type":"integer","format":"int64"},"objectName":{"type":"string"},"objectType":{"type":"string"},"parentId":
{"type":"integer","format":"int64"},"programUpdated":{"type":"string"},"projectId":
{"type":"integer","format":"int64"},"relationships":{"type":"array","items":{"$ref":"#/definitions/
RelationshipDetails"}},"rootFilePath":{"type":"string"},"schemaLocation":{"type":"string"},"tableColumnCount":
{"type":"string"},"tableForeignKeyCount":{"type":"string"},"tableIndexCount":{"type":"string"},"tableOrder":
{"type":"integer","format":"int64"},"tableOwner":{"type":"string"},"tablePrimaryKeyIndex":
{"type":"string"},"tableRegisteredDBMS":{"type":"string"},"versionId":
{"type":"integer","format":"int64"}}},"ObjectEntriesEffected":{"type":"object","properties":{"code":
{"type":"string","enum":
["100","101","102","103","200","201","202","203","204","205","206","207","208","226","300","301","302","303","304","305
{"type":"integer","format":"int64"},"objectId":{"type":"integer","format":"int64"},"objectsEffected":
{"type":"integer","format":"int32"},"successMsg":{"type":"string"}}},"ObjectList":
{"type":"object","properties":{"objectIds":{"type":"array","items":
{"type":"integer","format":"int64"}}}},"ObjectRequest":{"type":"object","required":
["connectionProfileName","objectName","objectType","schema","tableNames"],"properties":
{"connectionProfileName":{"type":"string","description":"Name of the connection profile
 that identifies the database from where you register the tables."},"dataStartsAt":
{"type":"integer","format":"int32","description":"Row number from where the data starts
 from."},"explicitNamespaces":{"type":"string","description":"semicolon delimited namespaces that need to
 be added explicitly in each exported XML document"},"fileEncoding":{"type":"string","description":"Encoding
 format of the file that you want to associate to the object that you are creating. Standard
 character sets include US-ASCII, ISO-8859-1, UTF-8, UTF-16BE, UTF-16LE, UTF-16. Default value is
 UTF-8."},"generatorId":{"type":"integer","format":"int64","description":"Id of the generator in which
 file data should get imported"},"headerAt":{"type":"integer","format":"int32","description":"Row
 number of the header."},"importData":{"type":"object","description":"Flag to import the file
 data to generator","additionalProperties":{"type":"boolean"}},"noNamespaceSchemaLocation":
{"type":"string","description":"Location of the XML Schema document that does not have a target
 namespace."},"objectName":{"type":"string","description":"Name of the object that you are
 creating. It should be non empty and should not contain the following characters: <,>,:,\",/,\
\,|,?,*."},"objectRemoteLocation":{"type":"string","description":"URI Location of the file to associate
 with the object you are creating.Either of files or objectRemoteLocation are mandatory for object
 types:XSD,WSDL."},"objectType":{"type":"string","description":"Type of the object you ant to create.
 Valid types are: XML,XSD,WSDL,RRPAIR,JSON."},"rootName":{"type":"string","description":"Name of the
 root element, in case of multiple files."},"schema":{"type":"string","description":"Name of the table
 location from which you want to register tables."},"schemaLocation":{"type":"string","description":"List
 of the locations of a schema that contains qualified (a schema with a namespace) schema constructs.
 The first URI reference in each pair is a namespace name, and the second is the location of
 \ta schema that describes that namespace."},"tableNames":{"type":"array","description":"List of
 table names that you want to register. In case of CSV registration, list of table names to be
 skipped in zip file","items":{"type":"string"}}}},"PIIScanFilter":{"type":"object","properties":
{"profile":{"type":"string"},"schema":{"type":"string"},"tables":{"type":"array","items":
{"type":"string"}}}},"PIIScanParameters":{"type":"object","properties":{"classifierPacks":
{"type":"array","items":{"type":"integer","format":"int64"}},"connectionProfiles":{"type":"array","items":
```

{"type":"string"}},"dataSourceNames":{"type":"array","items":{"type":"string"}},"environment":
{"type":"string"},"filters":{"type":"array","items":{"$ref":"#/definitions/PIIScanFilter"}},"isIncludeFilter":
{"type":"boolean"},"jobId":{"type":"integer","format":"int64"},"jobName":{"type":"string"},"projId":
{"type":"integer","format":"int64"},"pverId":{"type":"integer","format":"int64"},"scanLevel":
{"type":"integer","format":"int32"},"storeSamples":{"type":"boolean"},"userName":
{"type":"string"}}},"PageResult«DBPiiTag»":{"type":"object","properties":{"elements":{"type":"array","items":
{"$ref":"#/definitions/DBPiiTag"}},"numberOfElements":{"type":"integer","format":"int32"},"totalElements":
{"type":"integer","format":"int64"}}},"PageResult«DataModelEntityInfo»":
{"type":"object","properties":{"elements":{"type":"array","items":{"$ref":"#/definitions/
DataModelEntityInfo"}},"numberOfElements":{"type":"integer","format":"int32"},"totalElements":
{"type":"integer","format":"int64"}}},"PageResult«EntityMaskInfo»":{"type":"object","properties":
{"elements":{"type":"array","items":{"$ref":"#/definitions/EntityMaskInfo"}},"numberOfElements":
{"type":"integer","format":"int32"},"totalElements":
{"type":"integer","format":"int64"}}},"PageResult«EntityRelationshipDetails»":
{"type":"object","properties":{"elements":{"type":"array","items":{"$ref":"#/definitions/
EntityRelationshipDetails"}},"numberOfElements":{"type":"integer","format":"int32"},"totalElements":
{"type":"integer","format":"int64"}}},"PageResult«MaskConfigGroupsByTag»":
{"type":"object","properties":{"elements":{"type":"array","items":{"$ref":"#/definitions/
MaskConfigGroupsByTag"}},"numberOfElements":{"type":"integer","format":"int32"},"totalElements":
{"type":"integer","format":"int64"}}},"PageResult«MaskFunctionGroup»":
{"type":"object","properties":{"elements":{"type":"array","items":{"$ref":"#/definitions/
MaskFunctionGroup"}},"numberOfElements":{"type":"integer","format":"int32"},"totalElements":
{"type":"integer","format":"int64"}}},"PageResult«PiiDataColumn»":{"type":"object","properties":
{"elements":{"type":"array","items":{"$ref":"#/definitions/PiiDataColumn"}},"numberOfElements":
{"type":"integer","format":"int32"},"totalElements":
{"type":"integer","format":"int64"}}},"PageResult«PiiData»":{"type":"object","properties":
{"elements":{"type":"array","items":{"$ref":"#/definitions/PiiData"}},"numberOfElements":
{"type":"integer","format":"int32"},"totalElements":
{"type":"integer","format":"int64"}}},"PageResult«PiiTable»":{"type":"object","properties":
{"elements":{"type":"array","items":{"$ref":"#/definitions/PiiTable"}},"numberOfElements":
{"type":"integer","format":"int32"},"totalElements":{"type":"integer","format":"int64"}}},"PaginatedResult":
{"type":"object","properties":{"elements":{"type":"array","items":{"type":"object"}},"numberOfElements":
{"type":"integer","format":"int32"},"totalNumberOfElements":{"type":"integer","format":"int64"}}},"PiiColumn":
{"type":"object","properties":{"columnId":{"type":"integer","format":"int64"},"columnName":
{"type":"string"},"dataType":{"type":"string"},"rowCount":{"type":"integer","format":"int64"}}},"PiiData":
{"type":"object","properties":{"columnCount":{"type":"integer","format":"int64"},"confirmed":
{"type":"boolean"},"databaseName":{"type":"string"},"dateReviewed":{"type":"string","format":"date-
time"},"matchedSamples":{"type":"integer","format":"int64"},"notPII":{"type":"boolean"},"piiTags":
{"type":"array","items":{"type":"string"}},"profileJobId":{"type":"integer","format":"int64"},"profileName":
{"type":"string"},"reason":{"type":"string"},"reviewer":{"type":"string"},"rowCount":
{"type":"integer","format":"int64"},"schemaName":{"type":"string"},"severity":
{"type":"number","format":"double"},"tableId":{"type":"integer","format":"int64"},"tableName":
{"type":"string"},"tagHistory":{"type":"array","items":{"$ref":"#/definitions/
PiiDataTagHistory"}}}},"PiiDataColumn":{"type":"object","properties":{"action":
{"type":"string","enum":["ADD","REMOVE"]},"clsMatches":{"type":"object","additionalProperties":
{"type":"integer","format":"int64"}},"clsMaxMatch":{"type":"integer","format":"int64"},"columnId":
{"type":"integer","format":"int64"},"columnName":{"type":"string"},"dataType":
{"type":"string"},"dateReviewed":{"type":"string","format":"date-time"},"piiTags":
{"type":"array","items":{"type":"string"}},"primaryTag":{"type":"string"},"reason":
{"type":"string"},"reviewer":{"type":"string"},"severity":{"type":"number","format":"double"},"tagHistory":
{"type":"array","items":{"$ref":"#/definitions/PiiDataTagHistory"}},"tagsSet":
{"type":"boolean"}}},"PiiDataTagHistory":{"type":"object","properties":{"action":
{"type":"string","enum":["ADDED","REMOVED"]},"columnId":{"type":"integer","format":"int64"},"columnName":

{"type":"string"},"dateReviewed":{"type":"string","format":"date-time"},"reason":
{"type":"string"},"reviewer":{"type":"string"},"tag":{"type":"string"}}},"PiiHeatMap":
{"type":"object","properties":{"databaseNames":{"type":"array","items":{"type":"string"}},"heat":
{"type":"object","additionalProperties":{"type":"array","items":{"type":"object"}}},"jobProgress":
{"$ref":"#/definitions/JobProgress"},"profiles":{"type":"array","items":{"$ref":"#/definitions/
ConnectionProfile"}},"schemaNames":{"type":"array","items":{"type":"string"}},"tags":{"type":"array","items":
{"type":"string"}}}},"PiiJob":{"type":"object","properties":{"approved":{"type":"boolean"},"approvedBy":
{"type":"string"},"columnClassifierHash":{"type":"integer","format":"int32"},"columnSeedlistHash":
{"type":"integer","format":"int32"},"columnsClassified":{"type":"integer","format":"int64"},"columnsScanned":
{"type":"integer","format":"int64"},"completeDate":{"type":"string","format":"date-
time"},"contentClassifierHash":{"type":"integer","format":"int32"},"contentSeedlistHash":
{"type":"integer","format":"int32"},"environment":{"type":"string"},"jobID":
{"type":"integer","format":"int64"},"jobName":{"type":"string"},"listApprovers":
{"type":"array","items":{"$ref":"#/definitions/DBPiiApprover"}},"listReviewers":
{"type":"array","items":{"$ref":"#/definitions/DBPiiReviewer"}},"projectID":
{"type":"integer","format":"int64"},"projectName":{"type":"string"},"projectVersionID":
{"type":"integer","format":"int64"},"reason":{"type":"string"},"scanLevel":
{"type":"integer","format":"int32"},"setup":{"$ref":"#/definitions/PiiSetup"},"severity":
{"type":"number","format":"double"},"signOffRequestedDate":{"type":"string","format":"date-
time"},"startDate":{"type":"string","format":"date-time"},"state":{"type":"string","enum":
["CREATED","STARTED","CANCELLING","CANCELLED","SCAN_COMPLETE","APPROVAL_REQUIRED","APPROVED","APPROVAL_REJECTED","FAILE
{"type":"string","format":"date-time"},"storeSamples":{"type":"boolean"},"submittedBy":
{"type":"string"},"tablesClassified":{"type":"integer","format":"int64"},"tablesReviewed":
{"type":"integer","format":"int64"},"tablesScanned":{"type":"integer","format":"int64"},"totalApprovers":
{"type":"integer","format":"int64"},"totalColumns":{"type":"integer","format":"int64"},"totalPii":
{"type":"integer","format":"int64"},"totalReviewers":{"type":"integer","format":"int64"},"totalTables":
{"type":"integer","format":"int64"},"warnings":{"type":"string"}}},"PiiReviewer":
{"type":"object","properties":{"accepted":{"type":"boolean"},"dateReviewed":
{"type":"string","format":"date-time"},"reason":{"type":"string"},"reviewerId":
{"type":"integer","format":"int64"},"reviewerName":{"type":"string"}}},"PiiSample":
{"type":"object","properties":{"columnId":{"type":"integer","format":"int64"},"columnName":
{"type":"string"},"columnType":{"type":"string"},"values":{"type":"array","items":{"$ref":"#/definitions/
TaggedSample"}}}},"PiiSearch":{"type":"object","properties":{"columns":{"type":"array","items":
{"type":"string"}},"profiles":{"type":"array","items":{"type":"string"}},"rowCounts":{"$ref":"#/
definitions/SearchRowCounts"},"schemas":{"type":"array","items":{"type":"string"}},"tables":
{"type":"array","items":{"type":"string"}},"tags":{"type":"array","items":{"type":"string"}}}},"PiiSetup":
{"type":"object","properties":{"RefreshToken":{"type":"string"},"classifierPacks":
{"type":"array","items":{"type":"integer","format":"int64"}},"connProfiles":{"type":"array","items":
{"type":"string"}},"dataSources":{"type":"array","items":{"type":"string"}},"environment":
{"type":"string"},"environmentId":{"type":"integer","format":"int64"},"filters":
{"type":"array","items":{"$ref":"#/definitions/PIIScanFilter"}},"isIncludeFilter":
{"type":"boolean"},"origin":{"type":"string"},"refreshToken":{"type":"string"},"scanLevel":
{"type":"integer","format":"int32"},"scanLevelSet":{"type":"boolean"},"scanNumericKeys":
{"type":"boolean"},"scanNumericKeysSet":{"type":"boolean"},"scanStringKeys":
{"type":"boolean"},"scanStringKeysSet":{"type":"boolean"},"storeSamples":{"type":"boolean"},"storeSamplesSet":
{"type":"boolean"}}},"PiiTable":{"type":"object","properties":{"columnCount":
{"type":"integer","format":"int64"},"columns":{"$ref":"#/definitions/Iterable«PiiColumn»"},"databaseName":
{"type":"string"},"rowCount":{"type":"integer","format":"int64"},"schemaName":{"type":"string"},"tableId":
{"type":"integer","format":"int64"},"tableName":{"type":"string"}}},"PrimaryKeyDetails":
{"type":"object","properties":{"primaryKeyColumns":{"type":"array","items":
{"type":"string"}},"primaryKeyName":{"type":"string"}}},"ProfileRequestResponse":
{"type":"object","properties":{"jobId":{"type":"integer","format":"int64"}}},"ReconcileReport":
{"type":"object","properties":{"conflictDescription":{"type":"string"},"conflictType":{"type":"string","enum":

```
["COLUMN","FOREIGN_KEY","PRIMARY_KEY","UNIQUE_KEY","INDEX"]}}},"RelationshipColumnDetails":
{"type":"object","properties":{"childColumn":{"type":"string","description":"Relationship child column
 name"},"parentColumn":{"type":"string","description":"Relationship parent column name"},"sequence":
{"type":"integer","format":"int64","description":"Relationship column sequence"}}},"RelationshipDetails":
{"type":"object","properties":{"childCardinality":{"type":"string","description":"Relationship
 child Cardinality"},"childTableName":{"type":"string","description":"Relationship child table
 name"},"childTableOwner":{"type":"string"},"id":{"type":"integer","format":"int64","description":"ID
 of the relationship","readOnly":true},"parentCardinality":{"type":"string","description":"Relationship
 parent Cardinality"},"parentTableName":{"type":"string","description":"Relationship
 parent table name"},"parentTableOwner":{"type":"string"},"relationshipColumns":
{"type":"array","description":"Relationship Columns","items":{"$ref":"#/definitions/
RelationshipColumnDetails"}},"relationshipDesc":{"type":"string","description":"Relationship
 description"},"relationshipName":{"type":"string","description":"Relationship
 name"},"relationshipType":{"type":"integer","format":"int64","description":"Type of
 relationship"}}},"RootElementBean":{"type":"object","properties":{"name":{"type":"string"},"portBinding":
{"type":"string"},"portBindingNameSpace":{"type":"string"},"portType":{"type":"string"}}},"SearchRowCounts":
{"type":"object","properties":{"empty":{"type":"array","items":{"type":"integer","format":"int32"}},"large":
{"type":"array","items":{"type":"integer","format":"int32"}},"medium":
{"type":"array","items":{"type":"integer","format":"int32"}},"small":{"type":"array","items":
{"type":"integer","format":"int32"}},"vlarge":{"type":"array","items":
{"type":"integer","format":"int32"}}}},"SeedListSnapshot":{"type":"object","properties":
{"created":{"type":"string","format":"date-time"},"createdBy":{"type":"string"},"modified":
{"type":"string","format":"date-time"},"modifiedBy":{"type":"string"},"revision":
{"type":"string"},"seedListDescription":{"type":"string"},"seedListId":
{"type":"integer","format":"int64"},"seedListName":{"type":"string"},"seedListOrigin":
{"type":"string"},"values":{"type":"array","items":{"type":"string"}}}},"SystemExclusion":
{"type":"object","properties":{"connectionProfile":{"type":"string"},"dataSource":
{"type":"string"},"databaseExclusions":{"type":"array","items":{"type":"string"}},"dbType":
{"type":"string"},"schemaExclusions":{"type":"array","items":{"type":"string"}}}},"SystemExclusionInfo":
{"type":"object","properties":{"environmentId":{"type":"integer","format":"int64"},"exclusions":
{"type":"array","items":{"$ref":"#/definitions/SystemExclusion"}},"projectId":
{"type":"integer","format":"int64"},"versionId":{"type":"integer","format":"int64"}}},"TableDetails":
{"type":"object","properties":{"columns":{"type":"array","description":"List of columns of the table","items":
{"$ref":"#/definitions/ColumnDetails"}},"foreignKeys":{"type":"array","description":"List of foreign Keys of
 the table","items":{"$ref":"#/definitions/ForeignKeyDetails"}},"name":{"type":"string","description":"Name
 of the table"},"order":{"type":"integer","format":"int64","description":"Order of the table"},"primaryKey":
{"description":"Primary key of the table","$ref":"#/definitions/PrimaryKeyDetails"},"relationships":
{"type":"array","description":"List of Relationships of the table","items":{"$ref":"#/definitions/
RelationshipDetails"}},"rowCount":{"type":"integer","format":"int64","description":"No
 of rows in the table"},"schema":{"type":"string","description":"Location of the table
 (schema)"},"tableId":{"type":"integer","format":"int64","description":"ID of the
 table","readOnly":true}}},"TableForeignKeyDefinition":{"type":"object","properties":
{"createFields":{"type":"string"},"dateCreated":{"type":"string","format":"date-
time"},"dateUpdated":{"type":"string","format":"date-time"},"programCreated":
{"type":"string"},"programUpdated":{"type":"string"},"tfdColumnName":{"type":"string"},"tfdColumnPos":
{"type":"integer","format":"int64"},"tfdFkeyName":{"type":"string"},"tfdFkeySeq":
{"type":"integer","format":"int64"},"tfdFkeyStatus":{"type":"string"},"tfdProjId":
{"type":"integer","format":"int64"},"tfdPvId":{"type":"integer","format":"int64"},"tfdRefColumnName":
{"type":"string"},"tfdRefSchemaName":{"type":"string"},"tfdRefTableName":
{"type":"string"},"tfdRepeater":{"type":"string"},"tfdSchemaName":{"type":"string"},"tfdTableId":
{"type":"integer","format":"int64"},"tfdTableName":{"type":"string"},"updateFields":
{"type":"string"},"whoCreated":{"type":"string"},"whoUpdated":{"type":"string"}}},"TableInfo":
{"type":"object","properties":{"differences":{"type":"string"},"diffrenceReport":{"$ref":"#/
```

definitions/TableReconcileReport"},"fileName":{"type":"string"},"status":{"type":"string"},"tableName":
{"type":"string"},"tableType":{"type":"string"}}},"TableOrderRequest":{"type":"object","required":
["override"],"properties":{"ignoredForeignKeys":{"type":"array","description":"List of
the Foreign Keys you want to ignore while creating table order","items":{"$ref":"#/
definitions/ForeignKeyDetails"}},"ignoredRelationships":{"type":"array","description":"List of
Relationships you want to ignore while creating table order","items":{"$ref":"#/definitions/
RelationshipDetails"}},"override":{"type":"boolean","example":false,"description":"Flag used
to override existing table order"}}},"TableReconcileReport":{"type":"object","properties":
{"conflicts":{"type":"array","items":{"$ref":"#/definitions/ReconcileReport"}},"tableName":
{"type":"string"}}},"TablesInfo":{"type":"object","properties":{"tableInfoList":{"type":"array","items":
{"$ref":"#/definitions/TableInfo"}},"totalNoOfTables":{"type":"integer","format":"int64"}}},"TaggedSample":
{"type":"object","properties":{"sampleType":{"type":"string"},"tags":{"type":"array","items":
{"type":"string"}},"value":{"type":"string"}}},"Timestamp":{"type":"object","properties":
{"date":{"type":"integer","format":"int32"},"day":{"type":"integer","format":"int32"},"hours":
{"type":"integer","format":"int32"},"minutes":{"type":"integer","format":"int32"},"month":
{"type":"integer","format":"int32"},"nanos":{"type":"integer","format":"int32"},"seconds":
{"type":"integer","format":"int32"},"time":{"type":"integer","format":"int64"},"timezoneOffset":
{"type":"integer","format":"int32"},"year":{"type":"integer","format":"int32"}}},"URI":
{"type":"object","properties":{"absolute":{"type":"boolean"},"authority":{"type":"string"},"fragment":
{"type":"string"},"host":{"type":"string"},"opaque":{"type":"boolean"},"path":
{"type":"string"},"port":{"type":"integer","format":"int32"},"query":{"type":"string"},"rawAuthority":
{"type":"string"},"rawFragment":{"type":"string"},"rawPath":{"type":"string"},"rawQuery":
{"type":"string"},"rawSchemeSpecificPart":{"type":"string"},"rawUserInfo":{"type":"string"},"scheme":
{"type":"string"},"schemeSpecificPart":{"type":"string"},"userInfo":{"type":"string"}}},"URL":
{"type":"object","properties":{"authority":{"type":"string"},"content":{"type":"object"},"defaultPort":
{"type":"integer","format":"int32"},"file":{"type":"string"},"host":{"type":"string"},"path":
{"type":"string"},"port":{"type":"integer","format":"int32"},"protocol":{"type":"string"},"query":
{"type":"string"},"ref":{"type":"string"},"userInfo":{"type":"string"}}},"WhereClauseInfo":
{"type":"object","properties":{"attributeId":{"type":"integer","format":"int64"},"maskFunctionId":
{"type":"integer","format":"int64"},"maskGroupId":{"type":"integer","format":"int64"},"whereClause":
{"type":"string"},"whereClauseId":{"type":"integer","format":"int64"}}}}}

TDMProjectService

none

{"swagger":"2.0","info":{"description":"This section includes the APIs that perform various
operations for the projects.It also provides the REST API URL for the respective operation along
with sample request and response body content.","version":"1.0","title":"CA TDM Projects Service
API","termsOfService":"http://ca.com","contact":{"name":"CA Technologies"},"license":{"name":"The CA
License Version 2.0","url":"https://ca.com/LICENSE"}},"host":"far-demo.dhcp.broadcom.net:8443","basePath":"/
TDMProjectService","tags":[{"name":"program-controller","description":"Program Controller"},{"name":"version-
controller","description":"Version Controller"},{"name":"event-subscription-controller","description":"Event
Subscription Controller"},{"name":"project-controller","description":"Interface for projects"},
{"name":"variable-controller","description":"Interface for variables"}],"paths":{"/api/ca/v1/
eventSubscription":{"get":{"tags":["event-subscription-controller"],"summary":"Interface for get an
event subscription by subscription id","description":"Use this interface to get an event subscription by
id","operationId":"getSubscriptionByServiceAndEventTypeUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},

{"name":"service","in":"query","description":"service","required":true,"type":"string"},
{"name":"event","in":"query","description":"event","required":true,"type":"string"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/EventSubscribeResponse"}},"400":{"description":"Bad
 Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions
 to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found
 - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}},"post":{"tags":["event-subscription-controller"],"summary":"Interface for registering
 an event notification endpoint","description":"Use this interface to register an event notification
 endpoint","operationId":"eventSubscribeUsingPOST","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"request","description":"Request
 body for subscribing to an event","required":true,"schema":{"$ref":"#/definitions/
EventSubscribeRequest"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/
EventSubscribeResponse"}},"400":{"description":"Bad Request - Request does not have a valid format
 or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the
 resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not
 found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error -
 Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/
ca/v1/eventSubscription/{subId}":{"get":{"tags":["event-subscription-controller"],"summary":"Interface for
 get an event subscription by subscription id","description":"Use this interface to get an event subscription
 by id","operationId":"getSubscriptionByIdUsingGET","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"subId","in":"path","description":"subscription
 id","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/EventSubscribeResponse"}},"400":{"description":"Bad
 Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions
 to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found
 - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}},"delete":{"tags":["event-subscription-controller"],"summary":"Interface for
 get an event subscription by subscription id","description":"Use this interface to get an event
 subscription by id","operationId":"deleteSubscriptionByIdUsingDELETE","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic
 HTTP authorization scheme. The API responds with a security token, which is valid for
 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
 access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"subId","in":"path","description":"subscription
 id","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"type":"string"}},"400":{"description":"Bad Request - Request

does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/projects":{"get":{"tags":["project-controller"],"summary":"Interface to get all the projects for a given user.","description":"Use this interface to get all the projects for a given user. Returns an empty list if none found.","operationId":"getUserProjectsUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"includePermissions","in":"query","description":"Flag to include project permissions on each project","required":false,"type":"boolean"}],"responses":{"200":{"description":"Success.","schema":{"type":"array","items":{"$ref":"#/definitions/ProjectInfo"}}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["project-controller"],"summary":"Interface to create new project","description":"Use this interface to create a new project.","operationId":"createProjectUsingPOST","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"projectInfo","description":"Request body for creating a project.\n Mandatory parameters are: \n name: Specify project name, accepts strings; \ndescription: Specify project description, accepts strings","required":true,"schema":{"$ref":"#/definitions/ProjectInfo"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/ProjectResult"}},"201":{"description":"Created.","schema":{"$ref":"#/definitions/ProjectResult"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - A Project with the specified name already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/projects/{projectId}":{"get":{"tags":["project-controller"],"summary":"Interface to get the details of a project","description":"Use this interface to get the project details.(for example, project settings, description, and levels) based on project ID.","operationId":"getProjectInfoUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"Id of the project that you want to use to get project details.","required":true,"type":"integer","format":"int64"},{"name":"includePermissions","in":"query","description":"Flag to include project permissions on each project. defaults to False","required":false,"type":"boolean"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/ProjectInfo"}},"400":{"description":"Bad

Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Project not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"put":{"tags":["project-controller"],"summary":"Interface for updating the project name, description and job limit","description":"Use this interface to update the project name, description and job limit","operationId":"updateProjectUsingPUT","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"Id of the project that you want to update.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"project","description":"Project object containing the key value pairs of the properties that you want to update.","required":false,"schema":{"$ref":"#/definitions/ProjectInfo"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/ProjectResult"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Project with ID not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - A project with the specified name already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["project-controller"],"summary":"Interface to delete project","description":"Use this interface to delete a project","operationId":"deleteUsingDELETE","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"ID of the project to be deleted","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success","schema":{"type":"object","additionalProperties":{"type":"object"}}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Project with ID not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Cannot delete the project because associated jobs are still in running state.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/projects/{projectId}/variables":{"get":{"tags":["variable-controller"],"summary":"Interface to get all the variables in a Project. Supports paginated response.","description":"Use this interface to get all the variables in a Project. Supports paginated response which is optional.","operationId":"getProjectVariablesUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"Id of the project for which variables have to be retrieved.","required":true,"type":"integer","format":"int64"},

{"name":"page","in":"query","description":"Page number to retrieve in a paginated variables
result. Indexed with 0. Optional.","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"Page size of each page with which
you want to retrieve in a paginated variables result. Default value is 20.
Optional.","required":false,"type":"integer","format":"int32"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/PaginatedVariableBean"}},"400":{"description":"Bad
Request - Request does not have a valid format or has missing required parameters.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired
token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does
not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["variable-controller"],"summary":"Interface
to create new variable in a project","description":"Use this interface to create a new variable in a
project.","operationId":"createProjectVariableUsingPOST","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login
interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API
responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer
HTTP authorization scheme to access any protected resource through this API on behalf of the user. For
Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"Id
of the project for which variable has to be created.","required":true,"type":"integer","format":"int64"},
{"name":"validate","in":"query","description":"Set this parameter to true to validate the expressions used in
the variable","required":false,"type":"boolean"},{"in":"body","name":"variableInfo","description":"Request
body for creating a variable in a project.\n Mandatory parameters are: \n name: Specify variable
name, accepts strings; \ndescription: Specify variable description, accepts strings; \n defaultValue:
Specify default value for the variable, accepts strings; \nresolvePriorToPublish: Specify if variable
should be resolved prior to publishing, accepts true or false","required":true,"schema":{"$ref":"#/
definitions/VariableBean"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/
VariableBean"}},"201":{"description":"Created.","schema":{"$ref":"#/definitions/VariableBean"}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have
permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":
{"description":"Conflict - A Variable with the specified name already exists.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included
in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/projects/
{projectId}/variables/{variableName}":{"get":{"tags":["variable-controller"],"summary":"Interface to get
details of a variable in a Project.","description":"Use this interface to get details of a variable in a
Project.","operationId":"getProjectVariableDetailsUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login
interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API
responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer
HTTP authorization scheme to access any protected resource through this API on behalf of the user. For
Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"Id
of the project for which variable has to be retrieved.","required":true,"type":"integer","format":"int64"},
{"name":"variableName","in":"path","description":"Name of the variable whose details have to be
retrieved.","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/VariableBean"}},"400":{"description":"Bad Request - Request does not have a valid
format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the
project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found -
Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal

Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":{"tags":["variable-controller"],"summary":"Interface to update details of a variable in a Project.","description":"Use this interface to update details of a variable in a Project.","operationId":"updateProjectVariableDetailsUsingPUT","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"Id of the project for which variable has to be updated.","required":true,"type":"integer","format":"int64"},{"name":"variableName","in":"path","description":"Name of the variable whose details have to be updated.","required":true,"type":"string"},{"name":"validate","in":"query","description":"Set this parameter to true to validate the expressions used in the variable","required":false,"type":"boolean"},{"in":"body","name":"variableInfo","description":"Request body for creating a variable in a project. \n Mandatory parameters are: \n name: Specify variable name, accepts strings; \ndescription: Specify variable description, accepts strings; \n defaultValue: Specify default value for the variable, accepts strings; \nresolvePriorToPublish: Specify if variable should be resolved prior to publishing, accepts true or false","required":true,"schema":{"$ref":"#/definitions/VariableBean"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/VariableBean"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["variable-controller"],"summary":"Interface to delete project variables","description":"Use this interface to delete variables in a project","operationId":"deleteProjectVariableUsingDELETE","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"Id of the project for which the variable has to be deleted.","required":true,"type":"integer","format":"int64"},{"name":"variableName","in":"path","description":"Name of the variable to be deleted.","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":{"type":"object"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Variable not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/projects/{projectId}/versions":{"get":{"tags":["version-controller"],"summary":"Interface for getting a list of versions for a specific project","description":"Use this interface to get a list of versions for a specified project.","operationId":"getVersionsUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"Id of the project that you want to use to get list of versions.","required":true,"type":"integer","format":"int64"}],"responses":{"200":

{"description":"Success.","schema":{"type":"array","items":{"$ref":"#/definitions/VersionInfo"}}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have
 permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["version-controller"],"summary":"Interface
 to create project versions","description":"Use this interface to create versions in a
 project","operationId":"createUsingPOST","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"versionInfo","description":"Request
 body for creating a version. \n Mandatory parameters is\n name: Specify version name,
 accepts strings","required":true,"schema":{"$ref":"#/definitions/VersionInfo"}},
{"name":"projectId","in":"path","description":"projectId","required":true,"type":"integer","format":"int64"},
{"name":"copyFromVersion","in":"query","description":"Id of the version to copy the data
 from","required":false,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/VersionInfo"}},"201":{"description":"Created.","schema":{"$ref":"#/definitions/
VersionInfo"}},"400":{"description":"Bad Request - Specific reason is included in the error
 message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid
 or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User
 does not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":
{"description":"Conflict - A Version with the specified name already exists.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included
 in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/projects/
{projectId}/versions/{versionId}":{"get":{"tags":["version-controller"],"summary":"Interface for getting
 the details of a version","description":"Use this interface to get the details of a version under a
 project.","operationId":"getVersionUsingGET","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"Id of the
 project that you want to use to get version details.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"path","description":"Id of the project version for which you want
 to get details.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/VersionInfo"}},"400":{"description":"Bad
 Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access
 the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource
 not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error -
 Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":
{"tags":["version-controller"],"summary":"Interface to update project versions","description":"Use
 this interface to update version information","operationId":"updateUsingPUT","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through

this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"versionInfo","description":"Request body for updating a version. \n Mandatory
parameter is\n name: Specify new version name, accepts strings","required":true,"schema":
{"$ref":"#/definitions/VersionInfo"}},{"name":"projectId","in":"path","description":"Id of the
project","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"path","description":"Id
of the version to be updated","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/VersionInfo"}},"400":{"description":"Bad
Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access
the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Version
with ID not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - A
Version with the specified name already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["version-controller"],"summary":"Interface
to delete project versions","description":"Use this interface to delete versions in a
project","operationId":"deleteUsingDELETE_1","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
authorization scheme to access any protected resource through this API on behalf of the user. For Example:
Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"ID of the
project","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"path","description":"ID
of the version to be deleted","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success.","schema":{"type":"object"}},"400":{"description":"Bad Request - Specific
reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the
project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Version
with ID not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict
- Cannot delete the version because associated jobs are still in running state.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in
the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/projects/{projectId}/
versions/{versionId}/actions/upgrade":{"post":{"tags":["version-controller"],"summary":"Interface
to upgrade a version in a project","description":"Use this interface to upgrade a version in a
project.","operationId":"upgradeUsingPOST","consumes":["application/json"],"produces":["application/
json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
authorization scheme to access any protected resource through this API on behalf of the user. For Example:
Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"requestBody","description":"Request
body for upgrade.\n Mandatory fields: upgradeFromVersionId: Specify the version ID
from which this version need to be upgraded","required":true,"schema":{"$ref":"#/
definitions/VersionUpgradeModel"}},{"name":"projectId","in":"path","description":"ID of the
project","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"path","description":"ID
of the version to be upgraded","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"Success.","schema":{"type":"object"}},"400":{"description":"Bad
Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have
permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":

{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/projects/{projectId}/versions/{versionId}/programs":{"get":{"tags":["program-controller"],"summary":"Interface for getting all saved programs","description":"Use this interface to get all saved programs, user can also filter on language (SQL, JAVELIN)","operationId":"getAllProgramsUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"Project Id ","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"path","description":"Version Id","required":true,"type":"integer","format":"int64"},{"name":"language","in":"query","description":"Language parameter to filter program, values can be like SQL, JAVELIN","required":false,"type":"string"}],"responses":{"200":{"description":"Success.","schema":{"type":"array","items":{"$ref":"#/definitions/GtrepProgram"}}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/projects/{projectId}/versions/{versionId}/programs/{programId}":{"get":{"tags":["program-controller"],"summary":"Interface for getting saved programs bu Id","description":"Use this interface to get saved programs","operationId":"getProgramUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"Project Id ","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"path","description":"Version Id","required":true,"type":"integer","format":"int64"},{"name":"programId","in":"path","description":"Program Id","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/GtrepProgram"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/projects/{projectId}/versions/{versionId}/variables":{"get":{"tags":["variable-controller"],"summary":"Interface to get all the variables in a Project Version. Supports paginated response.","description":"Use this interface to get all the variables in a Project Version. Supports paginated response which is optional.","operationId":"getVersionVariablesUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"Id of the project for which variables have to be retrieved.","required":true,"type":"integer","format":"int64"},{"name":"versionId","in":"path","description":"Id of the version for which variables have to be retrieved.","required":true,"type":"integer","format":"int64"},{"name":"page","in":"query","description":"Page number to retrieve in a paginated variables result. Indexed with 0. Optional.","required":false,"type":"integer","format":"int32"},

{"name":"size","in":"query","description":"Page size of each page with which
you want to retrieve in a paginated variables result. Default value is 20.
Optional.","required":false,"type":"integer","format":"int32"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/PaginatedVariableBean"}},"400":{"description":"Bad
Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions
to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found
- Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}},"post":{"tags":["variable-controller"],"summary":"Interface to create new variable
in a project version","description":"Use this interface to create a new variable in a project
version.","operationId":"createProjectVersionVariableUsingPOST","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login
interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API
responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer
HTTP authorization scheme to access any protected resource through this API on behalf of the user. For
Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"Id
of the project for which variable has to be created.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"path","description":"Id of the version for which
variable has to be created.","required":true,"type":"integer","format":"int64"},
{"name":"validate","in":"query","description":"Set this parameter to true to validate the expressions used in
the variable","required":false,"type":"boolean"},{"in":"body","name":"variableInfo","description":"Request
body for creating a variable in a project.\n Mandatory parameters are: \n name: Specify variable
name, accepts strings; \ndescription: Specify variable description, accepts strings; \n defaultValue:
Specify default value for the variable, accepts strings; \nresolvePriorToPublish: Specify if variable
should be resolved prior to publishing, accepts true or false","required":true,"schema":{"$ref":"#/
definitions/VariableBean"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/
VariableBean"}},"201":{"description":"Created.","schema":{"$ref":"#/definitions/VariableBean"}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to
access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - A
Variable with the specified name already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/projects/{projectId}/versions/{versionId}/
variables/{variableName}":{"get":{"tags":["variable-controller"],"summary":"Interface to get details of
a variable in a Project Version.","description":"Use this interface to get details of a variable in a
Project Version.","operationId":"getProjectVersionVariableDetailsUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"Id of the
project for which variable has to be retrieved.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"path","description":"Id of the version for which
variable has to be retrieved.","required":true,"type":"integer","format":"int64"},
{"name":"variableName","in":"path","description":"Name of the variable whose details have to be
retrieved.","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/VariableBean"}},"400":{"description":"Bad Request - Request does not have a valid
format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/

ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the
project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found -
Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}},"put":{"tags":["variable-controller"],"summary":"Interface to update details of a
variable in a Project Version.","description":"Use this interface to update details of a variable in a
Project Version.","operationId":"updateProjectVersionVariableDetailsUsingPUT","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"Id of the
project for which variable has to be updated.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"path","description":"Id of the version for which
variable has to be updated.","required":true,"type":"integer","format":"int64"},
{"name":"variableName","in":"path","description":"Name of the variable whose details have to be
updated.","required":true,"type":"string"},{"name":"validate","in":"query","description":"Set this
parameter to true to validate the expressions used in the variable","required":false,"type":"boolean"},
{"in":"body","name":"variableInfo","description":"Request body for creating a variable in a project.
\n Mandatory parameters are: \n name: Specify variable name, accepts strings; \ndescription: Specify
variable description, accepts strings; \n defaultValue: Specify default value for the variable, accepts
strings; \nresolvePriorToPublish: Specify if variable should be resolved prior to publishing, accepts
true or false","required":true,"schema":{"$ref":"#/definitions/VariableBean"}}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/VariableBean"}},"400":{"description":"Bad
Request - Request does not have a valid format or has missing required parameters.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired
token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does
not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["variable-controller"],"summary":"Interface
to delete version variables","description":"Use this interface to delete variables in a
version","operationId":"deleteVersionVariableUsingDELETE","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login
interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API
responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer
HTTP authorization scheme to access any protected resource through this API on behalf of the user. For
Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"path","description":"Id
of the project for which the variable has to be deleted.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"path","description":"Id of the version for which the
variable has to be deleted.","required":true,"type":"integer","format":"int64"},
{"name":"variableName","in":"path","description":"Name of the variable to be
deleted.","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":
{"type":"object"}},"400":{"description":"Bad Request - Specific reason is included in the error
message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid
or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User
does not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Variable not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}}},"definitions":{"ErrorResponse":{"type":"object","properties":
{"errorCode":{"type":"string"},"errorDetail":{"type":"string"},"errorMsg":{"type":"string"},"status":
{"type":"integer","format":"int32"},"timestamp":{"type":"string"}}},"EventSubscribeRequest":

{"type":"object","properties":{"desc":{"type":"string"},"endpoint":{"type":"string"},"event":
{"type":"string"},"id":{"type":"integer","format":"int64"},"priority":
{"type":"integer","format":"int32"},"requestBodyTemplate":{"type":"string"},"rollbackEndpoint":
{"type":"string"},"rollbackRequestBodyTemplate":{"type":"string"},"rollbackUrlParams":
{"type":"string"},"rollbackVerb":{"type":"string"},"service":{"type":"string"},"urlParams":
{"type":"string"},"verb":{"type":"string"}}},"EventSubscribeResponse":{"type":"object","properties":
{"desc":{"type":"string"},"endpoint":{"type":"string"},"event":{"type":"string"},"id":
{"type":"integer","format":"int64"},"priority":{"type":"integer","format":"int32"},"requestBodyTemplate":
{"type":"string"},"rollbackEndpoint":{"type":"string"},"rollbackRequestBodyTemplate":
{"type":"string"},"rollbackUrlParams":{"type":"string"},"rollbackVerb":{"type":"string"},"service":
{"type":"string"},"urlParams":{"type":"string"},"verb":{"type":"string"}}},"GtrepProgram":
{"type":"object","properties":{"created":{"type":"string","format":"date-time"},"createdBy":
{"type":"string"},"dbms":{"type":"string"},"language":{"type":"string"},"name":{"type":"string"},"params":
{"type":"array","items":{"$ref":"#/definitions/GtrepProgramParam"}},"programCreated":
{"type":"string"},"programId":{"type":"integer","format":"int64"},"programSource":
{"type":"array","items":{"$ref":"#/definitions/GtrepProgramSource"}},"programUpdated":
{"type":"string"},"projectId":{"type":"integer","format":"int64"},"targetOs":{"type":"string"},"template":
{"type":"string"},"transMap":{"type":"string"},"type":{"type":"string"},"updated":
{"type":"string","format":"date-time"},"updatedBy":{"type":"string"},"versionId":
{"type":"integer","format":"int64"}}},"GtrepProgramParam":{"type":"object","properties":
{"paramId":{"type":"integer","format":"int64"},"paramName":{"type":"string"},"paramValue":
{"type":"string"},"projectId":{"type":"integer","format":"int64"},"projectVersionId":
{"type":"integer","format":"int64"}}},"GtrepProgramSource":{"type":"object","properties":{"date":
{"type":"string","format":"date-time"},"id":{"type":"integer","format":"int64"},"programSourceId":{"$ref":"#/
definitions/GtrepProgramSourcePK"},"projectId":{"type":"integer","format":"int64"},"projectVersionId":
{"type":"integer","format":"int64"},"savedSql":{"$ref":"#/definitions/
RepositoryClob"}}},"GtrepProgramSourcePK":{"type":"object","properties":{"date":
{"type":"string","format":"date-time"},"id":{"type":"integer","format":"int64"}}},"PaginatedVariableBean":
{"type":"object","properties":{"elements":{"type":"array","items":{"$ref":"#/definitions/
VariableBean"}},"numberOfElements":{"type":"integer","format":"int32"},"totalNumberOfElements":
{"type":"integer","format":"int64"}}},"ProjectInfo":{"type":"object","properties":
{"created":{"type":"string","description":"Timestamp of the project creation"},"dateOrder":
{"type":"string","description":"Date order of the project"},"description":
{"type":"string","description":"Description of the project"},"grantedFunctions":
{"type":"array","description":"List of granted functions","items":{"type":"string"}},"id":
{"type":"integer","format":"int64","description":"Id of the project"},"inheritTables":
{"type":"boolean","example":false,"description":"Indicates whether project inherits
 tables"},"jobLimit":{"type":"integer","format":"int32","description":"Maximum number of
 jobs that can be executed concurrently"},"levels":{"type":"array","items":{"$ref":"#/
definitions/ProjectLevel"}},"name":{"type":"string","description":"Name of the
 project"},"timestampPrecision":{"type":"number","description":"Timestamp precision of the project"},"type":
{"type":"string","description":"Type of the project"},"updated":{"type":"string","description":"Timestamp
 of the last update on project"},"versions":{"type":"array","items":{"$ref":"#/definitions/
VersionInfo"}}}},"ProjectLevel":{"type":"object","properties":{"created":{"type":"string","description":"Date
 of creation of the level"},"hasData":{"type":"string","description":"Field indicating whether
 it is capable of generating the data"},"keyOrder":{"type":"string","description":"Key Order of
 the level"},"level":{"type":"integer","format":"int64","description":"Level number"},"name":
{"type":"string","description":"Name of the level"},"publish":{"type":"string","description":"Field
 inidicationg if it supports publish"},"updated":{"type":"string","description":"Date of last updation of the
 level"}}},"ProjectResult":{"type":"object","properties":{"created":{"type":"string","description":"Timestamp
 of the project creation"},"dateOrder":{"type":"string","description":"Date order of the
 project"},"description":{"type":"string","description":"Description of the project"},"id":
{"type":"integer","format":"int64","description":"Id of the project"},"inheritTables":

{"type":"string","description":"Indicates whether project inherits tables"},"jobLimit":
{"type":"integer","format":"int32","description":"Maximum number of jobs that can be executed
concurrently"},"levels":{"type":"array","description":"Levels under the project","items":
{"$ref":"#/definitions/ProjectLevel"}},"name":{"type":"string","description":"Name of the
project"},"timestampPrecision":{"type":"number","description":"Timestamp precision of the project"},"type":
{"type":"string","description":"Type of the project"},"updated":{"type":"string","description":"Timestamp
of the last update on project"}}},"RepositoryClob":{"type":"object","properties":
{"clobData":{"type":"string"},"clobId":{"type":"integer","format":"int64"},"clobProjId":
{"type":"number"},"clobSize":{"type":"number"},"clobTimestamp":{"type":"string","format":"date-
time"},"dateCreated":{"type":"string","format":"date-time"},"dateUpdated":
{"type":"string","format":"date-time"},"programCreated":{"type":"string"},"programUpdated":
{"type":"string"},"whoCreated":{"type":"string"},"whoUpdated":{"type":"string"}}},"VariableBean":
{"type":"object","required":["defaultValue","description","name","type"],"properties":
{"defaultValue":{"type":"string","description":"Default Value of the variable"},"description":
{"type":"string","description":"Description of the variable"},"displayType":
{"type":"string","description":"Display type","enum":
["TextBox","CheckBox","DropDownList","MultiSelectList","RadioButton","DatePicker"]},"helpMessage":
{"type":"string","description":"Help message"},"isDisplayOnly":
{"type":"boolean","example":false,"description":"Variable value is read-only at runtime"},"isOptional":
{"type":"boolean","example":false,"description":"Is Optional"},"listDefinition":
{"type":"string","description":"Definition for list values"},"name":{"type":"string","description":"Name
of the variable"},"resolvePriorToPublish":{"type":"boolean","example":false,"description":"Resolve
this variable prior to publish"},"scope":{"type":"string","description":"Scope
of the variable","readOnly":true},"type":{"type":"string","description":"Type
of the variable","enum":["String","Number","Date","Boolean"]},"validation":
{"type":"string","description":"Validation rule for the variable value"}}},"VersionInfo":
{"type":"object","properties":{"created":{"type":"string","description":"Version created
date"},"description":{"type":"string","description":"Description of the version"},"id":
{"type":"integer","format":"int64","description":"ID of the version","readOnly":true},"levelDetails":
{"type":"array","description":"List of Version Level Details","items":{"$ref":"#/
definitions/VersionLevelDetails"}},"name":{"type":"string","description":"Name of the
version"},"projectName":{"type":"string","description":"Name of the Project to which the Version
belongs"},"registeredObjectCount":{"type":"integer","format":"int32","description":"Registered
Object count under a Version"},"tablesUsed":{"type":"array","description":"List of Used Tables
under the version","items":{"type":"string"}}}},"VersionLevelDetails":{"type":"object","properties":
{"level":{"type":"integer","format":"int64","description":"Level of the version"},"levelCount":
{"type":"integer","format":"int32","description":"Level Count of the version"},"levelName":
{"type":"string","description":"Level Name of the version"}}},"VersionUpgradeModel":
{"type":"object","required":["upgradeFromVersionId"],"properties":{"removeExistingData":
{"type":"boolean","example":false,"description":"Specify whether existing definition rows in
the target version will be removed in case of conflicting generators"},"upgradeFromVersionId":
{"type":"integer","format":"int64","description":"Id of the version to upgrade from"}}}}}

TDMvDataService

none

{"swagger":"2.0","info":{"description":"This section includes the APIs that perform various VTDM
operations. It also provides the REST API URL for the respective operation along with sample request
and response body content.","version":"1.0","title":"CA TDM vData Service API","termsOfService":"http://
ca.com","contact":{"name":"CA Technologies"},"license":{"name":"The CA License Version 2.0","url":"https://
ca.com/LICENSE"}},"host":"far-demo.dhcp.broadcom.net:8443","basePath":"/TDMvDataService","tags":[{"name":"v-
data-controller","description":"V Data Controller"}],"paths":{"/api/ca/v1/appliances":{"get":{"tags":["v-

data-controller"],"summary":"Interface to get vTDM appliances","description":"Use this interface to get
 a list of appliances.","operationId":"getAppliancesUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/
login interface to perform a user login using user credentials in the Basic HTTP authorization scheme.
 The API responds with a security token, which is valid for 24 hours by default. Use the security token
 in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf
 of the user. For Example: Bearer {{token}}","required":true,"type":"string"}],"responses":{"200":
{"description":"OK","schema":{"$ref":"#/definitions/Iterable«Appliance»"}}}},"post":{"tags":["v-data-
controller"],"summary":"Interface to register a vTDM appliance","description":"Use this interface to
 register an appliance.","operationId":"addApplianceUsingPOST","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"postAppliance","description":"postAppliance","required":true,"schema":{"$ref":"#/
definitions/Appliance"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/
Appliance"}}}}},"/api/ca/v1/checkpoints":{"get":{"tags":["v-data-controller"],"summary":"Interface
 to get vTDM checkpoints","description":"Use this interface to get a list of all checkpoints for all
 filesystems from all appliances.","operationId":"getCheckpointsUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"visible","in":"query","description":"visible","required":false,"type":"boolean"}],"responses":
{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/DBCheckpoint"}}}}},"post":
{"tags":["v-data-controller"],"summary":"Interface to create a vTDM checkpoint","description":"Use this
 interface to create a new checkpoint.","operationId":"createCheckpointUsingPOST","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"postCheckpoint","description":"postCheckpoint","required":true,"schema":{"$ref":"#/
definitions/DBCheckpoint"}},
{"name":"validate","in":"query","description":"validate","required":false,"type":"boolean"}],"responses":
{"200":{"description":"OK","schema":{"$ref":"#/definitions/Checkpoint"}}}}},"/api/ca/v1/checkpoints/
profile/{profile}":{"get":{"tags":["v-data-controller"],"summary":"Interface to find the checkpoint
 that uses a specified connection profile","description":"Use this interface to check if a profile
 is in use before deletion","operationId":"checkIfConprofInUseUsingGET","consumes":["application/
json"],"produces":["text/plain"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"profile","in":"path","description":"profile","required":true,"type":"string"}],"responses":
{"200":{"description":"OK","schema":{"type":"string"}}}}},"/api/ca/v1/checkpoints/{checkpoint}":{"get":
{"tags":["v-data-controller"],"summary":"Interface to get a single vTDM checkpoint","description":"Use this
 interface to get an existing checkpoint.","operationId":"getCheckpointUsingGET","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the

security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"checkpoint","in":"path","description":"checkpoint","required":true,"type":"integer","format":"int64"}],"respon{"200":{"description":"OK","schema":{"$ref":"#/definitions/DBCheckpoint"}}}},"delete":{"tags":["v-data-controller"],"summary":"Interface to delete a vTDM checkpoint","description":"Use this interface to delete a checkpoint.","operationId":"deleteCheckpointUsingDELETE","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"checkpoint","in":"path","description":"checkpoint","required":true,"type":"integer","format":"int64"}],"respon{"200":{"description":"OK","schema":{"type":"boolean"}}}},"patch":{"tags":["v-data-controller"],"summary":"Interface to update a vTDM checkpoint","description":"Use this interface to update an existing checkpoint.","operationId":"patchCheckpointUsingPATCH","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"checkpoint","in":"path","description":"checkpoint","required":true,"type":"integer","format":"int64"},{"in":"body","name":"patchCheckpoint","description":"patchCheckpoint","required":true,"schema":{"$ref":"#/definitions/DBCheckpoint"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/Checkpoint"}}}}},"/api/ca/v1/clonelog":{"get":{"tags":["v-data-controller"],"summary":"Interface to retrieve a log of recent clone activity","description":"Use this interface to retrieve a log of recent clone activity.","operationId":"cloneLogUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/InputStreamResource"}}}}},"/api/ca/v1/clones":{"get":{"tags":["v-data-controller"],"summary":"Interface to get all vTDM clones","description":"Use this interface to get a list of all clones from all checkpoints from all filesystems on all appliances","operationId":"getClonesUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"projectId","in":"query","description":"projectId","required":false,"type":"integer","format":"int64"},{"name":"details","in":"query","description":"details","required":false,"type":"boolean"}],"responses":{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/DBClone"}}}}},"post":{"tags":["v-data-controller"],"summary":"Interface to create a vTDM clone","description":"Use this interface to create a new clone.","operationId":"createCloneUsingPOST","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"postClone","description":"postClone","required":true,"schema":{"$ref":"#/definitions/DBClone"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/Clone"}}}}},"/api/ca/v1/clones/{clone}":{"delete":{"tags":["v-data-

controller"],"summary":"Interface to delete a vTDM clone","description":"Use this interface to
 delete a clone.","operationId":"deleteCloneUsingDELETE","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login
 interface to perform a user login using user credentials in the Basic HTTP authorization scheme.
 The API responds with a security token, which is valid for 24 hours by default. Use the security
 token in the Bearer HTTP authorization scheme to access any protected resource through this
 API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"clone","in":"path","description":"clone","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"OK","schema":{"type":"boolean"}}}}},"/api/ca/v1/clones/{clone}/email":{"post":{"tags":
["v-data-controller"],"summary":"Interface to email vTDM clone details to someone","description":"Use
 this interface to email clone details.","operationId":"emailCloneUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"clone","in":"path","description":"clone","required":true,"type":"integer","format":"int64"},
{"name":"email","in":"query","description":"email","required":false,"type":"string"}],"responses":
{"200":{"description":"OK","schema":{"type":"boolean"}}}}},"/api/ca/v1/clones/{id}":{"get":{"tags":["v-
data-controller"],"summary":"Interface to get a specific vTDM clone","description":"Use this interface
 to get details for an individual clone","operationId":"getCloneUsingGET","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"id","in":"path","description":"id","required":true,"type":"integer","format":"int64"},
{"name":"details","in":"query","description":"details","required":false,"type":"boolean"}],"responses":
{"200":{"description":"OK","schema":{"$ref":"#/definitions/DBClone"}}}}},"/api/ca/v1/clones/{id}/
log":{"get":{"tags":["v-data-controller"],"summary":"Interface to get the attach/detach log for a
 specific vTDM clone","description":"Use this interface to get the clone creation and destruction
 detailed logs entries","operationId":"getCloneLogUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"id","in":"path","description":"id","required":true,"type":"integer","format":"int64"}],"responses":
{"200":{"description":"OK","schema":{"type":"array","items":{"$ref":"#/definitions/
VDataJobLogEntry"}}}}}}},"/api/ca/v1/filesystems":{"get":{"tags":["v-data-controller"],"summary":"Interface
 to get vTDM filesystems","description":"Use this interface to get a list of
 filesystems.","operationId":"getFilesystemsUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login
 interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API
 responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer
 HTTP authorization scheme to access any protected resource through this API on behalf of the user. For
 Example: Bearer {{token}}","required":true,"type":"string"}],"responses":{"200":{"description":"OK","schema":
{"type":"array","items":{"$ref":"#/definitions/DBFilesystem"}}}}},"post":{"tags":["v-data-
controller"],"summary":"Interface to create a vTDM filesystem","description":"Use this interface to create a
 filesystem on a particular appliance.","operationId":"createFilesystemUsingPOST","consumes":["application/
json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the

security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"postFilesystem","description":"postFilesystem","required":true,"schema":{"$ref":"#/definitions/DBFilesystem"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/Filesystem"}}}}},"/api/ca/v1/filesystems/{fs}":{"delete":{"tags":["v-data-controller"],"summary":"Interface to delete a vTDM filesystem","description":"Use this interface to delete a filesystem.","operationId":"deleteFilesystemUsingDELETE","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"fs","in":"path","description":"fs","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"OK","schema":{"type":"boolean"}}}}},"/api/ca/v1/status":{"get":{"tags":["v-data-controller"],"summary":"Interface to get vTDM appliance state","description":"Use this interface to get a list of appliance states.","operationId":"getStatusUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/Iterable«Status»"}}}}}},"definitions":{"Appliance":{"type":"object","properties":{"id":{"type":"integer","format":"int64"},"name":{"type":"string"},"password":{"type":"string"},"revision":{"type":"integer","format":"int32"},"uuid":{"type":"string"},"version":{"type":"string"}}},"Checkpoint":{"type":"object","properties":{"applianceHostname":{"type":"string"},"branch":{"type":"string"},"checkpoint":{"type":"string"},"cloneNames":{"type":"array","items":{"type":"string"}},"created":{"type":"string","format":"date-time"},"description":{"type":"string"},"filesystem":{"type":"string"},"oracle_data":{"$ref":"#/definitions/OracleData"},"originFilesystem":{"type":"string"},"referenced":{"type":"integer","format":"int64"},"used":{"type":"string"},"user":{"type":"string"},"uuid":{"type":"string"}}},"Clone":{"type":"object","properties":{"applianceHostname":{"type":"string"},"basename":{"type":"string"},"branch":{"type":"string"},"cloneCheckpointBranchedFrom":{"type":"string"},"cloneDbPassword":{"type":"string"},"cloneDbUsername":{"type":"string"},"compression":{"type":"string"},"created":{"type":"string","format":"date-time"},"description":{"type":"string"},"files":{"type":"array","items":{"type":"string"}},"lastRevert":{"type":"string"},"logicalUsed":{"type":"string"},"name":{"type":"string"},"nfiles":{"type":"integer","format":"int64"},"nfspath":{"type":"string"},"oracle_data":{"$ref":"#/definitions/OracleData"},"originCheckpoint":{"type":"string"},"originFilesystem":{"type":"string"},"uncpath":{"type":"string"},"used":{"type":"string"},"user":{"type":"string"},"uuid":{"type":"string"}}},"ConnectionProfile":{"type":"object","required":["dbType","name","password","server","username"],"properties":{"additionalConnectionProperties":{"type":"string","description":"JDBC connection string properties. Applicable only for database type db2/400 sql"},"baseUrl":{"type":"string","description":"Base URL connection"},"connectionProperties":{"type":"object","description":"Connection Properties","additionalProperties":{"type":"object"}},"created":{"type":"string","format":"date-time","description":"Creation date"},"createdBy":{"type":"integer","format":"int64","description":"Created by"},"database":{"type":"string","description":"Database name"},"datasourceDriver":{"type":"string","description":"DataSource Driver"},"datasourceUrl":{"type":"string","description":"DataSource URL"},"dbType":{"type":"string","description":"Type of database","enum":["sql server","oracle","mysql","sybase","teradata","db2","db2/400 sql"]},"description":{"type":"string","description":"Descriptive text"},"instance":{"type":"string","description":"Sql server instance name"},"integratedSecurity":{"type":"boolean","example":false,"description":"Use Integrated Security for authentication. Applicable only for database type SQL Server"},"modified":{"type":"string","format":"date-time","description":"Last modified date"},"name":{"type":"string","description":"Name of the connection profile"},"password":

{"type":"string","description":"Password"},"port":{"type":"string","description":"Database
server port"},"schema":{"type":"string","description":"Sql server schema name"},"server":
{"type":"string","description":"Database server hostname"},"service":{"type":"string","description":"Oracle
service name"},"username":{"type":"string","description":"Username"}}},"DBCheckpoint":
{"type":"object","properties":{"applianceHostname":{"type":"string"},"branch":
{"type":"string"},"checkpoint":{"type":"string"},"cloneDbPassword":{"type":"string"},"cloneNames":
{"type":"array","items":{"type":"string"}},"connectionProfile":{"$ref":"#/definitions/
ConnectionProfile"},"created":{"type":"string","format":"date-time"},"description":
{"type":"string"},"filesystem":{"type":"string"},"filesystemId":{"type":"integer","format":"int64"},"id":
{"type":"integer","format":"int64"},"name":{"type":"string"},"oracle_data":
{"$ref":"#/definitions/OracleData"},"originFilesystem":{"type":"string"},"referenced":
{"type":"integer","format":"int64"},"rootPassword":{"type":"string"},"rootUser":
{"type":"string"},"timeToClone":{"type":"integer","format":"int64"},"used":{"type":"string"},"user":
{"type":"string"},"uuid":{"type":"string"},"visible":{"type":"boolean"}}},"DBClone":
{"type":"object","properties":{"applianceHostname":{"type":"string"},"attachJobId":
{"type":"integer","format":"int64"},"basename":{"type":"string"},"branch":{"type":"string"},"checkpointId":
{"type":"integer","format":"int64"},"cloneCheckpointBranchedFrom":{"type":"string"},"cloneDbPassword":
{"type":"string"},"cloneDbUsername":{"type":"string"},"compression":{"type":"string"},"created":
{"type":"string","format":"date-time"},"createdBy":{"type":"string"},"description":{"type":"string"},"files":
{"type":"array","items":{"type":"string"}},"id":{"type":"integer","format":"int64"},"jdbcurl":
{"type":"string"},"lastRevert":{"type":"string"},"logicalUsed":{"type":"string"},"misc":{"$ref":"#/
definitions/Misc"},"name":{"type":"string"},"nfiles":{"type":"integer","format":"int64"},"nfspath":
{"type":"string"},"oracleData":{"$ref":"#/definitions/OracleData"},"oracle_data":{"$ref":"#/definitions/
OracleData"},"originCheckpoint":{"type":"string"},"originFilesystem":{"type":"string"},"originFilesystemId":
{"type":"integer","format":"int64"},"projectId":{"type":"integer","format":"int64"},"smbpath":
{"type":"string"},"status":{"type":"string"},"uncpath":{"type":"string"},"used":{"type":"string"},"user":
{"type":"string"},"userId":{"type":"integer","format":"int64"},"uuid":{"type":"string"}}},"DBFilesystem":
{"type":"object","properties":{"applance":{"$ref":"#/definitions/Appliance"},"applianceHostname":
{"type":"string"},"applianceId":{"type":"integer","format":"int64"},"avail":{"type":"string"},"created":
{"type":"string","format":"date-time"},"description":{"type":"string"},"files":{"type":"array","items":
{"type":"string"}},"id":{"type":"integer","format":"int64"},"lastRollback":{"type":"string"},"misc":
{"$ref":"#/definitions/Misc"},"name":{"type":"string"},"nfiles":{"type":"integer","format":"int64"},"nfspath":
{"type":"string"},"projectID":{"type":"integer","format":"int64"},"smbpath":{"type":"string"},"uncpath":
{"type":"string"},"used":{"type":"string"},"user":{"type":"string"},"uuid":{"type":"string"}}},"File":
{"type":"object","properties":{"absolute":{"type":"boolean"},"absoluteFile":{"$ref":"#/
definitions/File"},"absolutePath":{"type":"string"},"canonicalFile":{"$ref":"#/
definitions/File"},"canonicalPath":{"type":"string"},"directory":{"type":"boolean"},"file":
{"type":"boolean"},"freeSpace":{"type":"integer","format":"int64"},"hidden":{"type":"boolean"},"name":
{"type":"string"},"parent":{"type":"string"},"parentFile":{"$ref":"#/definitions/
File"},"path":{"type":"string"},"totalSpace":{"type":"integer","format":"int64"},"usableSpace":
{"type":"integer","format":"int64"}}},"Filesystem":{"type":"object","properties":{"applianceHostname":
{"type":"string"},"avail":{"type":"string"},"created":{"type":"string","format":"date-
time"},"description":{"type":"string"},"files":{"type":"array","items":{"type":"string"}},"lastRollback":
{"type":"string"},"name":{"type":"string"},"nfiles":{"type":"integer","format":"int64"},"nfspath":
{"type":"string"},"projectID":{"type":"integer","format":"int64"},"uncpath":{"type":"string"},"used":
{"type":"string"},"user":{"type":"string"},"uuid":{"type":"string"}}},"InputStream":
{"type":"object"},"InputStreamResource":{"type":"object","properties":{"description":
{"type":"string"},"file":{"$ref":"#/definitions/File"},"filename":{"type":"string"},"inputStream":
{"$ref":"#/definitions/InputStream"},"open":{"type":"boolean"},"readable":{"type":"boolean"},"uri":
{"$ref":"#/definitions/URI"},"url":{"$ref":"#/definitions/URL"}}},"Iterable«Appliance»":
{"type":"object"},"Iterable«Status»":{"type":"object"},"Misc":{"type":"object","properties":
{"sharePassword":{"type":"string"},"shareUsername":{"type":"string"}}},"OracleData":
{"type":"object","properties":{"dataFiles":{"type":"object","additionalProperties":{"type":"array","items":

{"type":"string"}}},"sourceSchema":{"type":"string"},"sourceTablespaces":{"type":"array","items":
{"type":"string"}},"systemDump":{"type":"string"},"tablespaceDump":{"type":"string"}}},"Status":
{"type":"object","properties":{"alloc":{"type":"integer","format":"int64"},"applianceHostname":
{"type":"string"},"error":{"type":"string"},"expandsz":{"type":"integer","format":"int64"},"frag":
{"type":"string"},"free":{"type":"integer","format":"int64"},"health":{"type":"string"},"id":
{"type":"integer","format":"int64"},"numClonesCreated":{"type":"integer","format":"int64"},"revision":
{"type":"integer","format":"int32"},"roiTimeSaved":{"type":"integer","format":"int64"},"roiTimeUsed":
{"type":"integer","format":"int64"},"size":{"type":"integer","format":"int64"},"totalCloneSavedBytes":
{"type":"integer","format":"int64"},"totalCloneUsedBytes":{"type":"integer","format":"int64"},"version":
{"type":"string"}}},"URI":{"type":"object","properties":{"absolute":{"type":"boolean"},"authority":
{"type":"string"},"fragment":{"type":"string"},"host":{"type":"string"},"opaque":{"type":"boolean"},"path":
{"type":"string"},"port":{"type":"integer","format":"int32"},"query":{"type":"string"},"rawAuthority":
{"type":"string"},"rawFragment":{"type":"string"},"rawPath":{"type":"string"},"rawQuery":
{"type":"string"},"rawSchemeSpecificPart":{"type":"string"},"rawUserInfo":{"type":"string"},"scheme":
{"type":"string"},"schemeSpecificPart":{"type":"string"},"userInfo":{"type":"string"}}},"URL":
{"type":"object","properties":{"authority":{"type":"string"},"content":{"type":"object"},"defaultPort":
{"type":"integer","format":"int32"},"file":{"type":"string"},"host":{"type":"string"},"path":
{"type":"string"},"port":{"type":"integer","format":"int32"},"protocol":{"type":"string"},"query":
{"type":"string"},"ref":{"type":"string"},"userInfo":{"type":"string"}}},"VDataJobLogEntry":
{"type":"object","properties":{"cloneId":{"type":"integer","format":"int64"},"created":
{"type":"string","format":"date-time"},"details":{"type":"string"},"jobId":
{"type":"integer","format":"int64"},"level":{"type":"string"},"source":{"type":"string"}}}}}

TestDataManager

none

{"swagger":"2.0","info":{"description":"The TDM API provides basic operations on the Portal
framework.","version":"1.0","title":"CA TDM API","termsOfService":"http://ca.com","contact":
{"name":"CA Technologies"},"license":{"name":"The CA License Version 2.0","url":"https://
ca.com/LICENSE"}},"host":"far-demo.dhcp.broadcom.net:8443","basePath":"/TestDataManager","tags":
[{"name":"security-controller","description":"Interface for Users and Groups Management "},
{"name":"token-introspection-controller","description":"Token Introspection Controller"},{"name":"license-
controller","description":"License Controller"},{"name":"reservation-controller","description":"Interface
for Search and Reserve Data "},{"name":"auth-controller","description":"Auth Controller"},{"name":"variable-
controller","description":"Variable Controller"},{"name":"settings-controller","description":"Settings
Controller"},{"name":"event-subscription-controller","description":"Event Subscription Controller"},
{"name":"audit-logs-controller","description":"Interface to retrieve audit logs"},{"name":"integration-
accounts-controller","description":"Integration Accounts Controller"}],"paths":{"/api/ca/v1/activate":
{"post":{"tags":["license-controller"],"summary":"activate","operationId":"activateUsingPOST","consumes":
["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"licensingModel","in":"query","description":"Licensing Model","required":true,"type":"string"},
{"name":"siteId","in":"query","description":"Site Id","required":true,"type":"string"}],"responses":{"200":
{"description":"Success","schema":{"$ref":"#/definitions/ResponseEntity"}},"400":{"description":"Bad Request
- Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included
in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/auditlogs":

{"get":{"tags":["audit-logs-controller"],"summary":"Interface for getting a list of audit logs either as JSON or as a zipped CSV file","description":"Use this interface to retrieve a list of audit logs as JSON or as a zipped CSV file. You can select this using the format parameter. For the JSON format, the audit logs will be retrieved in pages, with a selectable page number and page size. You can also filter the audit logs by various filter parameters. For filter parameters of type String, e.g. origin, you can use wildcard characters: % to match zero, one, or multiple characters and _ to match a single parameter. For the % wildcard, use its URL encoding namely %25. To treat wildcard characters literally, enclose them in [], viz. [%25] and [_].","operationId":"getAuditLogsUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"format","in":"query","description":"Determines the format in which audit logs are returned. This can either be JSON (the default) or ZIP-CSV.","required":false,"type":"string"},{"name":"origin","in":"query","description":"Filter list by origin. Can contain wildcard characters.","required":false,"type":"string"},{"name":"description","in":"query","description":"Filter list by description. Can contain wildcard characters.","required":false,"type":"string"},{"name":"type","in":"query","description":"Filter list by type. Can contain wildcard characters.","required":false,"type":"string"},{"name":"status","in":"query","description":"Filter list by status. Can contain wildcard characters.","required":false,"type":"string"},{"name":"user_name","in":"query","description":"Filter list by user_name. Can contain wildcard characters.","required":false,"type":"string"},{"name":"link_id","in":"query","description":"Filter list by link id. Pass id or string null.","required":false,"type":"string"},{"name":"proj_id","in":"query","description":"Filter list by project id. Pass id or string null.","required":false,"type":"string"},{"name":"proj_version_id","in":"query","description":"Filter list by project version id. Pass id or string null.","required":false,"type":"string"},{"name":"timestamp_start","in":"query","description":"Filter list by logs later than start timestamp. Timestamp should be in the format yyyy-mm-dd hh:mm:ss[.fffffffff].","required":false,"type":"string"},{"name":"timestamp_end","in":"query","description":"Filter list by logs earlier than end timestamp. Timestamp should be in the format yyyy-mm-dd hh:mm:ss[.fffffffff]","required":false,"type":"string"},{"name":"sort","in":"query","description":"Column on which to sort the results. This can be one of origin, description, type, status, user_name, link_id, project_id, project_version_id or timestamp.","required":false,"type":"string"},{"name":"order","in":"query","description":"Order on which to sort the results. This can either be ASC or DESC, with the default being ASC.","required":false,"type":"string"},{"name":"page","in":"query","description":"Page number to retrieve for paginated list of audit logs. Default value is 1. Applicable only for JSON format.","required":false,"type":"integer","format":"int32"},{"name":"pagesize","in":"query","description":"Page size of each page to retrieve in paginated list of audit logs. Default value is 25. Applicable only for JSON format.","required":false,"type":"integer","format":"int32"},{"name":"cachesize","in":"query","description":"This is a tuning parameter for performance. This can range from 1 to 100,000, with a default value of 1,000. If you have a very large number of logs, a larger value should increase performance but will use more memory.","required":false,"type":"integer","format":"int32"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/PaginatedAuditLogsResult"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User needs administrator privileges to access the audit logs.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"When this REST end point is down or not accessible.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Check logs for more information.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["audit-logs-

controller"],"summary":"writeAuditLog","operationId":"writeAuditLogUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"log","description":"log","required":true,"schema":{"$ref":"#/definitions/
AuditLog"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/
AuditLog"}}}}},"/api/ca/v1/connectionProfiles/search/{profileName}":{"post":{"tags":["reservation-
controller"],"summary":"Interface for getting the searched data","description":"Use this interface to
 get the searched data.","operationId":"searchDataUsingPOST","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login
 interface to perform a user login using user credentials in the Basic HTTP authorization scheme.
 The API responds with a security token, which is valid for 24 hours by default. Use the security
 token in the Bearer HTTP authorization scheme to access any protected resource through this
 API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"profileName","in":"path","description":"Name of the connection profile that you want to search
 data.","required":true,"type":"string"},{"name":"projectId","in":"query","description":"ID of the project
 that is associated to the sql , you want to execute.","required":true,"type":"integer","format":"int64"},
{"name":"versionId","in":"query","description":"ID of the project version that is associated
 to the sql , you want to execute.","required":true,"type":"integer","format":"int64"},
{"name":"page","in":"query","description":"Page number which you want to retrieve in a
 paginated result. Default value is 0.","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"Page size of each page with which you want to retrieve
 paginated result. Default value is 15.","required":false,"type":"integer","format":"int32"},
{"name":"storedSql","in":"query","description":"Strored Sql template Name","required":true,"type":"string"},
{"in":"body","name":"body","description":"body","required":false,"schema":{"$ref":"#/definitions/
SearchDataInputs"}}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/
PaginatedResultsDTO"}},"400":{"description":"Bad Request - Specific reason is included in the
 error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server
 authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
 - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is
 included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/
eventSubscription":{"get":{"tags":["event-subscription-controller"],"summary":"Interface for get an
 event subscription by subscription id","description":"Use this interface to get an event subscription by
 id","operationId":"getSubscriptionByServiceAndEventTypeUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"service","in":"query","description":"service","required":true,"type":"string"},
{"name":"event","in":"query","description":"event","required":true,"type":"string"}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/EventSubscribeResponse"}},"400":{"description":"Bad
 Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions
 to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found
 - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}},"post":{"tags":["event-subscription-controller"],"summary":"Interface for registering

an event notification endpoint","description":"Use this interface to register an event notification endpoint","operationId":"eventSubscribeUsingPOST","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"request","description":"Request body for subscribing to an event","required":true,"schema":{"$ref":"#/definitions/EventSubscribeRequest"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/EventSubscribeResponse"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/eventSubscription/{subId}":{"get":{"tags":["event-subscription-controller"],"summary":"Interface for get an event subscription by subscription id","description":"Use this interface to get an event subscription by id","operationId":"getSubscriptionByIdUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"subId","in":"path","description":"subscription id","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/EventSubscribeResponse"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["event-subscription-controller"],"summary":"Interface for get an event subscription by subscription id","description":"Use this interface to get an event subscription by id","operationId":"deleteSubscriptionByIdUsingDELETE","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"subId","in":"path","description":"subscription id","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success.","schema":{"type":"string"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/groups":{"get":{"tags":["security-controller"],"summary":"Interface for getting list of groups","description":"Use this interface to get the list of groups.","operationId":"getGroupsUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login

interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"page","in":"query","description":"Page number which you want to retrieve in a paginated groups result. Default value is 0.","required":false,"type":"integer","format":"int32"},{"name":"size","in":"query","description":"Page size of each page with which you want to retrieve paginated groups result. Default value is 15.","required":false,"type":"integer","format":"int32"},{"name":"sortDir","in":"query","description":"Sorting order with which you want the sort the paginated groups result. Valid values are ASC and DESC.","required":false,"type":"string"},{"name":"sortField","in":"query","description":"Field on which you want to apply the sorting for the paginated groups result.","required":false,"type":"string"},{"name":"searchText","in":"query","description":"Search text you want to use to search on user group name and description to get paginated group result.","required":false,"type":"string"},{"name":"projectId","in":"query","description":"Project Id to filter out the user groups specific to a project.","required":false,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/PaginatedGroupDTO"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["security-controller"],"summary":"Interface for creating new groups","description":"Use this interface to create new group.","operationId":"createGroupUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"group","description":"Group to create.","required":true,"schema":{"$ref":"#/definitions/GroupDTO"}},{"name":"projectId","in":"query","description":"Id of the project which this group will be associated. If not provided the group will be assoicated to all projects","required":false,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/GroupDTO"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - User with user name already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/groups/{groupId}":{"get":{"tags":["security-controller"],"summary":"Interface for getting Details of a group","description":"Use this interface to get the details of a group.","operationId":"getGroupUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"groupId","in":"path","description":"ID of the group you want to get.","required":true,"type":"integer","format":"int64"},{"name":"projectId","in":"query","description":"Project Id to filter out the user groups specific to a project.","required":false,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/GroupDTO"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":

{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - User with user name already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":{"tags":["security-controller"],"summary":"Interface for editing Group","description":"Use this interface to edit Group.","operationId":"editGroupUsingPUT","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"groupId","in":"path","description":"ID of the group to be updated.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"group","description":"Group to edit.","required":false,"schema":{"$ref":"#/definitions/GroupDTO"}},{"name":"projectId","in":"query","description":"Id of the project which this group will be associated. If not provided the group will be assoicated to all projects","required":false,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/GroupDTO"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"405":{"description":"Method Not Allowed - Not Allowed to edit Admin Group.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - User with user name already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["security-controller"],"summary":"Interface for Deleting group","description":"Use this interface to delete a group.","operationId":"deleteGroupUsingDELETE","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"groupId","in":"path","description":"ID of the group you want to delete.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/GroupDTO"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"405":{"description":"Method Not Allowed - Not Allowed to delete Admin group.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - User with user name already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/groups/{groupId}/actions/addLdapUsers":{"post":{"tags":["security-controller"],"summary":"Interface for adding AD/LDAP users to a CA TDM Portal user group","description":"Use this interface to add AD/LDAP users to a CA TDM Portal user group.","operationId":"addGroupToLdapUsersUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"groupId","in":"path","description":"ID of the CA TDM Portal user group to which you want to add the AD/LDAP users.","required":true,"type":"integer","format":"int64"},{"name":"authorityName","in":"query","description":"Authority name of the AD/LDAP users that you want to add. Supported value is 'Default'","required":true,"type":"string"},

{"in":"body","name":"userNames","description":"List of AD/LDAP users to be added","required":true,"schema":{"type":"array","items":{"$ref":"#/definitions/LdapUserDTO"}}}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/LDAPConfigurationResult"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"412":{"description":"Precondition Failed - AD/LDAP server configuration is not valid.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/groups/{groupId}/actions/addUsers":{"post":{"tags":["security-controller"],"summary":"Interface for adding Users to a group","description":"Use this interface to add users to a group.","operationId":"addGroupToUsersUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"groupId","in":"path","description":"ID of the group you want to get the Users .","required":true,"type":"integer","format":"int64"},{"in":"body","name":"users","description":"List of users to be added","required":true,"schema":{"type":"array","items":{"$ref":"#/definitions/UserDTO"}}}],"responses":{"200":{"description":"Success","schema":{"type":"object"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - User with user name already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/groups/{groupId}/actions/deleteUsers":{"post":{"tags":["security-controller"],"summary":"Interface for removing users membership from a group","description":"Use this interface to remove users membership from a group.","operationId":"deleteGroupFromUsersUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"groupId","in":"path","description":"ID of the group you want to get the Users .","required":true,"type":"integer","format":"int64"},{"in":"body","name":"users","description":"List of users to be removed ","required":true,"schema":{"type":"array","items":{"$ref":"#/definitions/UserDTO"}}}],"responses":{"200":{"description":"Success","schema":{"type":"object"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/groups/{groupId}/actions/getExternalGroups":{"get":{"tags":["security-controller"],"summary":"Interface for getting external (AD/LDAP) groups associated with a CA TDM Portal user group","description":"Use this interface to get the list of external (AD/LDAP) groups that are associated with a specific CA TDM Portal user group.","operationId":"getExternalGroupsByGroupUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through

this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"groupId","in":"path","description":"ID of the CA TDM Portal user group for which you want to
get the associated external (AD/LDAP) groups.","required":true,"type":"integer","format":"int64"},
{"name":"page","in":"query","description":"Page number that you want to retrieve in the
paginated result. Default value is 0.","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"Page size of each page with which you want to retrieve
the paginated result. Default value is 15.","required":false,"type":"integer","format":"int32"},
{"name":"sortDir","in":"query","description":"Sorting order with which you want to sort
the paginated result. Valid values are ASC and DESC.","required":false,"type":"string"},
{"name":"sortField","in":"query","description":"Field on which you want to apply the sorting for the
paginated result.","required":false,"type":"string"},{"name":"searchText","in":"query","description":"Search
text you want to use to search to get the paginated result.","required":false,"type":"string"},
{"name":"externalGroupsNotInGroup","in":"query","description":"Set this flag to true to get the list of
external (AD/LDAP) groups that are not associated with the specified CA TDM Portal user group. Defaults
to false.","required":false,"type":"boolean"}],"responses":{"200":{"description":"Success","schema":
{"$ref":"#/definitions/PaginatedExternalGroupDTO"}},"400":{"description":"Bad Request - Specific
reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"409":{"description":"Conflict - User with user name already exists.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason
is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/
groups/{groupId}/actions/getUsers":{"get":{"tags":["security-controller"],"summary":"Interface for
getting users inside a group","description":"Use this interface to get the list of users inside
a group.","operationId":"getUsersByGroupUsingGET","consumes":["application/json"],"produces":["*/
*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to
perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
with a security token, which is valid for 24 hours by default. Use the security token in the Bearer
HTTP authorization scheme to access any protected resource through this API on behalf of the user. For
Example: Bearer {{token}}","required":true,"type":"string"},{"name":"groupId","in":"path","description":"ID
of the group you want to get the Users .","required":true,"type":"integer","format":"int64"},
{"name":"page","in":"query","description":"Page number which you want to retrieve in a paginated
data groups result. Default value is 0.","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"Page size of each page with which you want to retrieve
paginated groups result. Default value is 15.","required":false,"type":"integer","format":"int32"},
{"name":"sortDir","in":"query","description":"Sorting order with which you want the sort the
paginated groups result. Valid values are ASC and DESC.","required":false,"type":"string"},
{"name":"sortField","in":"query","description":"Field on which you want to apply the sorting for the paginated
groups result.","required":false,"type":"string"},{"name":"searchText","in":"query","description":"Search
text you want to use to search on generator name and description to get paginated groups
result.","required":false,"type":"string"},{"name":"usersNotInGroup","in":"query","description":"Set
this flag to true to get the list of users who do not belong to this group. Defaults to
false.","required":false,"type":"boolean"}],"responses":{"200":{"description":"Success","schema":
{"$ref":"#/definitions/PaginatedUserDTO"}},"400":{"description":"Bad Request - Specific reason is included
in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server
authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found -
Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":
{"description":"Conflict - User with user name already exists.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error
message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/groups/{groupId}/actions/
mapExternalGroups":{"post":{"tags":["security-controller"],"summary":"Interface for mapping external (AD/
LDAP) groups to a CA TDM Portal user group","description":"Use this interface to map external (AD/LDAP)
groups to a CA TDM Portal user group.","operationId":"addGroupToExternalGroupsUsingPOST","consumes":

["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"groupId","in":"path","description":"ID of the CA TDM Portal user group that you want to map to the external (AD/LDAP) user groups.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"externalGroups","description":"List of external (AD/LDAP) groups that you want to map to the specified CA TDM Portal user group. For more information about the parameters used in this object, click Model in the Data Type column.","required":true,"schema":{"type":"array","items":{"$ref":"#/definitions/ExternalGroupMapDTO"}}}],"responses":{"200":{"description":"Success","schema":{"type":"object"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - User name already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/groups/{groupId}/actions/unmapExternalGroups":{"post":{"tags":["security-controller"],"summary":"Interface for removing the mapping between external (AD/LDAP) groups and a CA TDM Portal user group","description":"Use this interface to remove the mapping between external (AD/LDAP) groups and a CA TDM Portal user group.","operationId":"deleteGroupFromExternalGroupsUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"groupId","in":"path","description":"ID of the CA TDM Portal user group from which you want to remove the mapping of external (AD/LDAP) groups.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"externalGroups","description":"List of external (AD/LDAP) groups that you want to remove from the mapping with the CA TDM Portal user group. For more information about the parameters used in this object, click Model in the Data Type column. ","required":true,"schema":{"type":"array","items":{"$ref":"#/definitions/ExternalGroupUnmapDTO"}}}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/StringResponse"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/integrationAccounts":{"post":{"tags":["integration-accounts-controller"],"summary":"Interface to create integration account.","description":"Use this API to create integration account.","operationId":"createIntegrationAccountUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"integrationAccount","description":"integrationAccount","required":true,"schema":{"$ref":"#/definitions/IntegrationAccount"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/IntegrationAccount"}}}}},"/api/ca/v1/integrationAccounts/{id}":{"get":{"tags":["integration-accounts-controller"],"summary":"Interface to get account information.","description":"Use this API to get the account information for the requested id.","operationId":"getAccountUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the

security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"id","in":"path","description":"id","required":true,"type":"string"}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/IntegrationAccount"}}}},"put":{"tags":["integration-accounts-controller"],"summary":"Interface to update the integration account.","description":"Use this API to update the integration account.","operationId":"updateAccountUsingPUT","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"integrationAccount","description":"integrationAccount","required":true,"schema":{"$ref":"#/definitions/IntegrationAccount"}},{"name":"id","in":"path","description":"id","required":true,"type":"string"}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/definitions/IntegrationAccount"}}}}},"/api/ca/v1/integrationAccounts/{id}/details":{"get":{"tags":["integration-accounts-controller"],"summary":"getAccountDetails","operationId":"getAccountDetailsUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"id","in":"path","description":"id","required":true,"type":"string"}],"responses":{"200":{"description":"OK","schema":{"type":"object"}}}}},"/api/ca/v1/license":{"get":{"tags":["license-controller"],"summary":"getLicense","operationId":"getLicenseUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/ResponseEntity"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/reservations":{"get":{"tags":["reservation-controller"],"summary":"Interface for getting reservation details from testmart","description":"Use this interface to get details of reservation from testmart.","operationId":"getReservationDetailUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"profileName","in":"query","description":"Name of the connection profile in which the testmart that you want to get the reservation detail.","required":true,"type":"string"},{"name":"environmentName","in":"query","description":"Name of the environment in which the testmart that you want to get the reservation detail.","required":true,"type":"string"},{"name":"userName","in":"query","description":"userName of the users for which the testmart that you want to get the reservation detail.","required":false,"type":"string"},{"name":"reservationType","in":"query","description":"reservationType of the users for which the testmart that you want to get the reservation detail.","required":false,"type":"string"},{"name":"reportKeys","in":"query","description":"reportKeys/primaryKeys

```
 of the users for which the testmart that you want to get the reservation
 detail.","required":false,"items":{"type":"object","additionalProperties":{"type":"string"}}},
{"name":"dateCreated","in":"query","description":"dateCreated  of the users for which the
 testmart that you want to get the reservation detail.","required":false,"type":"string"},
{"name":"testMartName","in":"query","description":"Table Name of the Test Mart in
 the Environment,Default value is MINI_MART.","required":false,"type":"string"},
{"name":"page","in":"query","description":"Page number which you want to retrieve in a
 paginated result. Default value is 0.","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"Page size of each page with which you want to retrieve
 paginated result. Default value is 15.","required":false,"type":"integer","format":"int32"}],"responses":
{"200":{"description":"Success","schema":{"$ref":"#/definitions/PaginatedReservationsDTO"}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access
 the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource
 not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error -
 Specific reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"post":
{"tags":["reservation-controller"],"summary":"Interface for reserving data from testmart","description":"Use
 this interface to reserve data from testmart.","operationId":"reserveDataUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"profileName","in":"query","description":"Name of the connection profile in which
 the testmart that you want to reserve data exists.","required":true,"type":"string"},
{"in":"body","name":"body","description":"body","required":false,"schema":{"$ref":"#/definitions/
Reservation"}}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/
Reservation"}},"400":{"description":"Bad Request - Specific reason is included in the error
 message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication
 failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
 - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"409":{"description":"Conflict  - Resource conflict found.","schema":{"$ref":"#/
definitions/PrimaryKeysError"}},"500":{"description":"Internal Server Error - Specific reason is included
 in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":{"tags":["reservation-
controller"],"summary":"Interface for modifying reservation data from testmart","description":"Use this
 interface to modify reservation data from testmart.","operationId":"modifyReserveDataUsingPUT","consumes":
["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"profileName","in":"query","description":"Name of the connection profile in which
 the testmart that you want to modify reserve data exists.","required":true,"type":"string"},
{"in":"body","name":"body","description":"body","required":false,"schema":{"$ref":"#/definitions/
Reservation"}}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/
Reservation"}},"400":{"description":"Bad Request - Specific reason is included in the error
 message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication
 failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
 - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"409":{"description":"Conflict  - Resource conflict found.","schema":{"$ref":"#/definitions/
```

PrimaryKeysError"}},"500":{"description":"Internal Server Error - Specific reason is included in the
 error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["reservation-
controller"],"summary":"Interface for unreserving data from testmart","description":"Use this interface
 to unreserve data from testmart.","operationId":"unreserveDataUsingDELETE","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"profileName","in":"query","description":"Name of the connection profile in which
 the testmart that you want to unreserve data exists.","required":true,"type":"string"},
{"in":"body","name":"body","description":"body","required":false,"schema":{"$ref":"#/definitions/
Reservation"}}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/
Reservation"}},"400":{"description":"Bad Request - Specific reason is included in the error
 message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication
 failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
 - User does not have permissions to access the resource","schema":{"$ref":"#/definitions/
ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"409":{"description":"Conflict - Resource conflict found.","schema":{"$ref":"#/
definitions/PrimaryKeysError"}},"500":{"description":"Internal Server Error - Specific reason is
 included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/
reservations/actions/bulkModify":{"post":{"tags":["reservation-controller"],"summary":"Interface for
 modifying reservation data from testmart","description":"Use this interface to modify bulk reservation
 data from testmart.","operationId":"modifyReserveBulkDataUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"profileName","in":"query","description":"Name of the connection profile in which
 the testmart that you want to modify reserve data exists.","required":true,"type":"string"},
{"in":"body","name":"body","description":"body","required":false,"schema":{"type":"array","items":{"$ref":"#/
definitions/BulkReservation"}}},
{"name":"partial","in":"query","description":"partial","required":true,"type":"boolean"}],"responses":{"200":
{"description":"Success","schema":{"type":"array","items":{"$ref":"#/definitions/BulkReservation"}}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access
 the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource
 not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Resource
 conflict found.","schema":{"$ref":"#/definitions/PrimaryKeysError"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}},"/api/ca/v1/reservations/actions/bulkReserve":{"post":{"tags":["reservation-
controller"],"summary":"Interface for reserving data from testmart","description":"Use this interface
 to reserve bulk data from testmart.","operationId":"reserveBulkDataUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"profileName","in":"query","description":"Name of the connection profile in which
 the testmart that you want to reserve data exists.","required":true,"type":"string"},
{"in":"body","name":"body","description":"body","required":false,"schema":{"type":"array","items":{"$ref":"#/

definitions/BulkReservation"}}},
{"name":"partial","in":"query","description":"partial","required":true,"type":"boolean"}],"responses":{"200":
{"description":"Success","schema":{"type":"array","items":{"$ref":"#/definitions/BulkReservation"}}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access
 the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource
 not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Resource
 conflict found.","schema":{"$ref":"#/definitions/PrimaryKeysError"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}},"/api/ca/v1/reservations/actions/bulkUnreserve":{"post":{"tags":["reservation-
controller"],"summary":"Interface for unreserving data from testmart","description":"Use this interface to
 unreserve bulk data from testmart.","operationId":"unreserveBulkDataUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"profileName","in":"query","description":"Name of the connection profile in which
 the testmart that you want to unreserve data exists.","required":true,"type":"string"},
{"in":"body","name":"body","description":"body","required":false,"schema":{"type":"array","items":{"$ref":"#/
definitions/BulkReservation"}}},
{"name":"partial","in":"query","description":"partial","required":true,"type":"boolean"}],"responses":{"200":
{"description":"Success","schema":{"type":"array","items":{"$ref":"#/definitions/BulkReservation"}}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access
 the resource","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource
 not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Resource
 conflict found.","schema":{"$ref":"#/definitions/PrimaryKeysError"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}},"/api/ca/v1/security/authorities/{authorityName}/actions/getLdapGroups":{"get":{"tags":
["security-controller"],"summary":"Interface for getting the list of AD/LDAP user groups","description":"Use
 this interface to get the list of AD/LDAP user groups.","operationId":"getLdapGroupsUsingGET","consumes":
["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use
 the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"authorityName","in":"path","description":"Authority name of the AD/LDAP sever that is
 associated with the AD/LDAP user groups. Supported value is 'Default'","required":true,"type":"string"},
{"name":"page","in":"query","description":"Page number that you want to retrieve in the
 paginated result. Default value is 1.","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"Page size of each page with which you want to retrieve
 the paginated result. Default value is 15.","required":false,"type":"integer","format":"int32"},
{"name":"sortDir","in":"query","description":"Sorting order that you want to use to sort
 the paginated result. Valid values are ASC and DESC.","required":false,"type":"string"},
{"name":"searchText","in":"query","description":"Search text you want to use to search on the AD/LDAP user
 group name and description to get the paginated result.","required":false,"type":"string"}],"responses":
{"200":{"description":"Success","schema":{"$ref":"#/definitions/PaginatedLdapUserGroup"}},"400":
{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific

reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/security/authorities/{authorityName}/actions/getLdapUsers":{"get":{"tags":["security-controller"],"summary":"Interface for getting list of AD/LDAP users","description":"Use this interface to get the list of AD/LDAP users.","operationId":"getLdapUsersUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"authorityName","in":"path","description":"Authority name of the AD/LDAP sever that is associated with the AD/LDAP users. Supported value is 'Default'","required":true,"type":"string"},{"name":"page","in":"query","description":"Page number that you want to retrieve in the paginated result. Default value is 1.","required":false,"type":"integer","format":"int32"},{"name":"size","in":"query","description":"Page size of each page with which you want to retrieve the paginated result. Default value is 15.","required":false,"type":"integer","format":"int32"},{"name":"sortDir","in":"query","description":"Sorting order that you want to use to sort the paginated result. Valid values are ASC and DESC.","required":false,"type":"string"},{"name":"searchText","in":"query","description":"Search text you want to use to search on the AD/LDAP user name and description to get the paginated result.","required":false,"type":"string"}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/PaginatedLdapUserGroup"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/settings/applicationProperties/{propertyName}":{"get":{"tags":["settings-controller"],"summary":"Get application properties","description":"Get application properties","operationId":"getApplicationPropertiesUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"propertyName","in":"path","description":"propertyName","required":true,"type":"string"}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/Property"}},"404":{"description":"Failed to get application properties","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/settings/dbseedlist":{"get":{"tags":["settings-controller"],"summary":"Get the settings for a Database Seedlist","description":"Use this interface to get the settings for a Database Seedlist","operationId":"getDBSeedlistSettingsUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/DBSeedlistSettings"}}}},"post":{"tags":["settings-controller"],"summary":"Set the settings for a Database Seedlist","description":"Use this interface to set the settings for a Database Seedlist","operationId":"setDBSeedlistSettingsUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"newSettings","description":"DBSeedlist Settings","required":true,"schema":{"$ref":"#/definitions/DBSeedlistSettings"}}],"responses":{"200":{"description":"Success","schema":

{"$ref":"#/definitions/DBSeedlistSettings"}},"400":{"description":"Missing required parameters. Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict. Entity was modified by other transaction.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/settings/devTestServer":{"get":{"tags":["settings-controller"],"summary":"Get Dev Test Server details","description":"Get Dev Test Server details","operationId":"getDevTestServerConfigurationUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/MailServerInfo"}},"404":{"description":"Dev Test server is not configured.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["settings-controller"],"summary":"Configure Dev Test Server","description":"Configure Dev Test Server","operationId":"setDevTestServerConfigurationUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"config","description":"Dev Test Server Configuration","required":true,"schema":{"$ref":"#/definitions/DevTestServerConfiguration"}},{"name":"forceSave","in":"query","description":"Set this value to true to save the configuration forcefully even if the connectivity fails.","required":false,"type":"boolean"}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/MailServerInfo"}},"400":{"description":"Missing required parameters. Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"412":{"description":"Dev Test server configuration is not valid. Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/settings/devTestServer/VSEs":{"get":{"tags":["settings-controller"],"summary":"Get Dev Test Server details","description":"Get Dev Test Server details","operationId":"getDevTestServerVSEsUsingGET_1","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/MailServerInfo"}},"404":{"description":"Dev Test server is not configured.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/settings/devTestServer/VSEs/{VSEName}":{"get":{"tags":["settings-controller"],"summary":"Get Dev Test Server details","description":"Get Dev Test Server details","operationId":"getDevTestServerVSEsUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"VSEName","in":"path","description":"Name of the Dev Test VSE for which you want to use to get the details.","required":true,"type":"string"}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/MailServerInfo"}},"404":{"description":"Dev Test server is not configured.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/settings/mailserver":{"get":{"tags":["settings-controller"],"summary":"Get Mail Server details","description":"Get Mail Server details","operationId":"getMailServerConfigurationUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through

this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"skipTest","in":"query","description":"Set this value to false to skip the test of the
mail server configuration","required":false,"type":"boolean","default":false}],"responses":
{"200":{"description":"Success","schema":{"$ref":"#/definitions/MailServerInfo"}},"404":
{"description":"Mail server is not configured.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"post":
{"tags":["settings-controller"],"summary":"Configure Mail Server","description":"Configure Mail
Server","operationId":"setMailServerConfigurationUsingPOST","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login
interface to perform a user login using user credentials in the Basic HTTP authorization scheme.
The API responds with a security token, which is valid for 24 hours by default. Use the security
token in the Bearer HTTP authorization scheme to access any protected resource through this
API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"info","description":"Mail Configuration","required":true,"schema":{"$ref":"#/definitions/
MailServerInfo"}},{"name":"forceSave","in":"query","description":"Set this value to true to save the
configuration forcefully even if the connectivity fails.","required":false,"type":"boolean"}],"responses":
{"200":{"description":"Success","schema":{"$ref":"#/definitions/MailServerInfo"}},"400":
{"description":"Missing required parameters. Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"412":{"description":"Mail server configuration is not valid.
Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/
api/ca/v1/settings/security":{"get":{"tags":["settings-controller"],"summary":"Interface for getting
the security configuration details","description":"Use this interface to get the security configuration
details.","operationId":"getSecuritySettingsConfigurationUsingGET","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security
token in the Bearer HTTP authorization scheme to access any protected resource through this API on
behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"}],"responses":
{"200":{"description":"Success","schema":{"$ref":"#/definitions/SecuritySettingsConfiguration"}},"401":
{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the
resource.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server
Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}},"put":{"tags":["settings-controller"],"summary":"Interface to configure the security
configuration details","description":"Use this interface to configure the security configuration
details.","operationId":"updateSecuritySettingsConfigurationUsingPUT","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
security token in the Bearer HTTP authorization scheme to access any protected resource through
this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"securitySettingsConfiguration","description":"Contains the appropriate key-
value pair to update the existing security configuration. For more information about the parameters
used in this object, click Model in the Data Type column.","required":true,"schema":{"$ref":"#/
definitions/SecuritySettingsConfiguration"}}],"responses":{"200":{"description":"Success","schema":
{"$ref":"#/definitions/LDAPConfigurationResult"}},"400":{"description":"Bad Request - Request does
not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the
resource.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error -
Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/
api/ca/v1/settings/security/actions/mapDefaultExternalGroups":{"post":{"tags":["settings-
controller"],"summary":"Interface for mapping external (AD/LDAP) groups as default groups","description":"Use
this interface to map external (AD/LDAP) groups as default groups for the corresponding ADMIN and TESTER

user groups of a project. These AD/LDAP groups are then automatically defined as default AD/LDAP groups for any newly created project.","operationId":"mapDefaultExternalGroupsUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"defaultExternalGroupsMap","description":"Contains external (AD/LDAP) groups information that you want to configure as default external (AD/LDAP) groups. For more information about the parameters used in this object, click Model in the Data Type column.","required":true,"schema":{"$ref":"#/definitions/DefaultExternalGroupsMapDTO"}}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/LDAPConfigurationResult"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"412":{"description":"Precondition failed - Operation is not allowed on the current resource state.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/settings/security/actions/unMapDefaultExternalGroups":{"post":{"tags":["settings-controller"],"summary":"Interface for removing external (AD/LDAP) groups as default groups","description":"Use this interface to remove the mapping that defines external (AD/LDAP) groups as default groups.","operationId":"unmapDefaultExternalGroupsUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"defaultExternalGroupsMap","description":"Contains external (AD/LDAP) groups information that you want to configure as default external (AD/LDAP) groups. For more information about the parameters used in this object, click Model in the Data Type column.","required":true,"schema":{"$ref":"#/definitions/DefaultExternalGroupsMapDTO"}}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/LDAPConfigurationResult"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"412":{"description":"Precondition failed - Operation is not allowed on the current resource state.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/settings/security/authorities":{"get":{"tags":["settings-controller"],"summary":"Interface to get the list of AD/LDAP server configurations","description":"Use this interface to get the list of AD/LDAP server configurations.","operationId":"getLDAPConfigurationsUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"}],"responses":{"200":{"description":"Success","schema":{"type":"array","items":{"$ref":"#/definitions/LDAPServerProperties"}}}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the resource.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found

```
 - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}}}},"post":{"tags":["settings-controller"],"summary":"Interface for saving the AD/LDAP
 server configuration details","description":"Use this interface to save the AD/LDAP server configuration
 details.","operationId":"addLDAPServerConfigurationUsingPOST","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"ldapServerProperties","description":"Contains the AD/LDAP server configuration
 information (key-value pairs) that you want to save. For more information about the parameters
 used in this object, click Model in the Data Type column.","required":true,"schema":{"$ref":"#/
definitions/LDAPServerProperties"}}],"responses":{"200":{"description":"OK","schema":{"$ref":"#/
definitions/LDAPServerProperties"}},"201":{"description":"Created.","schema":{"$ref":"#/definitions/
LDAPServerProperties"}},"400":{"description":"Bad Request - Request does not have a valid format
 or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the
 resource.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - Resource
 already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}}}}},"/api/ca/v1/settings/security/authorities/{authorityName}/":
{"put":{"tags":["settings-controller"],"summary":"Interface for updating the AD/LDAP server
 configuration details","description":"Use this interface to update the AD/LDAP server configuration
 details.","operationId":"updateLDAPServerConfigurationUsingPUT","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"authorityName","in":"path","description":"Specifies the authority name of the AD/LDAP
 sever that you want to update. Supported value is 'Default'","required":true,"type":"string"},
{"in":"body","name":"ldapServerProperties","description":"Contains the AD/LDAP server configuration
 information (key-value pairs) that you want to update. For more information about the parameters
 used in this object, click Model in the Data Type column.","required":true,"schema":{"$ref":"#/
definitions/LDAPServerProperties"}}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/LDAPServerProperties"}},"400":{"description":"Bad Request - Request does
 not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have
 permissions to access the resource.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/settings/security/authorities/{authorityName}/
validate":{"post":{"tags":["settings-controller"],"summary":"Interface to validate the AD/
LDAP server configuration","description":"Use this interface to validate the AD/LDAP server
 configuration.","operationId":"testLdapServerConfigurationUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"authorityName","in":"path","description":"Specifies the authority name of the AD/LDAP
```

sever that you want to use to validate the AD/LDAP server configuration. Supported value is 'Default'","required":true,"type":"string"},{"in":"body","name":"config","description":"Contains the AD/LDAP server configuration information (key-value pairs) that you want to validate. For more information about the parameters used in this object, click Model in the Data Type column.","required":true,"schema":{"$ref":"#/definitions/LDAPServerProperties"}}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/LDAPServerProperties"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"412":{"description":"Precondition Failed - AD/LDAP server configuration is not valid.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/user/actions/getUserInfo":{"post":{"tags":["auth-controller"],"summary":"Interface to get user details for a given user name","description":"Use this interface to get user details for a given user name.","operationId":"getUserInfoUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"userName","in":"query","description":"Name of the user that you want to fetch the user details.","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/UserDetails"}},"404":{"description":"Not Found - User with the specified information not found.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/user/actions/setProfile":{"post":{"tags":["auth-controller"],"summary":"Interface to set the users target and source profile","description":"Use this interface to set a users source and target profile names.","operationId":"setProfileUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"request","description":"Object identifying the target and source profiles","required":true,"schema":{"$ref":"#/definitions/SetProfileRequest"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/UserDetails"}},"404":{"description":"Not Found - User or Profile with the specified information not found.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/user/changePassword":{"post":{"tags":["auth-controller"],"summary":"Interface for validating old password and saving the new password","description":"Use this interface for validating the old password and saving the new password.","operationId":"validateOldPwdAndSaveNewPwdUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"password","in":"query","description":"New User credentials in Base 64 Encoded format ","required":true,"type":"string"},{"name":"oldPassword","in":"query","description":"Old User credentials in Base 64 Encoded format ","required":true,"type":"string"}],"responses":{"200":{"description":"Success","schema":{"type":"object"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/users":{"get":{"tags":["security-controller"],"summary":"Interface for getting list of users","description":"Use this interface to get the list of users.","operationId":"getUsersUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /

user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"page","in":"query","description":"Page number which you want to retrieve in a paginated users result. Default value is 0.","required":false,"type":"integer","format":"int32"},{"name":"size","in":"query","description":"Page size of each page with which you want to retrieve paginated users result. Default value is 15.","required":false,"type":"integer","format":"int32"},{"name":"sortDir","in":"query","description":"Sorting order with which you want the sort the paginated users result. Valid values are ASC and DESC.","required":false,"type":"string"},{"name":"sortField","in":"query","description":"Field on which you want to apply the sorting for the paginated users result.","required":false,"type":"string"},{"name":"searchText","in":"query","description":"Search text you want to use to search on generator name and description to get paginated user result.","required":false,"type":"string"},{"name":"projectId","in":"query","description":"Users associated to this project id will be retrieved.","required":false,"type":"integer","format":"int64"},{"name":"securityFunctions","in":"query","description":"List of Security Functions. Users having these security functions will be retrieved. An empty list will return all the users.","required":false,"type":"array","items":{"type":"string"},"collectionFormat":"multi"},{"name":"includeAdmins","in":"query","description":"Flag to indicate if admin users should be included in queries including security functions searches","required":false,"type":"boolean"}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/PaginatedUserDTO"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["security-controller"],"summary":"Interface for creating new users","description":"Use this interface to create new user.","operationId":"createUserUsingPOST","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"in":"body","name":"user","description":"New user to create.","required":true,"schema":{"$ref":"#/definitions/UserDTO"}},{"name":"host","in":"query","description":"host of TDM installation.","required":false,"type":"string"}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/UserDTO"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":{"description":"Conflict - User with user name already exists.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/users/{userId}":{"get":{"tags":["security-controller"],"summary":"Interface for getting Details of user","description":"Use this interface to get the Details of a User.","operationId":"getUserUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"userId","in":"path","description":"ID of the user you want to get.","required":true,"type":"integer","format":"int64"}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/definitions/UserDTO"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/

definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included
 in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":{"tags":["security-
controller"],"summary":"Interface for editing new users","description":"Use this interface to edit
 user.","operationId":"editUserUsingPUT","consumes":["application/json"],"produces":["*/*"],"parameters":
[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user
 login using user credentials in the Basic HTTP authorization scheme. The API responds with a security
 token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization
 scheme to access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"userId","in":"path","description":"Id of the user to be
 updated.","required":true,"type":"integer","format":"int64"},{"in":"body","name":"user","description":"user
 to be updated.","required":false,"schema":{"$ref":"#/definitions/UserDTO"}}],"responses":{"200":
{"description":"Success","schema":{"$ref":"#/definitions/UserDTO"}},"400":{"description":"Bad
 Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Server authentication failed.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific reason is included in the
 error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"405":{"description":"Method
 Not Allowed - Not Allowed to edit Administrator.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the
 error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["security-
controller"],"summary":"Interface for Deleting user","description":"Use this interface to delete a
 user.","operationId":"deleteUserUsingDELETE","consumes":["application/json"],"produces":["*/*"],"parameters":
[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user
 login using user credentials in the Basic HTTP authorization scheme. The API responds with a security
 token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization
 scheme to access any protected resource through this API on behalf of the user. For Example: Bearer
 {{token}}","required":true,"type":"string"},{"name":"userId","in":"path","description":"ID of the
 user you want to delete.","required":true,"type":"integer","format":"int64"}],"responses":{"200":
{"description":"Success","schema":{"$ref":"#/definitions/UserDTO"}},"400":{"description":"Bad Request -
 Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"405":{"description":"Method Not Allowed - Not Allowed to delete
 Administrator.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error
 - Specific reason is included in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/
api/ca/v1/users/{userId}/actions/addToGroups":{"post":{"tags":["security-controller"],"summary":"Interface
 for adding a user as a member in multiple groups","description":"Use this interface to add user as a member
 in multiple groups.","operationId":"addUserToGroupsUsingPOST","consumes":["application/json"],"produces":
["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface
 to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"userId","in":"path","description":"ID of the
 user .","required":true,"type":"integer","format":"int64"},{"in":"body","name":"groups","description":"List
 of groups","required":true,"schema":{"type":"array","items":{"$ref":"#/definitions/GroupDTO"}}}],"responses":
{"200":{"description":"Success","schema":{"$ref":"#/definitions/GroupDTO"}},"400":{"description":"Bad Request
 - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":
{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included
 in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/users/{userId}/
actions/getGroups":{"get":{"tags":["security-controller"],"summary":"Interface for getting Groups
 of an User","description":"Use this interface to get the list of groups that the user is a part
 of.","operationId":"getGroupsByUserUsingGET","consumes":["application/json"],"produces":["*/*"],"parameters":

[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a
user login using user credentials in the Basic HTTP authorization scheme. The API responds with a
security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
authorization scheme to access any protected resource through this API on behalf of the user. For
Example: Bearer {{token}}","required":true,"type":"string"},{"name":"userId","in":"path","description":"ID
of the user you want to get the Groups .","required":true,"type":"integer","format":"int64"},
{"name":"page","in":"query","description":"Page number which you want to retrieve in a paginated
data groups result. Default value is 0.","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"Page size of each page with which you want to retrieve
paginated groups result. Default value is 15.","required":false,"type":"integer","format":"int32"},
{"name":"sortDir","in":"query","description":"Sorting order with which you want the sort the
paginated groups result. Valid values are ASC and DESC.","required":false,"type":"string"},
{"name":"sortField","in":"query","description":"Field on which you want to apply the sorting for the paginated
groups result.","required":false,"type":"string"},{"name":"searchText","in":"query","description":"Search
text you want to use to search on generator name and description to get paginated groups
result.","required":false,"type":"string"},{"name":"userNotInGroups","in":"query","description":"Set
this flag to true to get the list of groups the user does not belong to. Defaults to
false.","required":false,"type":"boolean"}],"responses":{"200":{"description":"Success","schema":
{"$ref":"#/definitions/PaginatedGroupDTO"}},"400":{"description":"Bad Request - Specific reason is included
in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server
authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found -
Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message","schema":
{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/users/{userId}/actions/removeFromGroups":
{"post":{"tags":["security-controller"],"summary":"Interface for removing a user's membership in
multiple groups","description":"Use this interface to remove the user's membership in multiple
groups.","operationId":"deleteUserFromGroupsUsingPOST","consumes":["application/json"],"produces":["*/
*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to
perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
with a security token, which is valid for 24 hours by default. Use the security token in the Bearer
HTTP authorization scheme to access any protected resource through this API on behalf of the user. For
Example: Bearer {{token}}","required":true,"type":"string"},{"name":"userId","in":"path","description":"ID
of the user you want to get the Groups .","required":true,"type":"integer","format":"int64"},
{"in":"body","name":"groups","description":"List of groups","required":true,"schema":{"type":"array","items":
{"$ref":"#/definitions/GroupDTO"}}}],"responses":{"200":{"description":"Success","schema":{"$ref":"#/
definitions/GroupDTO"}},"400":{"description":"Bad Request - Specific reason is included in the error
message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server authentication
failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Specific
reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message","schema":{"$ref":"#/
definitions/ErrorResponse"}}}}},"/api/ca/v1/users/{userId}/actions/resetPassword":{"post":{"tags":["security-
controller"],"summary":"Interface for Generating Emails to Reset the Password","description":"Use this
interface to Generate Emails to Reset the Password.","operationId":"resetPasswordUsingPOST","consumes":
["application/json"],"produces":["*/*"],"parameters":[{"name":"Authorization","in":"header","description":"Use
the /user/login interface to perform a user login using user credentials in the Basic
HTTP authorization scheme. The API responds with a security token, which is valid for
24 hours by default. Use the security token in the Bearer HTTP authorization scheme to
access any protected resource through this API on behalf of the user. For Example: Bearer
{{token}}","required":true,"type":"string"},{"name":"userId","in":"path","description":"ID of the user to be
updated.","required":true,"type":"integer","format":"int64"},{"name":"host","in":"query","description":"host
of TDM installation to be included in password reset link.","required":false,"type":"string"}],"responses":
{"200":{"description":"Success","schema":{"$ref":"#/definitions/UserDTO"}},"400":{"description":"Bad Request
- Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":

{"description":"Server authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included
 in the error message","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/variables":{"get":
{"tags":["variable-controller"],"summary":"Interface to get all the variables at Repository level. Supports
 paginated response with filtering by search token.","description":"Use this interface to get all the
 variables at Repository level. Supports paginated response with filtering by search token which are
 optional.","operationId":"getRepositoryVariablesUsingGET","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"page","in":"query","description":"Page number to retrieve in a paginated variables
 result. Indexed with 0. Optional.","required":false,"type":"integer","format":"int32"},
{"name":"size","in":"query","description":"Page size of each page with
 which you want to retrieve in a paginated variables result. Default value
 is 20. Optional.","required":false,"type":"integer","format":"int32"},
{"name":"searchText","in":"query","description":"Search text you want to use to
 search on variable name, description and default value to get list of variable.
 Optional.","required":false,"type":"string"}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/PaginatedVariableBean"}},"400":{"description":"Bad Request - Request does
 not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/
ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":
{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have
 permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":
{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":
{"description":"Internal Server Error - Specific reason is included in the error message.","schema":
{"$ref":"#/definitions/ErrorResponse"}}}},"post":{"tags":["variable-controller"],"summary":"Interface
 to create new variable in repository","description":"Use this interface to create a new variable in
 repository.","operationId":"createRepositoryVariableUsingPOST","consumes":["application/json"],"produces":
["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /
user/login interface to perform a user login using user credentials in the Basic HTTP authorization
 scheme. The API responds with a security token, which is valid for 24 hours by default. Use the
 security token in the Bearer HTTP authorization scheme to access any protected resource through
 this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"in":"body","name":"variableInfo","description":"Request body for creating a variable in a project.
\n Mandatory parameters are: \n name: Specify variable name, accepts strings; \ndescription:
 Specify variable description, accepts strings; \n defaultValue: Specify default value for the
 variable, accepts strings; \nresolvePriorToPublish: Specify if variable should be resolved prior to
 publishing, accepts true or false","required":true,"schema":{"$ref":"#/definitions/VariableBean"}},
{"name":"validate","in":"query","description":"Set this parameter to true to validate the expressions used
 in the variable","required":false,"type":"boolean"}],"responses":{"200":{"description":"OK","schema":
{"$ref":"#/definitions/VariableBean"}},"201":{"description":"Created.","schema":{"$ref":"#/definitions/
VariableBean"}},"400":{"description":"Bad Request - Specific reason is included in the error
 message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid
 or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden
 - User does not have permissions","schema":{"$ref":"#/definitions/ErrorResponse"}},"409":
{"description":"Conflict - A Variable with the specified name already exists.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is
 included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/ca/v1/
variables/{variableName}":{"get":{"tags":["variable-controller"],"summary":"Interface to get details
 of a variable in repository.","description":"Use this interface to get details of a variable in

repository.","operationId":"getRepositoryVariableDetailsUsingGET","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"variableName","in":"path","description":"Name of the variable whose details have to be retrieved.","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/VariableBean"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"put":{"tags":["variable-controller"],"summary":"Interface to update details of a variable in repository.","description":"Use this interface to update details of a variable in repository.","operationId":"updateRepositoryVariableDetailsUsingPUT","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"variableName","in":"path","description":"Name of the variable whose details have to be updated.","required":true,"type":"string"},{"name":"validate","in":"query","description":"Set this parameter to true to validate the expressions used in the variable","required":false,"type":"boolean"},{"in":"body","name":"variableInfo","description":"Request body for creating a variable in a project. \n Mandatory parameters are: \n name: Specify variable name, accepts strings; \ndescription: Specify variable description, accepts strings; \n defaultValue: Specify default value for the variable, accepts strings; \nresolvePriorToPublish: Specify if variable should be resolved prior to publishing, accepts true or false","required":false,"schema":{"$ref":"#/definitions/VariableBean"}}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/VariableBean"}},"400":{"description":"Bad Request - Request does not have a valid format or has missing required parameters.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Resource not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}},"delete":{"tags":["variable-controller"],"summary":"Interface to delete repository variables.","description":"Use this interface to delete variables in a repository.","operationId":"deleteVariableUsingDELETE","consumes":["application/json"],"produces":["application/json"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP authorization scheme to access any protected resource through this API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},{"name":"variableName","in":"path","description":"Name of the variable to be deleted.","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":{"type":"object"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Invalid or expired token.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - User does not have permissions to access the project.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - Variable not found.","schema":{"$ref":"#/definitions/

```
ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the
 error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/api/introspect":{"post":
{"tags":["token-introspection-controller"],"summary":"API Endpoint for user token verification
 and introspection based on RFC7662 specification","operationId":"introspectUsingPOST","consumes":
["application/x-www-form-urlencoded"],"produces":["application/json"],"parameters":
[{"name":"Authorization","in":"header","description":"Use the /user/login interface to
 perform a user login using user credentials in the Basic HTTP authorization scheme. The API
 responds with a security token, which is valid for 24 hours by default. Use the security
 token in the Bearer HTTP authorization scheme to access any protected resource through this
 API on behalf of the user. For Example: Bearer {{token}}","required":true,"type":"string"},
{"name":"properties","in":"formData","description":"properties","required":true,"items":
{"type":"object","additionalProperties":{"type":"string"}}}],"responses":{"200":
{"description":"Success.","schema":{"$ref":"#/definitions/IntrospectionResponse"}},"401":
{"description":"Unauthorised when user doesn't have privilege to do introspection"},"422":
{"description":"Invalid introspection request, like token is missing."}}}},"/user/forgotPassword":
{"post":{"tags":["auth-controller"],"summary":"Interface for Generating Emails when User
 forgot the password","description":"Use this interface to Generate Emails to Reset the
 Password.","operationId":"forgotPasswordUsingPOST","consumes":["application/json"],"produces":["*/
*"],"parameters":[{"name":"Authorization","in":"header","description":"Use the /user/login interface to
 perform a user login using user credentials in the Basic HTTP authorization scheme. The API responds
 with a security token, which is valid for 24 hours by default. Use the security token in the Bearer HTTP
 authorization scheme to access any protected resource through this API on behalf of the user. For Example:
 Bearer {{token}}","required":true,"type":"string"},{"name":"userName","in":"query","description":"User Name
 of the Existing User","required":true,"type":"string"},{"name":"host","in":"query","description":"portal
 endpoint of TDM installation.","required":false,"type":"string"}],"responses":{"200":
{"description":"Success","schema":{"type":"object"}},"400":{"description":"Bad Request - Specific reason is
 included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Server
 authentication failed.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal
 Server Error - Specific reason is included in the error message","schema":{"$ref":"#/definitions/
ErrorResponse"}}}}},"/user/login":{"post":{"tags":["auth-controller"],"summary":"Login with credentials
 and receive a security token","description":"Use this interface to perform a user login using user
 credentials in Basic HTTP authorization scheme.  The API will respond with a security token,  use the
 security token in Bearer HTTP authorization scheme to access any protected resource via this API on
 behalf of the user. ","operationId":"loginUsingPOST","consumes":["application/json"],"produces":["*/
*"],"parameters":[{"name":"syncLogin","in":"query","description":"Set it to true if you want to wait
 for TDoD login. This may be necessary for automation scripts.","required":false,"type":"boolean"},
{"name":"Authorization","in":"header","description":"User credentials in Basic HTTP authorization
 scheme.","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":
{"$ref":"#/definitions/UserDetails"}},"400":{"description":"Bad Request - Specific reason is included in
 the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized -
 Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":
{"description":"Forbidden - Specific reason is included in the error message.","schema":{"$ref":"#/
definitions/ErrorResponse"}},"404":{"description":"Not Found - User with the specified information not
 found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error -
 Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/
user/loginFromService":{"post":{"tags":["auth-controller"],"summary":"Login from TDM Service with
 user credentials and receive a security token","description":"Use this interface to perform a
 login from TDM Service using credentials in Basic HTTP authorization scheme.  The API will respond
 with a security token,  use the security token in Bearer HTTP authorization scheme to access any
 protected resource via this API on behalf of the user. TDM Service will provide its session id which
 will get linked to the token.","operationId":"loginFromServiceUsingPOST","consumes":["application/
json"],"produces":["*/*"],"parameters":[{"in":"body","name":"sessionId","description":"Session id of TDM
 Service","required":true,"schema":{"type":"string"}},
```

{"name":"Authorization","in":"header","description":"User credentials in Basic HTTP authorization scheme.","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/UserDetails"}},"400":{"description":"Bad Request - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"401":{"description":"Unauthorized - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"403":{"description":"Forbidden - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}},"404":{"description":"Not Found - User with the specified information not found.","schema":{"$ref":"#/definitions/ErrorResponse"}},"500":{"description":"Internal Server Error - Specific reason is included in the error message.","schema":{"$ref":"#/definitions/ErrorResponse"}}}}},"/user/logout":{"put":{"tags":["auth-controller"],"summary":"Interface to log out or invalidate user session","description":"Use this interface to log out or invalidate user session.","operationId":"logoutUsingPUT","consumes":["application/json"],"produces":["*/*"],"parameters":[{"in":"body","name":"tokenTobeInvalidated","description":"tokenTobeInvalidated","required":true,"schema":{"type":"string"}},
{"name":"Authorization","in":"header","description":"Authorization","required":true,"type":"string"}],"responses":{"200":{"description":"Success.","schema":{"$ref":"#/definitions/UserDetails"}}}}}},"definitions":{"AccessBean":{"type":"object","properties":{"accessFunctions":{"type":"array","items":{"type":"string"}},"project":{"type":"string"}}},"AuditLog":{"type":"object","properties":{"description":{"type":"string","description":"Description of action or event being audited"},"id":{"type":"integer","format":"int64","description":"Unique identifier of audit log item"},"link_id":{"type":"integer","format":"int64","description":"Link identifier allow audit logs to be linked together"},"origin":{"type":"string","description":"Origin of action or event being audited"},"proj_id":{"type":"integer","format":"int64","description":"Identifier of project under which action/event occurred"},"proj_version_id":{"type":"integer","format":"int64","description":"Identifier of project version under which action/event occurred"},"status":{"type":"string","description":"Status of action or event"},"timestamp":{"type":"string","description":"Date and time of action/event"},"type":{"type":"string","description":"Type of action or event being audited"},"user_name":{"type":"string","description":"User who initiated the action/event being logged"}}},"BulkReservation":{"type":"object","required":["environmentName","primaryKeys","userName"],"properties":{"environmentName":{"type":"string","description":"Environment Name ,Currently Schema Name"},"primaryKeys":{"type":"array","description":"List of Primary Keys","items":{"$ref":"#/definitions/MultiValuedColumnInput"}},"reportKeys":{"type":"array","description":"List of Report Keys","items":{"$ref":"#/definitions/ColumnInput"}},"reservationType":{"type":"string","description":"Type of the reservation,Optional Default LK"},"testMartName":{"type":"string","description":"Table Name of the Test Mart in the Environment,Default value is MINI_MART"},"userName":{"type":"string","description":"User name of the reservation"}}},"ColumnInput":{"type":"object","properties":{"columnName":{"type":"string","description":"Name of the column"},"columnValue":{"type":"string","description":"value of the column"}}},"DBSeedlistSettings":{"type":"object","required":["connectionProfile","enabled","indexColumn","nameColumn","tableName","valueColumn"],"properties":{"connectionProfile":{"type":"string","description":"The connection profile pointing to the desired Database."},"enabled":{"type":"string","description":"Use DB seedlists"},"indexColumn":{"type":"string","description":"The seedlist index column."},"nameColumn":{"type":"string","description":"The seedlist name column."},"tableName":{"type":"string","description":"The desired table that holds the seedlist."},"valueColumn":{"type":"array","description":"The seedlist value column.","items":{"type":"string"}}}},"DefaultExternalGroupDTO":{"type":"object","required":["authorityName","distinguishedName","name"],"properties":{"authorityName":{"type":"string","description":"Name of the external group authority"},"distinguishedName":{"type":"string","description":"Distinguished name of the external group authority"},"name":{"type":"string","description":"Name of the external group"}}},"DefaultExternalGroupsMapDTO":{"type":"object","required":["adminGroups","testerGroups"],"properties":{"adminGroups":{"type":"array","description":"List of external group details to configure as default admin groups","items":{"$ref":"#/definitions/DefaultExternalGroupDTO"}},"testerGroups":{"type":"array","description":"List of external group details to configure as default tester

```
 groups","items":{"$ref":"#/definitions/DefaultExternalGroupDTO"}}}},"DevTestServerConfiguration":
{"type":"object","properties":{"host":{"type":"string"},"password":{"type":"string"},"port":
{"type":"integer","format":"int32"},"protocol":{"type":"string"},"rrPairImportCount":
{"type":"integer","format":"int64"},"userName":{"type":"string"},"valid":{"type":"boolean"}}},"ErrorResponse":
{"type":"object","properties":{"errorCode":{"type":"string"},"errorDetail":
{"type":"string"},"errorMsg":{"type":"string"},"status":{"type":"integer","format":"int32"},"timestamp":
{"type":"string"}}},"EventSubscribeRequest":{"type":"object","properties":{"desc":
{"type":"string"},"endpoint":{"type":"string"},"event":{"type":"string"},"id":
{"type":"integer","format":"int64"},"priority":{"type":"integer","format":"int32"},"requestBodyTemplate":
{"type":"string"},"rollbackEndpoint":{"type":"string"},"rollbackRequestBodyTemplate":
{"type":"string"},"rollbackUrlParams":{"type":"string"},"rollbackVerb":{"type":"string"},"service":
{"type":"string"},"urlParams":{"type":"string"},"verb":{"type":"string"}}},"EventSubscribeResponse":
{"type":"object","properties":{"desc":{"type":"string"},"endpoint":{"type":"string"},"event":
{"type":"string"},"id":{"type":"integer","format":"int64"},"priority":
{"type":"integer","format":"int32"},"requestBodyTemplate":{"type":"string"},"rollbackEndpoint":
{"type":"string"},"rollbackRequestBodyTemplate":{"type":"string"},"rollbackUrlParams":
{"type":"string"},"rollbackVerb":{"type":"string"},"service":{"type":"string"},"urlParams":
{"type":"string"},"verb":{"type":"string"}}},"ExternalGroupDTO":{"type":"object","required":
["authorityName","distinguishedName","id","isAdmin","isTester","name","type"],"properties":
{"authorityName":{"type":"string","description":"Name of the External Group's
 Authority"},"distinguishedName":{"type":"string","description":"Distinguished Name of the
 External Group"},"id":{"type":"integer","format":"int64","description":"ID of the External
 group","readOnly":true},"isAdmin":{"type":"boolean","example":false,"description":"Is Group an
 Default Admin Group"},"isTester":{"type":"boolean","example":false,"description":"Is Group an
 Default Tester Group"},"name":{"type":"string","description":"Name of the External Group"},"type":
{"type":"string","description":"Type of the Authority","enum":["LDAP"]}}},"ExternalGroupMapDTO":
{"type":"object","required":["authorityName","distinguishedName","name"],"properties":
{"authorityName":{"type":"string","description":"Name of the External Group's Authority","enum":
["Default"]},"distinguishedName":{"type":"string","description":"Distinguished Name
 of the External Group"},"name":{"type":"string","description":"Name of the External
 Group"}}},"ExternalGroupUnmapDTO":{"type":"object","required":["distinguishedName","id"],"properties":
{"distinguishedName":{"type":"string","description":"Distinguished Name of the External
 Group"},"id":{"type":"integer","format":"int64","description":"ID of the External group"}}},"File":
{"type":"object","properties":{"absolute":{"type":"boolean"},"absoluteFile":{"$ref":"#/definitions/
File"},"absolutePath":{"type":"string"},"canonicalFile":{"$ref":"#/definitions/File"},"canonicalPath":
{"type":"string"},"directory":{"type":"boolean"},"file":{"type":"boolean"},"freeSpace":
{"type":"integer","format":"int64"},"hidden":{"type":"boolean"},"name":{"type":"string"},"parent":
{"type":"string"},"parentFile":{"$ref":"#/definitions/File"},"path":{"type":"string"},"totalSpace":
{"type":"integer","format":"int64"},"usableSpace":{"type":"integer","format":"int64"}}},"GroupAttributes":
{"type":"object","properties":{"groupIdAttribute":{"type":"string","example":"cn","description":"The
 attribute field to use when loading the group's name. Applicable only to 'AD/LDAP' authentication
 source"},"groupMemberAttribute":{"type":"string","example":"member","description":"The attribute
 field to use when loading the group members from the group. Applicable only to 'AD/LDAP'
 authentication source"},"groupObjectClass":{"type":"string","example":"group","description":"The
 name of the AD/LDAP group object class to use when loading the groups. Applicable only to 'AD/LDAP'
 authentication source"},"groupOrganization":{"type":"string","example":"cn=groups","description":"The
 organization of the AD/LDAP group object class to use when loading the groups. Applicable
 only to 'AD/LDAP' authentication source"}}},"GroupDTO":{"type":"object","properties":
{"adGroup":{"type":"string","description":"Name of the AD group associated with this
 group"},"adminGroup":{"type":"boolean"},"description":{"type":"string","description":"Description
 of the Group"},"groupId":{"type":"integer","format":"int64","description":"ID of the
 group","readOnly":true},"groupName":{"type":"string","description":"Name of the Group"},"isAdminGroup":
{"type":"boolean","example":false,"description":"Flage to identify whether this group is and admin group
```

or not"},"projectId":{"type":"integer","format":"int64","description":"Id of the project associated with group"},"securityFunctions":{"type":"object","description":"Map of security functions available and their flags whether they are enabled or not","additionalProperties":{"type":"boolean"}}}},"InputStream":{"type":"object"},"InputStreamResource":{"type":"object","properties":{"description":{"type":"string"},"file":{"$ref":"#/definitions/File"},"filename":{"type":"string"},"inputStream":{"$ref":"#/definitions/InputStream"},"open":{"type":"boolean"},"readable":{"type":"boolean"},"uri":{"$ref":"#/definitions/URI"},"url":{"$ref":"#/definitions/URL"}}},"IntegrationAccount":{"type":"object","properties":{"name":{"type":"string"},"password":{"type":"string"},"setting":{"type":"integer","format":"int32"},"type":{"type":"string"},"url":{"type":"string"},"user":{"type":"string"}}},"IntrospectionResponse":{"type":"object","properties":{"accessFunctions":{"type":"object","additionalProperties":{"type":"array","items":{"type":"string"}}},"active":{"type":"boolean"},"admin":{"type":"boolean"},"exp":{"type":"integer","format":"int64"},"iat":{"type":"integer","format":"int64"},"integrator":{"type":"boolean"},"sessionId":{"type":"string"},"username":{"type":"string"}}},"LDAPConfigurationResult":{"type":"object","properties":{"message":{"type":"string","description":"Success message when API call is successful, error otherwise"}}},"LDAPServerProperties":{"type":"object","required":["authorityName","baseDN","hostName","password","port","userDN"],"properties":{"authorityName":{"type":"string","description":"Logical name for the AD/LDAP identity provider","enum":["Default"]},"baseDN":{"type":"string","example":"dc=example,dc=com or cn=users,dc=example,dc=com","description":"The base distinguished name to use for searching users and groups in AD/LDAP server. Applicable only to 'AD/LDAP' authentication source"},"customUserFilter":{"type":"string","description":"Additional LDAP filter for filtering the searched users."},"globalTDMGroup":{"type":"string","example":"GT_DM_ACCESS","description":"The controlling Active Directory(AD) group name specified in the CA TDM license. Applicable only to 'AD/LDAP' authentication source"},"groupAttributes":{"description":"The groupAttributes object containing the key value pairs of the group attributes. Applicable only to 'AD/LDAP' authentication source","$ref":"#/definitions/GroupAttributes"},"hostName":{"type":"string","example":"ldap.example.com","description":"The hostname of the AD/LDAP server. Applicable only to 'AD/LDAP' authentication source"},"ldapAdvanceConfiguration":{"description":"The advanceConfiguration object containing the key value pairs of the advance configuration. Applicable only to 'AD/LDAP' authentication source","$ref":"#/definitions/LdapAdvanceConfiguration"},"password":{"type":"string","description":"The password of the user specified in userDN field. Applicable only to 'AD/LDAP' authentication source"},"port":{"type":"string","example":"389","description":"The port on which AD/LDAP server is listening. Applicable only to 'AD/LDAP' authentication source"},"tlsAttributes":{"description":"The tlsAttributes object containing the key value pairs of the tls attributes. Applicable only to 'AD/LDAP' authentication source","$ref":"#/definitions/TLSAttributes"},"userAttributes":{"description":"The userAttributes object containing the key value pairs of the user attributes. Applicable only to 'AD/LDAP' authentication source","$ref":"#/definitions/UserAttributes"},"userDN":{"type":"string","example":"cn=administrator,cn=users,dc=example,dc=com or user@domain.name","description":"The distinguished name of the user to use when connecting to the AD/LDAP server. Applicable only to 'AD/LDAP' authentication source"}}},"LdapAdvanceConfiguration":{"type":"object","properties":{"referralStrategy":{"type":"string","example":"FOLLOW","description":"The referral strategy to use in order to redirect the requests to other servers. The default value is 'FOLLOW'. Applicable only to 'AD/LDAP' authentication source","enum":["IGNORE","FOLLOW"]}}},"LdapUserDTO":{"type":"object","required":["distinguishedName","mail","name"],"properties":{"distinguishedName":{"type":"string","description":"Distinguished Name of the External User"},"mail":{"type":"string","description":"Email of the user"},"name":{"type":"string","description":"Name of the user"}}},"LdapUserGroup":{"type":"object","properties":{"name":{"type":"string","description":"Name"},"distinguishedName":{"type":"string","description":"Distinguished Name"},"email":{"type":"string","description":"Mail Id"}}},"MailServerInfo":{"type":"object","properties":{"enableSSL":{"type":"boolean"},"fromAddress":{"type":"string","description":"From Email address which is used to send the mails"},"hostName":{"type":"string","description":"Mail server host name/ip address. "},"password":{"type":"string","description":"Password of the user. Required if auth parameter is set to true."},"port":{"type":"integer","format":"int32","description":"Port number where the mail server is listening to"},"protocol":{"type":"string","description":"Messaging protocol to be used: smtp"},"requireAuthentication":{"type":"boolean","example":false,"description":"Specifies

```
 whether authentication to be carried while connecting to the mail server. If this parameter is set
 to true, you need to set user name and password."},"userName":{"type":"string","description":"User
 name for authenticating with the server. Required if auth parameter is set to true."},"valid":
{"type":"boolean","example":false,"description":"Specifies whether this configuration
 is valid or not."}}},"Map«string,object»":{"type":"object","additionalProperties":
{"type":"object"}},"MultiValuedColumnInput":{"type":"object","properties":{"columnName":
{"type":"string","description":"Name of the column"},"columnValues":{"type":"array","description":"values
 of the column","items":{"type":"string"}}}},"PaginatedAuditLogsResult":{"type":"object","properties":
{"elements":{"type":"array","description":"List of actual audit logs retrieved","items":{"$ref":"#/
definitions/AuditLog"}},"numberOfElements":{"type":"integer","format":"int64","description":"Number of audit
 logs retrieved"},"totalElements":{"type":"integer","format":"int64","description":"Total number of filtered
 audit logs available"},"totalPages":{"type":"integer","format":"int64"}}},"PaginatedExternalGroupDTO":
{"type":"object","properties":{"groups":{"type":"array","items":{"$ref":"#/definitions/
ExternalGroupDTO"}},"numberOfGroups":{"type":"integer","format":"int32"},"totalNumberOfGroups":
{"type":"integer","format":"int64"}}},"PaginatedGroupDTO":{"type":"object","properties":
{"groups":{"type":"array","items":{"$ref":"#/definitions/GroupDTO"}},"numberOfGroups":
{"type":"integer","format":"int32"},"totalNumberOfGroups":
{"type":"integer","format":"int64"}}},"PaginatedLdapUserGroup":{"type":"object","properties":{"count":
{"type":"integer","format":"int32"},"totalCount":{"type":"integer","format":"int64"},"values":
{"type":"array","items":{"$ref":"#/definitions/LdapUserGroup"}}}},"PaginatedReservationsDTO":
{"type":"object","properties":{"numberOfReservations":{"type":"integer","format":"int32","description":"Number
 of PaginatedResultsDTO.java in current page"},"reservations":{"type":"array","description":"List
 of Reservations","items":{"$ref":"#/definitions/Map«string,object»"}},"totalReservations":
{"type":"integer","format":"int32","description":"Total number of reservations"}}},"PaginatedResultsDTO":
{"type":"object","properties":{"numberOfResults":{"type":"integer","format":"int32","description":"Number
 of Results in current page"},"results":{"type":"array","description":"List of
 Results","items":{"$ref":"#/definitions/Map«string,object»"}},"totalNumberOfResults":
{"type":"integer","format":"int32","description":"Total number of results"}}},"PaginatedUserDTO":
{"type":"object","properties":{"numberOfUsers":{"type":"integer","format":"int32"},"totalNumberOfUsers":
{"type":"integer","format":"int64"},"users":{"type":"array","items":{"$ref":"#/
definitions/UserDTO"}}}},"PaginatedVariableBean":{"type":"object","properties":{"elements":
{"type":"array","items":{"$ref":"#/definitions/VariableBean"}},"numberOfElements":
{"type":"integer","format":"int32"},"totalNumberOfElements":
{"type":"integer","format":"int64"}}},"ParameterInput":{"type":"object","properties":{"parameterDataType":
{"type":"string","description":"datatype of the named parameter, currently supports NUMERIC,STRING and
 DATETIME"},"parameterName":{"type":"string","description":"Name of the named parameter"},"parameterValue":
{"type":"string","description":"named parameter value"}}},"PrimaryKeysError":{"type":"object","properties":
{"errorCode":{"type":"string"},"errorDetail":{"type":"string"},"errorMsg":{"type":"string"},"primaryKeys":
{"type":"array","description":"List of Primary Keys","items":{"$ref":"#/definitions/ColumnInput"}},"status":
{"type":"integer","format":"int32"},"timestamp":{"type":"string"}}},"Property":{"type":"object","properties":
{"propertyName":{"type":"string"},"propertyValue":{"type":"string"}}},"Reservation":
{"type":"object","required":["environmentName","primaryKeys","userName"],"properties":
{"environmentName":{"type":"string","description":"Environment Name ,Currently Schema
 Name"},"primaryKeys":{"type":"array","description":"List of Primary Keys","items":{"$ref":"#/
definitions/ColumnInput"}},"reportKeys":{"type":"array","description":"List of Report Keys","items":
{"$ref":"#/definitions/ColumnInput"}},"reservationType":{"type":"string","description":"Type
 of the reservation,Optional Default LK"},"testMartName":{"type":"string","description":"Table
 Name of the Test Mart in the Environment,Default value is MINI_MART"},"userName":
{"type":"string","description":"User name of the reservation"}}},"ResponseEntity":
{"type":"object","properties":{"body":{"type":"object"},"statusCode":{"type":"string","enum":
["100 CONTINUE","101 SWITCHING_PROTOCOLS","102 PROCESSING","103 CHECKPOINT","200 OK","201
 CREATED","202 ACCEPTED","203 NON_AUTHORITATIVE_INFORMATION","204 NO_CONTENT","205 RESET_CONTENT","206
 PARTIAL_CONTENT","207 MULTI_STATUS","208 ALREADY_REPORTED","226 IM_USED","300 MULTIPLE_CHOICES","301
```

```
 MOVED_PERMANENTLY","302 FOUND","302 MOVED_TEMPORARILY","303 SEE_OTHER","304 NOT_MODIFIED","305
 USE_PROXY","307 TEMPORARY_REDIRECT","308 PERMANENT_REDIRECT","400 BAD_REQUEST","401 UNAUTHORIZED","402
 PAYMENT_REQUIRED","403 FORBIDDEN","404 NOT_FOUND","405 METHOD_NOT_ALLOWED","406 NOT_ACCEPTABLE","407
 PROXY_AUTHENTICATION_REQUIRED","408 REQUEST_TIMEOUT","409 CONFLICT","410 GONE","411 LENGTH_REQUIRED","412
 PRECONDITION_FAILED","413 PAYLOAD_TOO_LARGE","413 REQUEST_ENTITY_TOO_LARGE","414 URI_TOO_LONG","414
 REQUEST_URI_TOO_LONG","415 UNSUPPORTED_MEDIA_TYPE","416 REQUESTED_RANGE_NOT_SATISFIABLE","417
 EXPECTATION_FAILED","418 I_AM_A_TEAPOT","419 INSUFFICIENT_SPACE_ON_RESOURCE","420
 METHOD_FAILURE","421 DESTINATION_LOCKED","422 UNPROCESSABLE_ENTITY","423 LOCKED","424
 FAILED_DEPENDENCY","426 UPGRADE_REQUIRED","428 PRECONDITION_REQUIRED","429 TOO_MANY_REQUESTS","431
 REQUEST_HEADER_FIELDS_TOO_LARGE","451 UNAVAILABLE_FOR_LEGAL_REASONS","500 INTERNAL_SERVER_ERROR","501
 NOT_IMPLEMENTED","502 BAD_GATEWAY","503 SERVICE_UNAVAILABLE","504 GATEWAY_TIMEOUT","505
 HTTP_VERSION_NOT_SUPPORTED","506 VARIANT_ALSO_NEGOTIATES","507 INSUFFICIENT_STORAGE","508 LOOP_DETECTED","509
 BANDWIDTH_LIMIT_EXCEEDED","510 NOT_EXTENDED","511 NETWORK_AUTHENTICATION_REQUIRED"]},"statusCodeValue":
{"type":"integer","format":"int32"}}},"SearchDataInputs":{"type":"object","properties":{"parameterInputs":
{"type":"array","description":"List of Named Parameters","items":{"$ref":"#/definitions/
ParameterInput"}},"variableInputs":{"type":"array","description":"List of Variables","items":
{"$ref":"#/definitions/VariableInput"}}}},"SecuritySettingsConfiguration":{"type":"object","required":
["authenticationMode"],"properties":{"authenticationMode":{"type":"string","example":"AD/
LDAP","description":"The database to query for user authentication and authorization.The default value
 is 'Native TDM'","enum":["Native TDM","AD/LDAP"]}}},"SetProfileRequest":{"type":"object","properties":
{"sourceProfile":{"type":"string"},"targetProfile":{"type":"string"}}},"StringResponse":
{"type":"object","properties":{"response":{"type":"string"}}},"TLSAttributes":{"type":"object","properties":
{"useTLS":{"type":"boolean","example":false,"description":"The flag to enable SSL/TLS connections
 to the AD/LDAP server.The default value is 'false'.Applicable only to 'AD/LDAP' authentication
 source."}}},"URI":{"type":"object","properties":{"absolute":{"type":"boolean"},"authority":
{"type":"string"},"fragment":{"type":"string"},"host":{"type":"string"},"opaque":{"type":"boolean"},"path":
{"type":"string"},"port":{"type":"integer","format":"int32"},"query":{"type":"string"},"rawAuthority":
{"type":"string"},"rawFragment":{"type":"string"},"rawPath":{"type":"string"},"rawQuery":
{"type":"string"},"rawSchemeSpecificPart":{"type":"string"},"rawUserInfo":{"type":"string"},"scheme":
{"type":"string"},"schemeSpecificPart":{"type":"string"},"userInfo":{"type":"string"}}},"URL":
{"type":"object","properties":{"authority":{"type":"string"},"content":{"type":"object"},"defaultPort":
{"type":"integer","format":"int32"},"file":{"type":"string"},"host":{"type":"string"},"path":
{"type":"string"},"port":{"type":"integer","format":"int32"},"protocol":{"type":"string"},"query":
{"type":"string"},"ref":{"type":"string"},"userInfo":{"type":"string"}}},"UserAttributes":
{"type":"object","properties":{"userIdAttribute":{"type":"string","example":"cn","description":"The
 attribute field to use when loading the username. Applicable only to 'AD/LDAP' authentication
 source"},"userObjectClass":{"type":"string","example":"person","description":"The name of
 the AD/LDAP user object class to use when loading the users. Applicable only to 'AD/LDAP'
 authentication source"},"userOrganization":{"type":"string","example":"cn=users","description":"The
 organization of the AD/LDAP user object class to use when loading the users. Applicable
 only to 'AD/LDAP' authentication source"}}},"UserDTO":{"type":"object","required":
["distinguishedName"],"properties":{"authLink":{"type":"string","description":"AuthLink
 of the user"},"authorityName":{"type":"string","description":"Authority of the
 user"},"distinguishedName":{"type":"string","description":"Distinguished Name of the
 external user"},"email":{"type":"string","description":"Email of the user"},"extension":
{"type":"string","description":"Extension of the user"},"fullName":{"type":"string","description":"Full
 name of the user"},"location":{"type":"string","description":"Location of the user"},"userId":
{"type":"integer","format":"int64","description":"ID of the user","readOnly":true},"userName":
{"type":"string","description":"Name of the user"}}},"UserDetails":{"type":"object","properties":
{"accessPermissions":{"type":"array","items":{"$ref":"#/definitions/AccessBean"}},"authorityName":
{"type":"string"},"emailId":{"type":"string"},"id":{"type":"number"},"integrator":
{"type":"boolean"},"ldapAuthentication":{"type":"boolean"},"previewFeaturesEnabled":
{"type":"boolean"},"selfServiceEmailMandate":{"type":"boolean"},"sessionId":{"type":"string"},"sourceProfile":
```

```
{"type":"string"},"targetProfile":{"type":"string"},"token":{"type":"string"},"tokenExpiresAt":
{"type":"string","example":"yyyy-MM-dd'T'HH:mm:ssZ"},"tokenIssuedAt":{"type":"string","example":"yyyy-
MM-dd'T'HH:mm:ssZ"},"userDn":{"type":"string"},"userName":{"type":"string"}}},"VariableBean":
{"type":"object","required":["defaultValue","description","name","type"],"properties":
{"defaultValue":{"type":"string","description":"Default Value of the variable"},"description":
{"type":"string","description":"Description of the variable"},"displayType":
{"type":"string","description":"Display type","enum":
["TextBox","CheckBox","DropDownList","MultiSelectList","RadioButton","DatePicker"]},"helpMessage":
{"type":"string","description":"Help message"},"isDisplayOnly":
{"type":"boolean","example":false,"description":"Variable value is read-only at runtime"},"isOptional":
{"type":"boolean","example":false,"description":"Is Optional"},"listDefinition":
{"type":"string","description":"Definition for list values"},"name":{"type":"string","description":"Name
 of the variable"},"resolvePriorToPublish":{"type":"boolean","example":false,"description":"Resolve
 this variable prior to publish"},"scope":{"type":"string","description":"Scope of the
 variable","readOnly":true},"type":{"type":"string","description":"Type of the variable","enum":
["String","Number","Date","Boolean"]},"validation":{"type":"string","description":"Validation
 rule for the variable value"}}},"VariableInput":{"type":"object","properties":{"variableName":
{"type":"string","description":"Name of the variable,format supported for variable is %(variable_name)
 "},"variableValue":{"type":"string","description":"Value of the variable"}}}}}
```


REST RR Pair Format

The CA TDM Portal supports parsing and shredding of RR pair files that contain information in the form of HTTP headers and body. To support the content structure and format that these RR Pair files include, the CA TDM Portal accepts RR Pair files that have .txt extension. The CA TDM Portal accesses the .txt file, evaluates the included HTTP headers and body, and shreds the structure. The portal also establishes a relationship between HTTP headers and the corresponding body. You can perform other operations, for example, import and export on these RR pairs. You can also integrate these RR pairs with your virtual services in CA Service Virtualization.

A request .txt file constitutes the following sections; these sections are optional based on the type of the request (GET, POST, PUT, and DELETE):

- REST method and URI path details
- HTTP headers
- Payload request body (XML or JSON)

The general format of a REST call used in a request .txt file is as follows:

```
<METHOD><a space character><REST API path><space><HTTP-VERSION>

HEADERKEY1:HEADERVALUE

HEADERKEY2:HEADERVALUE

HEADERKEY3:HEADERVALUE

<BLANKLINE>

<PAYLOAD_BODY – COULD BE JSON OR XML>
```

An example of a request .txt file (for example, POST-JSON-req.txt) following the above format is as follows:

```
POST /api/sec/contacts/get-available-appointment-time-v1?key1=value1&key2=value2
 HTTP/1.1
accept: application/json
content-Type: application/json
Connection: Keep-Alive
User-Agent: LISA
Custom-Header: MyHeaderVal



{ "lisa-meta":{ "HTTP-Method":"POST /api/sec/contacts/get-available-appointment-
time-v1", "OLB_ACCESS_ID":"41472851", "content-type":"application/json",
 "Cookie":"SMSESSION=41472851" }, "meta":{ "lang":"en", "appVersion":"1.0.0.0",
 "osName":"ios", "osVersion":"6.0.0", "deviceModel":"iPad3", "deviceType":"iPad",
 "deviceAppToken": "xxxxxxxxxxx-TBD-xxxxxxxxxxxxx", "ipAddress":
 "0.0.0.0" }, "challengeRequest":{ "deviceFingerprint":"xxxxx-xxxxx-
xxxxx-fingerprint-to-be-generated-by-RSA-SDK-xxxxx-xxxxx-xxxxx-",
 "sessionCorrelationId":"XXXXXXXXXXXXXXXXXXXXXX" }, "request":{ "topicId":1 } }
```

Similarly, a response .txt file constitutes the following sections:

- HTTP header
- Body

The general format of a response .txt file is as follows:

```
<HTTP-VERSION><space><RESPONSE-STATUS-CODE>

HEADERKEY1:HEADERVALUE
HEADERKEY2:HEADERVALUE
HEADERKEY3:HEADERVALUE

<BLANKLINE>

<RESPONSE-BODY>
```

An example of a response .txt file (for example, POST-JSON-rsp.txt) is as follows; this response does not include headerkey:headervalue information:

```
HTTP/1.1 200



{ "result":{ "type":"OK", "sysmsg":"OK @ Fri Feb 15 10:43:01 PST 2013" }, "response":
{ "appointmentTimes":[ 1351234800000, 1351234801000, 1351234802000, 1351234803000,
 1351234804000, 1351234805000, 1351234806000, 1351234807000, 1351234808000,
 1351234809000 ] } }
```

Note: For more information about how to use RR pairs to prepare test data for non-relational data sources, see Prepare Test Data for Non-Relational Data Sources. While performing different operations for the RR Pair object type, you can use text RR pair files that contain data in the REST format. You can perform all the functions on these text RR pair files that you can perform on XML or JSON RR pair files.

Filter Options for Transformation Maps

The following list includes all the available filter options and transformation map conditions in the Datamaker UI.

Filter Name	Description	Condition
ALL	Identifies all the columns.	
CHECKED	Identifies all the checked columns.	(checked = 'Y')
VALIDATED	Identifies all the validated columns.	(validated = 'Y')
APPROVED	Identifies all the approved columns.	(approved = 'Y')
KEY	Identifies all the columns containing primary key records.	(tpd_pkey_count > 0) or (tpd_ukey_count > 0)
INDEXED	Identifies all the columns containing indexed records.	tid_index_count > 0
EMPTY	Identifies all the columns that are defined as empty.	(tcs_nullcount > 0) and (tcs_rowcount = 0)
UNIQUE	Identifies all the columns that contain unique values.	(tcs_distinct_count = tcs_rowcount) and (tcs_rowcount > 0)
GUID	Identifies all the columns that contain a GUID.	(upper (tcs_sample_analysis) like '%GUID %')
SEQUENTIAL	Identifies all the columns that contain a sequential numeric list.	(upper (tcs_sample_analysis) like '%SEQUENCE%')
DATE	Identifies all the columns that contain a date.	case (upper (left (tcd_datatype, 9)) when 'DATE","DATETIME","TIMESTAMP" then 1 else 0) = 1 or case (pos (tcd_format, "YY") when is > 0 then 1 else 0) = 1 or (pos (lower (tcd_column_name), "_dat") > 0 and pos (lower (tcd_column_name), "_dat") <> pos (lower (tcd_column_name), "_data"))
DOB	Identifies all the columns that contain a DOB.	((upper (tcs_sample_analysis) like '%DOB %') or (upper (tcd_column_name) like '%DAT %BIRTH%') or (upper (tcd_column_name) like '%DOB %'))
TAILING DATES	Identifies all the columns that contain values with tailing dates.	(upper (tcs_sample_analysis) like '%TAILING_DATES%')
NAME	Identifies all the columns that contain a name.	((upper (tcd_column_name) like '%NAM%') or (upper (tcd_column_name) like '%NM %')) + is_char_only

ADDRESS	Identifies all the columns that contain an address.	((upper (tcd_column_name) like '%ADD%') or (upper (tcd_column_name) like '%ADR %') or (upper (tcd_column_name) like '%CITY %') or (upper (tcd_column_name) like '%STREET%') or (upper (tcd_column_name) like '%STATE %') or (upper (tcd_column_name) like '%POST %CODE%') or (upper (tcd_column_name) like '%COUNT%Y') OR (upper (tcs_sample_analysis) like '%ADDRESS%') OR (upper (tcs_sample_analysis) like '%USZIP%') or (upper (tcs_sample_analysis) like '%UKPOSTCODE%') or (upper (tcs_sample_analysis) like '%POSTAL%')) + is_char_only
SSN	Identifies all the columns that contain US Social Security numbers.	((upper (tcd_column_name) like '%SS%') and (upper (tcd_column_name) not like '%ESS%')) or (upper (tcd_column_name) like '%SOCIAL%') OR (upper (tcs_sample_analysis) like '%US-SSN%')
EMAIL	Identifies all the columns that contain email values.	((upper (td_table_name) like '%EMAIL%') or (upper (tcd_column_name) like '%EMAIL') or (upper (td_table_name) like 'EMAIL%') OR (upper (tcs_sample_analysis) like '%EMAIL%'))
UK NINO	Identifies all the columns that contain UK NI numbers.	((upper (tcs_sample_analysis) like '%UK-NINO') or (upper (tcd_column_name) like '%NAT %INS%') or (upper (tcd_column_name) like '%NI %NO%')) + is_char_only
CREDIT CARD	Identifies all the columns that contain credit card numbers.	(upper (tcs_sample_analysis) like '%CREDITCARD%')
IBAN	Identifies all the columns that contain IBAN numbers.	(upper (tcs_sample_analysis) like '%IBAN %')

FINANCIAL DATA	Identifies all the columns that contain financial data.	(upper (tcd_column_name) like '%ACCOUNT%') or (upper (tcd_column_name) like '%SALARY%') or (upper (tcd_column_name) like '%REVENUE%') or (upper (tcd_column_name) like '%PROFIT%') or (upper (tcd_column_name) like '%SALES%') or (upper (tcd_column_name) like '%TRANSACTION%')
PHONE NUMBER	Identifies all the columns that contain phone numbers.	(upper (tcd_column_name) like '%PHONE %') or (upper(tcd_column_name) like '%NUMBER%') or (upper (tcs_sample_analysis) like '%PHONE%')
HIGH DISTINCT COUNT (CHARS)	Identifies all the columns with high distinct character data types.	(upper (tcs_sample_analysis) like '%HIGHDISTINCT_C%')
HIGH DISTINCT COUNT (NUMERIC)	Identifies all the columns with high distinct numeric data types.	(upper (tcs_sample_analysis) like '%HIGHDISTINCT_N%')
MIX OF ALPHANUMERIC	Identifies all the columns that contain alphanumeric data.	(upper (tcs_sample_analysis) like '%ALPHANUMERIC%')
MIXED CASE	Identifies all the columns that contain values in mixed case.	(upper (tcs_sample_analysis) like '%MIXEDCASE%')
CONTAINS SPACES	Identifies all the columns that contain spaces.	(upper (tcs_sample_analysis) like '%SPACES%')
CONTAINS SPECIAL CHARACTERS	Identifies all the columns that contain special characters.	(upper (tcs_sample_analysis) like '%SPECIALCHARS%')
NUMERIC CHARACTER	Identifies all the columns that contain numeric characters.	(upper (tcs_sample_analysis) like '%C-NUMERIC%')
FORMATTED NUMERIC DATA	Identifies all the columns that contain formatted numeric data.	(upper (tcs_sample_analysis) like '%NUM-PATTERN%')
TEXT	Identifies all the columns that contain text.	((upper (tcd_datatype) like '%CHAR%') and (tcd_precision > 254)) or (upper (tcd_datatype) like '%STRING%') or (upper (tcd_datatype) like '%TEXT%') or (upper (tcd_datatype) like '%CLOB%')
VALUES FROM SEED LIST	Identifies all the columns that contain values from seed lists.	<seedlist>
TRANSFORMED	Identifies all the columns that contain transformed values.	<transformed>
CUSTOM	Allows you to create a customized filter.	See Custom Filter Functions for Transformation Maps.

NOTE

In this table, is_char_only is short for the following expression:

and (NOT (case (upper (left (tcd_datatype, 9)) when "DATE","DATETIME","TIMESTAMP","NUMBER","NUMERIC","INTEGER" then 1 else 0) = 1 or case (pos (tcd_format, "YY") when is > 0 then 1 else 0) = 1))

Relevant Columns

checked	case when tfc_auth_stage >= 1 then 'Y' else 'N' end
validated	case when tfc_auth_stage >= 2 then 'Y' else 'N' end
approved	case when tfc_auth_stage >= 3 then 'Y' else 'N' end
td_table_name	coalesce (TFC_TABLE_NAME, TD_TABLE_NAME)
tcd_column_name	coalesce (TFC_COLUMN_NAME, TCD_COLUMN_NAME)
tcd_column_seq	
tcd_datatype	
tcd_precision	
tcd_scale	
tcd_nullable	CASE when tcd_nullable = '0' then 'N' else 'Y' end
TFC_ACTION	In gtrep_transformation_column table
TFC_DEFAULT	In gtrep_transformation_column table
TFC_FUNCTION	In gtrep_transformation_column table
tree_icon	case when tfd_fkey_count + tfd_rel_count > 0 then 'treeview16.gif' else null end
tfd_rel_count	(select count (rel_name) From gtrep_relationship join gtrep_rel_column on rc_rel_id = rel_id where rc_parent_column = tcd_column_name and rel_parent_table = td2.td_table_name and rel_proj_id in (select pv_proj_id from gtrep_project_version where pv_id = <projectVersion>))
tfd_fkey_count	(select count (tfd_fkey_name) From gtrep_table_fkey_def where tfd_ref_column_name = tcd_column_name and tfd_ref_table_name = td2.td_table_name and tfd_pv_id == <projectVersion>)
TFC_XREF	In gtrep_transformation_column table
TFC_XREF_IDENT	In gtrep_transformation_column table
TFC_KEEPNULLS	In gtrep_transformation_column table
TFC_LIST_COLNO	In gtrep_transformation_column table
TFC_OVERRIDE_LOOKUP	In gtrep_transformation_column table
TFC_UNIQUE_COLS	In gtrep_transformation_column table
TFC_NOTES	In gtrep_transformation_column table
TFC_PREFORMAT	In gtrep_transformation_column table
TCD_FORMAT	In gtrep_transformation_column table
where_clause_yes	case when coalesce (tfc_where_id, 0) > 0 then 'Y' else '' end
TFC_WHERE_SEQ	In gtrep_transformation_column table
tfc_where_clause	pk_gtrep_cl.f_get_data (TFC_WHERE_ID)

tcs_sample_analysis	(SELECT max (tcs_sample_value) FROM gtrep_tc_sample WHERE gtrep_tc_sample.tcs_sample_type = 'analysis' AND gtrep_tc_sample.tcs_column_name = tcd_column_name AND gtrep_tc_sample.tcs_table_id = td2.td_table_id AND gtrep_tc_sample.tcs_sample_date in (SELECT max(tcs_sample_date) FROM gtrep_tc_sample WHERE tcs_sample_type = 'analysis' AND tcs_column_name = tcd_column_name AND tcs_table_id = td2.td_table_id))
tcs_sample	'Sample'
TPD_PKEY_COUNT	PK_CNT
PK_CNT	count (PKEY_NAME) (see reference 2)
TPD_UKEY_COUNT	UK_CNT
UK_CNT	count (ukey_name) (see reference 2)
tid_index_count	IDX_CNT
IDX_CNT	count (TID_INDEX_NAME) (see reference 2)
auth_stage	tfc_auth_stage
id	coalesce (tfc_id, 0)
tfc_xpath_element	In gtrep_transformation_column table
tfc_dateformat	In gtrep_transformation_column table
tfc_column_part_s	In gtrep_transformation_column table
tfc_column_part_l	In gtrep_transformation_column table
tcs_rowcount	(SELECT cast (max (tcs_sample_value) as numeric) FROM gtrep_tc_sample WHERE gtrep_tc_sample.tcs_sample_type = 'rowcount' AND gtrep_tc_sample.tcs_column_name = tcd_column_name AND gtrep_tc_sample.tcs_table_id = td2.td_table_id AND gtrep_tc_sample.tcs_sample_date in (SELECT max(tcs_sample_date) FROM gtrep_tc_sample WHERE tcs_sample_type = 'rowcount' AND tcs_column_name = tcd_column_name AND tcs_table_id = td2.td_table_id))
tcs_distinct_count	(SELECT cast (max (tcs_sample_value) as numeric) FROM gtrep_tc_sample WHERE gtrep_tc_sample.tcs_sample_type = 'distcount' AND gtrep_tc_sample.tcs_column_name = tcd_column_name AND gtrep_tc_sample.tcs_table_id = td2.td_table_id AND gtrep_tc_sample.tcs_sample_date in (SELECT max(tcs_sample_date) FROM gtrep_tc_sample WHERE tcs_sample_type = 'distcount' AND tcs_column_name = tcd_column_name AND tcs_table_id = td2.td_table_id))

tcs_nullcount	(SELECT cast (max (tcs_sample_value) as numeric) FROM gtrep_tc_sample WHERE gtrep_tc_sample.tcs_sample_type = 'nullcount' AND gtrep_tc_sample.tcs_column_name = tcd_column_name AND gtrep_tc_sample.tcs_table_id = td2.td_table_id AND gtrep_tc_sample.tcs_sample_date in (SELECT max(tcs_sample_date) FROM gtrep_tc_sample WHERE tcs_sample_type = 'nullcount' AND tcs_column_name = tcd_column_name AND tcs_table_id = td2.td_table_id))
tfc_order	In gtrep_transformation_column table

Reference 1 - gtrep_transfomation_column

- tfc_transformation_map
- tfc_proj_id
- tfc_table_name
- tfc_pv_id
- tfc_column_name
- tfc_where_seq
- tfc_action
- tfc_default
- tfc_function
- tfc_list_colno
- tfc_keep_nulls
- tfc_xref
- tfc_xref_ident
- tfc_unique_cols
- tfc_notes
- tfc_override_lookup
- date_created
- who_created
- program_created
- date_updated
- who_updated
- program_updated
- tfc_auth_stage
- tfc_id
- tfd_dateformat
- tfc_xpath_element
- tfc_column_part_l
- tfc_column_part_s
- tfc_preformat
- tfc_where_id
- tfc_order

Reference 2

```
 select TD_TABLE_ID,
  count (TID_INDEX_NAME) IDX_CNT,
  count (PKEY_NAME) PK_CNT,
  count (ukey_name) UK_CNT
from gtrep_table_def
left outer join gtrep_table_ind_def on tid_table_id = td_table_id
left outer join (
    select tpd_table_id, tpd_pukey_name pkey_name
     from gtrep_table_pukey_def
     where tpd_pv_id = <projectVersion>
     and tpd_pukey_type = 'P'
) pk on pk.tpd_table_id = td_table_id
left outer join (
     select tpd_table_id, tpd_pukey_name ukey_name
     from gtrep_table_pukey_def
     where tpd_pv_id = <projectVersion>
     and tpd_pukey_type = 'U'
) uk on uk.tpd_table_id = td_table_id
where (td_table_id in (<list of tableIds>) or (coalesce(:ps_all,'N')='ALL'))
and td_pv_id = <projectVersion>
group by TD_TABLE_ID
```

Reference 3 - Main Query

```
 SELECT
    'columns16.gif' col_icon,
    'add16.gif' new_icon,
    'delete16.gif' delete_icon,
    case when tfc_auth_stage >= 1 then 'Y' else 'N' end as checked,
    case when tfc_auth_stage >= 2 then 'Y' else 'N' end as validated,
    case when tfc_auth_stage >= 3 then 'Y' else 'N' end as approved,
    coalesce (TFC_TABLE_NAME, TD_TABLE_NAME) as td_table_name,
    coalesce (TFC_COLUMN_NAME, TCD_COLUMN_NAME) as tcd_column_name,
    tcd_column_seq,
    tcd_datatype,
    tcd_precision,
    tcd_scale,
    tcd_nullable,
    TFC_ACTION,
    TFC_DEFAULT,
    TFC_FUNCTION,
    case when tfd_fkey_count + tfd_rel_count > 0 then 'treeview16.gif' else null end
 tree_icon,
    TFC_XREF,
    TFC_XREF_IDENT,
    TFC_KEEPNULLS,
```

```
    TFC_LIST_COLNO,
    TFC_OVERRIDE_LOOKUP,
    TFC_UNIQUE_COLS,
    TFC_NOTES,
    TFC_PREFORMAT,
    TCD_FORMAT,
    case when coalesce (tfc_where_id, 0) > 0 then 'Y' else '' end where_clause_yes,
    TFC_WHERE_SEQ,
    pk_gtrep_cl.f_get_data (TFC_WHERE_ID) tfc_where_clause,
    tcs_sample_analysis,
    'Sample' as tcs_sample,
    PK_CNT as TPD_PKEY_COUNT,
    UK_CNT as TPD_UKEY_COUNT,
    IDX_CNT as tid_index_count,
    tfc_auth_stage auth_stage,
    coalesce (tfc_id, 0) id,
    tfc_xpath_element,
    tfc_dateformat,
    tfc_column_part_s,
    tfc_column_part_l,
    tcs_rowcount,
    tcs_distinct_count,
    tcs_nullcount,
    tfc_order
FROM ( select td2.TD_TABLE_ID,
        td2.TD_PROJ_ID,
        td2.TD_TABLE_NAME,
        td2.TD_PV_ID,
        TCD_COLUMN_NAME,
        TCD_COLUMN_SEQ,
        TCD_DATATYPE,
        TCD_PRECISION,
        TCD_SCALE,
        TCD_FORMAT,
        CASE when tcd_nullable = '0' then 'N' else 'Y' end as tcd_nullable,
        IDX_CNT,
        PK_CNT,
        UK_CNT,
        (SELECT max (tcs_sample_value)
            FROM gtrep_tc_sample
            WHERE gtrep_tc_sample.tcs_sample_type = 'analysis'
            AND gtrep_tc_sample.tcs_column_name = tcd_column_name
            AND gtrep_tc_sample.tcs_table_id = td2.td_table_id
            AND gtrep_tc_sample.tcs_sample_date in (
                SELECT max(tcs_sample_date)
                FROM gtrep_tc_sample
```

```
        WHERE tcs_sample_type = 'analysis'
        AND tcs_column_name = tcd_column_name
        AND tcs_table_id = td2.td_table_id
    )
) as tcs_sample_analysis,
(SELECT cast (max (tcs_sample_value) as numeric)
    FROM gtrep_tc_sample
    WHERE gtrep_tc_sample.tcs_sample_type = 'rowcount'
    AND gtrep_tc_sample.tcs_column_name = tcd_column_name
    AND gtrep_tc_sample.tcs_table_id = td2.td_table_id
    AND gtrep_tc_sample.tcs_sample_date in (
        SELECT max(tcs_sample_date)
        FROM gtrep_tc_sample
        WHERE tcs_sample_type = 'rowcount'
        AND tcs_column_name = tcd_column_name
        AND tcs_table_id = td2.td_table_id
    )
) as tcs_rowcount,
(SELECT cast (max (tcs_sample_value) as numeric)
    FROM gtrep_tc_sample
    WHERE gtrep_tc_sample.tcs_sample_type = 'distcount'
    AND gtrep_tc_sample.tcs_column_name = tcd_column_name
    AND gtrep_tc_sample.tcs_table_id = td2.td_table_id
    AND gtrep_tc_sample.tcs_sample_date in (
        SELECT max(tcs_sample_date)
        FROM gtrep_tc_sample
        WHERE tcs_sample_type = 'distcount'
        AND tcs_column_name = tcd_column_name
        AND tcs_table_id = td2.td_table_id
    )
) as tcs_distinct_count,
(SELECT cast (max (tcs_sample_value) as numeric)
    FROM gtrep_tc_sample
    WHERE gtrep_tc_sample.tcs_sample_type = 'nullcount'
    AND gtrep_tc_sample.tcs_column_name = tcd_column_name
    AND gtrep_tc_sample.tcs_table_id = td2.td_table_id
    AND gtrep_tc_sample.tcs_sample_date in (
        SELECT max(tcs_sample_date)
        FROM gtrep_tc_sample
        WHERE tcs_sample_type = 'nullcount'
        AND tcs_column_name = tcd_column_name
        AND tcs_table_id = td2.td_table_id
    )
) as tcs_nullcount,
(select count (rel_name)
    from gtrep_relationship
```

```
            join gtrep_rel_column on rc_rel_id = rel_id
            where rc_parent_column = tcd_column_name
            and rel_parent_table = td2.td_table_name
            and rel_proj_id in (select pv_proj_id from gtrep_project_version where pv_id
 = :pl_pv_id)
        ) as tfd_rel_count,
        (select count (tfd_fkey_name)
            from gtrep_table_fkey_def
            where tfd_ref_column_name = tcd_column_name
            and tfd_ref_table_name = td2.td_table_name
            and tfd_pv_id = :pl_pv_id
        ) as tfd_fkey_count

    from ( select TD_TABLE_ID,
            count (TID_INDEX_NAME) IDX_CNT,
            count (PKEY_NAME) PK_CNT,
            count (ukey_name) UK_CNT
        from gtrep_table_def
        left outer join gtrep_table_ind_def on tid_table_id = td_table_id
        left outer join ( select tpd_table_id,
            tpd_pukey_name pkey_name
            from gtrep_table_pukey_def
            where tpd_pv_id = :pl_pv_id
            and tpd_pukey_type = 'P') pk on pk.tpd_table_id = td_table_id
        left outer join (select tpd_table_id,
            tpd_pukey_name ukey_name
            from gtrep_table_pukey_def
            where tpd_pv_id = :pl_pv_id
            and tpd_pukey_type = 'U') uk on uk.tpd_table_id = td_table_id
        where (td_table_id in (:pl_table_id) or (coalesce(:ps_all,'N')='ALL'))
        and td_pv_id = :pl_pv_id
        group by TD_TABLE_ID
    ) td1
    join gtrep_table_def td2 on td2.td_table_id = td1.td_table_id
    join gtrep_table_col_def on tcd_table_id = td1.td_table_id
) td
left outer join GTREP_TRANSFORMATION_COLUMN
ON TFC_TRANSFORMATION_MAP = :ps_transformation_map
AND TFC_TABLE_NAME = TD_TABLE_NAME
AND TFC_COLUMN_NAME = TCD_COLUMN_NAME
AND TFC_PV_ID = TD_PV_ID
```

Arguments are `<table list ids> <projectVersion> ps_transformation_map ps_all`.

Sorting was done on `td_table_name A tcd_column_seq A has_where_clause A tfc_where_seq A`.

Custom Filter Functions for Transformation Maps

When you work with transformation maps, you can use default filters, or create custom filters using the functions in this reference. These functions are a subset of PowerBuilder language. In the Transformation Maps window, choose **Custom filter…** and click the filter icon to open the **Specify Filter** dialog box.

Abs expression function

Calculates the absolute value of a number.

Syntax

```
Abs ( n )
```

- ***n*** — The number for which you want the absolute value

Return Values

The datatype of n. Returns the absolute value of n.

Examples

This expression counts all the product numbers where the absolute value of the product number is distinct:

```
Count(product_number for All DISTINCT Abs (product_number))
```

Only data with an absolute value greater than 5 passes this validation rule:

```
Abs(value_set) > 5
```

ACos expression function

Calculates the arc cosine of an angle.

Syntax

```
ACos ( n )
```

- ***n*** — The ratio of the lengths of two sides of a triangle for which you want a corresponding angle (in radians). The ratio must be a value between -1 and 1.

Return Values

Double. Returns the arc cosine of n if it succeeds.

Examples

This expression returns 0:

```
ACos(1)
```

This expression returns 3.141593 (rounded to six places):

```
ACos(-1)
```

This expression returns 1.000000 (rounded to six places):

```
ACos(.540302)
```

Asc expression function

Converts the first character of a string to its Unicode code point. A Unicode code point is the numerical integer value given to a Unicode character.

Syntax

```
Asc ( string )
```

- ***string*** — The string for which you want the code point value of the first character

Return Values

Unsigned integer. Returns the code point value of the first character in string.

Usage

Use Asc to test the case of a character or manipulate text and letters. To find out the case of a character, you can check whether its code point value is within the appropriate range.

Examples

This expression for a computed field returns the string in code_id if the code point value of the first character in code_id is A (65):

```
If (Asc(code_id) = 65, code_id, "Not a valid code")
```

This expression for a computed field checks the case of the first character of lname and if it is lowercase, makes it uppercase:

```
IF (Asc(lname) > 64 AND Asc(lname) < 91, lname, WordCap(lname))
```

AscA expression function

Converts the first character of a string to its ASCII integer value.

Syntax

```
AscA ( string )
```

- ***string*** — The string for which you want the ASCII value of the first character

Return Values

Integer. Returns the ASCII value of the first character in string.

Usage

Use AscA to test the case of a character or manipulate text and letters. To find out the case of a character, you can check whether its ASCII value is within the appropriate range.

Examples

This expression for a computed field returns the string in code_id if the ASCII value of the first character in code_id is A (65):

```
If (AscA(code_id) = 65, code_id, "Not a valid code")
```

This expression for a computed field checks the case of the first character of lname and if it is lowercase, makes it uppercase:

```
IF (AscA(lname) > 64 AND AscA(lname) < 91, lname, WordCap(lname))
```

ASin expression function

Calculates the arc sine of an angle.

Syntax

```
ASin ( n )
```

- ***n*** — The ratio of the lengths of two sides of a triangle for which you want a corresponding angle (in radians). The ratio must be a value between -1 and 1.

Return Values

Double. Returns the arc sine of n if it succeeds.

Examples

This expression returns .999998 (rounded to six places):

```
ASin(.84147)
```

This expression returns .520311 (rounded to six places):

```
ASin(LogTen (Pi (1)))
```

This expression returns 0:

```
ASin(0)
```

ATan expression function

Calculates the arc tangent of an angle.

Syntax

```
ATan ( n )
```

- ***n*** — The ratio of the lengths of two sides of a triangle for which you want a corresponding angle (in radians)

Return Values

Double. Returns the arc tangent of n if it succeeds.

Examples

This expression returns 0:

```
ATan(0)
```

This expression returns 1.000 (rounded to three places):

```
ATan(1.55741)
```

This expression returns 1.267267 (rounded to six places):

```
ATan(Pi(1))
```

Avg expression function

Calculates the average of the values of the column.

Syntax

```
Avg ( column { FOR range { DISTINCT { expres1 {, expres2 {, ... } } } } } )
```

- ***column*** — The column for which you want the average of the data values. Column can be the column name or the column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column. The datatype of column must be numeric.
- **FOR *range*** — (optional) The data to include in the average. For most presentation styles, values for range are:

  - **ALL** – (Default) The average of all values in column.
  - **GROUP *n*** – The average of values in column in the specified group. Specify the keyword GROUP followed by the group number, for example, GROUP 1.
  - **PAGE** – The average of the values in column on a page.
  - **CROSSTAB** – (Crosstabs only) The average of all values in column in the crosstab.
  - **GRAPH** – (Graphs only) The average of values in column in the range specified for the Rows option.
  - **OBJECT** – (OLE objects only) The average of values in column in the range specified for the Rows option.
- **DISTINCT** — (optional) Causes Avg to consider only the distinct values in column when calculating the average. For a value of column, the first row found with the value is used and other rows that have the same value are ignored.
- ***expresn*** — (optional) One or more expressions that you want to evaluate to determine distinct rows. Expresn can be the name of a column, a function, or an expression.

Return Values

The numeric datatype of the column. Returns the average of the values of the rows in range.

Usage

If you specify range, Avg returns the average value of column in range. If you specify DISTINCT, Avg returns the average value of the distinct values in column, or if you specify expresn, the average of column for each distinct value of expresn. For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data property page (the Range property), and the aggregation function uses that range. Settings for Rows include the following:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

In calculating the average, null values are ignored.

You cannot use this or other aggregate functions in validation rules or filter expressions. Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a `DataWindow` object always retrieves all rows.

Examples

This expression returns the average of the values in the column named salary:

```
Avg(salary)
```

This expression returns the average of the values in group 1 in the column named salary:

```
Avg(salary for group 1)
```

This expression returns the average of the values in column 5 on the current page:

```
Avg(#5 for page)
```

This computed field returns Above Average if the average salary for the page is greater than the average salary:

```
If(Avg(salary for page) > Avg(salary), "Above Average", " ")
```

This expression for a graph value sets the data to the average value of the sale_price column:

```
Avg(sale_price)
```

This expression for a graph value sets the data value to the average value of the sale_price column for the entire graph:

```
Avg(sale_price for graph)
```

Assuming a DataWindow object displays the order number, amount, and line items for each order, this computed field returns the average of the order amount for the distinct order numbers:

```
Avg(order_amt for all DISTINCT order_nbr)
```

Bitmap expression function

Displays the specified bitmap.

Syntax

```
Bitmap ( string )
```

- **string** — A column containing bitmap files, a string containing the name of an image file (a BMP, GIF, JPEG, RLE, or WMF file), or an expression that evaluates to a string containing the name of an image file.

Return Values

The special datatype bitmap, which cannot be used in any other function.

Usage

Use Bitmap to dynamically display a bitmap in a computed field. When *string* is a column containing bitmap files, a different bitmap can display for each row.

You can use the Bitmap function only in a computed field.

Examples

These examples are all expressions for a computed field.

This expression dynamically displays the bitmap file contained in the column named employees:

```
Bitmap(employees)
```

If the employees column is column 3, this next expression gives the same result as the expression above:

```
Bitmap(#3)
```

This expression displays the bitmap tools.bmp:

```
Bitmap("TOOLS.BMP")
```

This expression tests the value in the column named password and then uses the value to determine which bitmap to display:

```
Bitmap(If(password = "y", "yes.bmp", "no.bmp"))
```

Case expression function

Tests the values of a column or expression and returns values based on the results of the test.

Syntax

```
Case ( column WHEN value1 THEN result1 { WHEN value2 THEN result2 { ... } } { ELSE resultelse } )
```

- ***column*** — The column or expression whose values you want to test. Column can be the column name or the column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column. Column is compared to each valuen.
- **WHEN** — (optional) Introduces a value-result pair. At least one WHEN is required.
- ***valuen*** — One or more values that you want to compare to values of column. A value can be:

  - A single value
  - A list of values separated by commas (for example, 2, 4, 6, 8)
  - A TO clause (for example, 1 TO 20)
  - IS followed by a relational operator and comparison value (for example, IS>5)
  - Any combination of the above with an implied OR between expressions (for example, 1,3,5,7,9,27 TO 33, IS>42)
- **THEN** — Introduces the result to be returned when column matches the corresponding *valuen*.
- ***resultn*** — An expression whose value is returned by Case for the corresponding valuen. All *resultn* values must have the same datatype.
- **ELSE** — (optional) Specifies that for any values of column that do not match the values of *valuen* already specified, Case returns *resultelse*.
- ***resultelse*** — An expression whose value is returned by Case when the value of column does not match any WHEN *valuen* expression.

Return Values

The datatype of *resultn*. Returns the result you specify in *resultn*. If more than one WHEN clause matches *column*, Case returns the result of the first matching one.

Examples

This expression for the `Background.Color` property of a `Salary` column returns values that represent red when an employee's salary is greater than $70,000, green when an employee's salary is greater than $50,000, and blue otherwise:

```
Case(salary WHEN IS >70000 THEN RGB(255,0,0) WHEN IS
>50000 THEN RGB(0,255,0) ELSE RGB(0,0,255))
```

This expression for the Background.Color property of an employee Id column returns red for Id 101, gray for Id 102, and black for all other Id numbers:

```
Case(emp_id WHEN 101 THEN 255 WHEN 102 THEN
RGB(100,100,100) ELSE 0)
```

This expression for the Format property of the `Marital_status` column returns Single, Married, and Unknown based on the data value of the Marital_status column for an employee:

```
Case(marital_status WHEN 'S'THEN 'Single' WHEN 'M' THEN 'Married' ELSE 'Unknown')
```

Ceiling expression function

Retrieves the smallest whole number that is greater than or equal to a specified limit.

Syntax

```
Ceiling ( n )
```

- **n** — The number for which you want the smallest whole number that is greater than or equal to it.

Return Values

The datatype of n. Returns the smallest whole number that is greater than or equal to n.

Examples

These expressions both return -4:

```
Ceiling(-4.2)
Ceiling(-4.8)
```

This expression for a computed field returns ERROR if the value in discount_amt is greater than the smallest whole number that is greater than or equal to discount_factor times price. Otherwise, it returns discount_amt:

```
If(discount_amt <= Ceiling(discount_factor * price), String(discount_amt), "ERROR")
```

To pass this validation rule, the value in discount_amt must be less than or equal to the smallest whole number that is greater than or equal to discount_factor times price:

```
discount_amt <= Ceiling(discount_factor * price)
```

Char expression function

Converts an integer to a Unicode character.

Syntax

```
Char ( n )
```

- **n** — The integer you want to convert to a character

Return Values

String. Returns the character whose code point value is n.

Examples

This expression returns the escape character:

```
Char(27)
```

CharA expression function

Converts an integer to an ASCII character.

Syntax

```
CharA ( n )
```

- ***n*** — The integer you want to convert to a character.

Return Values

String. Returns the character whose ASCII value is n.

Examples

This expression returns the escape character:

```
CharA(27)
```

Cos expression function

Calculates the cosine of an angle.

Syntax

```
Cos ( n )
```

- ***n*** — The angle (in radians) for which you want the cosine

Return Values

Double. Returns the cosine of n.

Examples

This expression returns 1:

```
Cos(0)
```

This expression returns .540302:

```
Cos(1)
```

This expression returns -1:

```
Cos(Pi(1))
```

Count expression function

Calculates the total number of rows in the specified column.

Syntax

```
Count ( column { FOR range { DISTINCT { expres1 {, expres2 {, ... } } } } } )
```

- ***column*** — The column for which you want the number of rows. Column can be the column name or the column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column.
- **FOR *range*** — (optional) The data that will be included in the count. For most presentation styles, values for *range* are:
  - **ALL** — (Default) The count of all rows in column.
  - **GROUP *n*** — The count of rows in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** — The count of the rows in column on a page.
  - **CROSSTAB** — (Crosstabs only) The count of all rows in column in the crosstab.
  - **GRAPH** — (Graphs only) The count of values in column in the range specified for the Rows option.
  - **OBJECT** — (OLE objects only) The count of values in column in the range specified for the Rows option.
- **DISTINCT** — (optional) Causes Count to consider only the distinct values in column when counting the rows. For a value of column, the first row found with the value is used and other rows that have the same value are ignored.
- ***expresn*** — (optional) One or more expressions that you want to evaluate to determine distinct rows. Expresn can be the name of a column, a function, or an expression.

Usage

If you specify *range*, Count determines the number of rows in column in range. If you specify DISTINCT, Count returns the number of the distinct rows displayed in column, or if you specify *expresn*, the number of rows displayed in column where the value of *expresn* is distinct. For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data property page (the Range property), and the aggregation function uses that range.

Settings for Rows include the following:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

Null values in the column are ignored and are not included in the count.

You cannot use this or other aggregate functions in validation rules or filter expressions.

Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a DataWindow object always retrieves all rows.

Examples

This expression returns the number of rows in the column named emp_id that are not null:

```
Count(emp_id)
```

This expression returns the number of rows in the column named emp_id of group 1 that are not null:

```
Count(emp_id for group 1)
```

This expression returns the number of dept_ids that are distinct:

```
Count(dept_id for all DISTINCT)
```

This expression returns the number of regions with distinct products:

```
Count(region_id for all DISTINCT Lower(product_id))
```

This expression returns the number of rows in column 3 on the page that are not null:

```
Count(#3 for page)
```

CrosstabAvg expression function

Calculates the average of the values returned by an expression in the values list of the crosstab. When the crosstab definition has more than one column, CrosstabAvg can also calculate averages of the expression's values for groups of column values. For more information, see How to Use Functions in a Crosstab.

Syntax

```
CrosstabAvg ( n {, column, groupvalue } )
```

- ***n*** — The number of the crosstab-values expression for which you want the average of the returned values. The crosstab expression must be numeric.
- ***column*** — (optional) The number of the crosstab column as it is listed in the Columns box of the Crosstab Definition dialog box for which you want intermediate calculations.
- ***groupvalue*** — (optional) A string whose value controls the grouping for the calculation. Groupvalue is usually a value from another column in the crosstab. To specify the current column value in a dynamic crosstab, rather than a specific value, specify @ plus the column name as a quoted string.

Return Values

Double. Returns the average of the crosstab values returned by expression n for all the column values or, optionally, for a subset of column values.

Usage

This function is meaningful only for the average of the values of the expression in a row in the crosstab. This means you can use it only in the detail band, not in a header, trailer, or summary band. Null values are ignored and are not included in the average.

You can use this function only in a crosstab DataWindow object. For details, see How to Use Functions in a Crosstab.

Examples

The first two examples use the crosstab expressions shown below:

```
Count(emp_id for crosstab),Sum(salary for crosstab)
```

This expression for a computed field in the crosstab returns the average of the employee counts (the first expression):

```
CrosstabAvg(1)
```

This expression for a computed field in the crosstab returns the average of the salary totals (the second expression):

```
CrosstabAvg(2)
```

Consider a crosstab that has two columns (region and city) and the values expression Avg(sales for crosstab). This expression for a computed field in the detail band computes the average sales over all the cities in a region:

```
CrosstabAvg(1, 2, "@region")
```

This expression for another computed field in the same crosstab computes the grand average over all the cities:

```
CrosstabAvg(1)
```

CumulativePercent expression function

Calculates the total value of the rows up to and including the current row in the specified column as a percentage of the total value of the column (a running percentage).

Syntax

```
CumulativePercent ( column { FOR range } )
```

- ***column*** — The column for which you want the cumulative value of the rows up to and including the current row as a percentage of the total value of the column for range. Column can be the column name or the column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column. The datatype of column must be numeric.
- **FOR *range*** — (optional) The data that will be included in the cumulative percentage. For most presentation styles, values for range are:
  - **ALL** — (Default) The cumulative percentage of all rows in column.
  - **GROUP *n*** — The cumulative percentage of rows in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** — The cumulative percentage of the rows in column on a page.
  - **CROSSTAB** — (Crosstabs only) The cumulative percentage of all rows in column in the crosstab.
  - **GRAPH** — (Graphs only) The cumulative percentage of values in column in the range specified for the Rows option.
  - **OBJECT** — (OLE objects only) The cumulative percentage of values in column in the range specified for the Rows option.

Return Values

Long. Returns the cumulative percentage value.

Usage

If you specify range, CumulativePercent restarts the accumulation at the start of the range.

For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data property page (the Range property), and the aggregation function uses that range.
Settings for Rows include the following:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

In calculating the percentage, null values are ignored.

You cannot use this or other aggregate functions in validation rules or filter expressions.

Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a DataWindow object always retrieves all rows.

Examples

This expression returns the running percentage for the values that are not null in the column named salary:

```
CumulativePercent(salary)
```

This expression returns the running percentage for the column named salary for the values in group 1 that are not null:

```
CumulativePercent(salary for group 1)
```

This expression entered in the Value box on the Data property page for a graph returns the running percentage for the salary column for the values in the graph that are not null:

```
CumulativePercent(salary for graph)
```

This expression in a crosstab computed field returns the running percentage for the salary column for the values in the crosstab that are not null:

```
CumulativePercent(salary for crosstab)
```

CumulativeSum expression function

Calculates the total value of the rows up to and including the current row in the specified column (a running total).

Syntax

```
CumulativeSum ( column { FOR range } )
```

- ***column*** — The column for which you want the cumulative total value of the rows up to and including the current row for group. Column can be the column name or the column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column. The datatype of column must be numeric.
- **FOR *range*** — (optional) The data that will be included in the cumulative sum. For most presentation styles, values for range are:
  - **ALL** – (Default) The cumulative sum of all values in column.
  - **GROUP *n*** – The cumulative sum of values in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** – The cumulative sum of the values in column on a page.
  - **CROSSTAB** – (Crosstabs only) The cumulative sum of all values in column in the crosstab.
  - **GRAPH** – (Graphs only) The cumulative sum of values in column in the range specified for the Rows option.
  - **OBJECT** – (OLE objects only) The cumulative sum of values in column in the range specified for the Rows option.

Return Values

The appropriate numeric datatype. Returns the cumulative total value of the rows.

Usage

If you specify range, CumulativeSum restarts the accumulation at the start of the range.

For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data property page (the Range property), and the aggregation function uses that range. Settings for Rows include the following:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

In calculating the sum, null values are ignored.

Examples

This expression returns the running total for the values that are not null in the column named salary:

```
CumulativeSum(salary)
```

This expression returns the running total for the values that are not null in the column named salary in group 1:

```
CumulativeSum(salary for group 1)
```

This expression entered in the Value box on the Data property page for a graph returns the running total for the salary column for the values in the graph that are not null:

```
CumulativeSum(salary for graph)
```

This expression in a crosstab computed field returns the running total for the salary column for the values in the crosstab that are not null:

```
CumulativeSum(salary for crosstab)
```

CurrentRow expression function

Reports the number of the current row (the row with focus).

Definition: The current row is not always a row displayed on the screen. For example, if the cursor is on row 7 column 2 and the user uses the scroll bar to scroll to row 50, the current row remains row 7, unless the user clicks row 50.

Syntax

```
CurrentRow ( )
```

Return Values

Long. Returns the number of the row if it succeeds and 0 if no row is current.

Examples

This expression in a computed field returns the number of the current row:

```
CurrentRow()
```

This expression for a computed control displays an arrow bitmap as an indicator for the row with focus and displays no bitmap for rows not having focus. As the user moves from row to row, an arrow marks where the user is:

```
Bitmap(If(CurrentRow() = GetRow(),"arrow.bmp",""))
```

Alternatively, this expression for the Visible property of an arrow picture control makes the arrow bitmap visible for the row with focus and invisible for rows not having focus. As the user moves from row to row, an arrow marks where the user is:

```
If(CurrentRow() = GetRow(), 1, 0)
```

Date expression function

Converts a string whose value is a valid date to a value of datatype date.

Syntax

```
Date ( string )
```

- ***string*** — A string containing a valid date (such as Jan 1, 2004, or 12-31-99) that you want returned as a date.

Return Values

Date. Returns the date in string as a date. If string does not contain a valid date, Date returns null.

Usage

The value of the string must be a valid date.

NOTE

To make sure you get correct return values for the year, you must verify that yyyy is the Short Date Style for year in the Regional Settings of the user's Control Panel. Your program can check this with the RegistryGet function. If the setting is not correct, you can ask the user to change it manually or to have the application change it (by calling the RegistrySet function). The user might need to reboot after the setting is changed.

Valid dates

Valid dates can include any combination of day (1–31), month (1–12 or the name or abbreviation of a month), and year (two or four digits). Leading zeros are optional for month and day. If the month is a name or an abbreviation, it can come before or after the day; if it is a number, it must be in the month location specified in the Windows control panel. A 4-digit number is assumed to be a year.

If the year is two digits, the assumption of century follows this rule: for years between 00 and 49, the first two digits are assumed to be 20; for years between 50 and 99, the first two digits are assumed to be 19. If your data includes dates before 1950, such as birth dates, always specify a four-digit year to ensure the correct interpretation.

The function handles years from 1000 to 3000 inclusive.

An expression has a more limited set of datatypes than the functions that can be part of the expression. Although the Date function returns a date value, the whole expression is promoted to a DateTime value. Therefore, if your expression consists of a single Date function, it will appear that Date returns the wrong datatype. To display the date without the time, consult the PowerBuilder documentation, and choose an appropriate display format.

Examples

These expressions all return the date datatype for July 4, 2004 when the default location of the month in Regional Settings is center:

```
Date("2004/07/04")
Date("2004 July 4")
Date("July 4, 2004")
```

DateTime expression function

Combines a date and a time value into a DateTime value.

Syntax

```
DateTime ( date {, time } )
```

- ***date*** — A valid date (such as Jan 1, 2005, or 12-31-99) or a blob variable whose first value is a date that you want included in the value returned by DateTime.
- ***time*** — (optional) A valid time (such as 8am or 10:25:23:456799) or a blob variable whose first value is a time that you want included in the value returned by DateTime. If you include a time, only the hour portion is required. If you omit the minutes, seconds, or microseconds, they are assumed to be zeros. If you omit am or pm, the hour is determined according to the 24-hour clock.

Return Values

DateTime. Returns a DateTime value based on the values in date and optionally time. If time is omitted, DateTime uses 00:00:00.000000 (midnight).

Usage

To display microseconds in a time, the display format for the field must include microseconds. For information on valid dates, see Date.

Examples

This expression returns the values in the order_date and order_time columns as a DateTime value that can be used to update the database:

```
DateTime(Order_Date, Order_Time)
```

Using this expression for a computed field displays 11/11/01 11:11:00:

```
DateTime(11/11/01, 11:11)
```

Day expression function

Obtains the day of the month in a date value.

Syntax

```
Day ( date )
```

- ***date*** — The date for which you want the day

Return Values

Integer. Returns an integer (1–31) representing the day of the month in date.

Examples

This expression returns 31:

```
Day(2005-01-31)
```

This expression returns the day of the month in the start_date column:

```
Day(start_date)
```

DayName expression function

Gets the day of the week in a date value and returns the weekday's name.

Syntax

```
DayName ( date )
```

- ***date*** — The date for which you want the name of the day

Return Values

String. Returns a string whose value is the name of the weekday (Sunday, Monday, and so on) for date.

Usage

DayName returns a name in the language of the deployment files available on the machine where the application is run. If you have installed localized deployment files in the development environment or on a user's machine, then on that machine the name returned by DayName will be in the language of the localized files.

For information about localized deployment files, please consult the PowerBuilder documentation.

Examples

This expression for a computed field returns Okay if the day in date_signed is not Sunday:

```
If(DayName(date_signed) <> "Sunday", "Okay", "Invalid Date")
```

To pass this validation rule, the day in date_signed must not be Sunday:

```
DayName(date_signed) <> "Sunday"
```

DayName expression function

Gets the day of the week in a date value and returns the weekday's name.

Syntax

```
DayName ( date )
```

- ***date*** — The date for which you want the name of the day

Return Values

String. Returns a string whose value is the name of the weekday (Sunday, Monday, and so on) for date.

Usage

DayName returns a name in the language of the deployment files available on the machine where the application is run. If you have installed localized deployment files in the development environment or on a user's machine, then on that machine the name returned by DayName will be in the language of the localized files.

For information about localized deployment files, see the chapter on internationalizing an application in Application Techniques.

Examples

This expression for a computed field returns Okay if the day in date_signed is not Sunday:

```
If(DayName(date_signed) <> "Sunday", "Okay", "Invalid Date")
```

To pass this validation rule, the day in date_signed must not be Sunday:

```
DayName(date_signed) <> "Sunday"
```

DaysAfter expression function

Gets the number of days one date occurs after another.

Syntax

```
DaysAfter ( date1, date2 )
```

- ***date1*** — A date value that is the start date of the interval being measured.
- ***date2*** — A date value that is the end date of the interval.

Return Values

Long. Returns a long containing the number of days date2 occurs after date1. If date2 occurs before date1, DaysAfter returns a negative number.

Examples

This expression returns 4:

```
DaysAfter(2005-12-20, 2005-12-24)
```

This expression returns -4:

```
DaysAfter(2005-12-24, 2005-12-20)
```

This expression returns 0:

```
DaysAfter(2005-12-24, 2005-12-24)
```

This expression returns 5:

```
DaysAfter(2004-12-29, 2005-01-03)
```

Dec expression function

Converts the value of a string to a decimal.

Syntax

```
Dec ( string )
```

- ***string*** — The string you want returned as a decimal

Return Values

Decimal. Returns the contents of string as a decimal if it succeeds and 0 if string is not a number.

Usage

The decimal datatype supports up to 28 digits. You can also append the letter D in upper or lowercase to identify a number as a decimal constant in DataWindow expressions. For example, 2.0d and 123.456789012345678901D are treated as decimals.

Examples

This expression returns the string 24.3 as a decimal datatype:

```
Dec("24.3")
```

This expression for a computed field returns "Not a valid score" if the string in the score column does not contain a number. The expression checks whether the Dec function returns 0, which means it failed to convert the value:

```
If ( Dec(score) <> 0, score, "Not a valid score")
```

This expression returns 0:

```
Dec("3ABC") // 3ABC is not a number
```

This validation rule checks that the value in the column the user entered is greater than 1999.99:

```
Dec(GetText()) > 1999.99
```

This validation rule for the column named score insures that score contains a string:

```
Dec(score) <> 0
```

Describe method

Reports the values of properties of a DataWindow object and controls within the DataWindow object. Each column and graphic control in the DataWindow has a set of properties. You specify one or more properties as a string, and Describe returns the values of the properties.

Describe can also evaluate expressions involving values of a particular row and column. When you include Describe's Evaluate function in the property list, the value of the evaluated expression is included in the reported information.

Controls

The three DataWindow types apply to the following controls:

- PowerBuilder — Applies to DataWindow control, DataWindowChild object, DataStore object.
- Web — Applies to Server component .
- Web ActiveX — Applies to DataWindow control, DataWindowChild object.

Syntax

string dwcontrol.Describe (string propertylist)

- ***dwcontrol*** — A reference to a DataWindow control, DataStore, or child DataWindow.
- ***propertylist*** — A string whose value is a blank-separated list of properties or Evaluate functions. For a list of valid properties, see "DataWindow Object Properties."

Return Values

Returns a string that includes a value for each property or Evaluate function. A newline character (~n or \n) separates the value of each item in propertylist.

If the property list contains an invalid item, Describe returns an exclamation point (!) for that item and ignores the rest of the property list. Describe returns a question mark (?) if there is no value for a property.

When the value of a property contains an exclamation point or a question mark, the value is returned in quotes so that you can distinguish between the returned value and an invalid item or a property with no value.

If any argument's value is null, in PowerBuilder and JavaScript the method returns null.

Usage

Use Describe to understand the structure of a DataWindow. For example, you can find out which bands the DataWindow uses and what the datatypes of the columns are. You can also use Describe to find out the current value of a property and use that value to make further modifications.

Describe is often used to obtain the DataWindow's SELECT statement in order to modify it (for example, by adding a WHERE clause).

When you can obtain the DataWindow's SQL statement: When you use the Select painter to graphically create a SELECT statement, PowerBuilder saves its own SELECT statement (called a PBSELECT statement), and not a SQL SELECT statement, with the DataWindow definition. When you call Describe with the property Table.Select, it returns a SQL SELECT statement only if you are connected to the database. If you are not connected to the database, Describe returns a PBSELECT statement.

Property syntax

The syntax for a property in the property list is:

```
controlname.property
```

When a property returns a list, the tab character separates the values in the list. For example, the Bands property reports all the bands in use in the DataWindow as a list.

```
header[tab]detail[tab]summary[tab]footer[tab]header.1[tab]trailer.1
```

If the first character in a property's returned value list is a quotation mark, it means the whole list is quoted and any quotation marks within the list are single quotation marks.

For example, the following is a single property value.

```
" Student[tab]'Andrew'or'[newline]Andy' "
```

Specifying special characters

There are different ways of specifying special characters in a string in each environment:

Character	PowerBuilder	JavaScript
tab	~t	\t
newline	~n	\n
single quote	~'	\'
double quote	~"	\"

Quoted property values

Describe returns a property's value enclosed in quotes when the text would otherwise be ambiguous. For example, if the property's value includes a question mark, then the text is returned in quotes. A question mark without quotes means that the property has no value.

Column name or number

When the control is a column, you can specify the column name or a pound sign (#) followed by the column number. For example, if salary is column 5, then "salary.coltype" is equivalent to "#5.coltype".

Control names

The DataWindow painter automatically gives names to all controls. In previous versions of PowerBuilder, the painter only named columns and column labels.

Evaluating an expression

Describe's Evaluate function allows you to evaluate DataWindow painter expressions within a script using data in the DataWindow. Evaluate has the following syntax, which you specify for propertylist.

```
Evaluate ( 'expression', rownumber )
```

Expression is the expression you want to evaluate and rownumber is the number of the row for which you want to evaluate the expression. The expression usually includes DataWindow painter functions. For example, in the following statement, Describe reports either 255 or 0 depending on the value of the salary column in row 3:

```
ls_ret = dw_1.Describe( & "Evaluate('If(salary > 100000, 255, 0)', 3)")
```

You can call DataWindow control functions in a script to get data from the DataWindow, but some painter functions (such as LookUpDisplay) cannot be called in a script. Using Evaluate is the only way to call them.

Sample property values

To illustrate the types of values that Describe reports, consider a DataWindow called dw_emp with one group level. Its columns are named emp and empname, and its headers are named emp_h and empname_h. The following table shows several properties and the returned value. In the first example below, a sample command shows how you might specify these properties for Describe and what it reports.

The following table shows examples of return values for Describe method:

Property	Reported value	Comment
datawindow.Bands	header[tab]detail[tab]summary[tab]footer[tab]header.1[tab]trailer.1	
datawindow.Objects	emp[tab]empname[tab]emp_h[tab]empname_hemp.Type column	
empname.Type	column	
empname_h.Type	text	
emp.Coltype	char(20)	
state.Type	!	The exclamation point indicates an invalid item: There is no column named state.
empname_h.Visible	?	

PowerBuilder Examples

This example calls Describe with some of the properties shown in the previous table. The reported values (formatted with tabs and newlines) follow. Note that because state is not a column in the DataWindow, state.type returns an exclamation point:

```
string ls_request, ls_report
ls_request = "DataWindow.Bands DataWindow.Objects "&
    + "empname_h.Text " &
    + "empname_h.Type emp.Type emp.Coltype " &
    + "state.Type empname.Type empname_h.Visible"
ls_report = dw_1.Describe(ls_request)
```

Describe sets the value of ls_report to the following string:

```
header~tdetail~tsummary~tfooter~theader.1~ttrailer.1~N emp~tempname~temp_h~tempname_h~N "Employee~R~NName"~N
 text~N column~Nchar(20)~N!
```

These statements check the datatype of the column named salary before using GetItemNumber to obtain the salary value:

```
string ls_data_type
integer li_rate
ls_data_type = dw_1.Describe("salary.ColType")
IF ls_data_type = "number" THEN
li_rate = dw_1.GetItemNumber(5, "salary")
ELSE
. . . // Some processing
END IF
```

Example: Column name or number

This statement finds out the column type of the current column, using the column name:

```
s = This.Describe(This.GetColumnName()+ ".ColType")
```

For comparison, this statement finds out the same thing, using the current column's number:

```
s = This.Describe("#" + String(This.GetColumn()) &
+ ".ColType")
```

Example: Scrolling and the current row

This example, as part of the DataWindow control's ScrollVertical event, makes the first visible row the current row as the user scrolls through the DataWindow:

```
s = This.Describe("DataWindow.FirstRowOnPage")

IF IsNumber(s) THEN This.SetRow(Integer(s))
```

Example: Evaluating the display value of a DropDownDataWindow

This example uses Describe's Evaluate function to find the display value in a DropDownDataWindow column called state_code. You must execute the code after the ItemChanged event, so that the value the user selected has become the item value in the buffer. This code is the script of a custom user event called getdisplayvalue:

```
string rownumber, displayvalue
rownumber = String(dw_1.GetRow())
displayvalue = dw_1.Describe( &
"Evaluate('LookUpDisplay(state_code) ', " &
+ rownumber + ")")
```

This code, as part of the ItemChanged event's script, posts the getdisplayvalue event:

```
dw_1.PostEvent("getdisplayvalue")
```

Example: Assigning null values based on the column's datatype

The following excerpt from the ItemError event script of a DataWindow control allows the user to blank out a column and move to the next column. For columns with datatypes other than string, the user cannot leave the value empty (which is an empty string and does not match the datatype) without the return code. Data and row are arguments of the ItemError event:

```
string s
s = This.Describe(This.GetColumnName() &
+ ".Coltype")
CHOOSE CASE s
CASE "number"
IF Trim(data) = "" THEN
integer null_num
SetNull(null_num)
This.SetItem(row, &
This.GetColumn(), null_num)
RETURN 3
END IF
CASE "date"
IF Trim(data) = "" THEN
date null_date
SetNull(null_date)
This.SetItem(row, &
This.GetColumn(), null_date)
RETURN 3
END IF
. . . // Additional cases for other datatypes
END CHOOSE
```

Exp expression function

Raises e to the specified power *n*.

Syntax

```
Exp ( n )
```

- ***n*** — The power to which you want to raise e (2.71828)

Return Values

Double. Returns e raised to the power n.

Examples

This expression returns 7.38905609893065:

```
Exp(2)
```

Fact expression function

Gets the factorial of a number.

Syntax

```
Fact ( n )
```

- ***n*** — The number for which you want the factorial

Return Values

Double. Returns the factorial of n.

Examples

This expression returns 24:

```
Fact(4)
```

Both these expressions return 1:

```
Fact(1)
Fact(0)
```

Fill expression function

Builds a string of the specified length by repeating the specified characters until the result string is long enough.

Syntax

```
Fill ( chars, n )
```

chars A string whose value will be repeated to fill the return string

- ***n*** — A long whose value is the number of characters in the string you want returned

Return Values

String. Returns a string n characters long filled with repetitions of the characters in the argument chars. If the argument chars has more than n characters, the first n characters of chars are used to fill the return string. If the argument chars has fewer than n characters, the characters in chars are repeated until the return string has n characters.

Usage

Fill is used to create a line or other special effect. For example, asterisks repeated in a printed report can fill an amount line, or hyphens can simulate a total line in a screen display.

Examples

This expression returns a string containing 35 asterisks:

```
Fill("*", 35)
```

This expression returns the string "**-+-+-+-**":

```
Fill("-+", 7)
```

This expression returns 10 tildes (**~**):

```
Fill("~", 10)
```

FillA expression function

Builds a string of the specified length in bytes by repeating the specified characters until the result string is long enough.

Syntax

```
FillA ( chars, n )
```

- ***chars*** — A string whose value will be repeated to fill the return string.
- ***n*** — A long whose value is the number of bytes in the string you want returned.

Return Values

String. Returns a string n bytes long filled with repetitions of the characters in the argument chars. If the argument chars has more than n bytes, the first n bytes of chars are used to fill the return string. If the argument chars has fewer than n bytes, the characters in chars are repeated until the return string has n bytes.

Usage

FillA replaces the functionality that Fill had in DBCS environments in PowerBuilder 9. In SBCS environments, Fill and FillA return the same results.

First expression function

Reports the value in the first row in the specified column.

Syntax

```
First ( column { FOR range { DISTINCT { expresn {, expres2 {, ... } } } } } )
```

- ***column*** — The column for which you want the value of the first row. Column can be a column name or a column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column.
- **FOR *range*** — (optional) The data that will be included when the value in the first row is found. Values for range depend on the presentation style. For most presentation styles, values for range are:
  - **ALL** – (Default) The value in the first of all rows in column.
  - **GROUP *n*** – The value in the first of rows in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** – The value in the first of the rows in column on a page.
  - **CROSSTAB** – (Crosstabs only) The value in the first of all rows in column in the crosstab.
  - **GRAPH** – (Graphs only) The value in the first row in column in the range specified for the Rows option
  - **OBJECT** – (OLE objects only) The value in the first row in column in the range specified for the Rows option
- **DISTINCT** — (optional) Causes First to consider only the distinct values in column when determining the first value. For a value of column, the first row found with the value is used and other rows that have the same value are ignored.
- ***expresn*** — (optional) One or more expressions that you want to evaluate to determine distinct rows. Expresn can be the name of a column, a function, or an expression.

Return Values

The datatype of the column. Returns the value in the first row of column. If you specify range, First returns the value of the first row in column in range.

Usage

If you specify range, First determines the value of the first row in column in range. If you specify DISTINCT, First returns the first distinct value in column, or if you specify expresn, the first distinct value in column where the value of expresn is distinct.

For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data property page (the Range property), and the aggregation function uses that range. Settings for Rows include the following:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

You cannot use this or other aggregate functions in validation rules or filter expressions. Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a DataWindow object always retrieves all rows.

Examples

This expression returns the first value in column 3 on the page:

```
First(#3 for page)
```

This expression returns the first distinct value in the column named dept_id in group 2:

```
First(dept_id for group 2 DISTINCT)
```

This expression returns the first value in the column named dept_id in group 2:

```
First(dept_id for group 2)
```

GetRow expression function

Reports the number of a row associated with a band in a DataWindow object.

Syntax

```
GetRow ( )
```

Return Values

Long. Returns the number of a row if it succeeds, 0 if no data has been retrieved or added, and –1 if an error occurs. Where you call GetRow determines what row it returns, as follows:

If the control in the DataWindow object is in this band	then GetRow returns:
Header	First row on the page
Group header	First row in the group
Detail	The row in which the expression occurs
Group trailer	Last row in the group
Summary	Last row in the DataWindow object
Footer	Last row on the page

Examples

This expression for a computed field in the detail band displays the number of each row:

```
GetRow()
```

This expression for a computed field in the header band checks to see if there is data. It returns the number of the first row on the page if there is data, and otherwise returns No Data:

```
If(GetRow()= 0, "No Data", String(GetRow()))
```

Hour expression function

Obtains the hour in a time value. The hour is based on a 24-hour clock.

Syntax

```
Hour ( time )
```

- ***time*** — The time value from which you want the hour

Return Values

Integer. Returns an integer (00–23) containing the hour portion of time.

Examples

This expression returns the current hour:

```
Hour(Now())
```

This expression returns 19:

```
Hour(19:01:31)
```

If expression function

Evaluates a condition and returns a value based on that condition.

Syntax

```
If ( boolean, truevalue, falsevalue )
```

- ***boolean*** — A boolean expression that evaluates to true or false.
- ***truevalue*** — The value you want returned if the boolean expression is true. The value can be a string or numeric value.
- ***falsevalue*** — The value you want returned if the boolean expression is false. The value can be a string or numeric value.

Return Values

The datatype of truevalue or falsevalue. Returns truevalue if boolean is true and falsevalue if it is false. Returns null if an error occurs.

Examples

This expression returns Boss if salary is over $100,000 and Employee if salary is less than or equal to $100,000:

```
If(salary > 100000, "Boss", "Employee")
```

This expression returns Boss if salary is over $100,000, Supervisor if salary is between $12,000 and $100,000, and Clerk if salary is less than or equal to $12,000:

```
If(salary > 100000, "Boss", If(salary > 12000, "Supervisor", "Clerk"))
```

In this example of a validation rule, the value the user should enter in the commission column depends on the price. If price is greater than or equal to 1000, then the commission is between .10 and .20. If price is less than 1000, then the commission must be between .04 and .09. The validation rule is:

```
(Number(GetText()) >= If(price >=1000, .10, .04)) AND
(Number(GetText()) <= If(price >= 1000, .20, .09))
```

The accompanying error message expression might be:

```
"Price is " + If(price >= 1000, "greater than or
equal to", "less than") + " 1000. Commission must be
between " + If(price >= 1000, ".10", ".04") + " and " + If(price >= 1000, ".20.", ".09.")
```

Int expression function

Gets the largest whole number less than or equal to a number.

Syntax

```
Int ( n )
```

- ***n*** — The number for which you want the largest whole number that is less than or equal to it

Return Values

The datatype of n. Returns the largest whole number less than or equal to n.

Examples

These expressions return 3.0:

```
Int(3.2)
Int(3.8)
```

These expressions return -4.0:

```
Int(-3.2)
Int(-3.8)
```

Integer expression function

Converts the value of a string to an integer.

Syntax

```
Integer ( string )
```

- ***string*** — The string you want returned as an integer

Return Values

Integer. Returns the contents of string as an integer if it succeeds and 0 if string is not a number.

Examples

This expression converts the string 24 to an integer:

```
Integer("24")
```

This expression for a computed field returns "Not a valid age" if age does not contain a number. The expression checks whether the Integer function returns 0, which means it failed to convert the value:

```
If (Integer(age) <> 0, age, "Not a valid age")
```

This expression returns 0:

```
Integer("3ABC") // 3ABC is not a number
```

This validation rule checks that the value in the column the user entered is less than 100:

```
Integer(GetText()) < 100
```

This validation rule for the column named age insures that age contains a string:

```
Integer(age) <> 0
```

IsDate expression function

Tests whether a string value is a valid date.

Syntax

```
IsDate ( datevalue )
```

- ***datevalue*** — A string whose value you want to test to determine whether it is a valid date

Return Values

Boolean. Returns true if datevalue is a valid date and false if it is not.

Examples

This expression returns true:

```
IsDate("Jan 1, 99")
```

This expression returns false:

```
IsDate("Jan 32, 2005")
```

This expression for a computed field returns a day number or 0. If the date_received column contains a valid date, the expression returns the number of the day in date_received in the computed field, and otherwise returns 0:

```
If(IsDate(String(date_received)),DayNumber(date_received), 0)
```

IsExpanded expression function

Tests whether a node in a TreeView DataWindow with the specified TreeView level and that includes the specified row is expanded.

Syntax

```
IsExpanded(long row, long level)
```

- ***row*** — The number of the row that belongs to the node
- ***level*** — The TreeView level of the node

Return Values

Returns true if the group is expanded and false otherwise.

Usage

A TreeView DataWindow has several TreeView level bands that can be expanded and collapsed. You can use the IsExpanded function to test whether or not a node in a TreeView DataWindow is expanded.

Examples

This expression returns true if the node that contains row 3 at TreeView level 2 is expanded:

```
IsExpanded(3,2)
```

IsNull expression function

Reports whether the value of a column or expression is null.

Syntax

```
IsNull ( any )
```

- ***any*** — A column or expression that you want to test to determine whether its value is null

Return Values

Boolean. Returns true if any is null and false if it is not.

Usage

Use IsNull to test whether a user-entered value or a value retrieved from the database is null.

Examples

This expression returns true if either a or b is null:

```
IsNull(a + b)
```

This expression returns true if the value in the salary column is null:

```
IsNull(salary)
```

This expression returns true if the value the user has entered is null:

```
IsNull(GetText())
```

IsNumber expression function

Reports whether the value of a string is a number.

Syntax

```
IsNumber ( string )
```

- ***string*** — A string whose value you want to test to determine whether it is a valid number

Return Values

Boolean. Returns true if string is a valid number and false if it is not.

Examples

This expression returns true:

```
IsNumber("32.65")
```

This expression returns false:

```
IsNumber("A16")
```

This expression for a computed field returns "Not a valid age" if age does not contain a number:

```
If(IsNumber(age), age, "Not a valid age")
```

To pass this validation rule, Age_nbr must be a number:

```
IsNumber(Age_nbr) = true
```

IsRowModified expression function

Reports whether the row has been modified.

Syntax

```
IsRowModified ( )
```

Return Values

Boolean. Returns true if the row has been modified and false if it has not.

Usage

In a DataWindow object, when you use IsRowModified in bands other than the detail band, it reports on a row in the detail band. See GetRow for a table specifying which row is associated with each band for reporting purposes.

Examples

This expression in a computed field in the detail area displays true or false to indicate whether each row has been modified:

```
IsRowModified()
```

This expression defined in the Properties view for the Color property of the computed field displays the text (true) in red if the user has modified any value in the row:

```
If(IsRowModified(), 255, 0)
```

IsRowNew expression function

Reports whether the row has been newly inserted.

Syntax

```
IsRowNew ( )
```

Return Values

Boolean. Returns true if the row is new and false if it was retrieved from the database.

Usage

In a DataWindow object, when you call IsRowNew in bands other than the detail band, it reports on a row in the detail band. See GetRow for a table specifying which row is associated with each band for reporting purposes.

Examples

This expression defined in the Properties view for the Protect property of a column prevents the user from modifying the column unless the row has been newly inserted:

```
If(IsRowNew(), 0, 1)
```

IsSelected expression function

Determines whether the row is selected. A selected row is highlighted using reverse video.

Syntax

```
IsSelected ( )
```

Return Values

Boolean. Returns true if the row is selected and false if it is not selected.

Usage

When you use IsSelected in bands other than the detail band, it reports on a row in the detail band. See GetRow for a table specifying which row is associated with each band for reporting purposes.

Examples

This expression for a computed field in the detail area displays a bitmap if the row is selected:

```
Bitmap(If(IsSelected(), "beach.bmp", ""))
```

This example allows the DataWindow object to display a salary total for all the selected rows. The expression for a computed field in the detail band returns the salary only when the row is selected so that another computed field in the summary band can add up all the selected salaries.

The expression for cf_selected_salary (the computed field in the detail band) is:

```
If(IsSelected(), salary, 0)
```

The expression for the computed field in the summary band is:

```
Sum(cf_selected_salary for all)
```

IsTime expression function

Reports whether the value of a string is a valid time value.

Syntax

```
IsTime ( timevalue )
```

- ***timevalue*** — A string whose value you want to test to determine whether it is a valid time

Return Values

Boolean. Returns true if timevalue is a valid time and false if it is not.

Examples

This expression returns true:

```
IsTime("8:00:00 am")
```

This expression returns false:

```
IsTime("25:00")
```

To pass this validation rule, the value in *start_time* must be a time:

```
IsTime(start_time)
```

Large expression function

Finds a large value at a specified ranking in a column (for example, third- largest, fifth-largest) and returns the value of another column or expression based on the result.

Syntax

```
Large ( returnexp, column, ntop { FOR range { DISTINCT { expres1 {, expres2 {, ... } } } } } )
```

- ***returnexp*** — The value you want returned when the large value is found. Returnexp includes a reference to a column, but not necessarily the column that is being evaluated for the largest value, so that a value is returned from the same row that contains the large value.
- ***column*** — The column that contains the large value you are searching for. Column can be a column name or a column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column. The datatype of column must be numeric.
- ***ntop*** — The ranking of the large value in relation to the column's largest value. For example, when ntop is 2, Large finds the second-largest value.
- **FOR *range*** — (optional) The data that will be included when the largest value is found. For most presentation styles, values for range are:
  - **ALL** — (Default) The largest of all values in column.
  - **GROUP *n*** — The largest of values in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** — The largest of the values in column on a page.
  - **CROSSTAB** — (Crosstabs only) The largest of all values in column in the crosstab.
  - **GRAPH** — (Graphs only) The largest of values in column in the range specified for the Rows option.
  - **OBJECT** — (OLE objects only) The largest of values in column in the range specified for the Rows option.
- **DISTINCT** — (optional) Causes Large to consider only the distinct values in column when determining the large value. For a value of column, the first row found with the value is used and other rows that have the same value are ignored.
- ***expresn*** — (optional) One or more expressions that you need to evaluate to determine distinct rows. Expresn can be the name of a column, a function, or an expression.

Return Values

The datatype of returnexp. Returns the ntop-largest value if it succeeds and –1 if an error occurs.

Usage

If you specify range, Large returns the value in returnexp when the value in column is the ntop-largest value in range. If you specify DISTINCT, Large returns returnexp when the value in column is the ntop-largest value of the distinct values in column, or if you specify expresn, the ntop-largest for each distinct value of expresn.

For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data property page (the Range property), and the aggregation function uses that range. Settings for Rows are as follows:

- For the Graph or OLE presentation style, Rows is always All
- For Graph controls, Rows can be All, Page, or Group
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies

Tip: If you do not need a return value from another column, and you want to find the largest value (ntop = 1), use Max(), it is faster.

You cannot use this or other aggregate functions in validation rules or filter expressions.

Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a DataWindow object always retrieves all rows.

Examples

These expressions return the names of the salespersons with the three largest sales (sum_sales is the sum of the sales for each salesperson) in group 2, which might be the salesregion group. Note that sum_sales contains the values being compared, but Large returns a value in the name column:

```
Large(name, sum_sales, 1 for group 2)
Large(name, sum_sales, 2 for group 2)
```

```
Large(name, sum_sales, 3 for group 2)
```

This example reports the salesperson with the third-largest sales, considering only the first entry for each person:

```
Large(name, sum_sales, 3 for all DISTINCT sum_sales)
```

Last expression function

Gets the value in the last row in the specified column.

Syntax

```
Last ( column { FOR range { DISTINCT { expres1 {, expres2 {, ... } } } } } )
```

- ***column*** — The column for which you want the value of the last row. Column can be a column name or a column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column.
- **FOR *range*** — (optional) The data that will be included when the value in the last row is found. For most presentation styles, values for range are:
  - **ALL** — (Default) The value in the last of all rows in column.
  - **GROUP *n*** — The value in the last row in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** — The value in the last row in column on a page.
  - **CROSSTAB** — (Crosstabs only) The value in the last row in column in the crosstab.
  - **GRAPH** — (Graphs only) The value in the last row in column in the range specified for the Rows option.
  - **OBJECT** — (OLE objects only) The value in the last row in column in the range specified for the Rows option.
- **DISTINCT** — (optional) Causes Last to consider only the distinct values in column when determining the last value. For a value of column, the first row found with the value is used and other rows that have the same value are ignored.
- ***expresn*** — (optional) One or more expressions that you want to evaluate to determine distinct rows. Expresn can be the name of a column, a function, or an expression.

Return Values

The datatype of the column. Returns the value in the last row of column. If you specify range, Last returns the value of the last row in column in range.

Usage

If you specify range, Last determines the value of the last row in column in range. If you specify DISTINCT, Last returns the last distinct value in column, or if you specify expresn, the last distinct value in column where the value of expresn is distinct.

For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data property page (the Range property), and the aggregation function uses that range. Settings for Rows include the following:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

You cannot use this or other aggregate functions in validation rules or filter expressions.

Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a DataWindow object always retrieves all rows.

Examples

This expression returns the last distinct value in the column named dept_id in group 2:

```
Last(dept_id for group 2 DISTINCT)
```

This expression returns the last value in the column named emp_id in group 2:

```
Last(emp_id for group 2)
```

LastPos expression function

Finds the last position of a target string in a source string.

Syntax

```
LastPos ( string1, string2, searchlength )
```

- ***string1*** — The string in which you want to find string2.
- **string2** — The string you want to find in string1.
- ***searchlength*** — (optional) A long that limits the search to the leftmost *searchlength* characters of the source string *string1*. The default is the entire string.

Return Values

Long. Returns a long whose value is the starting position of the last occurrence of *string2* in *string1* within the characters specified in *searchlength*. If *string2* is not found in *string1*, or if *searchlength* is 0, LastPos returns 0. If any argument's value is null, LastPos returns null.

Usage

The LastPos function is case sensitive. The entire target string must be found in the source string.

Examples

This statement returns 8, because the position of the last occurrence of HI is position 8:

```
LastPos("CASTLE HILLS", "HI")
```

This statement returns 11:

```
LastPos("CASTLE HILLS", "L")
```

This statement returns 0, because the case does not match:

```
LastPos("CASTLE HILLS", "hi")
```

This statement searches the leftmost 6 characters and returns 0, because the only occurrence of HILL is after position 6:

```
LastPos("CASTLE HILLS", "HILL", 6)
```

Left expression function

Obtains a specified number of characters from the beginning of a string.

Syntax

```
Left ( string, n )
```

- ***string*** — The string containing the characters you want
- ***n*** — A long specifying the number of characters you want

Return Values

String. Returns the leftmost n characters in string if it succeeds and the empty string ("") if an error occurs.

If n is greater than or equal to the length of the string, Left returns the entire string. It does not add spaces to make the return value's length equal to n.

Examples

This expression returns CAST:

```
Left("CASTLE HILLS", 4)
```

This expression returns CASTLE HILLS:

```
Left("CASTLE HILLS", 40)
```

This expression for a computed field returns the first 40 characters of the text in the column home_address:

```
Left(home_address, 40)
```

LeftA expression function

Obtains a specified number of bytes from the beginning of a string.

Syntax

```
LeftA ( string, n )
```

- ***string*** — The string containing the characters you want
- ***n*** — A long specifying the number of bytes you want

Return Values

String. Returns the characters in the leftmost n bytes in string if it succeeds and the empty string ("") if an error occurs.

If n is greater than or equal to the length of the string, LeftA returns the entire string. It does not add spaces to make the return value's length equal to n.

Usage

LeftA replaces the functionality that Left had in DBCS environments in PowerBuilder 9. In SBCS environments, Left and LeftA return the same results.

LeftTrim expression function

Removes spaces from the beginning of a string.

Syntax

```
LeftTrim ( string )
```

- ***string*** — The string you want returned with leading spaces deleted

Return Values

String. Returns a copy of string with leading spaces deleted if it succeeds and the empty string ("") if an error occurs.

Examples

This expression returns CASTLE:

```
LeftTrim(" CASTLE")
```

This expression for a computed field deletes any leading blanks from the value in the column lname and returns the value preceded by the salutation specified in salut_emp:

```
salut_emp + " " + LeftTrim(lname)
```

Len expression function

Reports the length of a string in characters.

Syntax

```
Len ( string )
```

- ***string*** — The string for which you want the length

Return Values

Long. Returns a long containing the length of string in characters if it succeeds and –1 if an error occurs.

Examples

This expression returns 0:

```
Len("")
```

This validation rule tests that the value the user entered is fewer than 20 characters:

```
Len(GetText()) < 20
```

LenA expression function

Reports the length of a string in bytes.

Syntax

```
LenA ( string )
```

- ***string*** — The string for which you want the length

Return Values

Long. Returns a long containing the length of string in bytes if it succeeds and –1 if an error occurs.

Usage

LenA replaces the functionality that Len had in DBCS environments in PowerBuilder 9. In SBCS environments, Len and LenA return the same results.

Log expression function

Gets the natural logarithm of a number. The inverse of the Log function is the Exp function.

Syntax

```
Log ( n )
```

- ***n*** — The number for which you want the natural logarithm (base e). The value of n must be greater than 0.

Return Values

Double. Returns the natural logarithm of n. An execution error occurs if n is negative or zero.

Examples

This expression returns 2.302585092:

```
Log(10)
```

This expression returns -.693147 ... :

```
Log(0.5)
```

Both these expressions result in an error at runtime:

```
Log(0)
Log(-2)
```

LogTen expression function

Gets the base 10 logarithm of a number. The expression 10^n is the inverse for LogTen(n). To obtain n given number (nbr = LogTen(n)), use n = 10^nbr.

Syntax

```
LogTen ( n )
```

- ***n*** — The number for which you want the base 10 logarithm. The value of n must not be negative.

Return Values

Double. Returns the base 10 logarithm.

Examples

This expression returns 1:

```
LogTen(10)
```

The following expressions both return 0:

```
LogTen(1)
LogTen(0)
```

This expression results in an execution error:

```
LogTen(-2)
```

Long expression function

Converts the value of a string to a long.

Syntax

```
Long ( string )
```

- ***string*** — The string you want returned as a long

Return Values

Long. Returns the contents of string as a long if it succeeds and 0 if string is not a valid number.

Examples

This expression returns 2167899876 as a long:

```
Long("2167899876")
```

LookUpDisplay expression function

Obtains the display value in the code table associated with the data value in the specified column.

Syntax

```
LookUpDisplay ( column )
```

- ***column*** — The column for which you want the code table display value

Return Values

String. Returns the display value when it succeeds and the empty string ("") if an error occurs.

Usage

If a column has a code table, a buffer stores a value from the data column of the code table, but the user sees a value from the display column. Use LookUpDisplay to get the value the user sees.

When a column that is displayed in a graph has a code table, the graph displays the data values of the code table by default. To display the display values, call this function when you define the graph data.

Examples

This expression returns the display value for the column unit_measure:

```
LookUpDisplay(unit_measure)
```

Assume the column product_type has a code table and you want to use it as a category for a graph. To display the product type descriptions instead of the data values in the categories, enter this expression in the Category option on the Data page in the graph's property sheet:

```
LookUpDisplay(product_type)
```

Lower expression function

Converts all the characters in a string to lowercase.

Syntax

```
Lower ( string )
```

- ***string*** — The string you want to convert to lowercase letters

Return Values

String. Returns string with uppercase letters changed to lowercase if it succeeds and the empty string ("") if an error occurs.

Examples

This expression returns castle hill:

```
Lower("Castle Hill")
```

Match expression function

Determines whether a string's value contains a particular pattern of characters.

Syntax

```
Match ( string, textpattern )
```

- ***string*** — The string in which you want to look for a pattern of characters
- ***textpattern*** — A string whose value is the text pattern

Return Values

Boolean. Returns true if string matches textpattern and false if it does not. Match also returns false if either argument has not been assigned a value or the pattern is invalid.

Usage

Match enables you to evaluate whether a string contains a general pattern of characters. To find out whether a string contains a specific substring, use the Pos function.

Textpattern is similar to a regular expression. It consists of metacharacters, which have special meaning, and ordinary characters, which match themselves. You can specify that the string begin or end with one or more characters from a set, or that it contain any characters except those in a set.

A text pattern consists of metacharacters, which have special meaning in the match string, and nonmetacharacters, which match the characters themselves.

The following tables explain the meaning and use of these metacharacters:

Metacharacter	Meaning	Example
Caret (^)	Matches the beginning of a string	^C matches C at the beginning of a string.
Dollar sign ($)	Matches the end of a string	s$ matches s at the end of a string.
Period (.)	Matches any character	. . . matches three consecutive characters.
Backslash (\)	Removes the following metacharacter's special characteristics so that it matches itself	\$ matches $.
Character class (a group of characters enclosed in square brackets [ ])	Matches any of the enclosed characters	[AEIOU] matches A, E, I, O, or U. You can use hyphens to abbreviate ranges of characters in a character class. For example, [A-Za-z] matches any letter.
Complemented character class (first character inside the square brackets is a caret)	Matches any character *not* in the group following the caret	[^0-9] matches any character except a digit, and [^A-Za-z] matches any character except a letter.

The metacharacters asterisk (*), plus (+), and question mark (?) are unary operators that are used to specify repetitions in a regular expression:

Metacharacter	Meaning	Example
* (asterisk)	Indicates zero or more occurrences	A* matches zero or more As (no As, A, AA, AAA, and so on)
+ (plus)	Indicates one or more occurrences	A+ matches one A or more than one A (A, AAA, and so on)
? (question mark)	Indicates zero or one occurrence	A? matches an empty string ("") or A

Sample patterns

These text patterns match the following sample text:

- `AB` — Any string that contains AB, such as ABA, DEABC, graphAB_one.
- `B*` — Any string that contains 0 or more Bs, such as AC, B, BB, BBB, ABBBC, and so on. Since B* used alone matches any string, you would not use it alone, but notice its use in some the following examples.
- `AB*C` — Any string containing the pattern AC or ABC or ABBC, and so on (0 or more Bs).
- `AB+C` — Any string containing the pattern ABC or ABBC or ABBBC, and so on (1 or more Bs).
- `ABB*C` — Any string containing the pattern ABC or ABBC or ABBBC, and so on (1 B plus 0 or more Bs).
- `^AB` — Any string starting with AB.
- `AB?C` — Any string containing the pattern AC or ABC (0 or 1 B).
- `^[ABC]` — Any string starting with A, B, or C.
- `[^ABC]` — A string containing any characters other than A, B, or C.
- `^[^abc]` — A string that begins with any character except a, b, or c.
- `^[^a-z]$` — Any single-character string that is not a lowercase letter (^ and $ indicate the beginning and end of the string).
- `[A-Z]+` — Any string with one or more uppercase letters.
- `^[0-9]+$` — Any string consisting only of digits.
- `^[0-9][0-9][0-9]$` — Any string consisting of exactly three digits.
- `^([0-9][0-9][0-9])$` — Any string consisting of exactly three digits enclosed in parentheses.

Examples

This validation rule checks that the value the user entered begins with an uppercase letter. If the value of the expression is false, the data fails validation:

```
Match(GetText(), "^[A-Z]")
```

Max expression function

Gets the maximum value in the specified column.

Syntax

```
Max ( column { FOR range { DISTINCT { expres1 {, expres2 {, ... } } } } } )
```

- ***column*** — The column for which you want the maximum value. Column can be the column name or the column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column. The datatype of column must be numeric.
- **FOR *range*** — (optional) The data that will be included when the maximum value is found. For most presentation styles, values for range are:
  - **ALL** — (Default) The maximum value of all rows in column.
  - **GROUP *n*** — The maximum value of rows in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** — The maximum value of the rows in column on a page.
  - **CROSSTAB** — (Crosstabs only) The maximum value of all rows in column in the crosstab.
  - **GRAPH** — (Graphs only) The maximum value in column in the range specified for the Rows option.
  - **OBJECT** — (OLE objects only) The maximum value in column in the range specified for the Rows option.
- **DISTINCT** — (optional) Causes Max to consider only the distinct values in column when determining the largest value. For a value of column, the first row found with the value is used and other rows that have the same value are ignored.
- ***expresn*** — (optional) One or more expressions that you want to evaluate to determine distinct rows. Expresn can be the name of a column, a function, or an expression.

Return Values

The datatype of the column. Returns the maximum value in the rows of column. If you specify range, Max returns the maximum value in column in range.

Usage

If you specify range, Max determines the maximum value in column in range. If you specify DISTINCT, Max returns the maximum distinct value in column, or if you specify expresn, the maximum distinct value in column where the value of expresn is distinct.

For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data property page (the Range property), and the aggregation function uses that range. Settings for Rows include the following:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

Null values are ignored and are not considered in determining the maximum.

You cannot use this or other aggregate functions in validation rules or filter expressions.

Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a DataWindow object always retrieves all rows.

Examples

This expression returns the maximum of the values in the age column on the page:

```
Max(age for page)
```

This expression returns the maximum of the values in column 3 on the page:

```
Max(#3 for page)
```

This expression returns the maximum of the values in the column named age in group 1:

```
Max(age for group 1)
```

Assuming a DataWindow object displays the order number, amount, and line items for each order, this computed field returns the maximum of the order amount for the distinct order numbers:

```
Max(order_amt for all DISTINCT order_nbr)
```

Median expression function

Calculates the median of the values of the column. The median is the middle value in the set of values, for which there is an equal number of values greater and smaller than it.

Syntax

```
Median ( column { FOR range { DISTINCT { expres1 {, expres2 {, ... } } } } } )
```

- ***column*** — The column for which you want the median of the data values. Column can be the column name or the column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column. The datatype of column must be numeric.
- **FOR *range*** — (optional) The data that will be included in the median. For most presentation styles, values for range are:

  - **ALL** — (Default) The median of all values in column.
  - **GROUP *n*** — The median of values in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** — The median of the values in column on a page.
  - **CROSSTAB** — (Crosstabs only) The median of all values in column in the crosstab.
  - **GRAPH** — (Graphs only) The median of values in column in the range specified for the Rows.
  - **OBJECT** — (OLE objects only) The median of values in column in the range specified for the Rows option.
- **DISTINCT** — (optional) Causes Median to consider only the distinct values in column when determining the median. For a value of column, the first row found with the value is used and other rows that have the same value are ignored.
- **expresn** — (optional) One or more expressions that you want to evaluate to determine distinct rows. Expresn can be the name of a column, a function, or an expression.

Return Values

The numeric datatype of the column. Returns the median of the values of the rows in range if it succeeds and –1 if an error occurs.

Usage

If you specify range, Median returns the median value of column in range. If you specify DISTINCT, Median returns the median value of the distinct values in column, or if you specify expresn, the median of column for each distinct value of expresn.

For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data property page (the Range property), and the aggregation function uses that range.

Settings for Rows include the following:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

In calculating the median, null values are ignored.

You cannot use this or other aggregate functions in validation rules or filter expressions.

Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a DataWindow object always retrieves all rows.

Examples

This expression returns the median of the values in the column named salary:

```
Median(salary)
```

This expression returns the median of the values in the column named salary of group 1:

```
Median(salary for group 1)
```

This expression returns the median of the values in column 5 on the current page:

```
Median(#5 for page)
```

This computed field returns Above Median if the median salary for the page is greater than the median for the report:

```
If(Median(salary for page) > Median(salary), "Above Median", " ")
```

This expression for a graph value sets the data value to the median value of the sale_price column:

```
Median(sale_price)
```

This expression for a graph value entered on the Data page in the graph's property sheet sets the data value to the median value of the sale_price column for the entire graph:

```
Median(sale_price for graph)
```

Assuming a DataWindow object displays the order number, amount, and line items for each order, this computed field returns the median of the order amount for the distinct order numbers:

```
Median(order_amt for all DISTINCT order_nbr)
```

Mid expression function

Obtains a specified number of characters from a specified position in a string.

Syntax

```
Mid ( string, start {, length } )
```

- ***string*** — The string from which you want characters returned.
- ***start*** — A long specifying the position of the first character you want returned (the position of the first character of the string is 1).
- ***length*** — (optional) A long whose value is the number of characters you want returned. If you do not enter length or if length is greater than the number of characters to the right of start, Mid returns the remaining characters in the string.

Return Values

String. Returns characters specified in length of string starting at character start. If start is greater than the number of characters in string, the Mid function returns the empty string (""). If length is greater than the number of characters remaining after the start character, Mid returns the remaining characters. The return string is not filled with spaces to make it the specified length.

Examples

This expression returns "":

```
Mid("CASTLE HILLS", 40, 5)
```

This expression returns "LE HILLS":

```
Mid("CASTLE HILLS", 5)
```

This expression in a computed field returns ACCESS DENIED if the fourth character in the column password is not R:

```
If(Mid(password, 4, 1) = "R", "ENTER", "ACCESS DENIED")
```

To pass this validation rule, the fourth character in the column password must be 6:

```
Mid(password, 4, 1) = "6"
```

MidA expression function

Obtains a specified number of bytes from a specified position in a string.

Syntax

```
MidA ( string, start {, length } )
```

- ***string*** — The string from which you want characters returned.
- ***start*** — A long specifying the position of the first byte you want returned (the position of the first byte of the string is 1).
- ***length*** — (optional) A long whose value is the number of bytes you want returned. If you do not enter length or if length is greater than the number of bytes to the right of start, MidA returns the remaining bytes in the string.

Return Values

String. Returns characters specified by the number of bytes in length of string starting at the byte specified by start. If start is greater than the number of bytes in string, the MidA function returns the empty string (""). If length is greater than the number of bytes remaining after the start byte, MidA returns the remaining bytes. The return string is not filled with spaces to make it the specified length.

Usage

MidA replaces the functionality that Mid had in DBCS environments in PowerBuilder 9. In SBCS environments, Mid and MidA return the same results.

Min expression function

Gets the minimum value in the specified column.

Syntax

```
Min ( column { FOR range { DISTINCT { expres1 {, expres2 {, ... } } } } } )
```

- ***column*** — The column for which you want the minimum value. Column can be the column name or the column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column. The datatype of column must be numeric.
- **FOR *range*** — (optional) The data that will be included in the minimum. For most presentation styles, values for range are:
  - **ALL** — (Default) The minimum of all values in column.
  - **GROUP n** — The minimum of values in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** — The minimum of the values in column on a page.
  - **CROSSTAB** — (Crosstabs only) The minimum of all values in column in the crosstab.
  - **GRAPH** — (Graphs only) The minimum of values in column in the range specified for the Rows option.
  - **OBJECT** — (OLE objects only) The minimum of values in column in the range specified for the Rows option.
- **DISTINCT** — (optional) Causes Min to consider only the distinct values in column when determining the minimum value. For a value of column, the first row found with the value is used and other rows that have the same value are ignored.
- ***expresn*** — (optional) One or more expressions that you want to evaluate to determine distinct rows. Expresn can be the name of a column, a function, or an expression.

Return Values

The datatype of the column. Returns the minimum value in the rows of column. If you specify range, Min returns the minimum value in the rows of column in range.

Usage

If you specify range, Min determines the minimum value in column in range. If you specify DISTINCT, Min returns the minimum distinct value in column, or if you specify expresn, the minimum distinct value in column where the value of expresn is distinct.

For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data property page (the Range property), and the aggregation function uses that range. Settings for Rows include:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

Null values are ignored and are not considered in determining the minimum.

You cannot use this or other aggregate functions in validation rules or filter expressions.

Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a DataWindow object always retrieves all rows.

Examples

This expression returns the minimum value in the column named age in group 2:

```
Min(age for group 2)
```

This expression returns the minimum of the values in column 3 on the page:

```
Min(#3 for page)
```

Assuming a DataWindow object displays the order number, amount, and line items for each order, this computed field returns the minimum of the order amount for the distinct order numbers:

```
Min(order_amt for all DISTINCT order_nbr)
```

Minute expression function

Obtains the number of minutes in the minutes portion of a time value.

Syntax

```
Minute ( time )
```

- ***time*** — The time value from which you want the minutes.

Return Values

Integer. Returns the minutes portion of time (00 to 59).

Examples

This expression returns 1:

```
Minute(19:01:31)
```

Mod expression function

Obtains the remainder (modulus) of a division operation.

Syntax

```
Mod ( x, y )
```

- ***x*** — The number you want to divide by y
- ***y*** — The number you want to divide into x

Return Values

The datatype of x or y, whichever datatype is more precise.

Examples

This expression returns 2:

```
Mod(20, 6)
```

This expression returns 1.5:

```
Mod(25.5, 4)
```

This expression returns 2.5:

```
Mod(25, 4.5)
```

Mode expression function

Calculates the mode of the values of the column. The mode is the most frequently occurring value.

Syntax

```
Mode ( column { FOR range { DISTINCT { expres1 {, expres2 {, ... } } } } } )
```

- ***column*** — The column for which you want the mode of the data values. Column can be the column name or the column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column. The datatype of column must be numeric.
- **FOR *range*** — (optional) The data that will be included in the mode. For most presentation styles, values for range are:
  - **ALL** — (Default) The mode of all values in column.
  - **GROUP *n*** — The mode of values in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** — The mode of the values in column on a page.
  - **CROSSTAB** — (Crosstabs only) The mode of all values in column in the crosstab.
  - **GRAPH** — (Graphs only) The mode of values in column in the range specified for the Rows option.
  - **OBJECT** — (OLE objects only) The mode of values in column in the range specified for the Rows option.
- **DISTINCT** — (optional) Causes Mode to consider only the distinct values in column when determining the mode. For a value of column, the first row found with the value is used and other rows that have the same value are ignored.
- ***expresn*** — (optional) One or more expressions that you want to evaluate to determine distinct rows. Expresn can be the name of a column, a function, or an expression.

Return Values

The numeric datatype of the column. Returns the mode of the values of the rows in range if it succeeds and –1 if an error occurs.

Usage

If you specify range, Mode returns the mode of column in range. If you specify DISTINCT, Mode returns the mode of the distinct values in column, or if you specify expresn, the mode of column for each distinct value of expresn.

For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data property page (the Range property), and the aggregation function uses that range. Settings for Rows include:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

In calculating the mode, null values are ignored.

You cannot use this or other aggregate functions in validation rules or filter expressions.

Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a DataWindow object always retrieves all rows.

Examples

This expression returns the mode of the values in the column named salary:

```
Mode(salary)
```

This expression returns the mode of the values for group 1 in the column named salary:

```
Mode(salary for group 1)
```

This expression returns the mode of the values in column 5 on the current page:

```
Mode(#5 for page)
```

This computed field returns Above Mode if the mode of the salary for the page is greater than the mode for the report:

```
If(Mode(salary for page) > Mode(salary), "Above Mode", " ")
```

This expression for a graph value sets the data value to the mode of the sale_price column:

```
Mode(sale_price)
```

This expression for a graph value entered on the Data page in the graph's property sheet sets the data value to the mode of the sale_price column for the entire graph:

```
Mode(sale_price for graph)
```

Assuming a DataWindow object displays the order number, amount, and line items for each order, this computed field returns the mode of the order amount for the distinct order numbers:

```
Mode(order_amt for all DISTINCT order_nbr)
```

Month expression function

Gets the month of a date value.

Syntax

```
Month ( date )
```

- ***date*** — The date from which you want the month.

Return Values

Integer. Returns an integer (1 to 12) whose value is the month portion of date.

Examples

This expression returns 1:

```
Month(2005-01-31)
```

This expression for a computed column returns Wrong Month if the month in the column expected_grad_date is not 6:

```
If(Month(expected_grad_date) = 6, "June", "Wrong Month")
```

This validation rule expression checks that the value of the month in the date in the column expected_grad_date is 6:

```
Month(expected_grad_date) = 6
```

Now expression function

Obtains the current time based on the system time of the client machine.

Syntax

```
Now ( )
```

Return Values

Time. Returns the current time based on the system time of the client machine.

Usage

Use Now to compare a time to the system time or to display the system time on the screen. The timer interval specified for the DataWindow object determines the frequency at which the value of Now is updated. For example, if the timer interval is one second, it is updated every second. The default timer interval is one minute (60,000 milliseconds).

Examples

This expression returns the current system time:

```
Now()
```

This expression sets the column value to 8:00 when the current system time is before 8:00 and to the current time if it is after 8:00:

```
If(Now() < 08:00:00, '08:00:00', String(Now()))
```

The displayed time refreshes every time the specified time interval period elapses.

If a static value of time is required (for example, the time when a report has been executed or the retrieve has started), you can use a static text field that you modify as follows:

```
//Set the time when the report was executed in
//the text field t_now
dw1.Modify("t_now.text='"+ String(Now(),"hh:mm")+"'")
//execute the report
dw1.retrieve()
```

Number expression function

Converts a string to a number.

Syntax

```
Number ( string )
```

- ***string*** — The string you want returned as a number

Return Values

A numeric datatype. Returns the contents of string as a number. If string is not a valid number, Number returns 0.

Examples

This expression converts the string 24 to a number:

```
Number("24")
```

This expression for a computed field tests whether the value in the age column is greater than 55 and if so displays N/A; otherwise, it displays the value in age:

```
If(Number(age) > 55, "N/A", age)
```

This validation rule checks that the number the user entered is between 25,000 and 50,000:

```
Number(GetText())>25000 AND Number (GetText())<50000
```

Page expression function

Gets the number of the current page.

Syntax

```
Page ( )
```

Return Values

Long. Returns the number of the current page.

Usage

The vertical size of the paper less the top and bottom margins is used to calculate the page count. When the print orientation is landscape, the vertical size of the paper is the shorter dimension. If the DataWindow object is not set to print preview, then the size of the control determines the page number. When Page() is in the header, it uses the first row currently visible on the page to determine the page number. When it is in the footer, it uses the last row currently visible. Therefore, it is possible for the the values to be different.

Examples

This expression returns the number of the current page:

```
Page()
```

In the DataWindow object's footer band, this expression for a computed field displays a string showing the current page number and the total number of pages in the report. The result has the format "Page *n* of *total*":

```
'Page ' + Page() + ' of ' + PageCount()
```

PageAbs expression function

Gets the absolute number of the current page.

Syntax

```
PageAbs ( )
```

Return Values

Long. Returns the absolute number of the current page.

Usage

Use this function for group reports that have ResetPageCount = yes. It returns the absolute page number, ignoring the page reset count. This enables you to number the grouped pages, but also to obtain the absolute page when the user wants to print the current page, regardless of what that page number is in a grouped page report.

Examples

This expression returns the absolute number of the current page:

```
PageAbs()
```

This example obtains the absolute page number for the first row on the page in the string variable *ret*:

```
string ret, row
row = dw1.Object.DataWindow.FirstRowOnPage
ret = dw1.Describe("Evaluate('pageabs()', "+row+")")
```

PageAcross expression function

Gets the number of the current horizontal page. For example, if a report is twice the width of the print preview window and the window is scrolled horizontally to display the portion of the report that was outside the preview, PageAcross returns 2 because the current page is the second horizontal page.

Syntax

```
PageAcross ( )
```

Return Values

Long. Returns the number of the current horizontal page if it succeeds and –1 if an error occurs.

Examples

This expression returns the number of the current horizontal page:

```
PageAcross()
```

PageCount expression function

Gets the total number of pages when a DataWindow object is being viewed in Print Preview. This number is also the number of printed pages if the DataWindow object is not wider than the preview window. If the DataWindow object is wider than the preview window, the number of printed pages will be greater than the number PageCount gets.

Syntax

```
PageCount ( )
```

Return Values

Long. Returns the total number of pages.

Usage

PageCount applies to Print Preview. The vertical size of the paper less the top and bottom margins is used to calculate the page count. When the print orientation is landscape, the vertical size of the paper is the shorter dimension. If the DataWindow object is not set to print preview, then the size of the control determines the page count.

Examples

This expression returns the number of pages:

```
PageCount()
```

In the DataWindow object's footer band, this expression for a computed field displays a string showing the current page number and the total number of pages in the report. The result has the format "Page *n* of *total*":

```
'Page ' + Page() + ' of ' + PageCount()
```

PageCountAcross expression function

Gets the total number of horizontal pages that are wider than the Print Preview window when a DataWindow object is viewed in Print preview.

Syntax

```
PageCountAcross ( )
```

Return Values

Long. Returns the total number of horizontal pages if it succeeds and –1 if an error occurs.

Usage

PageCountAcross applies to Print Preview.

Examples

This expression returns the number of horizontal pages in the Print Preview window:

```
PageCountAcross()
```

Paint expression function

Takes a string expression argument and returns the same string, allowing you to paint inside a DataWindow object in a way that respect the position and z-order of other DataWindow objects.

Syntax

```
Paint ( expr )
```

- ***expr*** — Any valid DataWindow expression. It should contain a function call to a drawing global function with rendering logic. If expr is a string expression and the value is not null, the computed field will render the evaluated string expression.

Return Values

String. The Paint expression function takes a string expression argument and returns the same string.

Examples

This example instantiates the drawing functions and, if the drawing function returns false, the text "No Pie" displays.

```
Paint
(
    MyDrawPieSlice
    (
        GetPaintDC()
        GetPaintRectX()
        GetPaintRectY()
        GetPaintRectWidth()
        GetPaintRectHeight()
        GetRow()*100/RowCount()
    )
)
Paint
(
    MyDrawPieSlice
    (
        GetPaintDC(),
    GetRow()*100/RowCount()
    )
)
Paint
(
    if MyDrawPieSlice(GetPaintDC()),"","No Pie")
```

```
)
```

Percent expression function

Gets the percentage that the current value represents of the total of the values in the column.

Syntax

```
Percent ( column { FOR range { DISTINCT { expres1 {, expres2 {, ... } } } } } )
```

- ***column*** — The column for which you want the value of each row expressed as a percentage of the total of the values of the column. Column can be the column name or the column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column. The datatype of column must be numeric.
- **FOR *range*** — (optional) The data to be included in the percentage. For most presentation styles, values for range are:
  - **ALL** — (Default) The percentage that the current value represents of all rows in column.
  - **GROUP *n*** — The percentage that the current value represents of rows in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** — The percentage that the current value represents of the rows in column on a page.
  - **CROSSTAB** — (Crosstabs only) The percentage that the current value represents of all rows in column in the crosstab.
  - **GRAPH** — (Graphs only) The percentage that the current value represents of values in column in the range specified for the Rows option.
  - **OBJECT** — (OLE objects only) The percentage that the current value represents of values in column in the range specified for the Rows option.
- **DISTINCT** — (optional) Causes Percent to consider only the distinct values in column when determining the percentage. For a value of column, the first row found with the value is used and other rows that have the same value are ignored.
- ***expresn*** — (optional) One or more expressions that you want to evaluate to determine distinct rows. Expresn can be the name of a column, a function, or an expression.

Return Values

A numeric datatype (decimal, double, integer, long, or real). Returns the percentage the current row of column represents of the total value of the column.

Usage

Usually you use Percent in a column to display the percentage for each row. You can also use Percent in a header or trailer for a group. In the header, Percent displays the percentage for the first value in the group, and in the trailer, for the last value in the group.

If you specify range, Percent returns the percentage that the current row of column represents relative to the total value of range. For example, if column 5 is salary, Percent(#5 for group 1) is equivalent to salary/(Sum(Salary for group 1)).

If you specify DISTINCT, Percent returns the percent that a distinct value in column represents of the total value of column. If you specify expresn, Percent returns the percent that the value in column represents of the total for column in a row in which the value of expresn is distinct.

The percentage is displayed as a decimal value unless you specify a format for the result. A display format can be part of the computed field's definition.

For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data property page (the Range property), and the aggregation function uses that range. Settings for Rows include the following:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

Null values are ignored and are not considered in the calculation.

You cannot use Percent or other aggregate functions in validation rules or filter expressions. Percent does not work for crosstabs; specifying "for crosstab" as a range is not available for Percent.

Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a DataWindow object always retrieves all rows.

Examples

This expression returns the value of each row in the column named salary as a percentage of the total of salary:

```
Percent(salary)
```

This expression returns the value of each row in the column named cost as a percentage of the total of cost in group 2:

```
Percent(cost for group 2)
```

This expression entered in the Value box on the Data tab page in the Graph Object property sheet returns the value of each row in the qty_ordered as a percentage of the total for the column in the graph:

```
Percent(qty_ordered for graph)
```

Assuming a DataWindow object displays the order number, amount, and line items for each order, this computed field returns the order amount as a percentage of the total order amount for the distinct order numbers:

```
Percent(order_amt for all DISTINCT order_nbr)
```

Pi expression function

Multiplies pi by a specified number.

Syntax

```
Pi ( n )
```

- ***n*** — The number you want to multiply by pi (3.14159265358979323...)

Return Values

Double. Returns the result of multiplying n by pi if it succeeds and –1 if an error occurs.

Usage

Use Pi to convert angles to and from radians.

Examples

This expression returns pi:

```
Pi(1)
```

Both these expressions return the area of a circle with the radius Rad:

```
Pi(1) * Rad^2
```

```
Pi(Rad^2)
```

This expression computes the cosine of a 45-degree angle:

```
Cos(45.0 * (Pi(2)/360))
```

Pos expression function

Finds one string within another string.

Syntax

```
Pos ( string1, string2 {, start } )
```

- ***string1*** — The string in which you want to find string2.
- ***string2*** — The string you want to find in string1.
- ***start*** — (optional) A long indicating where the search will begin in string. The default is 1.

Return Values

Long. Returns a long whose value is the starting position of the first occurrence of string2 in string1 after the position specified in start. If string2 is not found in string1 or if start is not within string1, Pos returns 0.

Usage

The Pos function is case sensitive.

Examples

This expression returns the position of the letter a in the value of the last_name column:

```
Pos(last_name, "a")
```

This expression returns 8:

```
Pos("CASTLE HILLS", "HI")
```

This expression returns 3:

```
Pos("CASTLE HILLS", "S")
```

This expression returns 0 (because the case does not match):

```
Pos("CASTLE HILLS", "hi")
```

This expression returns 0 (because it starts searching at position 6, after the occurrence of CA):

```
Pos("CASTLE HILLS", "CA", 6)
```

PosA expression function

Finds one string within another string.

Syntax

```
PosA ( string1, string2 {, start } )
```

- ***string1*** — The string in which you want to find string2.
- ***string2*** — The string you want to find in string1.
- ***start*** — (optional) A long indicating the position in bytes where the search will begin in string. The default is 1.

Return Values

Long. Returns a long whose value is the starting position of the first occurrence of string2 in string1 after the position in bytes specified in start. If string2 is not found in string1 or if start is not within string1, PosA returns 0.

Usage

PosA replaces the functionality that Pos had in DBCS environments in PowerBuilder 9. In SBCS environments, Pos and PosA return the same results.

ProfileInt expression function

Obtains the integer value of a setting in the specified profile file.

Syntax

```
ProfileInt ( filename, section, key, default )
```

- ***filename*** — A string whose value is the name of the profile file. If you do not specify a full path, ProfileInt uses the operating system's standard file search order to find the file.
- ***section*** — A string whose value is the name of a group of related values in the profile file. In the file, section names are in square brackets. Do not include the brackets in section. Section is not case sensitive.
- ***key*** — A string specifying the setting name in section whose value you want. The setting name is followed by an equal sign in the file. Do not include the equal sign in key. Key is not case sensitive.
- ***default*** — An integer value that ProfileInt returns if filename is not found, if section or key does not exist in filename, or if the value of key cannot be converted to an integer.

Return Values

Integer. Returns default if filename is not found, section is not found in filename, key is not found in section, or the value of key is not an integer. Returns –1 if an error occurs.

Usage

Use ProfileInt and ProfileString to get configuration settings from a profile file you have designed for your application. ProfileInt and ProfileString can read files with ANSI or UTF16-LE encoding on Windows systems, and ANSI or UTF16–BE encoding on UNIX systems.

In PowerBuilder, you can use PowerScript SetProfileString to change values in the profile file to customize your application's configuration at runtime. Before you make changes, you can use ProfileInt and ProfileString to obtain the original settings so you can optionally restore them when the user exits the application.

Examples

This example uses the following PROFILE.INI file:

```
[MyApp]
Maximized=1
[Security]
Class = 7
```

This expression tries to return the integer value of the keyword Minimized in section MyApp of file C:\PROFILE.INI. It returns 3 if there is no MyApp section or no Minimized keyword in the MyApp section. Based on the sample file above, it returns 3:

```
ProfileInt("C:\PROFILE.INI", "MyApp", "minimized", 3)
```

ProfileString expression function

Obtains the string value of a setting in the specified profile file.

Syntax

```
ProfileString ( filename, section, key, default )
```

- ***filename*** — A string whose value is the name of the profile file. If you do not specify a full path, ProfileString uses the operating system's standard file search order to find the file.
- ***section*** — A string whose value is the name of a group of related values in the profile file. In the file, section names are in square brackets. Do not include the brackets in section. Section is not case sensitive.
- ***key*** — A string specifying the setting name in section whose value you want. The setting name is followed by an equal sign in the file. Do not include the equal sign in key. Key is not case sensitive.
- ***default*** — A string value that ProfileString returns if filename is not found, if section or key does not exist in filename, or if the value of key cannot be converted to an integer.

Return Values

String, with a maximum length of 4096 characters. Returns the string from key within section within filename. If filename is not found, section is not found in filename, or key is not found in section, ProfileString returns default. If an error occurs, it returns the empty string ("").

Usage

Use ProfileInt and ProfileString to get configuration settings from a profile file you have designed for your application. ProfileInt and ProfileString can read files with ANSI or UTF16-LE encoding on Windows systems, and ANSI or UTF16-BE encoding on UNIX systems.

In PowerBuilder, you can use PowerScript SetProfileString to change values in the profile file to customize your application's configuration at runtime. Before you make changes, you can use ProfileInt and ProfileString to obtain the original settings so you can optionally restore them when the user exits the application.

Examples

This example uses the following section in the PROFILE.INI file:

```
[Employee]
Name="Smith"
[Dept]
Name="Marketing"
```

This expression returns the string for the keyword Name in section Employee in file C:\PROFILE.INI. It returns None if the section or keyword does not exist. In this case it returns Smith:

```
ProfileString("C:\PROFILE.INI", "Employee", "Name", "None")
```

Rand expression function

Obtains a random whole number between 1 and a specified upper limit.

Syntax

```
Rand ( n )
```

- ***n*** — The upper limit of the range of random numbers you want returned. The lower limit is always 1. The upper limit cannot exceed 32,767.

Return Values

A numeric datatype, the datatype of n. Returns a random whole number between 1 and n.

Usage

The sequence of numbers generated by repeated calls to the Rand function is a computer-generated pseudorandom sequence.

You can control whether the sequence is different each time your application runs by calling the PowerScript Randomize function to initialize the random number generator.

Examples

This expression returns a random whole number between 1 and 10:

```
Rand(10)
```

Real expression function

Converts a string value to a real datatype.

Syntax

```
Real ( string )
```

- ***string*** — The string whose value you want to convert to a real

Return Values

Real. Returns the contents of a string as a real. If string is not a valid number, Real returns 0.

Examples

This expression converts 24 to a real:

```
Real("24")
```

This expression returns the value in the column temp_text as a real:

```
Real(temp_text)
```

RelativeDate expression function

Obtains the date that occurs a specified number of days after or before another date.

Syntax

```
RelativeDate ( date, n )
```

- ***date*** — A date value
- ***n*** — An integer indicating the number of days

Return Values

Date. Returns the date that occurs n days after date if n is greater than 0. Returns the date that occurs n days before date if n is less than 0.

Examples

This expression returns 2005-02-10:

```
RelativeDate(2005-01-31, 10)
```

This expression returns 2005-01-21:

```
RelativeDate(2005-01-31, -10)
```

RelativeTime expression function

Obtains a time that occurs a specified number of seconds after or before another time within a 24-hour period.

Syntax

```
RelativeTime ( time, n )
```

- ***time*** — A time value
- ***n*** — A long number of seconds

Return Values

Time. Returns the time that occurs n seconds after time if n is greater than 0. Returns the time that occurs n seconds before time if n is less than 0. The maximum return value is 23:59:59.

Examples

This expression returns 19:01:41:

```
RelativeTime(19:01:31, 10)
```

This expression returns 19:01:21:

```
RelativeTime(19:01:31, -10)
```

Replace expression function

Replaces a portion of one string with another.

Syntax

```
Replace ( string1, start, n, string2 )
```

- ***string1*** — The string in which you want to replace characters with string2.
- ***start*** — A long whose value is the number of the first character you want replaced. (The first character in the string is number 1.)
- ***n*** — A long whose value is the number of characters you want to replace.
- ***string2*** — The string that replaces characters in string1. The number of characters in string2 can be greater than, equal to, or fewer than the number of characters you are replacing.

Return Values

String. Returns the string with the characters replaced if it succeeds and the empty string ("") if it fails.

Usage

If the start position is beyond the end of the string, Replace appends string2 to string1. If there are fewer characters after the start position than specified in n, Replace replaces all the characters to the right of character start.

If n is zero, then in effect Replace inserts string2 into string1.

Examples

This expression changes the last two characters of the string David to e to make it Dave:

```
Replace("David", 4, 2, "e")
```

This expression returns MY HOUSE:

```
Replace("YOUR HOUSE", 1, 4, "MY")
```

This expression returns Closed for the Winter:

```
Replace("Closed for Vacation", 12, 8, "the Winter")
```

ReplaceA expression function

Replaces a portion of one string with another.

Syntax

```
ReplaceA ( string1, start, n, string2 )
```

- ***string1*** — The string in which you want to replace bytes with string2.
- ***start*** — A long whose value is the number of the first byte you want replaced. (The first byte in the string is number 1.)
- ***n*** — A long whose value is the number of bytes you want to replace.
- ***string2*** — The string that replaces bytes in string1. The number of bytes in string2 can be greater than, equal to, or fewer than the number of bytes you are replacing.

Return Values

String. Returns the string with the bytes replaced if it succeeds and the empty string ("") if it fails.

Usage

If the start position is beyond the end of the string, ReplaceA appends string2 to string1. If there are fewer bytes after the start position than specified in n, ReplaceA replaces all the bytes to the right of character start.

If n is zero, then in effect ReplaceA inserts string2 into string1.

ReplaceA replaces the functionality that Replace had in DBCS environments in PowerBuilder 9. In SBCS environments, Replace and ReplaceA return the same results.

RGB expression function

Calculates the long value that represents the color specified by numeric values for the red, green, and blue components of the color.

Syntax

```
RGB ( red, green, blue )
```

- ***red*** — The integer value of the red component of the color
- ***green*** — The integer value of the green component of the color
- ***blue*** — The integer value of the blue component of the color

Return Values

Long. Returns the long that represents the color created by combining the values specified in red, green, and blue. If an error occurs, RGB returns null.

Usage

The formula for combining the colors is:

```
Red + (256 * Green) + (65536 * Blue)
```

Use RGB to obtain the long value required to set the color for text and drawing objects. You can also set an object's color to the long value that represents the color. The RGB function provides an easy way to calculate that value.

The value of a component color is an integer between 0 and 255 that represents the amount of the component that is required to create the color you want. The lower the value, the darker the color; the higher the value, the lighter the color.

The following table lists red, green, and blue values for the 16 standard colors:

Color	Red value	Green value	Blue value
Black	0	0	0
White	255	255	255
Light Gray	192	192	192
Dark Gray	128	128	128
Red	255	0	0
Dark Red	128	0	0
Green	0	255	0
Dark Green	0	128	0
Blue	0	0	255
Dark Blue	0	0	128
Magenta	255	0	255
Dark Magenta	128	0	128
Cyan	0	255	255
Dark Cyan	0	128	128
Yellow	255	255	0
Brown	128	128	0

Examples

This expression returns as a long 8421376, which represents dark cyan:

```
RGB(0,128,128)
```

This expression for the Background.Color property of a salary column returns a long that represents red if an employee's salary is greater than $50,000 and white if salary is less than or equal to $50,000:

```
If(salary>50000, RGB(255,0,0), RGB(255,255,255))
```

RichText expression function

Takes as argument a string expression interpreted as RTF and renders it as such. If the argument is not RTF nothing is rendered.

Syntax

```
RichText ( string )
```

- ***string*** — The string expression to render as RTF

Return Values

None.

Examples

This expression displays the contents of the short_desc column's as rich text.

```
RichText( short_desc )
```

RichTextFile expression function

Takes as argument a string expression interpreted as a RTF file name and renders the contents. If the argument is not a RTF file, nothing is rendered.

Syntax

```
RichTextFile ( string )
```

- ***string*** — The string expression to render as RTF file

Return Values

None.

Examples

This expression displays the contents of the richtext.rtf file as rich text.

```
RichTextFile("richtext.rtf")
```

Right expression function

Obtains a specified number of characters from the end of a string.

Syntax

```
Right ( string, n )
```

- ***string*** — The string from which you want characters returned
- ***n*** — A long whose value is the number of characters you want returned from the right end of string

Return Values

String. Returns the rightmost n characters in string if it succeeds and the empty string ("") if an error occurs.

If n is greater than or equal to the length of the string, Right returns the entire string. It does not add spaces to make the return value's length equal to n.

Examples

This expression returns HILL:

```
Right("CASTLE HILL", 4)
```

This expression returns CASTLE HILL:

```
Right("CASTLE HILL", 75)
```

RightA expression function

Obtains a specified number of characters from the end of a string.

Syntax

```
Right ( string, n )
```

- ***string*** The string from which you want characters returned
- ***n*** — A long whose value is the number of characters you want returned from the right end of string

Return Values

String. Returns the rightmost n characters in string if it succeeds and the empty string ("") if an error occurs.

If n is greater than or equal to the length of the string, RightA returns the entire string. It does not add spaces to make the return value's length equal to n.

Usage

RightA replaces the functionality that Right had in DBCS environments in PowerBuilder 9. In SBCS environments, Right and RightA return the same results.

RightTrim expression function

Removes spaces from the end of a string.

Syntax

```
RightTrim ( string )
```

- ***string*** — The string you want returned with trailing blanks deleted

Return Values

String. Returns a copy of string with trailing blanks deleted if it succeeds and the empty string ("") if an error occurs.

Examples

This expression returns "CASTLE":

```
RightTrim("CASTLE ")
```

Round expression function

Rounds a number to the specified number of decimal places.

Syntax

```
Round ( x , n )
```

- ***x*** — The number you want to round.
- ***n*** — The number of decimal places to which you want to round x. Valid values are 0 through 28.

Return Values

Decimal. If n is positive, Round returns x rounded to the specified number of decimal places. If n is negative, it returns x rounded to (- n +1) places before the decimal point. Returns –1 if it fails.

Examples

This expression returns 9.62:

```
Round(9.624, 2)
```

This expression returns 9.63:

```
Round(9.625, 2)
```

This expression returns 9.600:

```
Round(9.6, 3)
```

This expression returns -9.63:

```
Round(-9.625, 2)
```

This expression returns -10:

```
Round(-9.625, -1)
```

RowCount expression function

Obtains the number of rows that are currently available in the primary buffer.

Syntax

```
RowCount ( )
```

Return Values

Long. Returns the number of rows that are currently available, 0 if no rows are currently available, and –1 if an error occurs.

Examples

This expression in a computed field returns a warning if no data exists and the number of rows if there is data:

```
If(RowCount() = 0, "No Data", String(RowCount()))
```

RowHeight expression function

Reports the height of a row associated with a band in a DataWindow object.

Syntax

```
RowHeight ( )
```

Return Values

Long. Returns the height of the row in the units specified for the DataWindow object if it succeeds, and –1 if an error occurs.

Usage

When you call RowHeight in a band other than the detail band, it reports on a row in the detail band. See GetRow for a table specifying which row is associated with each band for reporting purposes.

When a band has Autosize Height set to true, you should avoid using the RowHeight DataWindow expression function to set the height of any element in the row. Doing so can result in a logical inconsistency between the height of the row and the height of the element. If you need to use RowHeight, you must set the Y coordinate of the element to 0 on the Position page in the Properties view, otherwise the bottom of the element might be clipped. You must do this for every element that uses such an expression. If you move any elements in the band, make sure that their Y coordinates are still set to 0.

You should not use an expression whose runtime value is greater than the value returned by RowHeight. For example, you should not set the height of a column to rowheight() + 30. Such an expression produces unpredictable results at runtime.

Examples

This expression for a computed field in the detail band displays the height of each row:

```
RowHeight()
```

Second expression function

Obtains the number of seconds in the seconds portion of a time value.

Syntax

```
Second ( time )
```

- ***time*** — The time value from which you want the seconds

Return Values

Integer. Returns the seconds portion of time (00 to 59).

Examples

This expression returns 31:

```
Second(19:01:31)
```

SecondsAfter expression function

Gets the number of seconds one time occurs after another.

Syntax

```
SecondsAfter ( time1, time2 )
```

- ***time1*** — A time value that is the start time of the interval being measured
- ***time2*** — A time value that is the end time of the interval

Return Values

Long. Returns the number of seconds time2 occurs after time1. If time2 occurs before time1, SecondsAfter returns a negative number.

Examples

This expression returns 15:

```
SecondsAfter(21:15:30, 21:15:45)
```

This expression returns -15:

```
SecondsAfter(21:15:45, 21:15:30)
```

This expression returns 0:

```
SecondsAfter(21:15:45, 21:15:45)
```

Sign expression function

Reports whether the number is negative, zero, or positive by checking its sign.

Syntax

```
Sign ( n )
```

- ***n*** — The number for which you want to determine the sign.

Return Values

Integer. Returns a number (–1, 0, or 1) indicating the sign of n.

Examples

This expression returns 1 (the number is positive):

```
Sign(5)
```

This expression returns 0:

```
Sign(0)
```

This expression returns –1 (the number is negative):

```
Sign(-5)
```

Sin expression function

Calculates the sine of an angle.

Syntax

```
Sin ( n )
```

- ***n*** — The angle (in radians) for which you want the sine.

Return Values

Double. Returns the sine of n if it succeeds and –1 if an error occurs.

Examples

This expression returns .8414709848078965:

```
Sin(1)
```

This expression returns 0:

```
Sin(0)
```

This expression returns 0:

```
Sin(pi(1))
```

Small expression function

Finds a small value at a specified ranking in a column (for example, third-smallest, fifth-smallest) and returns the value of another column or expression based on the result.

Syntax

```
Small ( returnexp, column, nbottom { FOR range { DISTINCT { expres1 {, expres2 {, ... } } } } } } )
```

- **returnexp** — The value you want returned when the small value is found. Returnexp includes a reference to a column, but not necessarily the column that is being evaluated for the small value, so that a value is returned from the same row that contains the small value.
- **column** — The column that contains the small value you are searching for. Column can be a column name or a column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column. The datatype of column must be numeric.
- **nbottom** — The relationship of the small value to the column's smallest value. For example, when nbottom is 2, Small finds the second-smallest value.
- **FOR *range*** — (optional) The data that will be included when finding the small value. For most presentation styles, values for range are:
  - **ALL** — (Default) The small value of all rows in column.
  - **GROUP n** — The small value of rows in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** — The small value of the rows in column on a page.
  - **CROSSTAB** — (Crosstabs only) The small value of all rows in column in the crosstab.
  - **GRAPH** — (Graphs only) The small value in column in the range specified for the Rows option.
  - **OBJECT** — (OLE objects only) The small value in column in the range specified for the Rows option.
- **DISTINCT** — (optional) Causes Small to consider only the distinct values in column when determining the small value. For a value of column, the first row found with the value is used and other rows that have the same value are ignored.
- ***expresn*** — (optional) One or more expressions that you want to evaluate to determine distinct rows. Expresn can be the name of a column, a function, or an expression.

Return Values

The datatype of returnexp. Returns the nbottom-smallest value if it succeeds and -1 if an error occurs.

Usage

If you specify range, Small returns the value in returnexp when the value in column is the nbottom-smallest value in range. If you specify DISTINCT, Small returns returnexp when the value in column is the nbottom-smallest value of the distinct values in column, or if you specify expresn, the nbottom-smallest for each distinct value of expresn.

For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data property page (the Range property), and the aggregation function uses that range.

Settings for Rows include the following:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

Tip: If you do not need a return value from another column and you want to find the smallest value (nbottom = 1), use Min; it is faster.

You cannot use this or other aggregate functions in validation rules or filter expressions.

Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a DataWindow object always retrieves all rows.

Examples

These expressions return the names of the salespersons with the three smallest sales (sum_sales is the sum of the sales for each salesperson) in group 2, which might be the salesregion group. Note that sum_sales contains the values being compared, but Small returns a value in the name column:

```
Small(name, sum_sales, 1 for group 2)
Small(name, sum_sales, 2 for group 2)
Small(name, sum_sales, 3 for group 2)
```

This example reports the salesperson with the third-smallest sales, considering only the first entry for each salesperson:

```
Small(name, sum_sales, 3 for all DISTINCT sum_sales)
```

Space expression function

Builds a string of the specified length whose value consists of spaces.

Syntax

```
Space ( n )
```

- ***n*** — A long whose value is the length of the string you want filled with spaces

Return Values

String. Returns a string filled with n spaces if it succeeds and the empty string ("") if an error occurs.

Examples

This expression for a computed field returns 10 spaces in the computed field if the value of the rating column is Top Secret; otherwise, it returns the value in rating:

```
If(rating = "Top Secret", Space(10), rating)
```

Sqrt expression function

Calculates the square root of a number.

Syntax

```
Sqrt ( n )
```

- ***n*** — The number for which you want the square root.

Return Values

Double. Returns the square root of n.

Usage

Sqrt(n) is the same as n ^.5. Taking the square root of a negative number causes an execution error.

Examples

This expression returns 1.414213562373095:

```
Sqrt(2)
```

This expression results in an error at execution time:

```
Sqrt(-2)
```

StDev expression function

Calculates an estimate of the standard deviation for the specified column. Standard deviation is a measurement of how widely values vary from average.

Syntax

```
StDev ( column { FOR range { DISTINCT { expres1 {, expres2 {, ... } } } } } )
```

- **column** — The column for which you want an estimate for the standard deviation of the values in the rows. Column can be the column name or the column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column. The datatype of column must be numeric.
- ***FOR range*** — (optional) The data to be included in the estimate of the standard deviation. For most presentation styles, values for range are:
  - **ALL** — (Default) The estimate of the standard deviation for all values in column.
  - **GROUP *n*** — The estimate of the standard deviation for values in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** — The estimate of the standard deviation for the values in column on a page.
  - **CROSSTAB** — (Crosstabs only) The standard deviation for all values in column in the crosstab.
  - **GRAPH** — (Graphs only) The standard deviation in column in the range specified for the Rows option.
  - **OBJECT** — (OLE objects only) The standard deviation in column in the range specified for the Rows option.
- **DISTINCT** — (optional) Causes StDev to consider only the distinct values in column when determining the standard deviation. For a value of column, the first row found with the value is used and other rows that have the same value are ignored.
- ***expresn*** — (optional) One or more expressions that you want to evaluate to determine distinct rows. Expresn can be the name of a column, a function, or an expression.

Return Values

Double. Returns an estimate of the standard deviation for column.

Usage

If you specify range, StDev returns an estimate for the standard deviation of column within range. If you specify DISTINCT, StDev returns an estimate of the standard deviation for the distinct values in column, or if you specify expresn, the estimate of the standard deviation of the rows in column where the value of expresn is distinct.

For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data tab page (the Range property), and the aggregation function uses that range. Settings for Rows include the following:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

When estimating or calculating actual standard deviation, StDev assumes that the values in column are a sample of the values in the rows in the column in the database table. If you selected all the rows in the column in the DataWindow object's SELECT statement, use StDevP to compute the standard deviation of the population.

You cannot use this or other aggregate functions in validation rules or filter expressions. Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a DataWindow object always retrieves all rows.

Examples

These examples all assume that the SELECT statement did not retrieve all the rows in the database table. StDev is intended to work with a subset of rows, which is a sample of the full set of data.

This expression returns an estimate for standard deviation of the values in the column named salary:

```
StDev(salary)
```

This expression returns an estimate for standard deviation of the values in the column named salary in group 1:

```
StDev(salary for group 1)
```

This expression returns an estimate for standard deviation of the values in column 4 on the page:

```
StDev(#4 for page)
```

This expression entered in the Value box on the Data tab page in the graph's property sheet returns an estimate for standard deviation of the values in the qty_used column in the graph:

```
StDev(qty_used for graph)
```

This expression for a computed field in a crosstab returns the estimate for standard deviation of the values in the qty_ordered column in the crosstab:

```
StDev(qty_ordered for crosstab)
```

Assuming a DataWindow object displays the order number, amount, and line items for each order, this computed field returns the estimated standard deviation of the order amount for the distinct order numbers:

```
StDev(order_amt for all DISTINCT order_nbr)
```

StDevP expression function

Calculates the standard deviation for the specified column. Standard deviation is a measurement of how widely values vary from average.

Syntax

```
StDevP ( column { FOR range { DISTINCT { expres1 {, expres2 {, ... } } } } } )
```

- ***column*** — The column for which you want the standard deviation of the values in the rows. Column can be the column name or the column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column. The datatype of column must be numeric.
- ***FOR range*** — (optional) The data to be included in the standard deviation. For most presentation styles, values for range are:
  - **ALL** — (Default) The standard deviation for all values in column.
  - **GROUP *n*** — The standard deviation for values in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** — The standard deviation for the values in column on a page.
  - **CROSSTAB** — (Crosstabs only) The standard deviation for all values in column in the crosstab.
  - **GRAPH** — (Graphs only) The standard deviation for values in column in the range specified for the Rows option.
  - **OBJECT** — (OLE objects only) The standard deviation for values in column in the range specified for the Rows option.
- **DISTINCT** — (optional) Causes StDevP to consider only the distinct values in column when determining the standard deviation. For a value of column, the first row found with the value is used and other rows that have the same value are ignored.
- ***expresn*** — (optional) One or more expressions that you want to evaluate to determine distinct rows. Expresn can be the name of a column, a function, or an expression.

Return Values

Double. Returns the standard deviation for column.

Usage

If you specify range, StDevP returns the standard deviation for column within range. If you specify DISTINCT, StDevP returns an estimate of the standard deviation for the distinct values in column, or if you specify expresn, the estimate of the standard deviation of the rows in column where the value of expresn is distinct.

For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data tab page (the Range property), and the aggregation function uses that range. Settings for Rows include the following:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

When estimating or calculating actual standard deviation, StDevP assumes that the values in *column* are the values in all the rows in the column in the database table. If you did not select all rows in the column in the SELECT statement, use StDev to compute an estimate of the standard deviation of a sample.

You cannot use this or other aggregate functions in validation rules or filter expressions. Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a DataWindow object always retrieves all rows.

Examples

These examples all assume that the SELECT statement retrieved all rows in the database table. StDevP is intended to work with a full set of data, not a subset.

This expression returns the standard deviation of the values in the column named salary:

```
StDevP(salary)
```

This expression returns the standard deviation of the values in group 1 in the column named salary:

```
StDevP(salary for group 1)
```

This expression returns the standard deviation of the values in column 4 on the page:

```
StDevP(#4 for page)
```

This expression entered in the Value box on the Data tab page in the graph's property sheet returns the standard deviation of the values in the qty_ordered column in the graph:

```
StDevP(qty_ordered for graph)
```

This expression for a computed field in a crosstab returns the standard deviation of the values in the qty_ordered column in the crosstab:

```
StDevP(qty_ordered for crosstab)
```

Assuming a DataWindow object displays the order number, amount, and line items for each order, this computed field returns the standard deviation of the order amount for the distinct order numbers:

```
StDevP(order_amt for all DISTINCT order_nbr)
```

String expression function

Formats data as a string according to a specified display format mask. You can convert and format date, DateTime, numeric, and time data. You can also apply a display format to a string.

Syntax

```
String ( data {, format } )
```

- ***data*** — The data you want returned as a string with the specified formatting. Data can have a date, DateTime, numeric, time, or string datatype.
- ***format*** — (optional) A string of the display masks you want to use to format the data. The masks consist of formatting information specific to the datatype of data. If data is type string, format is required.
  The format string can consist of more than one mask, depending on the datatype of data. Each mask is separated by a semicolon. See Usage for details on each datatype.

Return Values

String. Returns data in the specified format if it succeeds and the empty string ("") if the datatype of data does not match the type of display mask specified or format is not a valid mask.

Usage

For date, DateTime, numeric, and time data, the system's default format is used for the returned string if you do not specify a format. For numeric data, the default format is the [General] format.

For string data, a display format mask is required. (Otherwise, the function would have nothing to do.)

The format can consist of one or more masks:

- Formats for date, DateTime, string, and time data can include one or two masks. The first mask is the format for the data; the second mask is the format for a null value.
- Formats for numeric data can have up to four masks. A format with a single mask handles both positive and negative data. If there are additional masks, the first mask is for positive values, and the additional masks are for negative, zero, and null values.

A format can include color specifications.

If the display format does not match the datatype, the attempt to apply the mask produces unpredictable results. For information on specifying display formats, consult the PowerBuilder documentation.

When you use String to format a date and the month is displayed as text (for example, when the display format includes "mmm"), the month is in the language of the deployment files available when the application is run. If you have installed

localized files in the development environment or on a user's machine, then on that machine the month in the resulting string will be in the language of the localized files.

For information about localized deployment files, see the chapter on internationalizing an application in Application Techniques.

Examples

This expression returns Jan 31, 2005:

```
String(2005-01-31, "mmm dd, yyyy")
```

This expression returns Jan 31, 2005 6 hrs and 8 min:

```
String(2005-01-31 06:08:00, 'mmm dd, yyyy, h "hrs and" m "min"')
```

This expression:

```
String(nbr, "0000;(000);****;empty")
```

returns:

- 0123 if nbr is 123
- (123) if nbr is -123
- **** if nbr is 0
- empty if nbr is null

This expression returns A-B-C:

```
String("ABC", "@-@-@")
```

This expression returns A*B:

```
String("ABC", "@*@")
```

This expression returns ABC:

```
String("ABC", "@@@")
```

This expression returns a space:

```
String("ABC", " ")
```

This expression returns 6 hrs and 8 min:

```
String(06:08:02,'h "hrs and" m "min"')
```

This expression returns 08:06:04 pm:

```
String(20:06:04, "hh:mm:ss am/pm")
```

This expression returns 8:06:04 am:

```
String(08:06:04, "h:mm:ss am/pm")
```

This expression returns 6:11:25.300000:

```
String(6:11:25.300000, "h:mm:ss.ffffff")
```

StripRTF expression function

Removes the rich text formatting from the specified column

Syntax

```
StripRTF ( string )
```

- ***string*** — The column to be stripped of rich text formatting.

Examples

This expression is used in a compute field expression to remove the formatting from a rich text edit column and display plain text in the compute field.

```
StripRTF(rte_description)
```

Sum expression function

Calculates the sum of the values in the specified column.

Syntax

```
Sum ( column { FOR range { DISTINCT { expres1 {, expres2 {, ... } } } } } )
```

- ***column*** — The column for which you want the sum of the data values. Column can be the column name or the column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column. The datatype of column must be numeric.
- **FOR *range*** — (optional) The data to be included in the sum. For most presentation styles, values for range are:
  - **ALL** — (Default) The sum of all values in column.
  - **GROUP n** — The sum of values in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** — The sum of the values in column on a page.
  - **CROSSTAB** — (Crosstabs only) The sum of all values in column in the crosstab.
  - **GRAPH** — (Graphs only) The sum of values in column in the range specified for the Rows option of the graph.
  - **OBJECT** — (OLE objects only) The sum of values in column in the range specified for the Rows option of the OLE object.
- **DISTINCT** — (optional) Causes Sum to consider only the distinct values in column when determining the sum. For a value of column, the first row found with the value is used and other rows that have the same value are ignored.
- ***expresn*** — (optional) One or more expressions that you want to evaluate to determine distinct rows. Expresn can be the name of a column, a function, or an expression.

Return Values

The appropriate numeric datatype. Returns the sum of the data values in column.

Usage

If you specify range, Sum returns the sum of the values in column within range. If you specify DISTINCT, Sum returns the sum of the distinct values in column, or if you specify expresn, the sum of the values of column where the value of expresn is distinct.

For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data property page (the Range property), and the aggregation function uses that range. Settings for Rows include the following:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

Null values are ignored and are not included in the calculation.

You cannot use this or other aggregate functions in validation rules or filter expressions.

Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a DataWindow object always retrieves all rows.

Examples

This expression returns the sum of the values in group 1 in the column named salary:

```
Sum(salary for group 1)
```

This expression returns the sum of the values in column 4 on the page:

```
Sum(#4 for page)
```

Assuming a DataWindow object displays the order number, amount, and line items for each order, this computed field returns the sum of the order amount for the distinct order numbers:

```
Sum(order_amt for all DISTINCT order_nbr)
```

Tan expression function

Calculates the tangent of an angle.

Syntax

```
Tan ( n )
```

- ***n*** — The angle (in radians) for which you want the tangent

Return Values

Double. Returns the tangent of n if it succeeds and –1 if an error occurs.

Examples

Both these expressions return 0:

```
Tan(0)
Tan(Pi(1))
```

This expression returns 1.55741:

```
Tan(1)
```

Time expression function

Converts a string to a time datatype.

Syntax

```
Time ( string )
```

- ***string*** — A string containing a valid time (such as 8 am or 10:25) that you want returned as a time datatype. Only the hour is required; you do not have to include the minutes, seconds, or microseconds of the time or am or pm. The default value for minutes and seconds is 00 and for microseconds is 000000. am or pm is determined automatically.

Return Values

Time. Returns the time in string as a time datatype. If string does not contain a valid time, Time returns 00:00:00.

Examples

This expression returns the time datatype for 45 seconds before midnight (23:59:15):

```
Time("23:59:15")
```

This expression for a computed field returns the value in the time_received column as a value of type time if time_received is not the empty string. Otherwise, it returns 00:00:00:

```
If(time_received = "", 00:00:00, Time(time_received))
```

This example is similar to the previous one, except that it returns 00:00:00 if time_received contains a null value:

```
If(IsNull(time_received), 00:00:00, Time(time_received))
```

Today expression function

Obtains the system date and time.

Syntax

```
Today ( )
```

Return Values

DateTime. Returns the current system date and time.

Usage

To display both the date and the time, a computed field must have a display format that includes the time.

The PowerScript and DataWindow painter versions of the Today function have different datatypes. The return value of the PowerScript Today function is date.

Examples

This expression for a computed field displays the date and time when the display format for the field is "mm/dd/yy hh:mm":

```
Today()
```

Trim expression function

Removes leading and trailing spaces from a string.

Syntax

```
Trim ( string )
```

string The string you want returned with leading and trailing spaces deleted

Return Values

String. Returns a copy of string with all leading and trailing spaces deleted if it succeeds and the empty string ("") if an error occurs.

Usage

Trim is useful for removing spaces that a user might have typed before or after newly entered data.

Examples

This expression returns "CASTLE HILLS":

```
Trim(" CASTLE HILLS ")
```

Truncate expression function

Truncates a number to the specified number of decimal places.

Syntax

```
Truncate ( x, n )
```

- ***x*** — The number you want to truncate.
- ***n*** — The number of decimal places to which you want to truncate x. Valid values are 0 through 28.

Return Values

The datatype of x. If n is positive, returns x truncated to the specified number of decimal places. If n is negative, returns x truncated to (- n +1) places before the decimal point. Returns –1 if it fails.

Examples

This expression returns 9.2:

```
Truncate(9.22, 1)
```

This expression returns 9.2:

```
Truncate(9.28, 1)
```

This expression returns 9:

```
Truncate(9.9, 0)
```

This expression returns -9.2:

```
Truncate(-9.29, 1)
```

This expression returns 0:

```
Truncate(9.2, -1)
```

This expression returns 50:

```
Truncate(54, -1)
```

Upper expression function

Converts all characters in a string to uppercase letters.

Syntax

```
Upper ( string )
```

- ***string*** — The string you want to convert to uppercase letters

Return Values

String. Returns string with lowercase letters changed to uppercase if it succeeds and the empty string ("") if an error occurs.

Examples

This expression returns "CASTLE HILLS":

```
Upper("Castle Hills")
```

Var expression function

Calculates an estimate of the variance for the specified column. The variance is the square of the standard deviation.

Syntax

```
Var ( column { FOR range { DISTINCT { expres1 {, expres2 {, ... } } } } } )
```

- **column** — The column for which you want an estimate for the variance of the values in the rows. Column can be the column name or the column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column. The datatype of column must be numeric.
- **FOR *range*** — (optional) The data to be included in the estimate of the variance. For most presentation styles, values for range are:
  - **ALL** — (Default) The estimate of the variance for all rows in column.
  - **GROUP *n*** — The estimate of the variance for rows in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** — The estimate of the variance for the rows in column on a page.
  - **CROSSTAB** — (Crosstabs only) The estimate of the variance for all rows in column in the crosstab.
  - **GRAPH** — (Graphs only) The estimate of the variance for rows in column in the range specified for the Rows option.
  - **OBJECT** — (OLE objects only) The estimate of the variance for rows in column in the range specified for the Rows option.
- **DISTINCT** — (optional) Causes Var to consider only the distinct values in column when determining the variance. For a value of column, the first row found with the value is used and other rows that have the same value are ignored.
- ***expresn*** — (optional) One or more expressions that you want to evaluate to determine distinct rows. Expresn can be the name of a column, a function, or an expression.

Return Values

Double or decimal if the arguments are decimal. Returns an estimate for the variance for column. If you specify group, Var returns an estimate for the variance for column within group.

Usage

If you specify range, Var returns an estimate for the variance for column within range. If you specify DISTINCT, Var returns the variance for the distinct values in column, or if you specify expresn, the estimate for the variance of the rows in column where the value of expresn is distinct.

For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data property page (the Range property), and the aggregation function uses that range.

Settings for Rows include the following:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

When estimating variance or calculating actual variance, Var assumes that the values in column are a sample of the values in rows in the column in the database table. If you select all rows in the column in the SELECT statement, use VarP to compute the variance of a population.

You cannot use this or other aggregate functions in validation rules or filter expressions. Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a DataWindow object always retrieves all rows.

Examples

These examples all assume that the SELECT statement did not retrieve all of the rows in the database table. Var is intended to work with a subset of rows, which is a sample of the full set of data.

This expression returns an estimate for the variance of the values in the column named salary:

```
Var(salary)
```

This expression returns an estimate for the variance of the values in the column named salary in group 1:

```
Var(salary for group 1)
```

This expression entered in the Value box on the Data property page in the graph's property sheet returns an estimate for the variance of the values in the quantity column in the graph:

```
Var(quantity for graph)
```

This expression for a computed field in a crosstab returns an estimate for the variance of the values in the quantity column in the crosstab:

```
Var(quantity for crosstab)
```

Assuming a DataWindow object displays the order number, amount, and line items for each order, this computed field returns the estimate for the variance of the order amount for the distinct order numbers:

```
Var(order_amt for all DISTINCT order_nbr)
```

VarP expression function

Calculates the variance for the specified column. The variance is the square of the standard deviation.

Syntax

```
VarP ( column { FOR range { DISTINCT { expres1 {, expres2 {, ... } } } } } )
```

- **column** — The column for which you want the variance of the values in the rows. Column can be the column name or the column number preceded by a pound sign (#). Column can also be an expression that includes a reference to the column. The datatype of column must be numeric.
- **FOR *range*** — (optional) The data that will be included in the variance. For most presentation styles, values for range are:
  - **ALL** — (Default) The variance for all rows in column.
  - **GROUP *n*** — The variance for rows in column in the specified group. Specify the keyword GROUP followed by the group number: for example, GROUP 1.
  - **PAGE** — The variance for the rows in column on a page.
  - **CROSSTAB** — (Crosstabs only) The variance for all rows in column in the crosstab.
  - **GRAPH** — (Graphs only) The variance for rows in column in the range specified for the Rows option.
  - **OBJECT** — (OLE objects only) The variance for rows in column in the range specified for the Rows option.
- **DISTINCT** — (optional) Causes VarP to consider only the distinct values in column when determining the variance. For a value of column, the first row found with the value is used and other rows that have the same value are ignored.
- **expresn** — (optional) One or more expressions that you want to evaluate to determine distinct rows. Expresn can be the name of a column, a function, or an expression.

Return Values

Double or decimal if the arguments are decimal. Returns the variance for column. If you specify group, Var returns the variance for column within range.

Usage

If you specify range, VarP returns the variance for column within range. If you specify DISTINCT, VarP returns the variance for the distinct values in column, or if you specify expresn, the variance of the rows in column where the value of expresn is distinct.

For graphs and OLE objects, you do not select the range when you call the function. The range has already been determined by the Rows setting on the Data property page (the Range property), and the aggregation function uses that range. Settings for Rows include the following:

- For the Graph or OLE presentation style, Rows is always All.
- For Graph controls, Rows can be All, Page, or Group.
- For OLE controls, Rows can be All, Current Row, Page, or Group. The available choices depend on the layer the control occupies.

When estimating variance or calculating actual variance, VarP assumes that the values in column are the values in all rows in the column in the database table. If you did not select all the rows in the column in the SELECT statement, use Var to compute an estimate of the variance of a sample.

You cannot use this or other aggregate functions in validation rules or filter expressions. Using an aggregate function cancels the effect of setting Retrieve Rows As Needed in the painter. To do the aggregation, a DataWindow object always retrieves all rows.

Examples

These examples all assume that the SELECT statement retrieved all rows in the database table. VarP is intended to work with a full set of data, not a subset.

This expression returns the variance of the values in the column named salary:

```
VarP(salary)
```

This expression returns the variance of the values in group 1 in the column named salary:

```
VarP(salary for group 1)
```

This expression returns the variance of the values in column 4 on the page:

```
VarP(#4 for page)
```

This expression entered in the Value box on the Data property page in the graph's property sheet returns the variance of the values in the quantity column in the graph:

```
VarP(quantity for graph)
```

This expression for a computed field in a crosstab returns the variance of the values in the quantity column in the crosstab:

```
VarP(quantity for crosstab)
```

Assuming a DataWindow object displays the order number, amount, and line items for each order, this computed field returns the variance of the order amount for the distinct order numbers:

```
VarP(order_amt for all DISTINCT order_nbr)
```

WordCap expression function

Sets the first letter of each word in a string to a capital letter and all other letters to lowercase (for example, ROBERT E. LEE would be Robert E. Lee).

Syntax

```
WordCap ( string )
```

- ***string*** — A string or expression that evaluates to a string that you want to display with initial capital letters (for example, Monday Morning)

Return Values

String. Returns string with the first letter of each word set to uppercase and the remaining letters lowercase if it succeeds, and null if an error occurs.

Examples

This expression returns "Boston, Massachusetts":

```
WordCap("boston, MASSACHUSETTS")
```

This expression concatenates the characters in the emp_fname and emp_lname columns and makes the first letter of each word uppercase:

```
WordCap(emp_fname + " " + emp_lname)
```

Year expression function

Gets the year of a date value.

Syntax

Year (date)

- ***date*** — The date value from which you want the year

Return Values

Integer. Returns an integer whose value is a 4-digit year adapted from the year portion of date if it succeeds and 1900 if an error occurs.

If the year is two digits, then the century is set as follows. If the year is between 00 to 49, the first two digits are 20; if the year is between 50 and 99, the first two digits are 19.

Usage

Obtains the year portion of date. Years from 1000 to 3000 inclusive are handled.

If your data includes dates before 1950, such as birth dates, always specify a 4–digit year so that Year (and other functions, such as Sort) interpret the date as intended.

> **NOTE**
>
> To make sure you get correct return values for the year, you must verify that yyyy is the Short Date Style for year in the Regional Settings of the user's Control Panel. Your program can check this with the RegistryGet function. If the setting is not correct, you can ask the user to change it manually or to have the application change it (by calling the RegistrySet function). The user might need to reboot after the setting is changed.

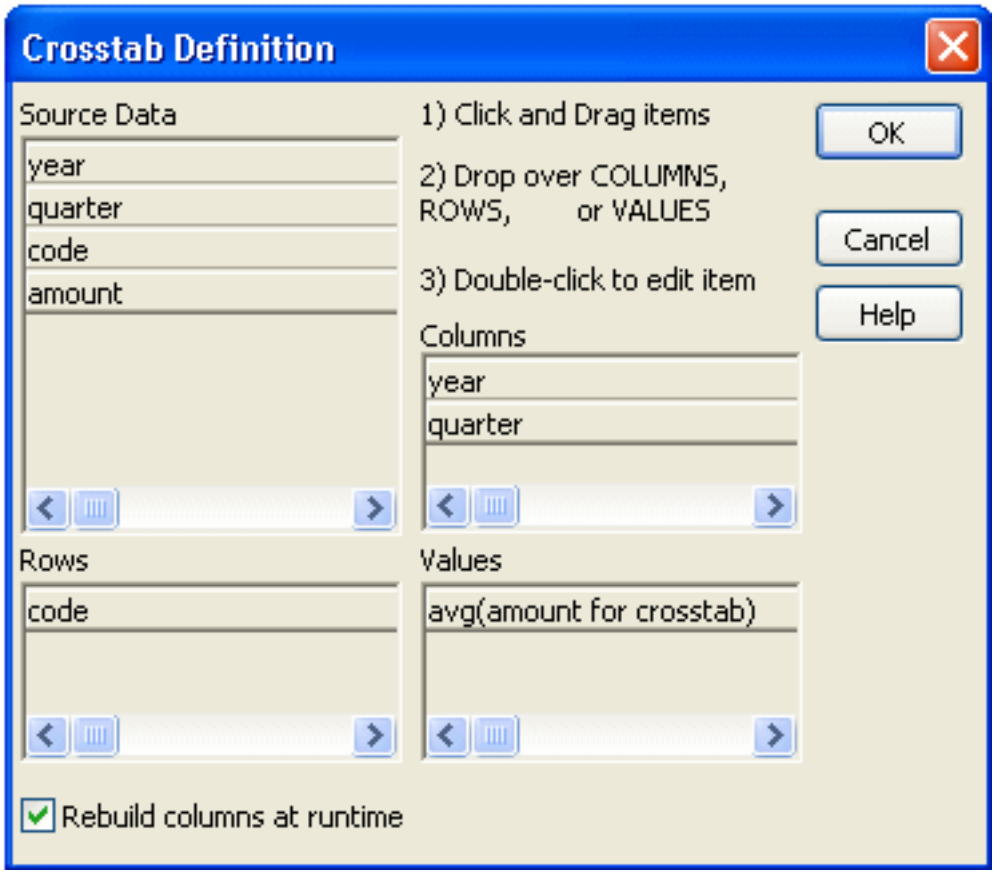
Examples

This expression returns 2005:

```
Year(2005-01-31)
```

How to Use Functions in a Crosstab

When a crosstab is generated from your definition, the appropriate computed fields are automatically created using the Crosstab functions. To understand the functions, consider a crosstab with two columns (year and quarter), a row (product), and the values expression Avg(amount for crosstab).



When you define the crosstab described in the screenshot, the painter automatically creates the appropriate computed fields. A computed field named avg_amount returns the average of the quarterly figures for each year. Its expression is:

```
CrosstabAvg(1, 2, "@year")
```

A second computed field named grand_avg_amount computes the average of all the amounts in the row. Its expression is:

```
CrosstabAvg(1)
```

Other computed fields in the summary band use the Avg function to display the average of the values in the amount column, the yearly averages, and the final average. Each row in the crosstab (after adjusting the column widths) has cells for the amounts in the quarters, a repeating cell for the yearly average, and a grand average. The crosstab also displays averages of the amounts for all the financial codes in the quarters in the summary band at the bottom.

What the function arguments mean

When the crosstab definition has more than one column, you can specify column qualifiers for any of the Crosstab functions, so that the crosstab displays calculations for groups of column values. As illustrated previously, when year and quarter are the columns in the crosstab, the expression for the computed field is:

```
CrosstabAvg(1, 2, "@year")
```

The value 2 refers to the quarter column (the second column in the Crosstab Definition dialog) and "@year" specifies grouping values from the year column (meaning the function will average values for the quarters within each year). The value 1 refers to the crosstab-values expression that will be averaged. In the resulting crosstab, the computed field repeats in each row after the cells for the quarters within each year.

Tips for defining crosstabs

When you define a crosstab with more than one column, the order of the columns in the Columns box of the Crosstab Definition dialog box governs the way the columns are grouped. To end up with the most effective expressions, make the column that contains the grouping values (for example, year or department) the first column in the Columns box and the column that contains the values to be grouped (for example, quarter or employee) second.

To display calculations for groups of rows, define groups as you would for other DataWindow presentation styles and define computed fields in the group header or footer using noncrosstab aggregation functions, such as Avg, Sum, or Max.

Reviewing the expressions

To review the expressions defined for the crosstab values, open the Crosstab Definition dialog box (select Design>Crosstab from the menubar).

Examples

The first two examples use the crosstab expressions shown below:

```
Count(emp_id for crosstab),Sum(salary for crosstab)
```

This expression for a computed field in the crosstab returns the average of the employee counts (the first expression):

```
CrosstabAvg(1)
```

This expression for a computed field in the crosstab returns the average of the salary totals (the second expression):

```
CrosstabAvg(2)
```

Consider a crosstab that has two columns (region and city) and the values expression Avg(sales for crosstab). This expression for a computed field in the detail band computes the average sales over all the cities in a region:

```
CrosstabAvg(1, 2, "@region")
```

This expression for another computed field in the same crosstab computes the grand average over all the cities:

```
CrosstabAvg(1)
```

Seed Lists

The seed list in the repository database is stored in the gtrep_reference_data table. Test Data Manager uses this data for synthetic data generation.

How you browse and edit seed lists depends on the component:

- Open Datamaker and click **Tools, Maintain Seed Data**.
- Browse the seedtables directory in the Fast Data Masker installation directory, for example: `C:\Program Files\Grid-Tools\FastDataMasker\seedtables\`

Datamaker ships with the following seed data included:

- International person names
- Job titles
- SSN, HIC, and EIN codes
- International postal codes and city names
- Country names
- International addresses and street names
- BIC codes
- Credit card types and numbers
- Currency codes
- Names of days and months
- Bank transaction types
- Phone numbers
- Products
- and more.

NOTE

More information:

- Fast Data Masker Best Practices
- Propagate Seed List Data Across Masking Engines

Documentation Legal Notice

This Documentation, which includes embedded help systems and electronically distributed materials, (hereinafter referred to as the “Documentation”) is for your informational purposes only and is subject to change or withdrawal by Broadcom at any time. This Documentation is proprietary information of Broadcom and may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of Broadcom.

If you are a licensed user of the software product(s) addressed in the Documentation, you may print or otherwise make available a reasonable number of copies of the Documentation for internal use by you and your employees in connection with that software, provided that all Broadcom copyright notices and legends are affixed to each reproduced copy.

The right to print or otherwise make available copies of the Documentation is limited to the period during which the applicable license for such software remains in full force and effect. Should the license terminate for any reason, it is your responsibility to certify in writing to Broadcom that all copies and partial copies of the Documentation have been returned to Broadcom or destroyed.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS DOCUMENTATION “AS IS” WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT WILL CA BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENTATION, INCLUDING WITHOUT LIMITATION, LOST PROFITS, LOST INVESTMENT, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF CA IS EXPRESSLY ADVISED IN ADVANCE OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.

The use of any software product referenced in the Documentation is governed by the applicable license agreement and such license agreement is not modified in any way by the terms of this notice.

The manufacturer of this Documentation is Broadcom Inc.

Provided with “Restricted Rights.” Use, duplication or disclosure by the United States Government is subject to the restrictions set forth in FAR Sections 12.212, 52.227-14, and 52.227-19(c)(1) - (2) and DFARS Section 252.227-7014(b)(3), as applicable, or their successors.


A Broadcom
Company
technologies