

What's New in TDM 4.1?

CA TDM Product Team

Legal Statement

© 2017 CA. All rights reserved. CA confidential & proprietary information. For CA, CA Partner and CA Customer use only. No unauthorized use, copying or distribution. All names of individuals or of companies referenced herein are fictitious names used for instructional purposes only. Any similarity to any real persons or businesses is purely coincidental. All trademarks, trade names, service marks and logos referenced herein belong to their respective companies. These Materials are for your informational purposes only, and do not form any type of warranty. The use of any software or product referenced in the Materials is governed by the end user's applicable license agreement. CA is the manufacturer of these Materials. 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, 52.227-19(c)(1)-(2) and DFARS Section 252.227-7014(b)(3), as applicable, or their successors.

Agenda

- Tester Self Service – Find and Reserve
- Parameterised Variables
- Set Source & Target in Meta Functions and TDM Portal
- Enterprise Option in Datamaker

Tester Self Service – Find and Reserve

Problem Statement - Test Data Engineer/Data Analyst

- Overhead of creating and maintaining TSS tiles for changing test data requirements
- Lack of common language between testers who work on application objects and TDE who work on backend data sources
- The current architecture restricted to relational data sources and the reserve action – no support for APIs, VSAM files as a data source

Problem Statement - Tester

- Lack of support to find and visualize the test data before initiating a test data engineering activity
- Lack of support for rapid release cycles – APIs, aggregate & reuse reservations
- Inability to synthetically generate in context of the missing data

Core Value Propositions

- Define an environment that allows logical grouping of application data sources so that we needn't create duplicate projects for every environment
- Build a logical model via the UI so that extensive effort is not needed to build a denormalized data mart
- Provide a self-service catalog UI so that the testers can find the data and then reserve or release it
- Provide REST API support so that I can find, reserve and release data from my automation scripts

Demo

Test Data Manager Portal

Project: Order Management System [1.0] administrator

Modeling

- Test Data Models
- Objects
- Variables
- Environments
- Generators
- Orchestration
- Self Service Flows
- Self Service Catalog
- Submitted Requests
- Configuration

Test Data Model > New Test Data Model

Build Test Data Model: Select Resource Key

Search

Name	Type
Orders	
OrderName	VarChar
OrderDate	Data
OrderDetails	
OrderID	Numerical
ProductID	Numerical
ShipperID	Numerical
Products	
ProductID	Numerical
ProductName	VarChar
ProductDescription	VarChar
Quotes	
QuotesID	Text
QuotesDescription	Text
QuotesInternational	Numerical

Name	Type	Actions
OrderID	Numerical	

Step 3 of 3

PREVIOUS FINISH CANCEL

Version 4.0.100.14 | Third Party Notices

Copyright © 2016 CA. All rights reserved.

Parameterised Variables

What are Parameterised Variables (macros for short)?

- A macro is an extension of the existing notion of a variable.
- Normal variables can contain expressions which get evaluated when required.
- Previously, expressions could contain only constants, variables and functions of constants, eg $\sim v1 \sim = @add(42, \sim v2 \sim)@$
- In effect, a variable was secretly a function with no parameters.
- In TDMWeb 4.1, a variable can have parameters which are used in the expression. Eg $\sim v1(param1) \sim = @add(\#param1\#, \sim v2 \sim)@$
- So if $v2=2$ then $\sim v1(100) \sim$ returns $100+2=102$

What are they for?

- A macro allows the TDE to re-use expressions.
 - Previously invocations of an expression in different columns or tables required replication of that expression. If a change was required then all occurrences had to be changed.
 - Multiple occurrences with variations can be defined which vary only in the parameters, not by duplicating and modifying code.
- A macro allows a library of more flexible common expressions to be created.
 - These macros would be defined with global scope.
 - They could then be used in multiple projects.
 - Complex expressions are written and tested once only.

How do you create a macro?

- Macros are created using the variables page in the usual way.
- The name of the macro specifies the parameters, eg `myvar(a,b,c)`.
- The expression is defined using datapainter as usual.
- Note that DP can't evaluate a macro because it doesn't know what the values of the arguments are so it just does a syntax check.

Examples

Variable	Expression	Purpose
ifmore5(a)	@if(#a# > 5, yes, no)@	Outputs yes if a>5 otherwise no
addcheck(a,b,c)	@if(@sum(#a#,#b#,#c# ,~OFFSET~)@ > 1000, std, non-std)@	Add three numbers and the value of variable OFFSET. If the result is more than 100, then output the string "std", otherwise "non-std".
sayhello(n)	@repeat(hello,#n#,-)@	Output "hello" n times
factorial(x)	@case(#x# < 0, error, #x# = 0, 1, #x#=1, 1, @multiply(#x# ,~factorial(@subtract(#x#,1)@)~)@)@	Factorial function eg @factorial(4)@ = 4x3x2x1 = 24

S & T Feature

Using S & T in Meta functions

SQL commands in meta-function expression can contain a connection profile name.

Examples:

```
@execsql(connection,select column from table where something)@
```

```
@seqlov(percnll,@sqlist(connection, select column from table where something)@
```

Previously TDMWeb(Portal) only supported connections of the format
P{profilename}

```
@execsql(Ptravel,select name top 1 from cities)@
```

Using S & T in Meta functions

In Datamaker

At login, the user selects a source & target profile.

These can be used by meta functions, replacing P{profile} with T or S.

```
@execsql(S,select name top 1 from cities)@
```

```
@execsql(T,select name top 1 from cities)@
```

This feature now available in TDM Portal.

Enterprise option in Datamaker

Enterprise Support in Datamaker

- Call the enhanced publish engine for large publish jobs
- Call Datamaker built in publish engine for smaller and quick job