TIMEXTENDER

TX2014 USER GUIDE

CONTENTS

This User Guide is a compilation of the following:

TX2014 Technical Documentation	p. 3
TX2014 SR1 Release Documentation	p. 256
TX2014 SR2 Release Documentation	p. 298

Visit our support site at support.timextender.com to download the newest version of the User Guide, browse the how-tos and frequently asked questions in the Knowledge Base or get in touch with our support staff.

TX2014
TECHNICAL DOCUMENTATION

TIMEXTENDER

Technical Documentation

Table of Contents

Table of Contents 1

Getting Started with TX2014 9

TX2014 Installation 9

TX2014 Projects 9

Connecting to Data Sources 9

Selecting and Cleansing Data 9

Designing the Data Warehouse 10

Modeling Cubes 10

Deploying and Executing Projects 10

Introducing the TX2014 User Interface 11

Data Tab 11

Cubes Tab 11

Execution Tab 11

Warnings Tab 12

Errors Tab 12

Working with Projects 13

Defining Project Repositories 13

Creating Projects 14

Opening Projects 15

Saving Projects 15

Exporting and Importing Projects 15

Importing Projects from XML Documents 15

Exporting Projects to XML Documents 15

Data Sources 16

Specifying the Database Collation 16

How to Specify the Database Collation 16

Adding SQL Server Data Sources 17

Add Microsoft Excel Data Sources 19

Add Text Files as Data Sources 19
To Add a Text File 19
To Specify the Field Properties of a Text File 20
Add MySQL Data Sources 21
Add ODBC Data Sources 22
To Change Data Source Providers 24
To Change the Provider to Microsoft SQL Server 24
To Change the Provider to Oracle 24
Synchronize Data from Data Sources 25
Preview Data Source Tables 25
Replicate Data Sources 25
Enabling Replicate Data Sources 25
About Adapters 28
Dynamics NAV Adapters 28
To Add Dynamics NAV Adapters 28
To Add an MS SQL Provider 28
To Add Navision Native Database Sources 28
To Change the Dynamics NAV Company Table 29
To Load and Select Data from Dynamics NAV Data Sources 30
To Set Up Dynamics NAV Companies 30
To Change Dynamics NAV Schemas 30
To Modify Table Usage on Dynamics NAV Tables 31
To Modify the Usage of a Single Dynamics NAV Table 31
Dynamics AX Adapters 32
To Import XPO Files into Dynamics AX 32
Add Dynamics AX Adapters 32
To Add a Microsoft SQL Server Provider 33
To Add an Oracle Provider 33
To Set the Account Table 33
To Set Up Dynamics AX Companies 34
To Load and Select Data from Dynamics AX Data Sources 34
To Add Dynamics AX Virtual Table References 34
To View Dynamics AX Table Information 35
To View Dynamics AX Enum Table Information 35
To Change Dynamics AX Schemas 36
To Modify Table Usage on Dynamics AX Tables 36

To Modify the Usage of a Single Dynamics AX Table 36
Dynamics GP Adapters 37
To Add Dynamics GP Adapters 37
To Set Up Dynamics GP Companies 37
To Load and Select Data from Dynamics GP Data Sources 38
Salesforce.com Adapters 39
Deploying and Executing Projects 41
Deploying a Project 41
Executing a Project 41
Manual Deployment and Execution 41
To Deploy Individual Objects 41
To Execute Individual Objects 43
Deploy/Execute Only Modified Objects 43
How to Deploy/Execute Modified Objects 43
To Deploy and Execute Objects at the Same Time 44
To Deploy and Execute the Entire Project 45
How to Only Deploy and Execute Modified Objects 45
Display Unused Fields 47
How to Display Unused Fields 47
To Add Business Units 49
To Specify a Staging Database 49
External Business Units 50
To Add External Business Units 50
To Synchronize External Business Units 51
Staging Databases 52
To Specify a Staging Database 52
Creating Relationships 54
To Create Relationships 54
Data Warehouses 55
To Create Data Warehouses 55
To Add Tables To A Data Warehouse 56
To Assign Primary Keys 57
To Add Custom Fields 57
To Add Conditional Lookup Fields 58
To Create a Field 58
Specify the Lookup Field 59
Add Joins 60

Specifying Conditions 60
Understanding Hierarchy Tables 61
To Add a Hierarchy Table 61
To Create a Hierarchy Table Structure 62
To Specify the Execution Order of Tables 63
To Move Tables Up and Down in the Tree 63
Copy Table Structure 64
Import Custom Data from Excel 66
Synchronization of Data Types in Data Warehouse 69
Working with Date Tables 72
To Create Date Tables 72
Selecting and Validating Data 74
Data Selection Rules 74
To Define Data Selection Rules 75
Data Validation 76
To Define Data Validation Rules 77
To Define Conditions 77
To View Validation Errors 78
To View Validation Warnings 78
Data Transformation 79
To Define Field Transformation Rules 79
To Define Conditions 80
Handling Early Arriving Facts 81
Enabling Early Arriving Facts 81
Incremental Loading 86
Full Load Process 86
Incremental Load Process 86
Does My Organization Need Incremental Loading Enabled? 86
How to Enable Incremental Loading 86
Enabling Incremental Loading for Staging Database Tables 87
Enabling Incremental Loading for Data Warehouse Tables 90
How to Implement Target Based Incremental Loading 90
User Defined Functions and Stored Procedures 95
User Defined Functions 95
User Defined Stored Procedures 95
To Add User Defined Functions 95
To Add User Defined Stored Procedures 96

Script Actions 98
Adding a Script Action 98
Associating a Script Action with a Table 98
SQL Snippets 100
How to Create a SQL Snippet 100
How to Edit a SQL Snippet 101
How to Delete a SQL Snippet 102
How to Implement a SQL Snippet 102
Strongly Typed Values in Custom Fields 104
How to Use Strongly Typed Values in Custom Fields 104
Creating Views 106
To Create Views Based on Lookup Fields 106
To Create a Lookup Field 106
To Specify a Join 107
To Specify a Sort Order 108
To Create Views Based on Standard Fields 109
Add multiple Standard measures, tables, and fields easily 111
Development Deployment Toolkit 113
Development Environments 113
Prerequisites 113
Setting Up a Development Environment 113
Production Environment 114
Setting Up the Production Server 114
Development Environment 116
Setting Up the Development Server 116
Create Global Databases 118
Configure Global Databases 120
Configuring the Data Source 120
Configuring the Staging Database 121
Configuring the Data Warehouse 122
Configuring the OLAP Database 123
Creating the Global Databases 124
Configure Project Connections 124
Transfer the Project from Development to Production 127
Execution Packages 129
Team Development Environment 130
Enabling the Team Development Environment 130

Adding Work Items 130
How to View Existing Work Items 132
How to Edit a Work Item 133
How to Delete Work Items 134
How to Add Version Notes 136
How to View Version Notes 137
Concurrent Users Modifying the Same Object 139
Dimensional Modeling 140
Star Schema 140
Snowflake Schema 140
About Dimensions 142
Regular Dimensions 142
Creating Regular Dimensions 142
Parent-Child Dimensions 143
Creating Parent-Child Dimensions 143
To Define Advanced Parent-Child Dimension Settings 145
To Add Dimension Levels 147
To Add Quick Levels 148
To Add Dimension Joins 149
To Add Dimension Hierarchies 150
Dimension Hierarchy Properties 151
To Edit a Dimension Hierarchy 152
To Delete a Dimension Hierarchy 153
To Create a Date Dimension 153
Enabling Slowly Changing Dimensions 154
Different Types of Slowly Changing Dimensions 154
Type I 154
Type II 154
How to Implement Type II Slowly Changing Dimensions 155
Enable Slowly Changing Dimensions on the Dimension Table 155
Bringing the Surrogate Key to the Transaction Table 157
Move Surrogate Key from Transaction Table in Staging Database to Data Warehouse 158
Add Surrogate Key to Dimension Table in Data Warehouse 159
Update Dimension Key 160
Update the Dimension Relationships in the Cube 161

Dimension Attribute Properties 163
Dimension Browser 165
Display folders on Dimensions 166
Rename of Unknown Member Name 168
Use an attribute more than once in “attribute relationship” 170
Creating Cubes 174
To Add OLAP Servers 174
To Edit OLAP Server Settings 175
To Delete an OLAP Database 175
To Add Cubes 175
To Add a Single Dimension to a Cube 177
To Add Multiple Dimensions to a Cube 177
To Add Role-Playing Dimensions 178
To Add Dimension Relationships 178
Adding Measures 179
To Add Standard Measures 179
To Add Derived Measures 181
To Add Calculated Measures 182
To Validate Cubes 183
To Validate Dimensions 184
Use of DWH relations for OLAP relations 185
Drop down for format string on measures 188
Changing NullProcessing Property on Measures 189
Default Measure on Cubes 190
Measure Expression 191
Hiding a cube dimension 192
Aggregations on cubes 193
Descriptions on Cubes and Measures 196
Export and Import of OLAP Translations 198
OLAP Security 201
Setting up OLAP Database Security in TX2014 201
SSAS Cube Perspectives 214
Physical SSAS Cube Perspectives 218
Prevent Deployment and Execution of OLAP Objects 219
Rename Measuregroup 221
Drill through Actions 223
Standard Actions 228

Reporting Actions 233
High Availability Cube Processing 237
How to Enable High Availability Cube Processing 237
Cube Browser 239
How to Launch the Cube Browser 239
How to Use the Cube Browser 240
Execution Packages 242
To Create Notifications 242
Email Notifications 243
To Add Execution Packages 244
To Add Advanced Execution Packages 246
To Specify Execution Schedules 247
To Specify a Daily Execution Schedule 247
To Specify a Weekly Execution Schedule 250
Remind users to deploy the project after execution package changes 251
Execution of external SSIS packages 252

Getting Started with TX2014

This section provides an overview of the components you'll be working with when you develop a solution in TX2014.

TX2014 Installation

The installation of TX2014 is covered in a separate document. The installation documentation can be viewed on the support site at: http://support.timextender.com.

TX2014 Projects

The project is a container for all of the elements of your solution. The key elements of a project are the data warehouse and the business unit.

The data warehouse is a Microsoft SQL Server database. The data warehouse stores all of the extracted and cleansed data that you need for query and analysis.

Business units represent separate units within your organization. For example, if you have a global organization, you could create one business unit for the world headquarters, and separate business units for each subsidiary. A business unit contains a staging database and one or more data sources.

For more information, see **Working with Projects**.

Connecting to Data Sources

The TX2014 supports a wide variety of data sources. Connectors facilitate access to all major ERP and CRM systems, major relational databases, MS Excel files, and text files. Furthermore, access to generic or legacy databases is possible through generic ODBC. For more information, see **Data Sources**.

Selecting and Cleansing Data

Selecting data is the process where you identify which data you need to extract from the data source.

After selecting tables and fields, you further limit your selection by applying Data Selection Rules.

The selected data is moved to a staging database where the cleansing process takes place according to the transformation and validation rules you have specified. The transformation and validation rules ensure consistent data, uniform formatting of data, and removal of duplicate data.

The staging database stores the cleansed data temporarily until it has been cleansed and stored in the data warehouse. Extracting the selected data to a staging database means that the cleansing process has very little effect on the transaction database, and thus on the daily business operations.

The staging database consists of different tables that store the extracted data before and after the cleansing process. To handle the data transformations, a number of views are created. The SQL code that is generated during the cleansing process is stored in the staging database. For more information, see **Selecting and Validating Data**.

Designing the Data Warehouse

When you design the data warehouse in the TX2014, you create relationships between the tables in your data warehouse, assign primary keys, create views, and specify mapping tables. For more information, see **Creating Data Warehouses**.

Modeling Cubes

The standard schemas for designing a data warehouse are star schemas and snowflake schemas.

To create cubes, you use dimensions to structure how you want to analyze your data and measures to specify which numerical values you want to analyze. Cubes are stored in an OLAP database, and you can use your preferred front-end application to drill-down or roll-up through the data.

You can also reverse engineer an existing OLAP database and then use TX2014 to maintain and change existing cubes. Reverse engineering existing cubes requires the purchase of an additional feature which will facilitate this process. For more information, see **Creating Cubes**.

Deploying and Executing Projects

During deployment, the structure of your data warehouse and of your cubes is created.

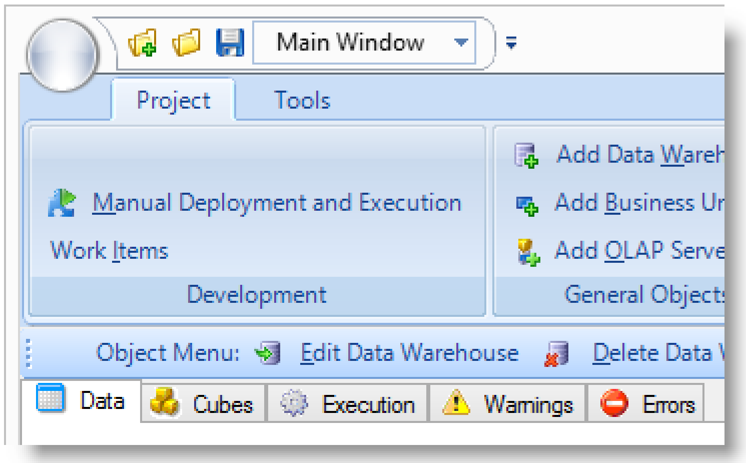
When you execute a project, data is loaded into the data warehouse and the cubes are processed.

During deployment, the underlying SQL code is automatically generated and stored in the data warehouse.

Introducing the TX2014 User Interface

This section provides an introduction to the main work areas of TX2014 user interface. The interface consists of a ribbon section and the main work-area that contains a set of tabs. These tabs contain information about different elements of your project.

The ribbon presents the tools and actions that are relevant to the current selection in the project tree.



Data Tab

The Data tab contains the project tree and all related elements. It is used for specifying business units, data warehouses, and for extracting, transforming, and loading data. When you edit the elements in the project tree, a Properties window is displayed to the right when applicable.

Cubes Tab

The Cubes tab is used for defining multi-dimensional cubes. The cubes tree contains all the elements you use to define cubes, such as dimensions and measures. Measures and dimensions are listed below the cube they are associated with. A master list of dimensions is listed separately because they can be used in more than one cube.

Execution Tab

The Execution tab is used for scheduling the automatic execution, and for keeping track of the execution process. In the Execution Package tree, you can add elements, such as custom actions, checkpoints, notifications, and schedules. The tree lists the element in the order in which they are processed. You can move custom actions up or down in the tree depending on when you want the actions to be executed.

Warnings Tab

The Warnings tab is used to view warnings resulting from violations of validation rules. You can see which row is affected and which rule has been violated for each warning . Records with warnings will still be promoted to the data warehouse or staging database.

Errors Tab

The Errors tab is used to view errors resulting from violations of validation rules. For each error, you can see which row is affected and which rule has been violated. Records with errors will not be promoted to the data warehouse or staging database and are excluded from the final data set.

Working with Projects

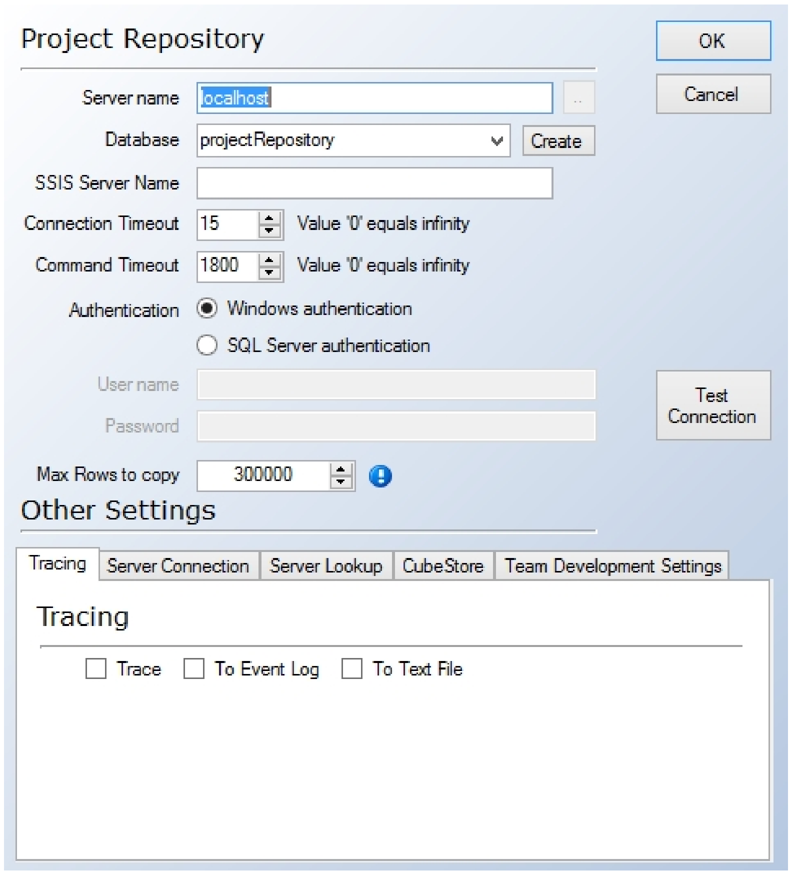
A project is a container for all the elements of your data warehouse solution. The primary components of a project are business units, data sources, and data warehouses.

You can only have one project open at a time in TX2014. However, if you want to compare different versions of a project, you can open another instance of TX2014, and load another version of the project to facilitate viewing projects side-by-side.

Defining Project Repositories

When you start using TX2014, you have to specify a project repository.

1. In the Quick Access Toolbar, from the **Tools** menu, select **General Settings**.


Project Repository

Server name	localhost
Database	projectRepository
SSIS Server Name	
Connection Timeout	15 Value '0' equals infinity
Command Timeout	1800 Value '0' equals infinity
Authentication	● Windows authentication ○ SQL Server authentication
User name	
Password	
Max Rows to copy	300000

Other Settings

Tracing

☐ Trace ☐ To Event Log ☐ To Text File

2. In the **Server Name** field, enter the name of the database server on which you want to store the project.

3. In the **Database** list, enter a name for the database, and then click **Create**. Alternatively, you can select an existing database from the list.

4. In the **Connection Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the server. The default is 15 seconds. A value of zero will disable the timeout.

5. In the **Command Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the database. The default is 1800 seconds. A value of zero will disable the time-out.

6. Specify the authentication mode. The default setting is Windows authentication. If you choose SQL Server authentication, you are prompted for a user name and a password.

7. Click **Test Connection** to verify that the connection is working.

8. **Active Directory Integration** is disabled by default. To enable **Active Directory Integration**, select **Enabled** in the **AD Integration** field.

9. You can use Trace to track the execution of your TX2014 projects and view the results.

• Select **To Event Log** to capture the trace to the Windows event log.

• Select **To Text File** to capture the trace to a text file.

All projects you create in the future will be saved in the specified repository, unless you change the repository. If the following message is displayed "Insufficient rights - must be a member ofTX2014 administrator group", please contact your system administrator for more information.

Note: When you install a new version of TX2014, you will be prompted to run an upgrade script that automatically updates the repository to ensure compatibility with the new software version.

Creating Projects

1. From the **File** menu, choose **New**.

2. In the **New Project** dialog, type a name for the new project.

3. Specify whether null values are allowed. To allow null values, check **Allow Null**.

4. To specify if null checks are field based, select **Field Based**, and to specify if checks are record based, check **Record Based**.

5. Specify the type of project you want to create. You have the following options:

Standard	Saves a new version whenever a project is executed.
History	Saves information about dimension attributes that change over time. This is also known as Slowly Changing Dimensions. For more information, see Enabling Slowly Changing Dimensions.

You can change a project type at any time during design time by returning to this screen.

You can only have one project open at a time in TX2014. If you try to create a new project or load an existing project when you already have one project open, you will be asked to save the current project.

Note: The first time you start working with TX2014, you must specify a project repository in which all projects are stored. For more information, see Defining Project Repositories.

Opening Projects

1. From the File menu, choose Open.

2. In the Project list, select the project that you want to open, and then click OK.

Saving Projects

1. From the File menu, choose Save or Save As.

2. Type a name for the project, and then click OK.

Exporting and Importing Projects

Being able to export a project to an XML document is useful when you want to save a copy of a running project for future reference, or if you want to reuse parts of a project in another project. You can export a project to an XML document, and you can import a project from an XML document.

Note: You cannot export a project that was downloaded directly from the CubeStore.

Importing Projects from XML Documents

1. From the File menu, choose Import/Export, and then click Import Project.

2. In the Import File field, click the ellipsis (…), and then navigate to and select the file you want to import.

3. Click Open, and then click OK.

Exporting Projects to XML Documents

1. From the File menu, choose Import/Export, and then click Export Project

2. In the Export File field, click the ellipsis (…), and then navigate to and select the file you want to export to.

3. Click Open, and then click OK.

Data Sources

Data sources contain the data that you want to retrieve and use for analysis queries. You have to add at least one data source per business unit. The current version of TX2014 connects to the following data sources:

• Microsoft SQL Server 2005 databases

• Microsoft SQL Server 2008 databases

• Microsoft SQL Server 2012 databases

• Microsoft Dynamics NAV versions 4.0 SP1 and later

• Microsoft Dynamics GP versions 9 and later

• Microsoft Excel files

• ODBC compliant data sources

• Plain text files

Licensing: Your license determines the number and type of data sources that you can connect to. A standard license permits a connection to one data source. Additional connectors can be purchased to permit the use of additional data sources.

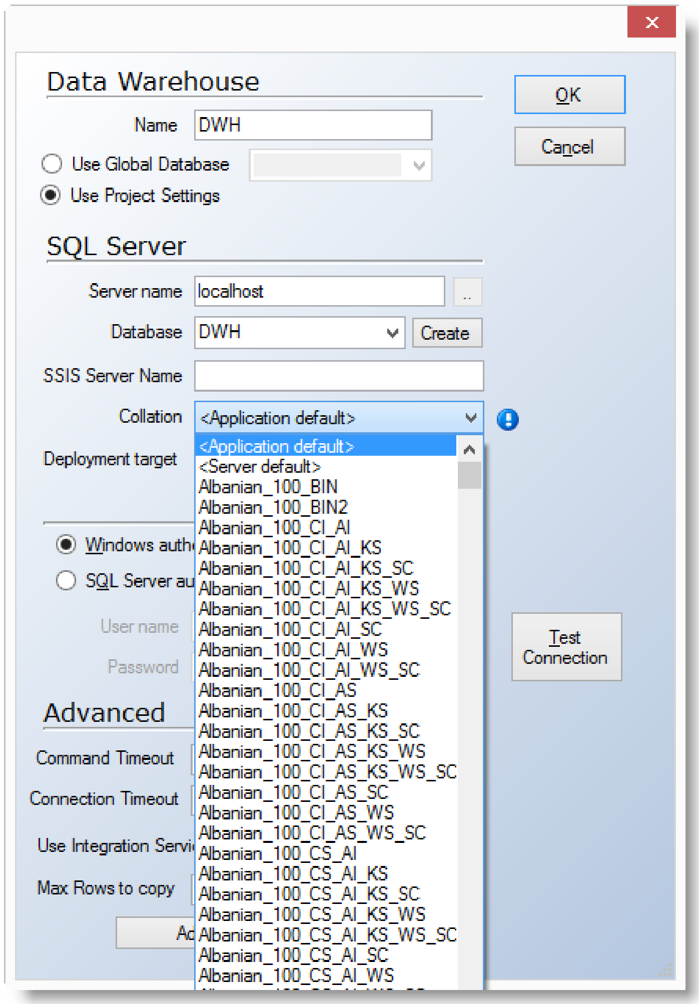
Note: The procedure for adding a data source varies depending on the type of data source.

Specifying the Database Collation

TX2014 has the ability to change the collation of the staging database, data warehouse, and OLAP databases from within TX2014 itself.

How to Specify the Database Collation

1. Right-click the database you would like to specify the collation for, and select Edit.
2. From the **Collation** drop-down, choose the desired collation.
3. <Application Default> will use the original **Latin1_General_CI_AS** collation.
4. <Server Default> will inherit the collation of the specified server.
5. Click **OK**.



Data Warehouse

Name: DWH

Use Global Database

[✓ Use Project Settings]

SQL Server

Server name: localhost

Database: DWH

SSIS Server Name:

Collation: <Application default>

<Application default>
<Server default>
Albanian_100_BIN
Albanian_100_BIN2
Albanian_100_CI_AI
Albanian_100_CI_AI_KS
Albanian_100_CI_AI_KS_SC
Albanian_100_CI_AI_KS_WS
Albanian_100_CI_AI_KS_WS_SC
Albanian_100_CI_AI_SC
Albanian_100_CI_AI_WS
Albanian_100_CI_AI_WS_SC
Albanian_100_CI_AS
Albanian_100_CI_AS_KS
Albanian_100_CI_AS_KS_SC
Albanian_100_CI_AS_KS_WS
Albanian_100_CI_AS_KS_WS_SC
Albanian_100_CI_AS_SC
Albanian_100_CI_AS_WS
Albanian_100_CI_AS_WS_SC
Albanian_100_CS_AI
Albanian_100_CS_AI_KS
Albanian_100_CS_AI_KS_SC
Albanian_100_CS_AI_KS_WS
Albanian_100_CS_AI_KS_WS_SC
Albanian_100_CS_AI_SC
Albanian_100_CS_AI_WS

Deployment target

[✓ Windows auth]

SQL Server au

User name

Password

Advanced

Command Timeout

Connection Timeout

Use Integration Servi

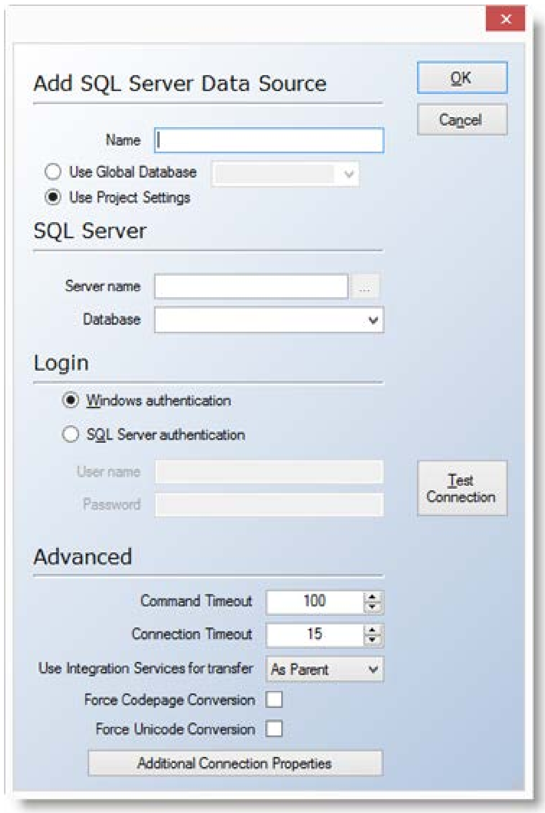
Max Rows to copy

Note: The best practice is to select the same collation setting for the staging database, data warehouse database, and OLAP database.

Adding SQL Server Data Sources

1. On the **Data** tab, expand the preferred business unit, then right-click Data Sources.

2. Select **Data Sources**, then select **Add SQL Server Data Source**.



Selected settings: Use Project Settings [✓]; Windows authentication [✓]; Command Timeout 100; Connection Timeout 15; Use Integration Services for transfer: As Parent; Force Codepage Conversion [ ]; Force Unicode Conversion [ ].

3. In the **Name** field, type a name for the data source. The name cannot exceed 15 characters in length.

4. In the **Server Name** field, enter the location of the database server.

5. In the **Database** field, enter the name of the database, or select it from the drop-down list.

6. Specify the authentication mode. The default setting is **Windows authentication**. If you choose **SQLServer authentication**, you are prompted for a user name and a password.

7. Click **Test Connection** to verify that the connection is working, and then click OK. The data source is added to the Data Sources folder in the project tree.

8. In the **Command Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the database. **Note: The recommended setting for this is 0 seconds to disable the timeout.**

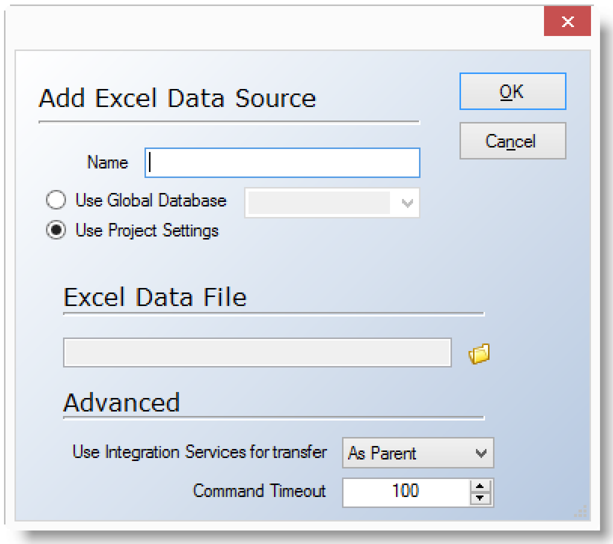
9. In the **Connection Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the server.

10. If you want to add additional connection strings, click the **Additional Connection Properties** button. In the **Connection String Properties** window, type the preferred connection strings, and then click **OK**.

Add Microsoft Excel Data Sources

To use Excel files as a data source, you must ensure that the worksheet data is in list format. This means that the data must be set up in a database format consisting of one or more named columns. The first row in each column must have a label, and there can be no blank columns or rows within the list.

1. On the **Data** tab, expand the preferred business unit, then right-click **Data Sources**.
2. Select **Data Sources**, and then select **Add Excel Data Source**to display the **Find Files or Folders** dialog.



Name:
Use Global Database [✓ Use Project Settings]
Use Integration Services for transfer: As Parent
Command Timeout: 100

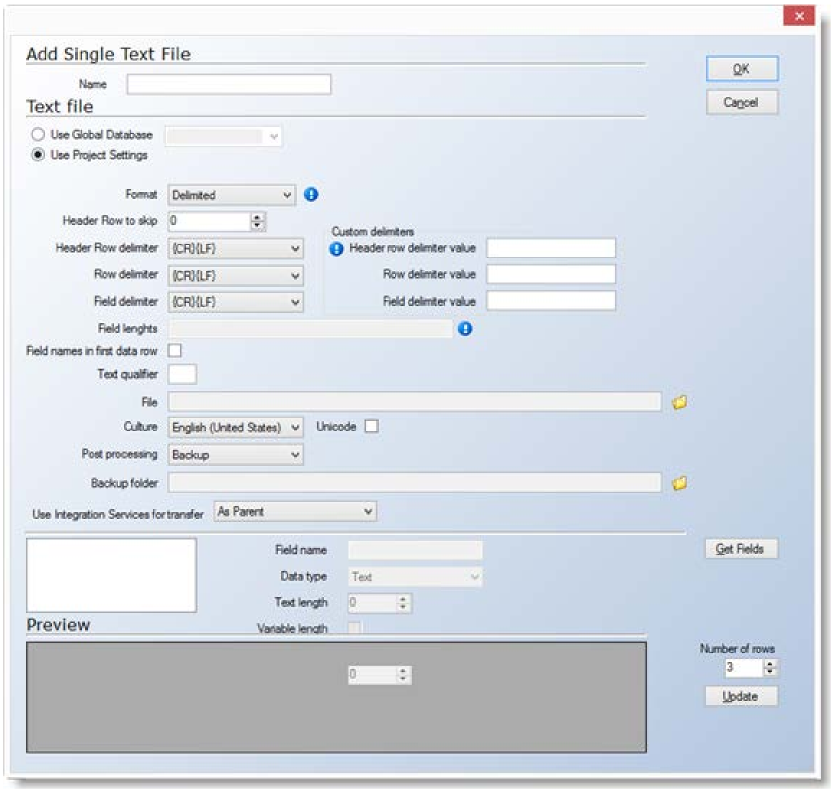
3. Navigate to and select the Excel file you want as data source, and then click **OK**.

Add Text Files as Data Sources

Text files are files that only contain text characters. Adding a text file as data source is a two-step process: adding the file and loading the fields; and then specifying properties for the fields once they have been loaded into the project. One text file corresponds to one table in TX2014.

To Add a Text File

1. Expand **Business Units**, and then expand the preferred business unit.
2. Right-click **Data Sources**, select **Data Source**, and then select **Add Single Text File data source**.


Add Single Text File

Name:

Text file

○ Use Global Database
● Use Project Settings

Format: Delimited
Header Row to skip: 0
Header Row delimiter: {CR}{LF}
Row delimiter: {CR}{LF}
Field delimiter: {CR}{LF}

Custom delimiters
Header row delimiter value:
Row delimiter value:
Field delimiter value:

Field lenghts:
Field names in first data row: ☐
Text qualifier:
File:
Culture: English (United States) Unicode: ☐
Post processing: Backup
Backup folder:
Use Integration Services for transfer: As Parent

Field name:
Data type: Text
Text length: 0
Variable length

Preview

Number of rows: 3

3. In the **Name** field, type a name for the data source.
4. Select **Header Row** if the first row contains column names.
5. In the **Row delimiter** list, specify how rows are separated.
6. In the **Field delimiter** list, specify how fields are separated.
7. In the **Text qualifier** field, type the text qualifier, if any.
8. In the **File** field, locate the file you want as data source.
9. In the **Culture** field, specify the language of the text file.
10. In the **Post processing** field, specify what to do with the file once it has been read.
11. In the **Backup** folder field, specify the path to the backup folder.
12. Click **Get Fields**to load the fields. You can view the fields in the **Fields**pane.

The fields are now loaded into your project, and by default, all fields are of the data type Text. However, you may have to specify a different data type for some of the fields.

To Specify the Field Properties of a Text File

1. Select a field in the **Fields** pane.

2. The name is entered by default, but you can specify a new name in the **Field Name** field.

3. In the **Data type** list, specify a data type.

4. In the **Text length** field, specify the maximum number of characters in the field.

5. Select **Variable Length**, if you do not want the field to have a fixed length.

6. Select **Unicode**, if you want to convert data to Unicode.

7. In the **Number of decimals** field, specify how many decimals are allowed in the field.

8. Next to the preview pane, in the **Number of rows** field, specify how many rows you want to preview, and then click **Update** to preview the rows.

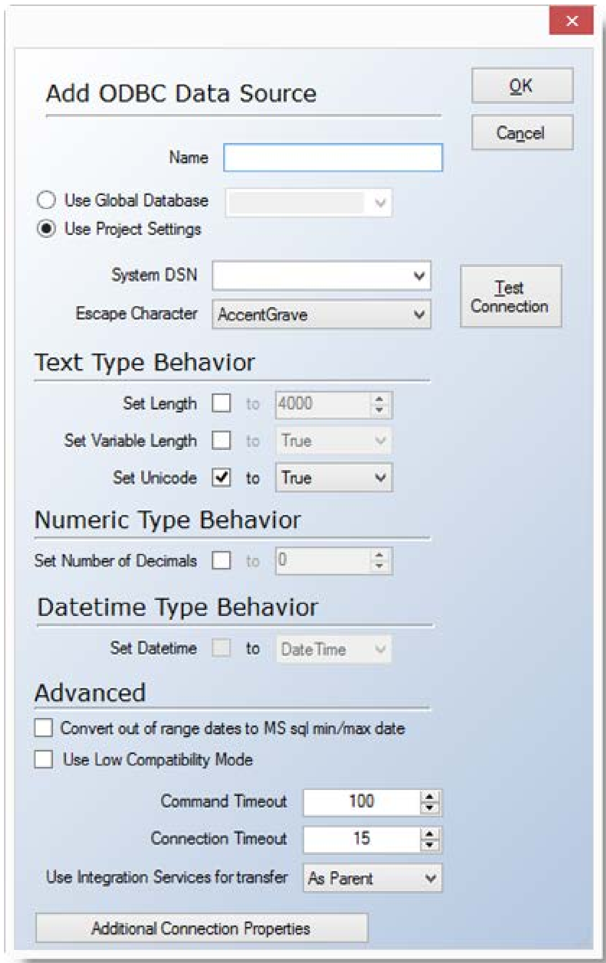
9. Click **OK**.

Add MySQL Data Sources

When you want to retrieve data from a MySQL 5.1 database, you have to use ODBC.

1. On the **Data** tab, expand the preferred business unit, and then right-click **Data Sources**.

2. Point to **Data Source**, select **Application specific ODBC**, and then select the preferred MySQL native database.



Name

( ) Use Global Database
(●) Use Project Settings

System DSN
Escape Character: AccentGrave

Text Type Behavior
Set Length ☐ to 4000
Set Variable Length ☐ to True
Set Unicode ✓ to True

Numeric Type Behavior
Set Number of Decimals ☐ to 0

Datetime Type Behavior
Set Datetime ☐ to DateTime

Advanced
☐ Convert out of range dates to MS sql min/max date
☐ Use Low Compatibility Mode
Command Timeout: 100
Connection Timeout: 15
Use Integration Services for transfer: As Parent

3. In the **Name** field, type the name of the data source.

4. In the **System DSN** list, select the Data Source Name.

5. In the **Escape Character** list, select the escape character specific to your ODBC driver.

6. The **Text Type Behavior** fields are used to control how the ODBC driver handles text. These fields are optional. You have the following options:

Set Length	Specifies an exact text string length
Set Variable Length	True, if you want a variable text string length
Set Unicode	True, if you want to use Unicode

7. In **Set Number of Decimals**, specify a fixed number of decimals. This field is optional.

8. Select **Convert out of range dates to MS SQL min/max** if you want to convert all dates older than January 01, 1753 to 01-01-1753.

9. Select **Use Low Compatibility Mode** if you have trouble retrieving data from the database.

10. In the **Command Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the database.

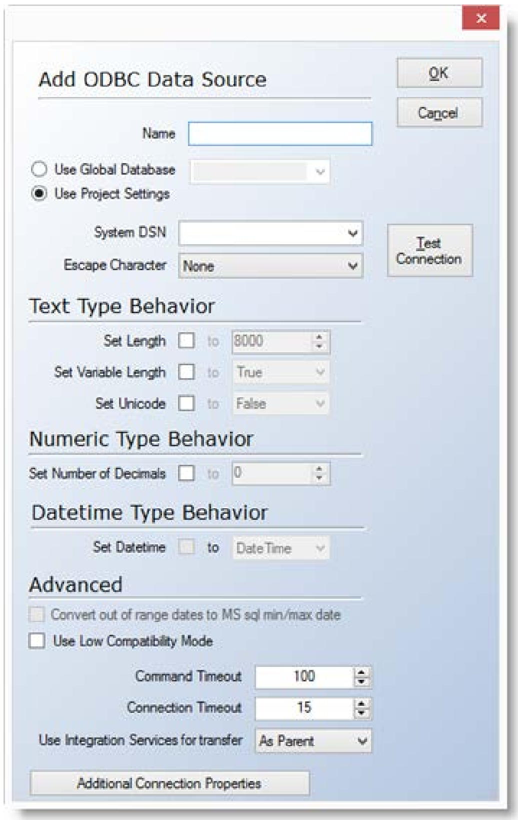
11. In the **Connection Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the server.

12. If you want to add additional connection strings, click the **Additional Connection Properties** button. In the **Connection String Properties** window, type the preferred connection strings, and then click **OK**.

Add ODBC Data Sources

When you want to retrieve data from legacy native databases, you can use ODBC.

1. On the **Data** tab, expand the preferred business unit, and then right-click **Data Sources**.


Add ODBC Data Source

Name:

( ) Use Global Database
(●) Use Project Settings

System DSN:
Escape Character: None

Text Type Behavior
Set Length [ ] to 8000
Set Variable Length [ ] to True
Set Unicode [ ] to False

Numeric Type Behavior
Set Number of Decimals [ ] to 0

Datetime Type Behavior
Set Datetime [ ] to DateTime

Advanced
[ ] Convert out of range dates to MS sql min/max date
[ ] Use Low Compatibility Mode
Command Timeout: 100
Connection Timeout: 15
Use Integration Services for transfer: As Parent

2. Point to **Data Source**, select **Application specific ODBC**, and then select the preferred data-base.

3. In the **Name** field, type the name of the data source.

4. In the **System DSN** list, select the Data Source Name.

5. In the **Escape Character** list, select the escape character specific to your ODBC driver.

6. The **Text Type Behavior** fields are used to control how the ODBC driver handles text. These fields are optional. You have the following options:

Set Length	Specifies an exact text string length
Set Variable Length	True, if you want a variable text string length
Set Unicode	True, if you want to use Unicode

7. In **Set Number of Decimals**, specify a fixed number of decimals. This field is optional.

Note: The **Convert out of range dates to MS SQL/min max** is not available for Navision native databases.

8. Select **Use low compatibility mode** if you have trouble retrieving data from the database.

9. In the **Command Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the database.

10. In the **Connection Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the server.

11. If you want to add additional connection strings, click the **Additional Connection Properties** button. In the **Connection String Properties** window, type the preferred connection strings, and then click **OK**.

To Change Data Source Providers

If you have moved your data sources to a new database, you have to change the provider specified for the data source. You can change provider from Microsoft SQL Server to Oracle, and vice versa.

1. On the **Data** tab, expand **Business Units**, then expand the preferred business unit.

2. Expand **Data Sources**, then select the data source that you want to change the provider for.

3. Right-click the data source, and select **Change Provider**, then select the preferred database.

To Change the Provider to Microsoft SQL Server

1. In the **Name** field, type a name for the data source. The name cannot exceed 15 characters in length.

2. In the **Server name**field, enter the location of the server.

3. In the **Database** field, enter the name of the database.

4. Specify the authentication mode. The default setting is **Windows authentication**. If you choose **SQL Server authentication**, you are prompted for a user name and a password.

5. Click **Test Connection** to verify that the connection is working, then click **OK**. The data source is added to the Data Sources folder in the project tree.

6. In the **Command Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the database.

7. In the **Connection Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the server.

8. If you want to add additional connection strings, click the **Additional Connection Properties** button. In the **Connection String Properties** window, type the preferred connection strings, and click **OK**.

To Change the Provider to Oracle

1. In the **TNS alias** field, type the alias that identifies the database.

2. In the **Owner** list, select the owner of the database.

3. Specify the authentication mode. When you select **Oracle authentication**, you are prompted for a user name and a password.

4. Select **Convert out of range dates to MS SQL min/max** if you want to convert all dates older than January 01, 1753 to 01-01-1753.

5. In the **Connection Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the server.

6. In the **Command Timeout field**, specify the number of seconds to wait before terminating the attempt to connect to the database.

7. If you want to set the character encoding to either Unicode or Non-Unicode, select **Force Character Setting**, and then select the preferred character encoding in the list.

Note: Forcing character encoding may affect performance.

8. If you want to add additional connection strings, click the **Additional Connection Properties** button. In the **Connection String Properties** window, type the preferred connection strings, and then click **OK**.

Synchronize Data from Data Sources

You can synchronize the data in your data source with the data in your project.

1. On the **Data** tab, expand **Business Units**, and then expand the preferred business unit.

2. Expand **Data Sources**, then right-click the preferred data source, and select **Synchronize Data Source**.

All tables and fields that have been removed from, or added to, the data source are listed in a separate window.

Preview Data Source Tables

If you want detailed information about a table in a data source, you can preview the table within the data source tree.

1. On the **Data** tab, expand the preferred business unit, and then expand **Data Sources**.

2. Expand the data source that contains the table you want to preview, right-click the table, and then select **Preview Table**.

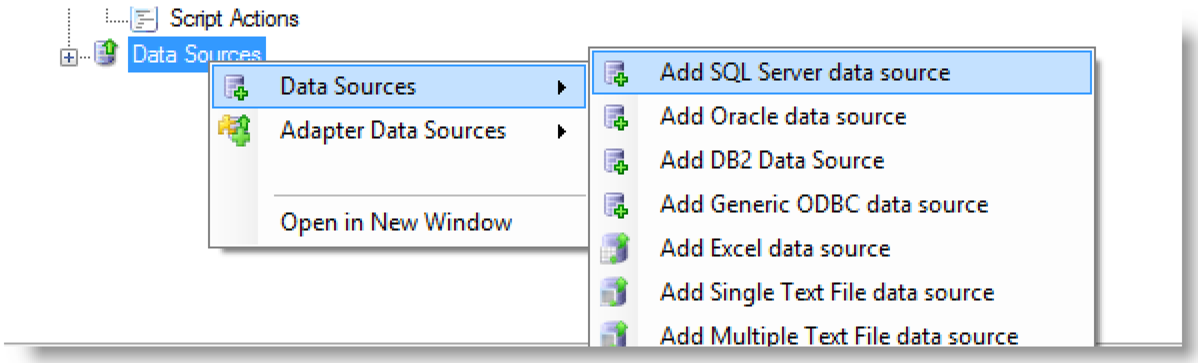
3. Click **Close**.

Replicate Data Sources

Replicate Data Sources allows a user to easily connect multiple data sources that have an identical data structure. After the data sources have been connected, the user can add fields or tables. These changes will be incorporated across all of the data sources that have been enabled as a template data source. **Replicate Data Sources is an add-on feature that is available for purchase.**

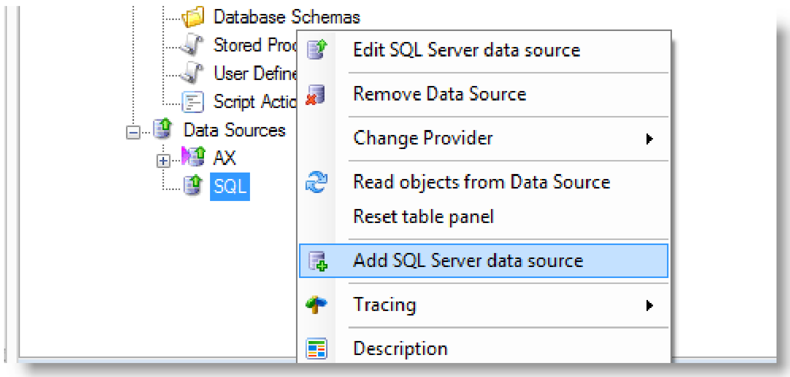
Enabling Replicate Data Sources

To enable Replicate Data Sources, the user must first go down to the **Data Source** section at the bottom of the **Data** tab. Right-click on **Data Sources**, and add the initial data source to be used as a template. For this example, a SQL data source will be added.



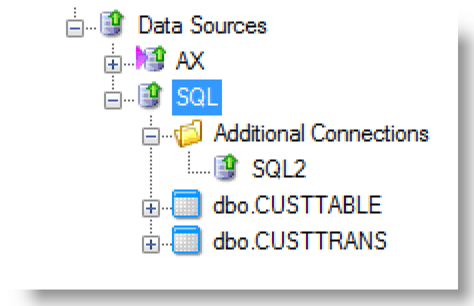
Note: Since all of the data sources will have a similar data structure, it does not matter which one is added first.

Configure the data source as needed. Once the initial data source has been configured, right-click on the data source name, and click **Add SQL Server Data Source.**



Note: When adding data sources other than SQL, it will display that data source type instead of SQL Server (ie: **Add ODBC Data Source, Add NAV Data Source, etc.)**

Configure this second data source with the needed parameters. This additional data source will then appear under the original data source in an **Additional Connections** folder.



When tables and fields are added to the original data source, these changes will automatically be propagated to all of the data sources that are configured under the **Additional Connections** folder.

Below is the result of adding the table and fields shown above to the SQL data source with the records from the SQL2 data source being brought in automatically.

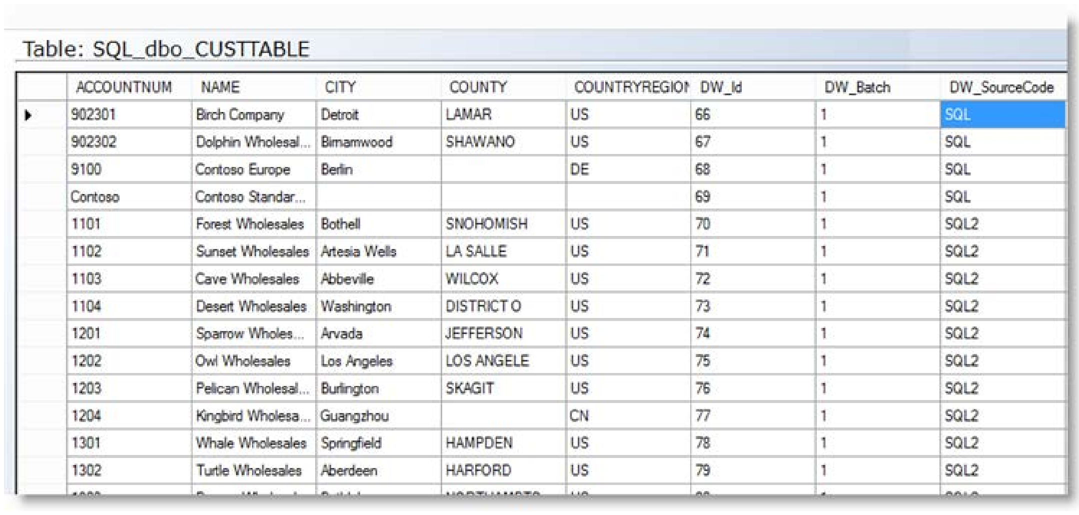


Table: SQL_dbo_CUSTTABLE

	ACCOUNTNUM	NAME	CITY	COUNTY	COUNTRYREGIOI	DW_Id	DW_Batch	DW_SourceCode
▸	902301	Birch Company	Detroit	LAMAR	US	66	1	SQL
	902302	Dolphin Wholesal...	Birnamwood	SHAWANO	US	67	1	SQL
	9100	Contoso Europe	Berlin		DE	68	1	SQL
	Contoso	Contoso Standar...				69	1	SQL
	1101	Forest Wholesales	Bothell	SNOHOMISH	US	70	1	SQL2
	1102	Sunset Wholesales	Artesia Wells	LA SALLE	US	71	1	SQL2
	1103	Cave Wholesales	Abbeville	WILCOX	US	72	1	SQL2
	1104	Desert Wholesales	Washington	DISTRICT O	US	73	1	SQL2
	1201	Sparrow Wholes...	Arvada	JEFFERSON	US	74	1	SQL2
	1202	Owl Wholesales	Los Angeles	LOS ANGELE	US	75	1	SQL2
	1203	Pelican Wholesal...	Burlington	SKAGIT	US	76	1	SQL2
	1204	Kingbird Wholesa...	Guangzhou		CN	77	1	SQL2
	1301	Whale Wholesales	Springfield	HAMPDEN	US	78	1	SQL2
	1302	Turtle Wholesales	Aberdeen	HARFORD	US	79	1	SQL2

About Adapters

An adapter is a component that enables you to easily extract and synchronize data from selected companies in Dynamics NAV, Dynamics GP, and Dynamics AX.

Dynamics NAV Adapters

The Dynamics NAV adapter simplifies the extraction of data from Microsoft Dynamics NAV.

If you connect to a Dynamics NAV database as a regular data source, you will have to apply and maintain selection rules on all tables because different companies are stored in separate tables. With Dynamics NAV adapter, you can select company accounts at a global level and apply onTX2014ly one set of selection rules. It is, however, also possible to overrule this behavior on a table by table basis.

To Add Dynamics NAV Adapters

Use the Dynamics NAV Adapter to load data from separate Dynamics company account tables in a single table.

1. On the **Data** tab, expand the preferred business unit, and then right-click **Data Sources**.

2. Point to **Add Adapter Data Sources**, and then select **Add Dynamics NAV Adapter**.

3. Enter a name for the adapter. Optionally, select **Read Aggregation Tables -SIFT** if you want to include Sum Index Flow Technology (SIFT) tables,-+*++999, and then click **OK**. This feature is not recommended for most clients.

You can now choose the provider which contains the data sources you want to connect to.

To Add an MS SQL Provider

1. Right-click the adapter and select **Source Providers**. Then select **Microsoft SQL Provider**.

2. In the **Server Name** field, enter the location of the server.

3. In the **Database** field, enter the name of the database.

4. Specify the authentication mode. The default setting is **Windows authentication**. If you choose

SQL Server authentication, you are prompted for a user name and a password.

5. In the **Command Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the database. The recommended value is 0 to disable the timeout. In the **Connection Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the server.

6. If you want to add additional connection strings, click the **Additional Connection Properties** button. In the **Connection String Properties** window, type the preferred connection strings, and then click **OK**.

To Add Navision Native Database Sources

When you want to retrieve data from Navision databases hosted in a Native Navision server environment, you will have to use ODBC. The Navision ODBC driver must be installed and configured prior to adding the Native NAV data source.

Note:If you are connecting to NAV through an ODBC connection, you must be using the 32-bit version of TX2014 as the NAV ODBC driver only supports 32-bit connections.

1. On the **Data** tab, expand the preferred business unit, and then right-click **Data Sources**.

2. Point to **Adapter Data Sources**, and select Add **Dynamics NAV Adapter**.

3. Select Wizard Setup.

4. Select Navision Native.

5. In the **Name** field, type the name of the data source.

6. In the **DSN Name**, select the ODBC connection that you have configured for the data source. In the Escape Character list, select the escape character specific to your ODBC driver. The Text Type Behavior fields are used to control how the ODBC driver handles text. These fields are optional. You have the following options:

Set Length	Specifies an exact text string length
Set Variable Length	True, if you want a variable text string length
Set Unicode	True, if you want to use Unicode

7. In **Set Number of Decimals**, specify a fixed number of decimals. This field is optional.

Note: The Convert Out of Range Dates to MS SQL/min max is not available for Navision native databases.

8. Select **Use low compatibility mode** if you have trouble retrieving data from the database.

9. In the **Command Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the database.

10. In the **Connection Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the server.

11. If you want to add additional connection strings, click the Additional Connection Properties button. In the Connection String Properties window, type the preferred connection strings, and then click **OK**.

To Change the Dynamics NAV Company Table

By default, when you add a Dynamics NAV adapter, the company account table is set to dbo.Company.

However, it is possible to change the account table. This is generally not recommended.

1. On the Data tab, expand **Business Units**, expand the preferred business unit, and then expand **Data Sources**.

2. Right-click the preferred Dynamics NAV adapter, and then select **Edit Account Table**. A message is displayed saying ,"Retrieving database structure".

3. In the **Table** list, select the account to table that you want to use.

4. In the **Name Field** list, select the field that contains the account name, and then click **OK**.

To Load and Select Data from Dynamics NAV Data Sources

1. On the **Data** tab, expand **Business Units**, expand the preferred business units, and then expand **Data Sources.**

2. Right-click the Dynamics NAV adapter containing the desired data, and then select **Read objects**. The Tables pane displays all tables, fields, and views.

3. In the **Tables** pane, select the tables, fields, and views you want to extract to your staging database.

4. There are two ways of viewing the data: Alphabetical view, which displays all tables alphabetically, and Group view, where you specify how many tables each group must contain.

To view data in groups, enter the number of tables in each group in the **Group View** field, and then click **Group View**. You can then group the tables alphabetically or by specifying the number of tables you want in each group. To view data alphabetically, click **Alphabetical View**.

The tables and fields are displayed in the data source tree and in the staging database tree.

To Set Up Dynamics NAV Companies

When you have added a Dynamics NAV adapter and specified a provider, you need to set up accounts representing the companies requiring the extracted data.

To set up accounts:

1. Right-click the adapter, and then choose **SetUp Accounts**. A dialog is displayed that shows all companies in the database.

2. In the **Template** list, select the company account you want to use as template for the table and column structure. If you are only selecting one company, then the template company must match the company that is selected.

3. Select **Use** to specify whether to retrieve data from the company.

4. In the **Default Table Usage** list, specify the order in which tables are retrieved and read. You have the following options:

Primary	Data from this company account is read and retrieved first
Secondary	Data from this company account is read and retrieved after the primary account if they have not already been retrieved from the primary account
None	Tables from this company are not retrieved, unless you specify at table level that you want to retrieve data from a specific table

To Change Dynamics NAV Schemas

You can change the schema for the entire Dynamics NAV adapter or for individual tables that belong to the adapter.

1. On the **Data** tab, expand **Business Units**, expand preferred business unit, and then expand Data Sources.

2. Right-click the NAV adapter whose schema you want to change, and then select **Change Schema**.

- OR

Expand the NAV adapter, and then select the table whose schema you want to change.

3. In the **Select Schema** to change list, select the schema you want to change, and then select **Change Schema**.

4. In the **New Schema Name** field, enter a name for the schema, and then click **OK**.

To Modify Table Usage on Dynamics NAV Tables

When you set up accounts, you specify the default order in which data is retrieved from the individual accounts. However, it is possible to specify a different order of priority for individual tables.

1. On the **Data** tab, expand **Business Units**, and then expand the preferred business unit.

2. Expand **Data Sources**, right-click the NAV adapter that contains the tables whose priority you want to change, and then select **Modify Table Usage**. The company accounts and the usage of all tables will be displayed.

3. Right-click the field that contains the setting for the table and the account for which you want to change priority of data retrieval. You have the following options:

Default	Data from this table is read and retrieved first
Primary	Data from this table is read and retrieved first
Secondary	Data from this table is read and retrieved after the primary table if they have not already been retrieved from the primary table
None	Data from this table is not retrieved
1-9	Specify the order of priority in the range from 1-9.
Enter priority	If the order of priority exceeds the numbers 1-9, you can specify additional numbers.

4. Click **OK**.

To Modify the Usage of a Single Dynamics NAV Table

You can change the order in which data is retrieved from individual tables.

1. On the **Data** tab, expand **Business Units**, and then expand the preferred business unit.

2. Expand **Data Sources**, right-click the NAV adapter that contains the table whose priority you want to change, and then select the preferred table.

3. Right-click the table and select **Modify Single Table Usage**. The company accounts and the usage specified in Setup Company Accounts will be displayed.

4. Right-click the field containing the settings for the table, and then specify the table usage. You have the following options:

Default	Data is retrieved based on the settings specified when you set up the company accounts.
Primary	Data from this table is read and retrieved first
Secondary	Data from this table is read and retrieved after the primary table if they have not already been retrieved from the primary table
None	Specify the order of priority in the range from 1-9
Enter priority	If the order of priority exceeds the numbers 1-9 you can specify additional numbers here

5. Click **OK**.

Dynamics AX Adapters

The Dynamics AX adapter simplifies the extraction of data from Dynamics AX.

If you connect to a Dynamics AX database as a regular data source, you will have to apply and maintain selection rules on all tables. With TX2014Dynamics AX adapter, you can select company accounts at a global level. You can, however, override this behavior on a table by table basis.

The adapter also extracts any virtual company accounts, including, table collections, and tables that are set up in the source database. The information can then be used in dimensions and cubes.

Furthermore, the adapter extracts all Base Enumerations and their associated labels and supports synchronization with the back-end application.

To Import XPO Files into Dynamics AX

The Dynamics AX adapter is only available if the .xpo file has been imported into Dynamics AX.

1. Import the .xpo file into Dynamics AX.

2. Compile the imported project within Dynamics AX.

3. Run all four classes in Dynamics AX to populate the tables.

4. Add a Dynamics AX adapter to your TX2014 project. For more information, see Add Dynamics AX Adapters.

For detailed instructions on how to import files, compile projects, and run classes in Dynamics AX, see the Installation documentation on the JetReports website.

Add Dynamics AX Adapters

Use the Dynamics AX Adapter to load data from separate Dynamics AX company accounts tables in a single table.

1. On the **Data** tab, expand the preferred business unit, and then right-click **Data Sources**.

2. Select **Add Adapter Data Sources**, and then select **Add Dynamics AX Adapter**.

3. Enter a name for the adapter, and then click **OK**.

You can now choose the provider which contains the data source you want to connect to.

To Add a Microsoft SQL Server Provider

1. Right-click the adapter, and select **Source Providers**. Then select **Add MS SQL Provider**.

2. In the **Server Name** field, enter the location of the server.

3. In the **Database** field, enter the name of the database.

4. Specify the authentication mode. The default setting is **Windows authentication**. If you choose **SQL Server authentication**, you are prompted for a user name and a password.

5. In the **Command Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the database.

6. In the Connection Timeout field, specify the number of seconds to wait before terminating the attempt to connect to the server, and then click OK.

To Add an Oracle Provider

1. Right-click the adapter, and select **Source Providers**. Then select **Oracle Provider**.

2. In the **TNS alias** field, type the alias that identifies the database.

3. In the Owner list, select the owner of the database.

4. Specify the authentication mode. When you select **Oracle authentication,** you are prompted for a user name and a password.

5. Select **Convert Out of Range Dates to MS SQL min/max** if you want to convert all dates older than January 01, 1753 to 01-01-1753.

6. In the **Connection Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the server.

7. In the **Command Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the database.

8. If you want to set the character encoding to either Unicode or Non-Unicode, select **Force Character Setting**. Then select the preferred character encoding in the list.

Note: Forcing character encoding may affect performance.

If you want to add additional connection strings, click the **Additional Connection Properties** button. In the Connection String Properties window, type the preferred connection strings, and then click **OK**.

To Set the Account Table

Before you can continue with setting up accounts, you have to verify that the company account table is correct.

• Right-click the adapter, and then choose **Set Account Table**. The table DATAAREA and the field ID are selected by default. Click **OK**.

To Set Up Dynamics AX Companies

When you have added a Dynamics AX adapter and specified a provider, you need to set up the accounts.

To set up accounts:

1. Right-click the adapter, and select **SetUp Accounts**. An information message is displayed, which lists the accounts that have been added. Click **OK**.

2. In the Accounts table, select the accounts from which you want to retrieve data. Select **Use** to specify whether to retrieve data from the account.

3. In the **Default Table Usage** list, specify the order in which data from the tables is retrieved and read, and then click **OK**. You have the following options:

Primary	Data from this company account is read and retrieved first
Secondary	Data from this company account is read and retrieved after the primary account if they have not already been retrieved from the primary account
None	Tables from this company are not retrieved unless you specify at the table level that you want to retrieve data from a specific table. For more information, see To modify table usage in Dynamics AX tables.

To Load and Select Data from Dynamics AX Data Sources

1. On the **Data** tab, expand **Business Units**, expand the preferred business units, and then expand **Data Sources**.

2. Right-click the Dynamics AX adapter you want to select data from, and then select **Read objects**. The Tables pane displays all tables and fields.

3. In the **Tables**, select the tables and fields you want to extract to your staging database.

4. There are two ways of viewing the data: Alphabetical view, which displays all tables alphabetically, and Group view, where you specify how many tables each group must contain. To view data in groups, enter the number of tables in each group in the Group view field, and then click **Group view**. You can then group the tables alphabetically or by specifying the number of tables you want in each group. To view data alphabetically, click **Alphabetical view**.

The tables and fields are displayed in the data source tree and in the staging database tree.

To Add Dynamics AX Virtual Table References

1. On the **Data** tab, expand **Business Units**, expand the preferred business units, and then expand **Data Sources.**

2. Expand the AX adapter that contains the table to which you want to add a virtual table reference, right-click the table, and then choose **Add Virtual Table Reference**.

3. In the Add Virtual Table Reference window, select the preferred virtual tables, and then click **OK**.

To View Dynamics AX Table Information

TX2014 can retrieve table information directly from your Dynamics AX database.

1. On the **Data** tab, expand **Business Units**, expand the preferred business units, and then expand **Data Sources**.

2. Expand the Dynamics AX adapter that contains the table you want to view information about, right- click the table, and then select **View Table Information**.

The three tabs in the View Table Information dialog contain the following information:

Name	Specifies the name of the field as it appears in the database
Label	Specifies the name of the field as it appears in the user interface
Help Text	Contains the help text for the field
EDT Name	Specifies the name of the extended Data Type if applicable
Enum Name	Specifies the name of the enumeration if applicable
System	Specifies whether the table is a system table or visible in the user interface

External Table	Specifies the name of the table the selected table is related to
Directions	Specifies whether the selected table is the child or the parent in the relation
Field	Specifies which field in the selected table that relates to a field in the related table
External Field	Specifies the field on the related table
Relation Type	Specifies the type of relation. **Field** specifies relation fields without conditions. **This-Fixed** specifies relation fields to restrict the records in the primary table. **ExternFixed** specifies relation fields that restrict the records in the related table

Company	The name of the company account
Virtual Company	The name of the Virtual Company that contains tables shared by several company accounts

To View Dynamics AX Enum Table Information

All Enum values in Dynamics AX are represented as integers in the tables. However, you can see the corresponding literal values by viewing the enumeration table information.

1. On the Data tab, expand Business Units, expand the preferred business units, and then expand Data Sources.

2. Expand the Dynamics AX adapter that contains the table you want to view information about, right- click the table, and then select Preview Enum Table.

To Change Dynamics AX Schemas

1. On the **Data** tab, expand **Business Units**, and then expand the preferred business unit.

2. Expand **Data Sources**, right-click the AX adapter that contains the table whose priority you want to change, and then select **Change Schema**.

3. In the **Select Schema To Change list**, select the schema you want to change.

4. In the **New Schema Name** field, enter a name for the schema.

To Modify Table Usage on Dynamics AX Tables

When you set up accounts, you specify the default order in which data is retrieved from the individual accounts. However, it is possible to specify a different order of priority for individual tables.

1. On the **Data** tab, expand **Business Units**, and then expand the preferred business unit.

2. Expand **Data Sources**, right-click the AX adapter that contains the table whose priority you want to change, and then select **Modify Table Usage**. The company accounts and the usage of all tables will be displayed.

3. Right-click the table and account field with the setting you wish to change the order of priority on for data retrieval. You have the following options:

Default	Data from this table is read and retrieved first
Primary	Data from this table is read and retrieved first
Secondary	Data from this table are read and retrieved after the primary table if they have not already been retrieved from the primary table
None	Data from this table is not retrieved
1-9	Specify the order priority in the range from 1-9
Enter priority	If the order of priority exceeds the numbers 1-9, you can specify additional numbers

4. Click **OK**.

To Modify the Usage of a Single Dynamics AX Table

If you want to change the order of priority in which data is retrieved on a single table, you can do so from the individual table.

1. On the **Data** tab, expand **Business Units**, and then expand the preferred business unit.

2. Expand **Data Sources**, right-click the AX adapter that contains the table whose priority you want to change, and then select the preferred table.

3. Right-click the table, and select **Modify Single Table Usage**. The company accounts and the usage specified in Setup Company Accounts will be displayed.

4. Right-click the field that contains the setting for the table, and then specify the table usage. You have the following options:

Default	Data is retrieved based on the settings specified when you set up the company accounts
Primary	Data from this table is read and retrieved first
Secondary	Data from this table is read and retrieved after the primary table if they have not already been retrieved from the primary table
None	Data from this table is not retrieved
1-9	Specify the order of priority in the range form 1-9
Enter priority	If the order of priority exceeds the numbers 1-9, you can specify additional numbers here.

Dynamics GP Adapters

The Dynamics GP adapter simplifies the extraction of data from Microsoft Dynamics GP.

If you connect to a Dynamics GP database as a regular data source, you will have to apply and maintain selection rules on all tables because different companies are stored in separate databases. WithTX2014 Dynamics GP Adapter, you can select company accounts at a global level and apply only one set of selection rules. It is, however, also possible to overrule this behavior on a table by table basis.

To Add Dynamics GP Adapters

Use the Dynamics GP Adapter to load data from separate Dynamics company account databases in a single table.

1. On the **Data** tab, expand the preferred business unit, and then right-click **Data Sources**.

2. Point to **Add Adapter Data Sources**, and then select **Add Dynamics GPAdapter**.

3. Enter a name for the adapter.

4. In the **Server name** field, enter the location of the server where Dynamics GP resides.

3. In the **Database** field, enter the name of the database (this should be the DYNAMICS database).

4. Specify the authentication mode. The default setting is **Windows authentication**. If you choose **SQL Server authentication**, you are prompted for a user name and a password.

5. In the **Command Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the database. The recommended value is 0 to disable the timeout. In the **Connection Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the server.

6. If you want to add additional connection strings, click the **Additional Connection Properties** button. In the **Connection String Properties** window, type the preferred connection strings, and then click **OK**.

7. The next window will be the GP Company Table Setup. All settings should be left as default and click **OK.**

To Set Up Dynamics GP Companies

After you have added a Dynamics GP adapter and specified a provider, you will set up the accounts which represent the companies for which data will be extracted from the data source.

To set up accounts:

1. Right-click the adapter, and then choose **Read Dynamics GP Companies**. A dialog is displayed that shows all companies in the database.

2. In the **Template** list, select the company account you want to use as template for the table and column structure. If you are only selecting one company, then the template company must match the company that is selected.

3. Select **Use** to specify whether to retrieve data from the company.

To Load and Select Data from Dynamics GP Data Sources

1. On the **Data** tab, expand **Business Units**, expand the preferred business units, and then expand **Data Sources.**

2. Right-click the Dynamics GP Adapter from which you want to select data, and then select **ReadObjects from Data Source**. The Tables pane on the right displays all tables, fields, and views.

3. In the **Tables** pane, select the tables, fields, and views you want to extract to your staging database.

4. There are two ways of viewing the data: Alphabetical view, which displays all tables alphabetically, and Group view, where you specify how many tables each group must contain.

5. To view data in groups, enter the number of tables in each group in the **Group view** field, and then click **Group view**. You can then group the tables alphabetically or by specifying the number of tables you want in each group. To view data alphabetically, click **Alphabetical view**.

The tables and fields are displayed in the data source tree and in the staging database tree.

Salesforce.com Adapters

With the new Salesforce adapter you can easily gain access to your salesforce data stored in your Salesforce Enterprise or Performance Edition.

To connect to your salesforce data, you will need the following information:

Username

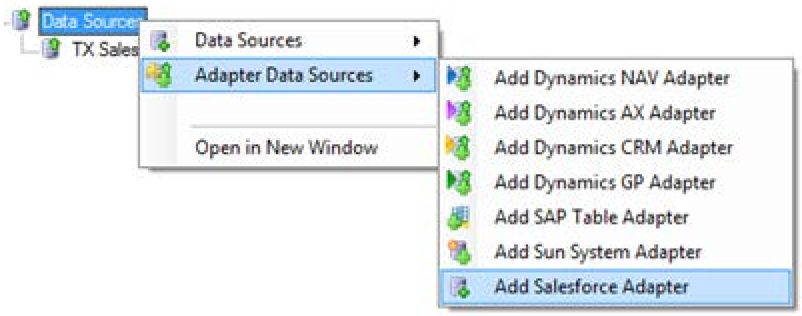
Password

Security Token

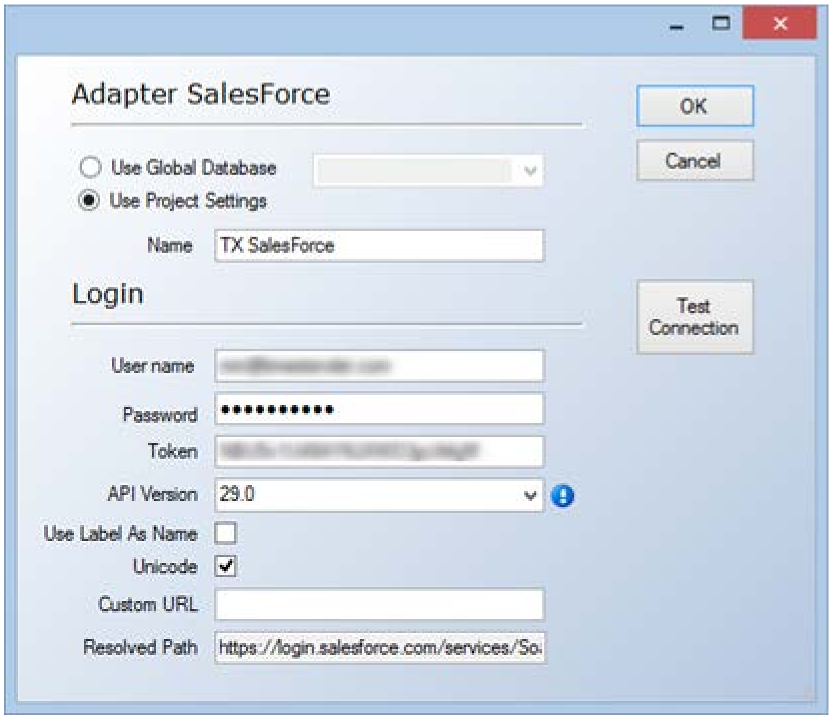
The Security Token is provided by salesforce and it will change if the password is changed. Salesforce can be configured not to use Tokens – in that case, simply leave the Token field empty.

This is how you connect to salesforce.

Select “Add Salesforce Adapter”:



Enter the relevant information:



Adapter SalesForce

OK

Cancel

( ) Use Global Database

(●) Use Project Settings

Name: TX SalesForce

Login

Test Connection

User name: [redacted]

Password: ●●●●●●●●●●●

Token: [redacted]

API Version: 29.0

Use Label As Name: ☐

Unicode: ✓

Custom URL:

Resolved Path: https://login.salesforce.com/services/So

Field explanations:

Name: The Name of the Data Source

User Name: The Salesforce username, usually an email address

Password: The Salesforce password

Token: The Salesforce Security Token

API Version: Select the Salesforce API version. The newest version should be selected, however if Salesforce make changes in the API that would prevent TX from integating with it, an earlier version could be selected.

Use Label as Name: Select to use labels as names instead of the physical system names.

UniCode: Select to make string based field Unicode ready.

Custom URL: TX will by default use the default URL to connect to salesforce. The Custom URL can be used if you want to connect to your Salesforce Sandbox (Test).

Resolved Path: If you enter a Custom URL, the Resolved Path will show you how the entire URL will end up being.

Deploying and Executing Projects

Deploying a Project

Deploying a project is the process of:

- Generating the structure of the staging database and the data warehouse

- Processing cubes

- Generating SQL code.

No data is loaded into the staging database or the data warehouse, and no cubes are processed at this time. When you successfully deploy a project, the project is automatically saved in the project repository.

Executing a Project

Executing a project is the process of loading data into the staging database, the data warehouse, and then processing the cubes.

Executing a project involves the following steps (with corresponding definitions):

1. **Transferring data**: The process of transferring data from the data source to the raw table of the staging database.

2. **Processing data**: The process of cleansing data; that is, validating the data against the business rules, and moving the validated data to the valid table. Status information is also generated at this point.

3. **Verifying data against checkpoints:** The process of checking the data that is being processed against the checkpoints you have specified. You can specify rules that will end the execution process if not met. This way, you avoid overwriting the data in your data warehouse with non-valuable data.

4. **Moving data:** The process of moving data from a business unit to a data warehouse, or from a data warehouse to a cube.

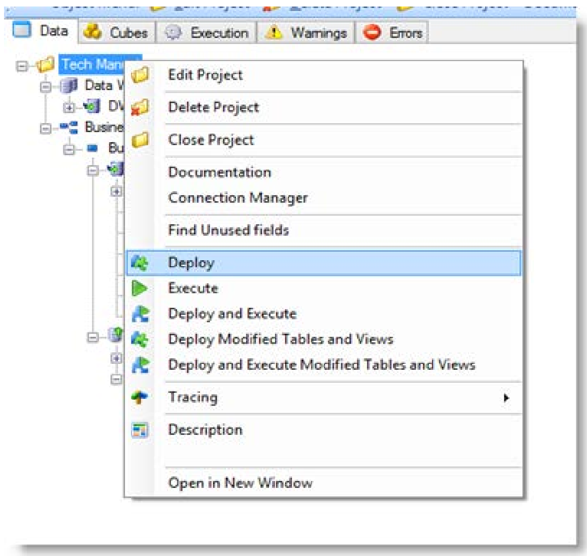
5. **Processing cubes:** The process of creating dimension hierarchies and retrieving values from the fact tables to populate the cubes with measures, including derived and calculated measures.

Manual Deployment and Execution

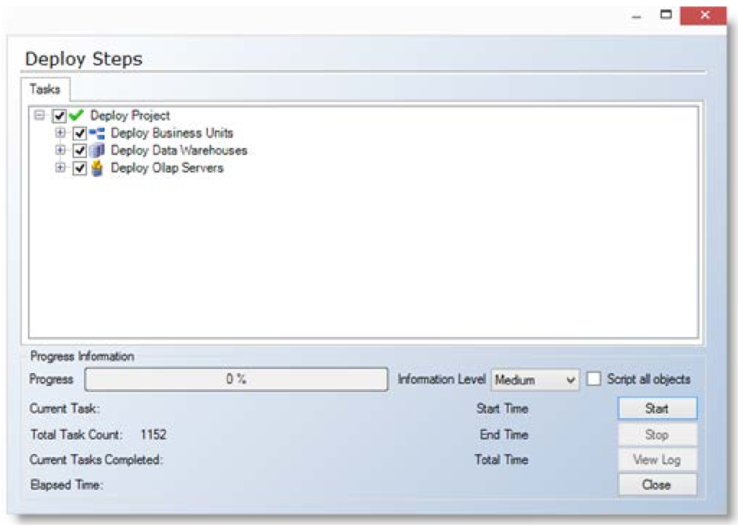
While you are developing and maintaining your project, you may want to deploy or execute project elements immediately. You then have the option of manually deploying and executing project elements all the way down to the table level.

To Deploy Individual Objects

1. On the **Data** tab, right-click the project or the project element you want to deploy, and then choose **Deploy**.



A window will appear with the following selections:



Progress: 0 %
Information Level: Medium
Script all objects: ☐
Current Task:
Total Task Count: 1152
Current Tasks Completed:
Elapsed Time:
Start Time
End Time
Total Time

2. From the **Information** level list, select the preferred level of information. The default setting is Medium, however the following options are available:

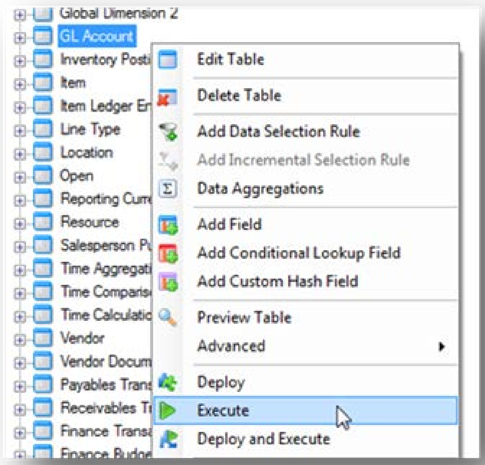
None	Displays no progress information
Low	Displays current task, the total task count, start time, end time, and total time
Medium	Displays progress information, name of the current task that is being deployed, number of completed tasks, the total number of tasks that have to be completed, start time, end time, and total time
High	Displays all deployment steps in the task window, progress information, name of the current task that is being deployed, number of completed tasks, the total number of tasks that have to be completed, start time, end time, and total time.

3. In the **Deploy Steps** dialog, click **Start** to deploy the project.

4. If there are any errors during deployment, click View Log to see which elements in the project that were not successfully deployed. Right-click **Error Information** to view an error description.

To Execute Individual Objects

1. On the **Data** tab, right-click the project or the project element you want to execute, and then choose **Execute**.



2. In the **Execute Steps** dialog, click **Start** to execute the object.

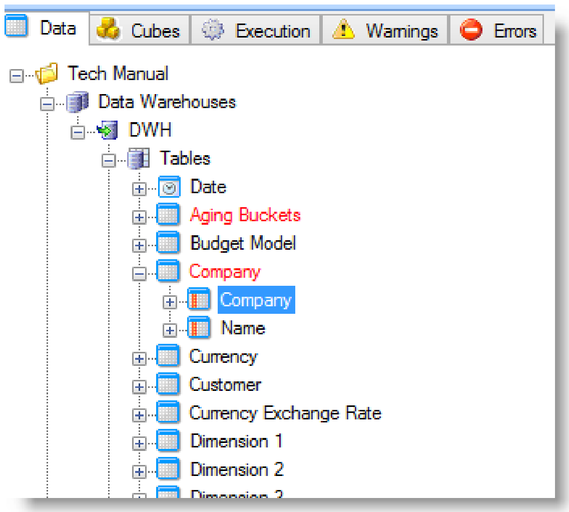
3. If there are any errors during execution, click **View Log** to see which elements in the project were not successfully executed. Right-click **Error Information** to view an error description.

Deploy/Execute Only Modified Objects

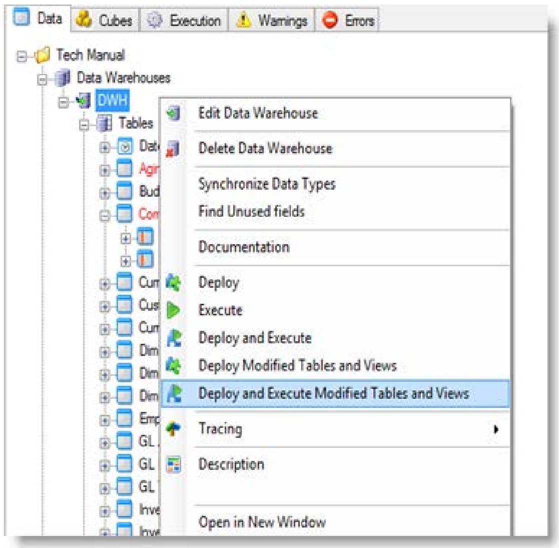
There is functionality that enables the user to deploy and execute only those objects that have been modified. This reduces processing time as there is no need to reprocess unnecessary objects.

How to Deploy/Execute Modified Objects

Note: Modified objects are highlighted in red.



1. To Deploy and Execute modified objects, right-click the Data Warehouse/Staging Database and select **Deploy and Execute Modified Tables and Views**. This selection can also be made at the project level. You also have the option to **Deploy Modified Tables and Views.** This will deploy the modified objects without executing them.



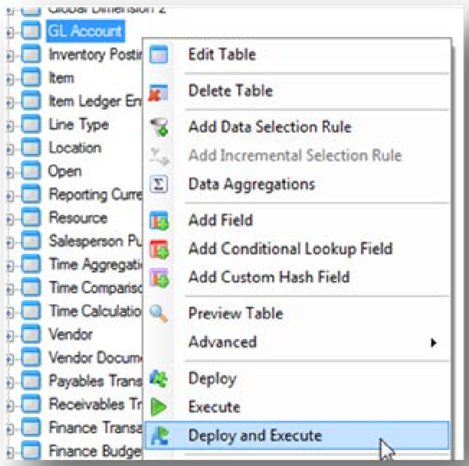
2. In the dialog box, select the **Start** button to start the process. Notice that only the modified objects are presented in the tasks pane.

Note: Once deployment has been completed and the project saved, objects will no longer be marked as changed and appear in red.

To Deploy and Execute Objects at the Same Time

It is also possible to run both the deployment and execution of an individual object at the same time without having to specify the deployment and execution phases separately.

1. 1. On the **Data** tab, right-click the project or the project element you want to execute, and then choose Deploy and Execute.

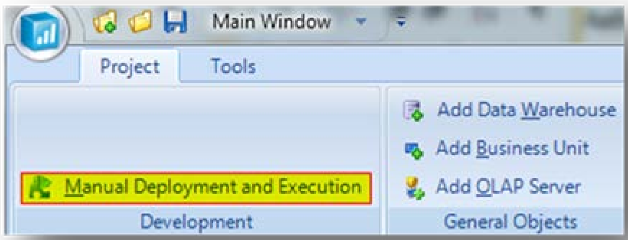


2. In the **Execute Steps** dialog, click **Start** to deploy and execute the object.

3. If there are any errors during execution, click **View Log** to see which elements in the project were not successfully executed. Right-click **Error information** to view an error description.

To Deploy and Execute the Entire Project

1. Click the “Manual Deployment and Execution” button on the **Project** ribbon.

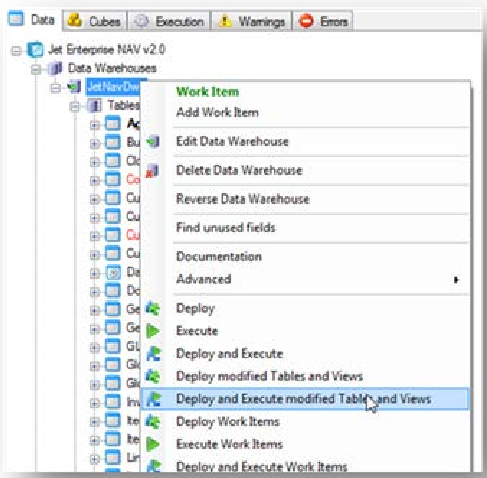


2. Click the **Start** button in the next window to start the deployment and execution process.

How to Only Deploy and Execute Modified Objects

It is possible to only deploy and execute those objects that have been modified. This can greatly cut down on the time needed to deploy changes without the user having to remember which objects were altered.

1. To Deploy and Execute modified objects, right-click the Data Warehouse/Staging Database, and select **Deploy and Execute Modified Tables and Views**.



This selection can also be made at the project level. You also have the option to **Deploy Modified Tables and Views.** This will deploy the modified objects without executing them.

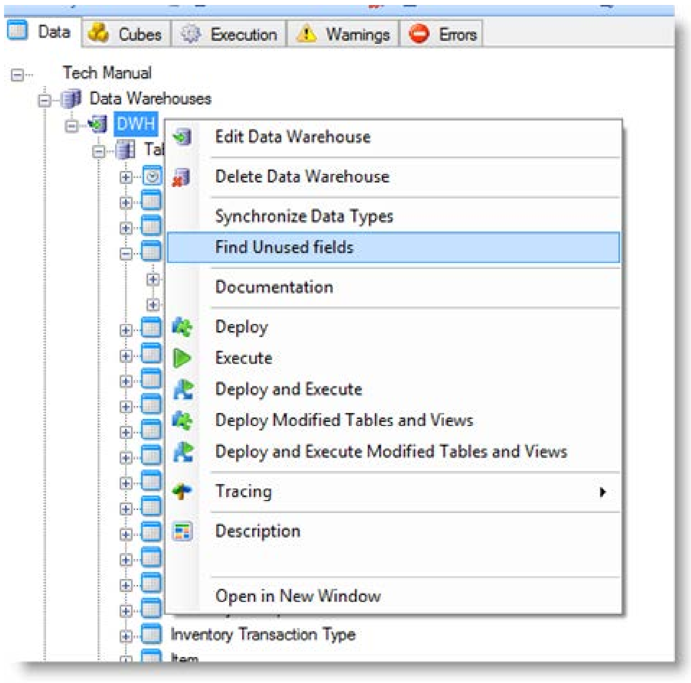
2. In the dialog box select the **Start** button to start the process. Only the modified objects are presented in the tasks pane.

Display Unused Fields

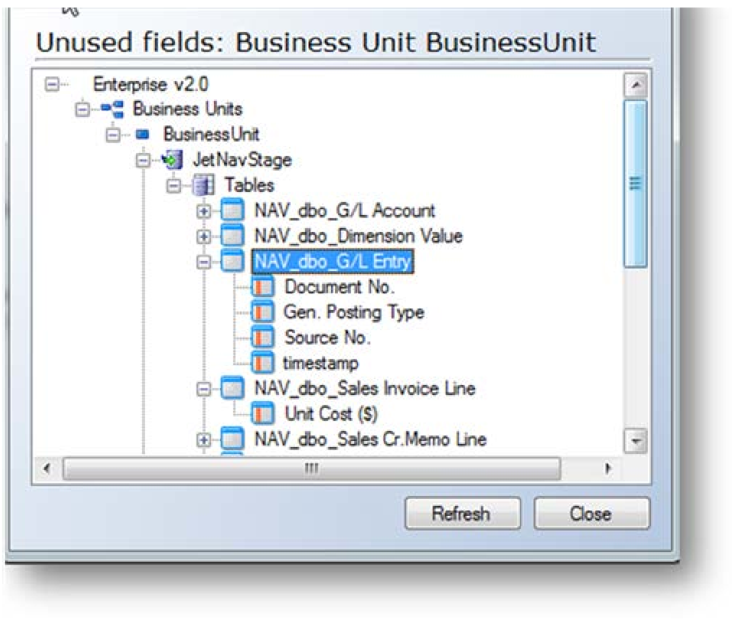
TX2014has the ability to display all unused fields in the project. This feature is useful for removing unnecessary objects from the project which decreases clutter and improves performance.

How to Display Unused Fields

1. From the data tab, right-click either the Business Unit or the Data Warehouse, and select Find Unused Fields.



A dialog appears that shows unused fields:


Unused fields: Business Unit BusinessUnit

- Enterprise v2.0
  - Business Units
    - BusinessUnit
      - JetNavStage
        - Tables
          - NAV_dbo_G/L Account
          - NAV_dbo_Dimension Value
          - NAV_dbo_G/L Entry
            - Document No.
            - Gen. Posting Type
            - Source No.
            - timestamp
          - NAV_dbo_Sales Invoice Line
            - Unit Cost ($)
          - NAV_dbo_Sales Cr.Memo Line

In the staging database, this would show fields that exist but are not:

- Promoted to the data warehouse
- Used as a data selection rule
- Used as an incremental selection rule
- Used in a strongly typed custom table
- Used in a SQL snippet
- Used as a conditional lookup in another table

In the data warehouse database, this would show fields that exist but are not:

- Promoted to the OLAP Cubes as measures or dimensions
- Used as a data selection rule
- Used as an incremental selection rule
- Used in a strongly typed custom table
- Used in a SQL snippet
- Used as a conditional lookup in another table

About Business Units

In TX2014, a business unit is any part of your organization that you want to treat as a separate entity in your project. For example, you may want to treat a company headquarters and each of its subsidiaries as separate business units.

Each business unit in your project has its own staging databases and its own data sources.

When you create a business unit, it contains the following elements:

• Default staging database

• Tables

• Views

• Stored procedures

• User defined functions

• Data sources

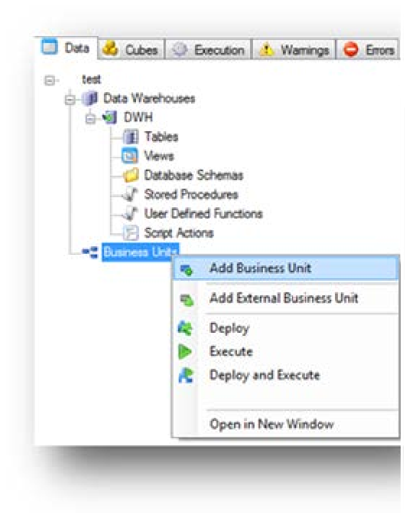
• Data Mapping

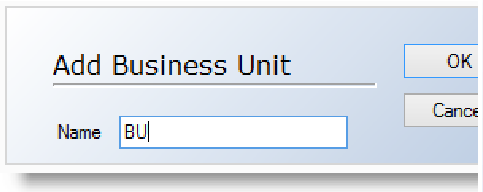
To Add Business Units

1. On the **Data** tab, right-click **Business Units**, and choose **Add Business Unit**.

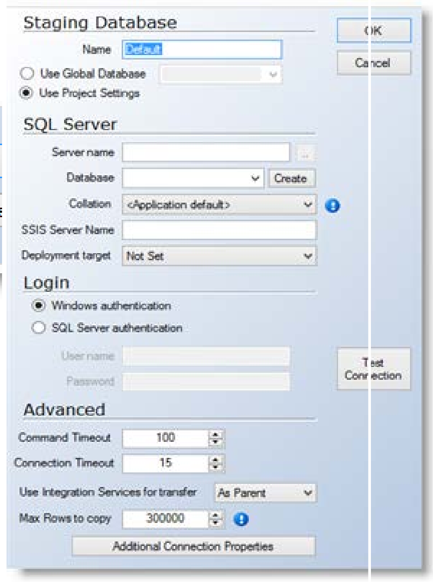
2. In the **Add Business Unit** dialog, type a name for the business unit, and then click **OK**.

When you create a business unit, you always have to specify a staging database. The **AddStagingDatabase** dialog is therefore displayed immediately after you have created the business unit.





Name: BU



Name: Default
Collation: <Application default>
Deployment target: Not Set
Command Timeout: 100
Connection Timeout: 15
Use Integration Services for transfer: As Parent
Max Rows to copy: 300000

To Specify a Staging Database

1. In the Name field, type a name for the staging database. The name cannot exceed 15 characters in length.

2. In the **Server Name** field, type the name of the database server.

3. In the Database list, select the preferred database from the drop-down list. If you wish to create a new database, then type a name for the new database, and then click **Create**.

4. Specify the authentication mode. The default setting is **Windows authentication**. If you choose

SQL Server authentication, you are prompted for a user name and a password.

5. Click **Test Connection** to verify that the connection is working.

6. In the **Command Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the database. The recommended value for this is 0 to disable the command timeout.

7. In the **Connection Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the server.

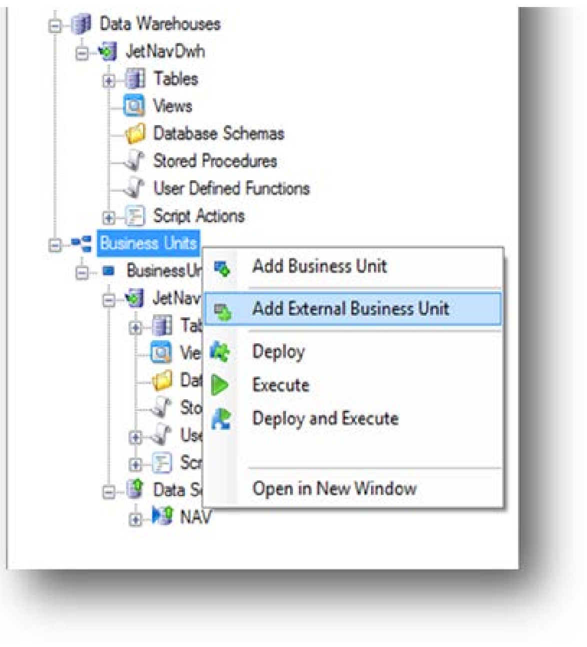
If you want to add additional connection strings, click the **Additional Connection Properties** button. In the **Connection String Properties** window, type the preferred connection strings, and then click **OK.**

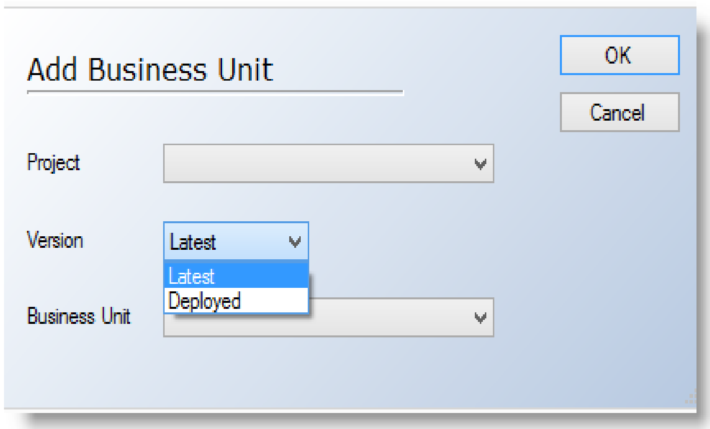
External Business Units

You can add business units from other projects to your project. This enables reuse of business units across different projects.

To Add External Business Units

1. On the **Data** tab, right-click **Business Units**, and then choose **Add External Business Unit**.





2. In the **Project** list, select the project to amend.

3. Select which version of the business unit you want to import. You have the following options:

Latest	Adds the last saved version of the project, which does not always correspond to the last deployed version
Deployed	Adds the last deployed version of the project

4. In the Business Unit list, select the business unit you want to add, and then click **OK**. The business unit is displayed as a separate entity in the project tree.

To Synchronize External Business Units

1. On the **Data** tab, expand **Business Units**, and then right-click the external business unit you want to synchronize.

2. Select **By ID** to synchronize fields by ID, or select **ByName** to synchronize fields by name.

Staging Databases

A business unit always contains a staging database. The staging database stores the selected data from the data sources. Additionally, many of the validation and transformation processes take place in the staging database. This ensures that the cleansing process has limited impact on the transaction database.

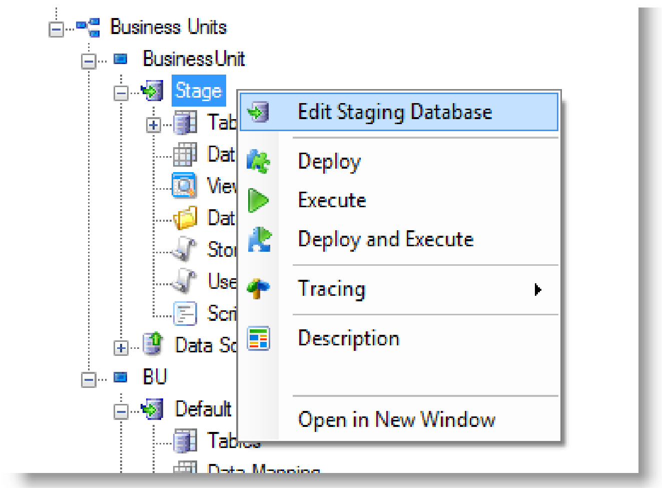
You create table relationships and mapping tables on the staging database. It is also possible to add views, stored procedures, and user defined functions to the staging database. When you execute a project, all invalid rows are stored in either the Warnings table or the Errors table.

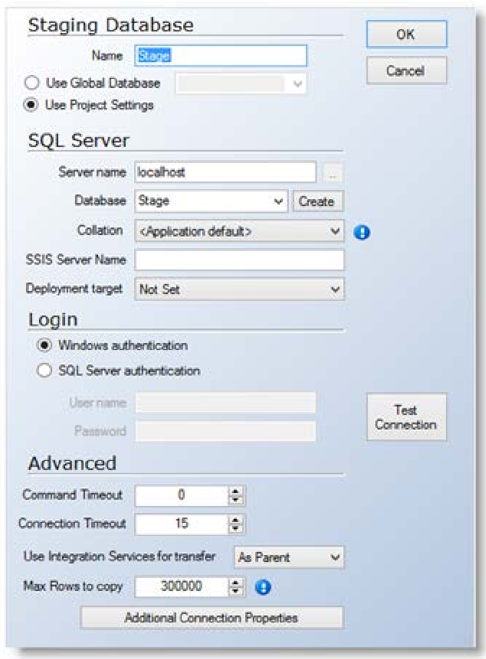
To Specify a Staging Database

When you create a new business unit, you are required to specify a staging database.

1. On the **Data** tab, expand the preferred business unit.

2. Right-click **Default**, which is the default staging database, and then select **Edit** to display the Staging Database dialog.




Staging Database

Name	Stage
○ Use Global Database	
● Use Project Settings	

SQL Server

Server name	localhost
Database	Stage
Collation	<Application default>
SSIS Server Name	
Deployment target	Not Set

Login

● Windows authentication
○ SQL Server authentication

User name	
Password	

Advanced

Command Timeout	0
Connection Timeout	15
Use Integration Services for transfer	As Parent
Max Rows to copy	300000

3. In the **Name** field, type a name for the staging database. The name cannot exceed 15 characters in length.

4. In the **Server Name** field type the name of the database server.

5. In the Database list, select the preferred database from the drop-down list. If you wish to create a new database, type a name for the new database, and then click **Create**.

6. Specify the authentication mode. The default setting is **Windows authentication**. If you choose **SQL Server authentication**, you are prompted for a user name and a password.

7. Click **Test Connection**, to verify that the connection is working.

8. In the Command Timeout field, specify the number of seconds to wait before terminating the attempt to connect to the database. The recommended value is 0 to disable the command timeout.

9. In the **Connection Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the server.

If you want to add additional connection strings, click the **Additional Connection Properties** button. In the **Connection String Properties** window, type the preferred connection strings, and then click OK.

Creating Relationships

To define how the tables in the staging database are related, you have to specify relationships. Relationships ensure that data which is used in different tables remains valid when records are changed. This is the first step to ensure referential integrity.

For example, if you have a relationship between an Order and an Order Detail, the Order Detail cannot exist if the order is deleted.

In TX2014, you create parent-child relationships from the master to the detail. This can serve multiple purposes. It is possible to set the relationship type so that records which appear in the Order Detail table with no foreign key present in the Order table are excluded from the staging database or data warehouse. Setting relationships also facilitates the ease of transferring data from one table to another through the use of conditional lookups (See To Add Conditional Lookup Fields) where the relationships have already been defined.

To Create Relationships

1. On the **Data** tab, expand the preferred business unit, and then expand the staging database.

2. Expand **Tables**, expand the table you want to create a relation from, and then expand the table that you want to create a relation to.

3. In the table that you want to create a relation from, select the preferred field.

4. Left-click the field, and drag and drop the field to the matching field in the table that you want to create a relation to. For example, you could drag the “No.” field from the Order table and drop it on the “Order No.” field in the Order Detail table if both of these fields represented the order number.

Note: You can only create relations between fields that have compatible data types.

When you have created a relation, the relation is displayed in the staging database tree below the parent of the relationship. If you expand the relation, you can view the related fields.

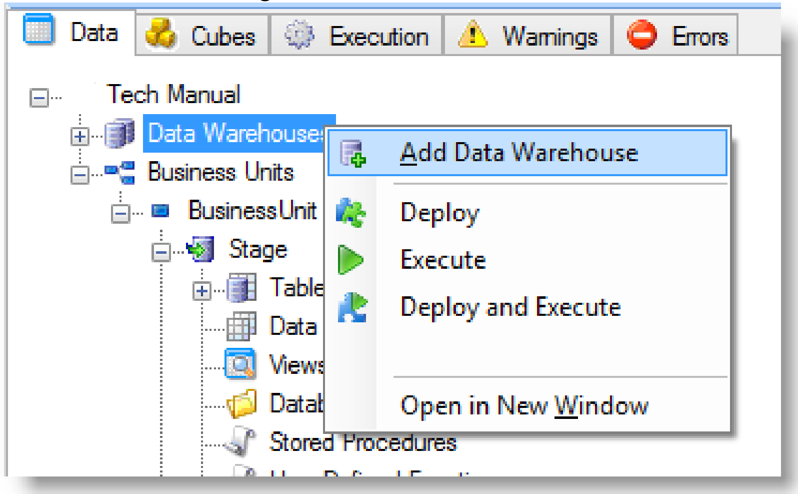
Data Warehouses

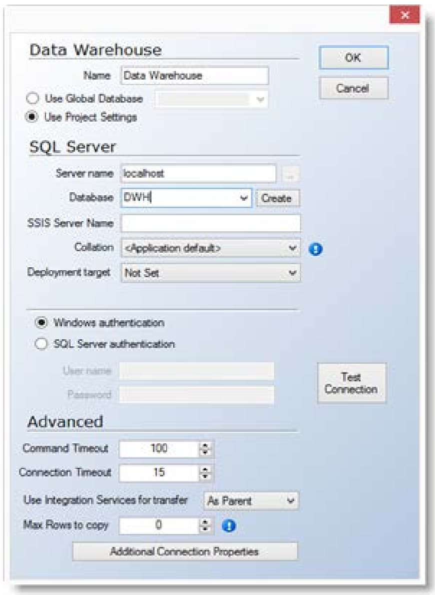
Data is extracted from the staging database and transferred to the Data Warehouse. This is loaded into the valid data warehouse tables after applying data transformation and data cleansing. You can then use the data warehouse for queries and analysis.

You can also move data from different staging databases into the same data warehouse for data consolidation. This is useful, for example, when you want to combine table fields from different business units.

To Create Data Warehouses

1. On the **Data** tab, right-click **Data Warehouse**, and then select **Add Data Warehouse**.





Data Warehouse

Name: Data Warehouse

[ ] Use Global Database
[✓ Use Project Settings]

SQL Server

Server name: localhost

Database: DWH

SSIS Server Name:

Collation: <Application default>

Deployment target: Not Set

[✓ Windows authentication]
[ ] SQL Server authentication

User name:

Password:

Advanced

Command Timeout: 100

Connection Timeout: 15

Use Integration Services for transfer: As Parent

Max Rows to copy: 0

2. In the **Name** field, type a name for the data warehouse. The name cannot exceed 15 characters in length.

3. In the **Server Name** field, type the name of the server that you want to store the database on. If it is a server instance, type the server name and the instance name.

4. In the **Database** field, select an existing database, or type the name of a new database, and then click **Create**.

5. If your SSIS Server is installed under a different name than the database, enter the name in the **SSIS Server Name** field.

6. Specify the authentication mode. The default setting is **Windows Authentication**. If you choose **SQL Server Authentication**, you are prompted for a user name and a password.

7. Click **Test Connection** to verify that the connection is working.

8. In the **Connection Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the server. The recommended setting 0 zero to disable Connection Timeout.

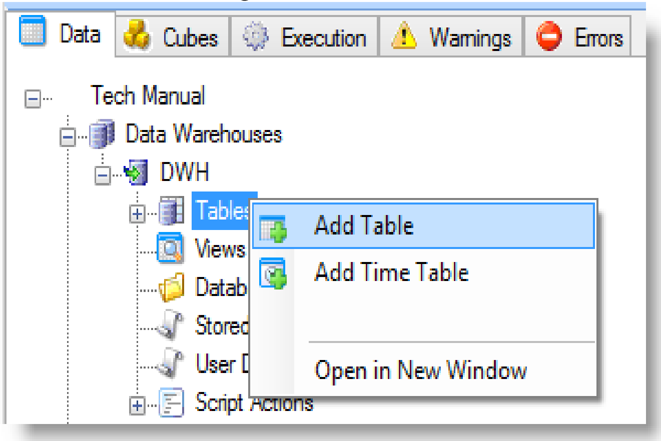
9. In the Command Timeout field, specify the number of seconds to wait before terminating the attempt to connect to the database.

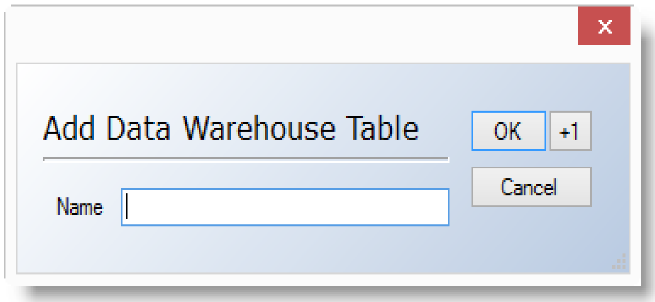
10. If you want to add additional connection strings, click the **Additional Connection Properties** button. In the **Connection String Properties** window, type the preferred connection strings, and then click **OK**.

To Add Tables To A Data Warehouse

1. Expand **Data Warehouses**, and then expand the preferred data warehouse.

2. Select **Tables**, right-click, and then select **Add table**.





3. In the **Name** field, type a name for the table, and then click **OK**.

Note: You can also move tables from a staging database to the data warehouse by dragging and dropping the tables. See To Move Data from the Staging Database to the Data Warehouse.

To Assign Primary Keys

All tables in your project can have a primary key that uniquely identifies every row in the table. If you are consolidating data from different business units, you must assign primary keys to avoid duplicate values in your dimensions.

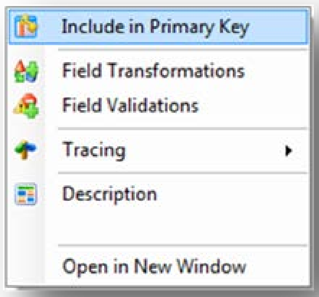
1. Expand **Business Units**, expand the preferred business unit, and then expand the staging data-base.

   - OR -

   Expand **Data Warehouses**, expand the preferred data warehouse, and then select**Tables**.

2. Expand the table that you want to modify.

3. Right-click the field you wish to modify, and then select **Include in Primary Key**. You can base the primary key on more than one field.



To Add Custom Fields

You can add custom fields to both the staging database and the data warehouse.

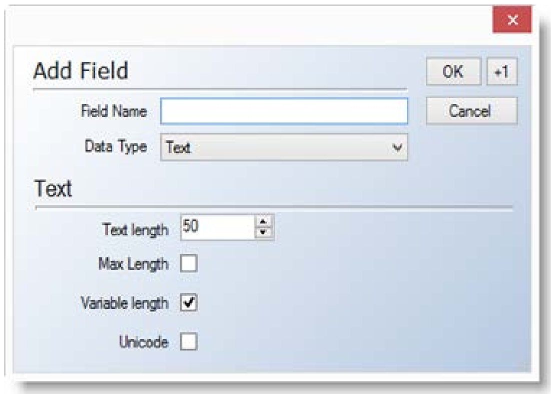
1. Expand the preferred business unit, expand the staging database, and then expand Tables.

   - OR -

   Expand the preferred data warehouse, and then expand Tables.

2. Select the table to which you want to add a custom field, right-click, and then select **Add Field**.

The **Edit Field** dialog is displayed:



3. In the **Field name** field, type a name for the field.

4. In the **Data type** list, select the preferred data type. You have the following options:

• Text

• Numeric

• Boolean

• Date and time

• Unique identifier

5. Define the attributes of the selected data type, and then click OK. You have the following options:

Text	Text length	Specifies the maximum number of characters the field can contain
Text	Variable length	Specifies that the field can be of variable length
Text	Unicode	Specifies whether to use Unicode character-encoding
Numeric	Number of decimals	Specifies the number of decimals in the field

To Add Conditional Lookup Fields

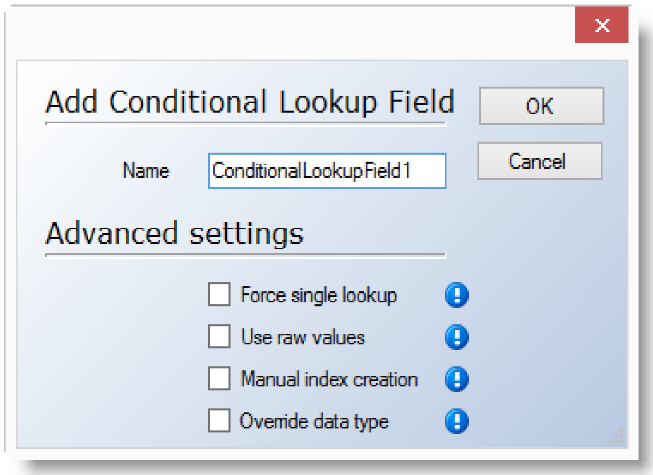
Lookup fields are used to add a field to a table in order to retrieve the value of the field in another table. The process of adding a conditional lookup field consists of a number of steps described below.

To Create a Field

1. Expand **Business Units**, expand the preferred business units, and then expand the staging database.

2. Expand **Tables**, and then select the table you wish to modify.

3. Right-click the table, and then select **Add Conditional Lookup Field.**


Add Conditional Lookup Field

Name: ConditionalLookupField1

Advanced settings

- [ ] Force single lookup
- [ ] Use raw values
- [ ] Manual index creation
- [ ] Override data type

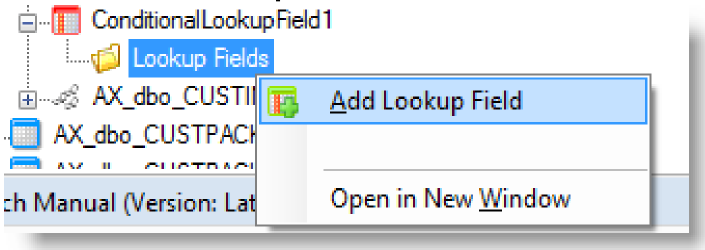
4. In the **Name** field, type a name for the lookup field, and then click **OK**. The field is added to the table tree.

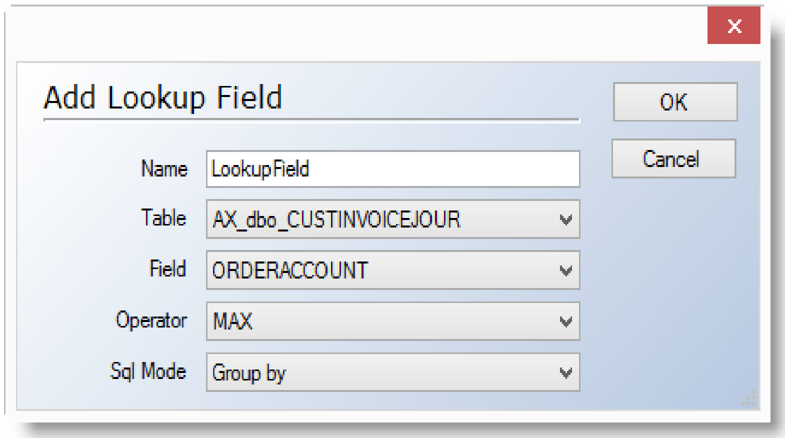
Specify the Lookup Field

The next step is to specify the lookup field that contains the values to be used in the field you just created.

1. Expand the field, and then right-click **Lookup Field**.

2. Select **Add Lookup Field**.




Add Lookup Field

Name: LookupField

Table: AX_dbo_CUSTINVOICEJOUR

Field: ORDERACCOUNT

Operator: MAX

Sql Mode: Group by

3. In the **Name** field, type a name for the field.

4. In the **Table** list, select the table containing your desired field.

5. In the **Field** list, select the field containing your desired value.

6. In the **Operator** list, specify how to return the values. You have the following options:

TOP	Returns the value of the first field in the column
SUM	Returns the sum of all field values in the column
COUNT	Returns the number of field entries in the column
MAX	Returns the maximum value of the fields in the column
MIN	Returns the minimum value of the fields in the column

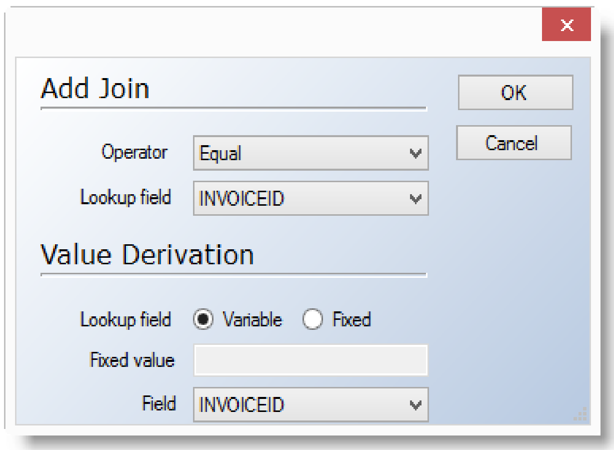
7. In the Sorting list, specify whether you want the results sorted in ascending or descending order, and then click **OK**.

Note: You could also drag a field from one table and drop it on the name of another table. This will automatically create the Conditional Lookup field with the exception of the joins, which are covered below.

Add Joins

Next you have to add a join between the two tables.

1. Expand the lookup field, right-click **Joins**, and then select **Add Join**.



Add Join

Operator: Equal

Lookup field: INVOICEID

Value Derivation

Lookup field: (●) Variable ( ) Fixed

Fixed value:

Field: INVOICEID

2. In the **Operator** field, specify when to look up a value.

3. In the **Lookup** field list, select the field that uses the lookup.

4. Specify whether the value is **Variable** or **Fixed**.

5. If you select **Fixed**, enter a fixed value in the **Fixed Value** field.

6. If you select **Variable**, select the matching field to lookup.

7. Click **OK**.

Specifying Conditions

You can now specify conditions for when to lookup, and you can add more lookup fields to the same field.

1. Expand the lookup field, right-click **Conditions**, and then select **Add Condition**.

2. In the **Field** list, select the field that uses the lookup value.

3. In the **Operator** field, specify when to lookup the value.

4. In the **Value** field, specify the value that results in a new lookup, and then click **OK**.

Understanding Hierarchy Tables

A Hierarchy Table is used to select data from a table and create a new reporting structure which is different from the structure in the data source. You will typically use a hierarchy table for finance reporting where you want to consolidate data from a number of different accounts, such as ledger accounts.

Creating and using hierarchy tables requires a multi-step operation that combines it with a parent-child dimension.

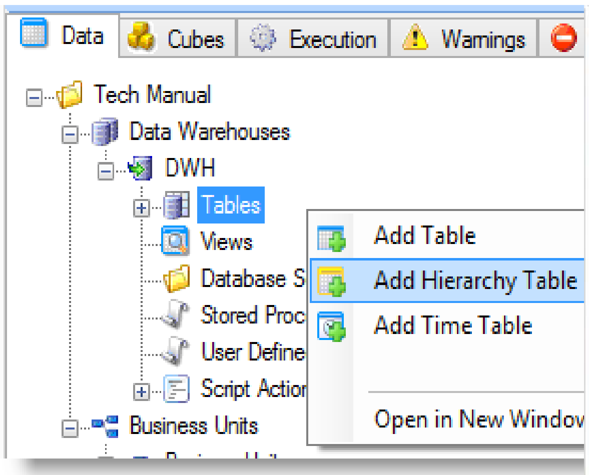
1. Create the hierarchy table and specify the contents of the table.

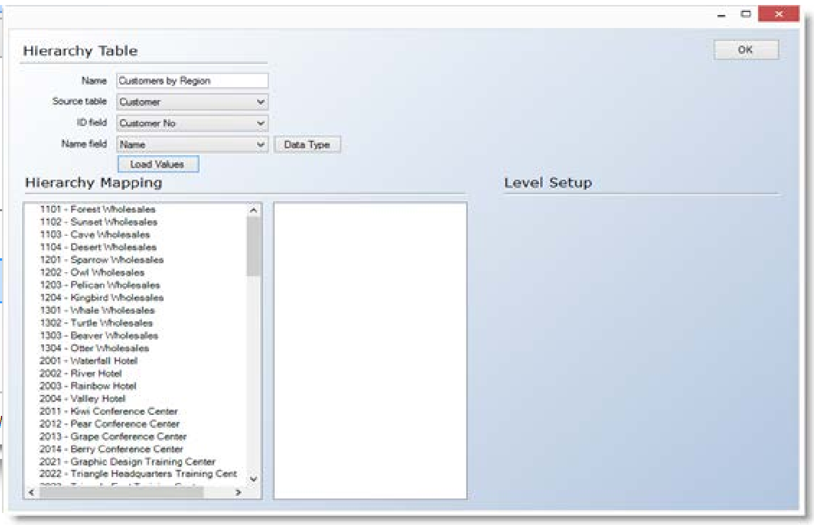
2. Then create a parent-child dimension and add it to a cube. When you build the structure, be sure to choose names that are meaningful to the end-user.

To Add a Hierarchy Table

1. On the **Data** tab, expand **Data Warehouses**, and expand the data warehouse you wish to modify.

2. Right-click **Tables**, and then select Add Hierarchy Table.





Hierarchy Table

Name: Customers by Region
Source table: Customer
ID field: Customer No
Name field: Name

Hierarchy Mapping

1101 - Forest Wholesales
1102 - Sunset Wholesales
1103 - Cave Wholesales
1104 - Desert Wholesales
1201 - Sparrow Wholesales
1202 - Owl Wholesales
1203 - Pelican Wholesales
1204 - Kingbird Wholesales
1301 - Whale Wholesales
1302 - Turtle Wholesales
1303 - Beaver Wholesales
1304 - Otter Wholesales
2001 - Waterfall Hotel
2002 - River Hotel
2003 - Rainbow Hotel
2004 - Valley Hotel
2011 - Kiwi Conference Center
2012 - Pear Conference Center
2013 - Grape Conference Center
2014 - Berry Conference Center
2021 - Graphic Design Training Center
2022 - Triangle Headquarters Training Cent

3. In the Name field, enter a name for the table.

4. From the **Source** table list, select the table containing the desired data.

5. In the **ID field**, select the field that identifies the individual entries in the table; for example, the customer number.

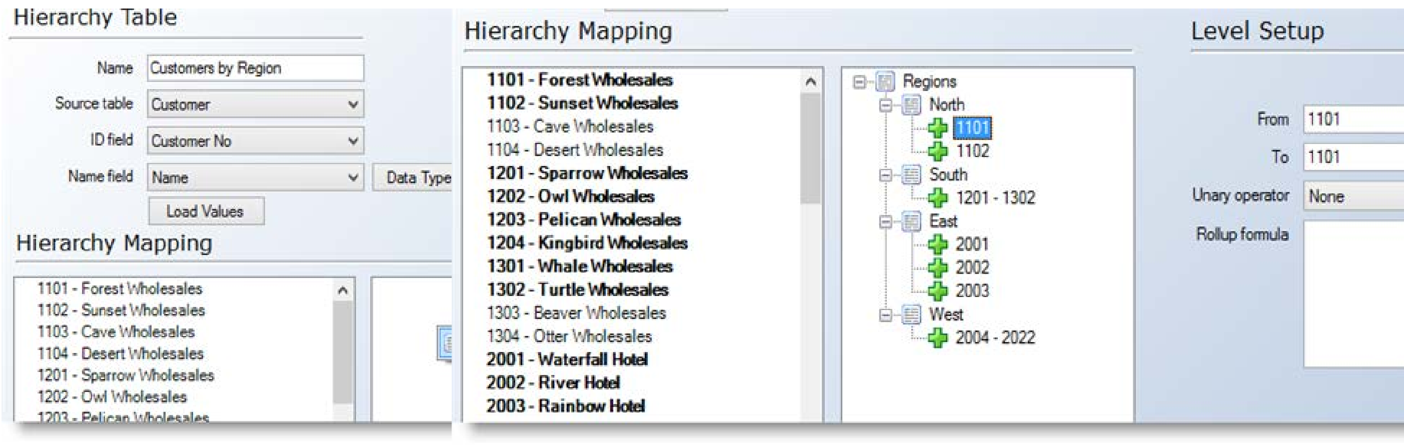
Note: If you need more than one field to identify the entries, you have to create a concatenated field before you create the hierarchy table.

6. In the **Name field**, enter the name that identifies the individual entries; for example, account name, and then click **Load**. The Hierarchy Mapping pane is now populated with the entries of the source table. You can now create the report structure.

To Create a Hierarchy Table Structure

The structure you create corresponds to the structure of the report that is displayed to the end-user.

1. Right-click in the Blank pane, and then select Add Root Heading. The root headings become root nodes in the final report.


Hierarchy Table

Name: Customers by Region
Source table: Customer
ID field: Customer No
Name field: Name

Hierarchy Mapping

1101 - Forest Wholesales
1102 - Sunset Wholesales
1103 - Cave Wholesales
1104 - Desert Wholesales
1201 - Sparrow Wholesales
1202 - Owl Wholesales
1203 - Pelican Wholesales
1204 - Kingbird Wholesales
1301 - Whale Wholesales
1302 - Turtle Wholesales
1303 - Beaver Wholesales
1304 - Otter Wholesales
2001 - Waterfall Hotel
2002 - River Hotel
2003 - Rainbow Hotel

```
Regions
  North
    1101
    1102
  South
    1201 - 1302
  East
    2001
    2002
    2003
  West
    2004 - 2022
```

Level Setup

From: 1101
To: 1101
Unary operator: None
Rollup formula:

2. In the **Level Setup** area, enter a name for the heading in the **Name** field.

3. Right-click the root heading, and select Add Sub Heading to add a child node to the structure.

4. In the **Name** field, enter a name for the child node.

5. In the **Unary Operator** list, specify how you want the value of the child node to be aggregated to the sum of all the values in the subheading. The unary operator ensures that the values are aggregated properly in the final report. You have the following options:

None	The value is ignored
Add	The value is added to the sum of the values
Subtract	The value is subtracted from the sum of the values
Multiply	The value is multiplied by the sum of values
Divide	The value is divided by the sum of the values

Repeat steps 1-5 for all root headings and subheadings you want to add.

6. Select an entry in the **Hierarchy Mapping** pane, and drag it to the preferred subheading in the **Blank** pane. Alternatively, you can specify a range of entries in the **From** and **To** fields.

7. To exclude an entry from a given range, right-click the preferred subheading, select Add Exclude, and then specify a range in the **From** and **To** fields. Alternatively, right-click the specific entry and select **Change to Exclude**.

8. If a root heading or subheading represents the sum of other subheadings, such as Contribution Margin, you can use a formula to determine the content of the heading. Enter a formula in the Roll-up formula field. Formulas are written in MDX.

9. Click **OK**when you have completed the structure. You can now create the parent-child dimension where the consolidation table will be used.

Note: When you create the parent-child dimension you will typically use Sort By Attribute. You therefore need to create a Sort order dimension level where the key column is Sort order. It is also necessary to enable Unary column and Roll-up column on the dimension. You can then set the parent-child dimension to Sort By Attribute.

To Specify the Execution Order of Tables

When you create a standard execution package, tables are executed in the order in which they appear in the tree on the **Data** tab. To avoid errors when executing, you must therefore ensure that tables are executed in logical order. For example, an Order table must be executed before the related Order Detail table. You may, therefore, have to move tables up and down in the data warehouse tree.

Note: For advanced execution packages, the order will be determined in the **Execution Setup** window itself. For more information, see To Add Advanced Execution Packages.

To Move Tables Up and Down in the Tree

1. On the Data tab, expand data warehouse or staging database, and select the table that you want to move.

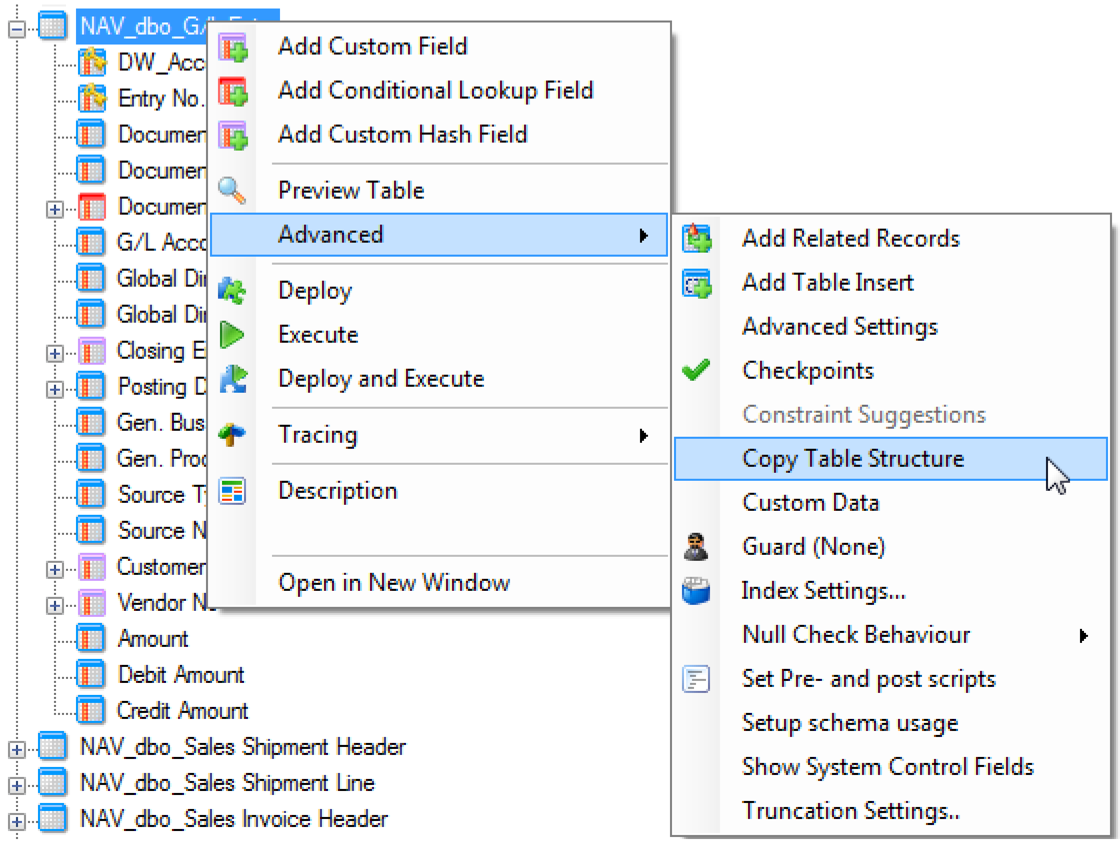
2. Left-click and drag and drop the table to the preferred location. Alternatively, use ALT+UP ARROW or ALT + DOWN ARROW to change the order in which the table will appear in the execution order.

Copy Table Structure

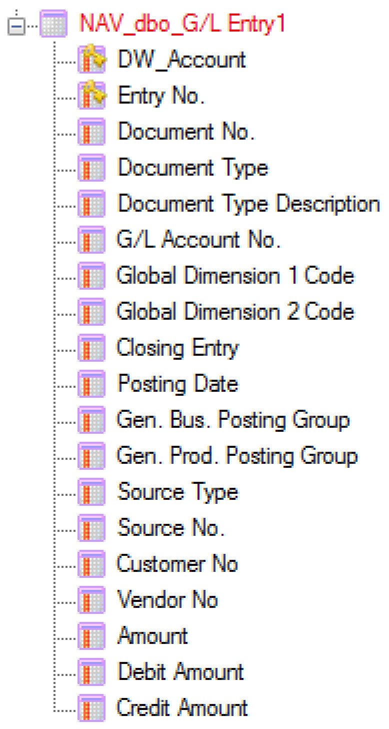
This feature enables you to create a structural copy of a Staging area- or Data Warehouse table. The copy of the table will contain the same fields as the source table, however, lookup fields are converted to regular fields ,and transformations are removed.

A structural copy of a table can be useful in many scenarios. It can be used as a destination of a “Table Insert” on the staging area, and it is very useful on the data warehouse level if you need a second copy of a table with many data movements for each field.

A structural copy of a table on stage is created as a custom table without any data source connection.



By default, the copy of the table is name <TableName>1. If <TableName>1 Exists, it will use <TableName>2 etc.



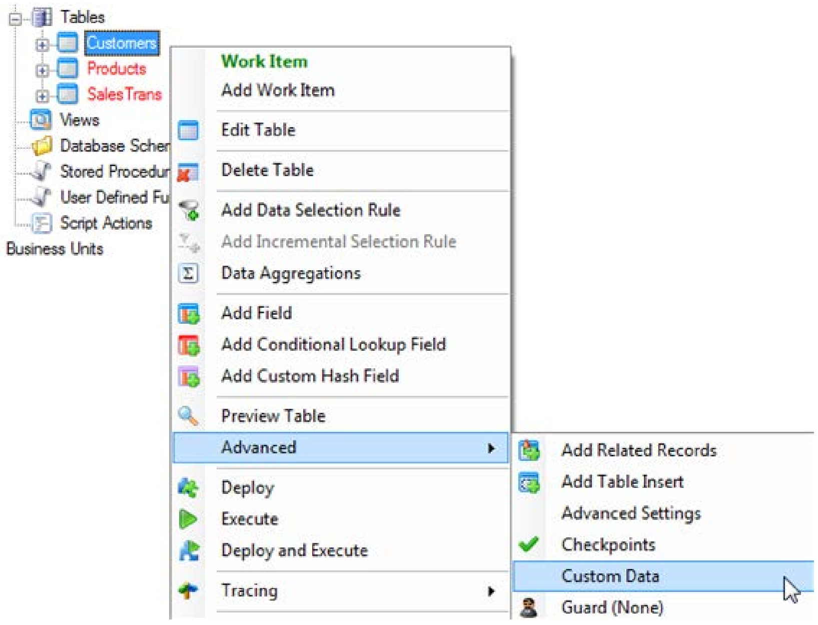
A structural copy of a table on the data warehouse also contains the data movement information.

Import Custom Data from Excel

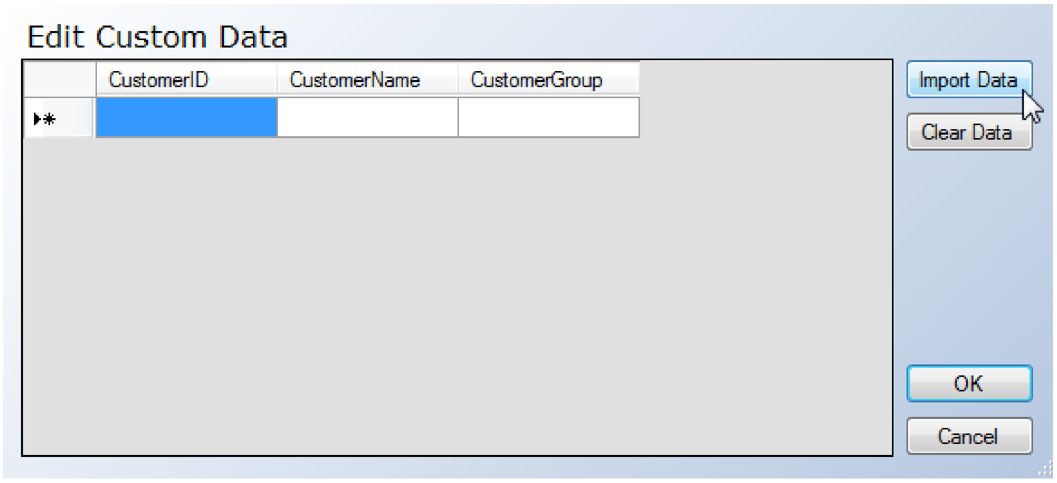
Custom data can be used for multiple purposes in TX2014.

This feature enables users to import custom data from Excel. Excel offers excellent input and copy / paste features which makes it a good choice when it comes to selecting a "Custom Data Frontend".

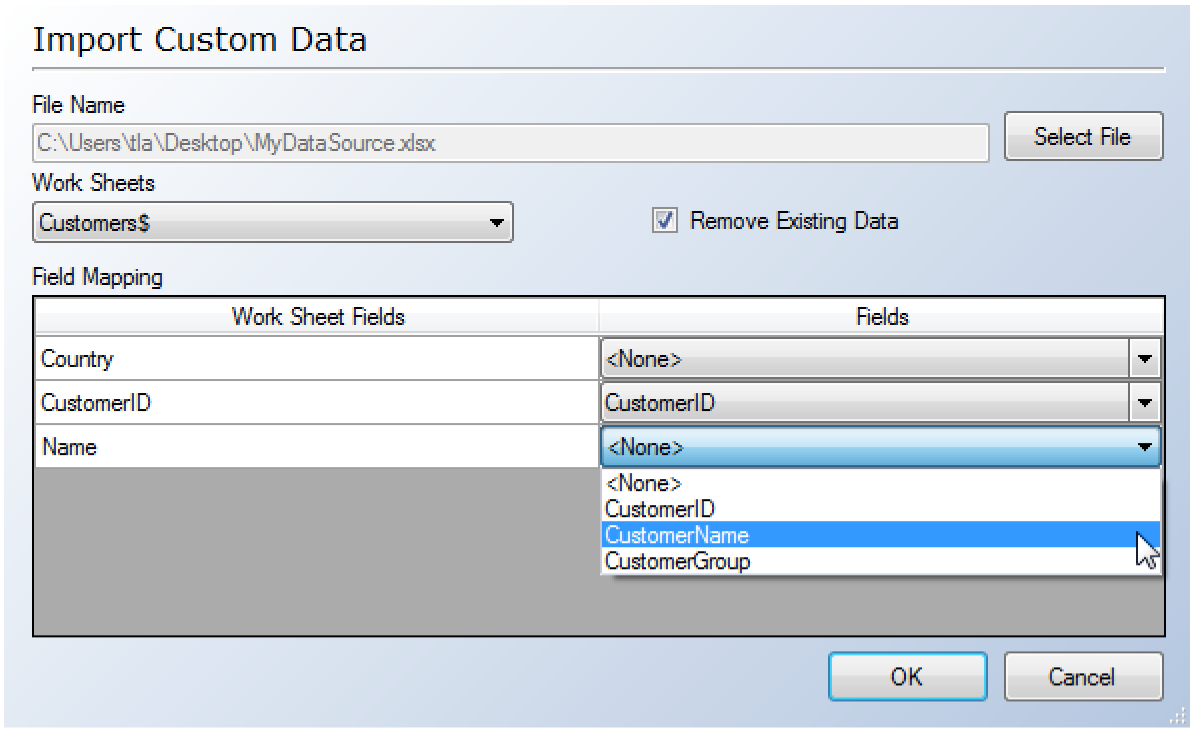
To import custom data, open the custom data form:



In the custom data form, select "Import Data":



A file-picker window will appear. Select the Excel sheet where the data is entered, and the Import Custom Data Dialogue will appear:



Import Custom Data

File Name: C:\Users\tla\Desktop\MyDataSource.xlsx

Work Sheets: Customers$

[✓] Remove Existing Data

Field Mapping

Work Sheet Fields	Fields
Country	<None>
CustomerID	CustomerID
Name	<None>

File Name: Path to the input file.

Work Sheets: Select the sheet within the excel workbook that holds the data.

Remove Existing Data: Check this to delete existing custom data – or uncheck it to append to the existing custom data in TX2014.

Field Mapping: Select the mapping between the fields in the worksheet and the table in TX2014. If the names are the same in both source and destination, TX2014 will automatically map these. If no match is found in any field, TX2014 will do the mapping based on the order of the fields.

Click OK to start the import of the data.

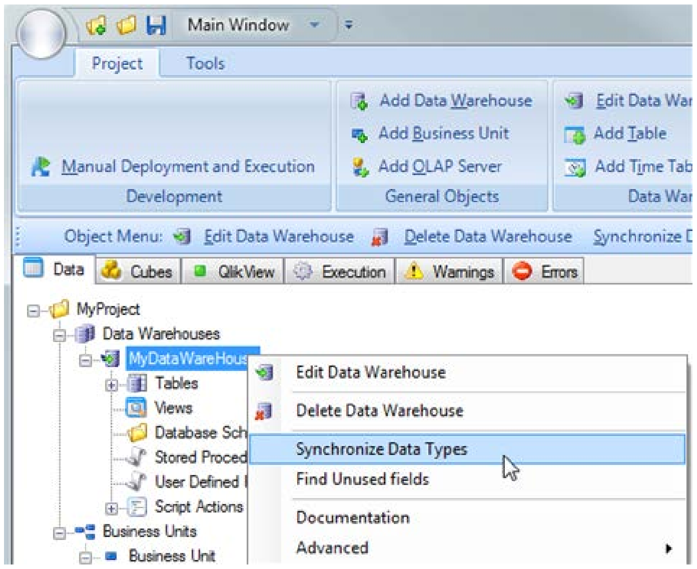
Please note that the import of Excel data requires that the drivers for Excel are installed in the correct bitversion.

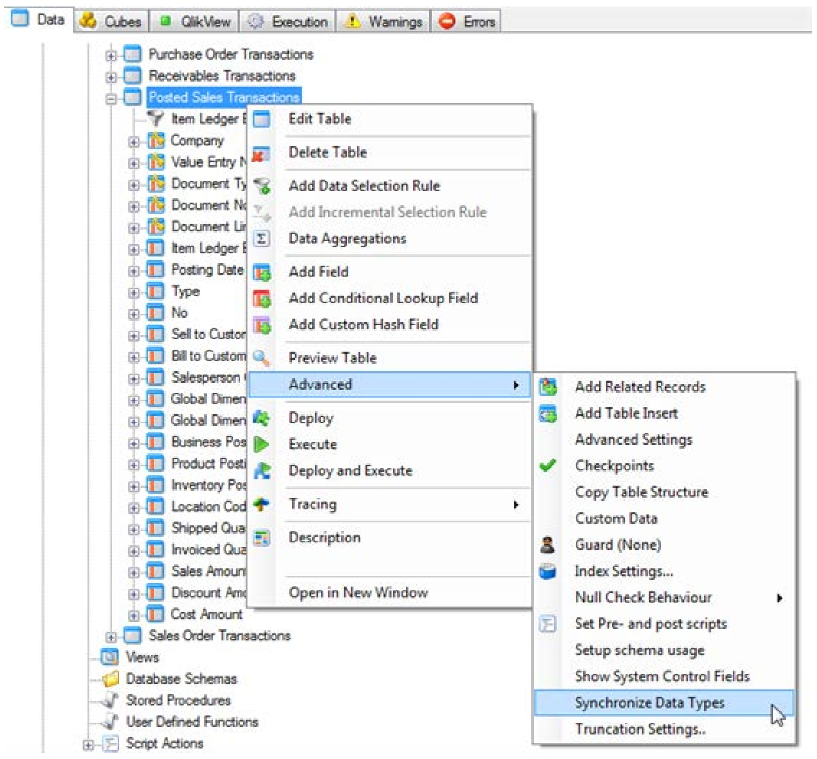
Synchronization of Data Types in Data Warehouse

Sometimes, when connecting a data source to a pre-built Data Warehouse structure or when changing a data source connection, some of the data types might not match up. This can result in errors during execution or data being truncated / left out of the data warehouse table.

To prevent this, use the Synchronize Data Types feature.

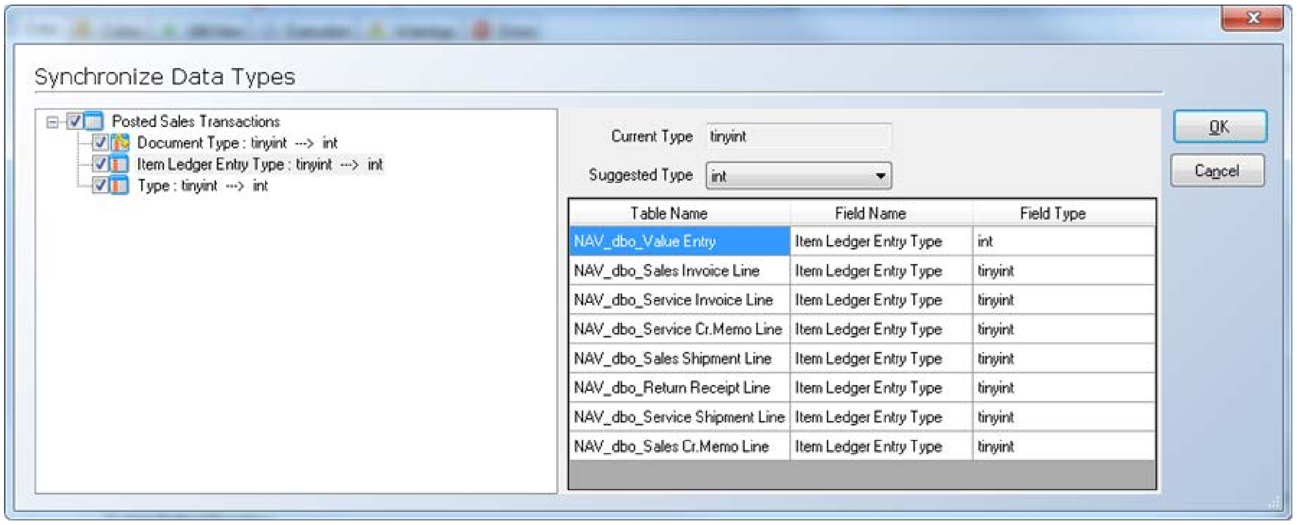
The feature can be used on the data warehouse level or on the single table level:





The feature will check if the data type on the data warehouse table matches the data that is copied into the field from the source of the copy. If it encounter any differences, it will try to determine and suggest the best suited data type.

The suggestion and the data types of the source(s) are shown in the dialogue:



Synchronize Data Types

Posted Sales Transactions
- Document Type : tinyint ---> int
- Item Ledger Entry Type : tinyint ---> int
- Type : tinyint ---> int

Current Type: tinyint

Suggested Type: int

Table Name	Field Name	Field Type
NAV_dbo_Value Entry	Item Ledger Entry Type	int
NAV_dbo_Sales Invoice Line	Item Ledger Entry Type	tinyint
NAV_dbo_Service Invoice Line	Item Ledger Entry Type	tinyint
NAV_dbo_Service Cr.Memo Line	Item Ledger Entry Type	tinyint
NAV_dbo_Sales Shipment Line	Item Ledger Entry Type	tinyint
NAV_dbo_Return Receipt Line	Item Ledger Entry Type	tinyint
NAV_dbo_Service Shipment Line	Item Ledger Entry Type	tinyint
NAV_dbo_Sales Cr.Memo Line	Item Ledger Entry Type	tinyint

When the window is exised using the OK button, all Accepted suggestions will be effectuated.

The following Comparison/Conversion Matrix is used when deciding suggestion between data types:

	Binary	Boolean	Date	DataTime	DataTime2	Geography	Geometry	Integer	Numeric	Text	Unique Identifier
Binary	Binary										
Boolean		Boolean						**tinyint**	**decimal**	**text**	
Date			Date	datetime	**DataTime2**					**text**	
DataTime			**datetime**	datetime	**datetime2**					**text**	
DataTime2			**datetime2**	**datetime2**	datetime2					**text**	
Geography						Geography					
Geometry							Geometry				
Integer		**int**						Integer	**decimal**	**text**	
Numeric		**decimal**						**decimal**	Numeric	**text**	
Text		**text**	**text**	**text**	**text**			**text**	**text**	text	**text**
Unique Identifier										**text**	Unique Identifier

Working with Date Tables

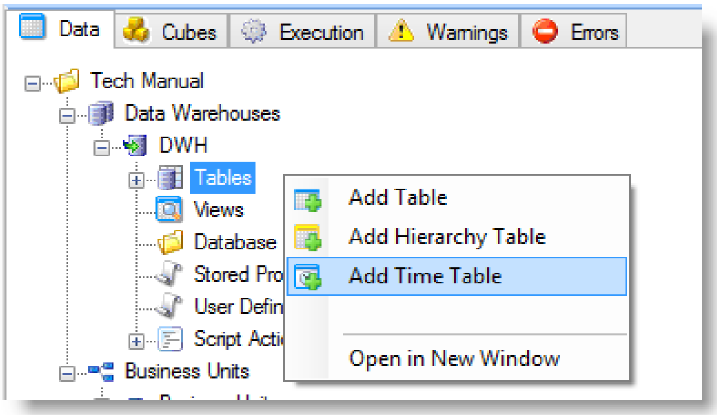
Typically the cubes you create will contain a date dimension. In TX2014 you use date tables to define your date dimensions. Date tables make it possible for you to analyze data over time. For example, you may report data on a daily, weekly, or monthly basis. In TX2014 the date tables are stored in the data warehouse.

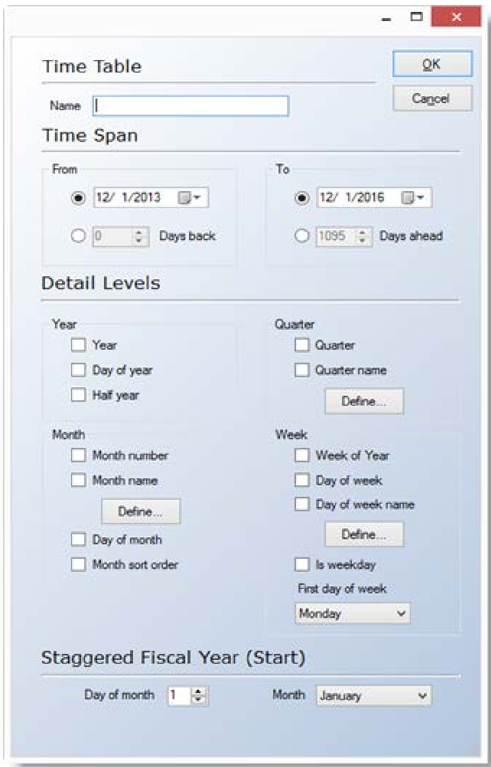
The format of the calendar is dependent on which language version of Windows you have installed on your computer.

To Create Date Tables

1. On the **Data** tab, expand **Data Warehouses**, expand the preferred data warehouse, and then right-click **Tables**.

2. Select **Add Time Table**.





Name

Time Span

From: (●) 12/ 1/2013 ( ) 0 Days back

To: (●) 12/ 1/2016 ( ) 1095 Days ahead

Detail Levels

Year: ☐ Year ☐ Day of year ☐ Half year

Quarter: ☐ Quarter ☐ Quarter name

Month: ☐ Month number ☐ Month name ☐ Day of month ☐ Month sort order

Week: ☐ Week of Year ☐ Day of week ☐ Day of week name ☐ Is weekday

First day of week: Monday

Staggered Fiscal Year (Start)

Day of month: 1 Month: January

3. In the **Name** field, type a name for the date table.

4. In the **Time Span** section, you will select the **From** and **To** dates for the date table. The recommendation is that the **From** date should be the first day of the year that you have transactions in your data source and that the **To** date is some number of days ahead. By selecting **Days Ahead** (for example 1095 = 3 years) TX2014will dynamically extend the size of the date table to a specified number of days in the future.

5. In the **Detail Levels**, specify which fields you want to use in your date dimension. You have the following options:

Year	Specifies the year

Day of the year	Specifies the day number of a year
Half year	Specifies the half year of the fiscal year
Day of week	Specifies the day number of a week
Day of week name	Specifies the name of the week
Quarter	Specifies the quarter number of the fiscal year
Quarter name	Specifies the quarter name
Month number	Specifies the month number of a year
Month name	Specifies the name of a month
Month sort order	Specifies whether to sort months according to the fiscal year. For example, if the fiscal year starts in July the sort order starts from July
Week of Year	Specifies the number of the week in a year
Is weekday	Specifies whether a day is a weekday
Week number	Specifies the week number of a year
First day of week	Specifies whether the first day of a week is Monday or Sunday. The default setting is Monday
Day of month	Specifies the number of the day in a month

If your organization uses a non-calendar fiscal year, you can select the fiscal year start date under the **Staggered Fiscal Year** section at the bottom.

6. Click **OK**.

Selecting and Validating Data

Specifying selection rules ensures that only the specific data needed for you analysis is extracted from the data source.

The data is moved to the staging database after the selection. On the staging database, the data goes through a data cleansing process that uses validation and transformation rules to ensure that only valid data is loaded into the data warehouse.

However, you can also apply validation and transformation rules on a data warehouse. This is useful when you have moved data from different business units into the data warehouse and want to ensure the validity of the consolidated data.

Data Selection Rules

Data selection rules are used to specify a set of conditions that data extracted from the data source must satisfy. By applying selection rules, only the subset of data that you actually need is loaded into the staging database. Data selection rules for the staging database are applied at the field level in the data source tree. You apply data selection rules for the data warehouse database at the field level of the data warehouse tree.

Values must be either integers or letters. You can also specify a list of values by entering comma-separated values. The following operators are available when you create selection rules:

Not Empty	Selects records where the value of a field is not empty or NULL
Equal	Selects records where the value of a field is equal to the specified value
Greater Than	Selects records where the value of a field is greater than the specified value
Less Than	Selects records where the value of a field is less than the specified value
Not Equal	Selects records where the value of a field is not equal to the specified value
Greater or Equal	Selects records where the value of a field is greater than or equal to the specified value
Less or Equal	Selects records where the value of a field is less than or equal to the specified value
Min. Length	Selects records that contain at least the specified number of characters
Max. Length	Selects records that contain no more than the specified number of characters
List	Selects records where the value of a field is equal to one of the specified comma separated values
Empty	Selects records where the value of a field is empty or NULL
Not in List	Selects records where the value of a field is not equal to one of the specified comma separated values
Like	Selects records where the value of a field is similar to the specified value. A percent sign (%) can be used as a wildcard. *ABC%* will return all records where the value in the spec-

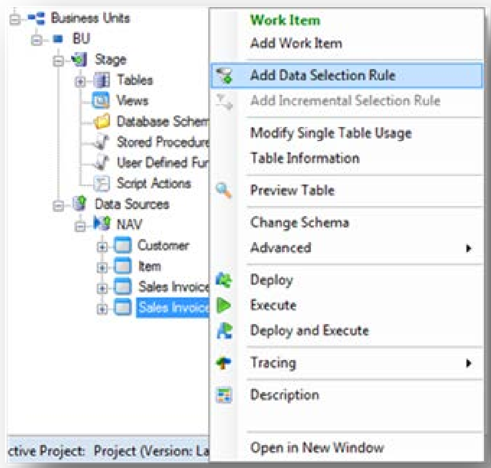
	ified field starts with *ABC.*
Not Like	Selects records where the value of a field is not similar to the specified value. A percent sign (%) can be used as a wildcard. *ABC%* will return all records where the value in the specified field does not start with *ABC.*

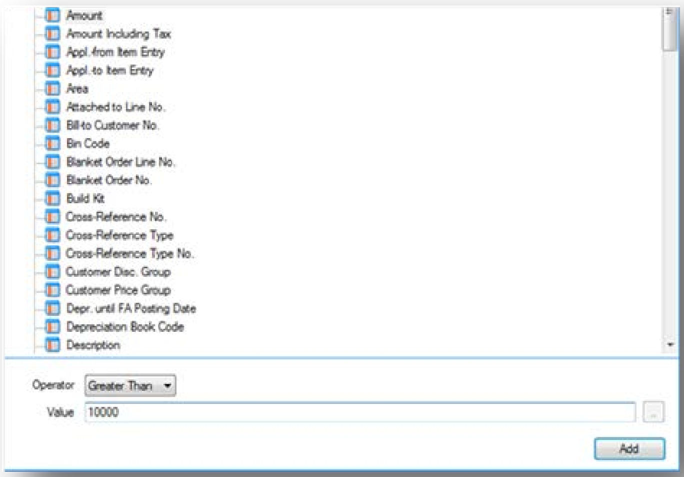
To Define Data Selection Rules

1. Expand **Business Units**, select the preferred business unit, and then expand **Data Sources**.

2. Expand the preferred data source, and then select the table for modification.

3. Right-click the table, and then choose Add Data Selection Rule.





Operator: Greater Than
Value: 10000

4. In the **Data Selection** pane, select the field to be modified.

5. In the **Operator** box, select the preferred operator.

6. In the **Value** field, type the preferred value if applicable, and then click **Add**.

All selection rules that you have applied to a table are displayed in the Project tree below the relevant table.

Note: To add Data Selection Rules to the data warehouse, you will locate the table in the data warehouse tree and follow steps 3 through 6 above.

Data Validation

Validation rules ensure a high level of accuracy and reliability of the data in the data warehouse and are used to discover invalid data. You can apply validation rules at the field level in the staging database or at field level in the data warehouse.

While data is cleansed on the staging database, it often has to be cleansed again if you have consolidated data from different business units in the data warehouse.

You can make a validation rule conditional if you want the rule to apply in specific situations only. The following operators are available both when you create validation rules, and when you create conditional validation rules.

Not Empty	Selects records where the value of a field is not empty or NULL
Equal	Selects records where the value of a field is equal to the specified value
Greater Than	Selects records where the value of a field is greater than the specified value
Less Than	Selects records where the value of a field is less than the specified value
Not Equal	Selects records where the value of a field is not equal to the specified value
Greater or Equal	Selects records where the value of a field is greater than or equal to the specified value
Less or Equal	Selects records where the value of a field is less than or equal to the specified value
Min. Length	Selects records that contain at least the specified number of characters
Max. Length	Selects records that contain no more than the specified number of characters
List	Selects records where the value of a field is equal to one of the specified comma separated values
Empty	Selects records where the value of a field is empty or NULL
Not in List	Selects records where the value of a field is not equal to one of the specified comma separated values
Like	Selects records where the value of a field is similar to the specified value. A percent sign (%) can be used as a wildcard. *ABC%* will return all records where the value in the specified field starts with *ABC.*
Not Like	Selects records where the value of a field is not similar to the specified value. A percent sign (%) can be used as a wildcard. *ABC%* will return all records where the value in the specified field does not start with *ABC.*

For each validation rule you apply to a field, you must also classify the severity of a violation. The following classifications are available:

Warning	The violation is not critical to the data quality and does not require immediate attention. The data is considered valid and will still be made available to the end users.
Error	The violation is critical to the data quality and requires immediate attention. The data is considered invalid and will not be made available to the end users.

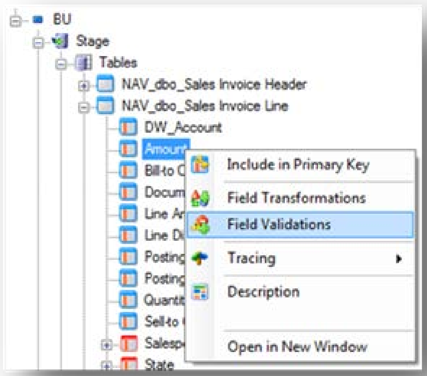
To Define Data Validation Rules

1. Expand **Business Units**, expand the preferred business unit, and then expand the staging database.

   - OR -

   Expand **Data Warehouses**, and then expand the preferred data warehouse.

2. Identify the table to which you want to add a validation rule, right-click the field, and then choose Field Validations.



3. In the **Data Validation** pane, select the preferred field.

4. In the **Operator** box, select the preferred operator,

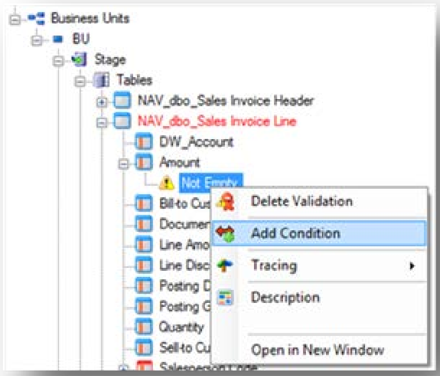
5. In the **Value** field, enter the preferred value.

6. Specify whether a violation of the rule is to be classified as an **Error** or as a **Warning**, and then click **Add**.

To Define Conditions

1. Identify the validation rule you wish to modify.

2. Right-click the rule and then select **Condition**.



3. In the **Condition** pane, select the preferred operator from the **Operator** list.

4. In the **Value** field, enter the preferred value, and then click **Add**. The condition is displayed in the staging database tree below the validation rule or transformation rule it belongs to.

To View Validation Errors

1. Select the **Errors** tab.

2. In the **Database** list, select the database you wish to view.

3. In the **Table** list, select the table for viewing. The **No. of rows** field lists how many rows have errors, and the rows that violate the validation rules are displayed in the pane below.

4. Select a row to display the warning message in the **Warning Message** field.

To View Validation Warnings

1. Select the **Warnings** tab.

2. In the **Database** list, select the database for viewing.

3. In the Table list, select the table for viewing. The **No. of rows** field lists how many rows have warnings, and the rows that violate the validation rules are displayed in the pane below.

4. Select a row to display the warning message in the **Warning Message** field.

Data Transformation

Data transformations allow existing data to be modified in a specified manner.

This allows you to:

-Easily reverse the sign of numeric values

-Trim fields

-Return a specified number of characters from the original field value

To Define Field Transformation Rules

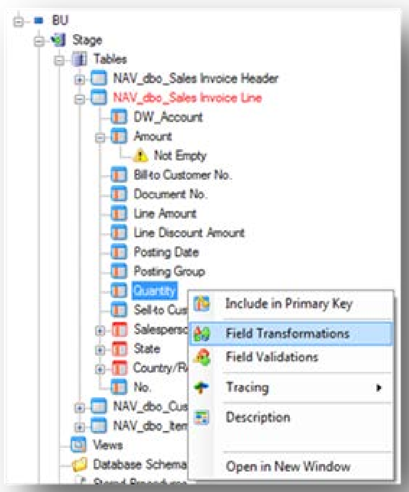
1. Expand **Business Units**, expand the preferred business unit, and then expand the staging database.

   - OR -

   Expand **Data Warehouses**, and then expand the preferred data warehouse.

2. Right-click the field to which you want to add a transformation rule, and then choose **Field Transformations**.

3. In the **Field Transformation** pane, select the preferred field.



4. In the **Operator** box, select the preferred operator, and then click **Add**.

5. If you have selected **First** or **Last**, specify in the **Length** field how many characters you want included.

Upper	Converts all text values to upper-case
Lower	Converts all text values to lower-case
First	Returns the number of beginning characters specified by the user

Last	Returns the number of ending characters specified by the user
TrimLeft	Trims padded spaces from the left of the data
TrimRight	Trims padded spaces from the right of the data
Trim	Trims padded spaces from the left and right of the data
Fixed	Inserts a fixed value that is specified by the user
Custom	Allows for custom SQL code to be executed
ReverseSign	Reverses the sign for numeric values
TimeOnly	Returns only the time portion of a datetime field
DateOnly	Returns only the date portion of a datetime field
Replace	Replaces one set of characters with another

To Define Conditions

1. Identify the field transformation to add the condition to.

2. Right-click the rule, and then select **Add Condition**.

3. In the **Condition** pane, select the preferred operator from the **Operator** list.

4. In the **Value** field, enter the preferred value, and then click **Add**. The condition is displayed below the field transformation rule it belongs to.

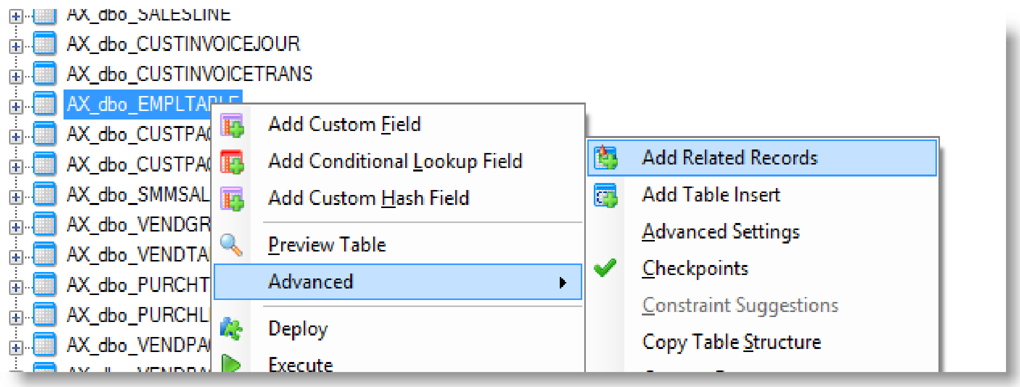
Handling Early Arriving Facts

In a live working environment, it is possible that transactional data may contain values that have not yet been added to the source database in the corresponding dimension table. An example of this could be a Sales Invoice that has a Salesperson Code where the Salesperson Code does not yet exist in the Salesperson table. When the data warehouse is updated and the cubes are processed, the values for this salesperson will fall under the “Unknown” member for the Salesperson dimension. This happens because the cube does not see the Salesperson Code on the transaction as being a known value when compared to the list of salespeople in the Salesperson dimension.

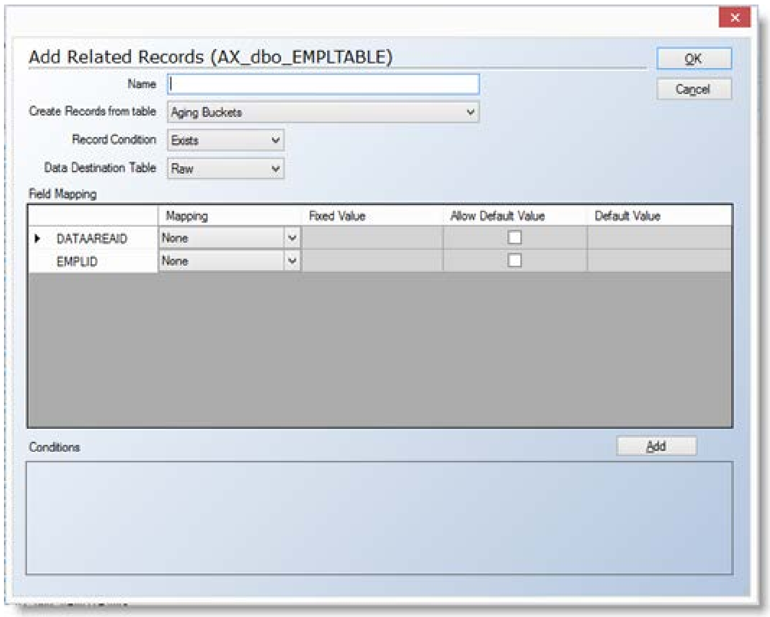
In TX2014,it is possible to handle these “early arriving facts” in such a manner that they will show at least partial information until the data source is properly updated with all of the normal dimension information. This prevents information from being placed into the “Unknown” member when the data is consumed by end-users. Once the dimension value is properly added to the ERP system or data source by a user, all fields for the previously missing record will then be populated according to the values in the data source.

Enabling Early Arriving Facts

1. Identify the dimension table to which relevant values from the transaction table should be added, right-click on the table name, and go to **Advanced -> Add Related Records.**



The **Add Related Records** window will open:


Add Related Records (AX_dbo_EMPLTABLE)

Name:
Create Records from table: Aging Buckets
Record Condition: Exists
Data Destination Table: Raw

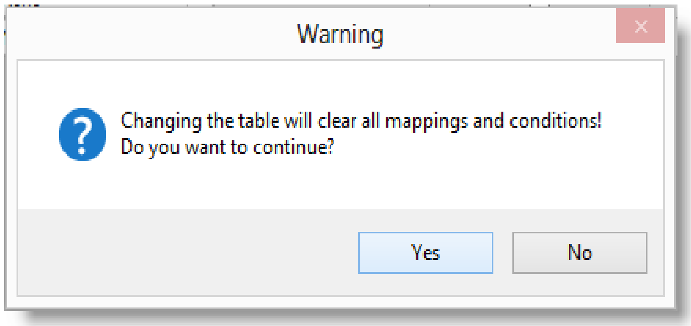
Field Mapping

	Mapping	Fixed Value	Allow Default Value	Default Value
DATAAREAID	None		☐	
EMPLID	None		☐	

Conditions

2. Give a descriptive name in the **Name** field to the **Add Related Records** rule that is currently being created.

3. Select the transaction table in the **Create Records from Table** drop-down that will identify the table from which to bring in potential new values. A window may appear stating that all mappings and conditions will be cleared. Click **Yes**.

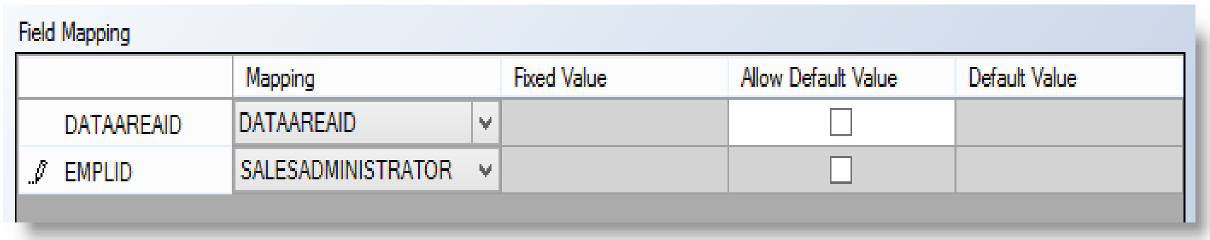

Warning

Changing the table will clear all mappings and conditions!
Do you want to continue?

4. In the **Record Condition** drop-down, select the option to determine when data will be inserted into the dimension table if new values are found in the transaction table. The most common option is **Not Exist**, which will add in values that do not currently exist in the dimension table.

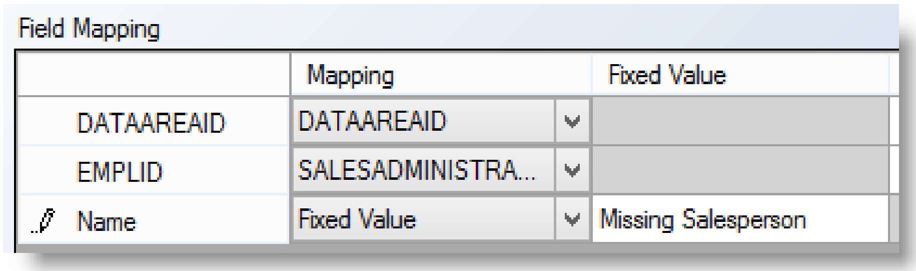
5. Select the **Data Destination Table** to insert the values into. The default option is the Raw table.

6. In the **Field Mapping** section, select the fields to be mapped from the transaction table and inserted into the dimension table. In the example below, the DW_Account field (Company) and Salesperson Code fields will be extracted from the transaction table and inserted into the dimension table.


Field Mapping

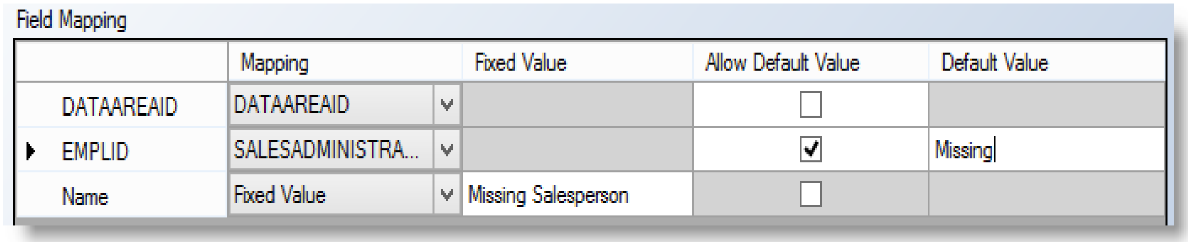
	Mapping	Fixed Value	Allow Default Value	Default Value
DATAAREAID	DATAAREAID		☐	
EMPLID	SALESADMINISTRATOR		☐	

7. It is possible to add in fixed values for fields in the dimension table that the transaction may not have data for. In the example below, the fixed value "Missing Salesperson" will be added in the **Name** field for all Salesperson Codes added from the transaction table. This is achieved by selecting the **Fixed Value** option in the **Mapping** column for the **Name** field and entering the desired fixed value in the **Fixed Value** column.


Field Mapping

	Mapping	Fixed Value
DATAAREAID	DATAAREAID	
EMPLID	SALESADMINISTRA...	
Name	Fixed Value	Missing Salesperson

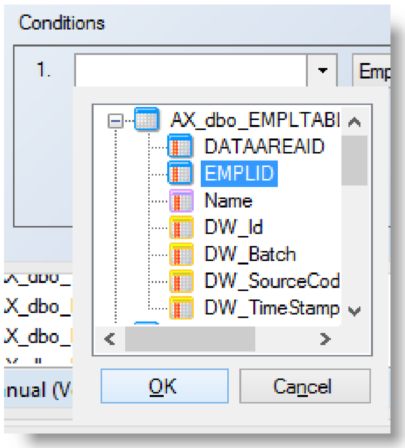
8. If desired, a default value can be inserted instead of bringing in the values that exist in the transaction table. This could be used to assign fixed values to all data brought in for early arriving facts. This is achieved by clicking the checkbox in the **Allow Default Value** column and typing the corresponding fixed value in the **Default Value** column. This is not common.


Field Mapping

	Mapping	Fixed Value	Allow Default Value	Default Value
DATAAREAID	DATAAREAID		☐	
EMPLID	SALESADMINISTRA...		☑	Missing
Name	Fixed Value	Missing Salesperson	☐	

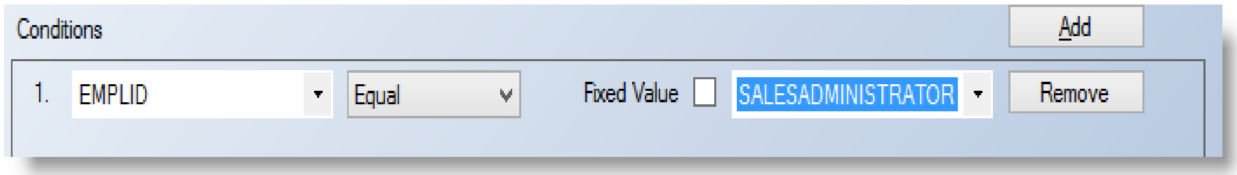
9. The last step is to define the relationship between the two tables. Click the **Add** button in the **Conditions** section.

10. Select the first field to join in the dimension table (**Code**), and click **OK.**



11. Select the operator to be used for the join. The most common operator is **Equal**.

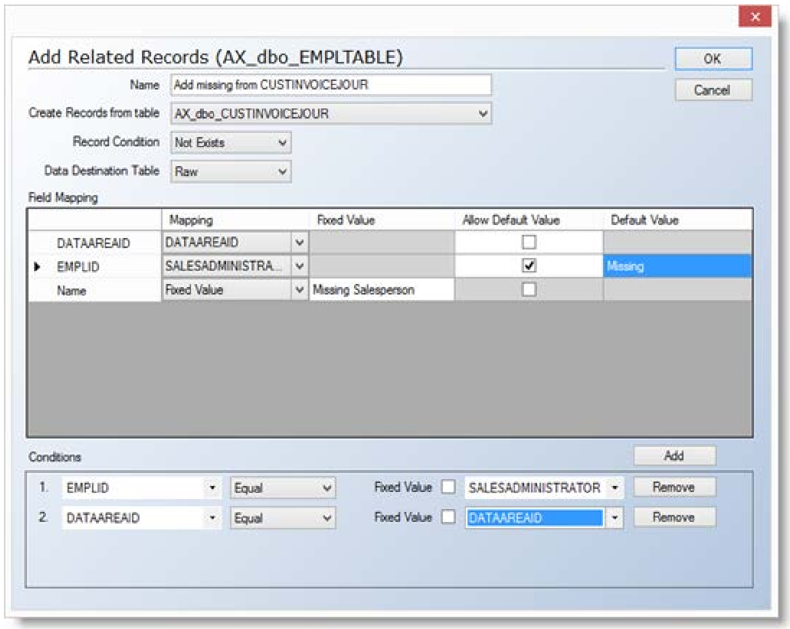
12. Select the matching field in the transaction table (**Salesperson Code**), and click **OK.**



1. EMPLID | Equal | Fixed Value ☐ | SALESADMINISTRATOR

13. Repeat steps 9 through 12 for any additional joins that need to be made (such as Company).

The final result will look similar to the screen-shot below. Click **OK** when finished to save the settings, and close the **Add Related Records** window.



Add Related Records (AX_dbo_EMPLTABLE)

Name: Add missing from CUSTINVOICEJOUR
Create Records from table: AX_dbo_CUSTINVOICEJOUR
Record Condition: Not Exists
Data Destination Table: Raw

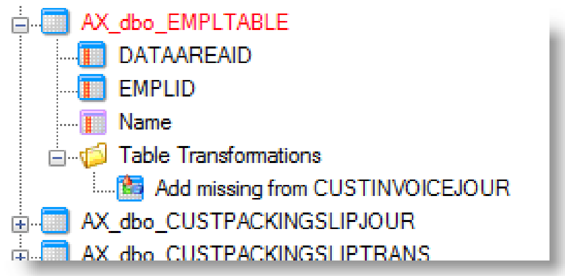
Field Mapping

	Mapping	Fixed Value	Allow Default Value	Default Value
DATAAREAID	DATAAREAID		☐	
EMPLID	SALESADMINISTRA...		✓	Missing
Name	Fixed Value	Missing Salesperson	☐	

Conditions

1. EMPLID | Equal | Fixed Value ☐ | SALESADMINISTRATOR
2. DATAAREAID | Equal | Fixed Value ☐ | DATAAREAID

A folder for **Table Transformations** will be added to the bottom of the dimension table. The selection criteria that were previously set can be edited by right-clicking the transformation and selecting **Edit Related Record**.


AX_dbo_EMPLTABLE
DATAAREAID
EMPLID
Name
Table Transformations
Add missing from CUSTINVOICEJOUR
AX_dbo_CUSTPACKINGSLIPJOUR
AX_dbo_CUSTPACKINGSLIPTRANS

14. Deploy and execute the dimension table. Any records that exist in the transaction table, but not in the dimension table, will be added during the data cleansing process. A screen-shot of the result based on the example in this document is shown below. The salesperson code “BP” existed on a sales document, but no corresponding Salesperson Code existed in the Salesperson table. Once the salesperson is properly added to the ERP system and the table is refreshed, all proper information will be pulled in from the ERP system, and the name will no longer say “Missing Salesperson.”

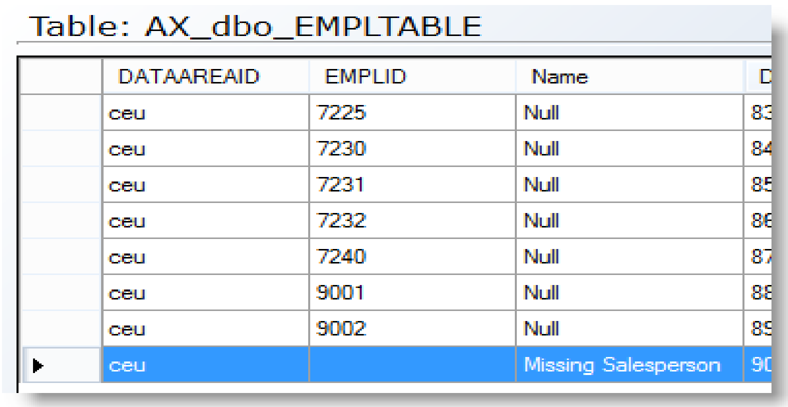


Table: AX_dbo_EMPLTABLE

	DATAAREAID	EMPLID	Name	D
	ceu	7225	Null	83
	ceu	7230	Null	84
	ceu	7231	Null	85
	ceu	7232	Null	86
	ceu	7240	Null	87
	ceu	9001	Null	88
	ceu	9002	Null	89
▶	ceu		Missing Salesperson	90

Incremental Loading

Incremental loading facilitates faster load times by allowing an organization to load only the most recent transactional data into their data warehouse and staging databases. This can be beneficial when the volume of transactional data in the data source causes scheduled execution times to take longer than desired. Incremental loading can also help facilitate more reasonable execution times when multiple updates throughout the day are desired.

Full Load Process

The default load plan during scheduled execution is a Full Load. During a Full Load, all of the tables and fields that are used in a project are completely refreshed, and the full data set is moved from the data source to the staging database and data warehouse. During the full load process, the existing tables in the staging database and data warehouse are truncated, which removes all of the existing data first, and then the new data is subsequently loaded from the data source.

Incremental Load Process

During the incremental load process TX2014 looks at a field or fields that are defined by the user to automatically determine the amount of data that will be moved over. During the first deployment after incremental loading has been enabled, TX2014 will create additional tables in the staging database and data warehouse that have an _INCR or _I suffix. TX2014 will then do a Full Load to bring over all of the required data from the data source. During subsequent execution of the project, truncation is disabled on these tables so that the original data is not removed. TX2014 then determines which records have been added to the data source since the last load and only transfers these new records to the appropriate tables in the staging database and data warehouse. This can dramatically cut down on the amount of time necessary for execution of the objects in the project.

Does My Organization Need Incremental Loading Enabled?

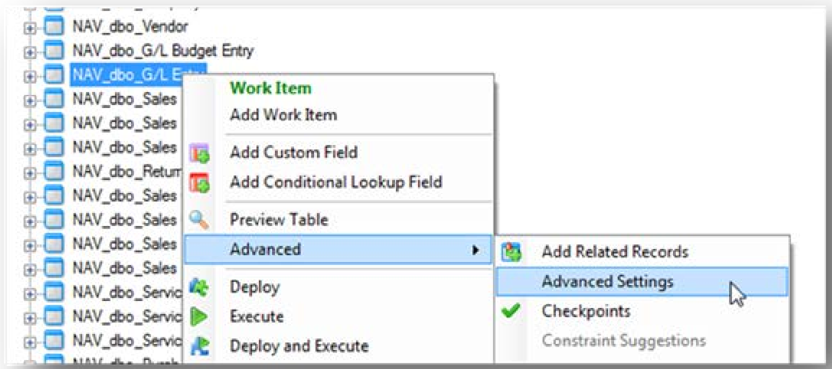
A common misconception is that candidacy for incremental loading is directly dependent on the size of the data source (NAV, GP, AX, etc.). The best indicator as to whether or not incremental loading should be enabled, is not the amount of data that needs to be transferred, but is actually the amount of time that it takes to transfer the data. Every organization has unique environments in which TX2014 is installed. One organization may experience an execution time of 30 minutes to transfer 50gb of data while another organization may experience only 10 minutes to transfer the same amount. Once the amount of time required to do a full load begins to run longer than the organization's execution strategy allows(nightly updates, multiple updates throughout the day, etc.), then incremental loading becomes a viable option.

How to Enable Incremental Loading

The first step in setting up an incremental loading plan is to identify those tables which should have incremental loading enabled. Most smaller or summary level tables such as, Customer, Item, and G/L Account tables, will not generally impact performance as they will not have a substantial number of records. The tables that are candidates to have incremental loading enabled will be larger transaction tables, such as those that contain large volumes of general ledger and inventory transactions. The examples used in this document will use the **G/L Entry** table from Dynamics NAV. These concepts are universal and apply to all data sources.

Enabling Incremental Loading for Staging Database Tables

1. Go to the desired table in the staging database, right-click the table name, and select **Advanced -> Advanced Settings.**



2. Check the box for **Source Based Incremental Loading**. Click **OK.**

Note: If a red "X" denoting an error appears near the **Physical Valid Table** checkbox, it means that this box must be checked in order to use incremental loading on this table. Check the box for **Physical Valid Table** prior to clicking **OK.**

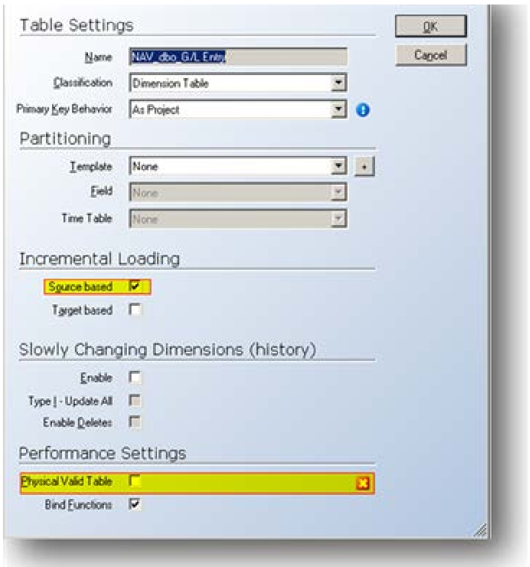


Table Settings

Name: NAV_dbo_G/L Entry

Classification: Dimension Table

Primary Key Behavior: As Project

Partitioning

Template: None

Field: None

Time Table: None

Incremental Loading

Source based ✓

Target based ☐

Slowly Changing Dimensions (history)

Enable ☐

Type I - Update All ☐

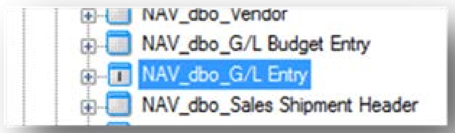
Enable Deletes ☐

Performance Settings

Physical Valid Table ☐

Bind Functions ✓

The table icon in the staging database will now have an "I" icon identifying it as an incrementally loaded table.

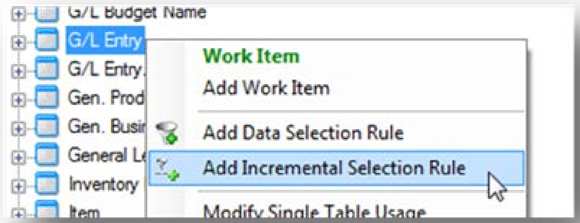


Note: Source based incremental loading is the most common form of incremental loading and is used when there is a known field that represents new data. This could represent an identifier field, an entry number, or possibly a date. Target based incremental loading is used when there is no definite way to know which records have been added or changed in the data source since the last incremental update. Target based incremental loading is covered later in this document.

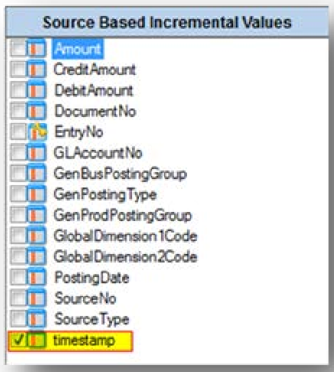
Note: In order for incremental loading to be successful, the table must have a primary key defined. In all of the timeXtenderprojects available for download on the CubeStore, the most common tables used with incremental loading already have the proper primary keys defined. To define a field or fields to be used as the primary key for the table, right-click the field name, and select **Include in Primary Key**.

It is very important that primary keys are defined correctly!If you are not sure about which fields comprise the correct primary key for a table, consult with your ERP software provider or contact a member of the timeXtenderteam.

3. Locate the table in the **Data Source** section at the bottom of the Data tab. Right-click the table and name, and select **Add Incremental Selection Rule.**



4. Check the box(es) that mark the fields identifying which records have been added or changed since the last incremental load. These will ideally be fields that are generated by the system and incremented sequentially when new records are added.



Source Based Incremental Values

- [ ] Amount
- [ ] CreditAmount
- [ ] DebitAmount
- [ ] DocumentNo
- [ ] EntryNo
- [ ] GLAccountNo
- [ ] GenBusPostingGroup
- [ ] GenPostingType
- [ ] GenProdPostingGroup
- [ ] GlobalDimension1Code
- [ ] GlobalDimension2Code
- [ ] PostingDate
- [ ] SourceNo
- [ ] SourceType
- [✓] timestamp

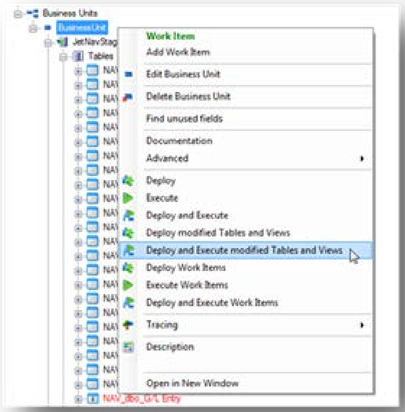
Note: In Dynamics NAV the Timestamp field is generally regarded as the best selection. When records are added or updated, this field is updated in the data source, which will always flag these as records that need to be added or updated by TX2014.

Note: In Dynamics GP, the **DEX_ROW_ID** field is generally regarded as the best selection. This field is automatically incremented by GP when new records are added.

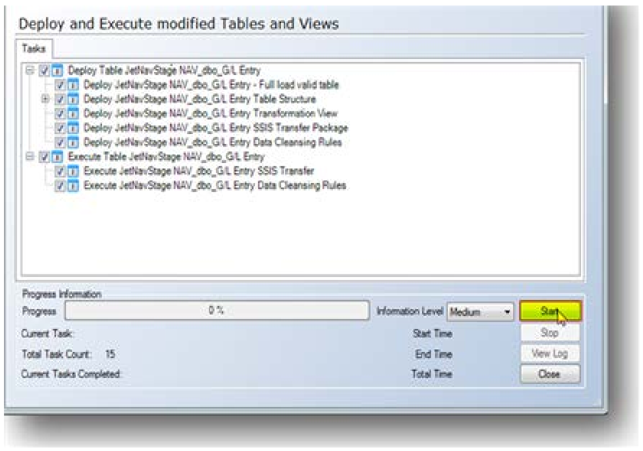
Note: In Dynamics AX, the **RECID** field is generally regarded as the best selection. This field is automatically incremented by AX when new records are added.

Repeat the steps above for all tables to which incremental loading will be enabled.

5. Right-click the **Business Unit**, and select **Deploy and Execute Modified Tables and Views.**



6. Click the **Start**button to initiate the first full load of the tables with source based incremental loading now enabled.



Deploy and Execute modified Tables and Views

Tasks

```
⊟ ☑ Deploy Table JetNavStage NAV_dbo_G/L Entry
    ☑ Deploy JetNavStage NAV_dbo_G/L Entry - Full load valid table
  ⊞ ☑ Deploy JetNavStage NAV_dbo_G/L Entry Table Structure
    ☑ Deploy JetNavStage NAV_dbo_G/L Entry Transformation View
    ☑ Deploy JetNavStage NAV_dbo_G/L Entry SSIS Transfer Package
    ☑ Deploy JetNavStage NAV_dbo_G/L Entry Data Cleansing Rules
⊟ ☑ Execute Table JetNavStage NAV_dbo_G/L Entry
    ☑ Execute JetNavStage NAV_dbo_G/L Entry SSIS Transfer
    ☑ Execute JetNavStage NAV_dbo_G/L Entry Data Cleansing Rules
```

Progress Information

Progress: 0 %

Information Level: Medium

Current Task:

Total Task Count: 15

Current Tasks Completed:

Start Time

End Time

Total Time

The necessary incremental tables will be automatically added to the staging database and populated with the latest incremental values. The next time that the table is executed, TX2014will query these tables to determine the last record that was previously loaded and will only extract data from the data source that occurred after the last execution.

Enabling Incremental Loading for Data Warehouse Tables

The steps for enabling incremental loading on a table in the data warehouse are very similar to the steps above for enabling them on a table in the staging database. In the staging database, the incremental loading was enabled on the staging database, and the data selection rule was set in the data source, both of these are enabled on the data warehouse table

How to Implement Target Based Incremental Loading

Target based incremental loading is primarily used when there are no identifying fields that determine which records have been added since the last incremental update.

With target based incremental loading,

- All of the data is moved over from the data source

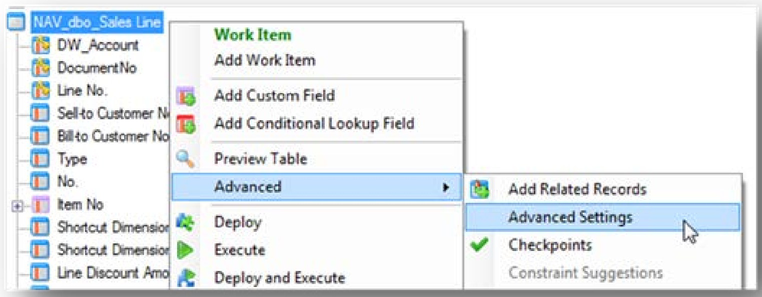
- Records are compared against the existing records in the table

- Only new, updated, or deleted records are added to the staging database or data warehouse.

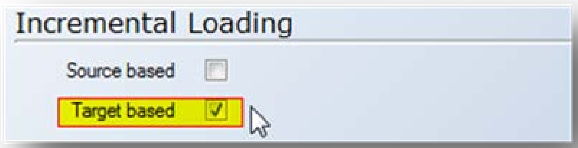
Note: Target based incremental loading is not as fast as source based incremental loading, but is faster than a full load strategy. This is due to the method needed to handle these types of tables.

To enable target based incremental loading for a table:

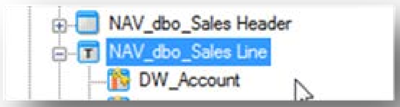
1. Right-click the table name, and select **Advanced -> Advanced Settings.**



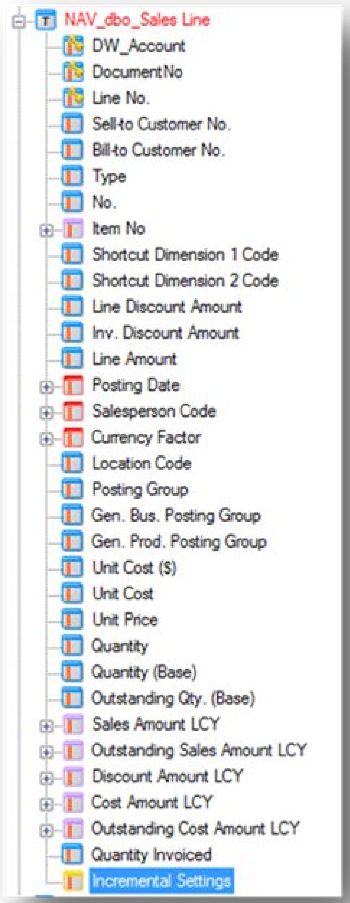
2. Check the box for **Target Based**, and click **OK.**



The table icon will now have a “T” in it identifying this table as one that has target based incremental loading enabled.

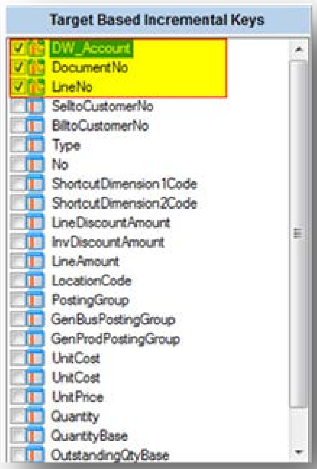


At the bottom of the list of fields for the table there is now an option for **Incremental Settings.**



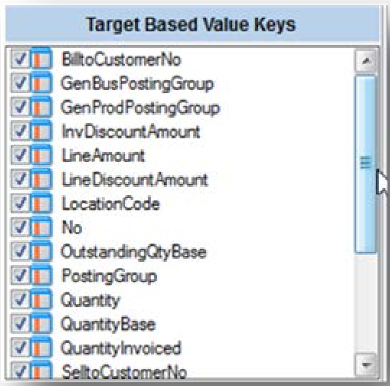
Clicking on this will populate the target based incremental load selection window on the right-hand side of the screen.

The first pane represents the **Target Based Incremental Keys**. The field or fields that represent the primary key for the table should be checked.


Target Based Incremental Keys

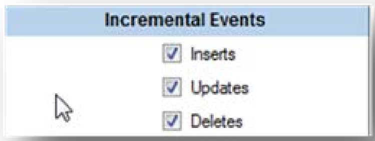
- [✓] DW_Account
- [✓] DocumentNo
- [✓] LineNo
- [ ] SelltoCustomerNo
- [ ] BilltoCustomerNo
- [ ] Type
- [ ] No
- [ ] ShortcutDimension1Code
- [ ] ShortcutDimension2Code
- [ ] LineDiscountAmount
- [ ] InvDiscountAmount
- [ ] LineAmount
- [ ] LocationCode
- [ ] PostingGroup
- [ ] GenBusPostingGroup
- [ ] GenProdPostingGroup
- [ ] UnitCost
- [ ] UnitCost
- [ ] UnitPrice
- [ ] Quantity
- [ ] QuantityBase
- [ ] OutstandingQtyBase

The second pane represents the **Target Based Value Keys**. TX2014will create a hash key field based on the field values for all of the fields selected in this window. This is what will be used to determine if a record has been updated. Check all of the fields that would represent a change in this table. In this example, all fields have been selected.


Target Based Value Keys

- [✓] BilltoCustomerNo
- [✓] GenBusPostingGroup
- [✓] GenProdPostingGroup
- [✓] InvDiscountAmount
- [✓] LineAmount
- [✓] LineDiscountAmount
- [✓] LocationCode
- [✓] No
- [✓] OutstandingQtyBase
- [✓] PostingGroup
- [✓] Quantity
- [✓] QuantityBase
- [✓] QuantityInvoiced
- [✓] SelltoCustomerNo

The third pane represents the **Incremental Events** that TX2014will take into consideration.


Incremental Events

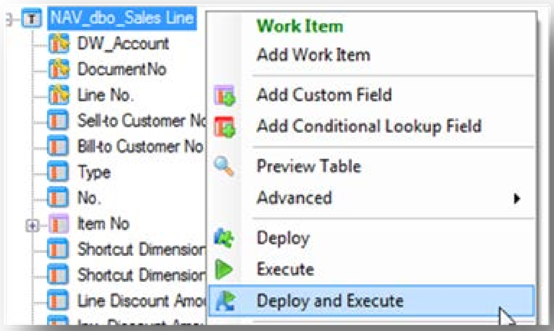
- [✓] Inserts
- [✓] Updates
- [✓] Deletes

Inserts represent any new records that have been inserted based on the primary key on the table.

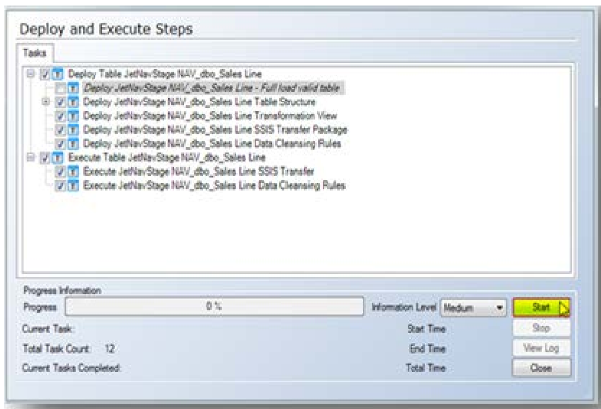
Updates represent any records that have modified or been changed since the last incremental update. This is determined based on the **Target Based Value Keys** selected.

Deletes represent any records that previously existed in the data source but no longer exists based on the primary key. These will be removed from the staging database or data warehouse table.

3. Right-click on the table name and select **Deploy and Execute** to perform the initial load of the table.



4. Click the **Start** button to begin the deployment and execution process.



Deploy and Execute Steps

Tasks

- Deploy Table JetNavStage NAV_dbo_Sales Line
  - Deploy JetNavStage NAV_dbo_Sales Line - Full load valid table
  - Deploy JetNavStage NAV_dbo_Sales Line Table Structure
  - Deploy JetNavStage NAV_dbo_Sales Line Transformation View
  - Deploy JetNavStage NAV_dbo_Sales Line SSIS Transfer Package
  - Deploy JetNavStage NAV_dbo_Sales Line Data Cleansing Rules
- Execute Table JetNavStage NAV_dbo_Sales Line
  - Execute JetNavStage NAV_dbo_Sales Line SSIS Transfer
  - Execute JetNavStage NAV_dbo_Sales Line Data Cleansing Rules

Progress Information

Progress: 0 %

Information Level: Medium

Current Task:

Total Task Count: 12

Current Tasks Completed:

Start Time

End Time

Total Time

During this process, all of the data will be loaded, and the hash keys for the target based incremental load will automatically be generated by TX2014. Subsequent executions of the table will only load and process data cleansing operations on records that have been added, modified, or deleted from the data source since the last execution.

User Defined Functions and Stored Procedures

TX2014utilizes the built-in functions of Microsoft SQL Server. However, you can also create your own functions and procedures to extend the functionality of TX2014.

User Defined Functions

User defined functions allow you to define your own Transact SQL functions. A user defined function returns a table or a single data value, also known as a scalar value. You can, for example, create a function that can be used to perform complex calculations.

User Defined Stored Procedures

User defined stored procedures allow you to define you own Transact SQL stored procedures. You can, for example, create a stored procedure that can be called from the execution package.

To Add User Defined Functions

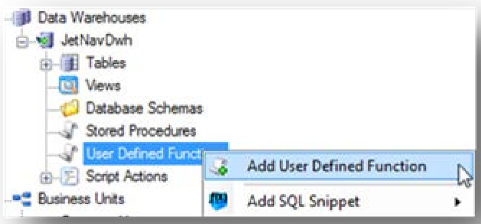
You can add user defined functions to both the staging database and to the data warehouse.

1. Expand **Business Units**, and then expand the staging database.

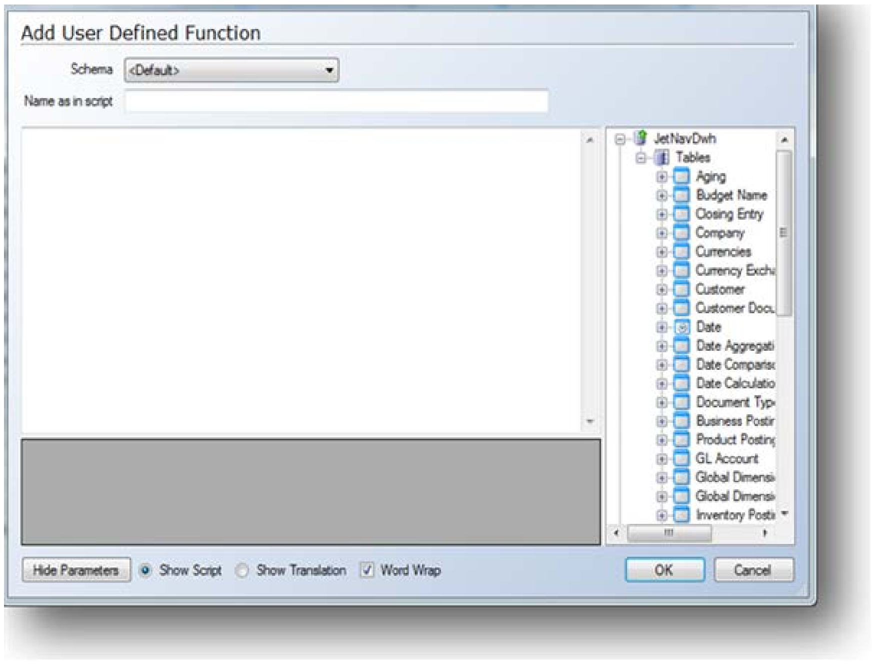
   - OR -

   Expand **Data Warehouses**, and then expand the preferred data warehouse.

2. Right-click **User Defined Functions**, and then select **Add User Defined Function**.



A window with the following options will appear:



3. In the **Name** field, type a name for the function.

4. In the text box, write or paste in the user defined function.

5. A **Script Action** can then be created, if necessary, to call the User Defined Function.

To Add User Defined Stored Procedures

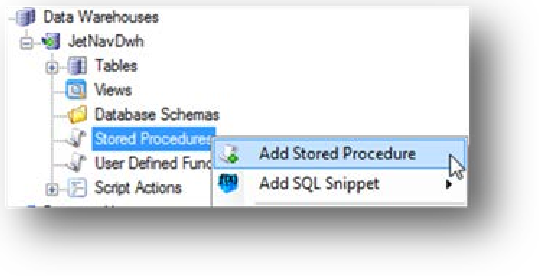
You can add user defined procedures to both the staging database and to the data warehouse.

1. Expand **Business Units**, and then expand the staging database.

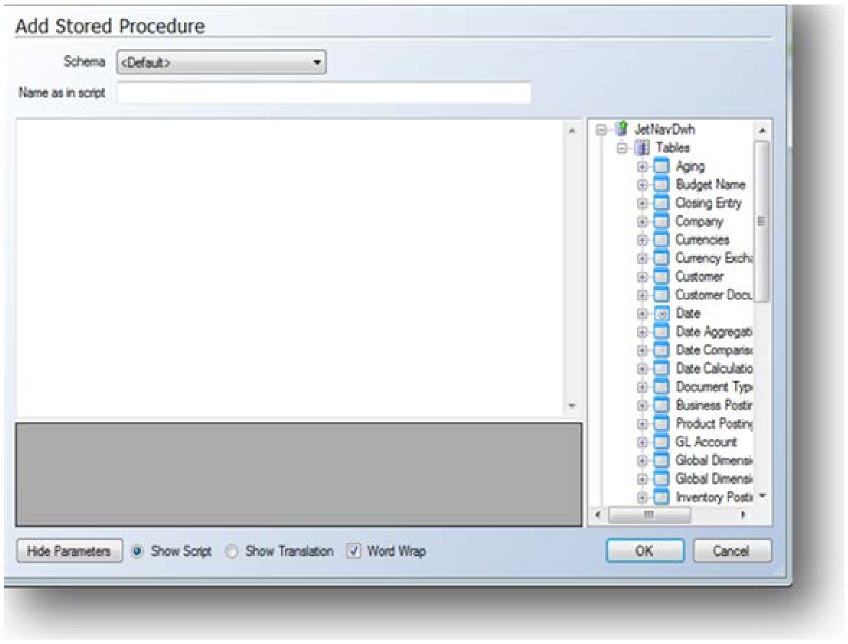
   - OR -

   Expand **Data Warehouses**, and then expand the data warehouse.

2. Right-click **Stored Procedures**, and then select **Add Stored Procedure**.



A window with the following options will appear:



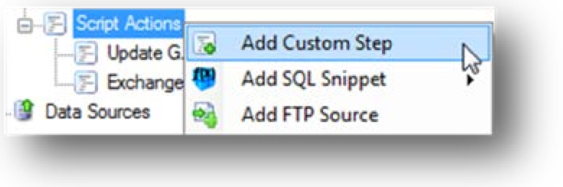
3. In the **Name** field, type a name for the procedure.

4. In the text box, write or paste in the user defined stored procedure.

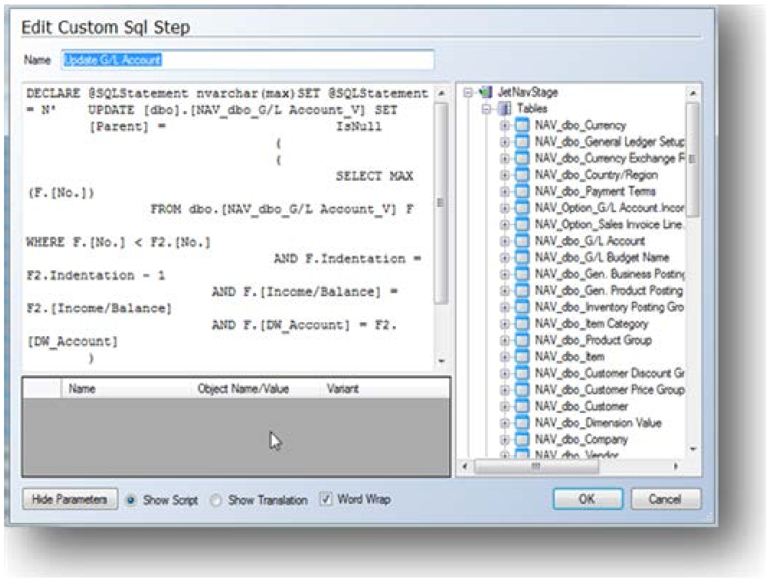
5. A **Script Action** can then be created, if necessary, to call the stored procedure.

Script Actions
Script Actions are SQL scripts that can be added to TX2014that can use User Defined Functions and Stored Procedures that have already been created, as well as a variety of other tasks. Script Actions can then be associated with a table via a configurable trigger to control when the Script Action will execute in relation to the execution of the table.

Adding a Script Action
1. At the bottom of the **Data Warehouse** or **Staging Database** hierarchy, right-click on the **Script Actions** node, and select **Add Custom Step.**



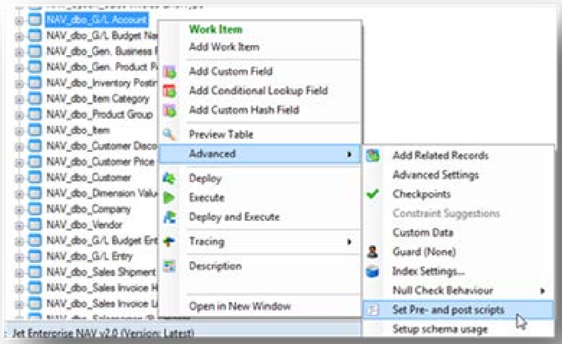
2. The **Edit Custom SQL Script** window will open. Type or paste the desired SQL script into the window, and click **OK.**


Edit Custom Sql Step

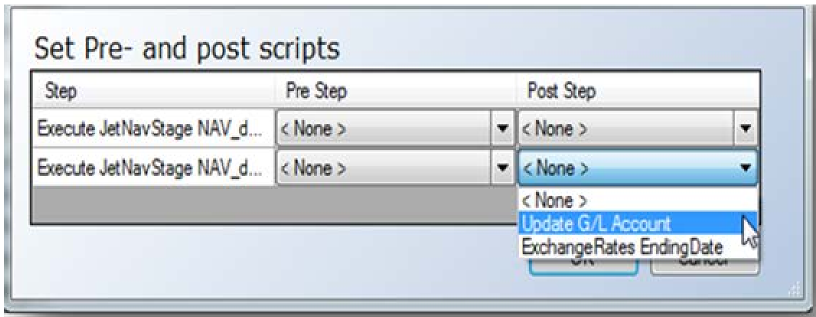
Name: Update G/L Account

```
DECLARE @SQLStatement nvarchar(max)SET @SQLStatement
= N'    UPDATE [dbo].[NAV_dbo_G/L Account_V] SET
        [Parent] =                          IsNull
                                  (
                                  (
                                          SELECT MAX
(F.[No.])
             FROM dbo.[NAV_dbo_G/L Account_V] F

WHERE F.[No.] < F2.[No.]
                                  AND F.Indentation =
F2.Indentation - 1
                         AND F.[Income/Balance] =
F2.[Income/Balance]
                         AND F.[DW_Account] = F2.
[DW_Account]
        )
```

Associating a Script Action with a Table
1. Identify the table to associate the **Script Action** with, right-click on the table name, and go to **Advanced -> Set Pre- and Post Scripts.**



2. The **Set Pre- and Post Scripts** window will appear. Select the drop-down list for the appropriate step in the table execution process to associate the script with, and select the name of the script. Click **OK.**


Set Pre- and post scripts

Step	Pre Step	Post Step
Execute JetNavStage NAV_d...	< None >	< None >
Execute JetNavStage NAV_d...	< None >	< None >

The possible execution steps at which to call the script are:

Transfer Pre Step: This will cause the script to be called prior to the beginning of the data transfer process which will move data into the table.

Transfer Post Step: This will cause the script to be called after the transfer of data into the table has been complete but prior to the beginning of the data cleansing process

Data Cleansing Pre Step: This will cause the script to be called after the transfer of data into the table has been complete but prior to the beginning of the data cleansing process

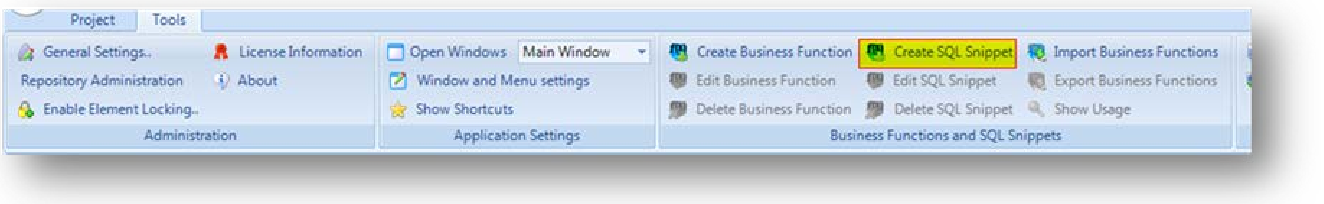
Data Cleansing Post Step: This will cause the script to be called after the data cleansing process has completed.

SQL Snippets

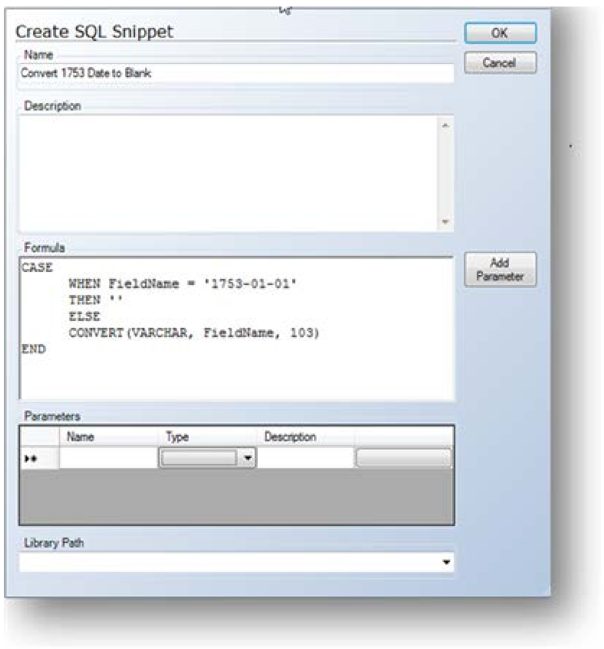
In TX2014 you can create commonly used pieces of SQL code and parameterize them as SQL Snippets. This allows frequently used pieces of functionality to be saved once and then easily deployed across different tables or fields without recreating all of the SQL functionality.

How to Create a SQL Snippet

1. On the **Tools** menu, click the **Create SQL Snippet** button.



A window with the following options will appear:



Name: Convert 1753 Date to Blank

Description:

Formula:

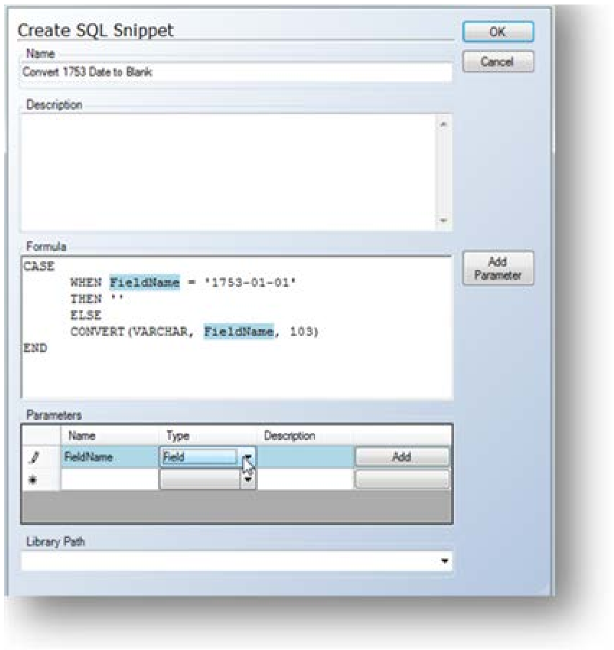
```
CASE
        WHEN FieldName = '1753-01-01'
        THEN ''
        ELSE
        CONVERT(VARCHAR, FieldName, 103)
END
```

Parameters

Name	Type	Description

Library Path

2. Type and **Name** and **Description** for the SQL Snippet. The **Description** is optional but allows other users to know what the Snippet will do.
3. Type or paste the SQL command into the **Formula** section.
4. For any variables (in this example, **FieldName**), highlight the variable, and click **Add Parameter**. This will add the highlighted text as a parameter name in the **Parameters** section.



Name: Convert 1753 Date to Blank

Description:

Formula:

```
CASE
      WHEN FieldName = '1753-01-01'
      THEN ''
      ELSE
      CONVERT(VARCHAR, FieldName, 103)
END
```

Parameters:

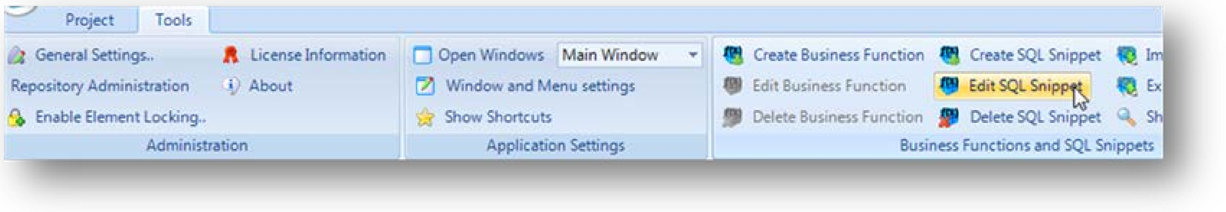
Name	Type	Description
FieldName	Field	

5. Change the **Type** to match what the variable represents. The available options are: Table, Field, Database, User Defined Function, Stored Procedure, and Value.

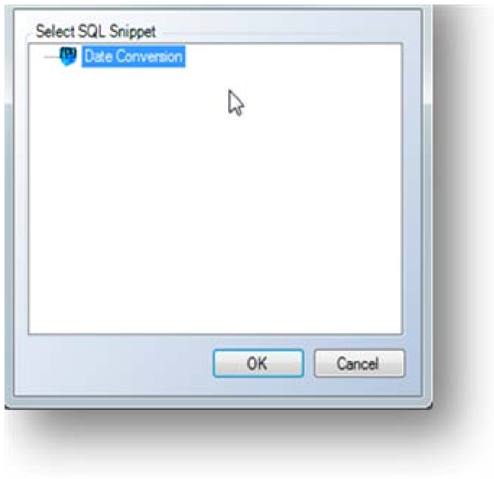
6. Click **OK** to save the SQL Snippet.

How to Edit a SQL Snippet

1. On the **Tools** menu, select **Edit SQL Snippet.**



2. Select the SQL Snippet to be edited from the list, and click **OK**.

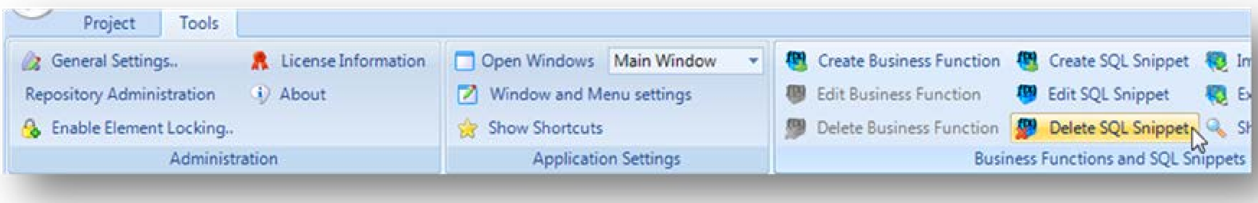


3. Adjust the SQL Snippet as needed.

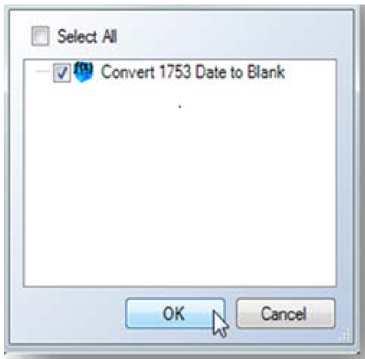
4. Click **OK** to save the SQL Snippet.

How to Delete a SQL Snippet

1. On the **Tools** menu, select **Delete SQL Snippet.**

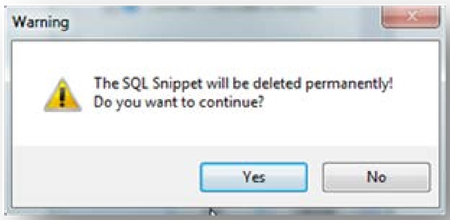


2. Check the box next to the SQL Snippet(s) to be deleted. Alternatively, **Select All** can be checked to automatically select all SQL Snippets.



3. Click **OK.**

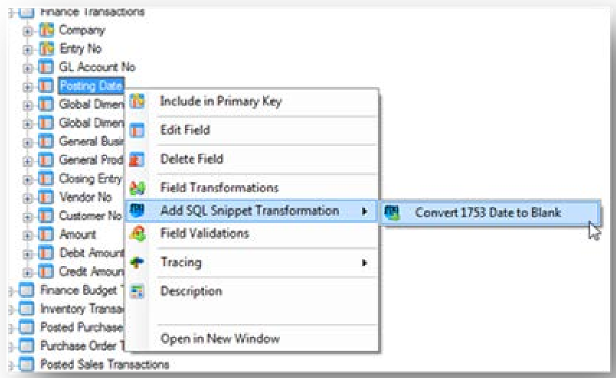
4. Click **Yes** to permanently delete the SQL Snippet(s).


Warning

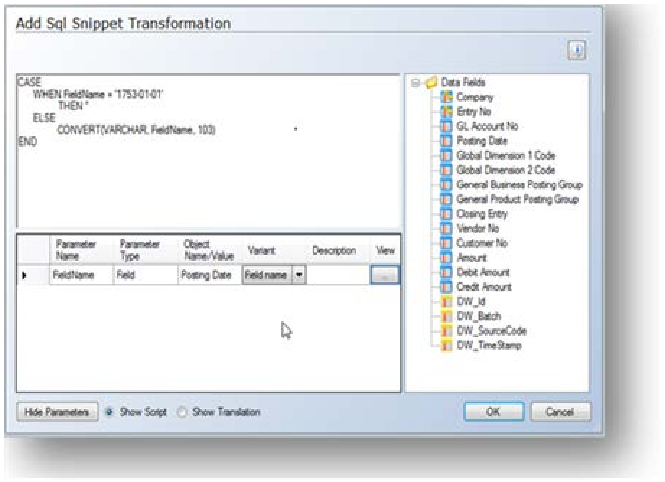
The SQL Snippet will be deleted permanently!
Do you want to continue?

How to Implement a SQL Snippet

1. Right-click on the field to add the SQL Snippet to. Go to **Add SQL Snippet Transformation**, and select the desired SQL Snippet from the available list.



2. Drag the desired field(s) from the **Data Fields** pane on the right, and drop the field on the **Object Name/Value** column for the desired variable. The **Object Name/Value** column and **Variant** column will populate automatically.



```
CASE
   WHEN FieldName = '1753-01-01'
        THEN ''
   ELSE
        CONVERT(VARCHAR, FieldName, 103)
END
```

	Parameter Name	Parameter Type	Object Name/Value	Variant	Description	View
▸	FieldName	Field	Posting Date	Field name		...

3. Click **OK**.

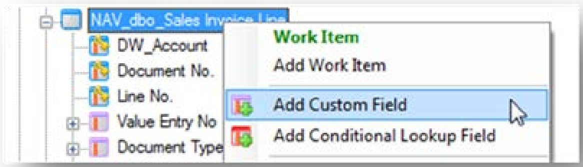
Strongly Typed Values in Custom Fields

Strongly Typed Values are available for use in Custom Fields within a project. Strongly typed values closely coincide with other objects within the project. For example, if there is a field named **Sales Amount** that is used in a custom field that is not strongly typed, the custom field will cause an error if the **Sales Amount** field is renamed to something else. The reason for this is that the values are simply stored as text. Strongly typed values are inherently linked to the object it refers to, so if the object is renamed, it will be dynamically changed in the custom field as well.

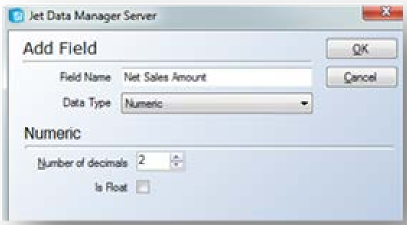
How to Use Strongly Typed Values in Custom Fields

1. Insert a custom field by right-clicking on the desired table, and selecting **Add Custom Field.**

Note: In the data warehouse the option will simply read **Add Field**.

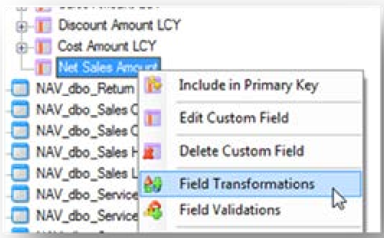


2. Define the name and the field information for the custom field, and click **OK**.

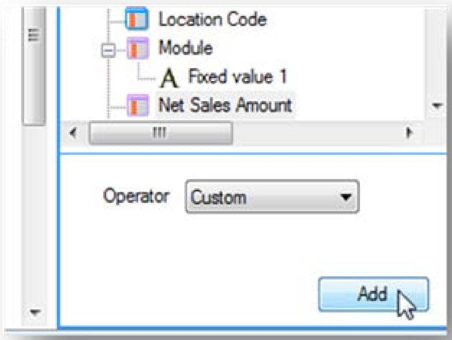


Field Name: Net Sales Amount
Data Type: Numeric
Number of decimals: 2

3. Right-click on the custom field that was just added, and select **Field Transformations**.



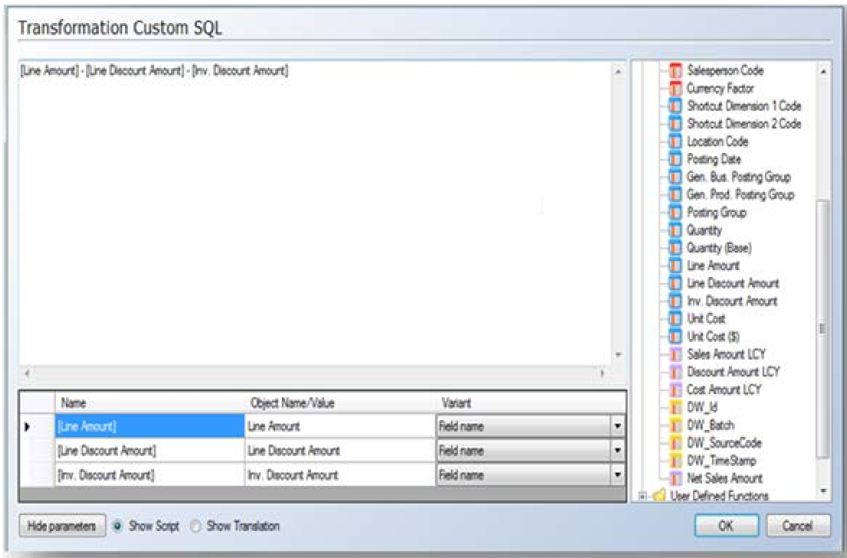
4. For the **Field Transformation**, set the **Operator** to **Custom**, and click **Add.**



Operator: Custom

5. The fields in the pane on the right can be dragged and dropped into the main work area in the center of the window. As the fields are dropped, the parameters are automatically created at the bottom. Click **OK.**

If the underlying objects are renamed, they will dynamically be updated in the custom field without any user interaction required.



```
[Line Amount] - [Line Discount Amount] - [Inv. Discount Amount]
```

Name	Object Name/Value	Variant
[Line Amount]	Line Amount	Field name
[Line Discount Amount]	Line Discount Amount	Field name
[Inv. Discount Amount]	Inv. Discount Amount	Field name

[✓ Show Script] Show Translation

Creating Views

A view is a virtual table in your data warehouse or in your staging database where you can group together information from two or more tables in your data source. Views can, for example, be used to provide a user with a simplified view of a table and to limit access to sensitive data. Creating views in TX2014follows the same methodology as creating standard SQL views.

In TX2014 you can create two types of views: views that consist of a subset of columns or rows in one or more tables and views that are joins of one or more tables.

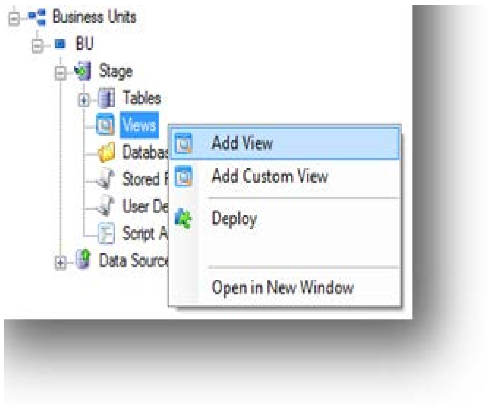
To Create Views Based on Lookup Fields

You can create views in the data warehouse or in the staging database. Creating a view based on a lookup field consists of the following steps. The steps are all carried out in the View dialog.

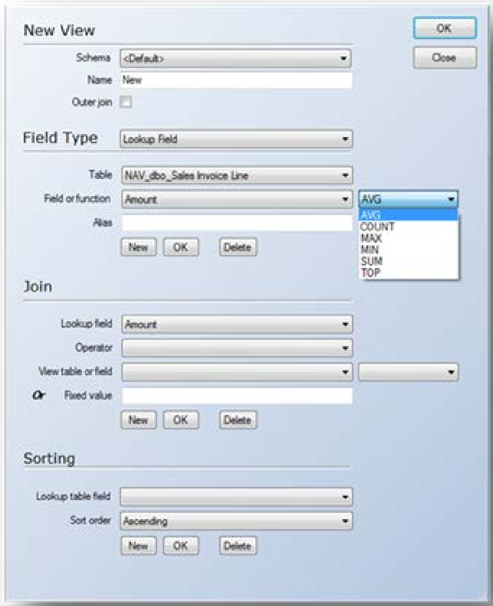
To Create a Lookup Field

1. On the **Data** tab, expand the chosen data warehouse or staging database.

2. Right-click **Views**, and then select **Add View**.



A window with the following options will appear:


New View

Schema <Default>

Name New

Outer join ☐

Field Type Lookup Field

Table NAV_dbo_Sales Invoice Line

Field or function Amount AVG

Alias

Join

Lookup field Amount

Operator

View table or field

Or Fixed value

Sorting

Lookup table field

Sort order Ascending

3. In the **Name** field, type a name for the view.

4. In the **Field Type** list, select **Lookup Field**. The dialog changes so that you can create and specify the properties of the lookup field.

5. In the **Table** list, select the table that holds the lookup field.

6. In the **Field/Function** list, select the preferred field or function, and then specify which values to return. You have the following options:

TOP	Returns the value of the first record in the column
SUM	Returns the sum of all field values in the column
COUNT	Returns the number of records
MAX	Returns the maximum value of the records in the column
MIN	Returns the minimum value of the records in the column

7. In the **Alias** field, type a name for the lookup field if you want the name to be different from the source field name.

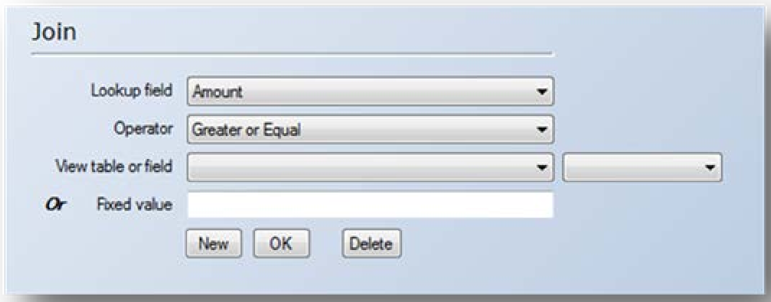
8. Click **OK**. The field is displayed in the **View** pane of the dialog.

9. Click **New** if you want to create another field.

To Specify a Join

You have to specify a join between the view table and the lookup tables.

1. In the Lookup field list, select the field to look up.



2. In the Operator field, select the operator that determines how you want the columns compared.

Equal	Returns values that are equal
Greater	Returns values that are greater than the value of the lookup field or the specified fixed value
Greater or Equal	Returns values that are greater than or equal to the value of the lookup field or the specified fixed value
Less or Equal	Returns values that are less than or equal to the value of the lookup field or the specified fixed value
Less Than	Returns values that are less than the value of the lookup field or the specified fixed value
Not Equal	Returns values that are different from that of the lookup field or the specified fixed value

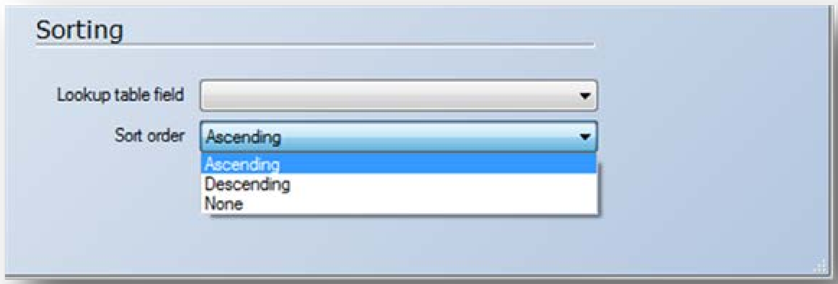
Note:A default inner join is created which only returns results from the rows common to the two joined tables. For the complete set of records from the joined tables, check the **Outer Join** box at the top of the dialog.

3. Click **OK**, and then click **New** if you want to specify a new join.

To Specify a Sort Order

1. In the **Lookup Table** field, select the preferred field.

2. In the **Sort Order** list, select how you want the results sorted. Results can be sorted in either ascending or descending order.

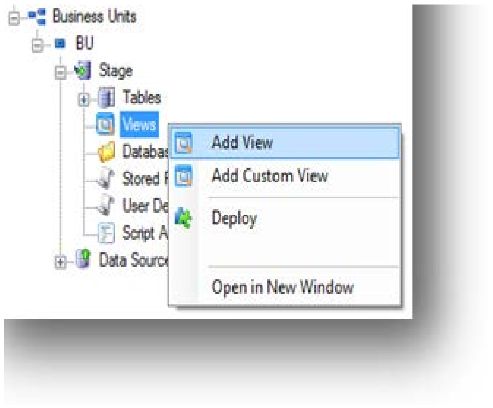


3. Click **OK**, and then click **New** if you want to specify a sort order for another field in the view.

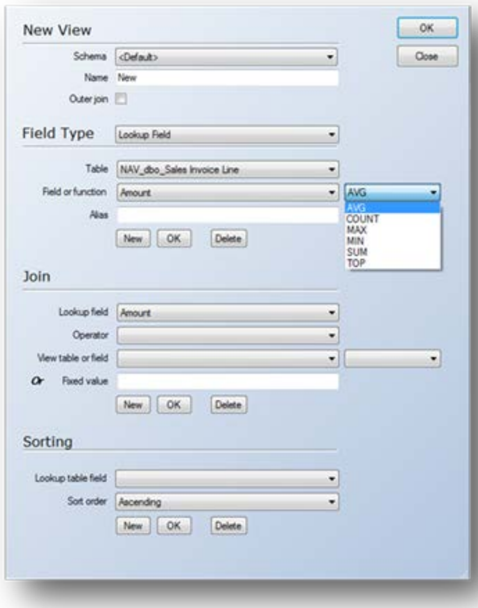
Once you have completed all steps, and created all the joins you need, click **Ok** in the upper right corner of the dialog to create the view.

To Create Views Based on Standard Fields

1. On the **Data** tab, expand the chosen data warehouse or staging database , right-click Views, and then select **Add View**.



A window with the following options will appear:



New View

Schema: <Default>
Name: New
Outer join: ☐

Field Type: Lookup Field

Table: NAV_dbo_Sales Invoice Line
Field or function: Amount | AVG
Alias:

Join

Lookup field: Amount
Operator:
View table or field:
Or Fixed value:

Sorting

Lookup table field:
Sort order: Ascending

2. In the **Name**field, type a name for the view.

3. In the **Field** type field, select **Standard Table Field**.

4. In the **Table** list, select the table you want to retrieve data from.

5. In the **Field** list, select the field you want to use in the view.

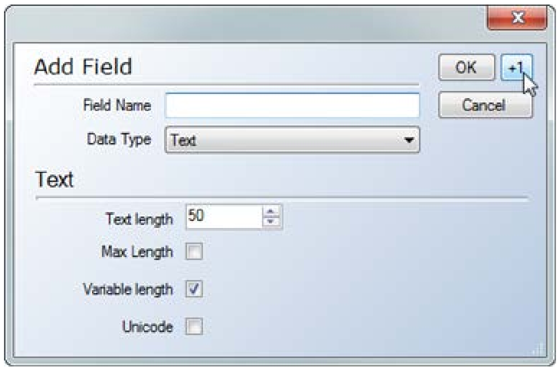
6. In the **Alias** field, type a name for the view, and then click **OK**. The selected field is displayed in the **View** pane.

7. To add more fields from the same table or a field from another table, click **New**, and then repeat steps 3-6. Do this for all the tables you want in the view.

Note: If you want to add fields from more than one table, the tables must be related.

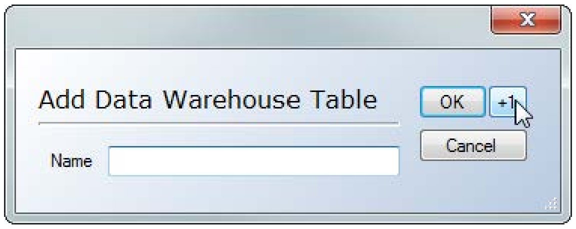
Add multiple Standard measures, tables, and fields easily

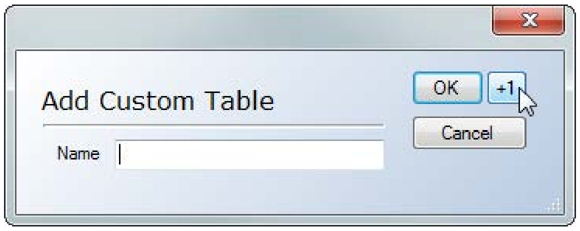
When adding a new table to the data warehouse, you typically want to add a number of fields. For each field, you need to right click, and select Add New Field and finally click OK. To take away those extra clicks, we have added the +1 button:

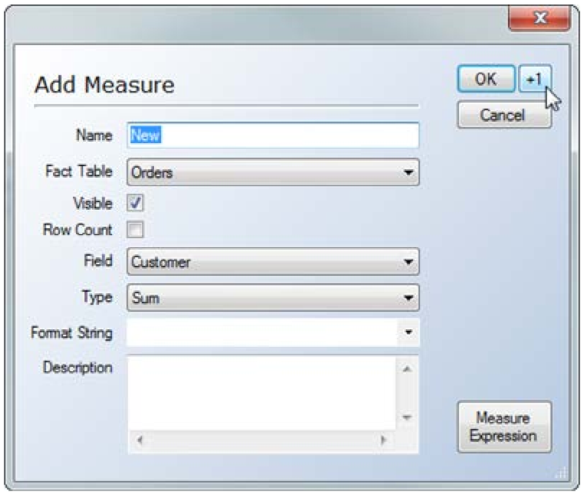


Clicking the +1 Field means OK to this, and please give me one more!

The +1 button has been added to a number of dialogues, including “Add Data Warehouse Table”, “Add Custom Table,” and “Add Standard Measure”.







Add Measure

Name	New
Fact Table	Orders
Visible	☑
Row Count	☐
Field	Customer
Type	Sum
Format String	
Description	

Development Deployment Toolkit

The Development Deployment Toolkit is a collection of tools that provides a dedicated development environment and automatic transfer of the latest version of the project to the production environment. This toolkit ensures that the production environment is always online and available for users. **The Development Deployment Toolkit is an add-on feature that is available for purchase.**

Development Environments

A dedicated Development Environment enables users to work within non-production environments. This is useful when an organization needs to ensure that the production environment is always available for end-users. For example, the organization could have an environment called *"Development"* where changes are made, dimensions are updated, and measures are created. Once these modifications are tested, they can be transferred to the live production environment directly from TX2014.

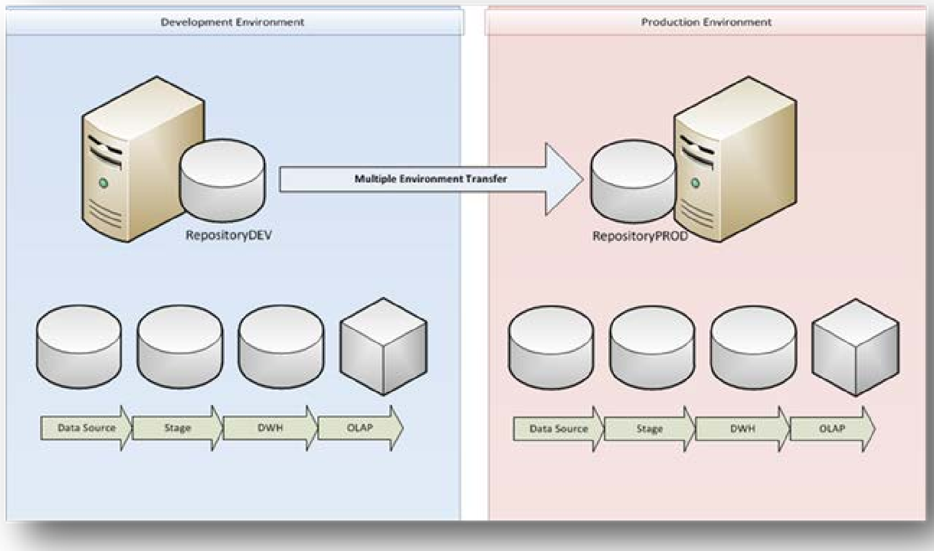
Prerequisites

Before setting up development environments, ensure that the following prerequisites are met:

- All servers used in the development and production environments must have the same version of TX2014installed. This applies to bit version(32-bit or 64-bit) as well.
- Ensure that TX2014 service is installed and started on the server(s) you wish to deploy. Detailed instructions are provided below.
- Ensure that a project repository has been created on all the servers that are used in the production environment and development environment.
- The user account(s) that will be used to set up the multiple environments **may** need to have Read permissions to the Event Log. This can be set up in the Registry Editor under the "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\eventlog" node. You can right-click the "eventlog" node, select **Permissions**, and add the users that will utilize Multiple Environment Deployment and assign them "Read" permissions.

Setting Up a Development Environment

The following example shows multiple environments using a single **Development** server and a single **Production** server.



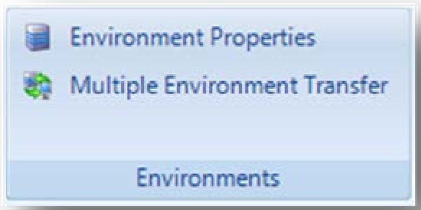
Note: It is possible to set up additional environments as needed. Setup of additional environments follows the same steps listed below.

Production Environment

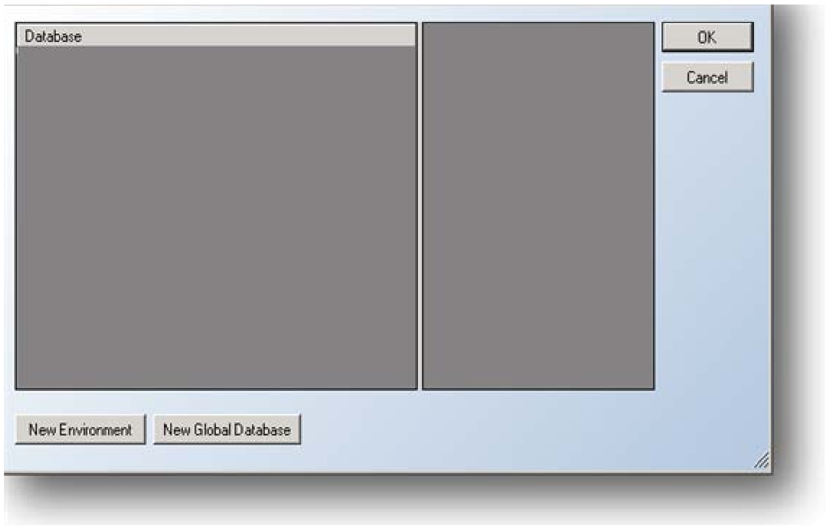
The first step is to set up the **Production** environment on the production server.

Setting Up the Production Server

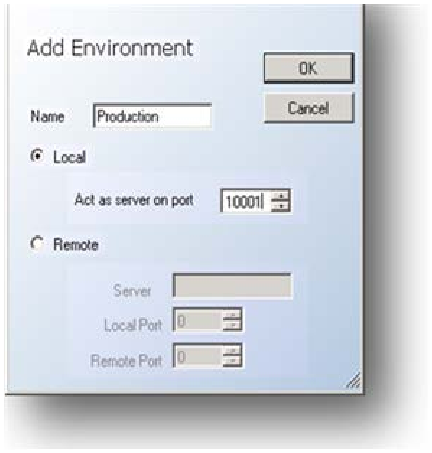
1. Log on to the production server and open TX2014.
2. On the **Tools** tab in TX2014, select **Environment Properties**.



3. The **Add Environments** window will then open.

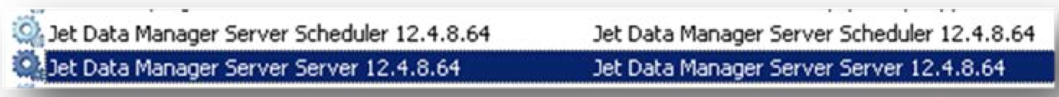


4. Click **New Environment**.
5. The Add Environment window will open:



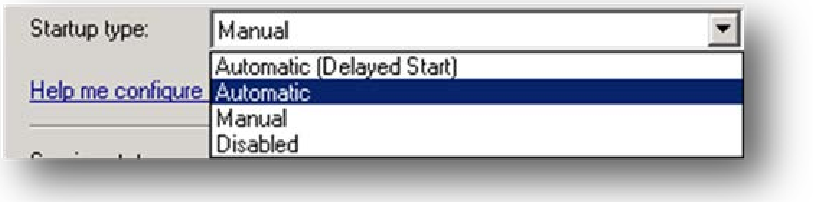
Name: Production
Local (selected)
Act as server on port: 10001
Local Port: 0
Remote Port: 0

6. Assign a name. In this example, the name is “Production”.
7. Select **Local**, as deployment will only be done into this environment and not from it.
8. In the Act As Server On Port field, enter a port to use. Make sure the port is free to avoid any conflicts on the network.
9. Close TX2014.
10. Open **Services** from **Control Panel -> Administrative Tools** on the Production Server, and configure TX2014 Server Service.

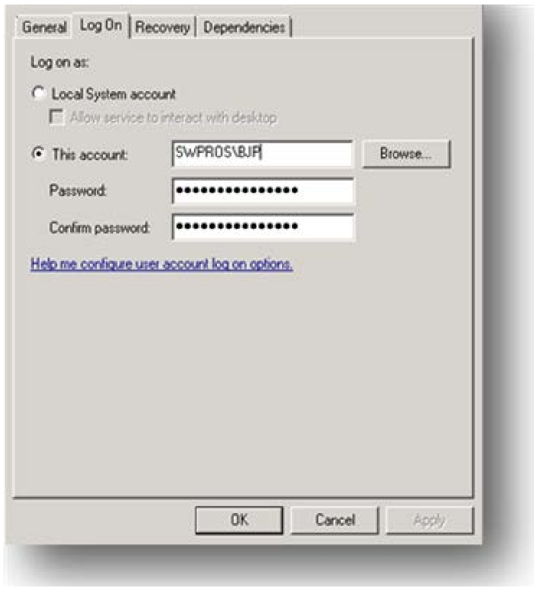


Jet Data Manager Server Scheduler 12.4.8.64 Jet Data Manager Server Scheduler 12.4.8.64
Jet Data Manager Server Server 12.4.8.64 Jet Data Manager Server Server 12.4.8.64

11. The service should be set to start automatically.



The **Log on as** user for the service should be the same Windows account that was used for setting up the **Production** environment.


Log on as:

Local System account

Allow service to interact with desktop

This account: SWPROS\BJP

Password: ••••••••••••••••

Confirm password: ••••••••••••••••

12. **Start** the service.

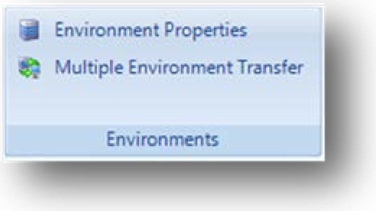
13. **Log off from the production server.**

Development Environment

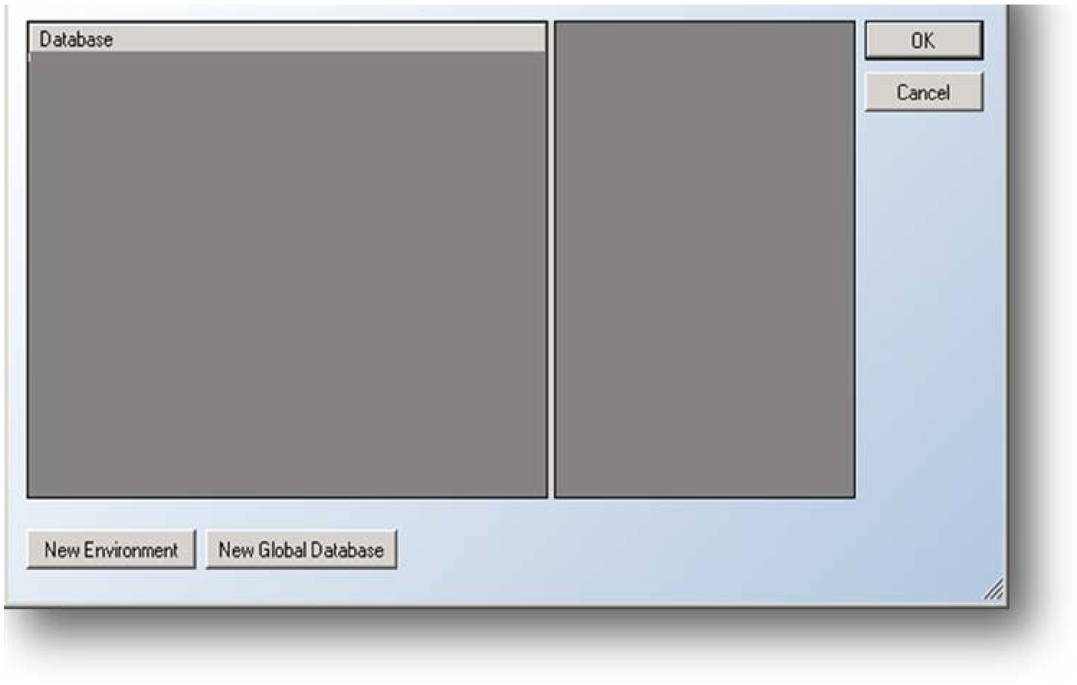
The next step is to set up the development environment on the development server.

Setting Up the Development Server

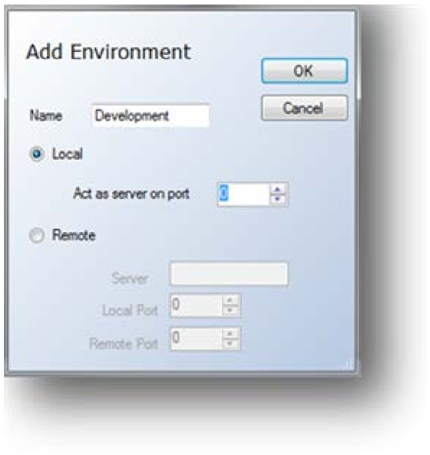
1. Log on to the development server, and open TX2014.
2. On the **Tools** tab in TX2014,select **Environment Properties**.



3. The **Environment Properties**window will open:



4.Click the **New Environment** button to create the development environment. The **Add Environment** window will open:

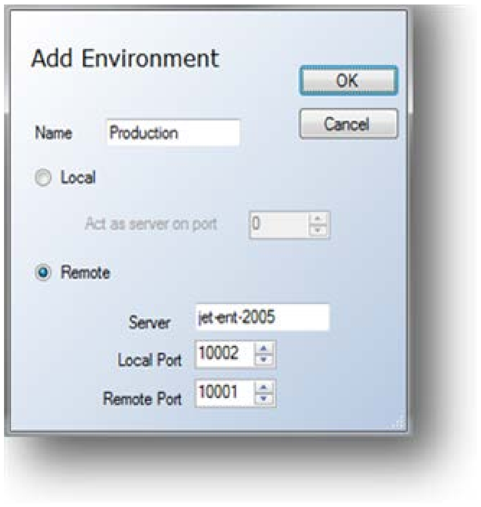


Name: Development
(● Local)
Act as server on port: 0
○ Remote
Server
Local Port: 0
Remote Port: 0

5. Assign a name. In this example, the name is Development.

6. Select **Local**, and leave **Act as Server on Port** set to 0. Click **Ok.**

Click **New Environment** again to specify the production environment on the development server. This is so that the development environment knows where the production environment exists.



Name: Production

( ) Local

Act as server on port: 0

(●) Remote

Server: jet-ent-2005

Local Port: 10002

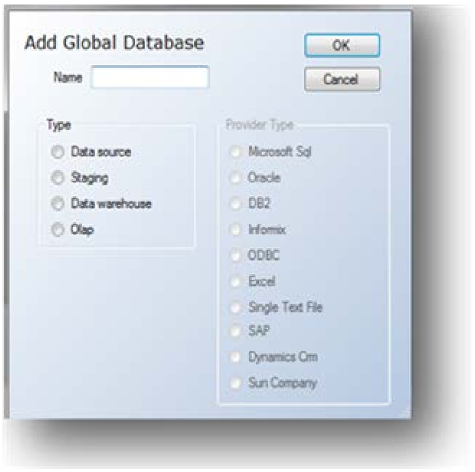
Remote Port: 10001

7. Assign a name. In this example, the name is Production.
8. Select **Remote.**Populate the following fields within the Add Environment Dialog.
    - **Server**: Enter the Server Name or IP address of the production server.
    - **Local Port:**This can be any open port that is not being used by another application.
    - **Remote Port:**This is the port specified in the Local Port in the previous section when setting up the production environment on the production server.

Create Global Databases

Global databases allow TX2014to know where the related databases reside for the Production and Development environments. For example, the location of the Staging Database for both the Production and Development environments will be specified.

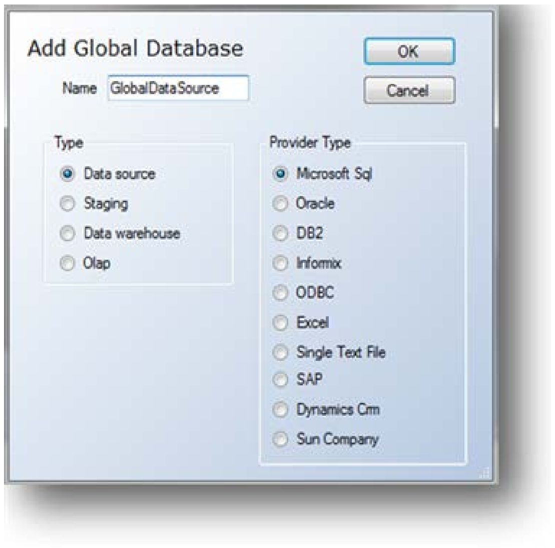
1. Click **New Global Database** from the Environment Dialog window.
2. The Add Global Database window will open.



3. Within the Add Global Database Dialog, you will be creating a series of databases that will be used in your project. You will create the following Global Databases:

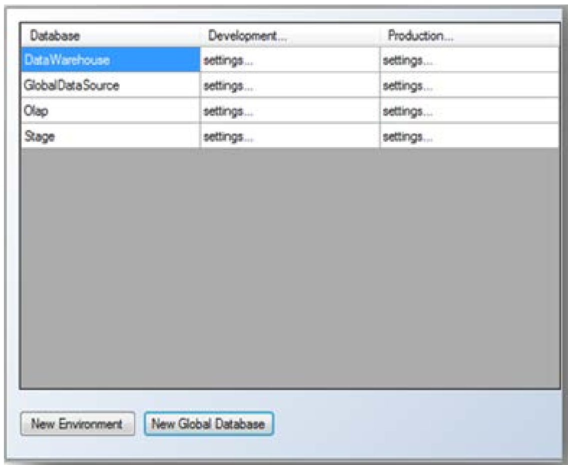
- Data Source
- Staging Database
- Data Warehouse
- OLAP

4. Assign a name to your data source, select**Data Source**in the **Type**section and select the relevant Provider Type. In this example, we will name our global database “GlobalDataSource” and use a provider type of Microsoft SQL.



5. Repeat the previous step for all databases in the project (Staging, Data warehouse, and OLAP).

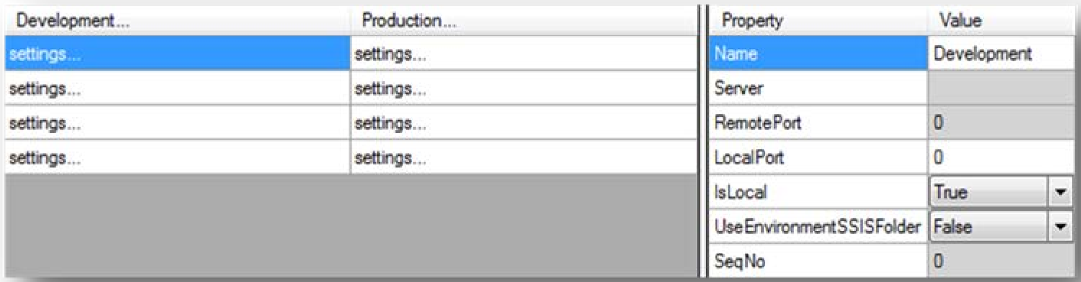
Note: You do not need to select a **Provider Type** for the Staging, Data warehouse, and OLAP. Your results should look similar to those shown below:



Database	Development...	Production...
DataWarehouse	settings...	settings...
GlobalDataSource	settings...	settings...
Olap	settings...	settings...
Stage	settings...	settings...

Configure Global Databases

Within the **EnvironmentProperty**window you should have a data source, data warehouse, OLAP, and staging database. Each environment, Development and Production, has a "**Settings..."** section for each database. You can also click on the environment name to access additional settings.



Development...	Production...
settings...	settings...
settings...	settings...
settings...	settings...
settings...	settings...

Property	Value
Name	Development
Server	
RemotePort	0
LocalPort	0
IsLocal	True
UseEnvironmentSSISFolder	False
SeqNo	0

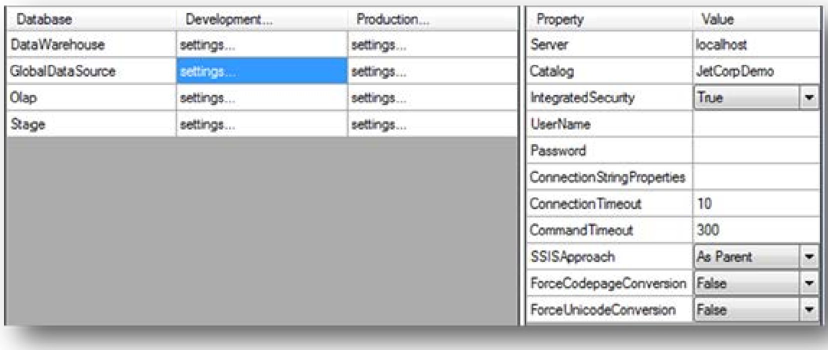
Configuring the Data Source

1. Select the **"Settings…"** field from the data source row in the Development column from the **Environment Properties** window. This will display the Settings pane to the right.

2. **Enter the following information:**

   - Server: This will be the server address of the development server. Since this is currently on the development server, this can be localhost or the name of the server.

   - Name of the database from which data is extracted. This will be the name of the NAV, AX, GP, or other database. In our example, this is JetCorpDemo.



Database	Development...	Production...
DataWarehouse	settings...	settings...
GlobalDataSource	settings...	settings...
Olap	settings...	settings...
Stage	settings...	settings...

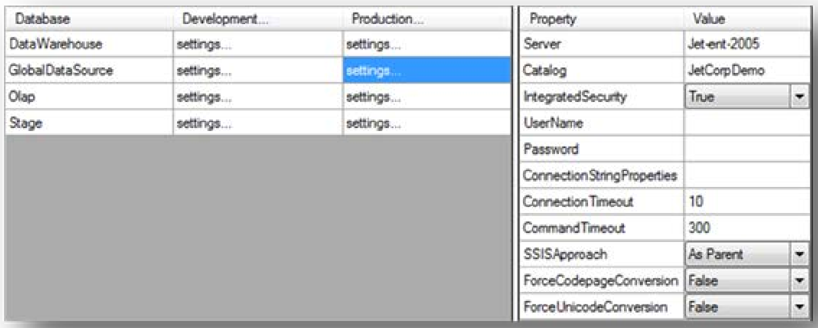
Property	Value
Server	localhost
Catalog	JetCorpDemo
IntegratedSecurity	True
UserName	
Password	
ConnectionStringProperties	
ConnectionTimeout	10
CommandTimeout	300
SSISApproach	As Parent
ForceCodepageConversion	False
ForceUnicodeConversion	False

3. Next click “**Settings…"** on the data source row in the Production column. Enter the following configuration:

-**Server**: This will be the name of the server on the Production Environment. In our example, the server name is jet-ent-2005.

-**Catalog:**This will be the name of the database from which data is extracted. This will be the name of the NAV, AX, GP, or other database. In our example, this is JetCorpDemo.

Note: If you are using your live ERP database for extracting data in both the development and production environments, then the server name and catalog in both the Development and Production columns will be the same.



Database	Development...	Production...
DataWarehouse	settings...	settings...
GlobalDataSource	settings...	settings...
Olap	settings...	settings...
Stage	settings...	settings...

Property	Value
Server	Jet-ent-2005
Catalog	JetCorpDemo
IntegratedSecurity	True
UserName	
Password	
ConnectionStringProperties	
ConnectionTimeout	10
CommandTimeout	300
SSISApproach	As Parent
ForceCodepageConversion	False
ForceUnicodeConversion	False

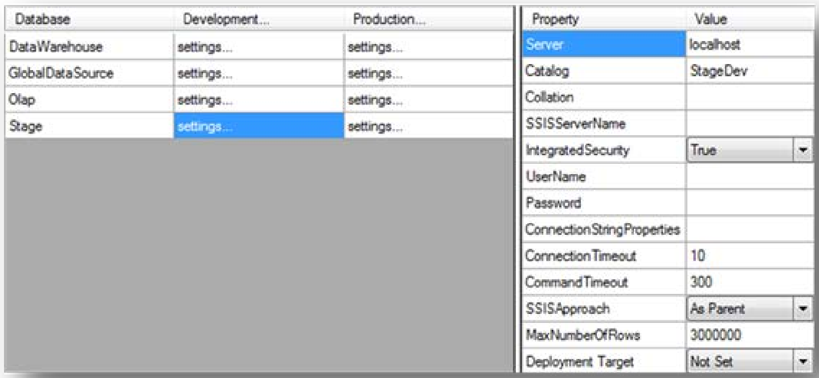
Configuring the Staging Database

1. Next you will need to configure another Global Database for the staging. Click **"Settings…"**on the Stage row in the Development column to display the Settings pane to the right.

2. Enter the following configuration:

- Server: This will be the server address of the development server. In our example, this is localhost.

-Catalog: This will be the name associated with the staging database in the development environment. In our example, we use StageDev.



Database	Development...	Production...
DataWarehouse	settings...	settings...
GlobalDataSource	settings...	settings...
Olap	settings...	settings...
Stage	settings...	settings...

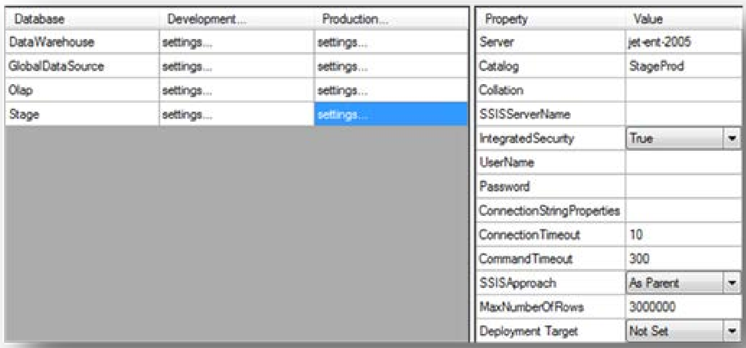
Property	Value
Server	localhost
Catalog	StageDev
Collation	
SSISServerName	
IntegratedSecurity	True
UserName	
Password	
ConnectionStringProperties	
ConnectionTimeout	10
CommandTimeout	300
SSISApproach	As Parent
MaxNumberOfRows	3000000
Deployment Target	Not Set

3. Next Click "Settings..." on the Stage row of the **Production** column.

4. Enter the following configuration:

-**Server**: This will be the name of the server for the Production Environment. In our example, the server name is **jet-ent-2005**.

-**Catalog:**This will be the name associated with your staging database in the production environment. In our example, we use StageProd.



Database	Development...	Production...
DataWarehouse	settings...	settings...
GlobalDataSource	settings...	settings...
Olap	settings...	settings...
Stage	settings...	settings...

Property	Value
Server	jet-ent-2005
Catalog	StageProd
Collation	
SSISServerName	
IntegratedSecurity	True
UserName	
Password	
ConnectionStringProperties	
ConnectionTimeout	10
CommandTimeout	300
SSISApproach	As Parent
MaxNumberOfRows	3000000
Deployment Target	Not Set

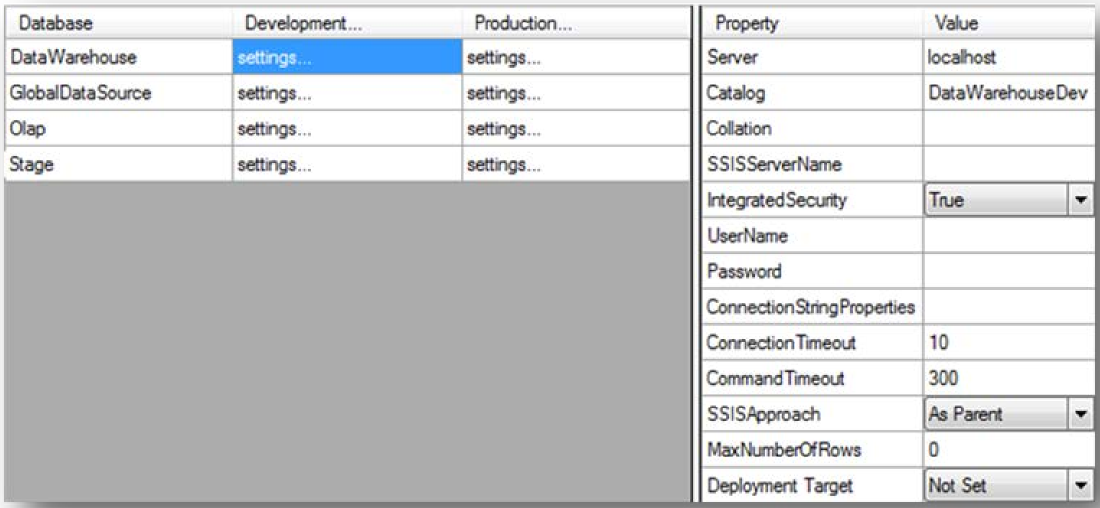
Configuring the Data Warehouse

1. Next you will need to configure another Global Database for the data warehouse. Click "Settings...: on the data warehouse row in the Development column to display the Settings pane to the right.

2. Enter the following configuration:

-Server: This will be the name of the server for the development environment. In our example, this is localhost.

-Catalog: This will be the name associated with your data warehouse in the development environment. In our example, we use DataWarehouseDev.



Database	Development...	Production...
DataWarehouse	settings...	settings...
GlobalDataSource	settings...	settings...
Olap	settings...	settings...
Stage	settings...	settings...

Property	Value
Server	localhost
Catalog	DataWarehouseDev
Collation	
SSISServerName	
IntegratedSecurity	True
UserName	
Password	
ConnectionStringProperties	
ConnectionTimeout	10
CommandTimeout	300
SSISApproach	As Parent
MaxNumberOfRows	0
Deployment Target	Not Set

3. Next Click “Settings..." on the data warehouse row of the **Production** column.

4. Enter the following configuration:

Server: This will be the name of the server for the Production Environment. In our example the server name is **jet-ent-2005**.

Catalog: This will be the name associated with your data warehouse in the production environment. In our example, we use **DataWarehouseDev.**

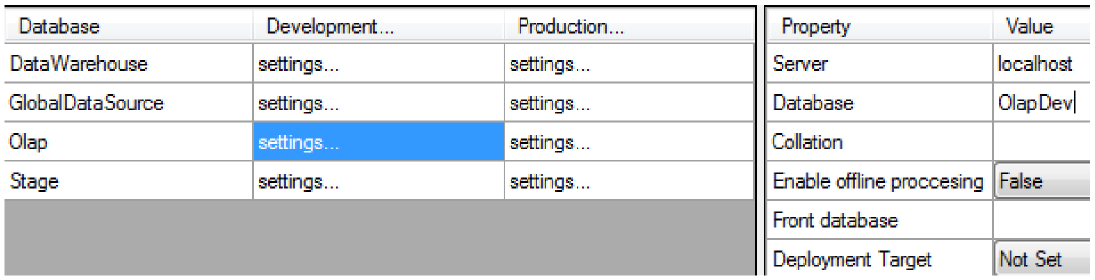
Configuring the OLAP Database

1. Next you will need to configure another Global Database for the OLAP cubes. Click "Settings..." on the OLAP row in the Development column to display the Settings pane to the right..

2. Enter the following configuration:

-Server: This will be the server address of the development server. In our example, this is**localhost**.

-Catalog:This will be the name associated with the OLAP database in the development environment. In our example, we use **OlapDev.**



Database	Development...	Production...
DataWarehouse	settings...	settings...
GlobalDataSource	settings...	settings...
Olap	settings...	settings...
Stage	settings...	settings...

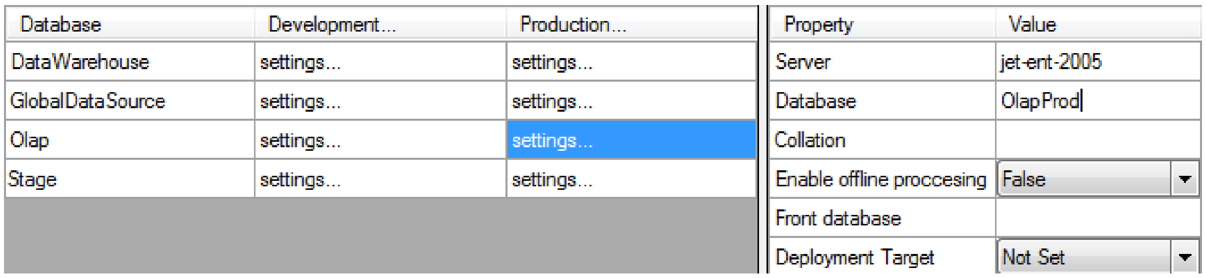
Property	Value
Server	localhost
Database	OlapDev
Collation	
Enable offline proccesing	False
Front database	
Deployment Target	Not Set

3. Next, Click "Settings..." on the OLAP row of the **Production** column.

4. Enter the following configuration:

-Server: This will be the name of the server for the Production Environment. In our example, the server name is **jet-ent-2005**.

-Catalog:This will be the name associated with your OLAP database in the production environment. In our example, we use**OlapProd**.

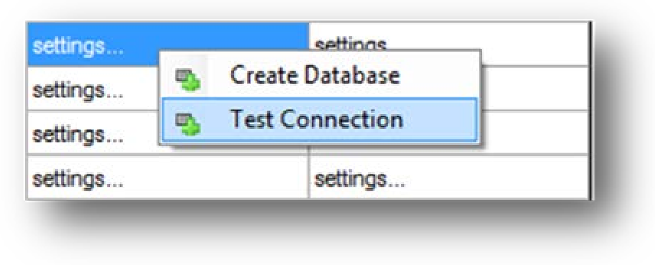


Database	Development...	Production...
DataWarehouse	settings...	settings...
GlobalDataSource	settings...	settings...
Olap	settings...	settings...
Stage	settings...	settings...

Property	Value
Server	jet-ent-2005
Database	OlapProd
Collation	
Enable offline proccesing	False
Front database	
Deployment Target	Not Set

Creating the Global Databases

The final step in the configuration process is to test and create the global databases on SQL Server. This can be done from inside the Environmental Properties window. This needs to be done for both the development and production environments. Right-click on "Settings..." and select Test Connection. If you get an error message, it generally means that the database has not been created yet. Right-click on "Settings...", and select Create Database. Then retest the connection.



Perform this check on all Global Databases for both the Development and Production environments. Once this has been completed and all "Test Connection" responses return "Connection OK", click the OK button to close the Environment Properties window, and save all changes.

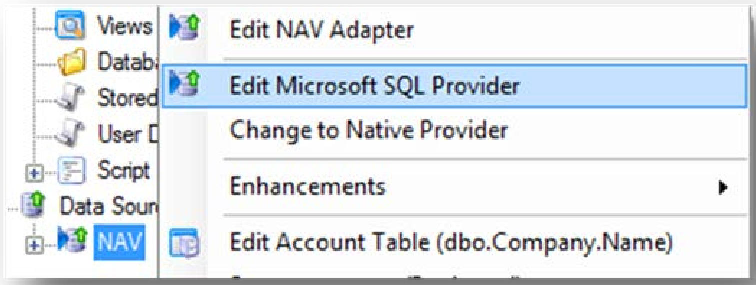
Note: The data source does not have the option to Create Database. This database represents the data source that TX2014 is extracting from and will already exist in your infrastructure. An example of this will be your NAV, GP, or AX database.

Configure Project Connections

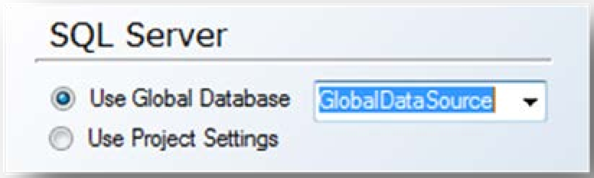
The environments have now been set up, and the global databases have been configured. The next step is to configure the connections in the project to utilize these Global Databases.

1. Open your project, and navigate to Data Sources at the bottom of the Data Tab. Right-click the adapter and select Edit Microsoft SQL Provider.

Note: This will vary depending on your data source type.


Views
Datab
Stored
User D
Script
Data Sour
NAV
Edit NAV Adapter
Edit Microsoft SQL Provider
Change to Native Provider
Enhancements
Edit Account Table (dbo.Company.Name)

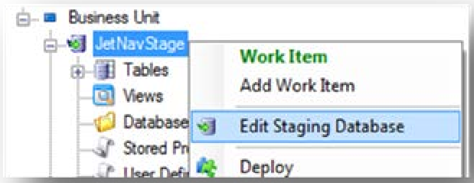
2. Select **Use Global Database** for the data source, choose the Global Database that represents your data source, and click OK. There will generally be only one Global Database displayed in the drop-down list.


SQL Server

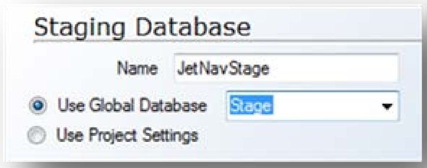
Use Global Database: GlobalDataSource

Use Project Settings

3. Navigate to your Staging Database, right-click the database, and select Edit Staging Database.



4. Select **Use Global Database** for the data source, choose the Global Database that represents your staging database, and click OK. There will generally be only one Global Database displayed in the drop-down list.

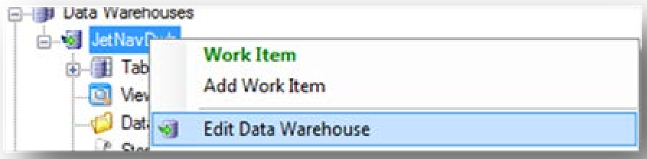

Staging Database

Name: JetNavStage

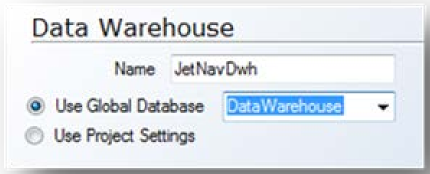
Use Global Database: Stage

Use Project Settings

5. Navigate to your **Data warehouse**, right-click the database, and select Edit Data Warehouse.



6. Select **Use Global Database** for the data source, choose the Global Database that represents your data warehouse, and click There will generally be only one Global Database displayed in the drop-down list.

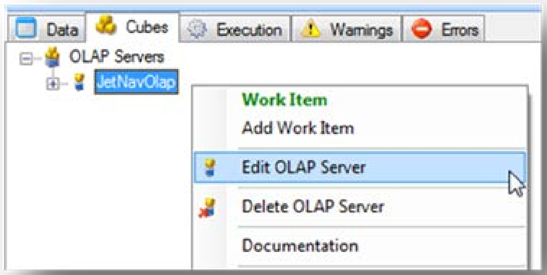

Data Warehouse

Name JetNavDwh

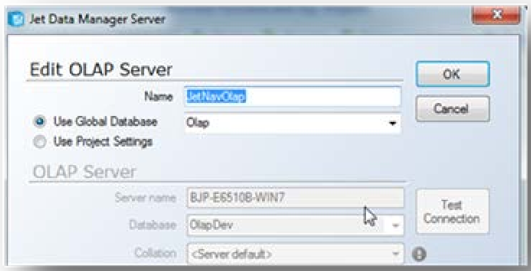
(●) Use Global Database DataWarehouse

( ) Use Project Settings

7. Navigate to the OLAP Database on the Cubes tab, right-click the OLAP Database, and select Edit OLAP Server.



8. Select **Use Global Database** for the data source, choose the Global Database that represents your OLAP database, and click OK. There will generally be only one Global Database displayed in the drop-down list.


Jet Data Manager Server

Edit OLAP Server

Name: JetNavOlap

(● Use Global Database) Olap

Use Project Settings

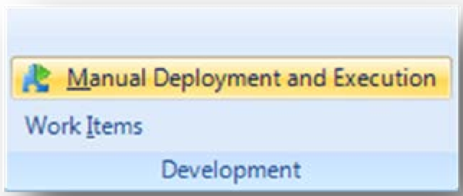
OLAP Server

Server name: BJP-E6510B-WIN7

Database: OlapDev

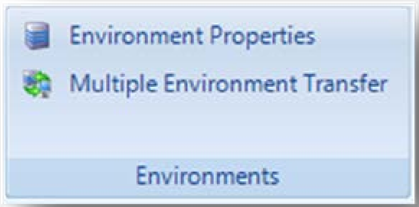
Collation: <Server default>

9. The final step is to deploy and execute the project to ensure your project is properly configured and ready for transfer. From the **Project** tab, click **Manual Deployment and Execution**, and then click **Start.**

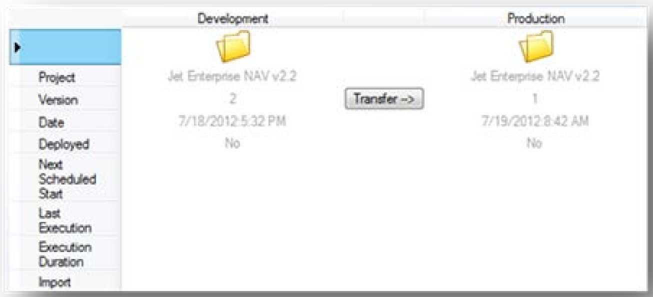


Transfer the Project from Development to Production

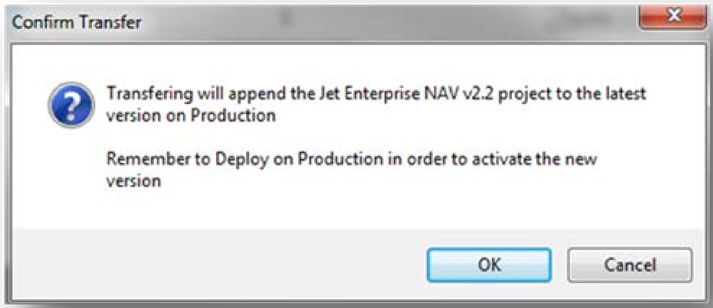
1. Log in to the **Development Environment** server, and open TX2014.
2. On the **Tools** tab, click **Multiple Environment Transfer**.



3. Click **Transfer** to migrate the project from the development server to the production server.



A dialog will appear asking you to confirm the transfer. Click **OK.**

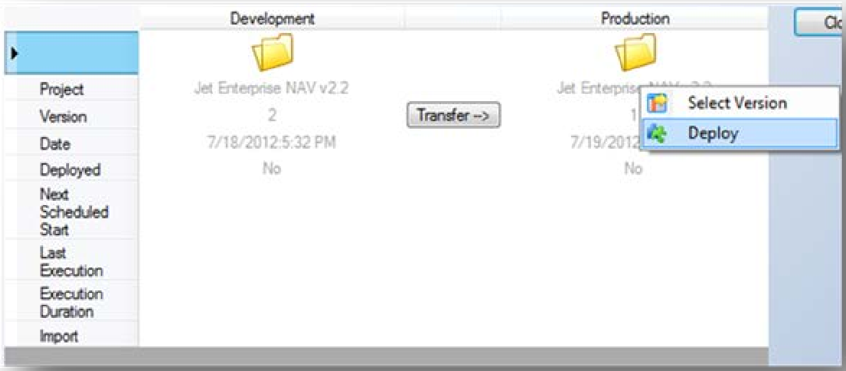


Confirm Transfer

Transfering will append the Jet Enterprise NAV v2.2 project to the latest version on Production

Remember to Deploy on Production in order to activate the new version

4. Deploy the project on the production environment. Right click the Production Environment folder and select **Deploy.**Wait for the deployment process to complete before moving on to the next step. This is physically deploying all of the objects on the Production Server. You will know the process is finished when the Deployed line in the Production column changes to Yes. Click Close to close the window.



Execution Packages

Execution packages will automatically update the staging database, data warehouse, and OLAP cubes on a scheduled basis. Since projects deployed from the Development Environment will replace packages in the Production Environment, it is recommended that the desired execution packages be set up in the Development Environment. This way, they are seamlessly transferred to the Production Environment with the package transfer. It may not be desirable to have automatic execution enabled in the development environment. This can be disabled by ensuring that the "TX2014 Server Scheduler" service is disabled on the machine hosting the development environment. For more information regarding the configuration of Execution Packages, see Execution Packages.

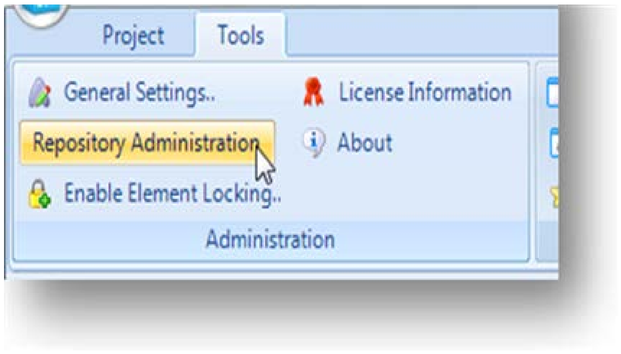
Team Development Environment

TX2014offers functionality that supports multiple users working on a project concurrently. Without this functionality, only one user at a time can be in a project. In addition, features have been added to allow more detailed tracking of changes through the use of Version Notes. The use of Work Items allows other members of the team to view any user's current work. **Team Development is an add-on feature that is available for purchase.**

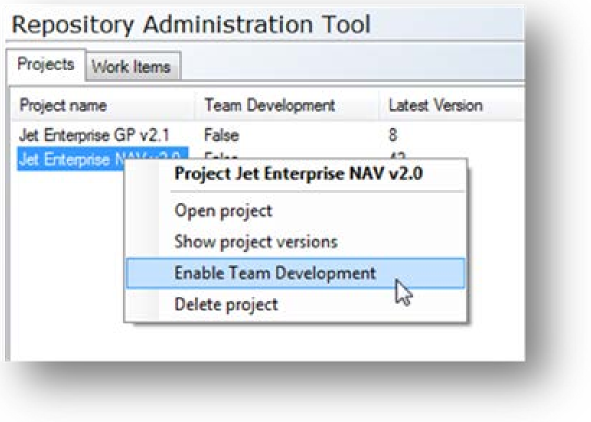
Enabling the Team Development Environment

Team Development must be turned on to be used for a particular project.

1. From the **Tools** menu, select **Repository Administration**



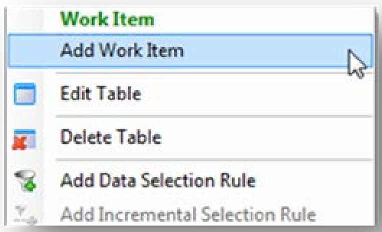
2. From the **Projects** tab, right-click the desired project, and select **Enable Team Development.**Close the window. Multiple users can now access the project concurrently.



Adding Work Items

Work Items allow the team to know what its users are working on. This will give a visual indication as to what areas of the project are currently under development. Work items are meant to be created immediately prior to starting work on a set of objects and can be either manually deleted or removed during deployment of the object.

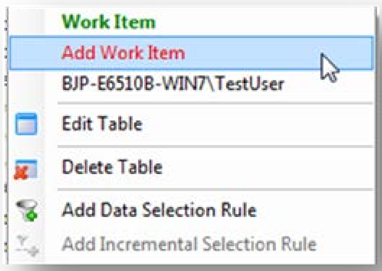
1. To add a Work Item, right-click the object to add the Work Item for, and select **Add Work Item**.



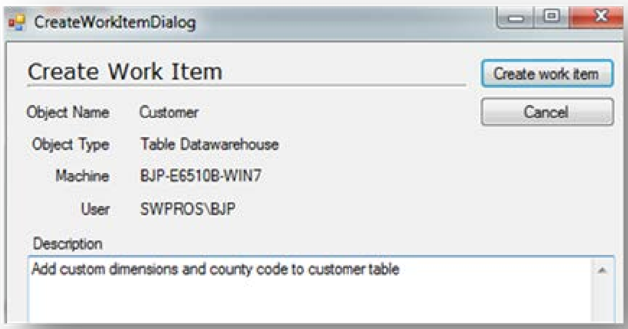
Work Items can be added to the following objects:

- Project
- Data Warehouse
- Data Warehouse table
- Staging Database
- Staging Database table
- Data Source Table
- OLAP Database
- Cube

If the object that the user is adding the Work Item to already has a Work Item created by another user, it will display **Add Work Item** in red and identify the other user below. This allows users to know if other users are modifying the same object in order to facilitate collaboration.



2. Type in a description for the Work Item.


CreateWorkItemDialog

Create Work Item

Object Name	Customer
Object Type	Table Datawarehouse
Machine	BJP-E6510B-WIN7
User	SWPROS\BJP

Description

Add custom dimensions and county code to customer table

3. Click **Create Work Item.**

How to View Existing Work Items

Viewing existing Work Items will give the user the following information:

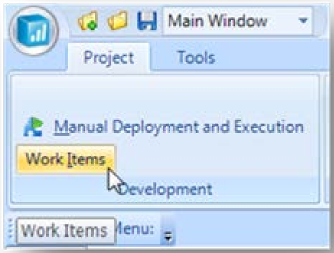
-Work Items that they have created

-Work Items that other users have created

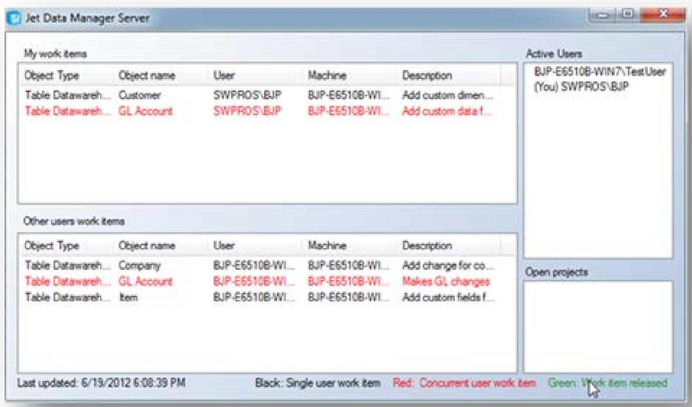
-Which users are currently in the project

-If there is any risk of overlap regarding changes made to objects based on the existing Work Items.

1. To view existing Work Items, go to the **Project** menu, and select **Work Items.**



The top pane will show **Work Items** that have been created by the current user. The bottom pane will show **Work Items** that have been created by other users.



Object Type	Object name	User	Machine	Description
Table Datawareh...	Customer	SWPROS\BJP	BJP-E6510B-WI...	Add custom dimen...
Table Datawareh...	GL Account	SWPROS\BJP	BJP-E6510B-WI...	Add custom data f...

Object Type	Object name	User	Machine	Description
Table Datawareh...	Company	BJP-E6510B-WI...	BJP-E6510B-WI...	Add change for co...
Table Datawareh...	GL Account	BJP-E6510B-WI...	BJP-E6510B-WI...	Makes GL changes
Table Datawareh...	Item	BJP-E6510B-WI...	BJP-E6510B-WI...	Add custom fields f...

BJP-E6510B-WIN7\TestUser
(You) SWPROS\BJP

Last updated: 6/19/2012 6:08:39 PM   Black: Single user work item   Red: Concurrent user work item   Green: Work item released

The information displayed is described below:

Object Type	The type of object the **Work Item** is associated with (project, database, table, or cube)
Object Name	The name of the project, database, table, or cube associated with the **Work Item**
User	The user name of the person who created the **Work Item**
Machine	The machine that the **Work Item** was created on
Description	Descriptive text defined by the user for the **Work Item**

In the upper right-hand corner in the **Active Users** pane, all users currently accessing the project are shown.

To facilitate easy identification of possible conflicts, **Work Items** that share the same Object Name for different users are highlighted red. This will allow the users to know that some collaboration will be needed regarding which user is accessing the data. While Team Development allows multiple users to modify the project concurrently, these users should not modify the same object at the same time.

For more information, see Concurrent Users Modifying the Same Object.

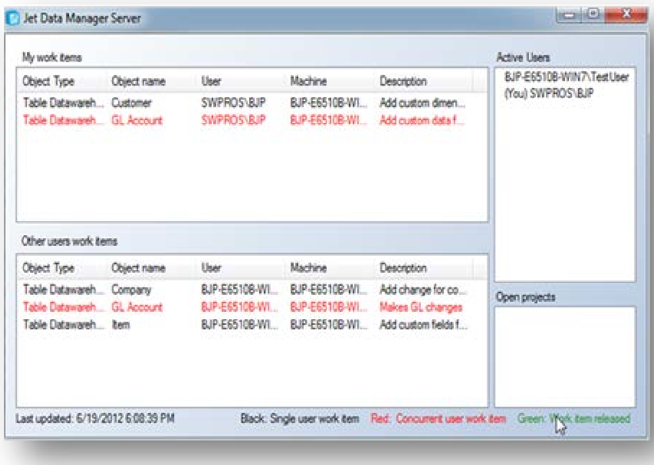
All **Work Items** that are black represent a **Work Item** that is only flagged by a single user.

Work Items that have been completed by another user will be shown as green to other users once the user has saved or deployed, and marked the **Work Item** as completed.

How to Edit a Work Item

Work Items can only be edited by the user that created the **Work Item**.

1. From the **Project** menu, click **Work Item.**


Jet Data Manager Server

My work items

Object Type	Object name	User	Machine	Description
Table Datawareh...	Customer	SWPROS\BJP	BJP-E6510B-WI...	Add custom dimen...
Table Datawareh...	GL Account	SWPROS\BJP	BJP-E6510B-WI...	Add custom data f...

Other users work items

Object Type	Object name	User	Machine	Description
Table Datawareh...	Company	BJP-E6510B-WI...	BJP-E6510B-WI...	Add change for co...
Table Datawareh...	GL Account	BJP-E6510B-WI...	BJP-E6510B-WI...	Makes GL changes
Table Datawareh...	Item	BJP-E6510B-WI...	BJP-E6510B-WI...	Add custom fields f...

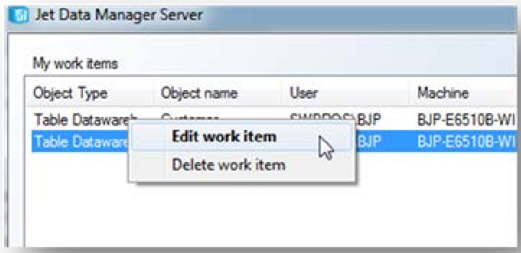
Active Users

BJP-E6510B-WIN7\TestUser
(You) SWPROS\BJP

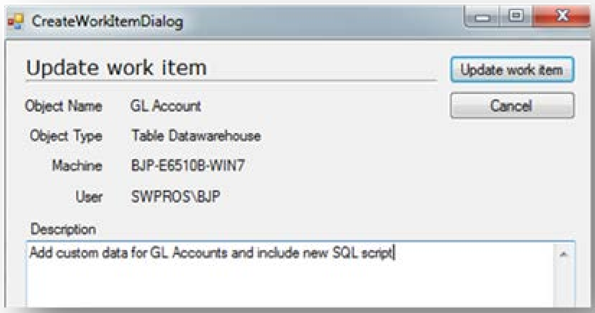
Open projects

Last updated: 6/19/2012 6:08:39 PM   Black: Single user work item   Red: Concurrent user work item   Green: Work item released

2. In the top pane of the **Work Items** screen, right-click the **Work Item** to edit, and select **Edit Work Item.**



3. Update the **Work Item** description, and click **Update Work Item** to save the changes.


CreateWorkItemDialog

Update work item

Object Name	GL Account
Object Type	Table Datawarehouse
Machine	BJP-E6510B-WIN7
User	SWPROS\BJP

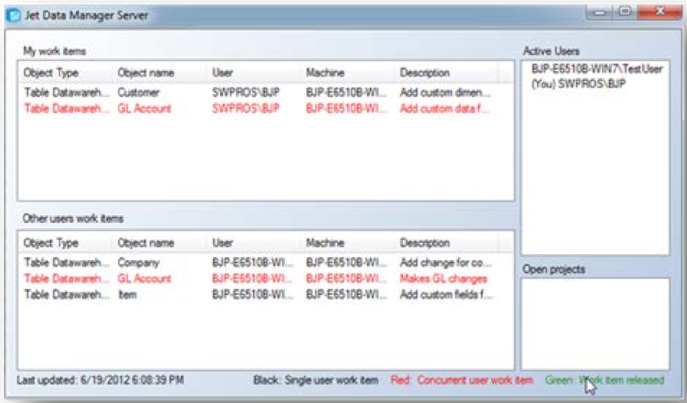
Description

Add custom data for GL Accounts and include new SQL script

How to Delete Work Items

Work Items can only be edited by the user that created the **Work Item**.

1. From the **Project** menu, click **Work Items.**



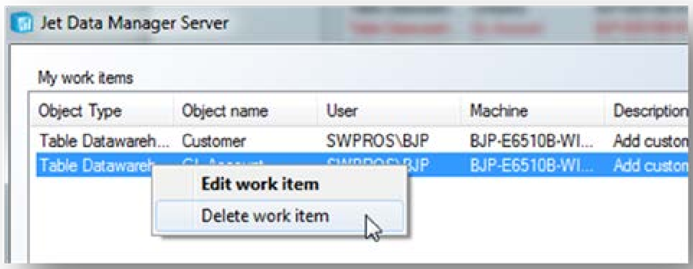
Object Type	Object name	User	Machine	Description
Table Datawareh...	Customer	SWPROS\BJP	BJP-E6510B-WI...	Add custom dimen...
Table Datawareh...	GL Account	SWPROS\BJP	BJP-E6510B-WI...	Add custom data f...

BJP-E6510B-WIN7\TestUser
(You) SWPROS\BJP

Object Type	Object name	User	Machine	Description
Table Datawareh...	Company	BJP-E6510B-WI...	BJP-E6510B-WI...	Add change for co...
Table Datawareh...	GL Account	BJP-E6510B-WI...	BJP-E6510B-WI...	Makes GL changes
Table Datawareh...	Item	BJP-E6510B-WI...	BJP-E6510B-WI...	Add custom fields f...

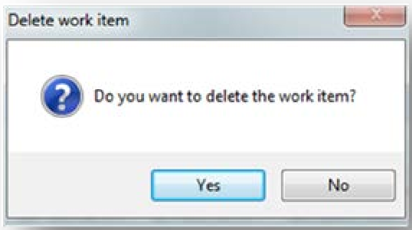
Last updated: 6/19/2012 6:08:39 PM Black: Single user work item Red: Concurrent user work item Green: Work item released

2. In the top pane of the **Work Items** screen, right-click the **Work Item** to edit, and select **Delete Work Item.**



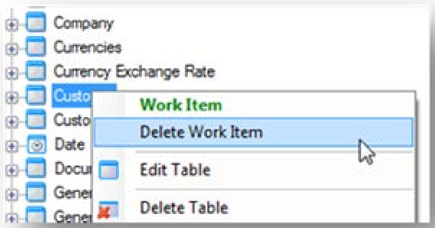
Object Type	Object name	User	Machine	Description
Table Datawareh...	Customer	SWPROS\BJP	BJP-E6510B-WI...	Add custon
Table Datawareh	GL Account	SWPROS\BJP	BJP-E6510B-WI...	Add custon

3. Confirm the **Work Item** deletion by clicking **Yes.**



Do you want to delete the work item?

Alternatively, **Work Items** can be deleted by right-clicking on the object from the tree itself and selecting **Delete Work Item.**

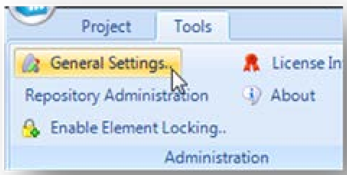


Work Items can also be removed when adding Version Notes, which is discussed below.

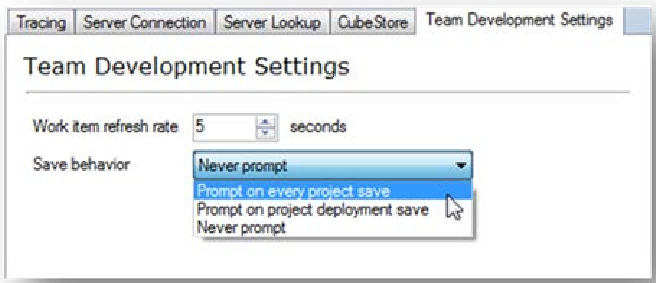
How to Add Version Notes

Version notes allow users to type comments about changes that they have made to a project. These comments can then be viewed in the future to reference changes that were made.

1. To ensure Version Notes are enabled, go to the **Tools** menu, and click **General Settings.**



2. Click the **Team Development Settings** tab at the bottom, and set the **Save Behavior** to either **Prompt on Every Project Save** or **Prompt on Project Deployment Save.**



Team Development Settings

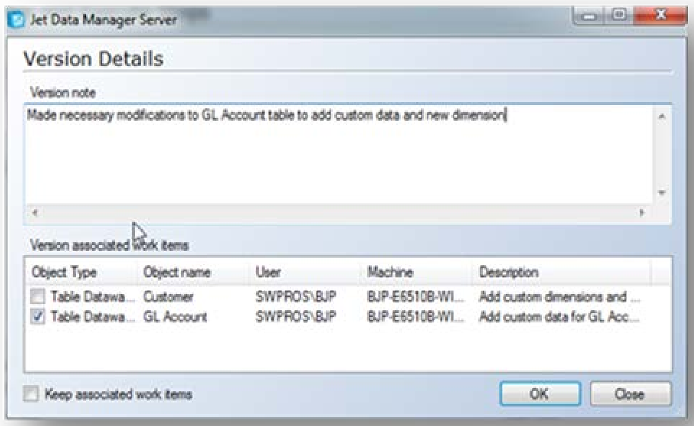
Work item refresh rate 5 seconds

Save behavior Never prompt

Prompt on every project save
Prompt on project deployment save
Never prompt

If **Prompt on Every Project Save** is selected, it will prompt the user to enter a Version Note when an object is deployed or when the **Save** button is pressed. If **Prompt on Project Deployment Save** is selected, it will only prompt the user to enter a Version Note when an object is deployed.

When deploying an object in the project, the Version Note window will appear after the Start button is pressed.



Made necessary modifications to GL Account table to add custom data and new dimension

Object Type	Object name	User	Machine	Description
☐ Table Datawa...	Customer	SWPROS\BJP	BJP-E6510B-WI...	Add custom dimensions and ...
☑ Table Datawa...	GL Account	SWPROS\BJP	BJP-E6510B-WI...	Add custom data for GL Acc...

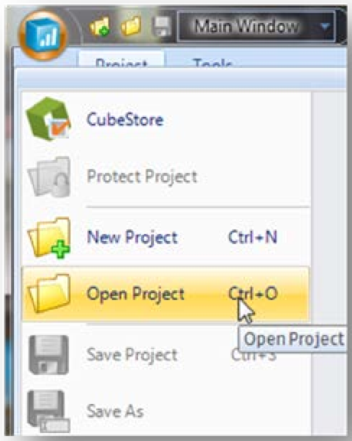
The user can type in any details they wish to include in the Version Note. If relevant, they can also select one of their existing **Work Items** to associate with the Version Note. Selecting a **Work Item** will also remove the **Work Item** from the list of existing **Work Items**. If the user wishes to associate a **Work Item** with the Version Note, but not remove the Work Item, the **Keep Associated Work Items** box can be checked. If the user does not wish to save a Version Note for this deployment, they can click **Close.**This will close the Version Note window without saving a Version Note.

How to View Version Notes

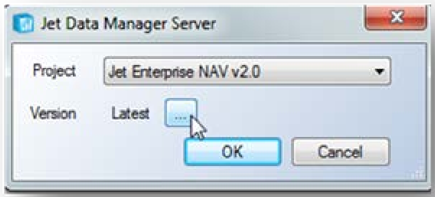
Viewing Version Notes can be useful when a user would like to look back and see when particular changes were made.

Version Notes are viewed when the project is opened.

1. In the upper left-hand corner, click the main Cube Icon, and select Open Project.



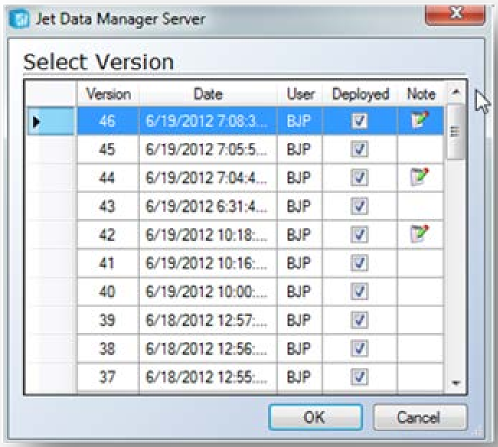
2. To view the versions, click the ellipsis (…) button to the right of **Version.**


Jet Data Manager Server

Project: Jet Enterprise NAV v2.0

Version: Latest

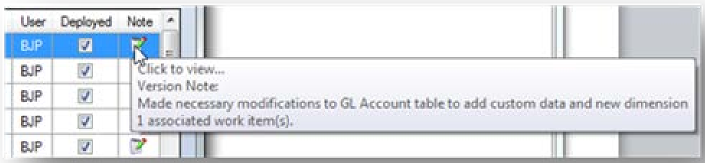
3. All saved versions of the project will be shown. Versions with an associated Version Note will have a notepad icon in the **Note** column.


Jet Data Manager Server

Select Version

Version	Date	User	Deployed	Note
46	6/19/2012 7:08:3...	BJP	✓	📝
45	6/19/2012 7:05:5...	BJP	✓	
44	6/19/2012 7:04:4...	BJP	✓	📝
43	6/19/2012 6:31:4...	BJP	✓	
42	6/19/2012 10:18:...	BJP	✓	📝
41	6/19/2012 10:16:...	BJP	✓	
40	6/19/2012 10:00:...	BJP	✓	
39	6/18/2012 12:57:...	BJP	✓	
38	6/18/2012 12:56:...	BJP	✓	
37	6/18/2012 12:55:...	BJP	✓	

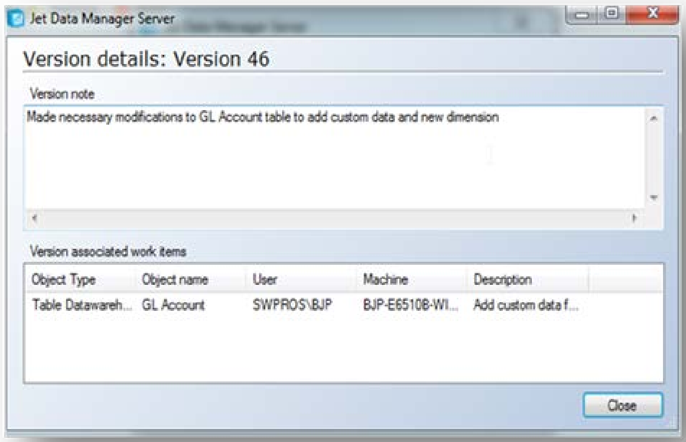
Hovering over the notepad icon will give a quick description of the Version Note.



User	Deployed	Note
BJP	✓	
BJP	✓	
BJP	✓	
BJP	✓	
BJP	✓	

Click to view...
Version Note:
Made necessary modifications to GL Account table to add custom data and new dimension
1 associated work item(s).

Clicking on the notepad icon will display the full version note as well as any **Work Items** associated with the Version Note.



Jet Data Manager Server

Version details: Version 46

Version note

Made necessary modifications to GL Account table to add custom data and new dimension

Version associated work items

Object Type	Object name	User	Machine	Description
Table Datawareh...	GL Account	SWPROS\BJP	BJP-E6510B-WI...	Add custom data f...

Concurrent Users Modifying the Same Object

The Team Development feature,Work Items, allows multiple users to collaborate on a project by providing open viewing access to each user's intended work. For optimal results, an object should not modified by several users concurrently. Though rare, if this scenario does occur, the expected outcome is as follows:

- If multiple users add or modify different sub-objects (such as fields) within the same object (such as a table), then TX2014will save these changes and display them the next time that TX2014is opened. For example, if a user adds two fields to a table, and another user adds two different fields in the same table, all four fields will be visible to users the next time they close and re-open TX2014.
- If multiple users modify the same sub-object (such as renaming a field name) at the same time, the user who deploys and executes the changes last will take precedence, and the project will reflect these changes.

Dimensional Modeling

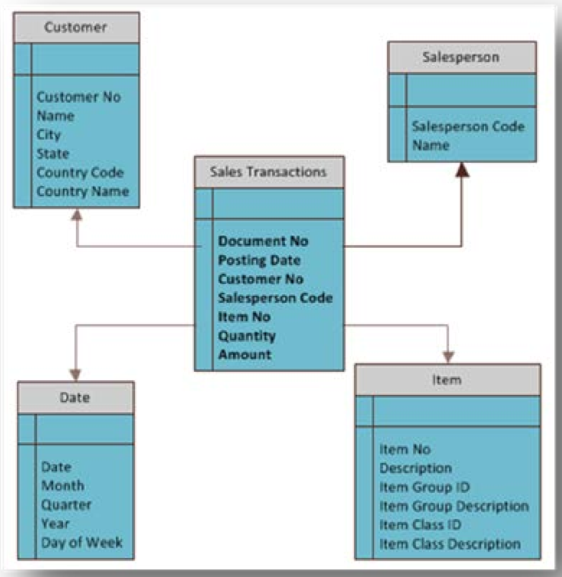
Dimensional modeling is the technique and methodology used in data warehouse design. A dimensional model typically consists of a fact table and a number of dimension tables taking the form of a star schema or a snowflake schema.

The fact table defines what you are going to analyze and contains numerical values. Fact tables are more commonly known as transaction tables and generally may contain a large number of records.

Dimension tables define how to analyze the information in the fact table. These tables are much smaller as they do not contain transactional information. Instead, these tables contain summary or attribute information about things such as: customers, items, general ledger accounts, etc.

Star Schema

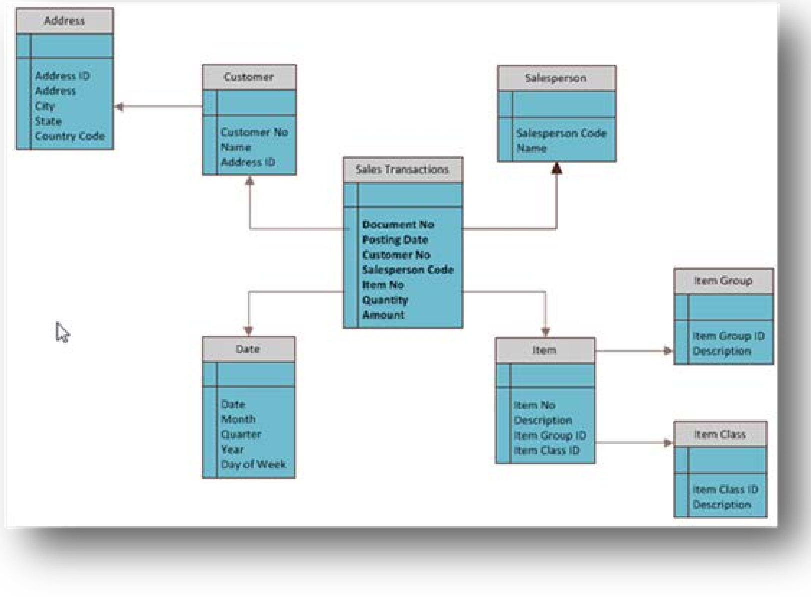
A star schema represents a fact table that is directly joined to all dimension tables. The schema resembles a star with the fact table at the center and the dimension tables as the points of the star. In a star schema, all of the descriptive information for a particular dimension is contained in a single table. This is the methodology used in most of the standard projects. Below is an example of a standard star schema.



Snowflake Schema

A snowflake schema is based on a variation of the star schema. The snowflake schema is more normalized with a fact table at the center and dimension tables surrounding this fact table. One of the main differences between the snowflake schema and star schema is that all the descriptive information in a snowflake schema can be stored in multiple tables, whereas with a star schema, all of the descriptive information is in a single table. Notice in the diagram below, the Address, Item Class, and

Item Group tables, whereas in the star schema all of these details were stored in the Customer and Item tables.



About Dimensions

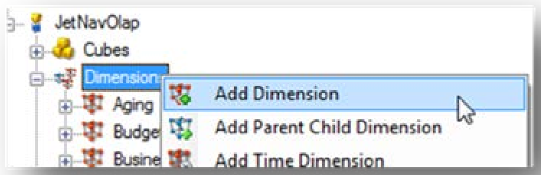
Dimensions define how a user looks at data in a cube. Dimensions are created independently of a particular cube and can be used within several cubes at the same time. The same dimension can also be utilized several times within the same cube, which is referred to as a role-playing dimension. A common example of this would be the Date dimension, which can represent both the Document Date and Posting Date in a cube, thus having a single dimension play two roles.

Regular Dimensions

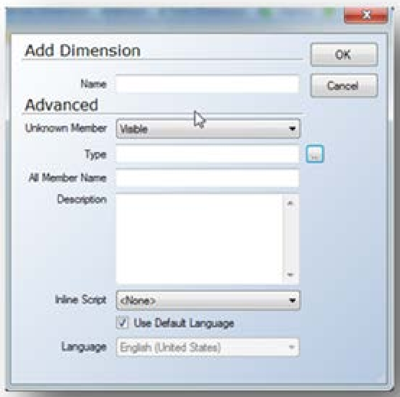
Regular dimensions are based on a snowflake or a star schema, and are used to create balanced hierarchies or ragged hierarchies.

Creating Regular Dimensions

1. On the Cubes tab, expand the OLAP server to be modified, right-click **Dimensions**, and then select **Add Dimension**.



2. In the **Name** field, type a name for the dimension.



3. To apply an Unknown Member to dimension keys in the fact table with no matching dimension members, go to**Unknown Member,**and select**Visible**. This will allow dimension values that exist in the fact table but not the dimension table to be combined together and displayed to the user as an Unknown value. This is the default and recommended value in TX2014. An example would be a Salesperson Code that exists in the sales transactions fact table but does not exist in the Sales-person dimension table.

4. In the **Type** field, click the ellipsis button **(...)**, and select the type of dimension you want to create. This can also be left blank for Regular dimension types, which is generally most common.

Regular	Default for dimensions that are not set to a specified type
Time	Used for dimensions whose attributes represent time periods
Geography	Used for dimensions whose attributes represent geographical information
Organization	Used for dimensions whose attributes represent organizational information
BillOfMaterials	Used for dimensions whose attributes represent inventory and manufacturing information
Accounts	Used for dimensions whose attributes represent information used for financial reporting
Customers	Used for dimensions whose attributes represent information about customers
Products	Used for dimensions whose attributes represent information about products
Scenario	Used for dimensions whose attributes represent information about plans and strategies
Quantitative	Used for dimensions whose attributes represent quantitative information
Utility	Used for dimensions whose attributes represent utility information
Currency	Used for dimensions whose attributes represent currency information
Rates	Used for dimensions whose attributes represent currency rate information
Channel	Used for dimensions whose attributes represent channel information
Promotion	Used for dimensions whose attributes represent marketing promotion information

5. In the **All Member Name** field, type a name for the **All** Member. This is left blank by default, and will have Analysis Services create the **All Member Name** automatically. The **All Member** is the dimension value which represents all members of the dimension. An example would be a dimension value of “All Customers”, which would represent every customer in the Customer dimension.

6. In the **Description** field, describe the dimension. This field is optional. Click **OK**.

When you have added a dimension, you also have to add at least one dimension level. The **AddDimension Level** dialog is displayed after you click OK. For more information, see To Add Dimension Levels.

Once you have created a dimension, the dimension can be used in several cubes at the same time. These are known as role playing dimensions.

Parent-Child Dimensions

Parent-child dimensions are used to create un-balanced hierarchies where the branches descend to different levels, and where the parent and the child exist in the same table. Typically, parent-child hierarchies are used for creating organizational hierarchies.

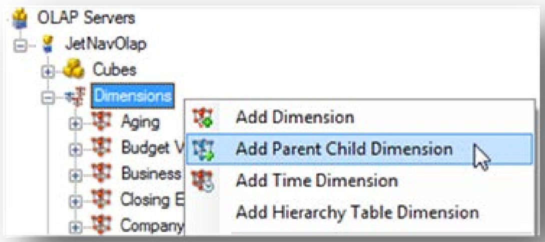
Creating Parent-Child Dimensions

A parent-child dimension is a hierarchy that is defined by a parent column and a child column in the same table. A member of the hierarchy can appear more than once in the hierarchy.

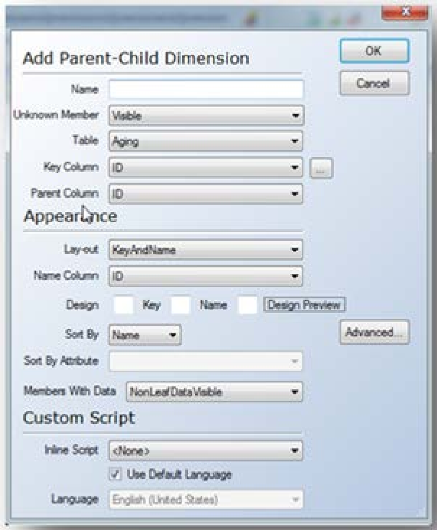
Follow the steps below to create a Parent-Child Dimension:

1. Expand OLAP servers, and then expand the OLAP server you wish to modify.

2. Right-click **Dimensions**, and then select **Add Parent-Child Dimension**.



You will then see the following window:



Name	
Unknown Member	Visible
Table	Aging
Key Column	ID
Parent Column	ID
Lay-out	KeyAndName
Name Column	ID
Design	Key Name
Sort By	Name
Sort By Attribute	
Members With Data	NonLeafDataVisible
Inline Script	<None>
	✓ Use Default Language
Language	English (United States)

3. In the **Name** field, type a name for the dimension.

4. To apply an Unknown Member to dimension keys in the fact table with no matching dimension members, go to**Unknown Member**,and select **Visible**. This will allow dimension values that exist in the fact table but not the dimension table to be combined together and displayed to the user as an Unknown value. An example would be a customer number that exists in the Sales Transactions table in the data warehouse but not in the Customer table. This would result in all transactions for this customer being associated with the **Unknown** value. This is the default and recommended value in TX2014.

5. In the **Table** list, select the main table of the dimension.

6. In the **Key Column** list, select the key column of the child table. This column identifies each member of the dimension.

7. In the **Parent Column** list, select the key column of the parent field. This column identifies the parent of each member.

8. In the **Lay-out** list, select how you want the dimension level displayed to the end user. The following options are available:

Key	Displays only key column values
Name	Displays only name column values
KeyandName	Displays the key column first and then name column values
NameandKey	Displays the name column values first and then the key column values

9. In the **Name column** list, select the column that provides a meaningful value to the user. This field is only available if you have selected the **Name**,**KeyAndName**,or**NameAndKey** layout.

10. In the **Design** fields, specify which separator to use in the front-end application to separate Key and Name. This field is only enabled if you have selected KeyAndName or NameAndKey. The order in which the Key and Name text fields appear depends on your selection in the Layout list. To preview the design of the layout, move the pointer over the **Design Preview** button.

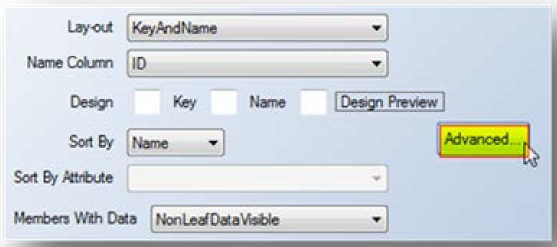
11. In the **Sort By**list, select whether you want the values sorted by Key or Name.

12. In the **Sort by Attribute** list, select the specific attribute key or name that you want the values sorted by. This list is only available when you are working with key levels, and Sort By is set to **AttributeKey** or **AttributeName**.

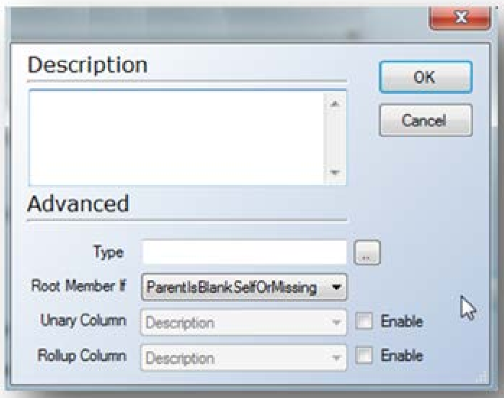
Note: When you create the parent-child dimension based on a consolidation table, you will typically use a **SortBy Attribute**. You therefore need to create a Sort order dimension level where the key column is **SortOrder**. Then, you must enable Unary column and Roll up column on the dimension. You can then set the parent-child dimension to **Sort By Attribute**.

To Define Advanced Parent-Child Dimension Settings

To access the Advanced Settings for Parent-Child dimensions, click the **Advanced...**button from within the Parent-Child dimension setup window.


Lay-out: KeyAndName
Name Column: ID
Design: Key Name Design Preview
Sort By: Name
Sort By Attribute:
Members With Data: NonLeafDataVisible

The following window will then be shown:


Description

Advanced

Type:

Root Member If: ParentIsBlankSelfOrMissing

Unary Column: Description [ ] Enable

Rollup Column: Description [ ] Enable

1. In the Type field, specify the type of parent-child dimension you want to create.

Regular	Default for dimensions that are not set to a specified type
Time	Used for dimensions whose attributes represent time periods
Geography	Used for dimensions whose attributes represent geographical information
Organization	Used for dimensions whose attributes represent inventory and manufacturing information
BillOfMaterials	Used for dimensions whose attributes represent inventory and manufacturing information
Accounts	Used for dimensions whose attributes represents information used for financial reporting
Products	Used for dimensions whose attributes represent information about products
Scenario	Used for dimensions whose attributes represent information about plans and strategies
Quantitative	Used for dimensions whose attributes represent quantitative information
Utility	Used for dimensions whose attributes represent a variety of information
Currency	Used for dimensions whose attributes represent currency information
Rates	Used for dimensions whose attributes represent currency rate information
Channel	Used for dimensions whose attributes represent channel information
Promotion	Used for dimensions whose attributes represent marketing promotion information

2. In the **Root Member If** list, specify when the dimension is the root member. You have the following options:

ParentIsBlank	Hides the root if the member is a null, a zero or an empty string
ParentIsBlankSelfOrMissing	Hides the root if the member is a null, a zero, an empty string, if the parent is missing, and if the member itself is a parent
ParentIsMissing	Hides the root if the parent is missing
ParentIsSelf	Hides the root if the member itself is a parent

3. In the **Unary Column** list, select the column that contains the unary operators that are used in this dimension level. You have to select **Enabled** to select from this list.

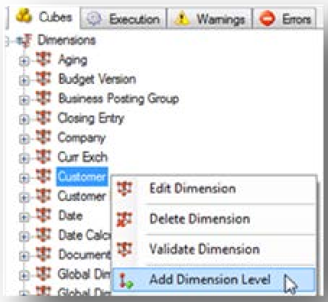
4. In the **Roll-up Columnlist**, select the column that contains the roll-up values used in this dimension level. You have to select **Enabled** to select from this list.

To Add Dimension Levels

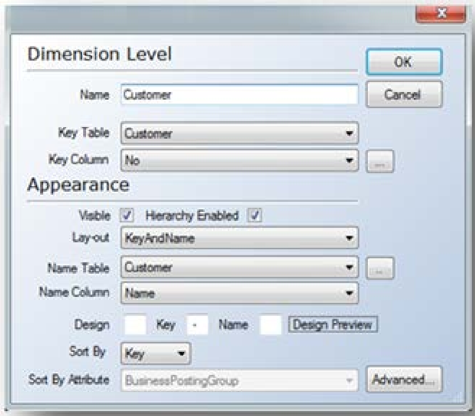
Dimension levels are used to create a dimension attribute within a cube, which enables a user to drill down or roll-up through data. A dimension must contain at least one dimension level.

1. On the **Cubes** tab, expand the OLAP server that contains the dimension you wish to modify by adding a level.

2. Expand **Dimensions**, then right-click the dimension for modifying, and select**Add Dimension Level**.



A window will open that contains the following information:


Dimension Level

Name: Customer

Key Table: Customer

Key Column: No

Appearance

Visible ✓ Hierarchy Enabled ✓

Lay-out: KeyAndName

Name Table: Customer

Name Column: Name

Design: Key - Name | Design Preview

Sort By: Key

Sort By Attribute: BusinessPostingGroup

OK | Cancel | ... | Advanced...

3. In the **Name** field, type a name for the dimension level.

4. In the **Key Table** list, select the table in the data warehouse to add the dimension from.

5. In the **Key Column**list, select the column which uniquely identifies the records in the table. If the key is a composite key, click the ellipsis button (…), to select the attributes on which the Key column is based.

6. Check the **Visible** box if you want the level to be displayed in the front-end application.

7. In the **Layout** drop-down, select the way that the dimension level should be displayed to users. The options are:

Key	Displays only key column values
Name	Displays only name column values
KeyandName	Displays the key column values first and then name column values
NameandKey	Displays the name column values first and then the key column values

8. In the **Name Table** list, select the table that provides a meaningful name to the user. To set the Name Table value to the same value as the Key Table value, click the ellipsis button.

9. In the **Name Column** list, select the column that provides a meaningful value to the user.

10. In the **Design** fields, specify which separator to use in the front-end application to separate Key-AndName. This field is only enabled if you have selected KeyAndName or NameAndKey. The order in which the Key and Name text fields appear depends on your selection in the Layout list. To review the design of the layout, move the pointer over the **Design Preview** button.

11. In the **Sort By** list, select if you want the values sorted by **Key** or **Name**. If you are adding a key level, you can also sort by **AttributeKey** and **AttributeName**.

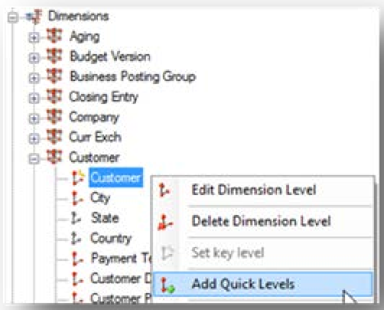
12. In the **Sort By Attribute** list, select the specific attribute key or name that you want the value sorted by. This list is only available when you are working with key levels, and Sort By is set to Attrib-uteKey or AttributeName.

13. Click **OK**. The dimension level is added to the **Dimensions** tree below the dimension it belongs to. The OLAP database must be deployed and executed for this to be finalized for the end users.

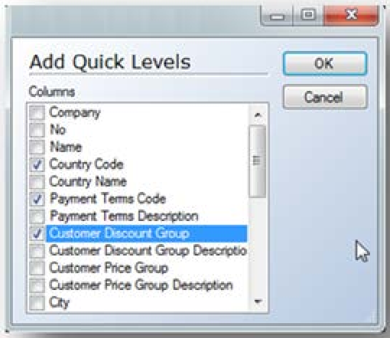
To Add Quick Levels

Quick Levels provide an easy way to add new dimension levels. It will automatically set defaults which can later be changed by editing the dimension level.

1. On the **Cubes** tab, expand the preferred OLAP server. Then expand **Dimensions**, and right-click the parent-child dimension or key level on a dimension to which you want to add a level. Select **Add Quick Levels.**



2. Select the columns you want as levels from the source table in the data warehouse, and then click **OK**.



The levels you have added are now available when you create a hierarchy. The OLAP database must be deployed and executed for this to be finalized for the end users.

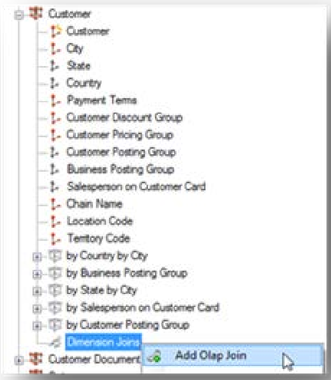
To Add Dimension Joins

Dimension joins are joins between two tables that are not directly related to the dimension's fact table. You only use dimension joins in snowflake schemas where you want more than one table in a dimension. The join is a one-to-many join.

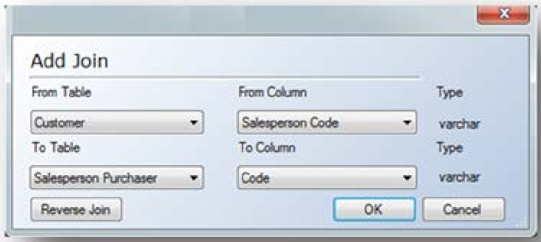
1. On the **Cubes** tab, expand the OLAP server that contains the dimension you wish to modify.

2. Expand **Dimensions**, and then expand the dimension to which you want to add a join.

3. Right-click **Dimension Joins**, and then select **Add Olap Join**.



A window will appear that contains the following information:



Add Join

From Table	From Column	Type
Customer	Salesperson Code	varchar
To Table	**To Column**	**Type**
Salesperson Purchaser	Code	varchar

4. In the **From Table**list, select the table *from* which you want to create a join.

5. In the **To Table** list, select the table *to* which you want to create a join.

6. In the **From Column** list, select the column from which you want to create a join. The column's data type is displayed next to the list. You can only make joins between fields with compatible data types.

7. In the **To Column** list, select the column to which you want to create a join. If the column's data type is not compatible with the data type of the **From Column**, the data type is displayed in red.

8. If you want to reverse the direction of the join, click the **Reverse Join** button.

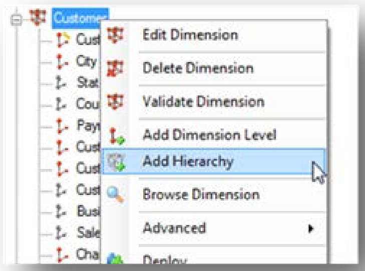
9. Click **OK**. The dimension join is displayed in the **Dimension Joins** folder in the **Dimensions** tree.

To Add Dimension Hierarchies

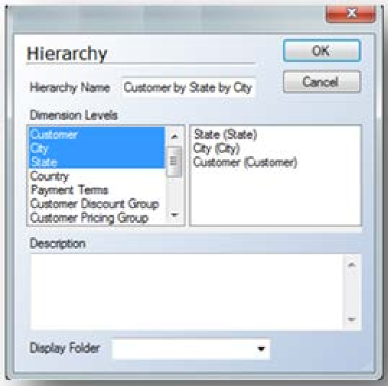
Once you have added dimension levels to a dimension, you can create a dimension hierarchy. Dimension hierarchies make it easier for users to look at commonly used dimension groupings by only having to drag one icon into a report. An example of this could be Customers by Country or Items by Item Category.

1. Expand the preferred OLAP server, and then expand **Dimensions**.

2. Right-click the dimension to which you want to add a hierarchy, and then select **Add Hierarchy**.



A window will appear that has the following information:



Hierarchy Name: Customer by State by City

Dimension Levels: Customer, City, State, Country, Payment Terms, Customer Discount Group, Customer Pricing Group

State (State)
City (City)
Customer (Customer)

3. In the **Hierarchy Name** field, enter a name for the hierarchy. The name cannot be the same as the name of a dimension level.

4. In the **Dimension Levels** pane, click the levels you want to be part of the hierarchy. The hierarchy elements are then listed in the right pane. You can drag the dimension levels in the right pane up and down to specify the order they should exist in from top to bottom

5. In the **Description** field, type a description of the hierarchy. This field is optional.

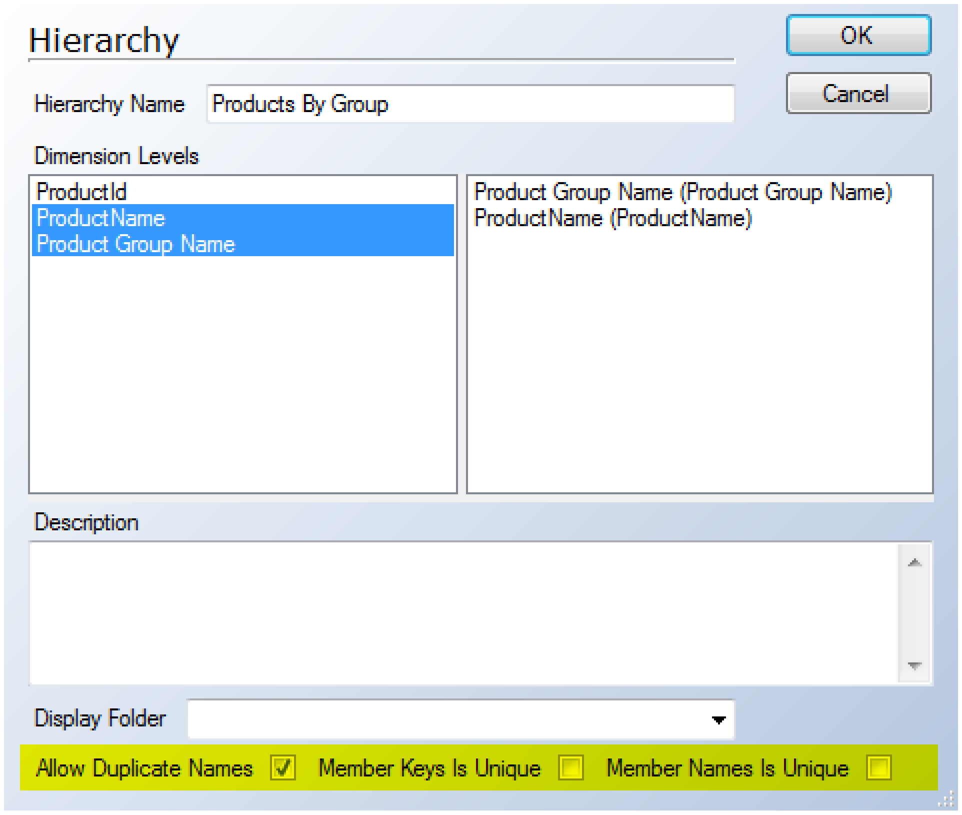
6. In the **Display Folder** list, select the folder where the hierarchy is displayed by the front-end application. This is optional.

7. Click **Ok.** The OLAP database must be deployed and executed for this to be finalized for the end users.

Note: Since the hierarchy is associated with the dimension itself, once the dimensions and cubes are deployed and executed, the hierarchy will automatically show up in all cubes in which the dimension exists.

Dimension Hierarchy Properties

On the dimension hierarchy, it is possible to change the settings for AllowDuplicateNames, MemberKeysUnique, and MemberNamesUnique.



Hierarchy Name: Products By Group

Dimension Levels:
- ProductId
- ProductName
- Product Group Name

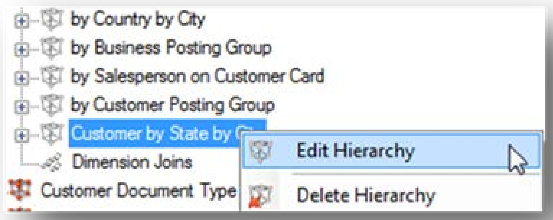
- Product Group Name (Product Group Name)
- ProductName (ProductName)

Allow Duplicate Names [✓] Member Keys Is Unique [ ] Member Names Is Unique [ ]

To Edit a Dimension Hierarchy

1. Expand **Dimensions**,and expand the preferred dimension. Then right-click the hierarchy you want to edit.

2. Select **Edit Hierarchy**.

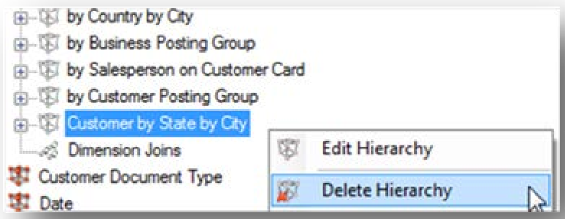


3. Make your changes. To remove a level, click the level in the **Dimensions Levels** pane, and then click **OK**. The OLAP database must be deployed and executed for this to be finalized for the end users.

To Delete a Dimension Hierarchy

1. Expand **Dimensions**and expand the preferred dimension. Then right-click the hierarchy you want to delete.

2. Select **Delete Hierarchy.**


by Country by City
by Business Posting Group
by Salesperson on Customer Card
by Customer Posting Group
Customer by State by City
Dimension Joins
Customer Document Type
Date
Edit Hierarchy
Delete Hierarchy

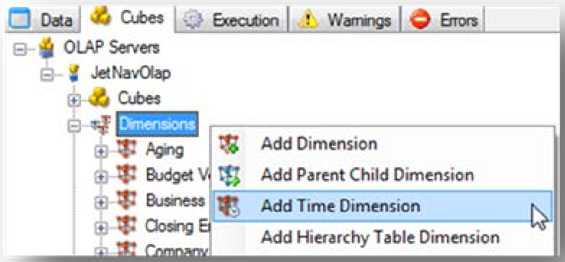
3. Select **Yes** to confirm the deletion of the hierarchy. The OLAP database must be deployed and executed for this to be finalized for the end users.

To Create a Date Dimension

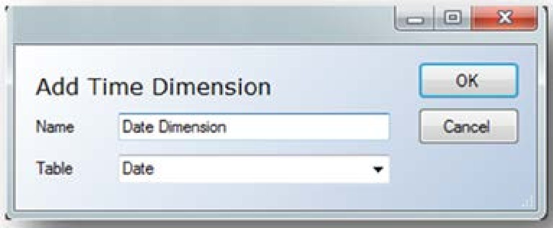
Date dimensions are based on time or date tables, so you therefore have to create a time or date table in the data warehouse before you can create a time dimension. For more information, see To Create Date Tables.

1. On the **Cubes** tab, expand the preferred OLAP server, and then right-click **Dimensions**.

2. Select **Add Time Dimension**.



A window with the following information will appear:


Add Time Dimension

Name: Date Dimension

Table: Date

3. In the **Name** field, type a name for the dimension.

4. In the **Table** list, select the table on which you want to base your time dimension, and then click **Ok**.

When you expand the date dimension, you can see that the levels which correspond to fields on the date table in the data warehouse have already been added. However, you can add more levels simply by adding Quick Levels or by adding regular levels. Hierarchies for the users can be easily added as well.

Enabling Slowly Changing Dimensions

Slowly Changing Dimensions (SCD) enable an organization to track how dimension attributes change over time. For example, it is possible that an item may be associated with a particular product group code but that it is later reclassified into a different product group. The organization wants to be able to analyze the historical sales data that occurred when the item was assigned to the original product group as well as more recent sales data that has occurred after the item was reclassified to the new product group.

Different Types of Slowly Changing Dimensions

Type I

Type I dimensions will automatically overwrite old data with updated data from the data source. An example of this would be a change in a customer name. In the data source, the name for a particular customer is changed from ABC Consulting to Acme Consulting. The next time that the data warehouse is updated the customer name will be changed from ABC Consulting to Acme Consulting and no historical record of the change is kept. All historical, current, and future transactions will be displayed under the new customer name of Acme Consulting. This is the default methodology of updating data in the data warehouse and no setup is required.

Type II

Type II dimensions will enable the tracking of dimension attributes historically by inserting additional records into the table as the values in specified fields are changed. TX2014will administer the tracking of the dimension values as well as the updating of the table. Each record for a particular value, such as an item number, can be viewed as a different version of this item. The transaction table can then be linked to this table to display which version of the item was associated with the transaction based on the transaction date.

The follow example illustrates what the Customer dimension table would resemble if the example in the Type I section was tracked using Type II functionality.

Customer No	Name	City	State	Version	From Date	To Date
123	ABC Consulting	Portland	OR	1	7/1/2009	9/18/2012
123	Acme Consulting	Portland	OR	2	9/18/2012	12/31/9999

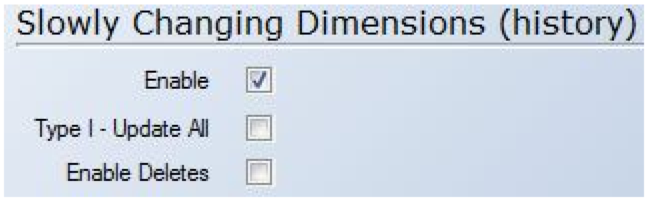
How to Implement Type II Slowly Changing Dimensions

The steps below will explain how to utilize Slowly Changing Dimensions in a TX2014project. In the example, there will be an item named "Bicycle" that has historically had an Inventory Posting Group of "Finished". Recently, however, this item has been reclassified and is now associated with the Inventory Posting Group "Resale". The organization wishes to track sales for this item under both the historical Inventory Posting Group as well as the new one.

Enable Slowly Changing Dimensions on the Dimension Table

1. Identify the table where historical changes need to be tracked. Right-click the table name, and go to Advanced -> Advanced Settings.

2. Check the box to Enable Slowly Changing Dimensions and click Ok.


Slowly Changing Dimensions (history)

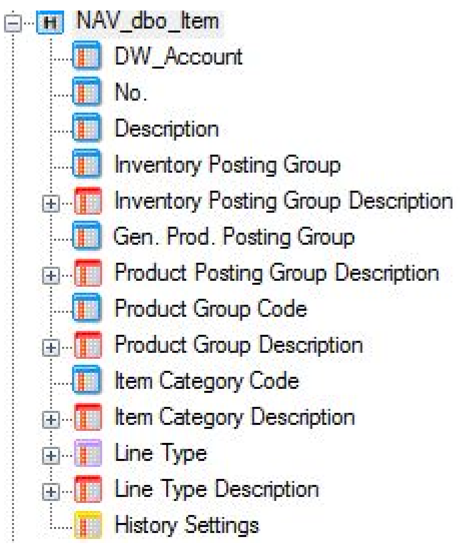
Enable ✓

Type I - Update All ☐

Enable Deletes ☐

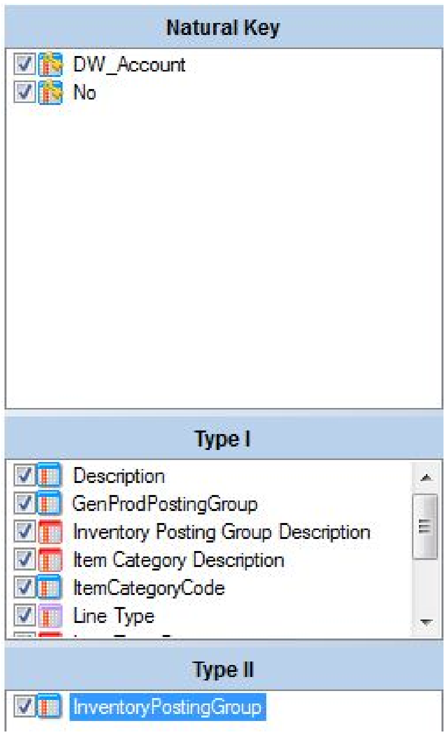
If a red "X" appears on the right-hand side of the Advanced Settings box, it indicates that the conditions required to utilize Slowly Changing Dimensions are not met. Placing the mouse over the red "X" will provide a dialog explaining which requirements need to be enabled.

The table will now display an "H" inside of the table icon denoting that historical tracking has been enabled for this table. Expanding the table will show a node named "History Settings".


```
NAV_dbo_Item
    DW_Account
    No.
    Description
    Inventory Posting Group
    Inventory Posting Group Description
    Gen. Prod. Posting Group
    Product Posting Group Description
    Product Group Code
    Product Group Description
    Item Category Code
    Item Category Description
    Line Type
    Line Type Description
    History Settings
```

Clicking on the "History Settings" node will display the Slowly Changing Dimensions configuration options on the right-hand side of the screen. This screen is broken down into three sections:

- **Natural Key:** Check boxes should be selected for all fields that represent the natural key of the table. In the current example this would be the "DW_Account" and "No." fields
- **Type I:** Check boxes should be selected for all fields which should not have history tracking enabled. If values in these fields are changed in the data source, the old values will be over-written by the new values when the table is next executed.
- **Type II:** Check boxes should be selected for all fields which should have history tracking enabled.


Natural Key

- [✓] DW_Account
- [✓] No

Type I

- [✓] Description
- [✓] GenProdPostingGroup
- [✓] Inventory Posting Group Description
- [✓] Item Category Description
- [✓] ItemCategoryCode
- [✓] Line Type

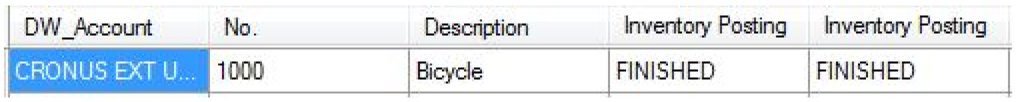
Type II

- [✓] InventoryPostingGroup

The table should now be deployed and executed so that the initial history data can be stored.

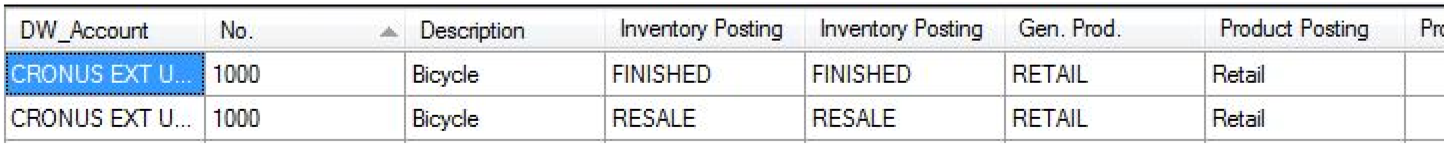
A primary key comprising at least one field must be set on the table in order for the table to deploy and execute properly. The primary key fields for a table can be set by right-clicking the desired fields within the table and selecting "Include in Primary Key". This can be done for multiple fields in the same table.

The screen-shot below illustrates what the Item table currently looks like for the item that is being used in this example. There is one record for the item and currently the Inventory Posting Group is set to "Finished".



DW_Account	No.	Description	Inventory Posting	Inventory Posting
CRONUS EXT U...	1000	Bicycle	FINISHED	FINISHED

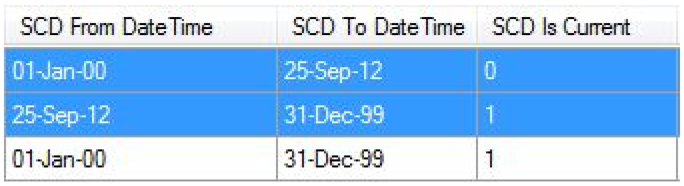
The item is now reclassified into the Inventory Posting Group Code of "Resale.". An invoice is then posted that reflect a sale of 100 of the bicycles with the new Inventory Posting Group. To illustrate how the Item table now looks in the staging database, the table is executed to reflect the changes and the results are displayed below:



DW_Account	No.	Description	Inventory Posting	Inventory Posting	Gen. Prod.	Product Posting	Pro
CRONUS EXT U...	1000	Bicycle	FINISHED	FINISHED	RETAIL	Retail	
CRONUS EXT U...	1000	Bicycle	RESALE	RESALE	RETAIL	Retail	

The DW_Account and No. fields are the same, but the Inventory Posting Groups now reflect the new value. The DW_ID, which represents a unique record number in the table, is also different as illustrated on the right in the screen-shot above.

There are a few more fields that pertain to the historical values that are useful as well. When the table is executed and notices a change in one of the Type II fields, it will automatically add in the dates for which the old value ended and the new value begins. These are the "SCD From Datetime", "SCD To Datetime", and "SCD "IsCurrent" fields.



SCD From DateTime	SCD To DateTime	SCD Is Current
01-Jan-00	25-Sep-12	0
25-Sep-12	31-Dec-99	1
01-Jan-00	31-Dec-99	1

The To and From field represent the date ranges that this version of the dimension was used in and which record is the current record.

Bringing the Surrogate Key to the Transaction Table

In a standard transaction table, the transaction itself will only be linked to the Item No. This is problematic, as the item number alone does not identify which version of the item record the transaction applies to. In order to see this detail, the surrogate key from the Item table will be brought into the transaction table. This surrogate key is based on the To and From dates in the Item table, as compared with the Posting Date of the transaction. In the screen-shot above, all transactions between January 1, 1900 and September 25, 2012 will be associated with the first version of the Bicycle item where the Inventory Posting Group is "Finished". Any transaction after September 25, 2012 will be associated with the latest version of the Bicycle item where the Inventory Posting Group is "Resale".

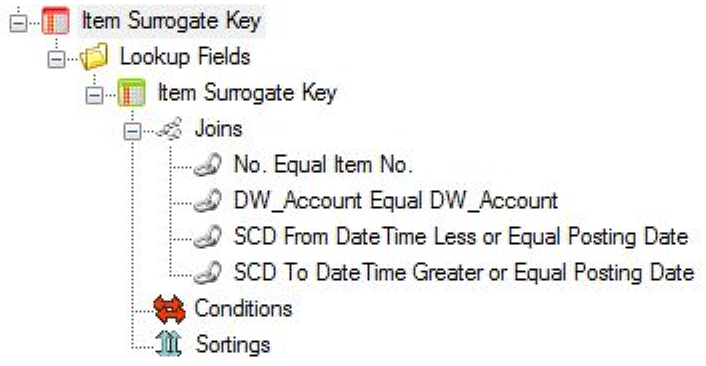
A surrogate key is a substitution for the primary key in a table. The surrogate key most often represents the unique row number in the table. It can be used in one table to refer back to a specific record in another table without having to utilize the natural primary key. In TX2014, all tables in the staging database and data warehouse have a called named "DW_ID" which represents the surrogate key in each respective table.

In order to see the DW_ID field in TX2014 refer to the following steps and screen-shot below:

1. Right-click on the table, and go to Advanced -> Show System Control Fields.

2. Move the DW_ID field from the Item table, and add it to the transaction table.
3. Rename the field to "Item Surrogate Key" to make it easily understandable to other users.
4. Add Standard joins between the two tables for DW_Account and the Item No.

5. Add Additional joins for "SCD From Date Time" Less Than or Equal to "Posting Date" and "SCD To Date Time" Greater Than or Equal to "Posting Date". This will capture the correction version of the item based on the Posting Date of the transaction.


Item Surrogate Key
Lookup Fields
Item Surrogate Key
Joins
No. Equal Item No.
DW_Account Equal DW_Account
SCD From DateTime Less or Equal Posting Date
SCD To DateTime Greater or Equal Posting Date
Conditions
Sortings

The transaction table is then deployed and executed so that the new field is added and populated.

The process above should be repeated for any additional transaction tables when history needs to be tracked.

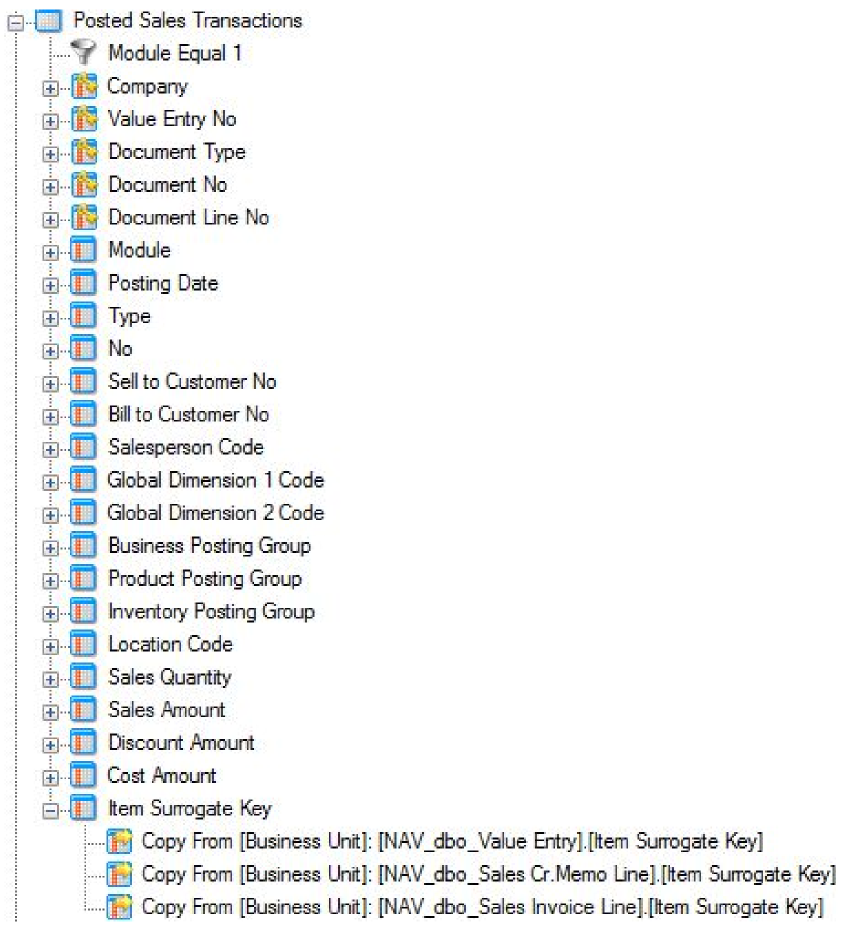
Move Surrogate Key from Transaction Table in Staging Database to Data Warehouse

Now that the surrogate key has been added to the transaction table(s), this field needs to be added to the relevant transaction tables in the data warehouse.

This is accomplished in the following steps:

1. Drag the surrogate key field (in this case "Item Surrogate Key") from the table(s) in the staging database and drop them onto the relevant tables on the data warehouse.

2. Deploy and execute the transaction table in the data warehouse for these changes to be propagated to the SQL tables.



Posted Sales Transactions
Module Equal 1
Company
Value Entry No
Document Type
Document No
Document Line No
Module
Posting Date
Type
No
Sell to Customer No
Bill to Customer No
Salesperson Code
Global Dimension 1 Code
Global Dimension 2 Code
Business Posting Group
Product Posting Group
Inventory Posting Group
Location Code
Sales Quantity
Sales Amount
Discount Amount
Cost Amount
Item Surrogate Key
Copy From [Business Unit]: [NAV_dbo_Value Entry].[Item Surrogate Key]
Copy From [Business Unit]: [NAV_dbo_Sales Cr.Memo Line].[Item Surrogate Key]
Copy From [Business Unit]: [NAV_dbo_Sales Invoice Line].[Item Surrogate Key]

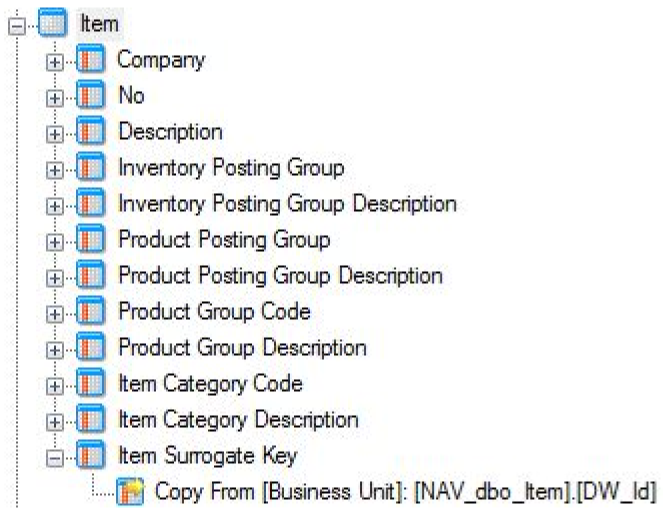
Add Surrogate Key to Dimension Table in Data Warehouse

The DW_ID field now needs to be added to the related dimension table in the data warehouse. This ensures that the proper mapping will be made between the dimension table and the transaction table in the cubes.

To perform this action:

1. Drag the DW_ID field from the dimension table in the staging data (in this example, the Item table) to the related dimension table in the data warehouse (in this example, the Item table).

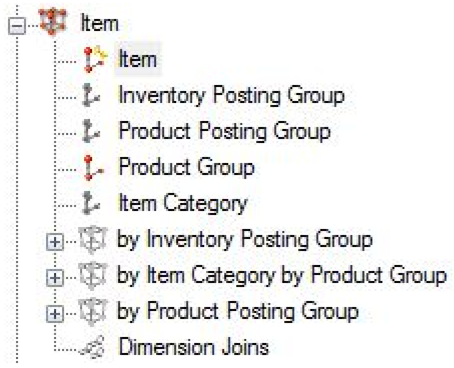
2. Deploy and execute the dimension table for the changes to be propagated to the SQL database.


Item
Company
No
Description
Inventory Posting Group
Inventory Posting Group Description
Product Posting Group
Product Posting Group Description
Product Group Code
Product Group Description
Item Category Code
Item Category Description
Item Surrogate Key
Copy From [Business Unit]: [NAV_dbo_Item].[DW_Id]

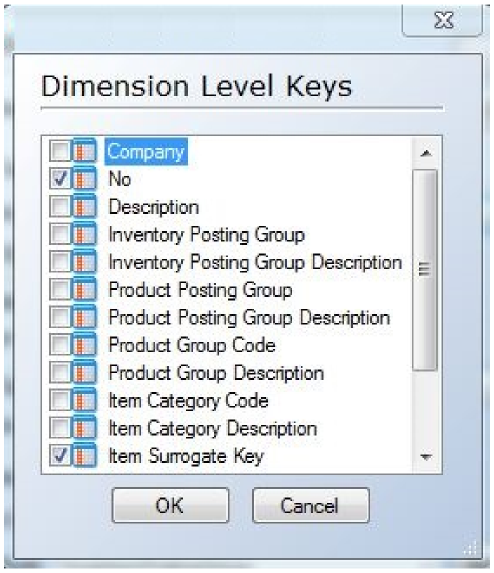
Update Dimension Key

The dimension key must now be updated to include this surrogate key. This will ensure that Analysis Services sees the uniqueness of the dimension, not as the natural key (in this example Item No.), but as the combination of Item No. and the surrogate key.

1. On the Cubes tab, locate the dimension under the Dimensions node, and edit the key level (in this example "Item").


Item
Item
Inventory Posting Group
Product Posting Group
Product Group
Item Category
by Inventory Posting Group
by Item Category by Product Group
by Product Posting Group
Dimension Joins

2. To the right of the Key Column click the ellipsis (...), and add the surrogate key to the dimension key (in this example, it is the "ItemSurrogate Key" field). Click Ok.


Dimension Level Keys

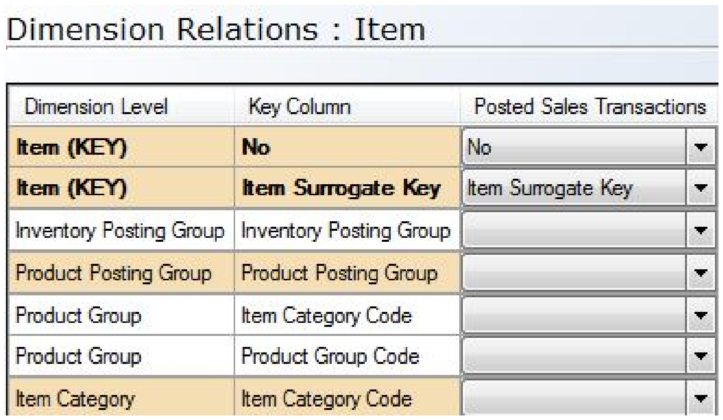
☐	Company
✓	No
☐	Description
☐	Inventory Posting Group
☐	Inventory Posting Group Description
☐	Product Posting Group
☐	Product Posting Group Description
☐	Product Group Code
☐	Product Group Description
☐	Item Category Code
☐	Item Category Description
✓	Item Surrogate Key

Update the Dimension Relationships in the Cube

The relationships between the dimension and the transaction table must be updated in the cube to reflect the change made to the dimension key in the previous step.

1. Right-click the relevant dimension in the cube(s), and select Dimension Relations -> All Fact Tables.

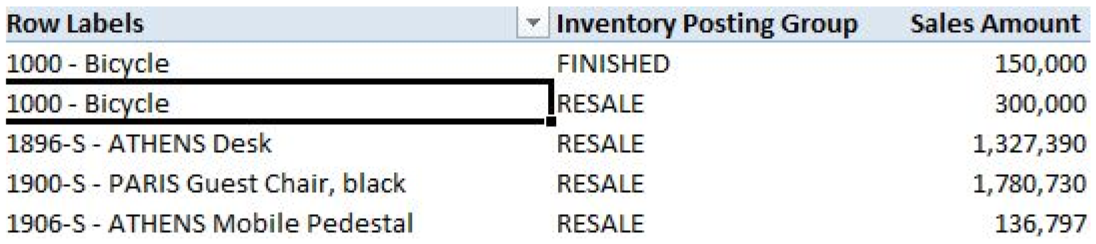
2. Set the dimension relationship to use the surrogate key that was added to the transaction table in a previous step.


Dimension Relations : Item

Dimension Level	Key Column	Posted Sales Transactions
Item (KEY)	**No**	No
Item (KEY)	**Item Surrogate Key**	Item Surrogate Key
Inventory Posting Group	Inventory Posting Group	
Product Posting Group	Product Posting Group	
Product Group	Item Category Code	
Product Group	Product Group Code	
Item Category	Item Category Code	

3. Deploy and execute the OLAP database on the cubes tab for the changes to take effect.

The final result is that users can see data based on the historical attributes that may no longer exist in their ERP system because the information has been overwritten. In the screen-shot below, the "Bicycle" item shows up twice with the sales amounts associated with the various Inventory Posting Groups that have been used for the item over time.

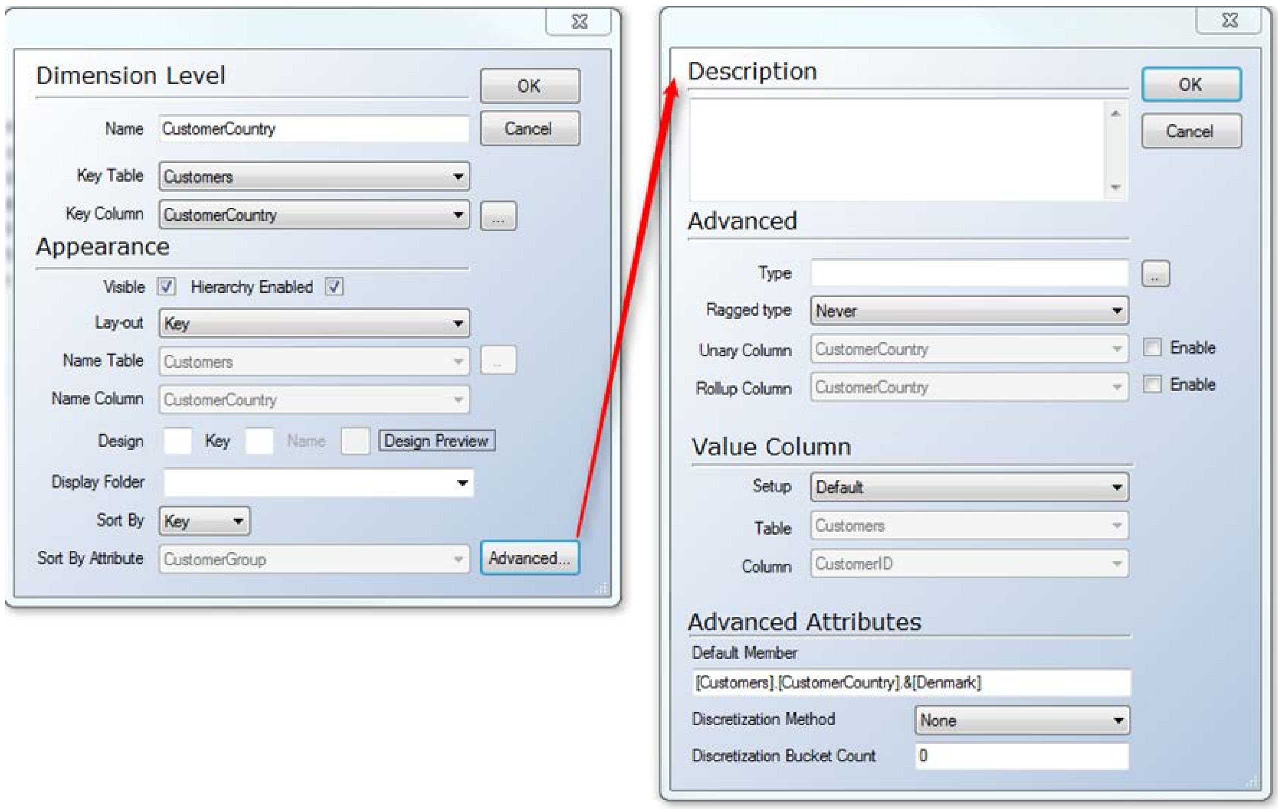


Row Labels	Inventory Posting Group	Sales Amount
1000 - Bicycle	FINISHED	150,000
1000 - Bicycle	RESALE	300,000
1896-S - ATHENS Desk	RESALE	1,327,390
1900-S - PARIS Guest Chair, black	RESALE	1,780,730
1906-S - ATHENS Mobile Pedestal	RESALE	136,797

Dimension Attribute Properties

It is possible to control additional properties on the Dimension Levels (Attributes).

The properties can be controlled by clicking the ‘Advanced…’ button:



Dimension Level

Name	CustomerCountry
Key Table	Customers
Key Column	CustomerCountry

Appearance

Visible	✓
Hierarchy Enabled	✓
Lay-out	Key
Name Table	Customers
Name Column	CustomerCountry
Design	Key Name
Display Folder	
Sort By	Key
Sort By Attribute	CustomerGroup

Description

Advanced

Type	
Ragged type	Never
Unary Column	CustomerCountry
Rollup Column	CustomerCountry

Value Column

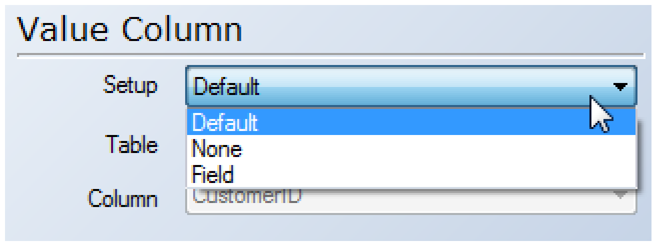
Setup	Default
Table	Customers
Column	CustomerID

Advanced Attributes

Default Member	[Customers].[CustomerCountry].&[Denmark]
Discretization Method	None
Discretization Bucket Count	0

Value Column

It is possible to change the Value Column. Choose between Default, None, or Field:



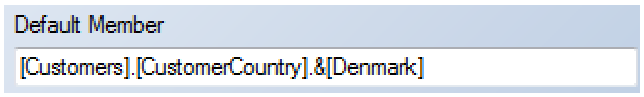
Default = Name Column for the level

None = No Value Column

Field = Select a Table / Field to function as the Value Column

Default Member

You can enter the MDX to identify the Default Member.

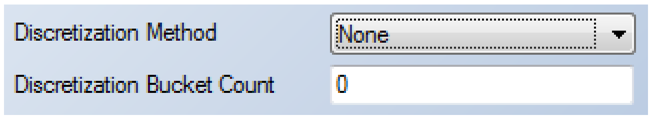


Default Member

[Customers].[CustomerCountry].&[Denmark]

Discretization

Discretizations are used in some Data Mining scenarios, and the settings can now be controlled from within tX:

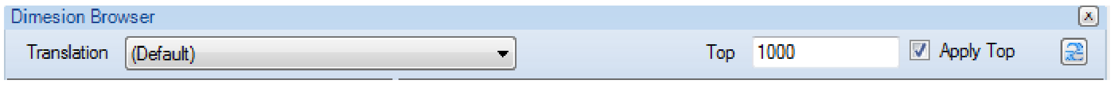


Discretization Method	None
Discretization Bucket Count	0

Dimension Browser

To browse the contents of a dimension, right click the dimension and click Browse Dimension.

There is an option to only show a specified number of members when browsing a dimension. This is quite useful on dimension levels with many members.


Dimesion Browser

Translation (Default)

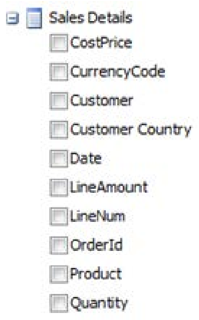
Top 1000

[✓] Apply Top

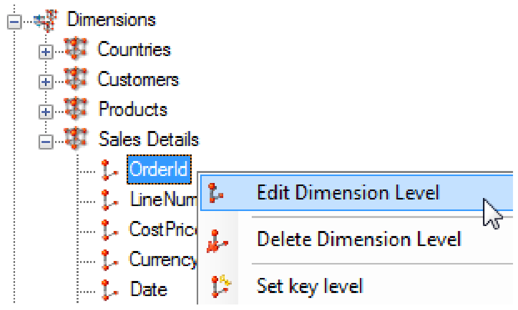
Display folders on Dimensions

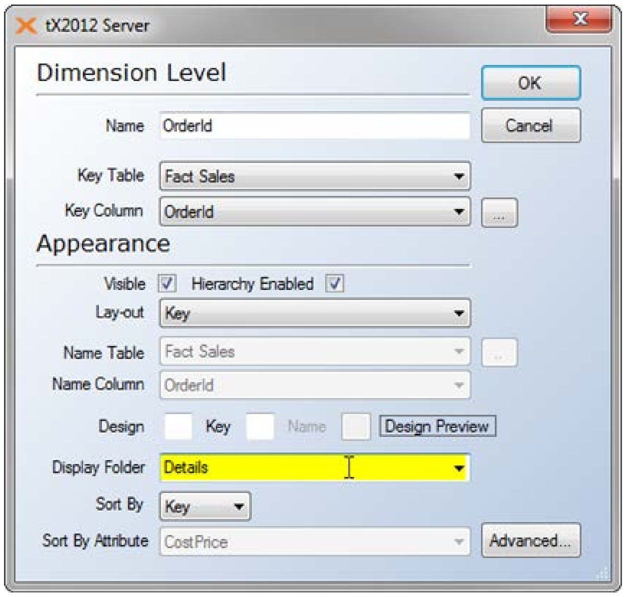
The individual attributes on a dimension can be organized in folders. This is very helpful when the dimension contains many levels (attributes). On the dimension node, a default folder can be entered. The default folder will be used for all attributes that do not have another folder assigned.

Here is an example of a dimension without display folders as shown in Excel 2010:



To add a level to a display folder, simply enter the name of the folder in the Edit Dimension Level dia-logue:




tX2012 Server

Dimension Level

Name: OrderId

Key Table: Fact Sales

Key Column: OrderId

Appearance

Visible [✓] Hierarchy Enabled [✓]

Lay-out: Key

Name Table: Fact Sales

Name Column: OrderId

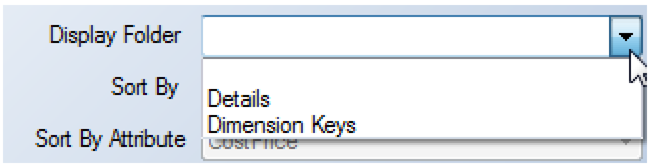
Design: Key, Name, Design Preview

Display Folder: Details

Sort By: Key

Sort By Attribute: CostPrice

In the dropdown, a list of already used Display Folders is displayed:


Display Folder

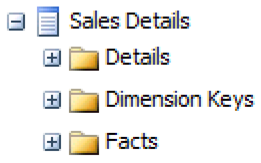
Sort By

Sort By Attribute

Details

Dimension Keys

The result as seen in Excel 2010:


Sales Details

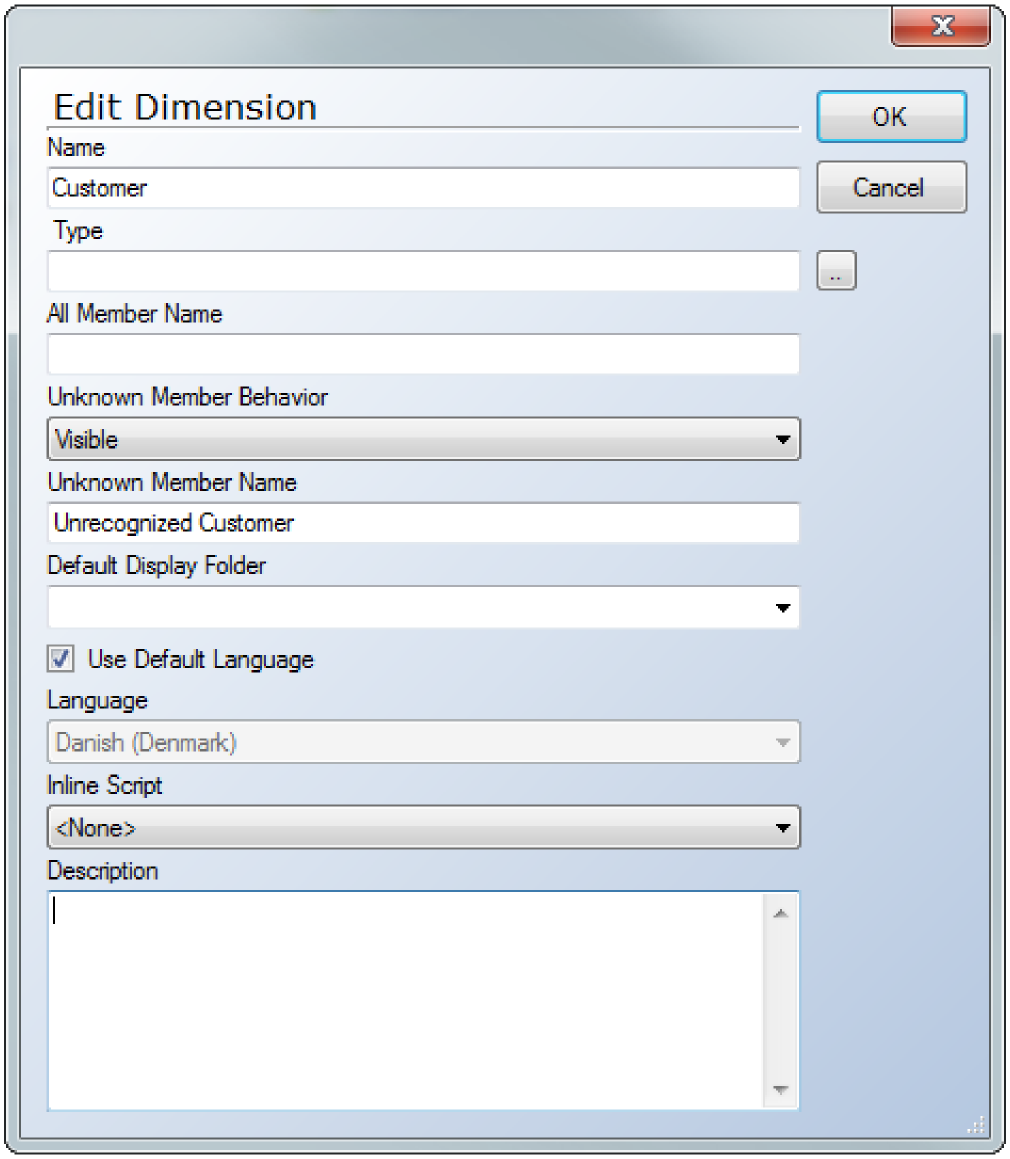
Details

Dimension Keys

Facts

Rename of Unknown Member Name

The Unknown Member Name can be changed from the Edit Dimension dialogue:



Edit Dimension

Name: Customer

Type:

All Member Name:

Unknown Member Behavior: Visible

Unknown Member Name: Unrecognized Customer

Default Display Folder:

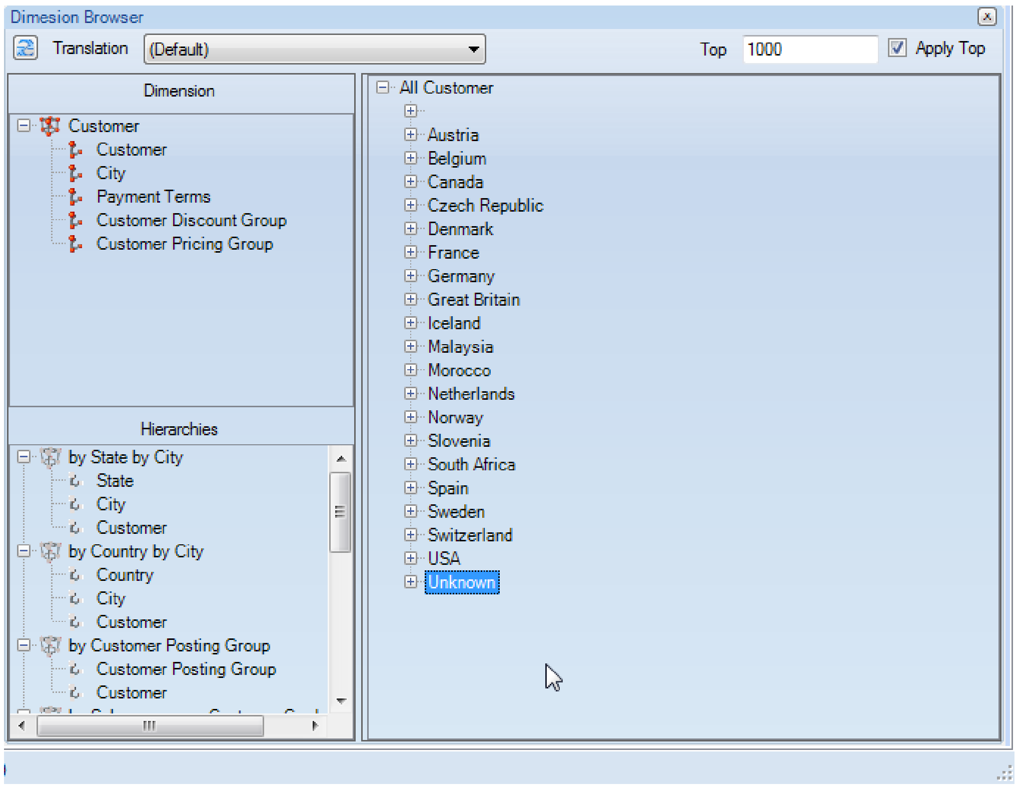
[✓] Use Default Language

Language: Danish (Denmark)

Inline Script: <None>

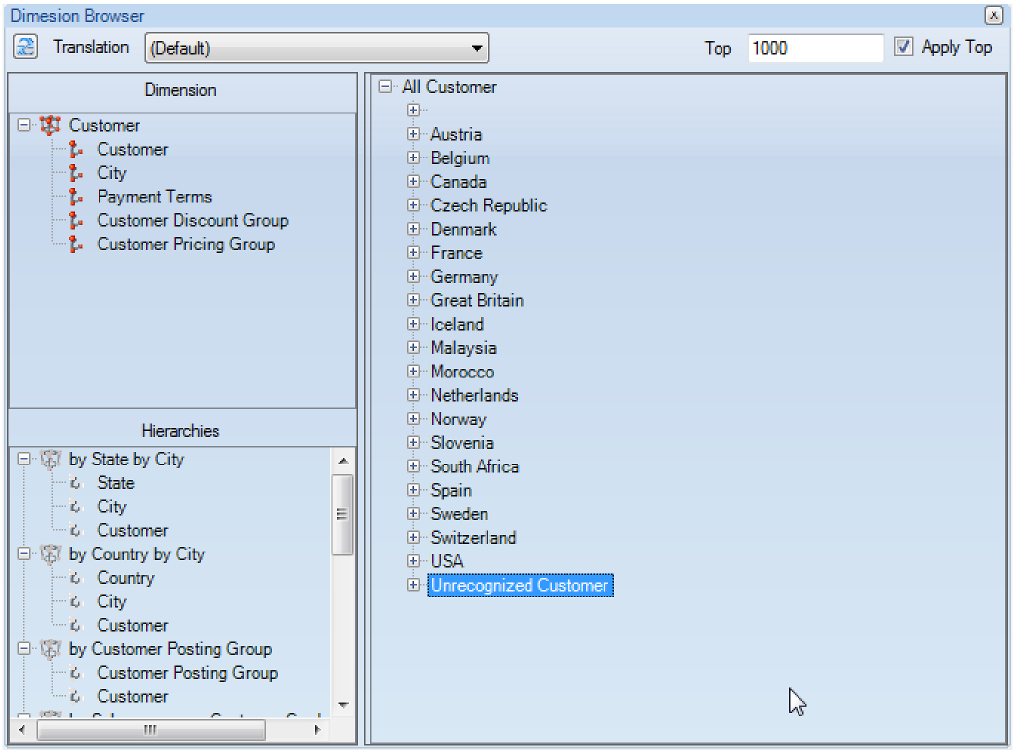
Description:

Before:



Dimesion Browser
Translation
(Default)
Top
1000
Apply Top
Dimension
Customer
Customer
City
Payment Terms
Customer Discount Group
Customer Pricing Group
Hierarchies
by State by City
State
City
Customer
by Country by City
Country
City
Customer
by Customer Posting Group
Customer Posting Group
Customer
All Customer
Austria
Belgium
Canada
Czech Republic
Denmark
France
Germany
Great Britain
Iceland
Malaysia
Morocco
Netherlands
Norway
Slovenia
South Africa
Spain
Sweden
Switzerland
USA
Unknown

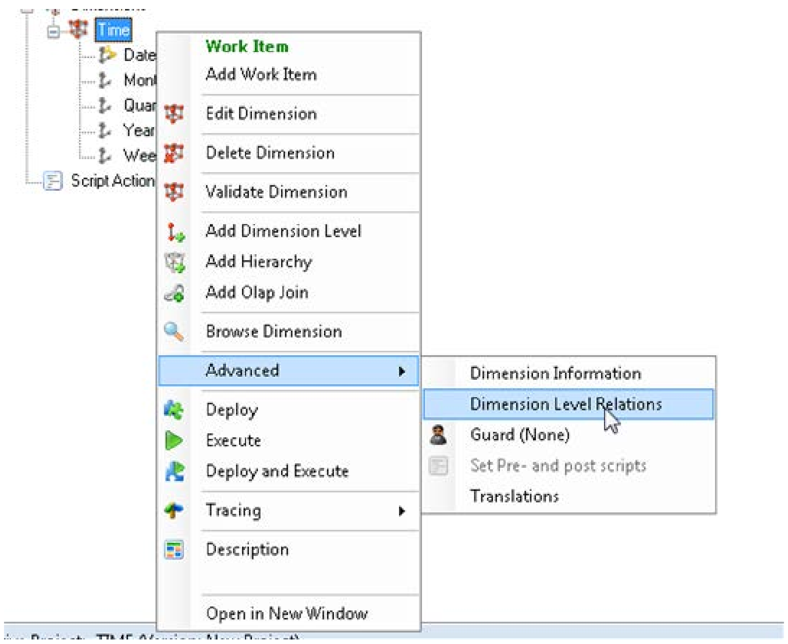
After:



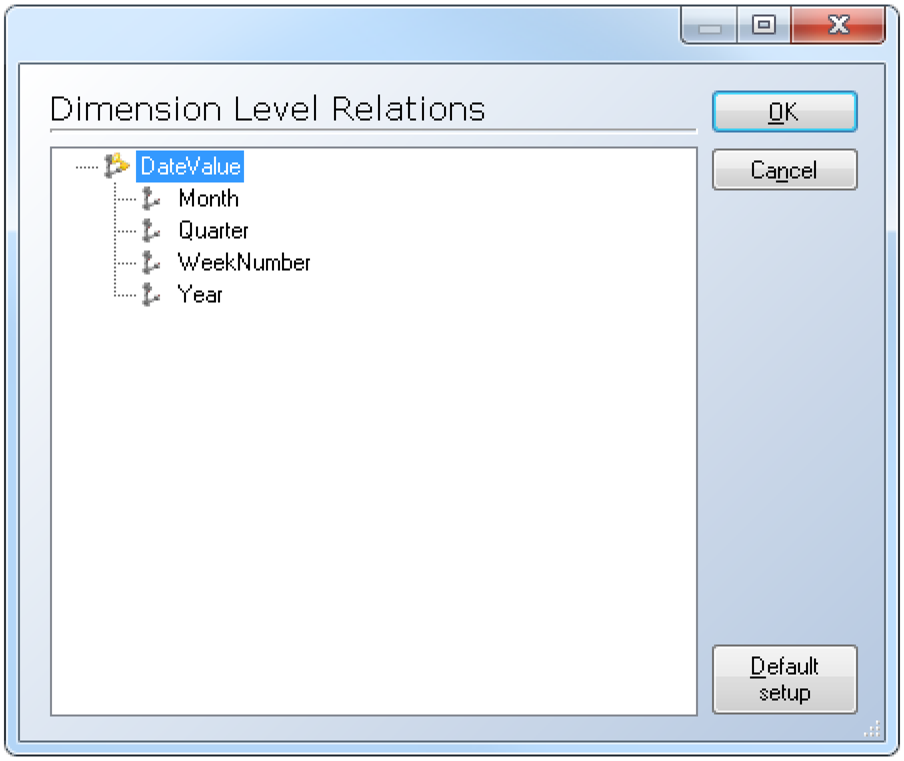
Dimesion Browser
Translation
(Default)
Top
1000
Apply Top
Dimension
Customer
Customer
City
Payment Terms
Customer Discount Group
Customer Pricing Group
Hierarchies
by State by City
State
City
Customer
by Country by City
Country
City
Customer
by Customer Posting Group
Customer Posting Group
Customer
All Customer
Austria
Belgium
Canada
Czech Republic
Denmark
France
Germany
Great Britain
Iceland
Malaysia
Morocco
Netherlands
Norway
Slovenia
South Africa
Spain
Sweden
Switzerland
USA
Unrecognized Customer

Use an attribute more than once in “attribute relationship”

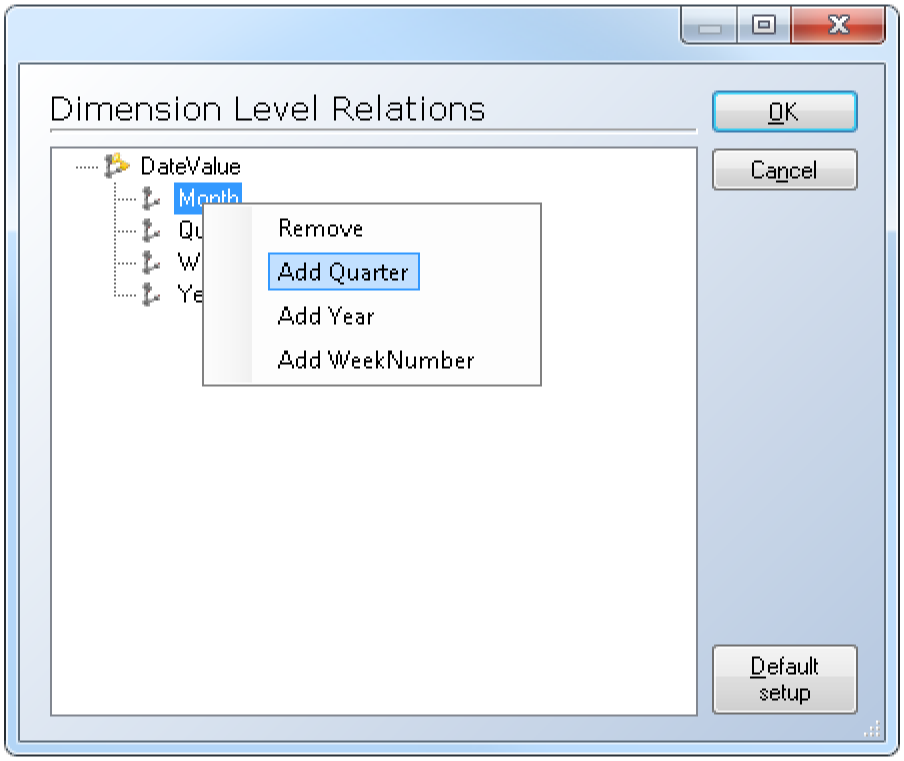
The setup dialog is found on the dimension node under Advanced:



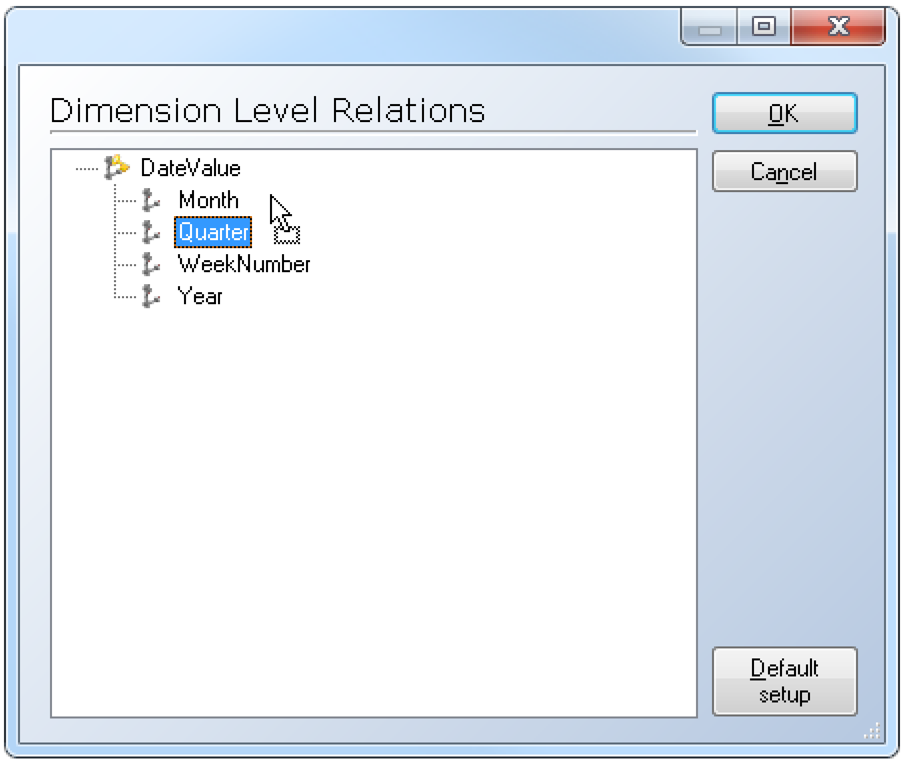
The default setup is that key level is the root and all other dimension levels are related to the key level:



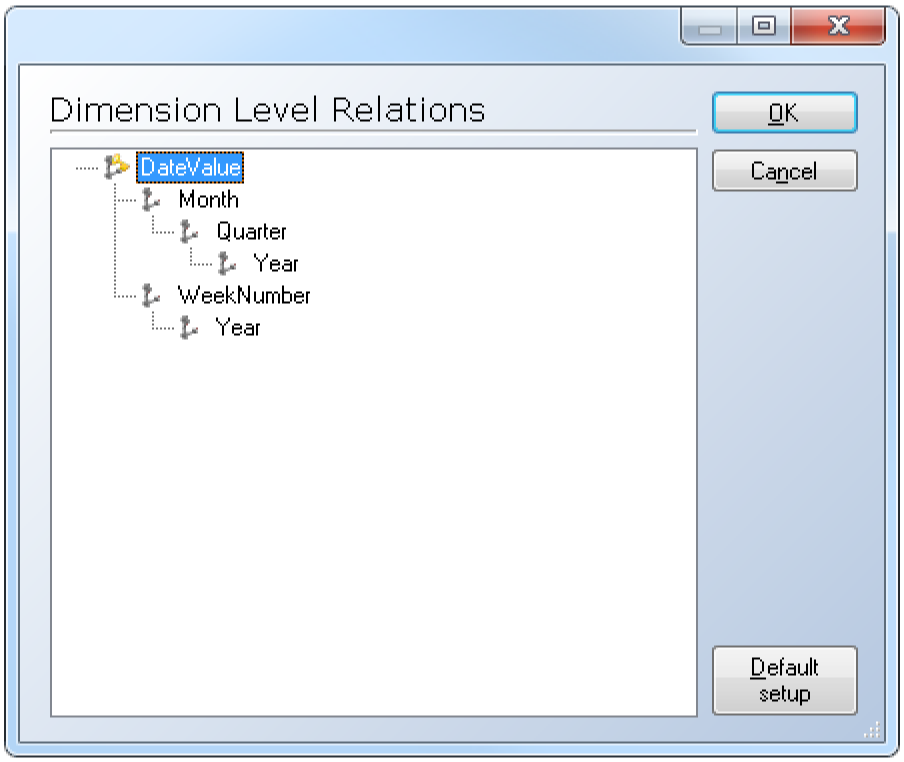
Relations between dimension levels are done by right clicking the parent dimension level and selecting the dimension level to relate:



Relations between dimension levels can also be done by dragging a dimension level on to a parent dimension level:



Here is an example of dimension level used more than once:



```
DateValue
    Month
        Quarter
            Year
    WeekNumber
        Year
```

Circular references are not allowed, and tX will validate references to ensure this.

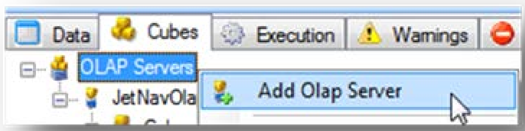
Creating Cubes

OLAP cubes allow you to present data in a multidimensional model. You can break down data in your data warehouse into smaller units, enabling you to drill-down, or roll-up through data, depending on the level of detail you want to view. You can, for example, create a sales cube, a production cube, a finance cube, and so on.

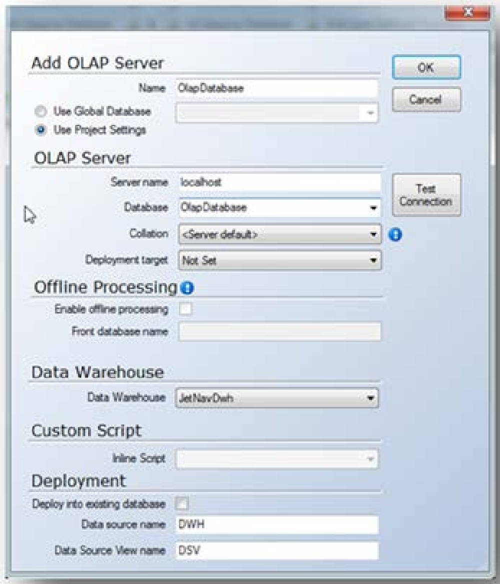
A cube consists of a number of dimensions and measures. The dimensions determine the structure of the cube, and the measures represent the numerical values. You can, furthermore, define hierarchies within a dimension by using dimension levels. See To Add Dimension Levels.

To Add OLAP Servers

1. On the **Cubes** tab, right-click **OLAP Servers**, and then select **Add OLAP Server**.



A window with the following selections will appear:



Add OLAP Server

Name: OlapDatabase

( ) Use Global Database
(●) Use Project Settings

OLAP Server

Server name: localhost

Database: OlapDatabase

Collation: <Server default>

Deployment target: Not Set

Offline Processing

Enable offline processing: ☐

Front database name:

Data Warehouse

Data Warehouse: JetNavDwh

Custom Script

Inline Script:

Deployment

Deploy into existing database: ☐

Data source name: DWH

Data Source View name: DSV

2. In the **Name** field, type a name for the OLAP server. The name cannot exceed 15 characters in length.

3. In the **Server Name** field, type the name of the OLAP database server.

4. In the **Database** field, type a name for the database.

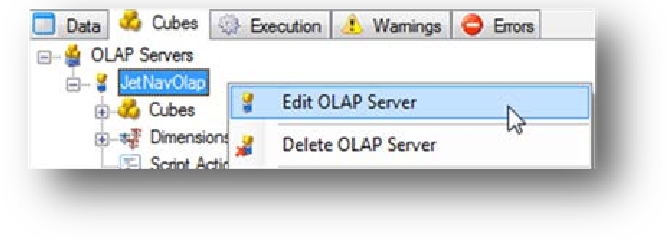
5. In the **Collation** field select the database collation to use for the OLAP database. **<Server Default>** will inherit the default collation currently set in Analysis Services. **<Application Default>** will use Latin1_General_CI_AS. This collation should correspond with the collation that is set for SQL Analysis Services.

6. Select a data warehouse from the **Data Warehouse** list, and then click **OK**. Each OLAP database can pull from a single data warehouse database.

To Edit OLAP Server Settings

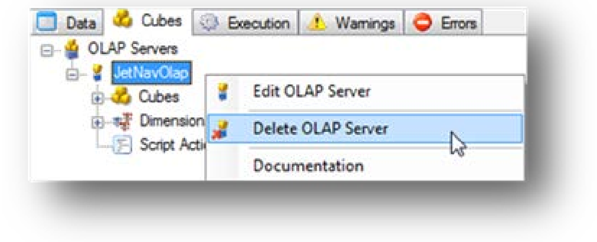
1. On the **Cubes** tab, expand OLAP servers, and then right-click the OLAP server you wish to edit.

2. Select **Edit OLAP Server**,make the appropriate changes, and then click **Ok**.



To Delete an OLAP Database

1. In the **Cubes** tree, right-click the OLAP database you want to delete, and then select **Delete OLAP Server**.

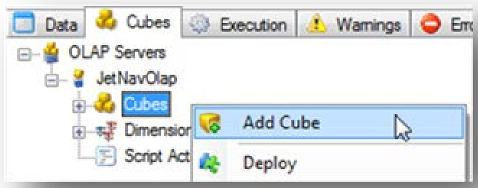


2. Click **Yes** to delete the server.

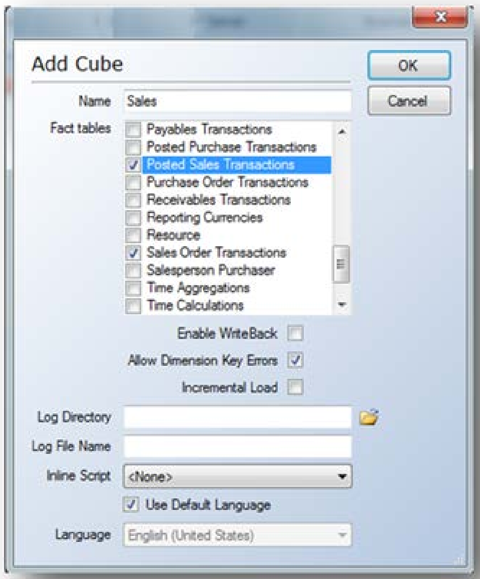
Note: Deleting the OLAP database will remove the database from the project, but it will not delete the physical database on the OLAP Server itself. This must be done manually through SQL Management Studio.

To Add Cubes

1. On the **Cubes** tab, expand the OLAP server, right-click **Cubes**, and then select **Add Cube**.



A window will appear with the following selections:


Add Cube

Name: Sales

Fact tables:
- [ ] Payables Transactions
- [ ] Posted Purchase Transactions
- [✓] Posted Sales Transactions
- [ ] Purchase Order Transactions
- [ ] Receivables Transactions
- [ ] Reporting Currencies
- [ ] Resource
- [✓] Sales Order Transactions
- [ ] Salesperson Purchaser
- [ ] Time Aggregations
- [ ] Time Calculations

Enable WriteBack [ ]
Allow Dimension Key Errors [✓]
Incremental Load [ ]

Log Directory:
Log File Name:
Inline Script: <None>
[✓] Use Default Language
Language: English (United States)

2. In the **Name** field, type a name for the cube.

3. In the **Fact Table** list, select the table(s) you want to use for the cube. All of the tables in the data warehouse are displayed, and you can select more than one table to use.

4. If you want end users to be able to change cube data while they browse it, select **Enable Write-back**. Any changes the end users make are saved in the write-back table.

Note: You can only enable write-back if the front-end application supports write-back.

5. If you want to continue processing the cube, even if dimension key errors occur, select **AllowDimension Key Errors**. When you allow dimension key errors, all errors are reported to a log, hence it is necessary to specify a Log Directory. Checking this box is the recommended default setting for TX2014.

6. In the **Log Directory**, click the file folder, and navigate to where you want to store the log.

7. In the **Log File Name** field, type a name for the log, and then click **OK**.

To Add a Single Dimension to a Cube

1. Expand **Dimensions**, left-click the preferred dimension, and then drag it to the preferred cube.

2. Drop the dimension on either the cube itself or on the **Dimensions** node of the cube.

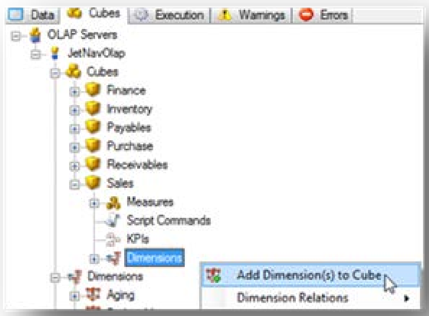
3. Set the relationship to the fact table in the cube. See To Add Dimension Relationships.

To Add Multiple Dimensions to a Cube

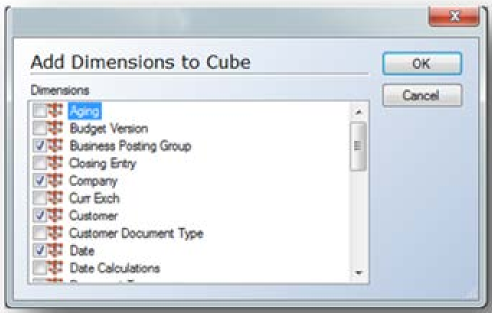
You can also add multiple dimensions to a cube in one operation.

1. On the **Cubes** tab, expand the preferred OLAP server, expand **Cubes**, and then expand the cube chosen for editing.

2. Right-click **Dimensions**, and then select **Add Dimension to Cube**. All dimensions in your project are listed.



A window will appear with the following selections:



- [ ] Aging
- [ ] Budget Version
- [✓] Business Posting Group
- [ ] Closing Entry
- [✓] Company
- [ ] Curr Exch
- [✓] Customer
- [ ] Customer Document Type
- [✓] Date
- [ ] Date Calculations

3. Select the dimension or dimensions you want to add, and then click **OK**.

You must then set the relationship to the fact table in the cube. See To Add Dimension Relationships.

To Add Role-Playing Dimensions

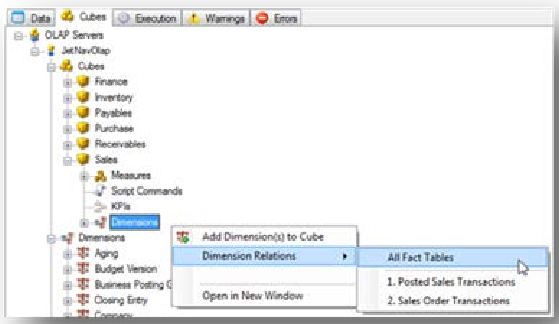
Role-playing dimensions are dimensions that are used more than once in the same cube. For example, you can use a Customer dimension more than once in the same cube.

When you have added a role-playing dimension to a cube, you will follow the steps described above to add the dimension and then specify a new name for the dimension to distinguish it from the other dimensions of the same type. Then define the relationship to the proper field in the fact table. For example, the Customer dimension may be used as the Bill-to Customer dimension, which relates to the Bill-to Customer No. field in the fact table, while the Sell-to Customer dimension may relate to the Sell-to Customer No. field in the same fact table.

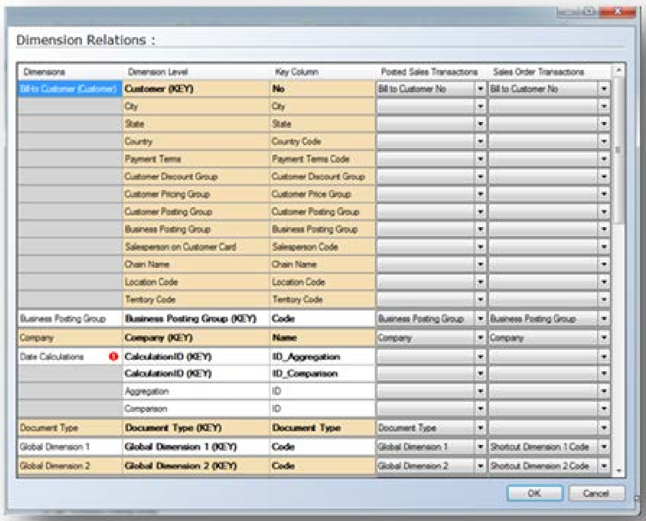
To Add Dimension Relationships

Dimension relationships specify how a dimension is related to a fact table. You must define how each level in a dimension is related to a fact table.

1. On the **Cubes** tab, expand **OLAP Servers**, and then expand the OLAP server that contains your chosen cube. Expand the preferred cube, right-click **Dimensions**, and then select **Dimension Relations -> All Fact Tables**.



2. The Dimension Relations window is displayed. The table contains the following columns:



Dimension Relations :

Dimensions	Dimension Level	Key Column	Posted Sales Transactions	Sales Order Transactions
Bill to Customer (Customer)	**Customer (KEY)**	**No**	Bill to Customer No	Bill to Customer No
	City	City		
	State	State		
	Country	Country Code		
	Payment Terms	Payment Terms Code		
	Customer Discount Group	Customer Discount Group		
	Customer Pricing Group	Customer Price Group		
	Customer Posting Group	Customer Posting Group		
	Business Posting Group	Business Posting Group		
	Salesperson on Customer Card	Salesperson Code		
	Chain Name	Chain Name		
	Location Code	Location Code		
	Territory Code	Territory Code		
Business Posting Group	**Business Posting Group (KEY)**	**Code**	Business Posting Group	Business Posting Group
Company	**Company (KEY)**	**Name**	Company	Company
Date Calculations	**CalculationID (KEY)**	**ID_Aggregation**		
	CalculationID (KEY)	**ID_Comparison**		
	Aggregation	ID		
	Comparison	ID		
Document Type	**Document Type (KEY)**	**Document Type**	Document Type	
Global Dimension 1	**Global Dimension 1 (KEY)**	**Code**	Global Dimension 1	Shortcut Dimension 1 Code
Global Dimension 2	**Global Dimension 2 (KEY)**	**Code**	Global Dimension 2	Shortcut Dimension 2 Code

Dimension	Displays all dimensions in the cube
Dimension Level	Displays all dimension levels associated with each dimension
Key Column	Displays the key column for each dimension level
<Fact Table Name>	Displays the name of the fact table

3. In the Fact Table column, select the field where you will join the dimension key with the dimension. The dimension levels in **Bold** are the required values to be set for each dimension.

4. Click **OK** when you have created all of the desired relationships.

Adding Measures

Measures determine the numerical values of a cube. A cube must contain at least one measure.

You can define the following three types of measures:

• Standard measures obtain their values directly from a column in a source fact table.

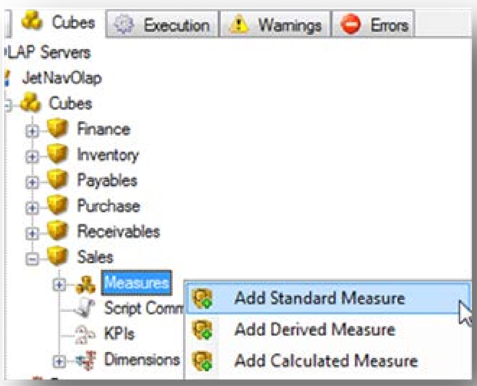
• Derived measures are derived before aggregation or summing of columns. This means that they are calculated when the cube is processed and are stored in the fact table. You can use standard arithmetic operators and MDX statements to create derived measures.

• Calculated measures are calculated after aggregation and summing. They are calculated at query time and are never stored. You can create calculated measures using standard arithmetic operators and MDX statements and can also combine them with other measures.

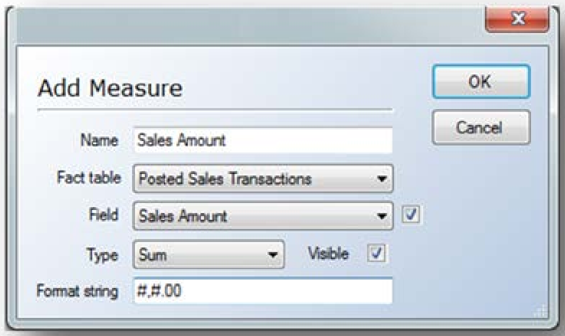
To Add Standard Measures

1. On the **Cubes** tab, expand the OLAP server that contains the cube for modification.

2. Expand the preferred cube, right-click **Measures**, and then select **Add Standard Measure**.



A window will appear with the following selections:



Add Measure

Name: Sales Amount

Fact table: Posted Sales Transactions

Field: Sales Amount ✓

Type: Sum Visible ✓

Format string: #,#.00

3. In the **Name** field, type a name for the measure.

4. In the **Fact Table**list, select the fact table that you want to use for the measure.

5. In the **Field** list, select the field that you want as the measure. Disable this field by clicking the box next to it.

6. In the **Type** field, select the preferred aggregation method. You have the following options:

SUM	Returns the sum of all values
COUNT	Counts all rows and returns the total number of rows
MIN	Returns the lowest value
MAX	Returns the highest value
DistinctCount	Returns the number of unique values

7. Select the **Visible** box if you want the measure to be displayed in the front-end application.

8. In the Format string field, specify how you want the numeric results displayed. You have the following options:

None	Applies no formatting
0	Displays a digit if the value has a digit where the zero (0) appears in the string, otherwise a zero is displayed
#	Displays a digit if the value has a digit where the number sign (#) appears in the string, otherwise nothing is displayed
.	Determines the number of digits displayed to the left and right of the decimal separator.
%	Is a percentage placeholder
,	Separates thousands from hundreds
Percent	Typing **Percent** will default the measure to showing as a percentage with two decimal places

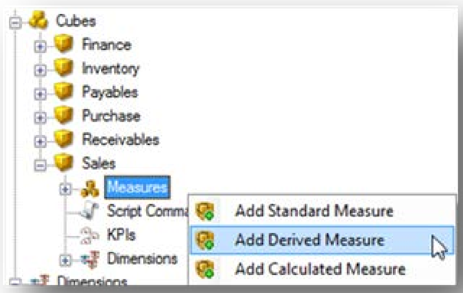
Below are examples of what the output will look like for various combinations of the format strings:

None	1234567.89
#,#	1,234,567
#,#.00	1,234,567.89
#,#%	1%
#,#.0%	1.2%
Percent	1.23%

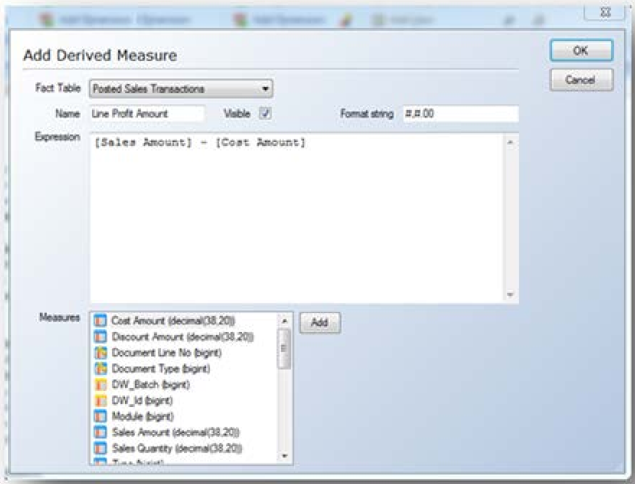
To Add Derived Measures

1. On the **Cubes** tab, expand Cubes, and then expand the preferred cube.

2. Right-click Measures, and then select **Add Derived Measure**.



A window with the following selections will appear:


Add Derived Measure

Fact Table: Posted Sales Transactions

Name: Line Profit Amount

Visible: ✓

Format string: #,#.00

Expression:

```
[Sales Amount] - [Cost Amount]
```

Measures:

- Cost Amount (decimal(38,20))
- Discount Amount (decimal(38,20))
- Document Line No (bigint)
- Document Type (bigint)
- DW_Batch (bigint)
- DW_Id (bigint)
- Module (bigint)
- Sales Amount (decimal(38,20))
- Sales Quantity (decimal(38,20))

3. In the Fact Table list, select the fact table that you want to use for the measure.

4. In the Name field, type a name for the derived measure.

5. Check the Visible box if you want the measure to be displayed in the front-end application.

6. In the Format String field, specify how you want the numeric results displayed. These are the same as for the Standard Measures above.

7. In the Expression field, write an MDX statement

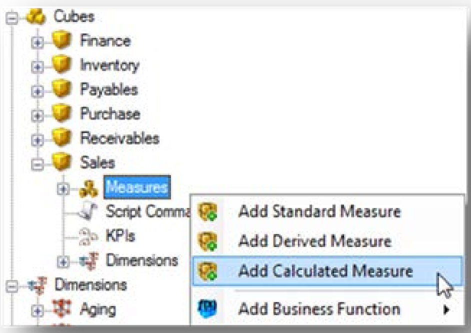
- OR-

In the Measures list, select the measures from the fact table that you want to use for the derived measure, and then click Add. Mathematical operators can be used as well.

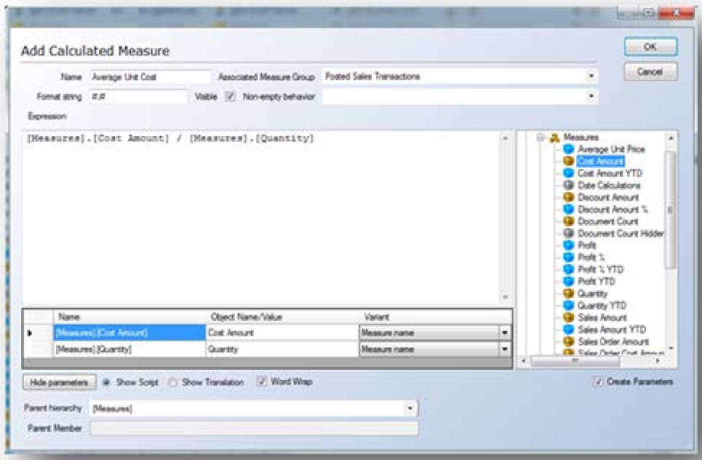
8. Enter the preferred operators, and then click **OK**.

To Add Calculated Measures

1. On the **Cubes** tab, expand **Cubes**, and then right-click the preferred cube. Select **Add Calculated Measure**.



A window with the following selections will appear:


Add Calculated Measure

Name: Average Unit Cost   Associated Measure Group: Posted Sales Transactions

Format string: #,#   Visible [✓]   Non-empty behavior:

Expression

```
[Measures].[Cost Amount] / [Measures].[Quantity]
```

	Name	Object Name/Value	Variant
▸	[Measures].[Cost Amount]	Cost Amount	Measure name
	[Measures].[Quantity]	Quantity	Measure name

Hide parameters   ● Show Script   ○ Show Translation   [✓] Word Wrap   [✓] Create Parameters

Parent hierarchy: [Measures]

Parent Member:

2. In the **Name** field, type a name for the calculated measure.

3. Check the **Visible** box if you want the value to be displayed in the front-end application.

4. In the **Format string** field, specify how you want the numeric results displayed. These are the same as the values for Standard Measures above.

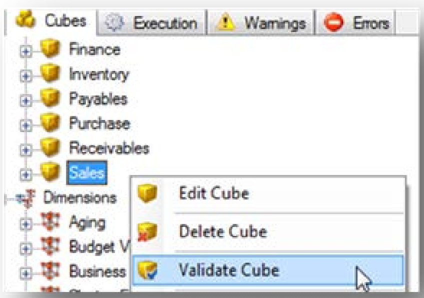
5. In the **Non-empty** behavior list, select the measure or measures used to resolve NON EMPTY queries in MDX. This is optional and left blank by default.

6. In the **Expression** field, write an MDX statement or, in the **Measures** list, drag the measures to be used for the calculated measure into the workspace in the middle.

To Validate Cubes

1. On the **Cubes** tab, expand the OLAP server that contains the cubes you want to validate, and then expand **Cubes**.

2. Right-click the cube you want to validate, and then select **Validate Cube**.

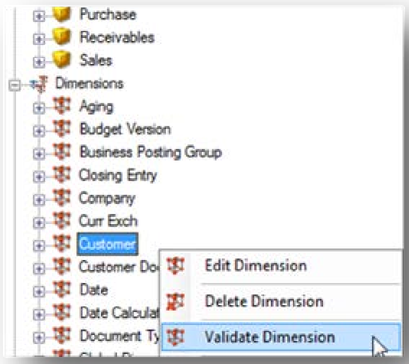


3. If the cube is valid, an OK message is displayed. Click OK to close the message dialog.

4. If the cube is invalid, a message is displayed outlining which changes need to be made.

To Validate Dimensions

1. On the **Cubes** tab, expand **Dimensions**, right-click the dimension you want to validate, and then select **Validate Dimension**.



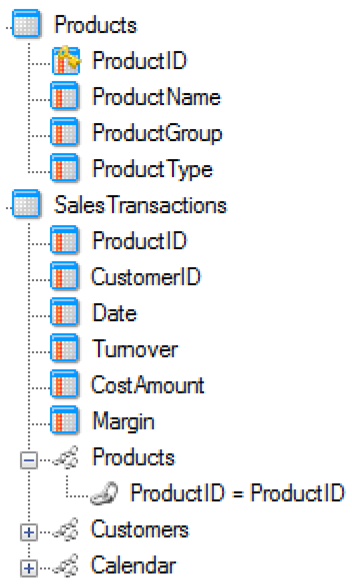
2. If the dimension is valid, an OK message is displayed. Click **OK** to close the message.

3. If the cube is invalid, a message is displayed outlining which changes need to be made.

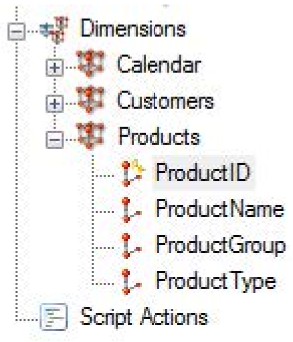
Use of DWH relations for OLAP relations

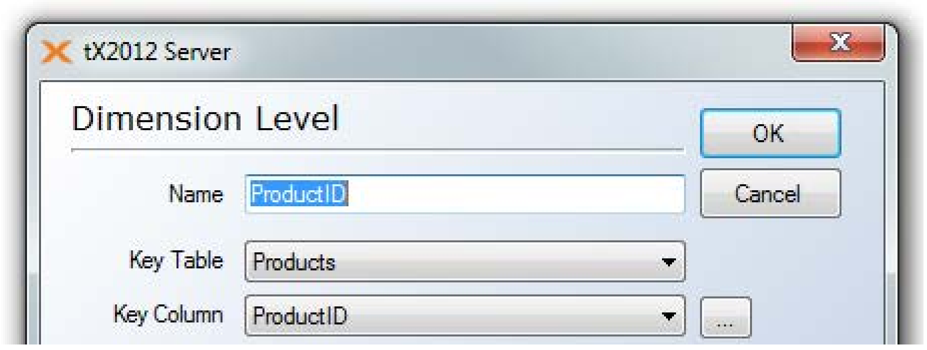
When adding dimensions to cubes, TX2014 will check if a relation between the dimension table and fact table(s) exists on the data warehouse. If it does, it can automatically relate the dimension to the fact table(s) on the cube.

In this example, we have a Products table that is related to a SalesTransactions table:


```
Products
    ProductID
    ProductName
    ProductGroup
    ProductType
SalesTransactions
    ProductID
    CustomerID
    Date
    Turnover
    CostAmount
    Margin
    Products
        ProductID = ProductID
    Customers
    Calendar
```

We have built a Products dimension based on the Products Table:


```
Dimensions
    Calendar
    Customers
    Products
        ProductID
        ProductName
        ProductGroup
        ProductType
Script Actions
```


tX2012 Server

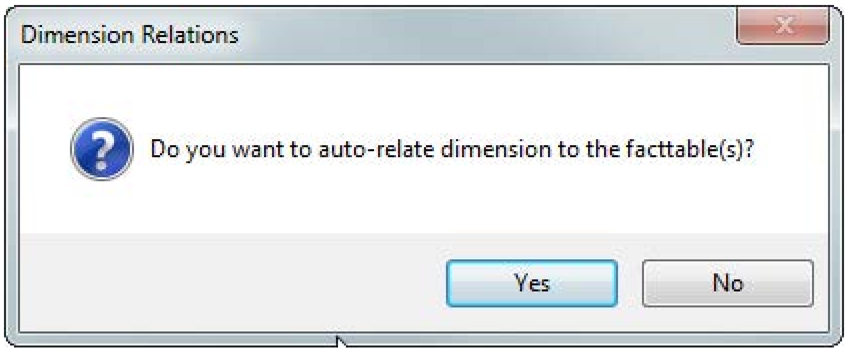
Dimension Level

Name: ProductID

Key Table: Products

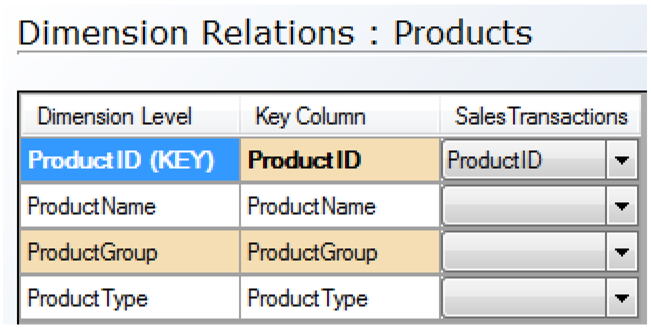
Key Column: ProductID

When dragging the Products dimension to the cube based on SalesTransactions, tX will ask you if you want to auto relate the dimension to the fact table(s):


Dimension Relations

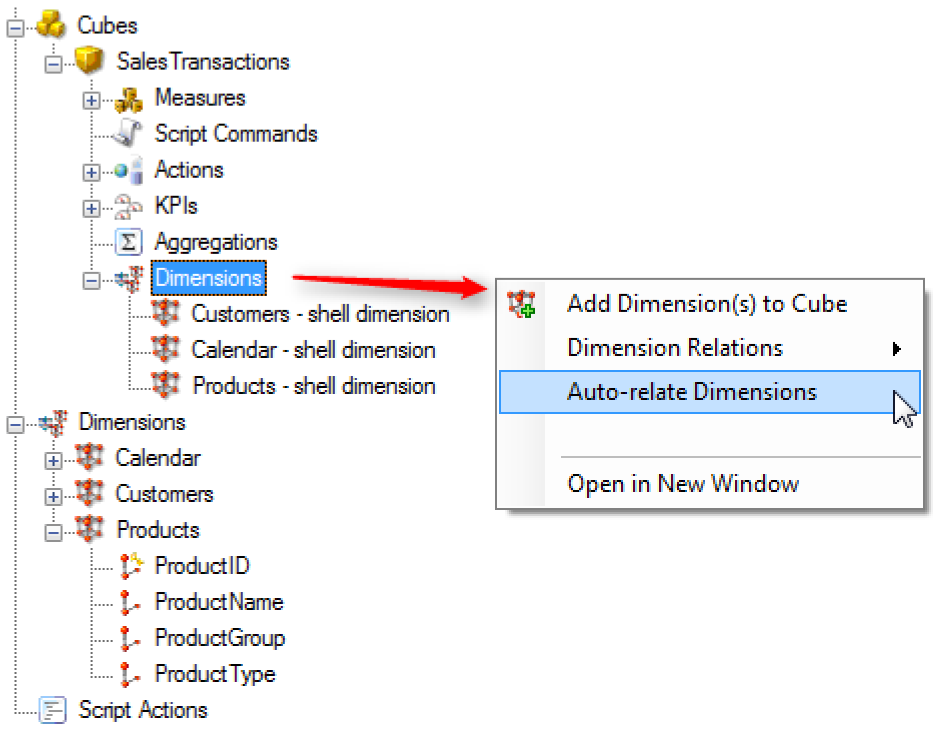
Do you want to auto-relate dimension to the facttable(s)?

The end result:


Dimension Relations : Products

Dimension Level	Key Column	SalesTransactions
ProductID (KEY)	ProductID	ProductID
ProductName	ProductName	
ProductGroup	ProductGroup	
ProductType	ProductType	

The feature can also be used on the cube-level, auto-relating all un-related (Shell) dimensions to the fact table(s):


Cubes
SalesTransactions
Measures
Script Commands
Actions
KPIs
Aggregations
Dimensions
Customers - shell dimension
Calendar - shell dimension
Products - shell dimension
Dimensions
Calendar
Customers
Products
ProductID
ProductName
ProductGroup
ProductType
Script Actions
Add Dimension(s) to Cube
Dimension Relations
Auto-relate Dimensions
Open in New Window

Drop down for format string on measures

To help with setting up format strings on measures, there is a drop down to select format strings on measures.

By default, the drop down list is populated by 4 different suggestions:

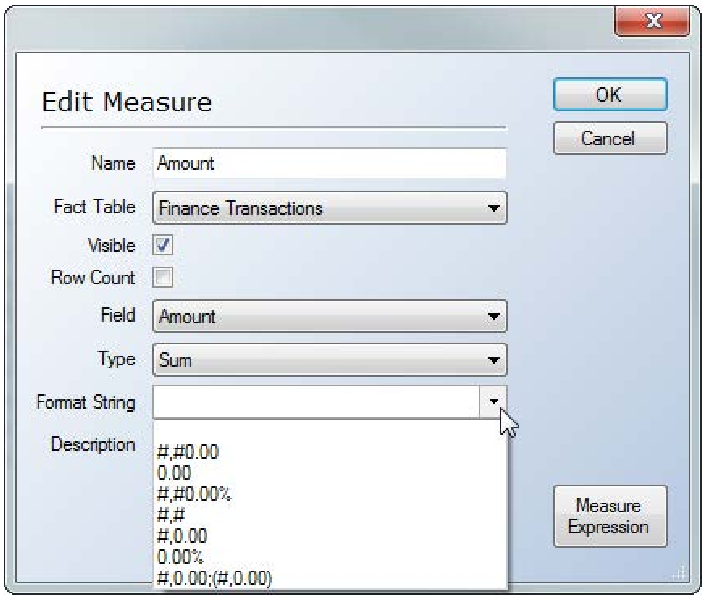
#,#

#,0.00

0.00%

#,0.00;(#,0.00)

If other format strings are used in the project, then these will be appended to the drop down list.


Edit Measure

Name: Amount

Fact Table: Finance Transactions

Visible: ✓

Row Count:

Field: Amount

Type: Sum

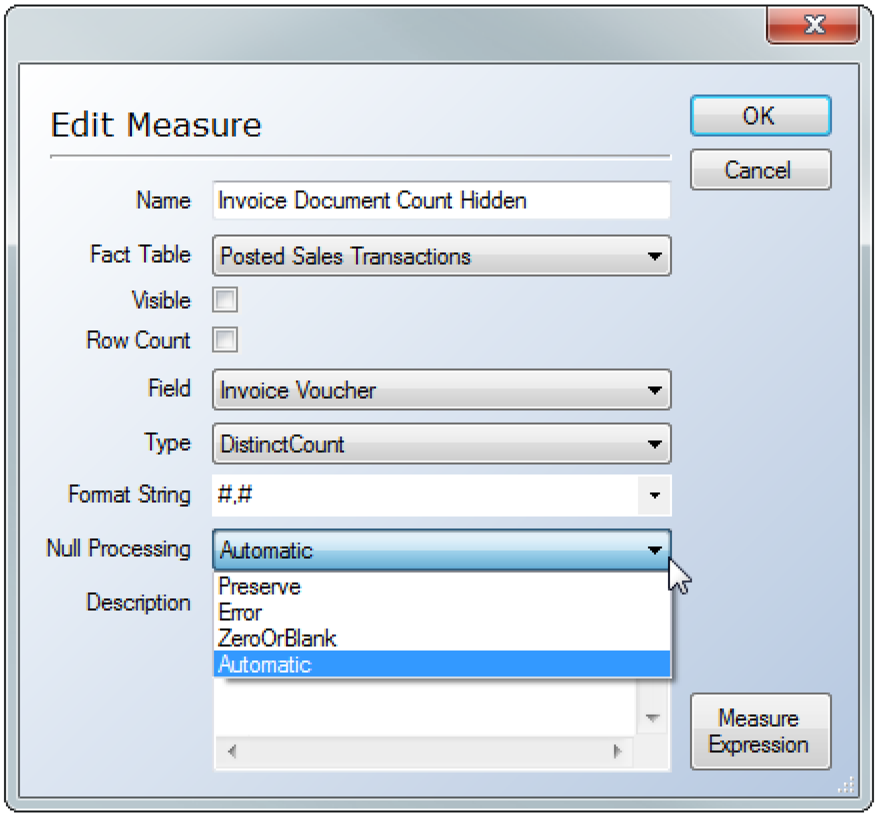
Format String:

Description

#,#0.00
0.00
#,#0.00%
#,#
#,0.00
0.00%
#,0.00;(#,0.00)

Changing NullProcessing Property on Measures

The NullProcessing Property on Measures can be changed from the Edit Measure dialogue:


Edit Measure

Name: Invoice Document Count Hidden

Fact Table: Posted Sales Transactions

Visible: ☐

Row Count: ☐

Field: Invoice Voucher

Type: DistinctCount

Format String: #,#

Null Processing: Automatic

Description:

The Null Processing property specifies which action Analysis Services takes when it encounters a Null value.

Automatic (Default): Null values are treated as ZeroOrBlank

Error: Null value is illegal

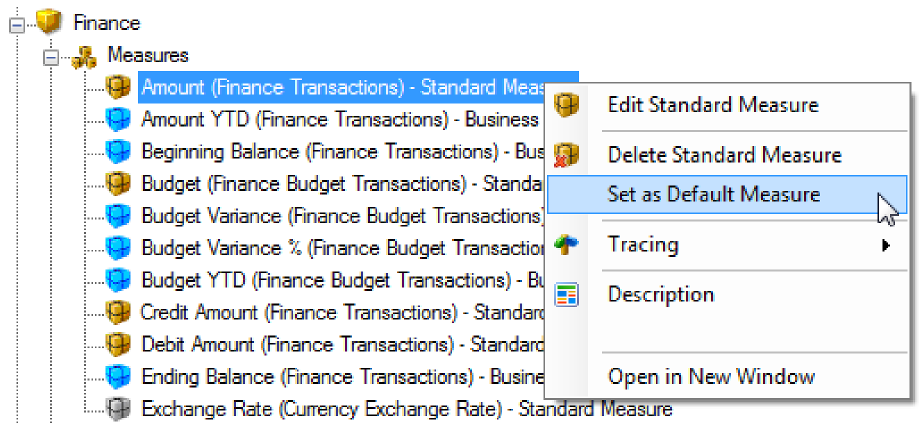
Preserve: Specifies that the Null value is preserved

ZeroOrBlank: Specifies that the Null value is converted to zero (for numeric data types) or blank string (for string data types).

Default Measure on Cubes

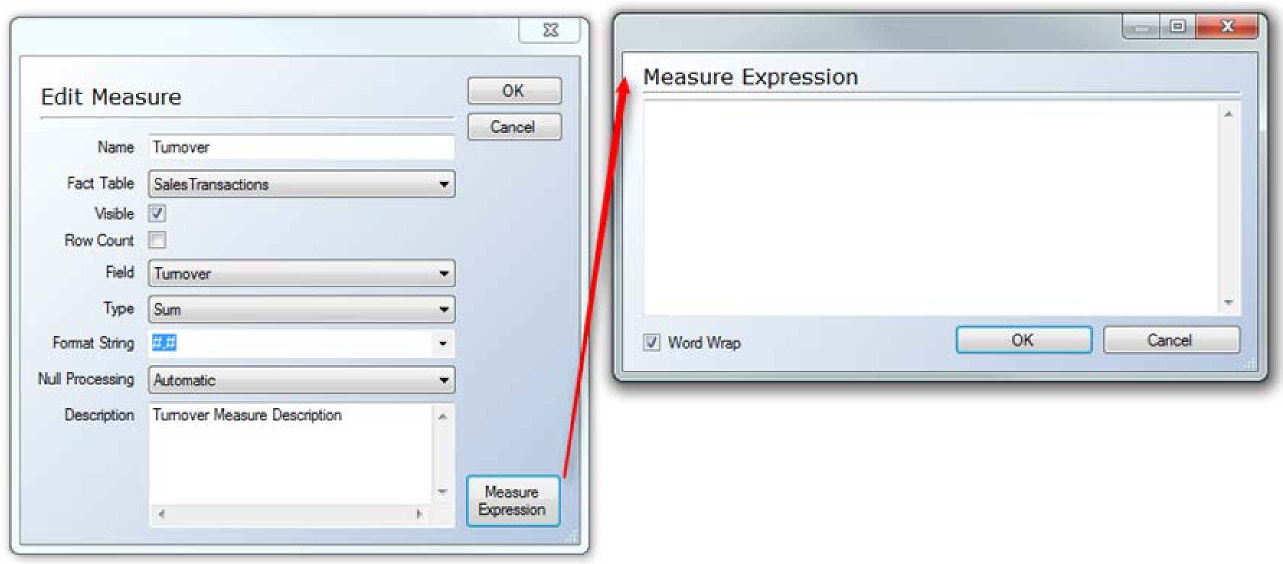
The default cube measure is used every time you drag and drop a dimension member in your browser/excel without using a measure – this can result in bad performance if the wrong default measure is chosen.

The default measure is a toggle function and can be applied to 1 of the standard or derived measures in the cube.


Finance
Measures
Amount (Finance Transactions) - Standard Meas
Amount YTD (Finance Transactions) - Business
Beginning Balance (Finance Transactions) - Bus
Budget (Finance Budget Transactions) - Standa
Budget Variance (Finance Budget Transactions)
Budget Variance % (Finance Budget Transactio
Budget YTD (Finance Budget Transactions) - Bu
Credit Amount (Finance Transactions) - Standard
Debit Amount (Finance Transactions) - Standard
Ending Balance (Finance Transactions) - Busine
Exchange Rate (Currency Exchange Rate) - Standard Measure
Edit Standard Measure
Delete Standard Measure
Set as Default Measure
Tracing
Description
Open in New Window

Measure Expression

It is possible to modify the Member Expression Property on a Measure:

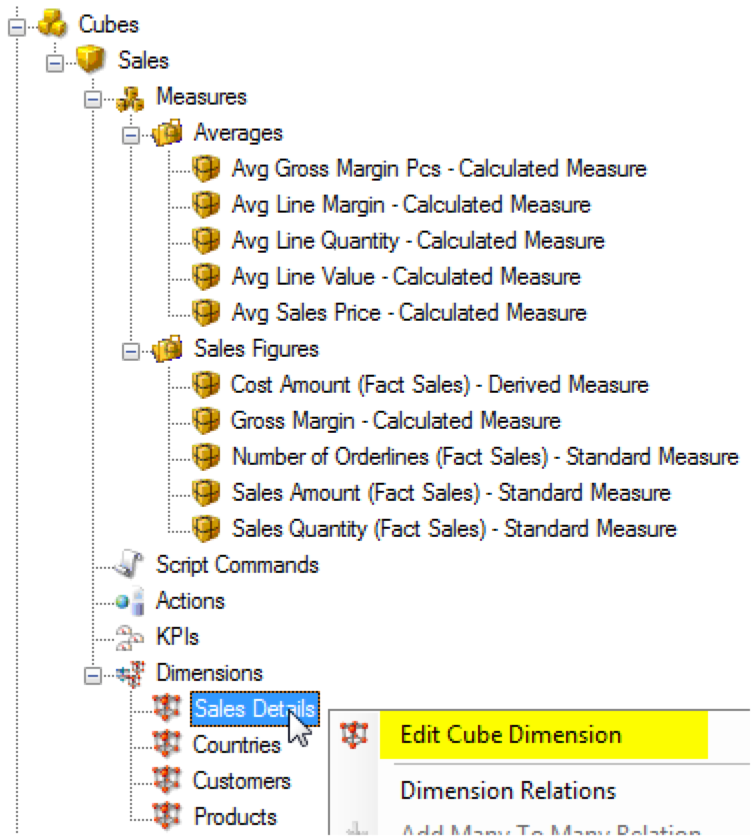

Edit Measure

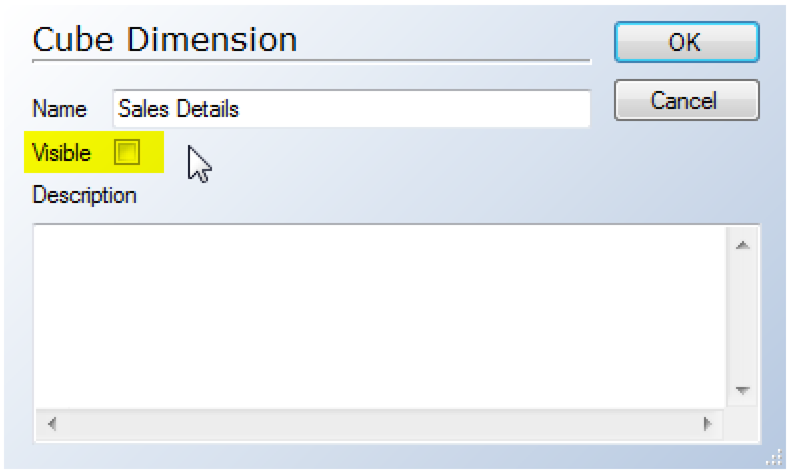
Name: Turnover
Fact Table: SalesTransactions
Visible: ✓
Row Count:
Field: Turnover
Type: Sum
Format String: #,#
Null Processing: Automatic
Description: Turnover Measure Description

Be aware that Measure Expressions are evaluated at the Leaf Level for every dimension which can lead to very bad cube performance. Consider using a Scoped assignment instead and only use the Measure Expression for specific features where no other solution exists.

Hiding a cube dimension

It is possible to hide a cube dimension on a cube. This can be useful if there is a very detailed dimension on a cube and you only want to include the details in a drill through action or if the dimension is used in a many to many relationship and should not be visible to the user.


Cubes
Sales
Measures
Averages
Avg Gross Margin Pcs - Calculated Measure
Avg Line Margin - Calculated Measure
Avg Line Quantity - Calculated Measure
Avg Line Value - Calculated Measure
Avg Sales Price - Calculated Measure
Sales Figures
Cost Amount (Fact Sales) - Derived Measure
Gross Margin - Calculated Measure
Number of Orderlines (Fact Sales) - Standard Measure
Sales Amount (Fact Sales) - Standard Measure
Sales Quantity (Fact Sales) - Standard Measure
Script Commands
Actions
KPIs
Dimensions
Sales Details
Countries
Customers
Products
Edit Cube Dimension
Dimension Relations

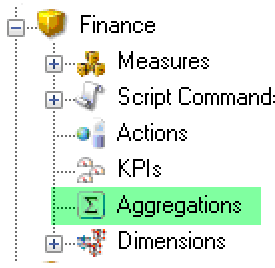


Name: Sales Details
Visible: ☐

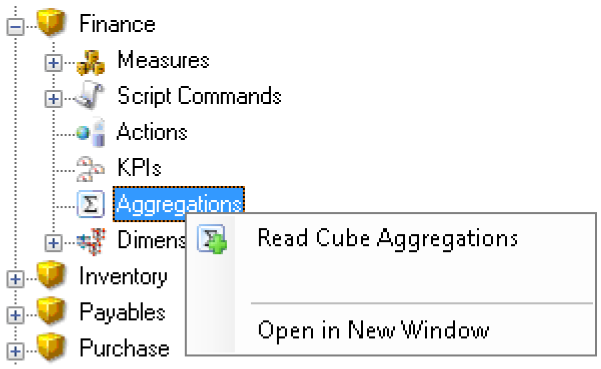
Aggregations on cubes

This feature does not provide a place to design aggregations in TX2014. The feature is can extract and store aggregations designed in an aggregation design tool. TX2014 reads the definition of the aggregation and stores it in the project. During deployment of the cube, the aggregations will be applied to the cube.

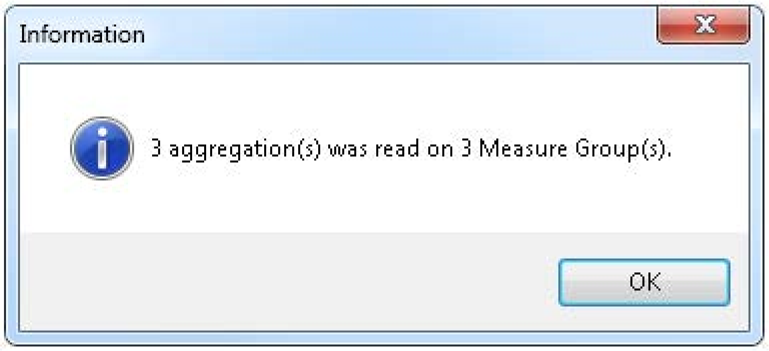
Under the node of each cube, you will find the Aggregations node:



When right clicking the node, you will get a menu called Read Cube Aggregations:

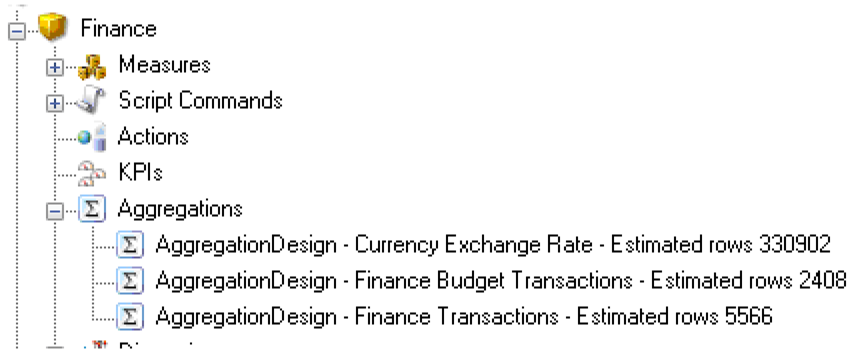


When selecting the Read Cube Aggregation, TX2014 will read the aggregation on the cube and report the numbers of aggregation that was read:

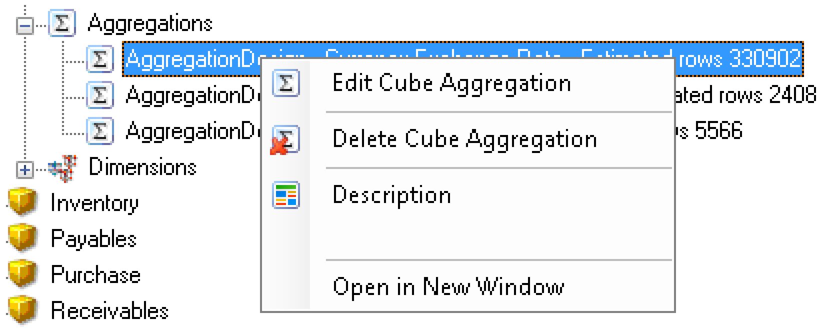


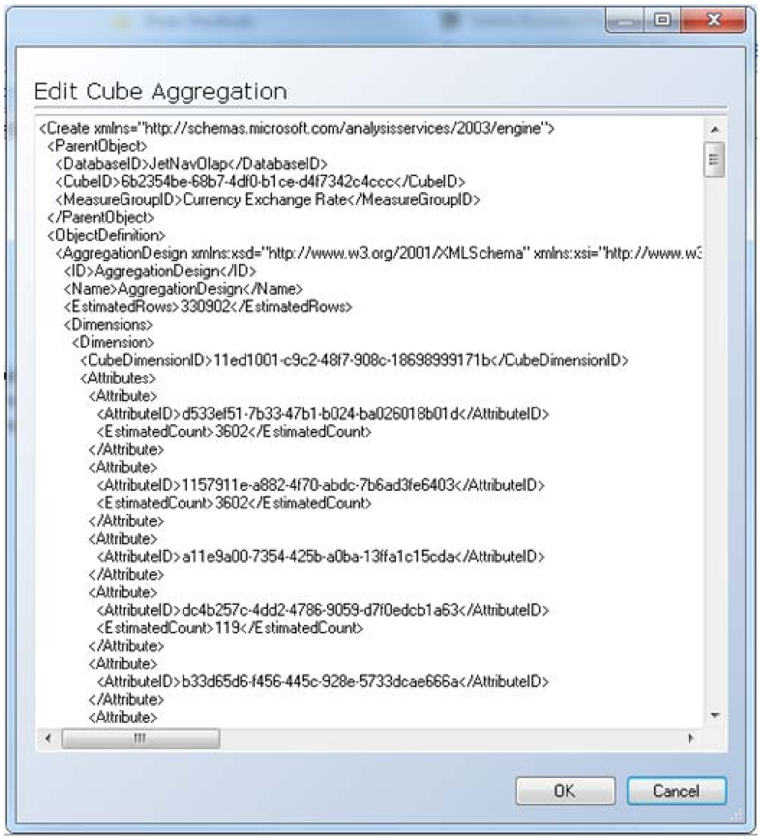
3 aggregation(s) was read on 3 Measure Group(s).

The aggregation will be listed under the Aggregations node:


Finance
Measures
Script Commands
Actions
KPIs
Aggregations
AggregationDesign - Currency Exchange Rate - Estimated rows 330902
AggregationDesign - Finance Budget Transactions - Estimated rows 2408
AggregationDesign - Finance Transactions - Estimated rows 5566

The aggregations are stored in an xml structure which can be edited in a simple text editor in TX2014. Right click the aggregation you want edit and select Edit Cube Aggregation.





Edit Cube Aggregation

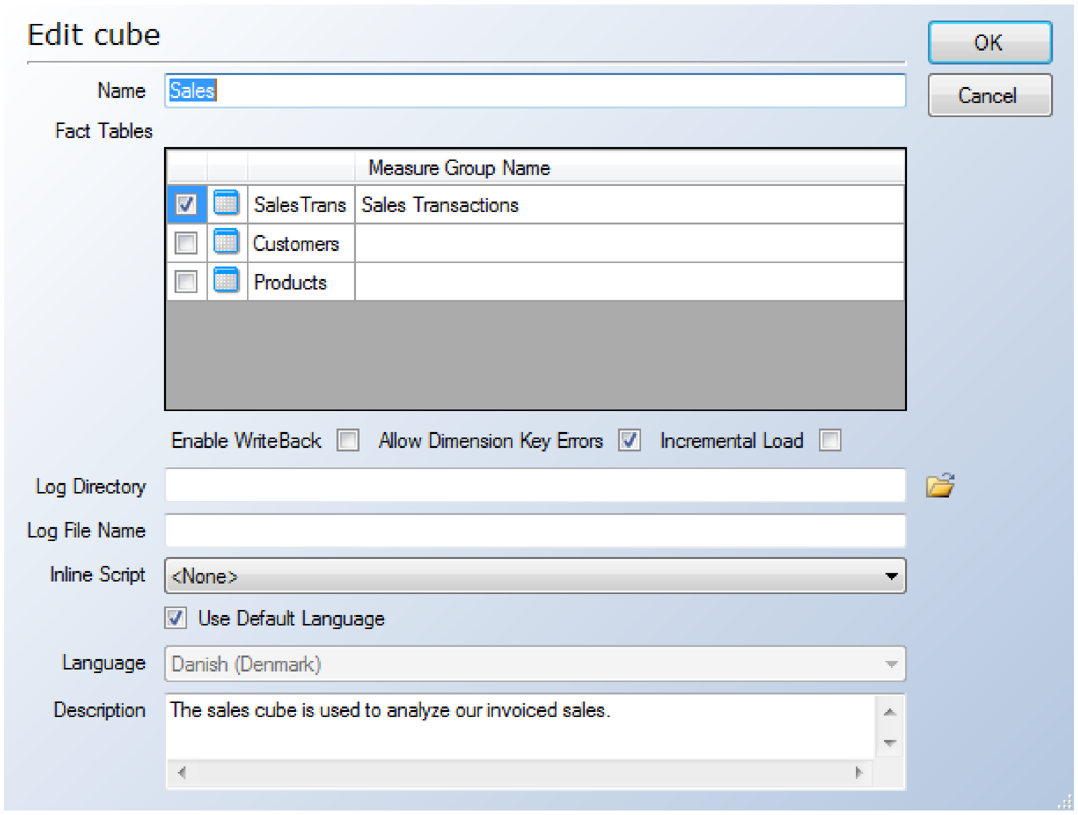
```
<Create xmlns="http://schemas.microsoft.com/analysisservices/2003/engine">
 <ParentObject>
  <DatabaseID>JetNavOlap</DatabaseID>
  <CubeID>6b2354be-68b7-4df0-b1ce-d4f7342c4ccc</CubeID>
  <MeasureGroupID>Currency Exchange Rate</MeasureGroupID>
 </ParentObject>
 <ObjectDefinition>
  <AggregationDesign xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3
   <ID>AggregationDesign</ID>
   <Name>AggregationDesign</Name>
   <EstimatedRows>330902</EstimatedRows>
   <Dimensions>
    <Dimension>
     <CubeDimensionID>11ed1001-c9c2-48f7-908c-18698999171b</CubeDimensionID>
     <Attributes>
      <Attribute>
       <AttributeID>d533ef51-7b33-47b1-b024-ba026018b01d</AttributeID>
       <EstimatedCount>3602</EstimatedCount>
      </Attribute>
      <Attribute>
       <AttributeID>1157911e-a882-4f70-abdc-7b6ad3fe6403</AttributeID>
       <EstimatedCount>3602</EstimatedCount>
      </Attribute>
      <Attribute>
       <AttributeID>a11e9a00-7354-425b-a0ba-13ffa1c15cda</AttributeID>
      </Attribute>
      <Attribute>
       <AttributeID>dc4b257c-4dd2-4786-9059-d7f0edcb1a63</AttributeID>
       <EstimatedCount>119</EstimatedCount>
      </Attribute>
      <Attribute>
       <AttributeID>b33d65d6-f456-445c-928e-5733dcae666a</AttributeID>
      </Attribute>
      <Attribute>
```

Descriptions on Cubes and Measures

In TX2014, it is possible to add descriptions on Dimensions, Dimension Levels, and Dimension Hierarchies. These descriptions are displayed as tool-tips to the end-users that are browsing the analysis database (Front-end dependent).

It is also possible to add descriptions to Cubes, Cube Dimensions, and Measures.

Descriptions on Cubes:



Name: Sales

Fact Tables

			Measure Group Name
✓	▦	SalesTrans	Sales Transactions
☐	▦	Customers	
☐	▦	Products	

Enable WriteBack ☐ Allow Dimension Key Errors ✓ Incremental Load ☐

Log Directory:

Log File Name:

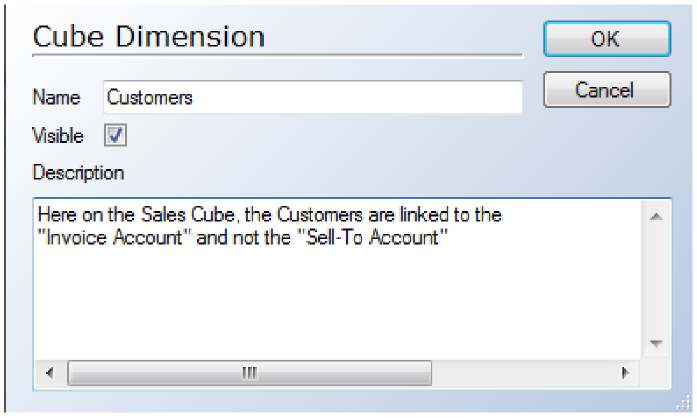
Inline Script: <None>

✓ Use Default Language

Language: Danish (Denmark)

Description: The sales cube is used to analyze our invoiced sales.

Description on Cube Dimensions:



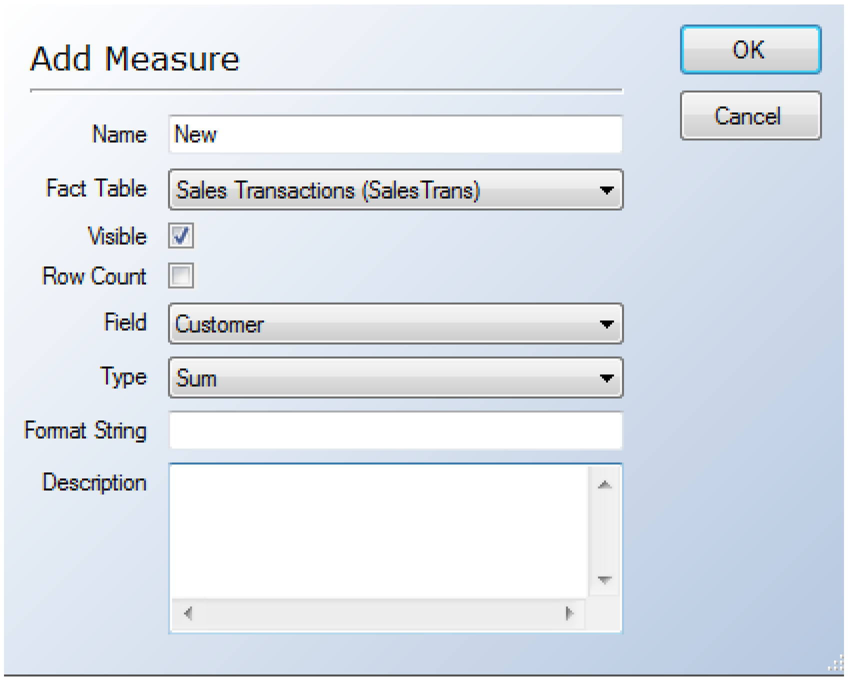
Name: Customers

Visible ✓

Description

Here on the Sales Cube, the Customers are linked to the "Invoice Account" and not the "Sell-To Account"

Description on Measures:


Add Measure

Name: New

Fact Table: Sales Transactions (SalesTrans)

Visible: ✓

Row Count: ☐

Field: Customer

Type: Sum

Format String:

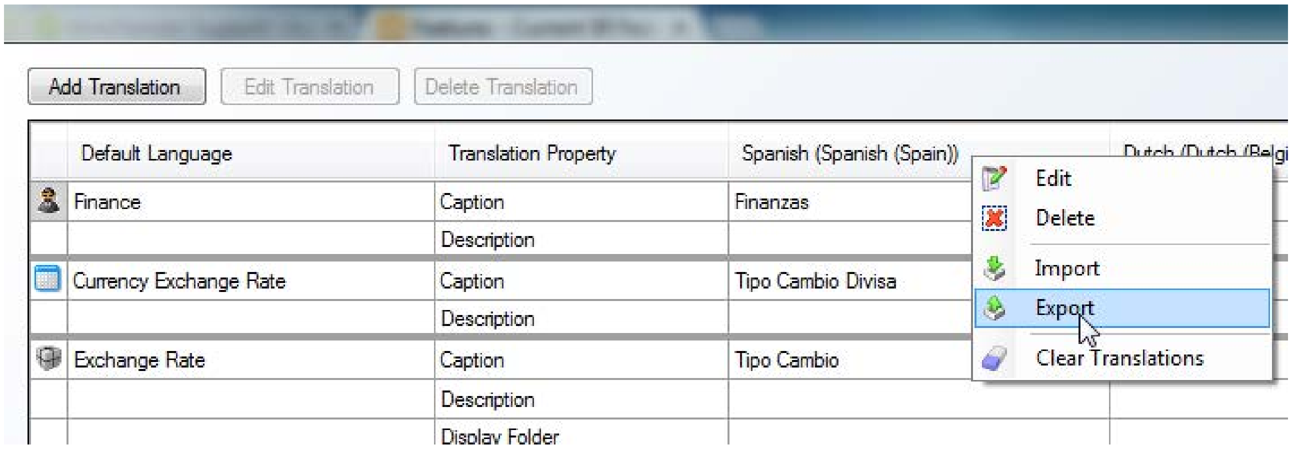
Description:

There are similar Description fields on Derived, Calculated, and Business Function based measures.

Export and Import of OLAP Translations

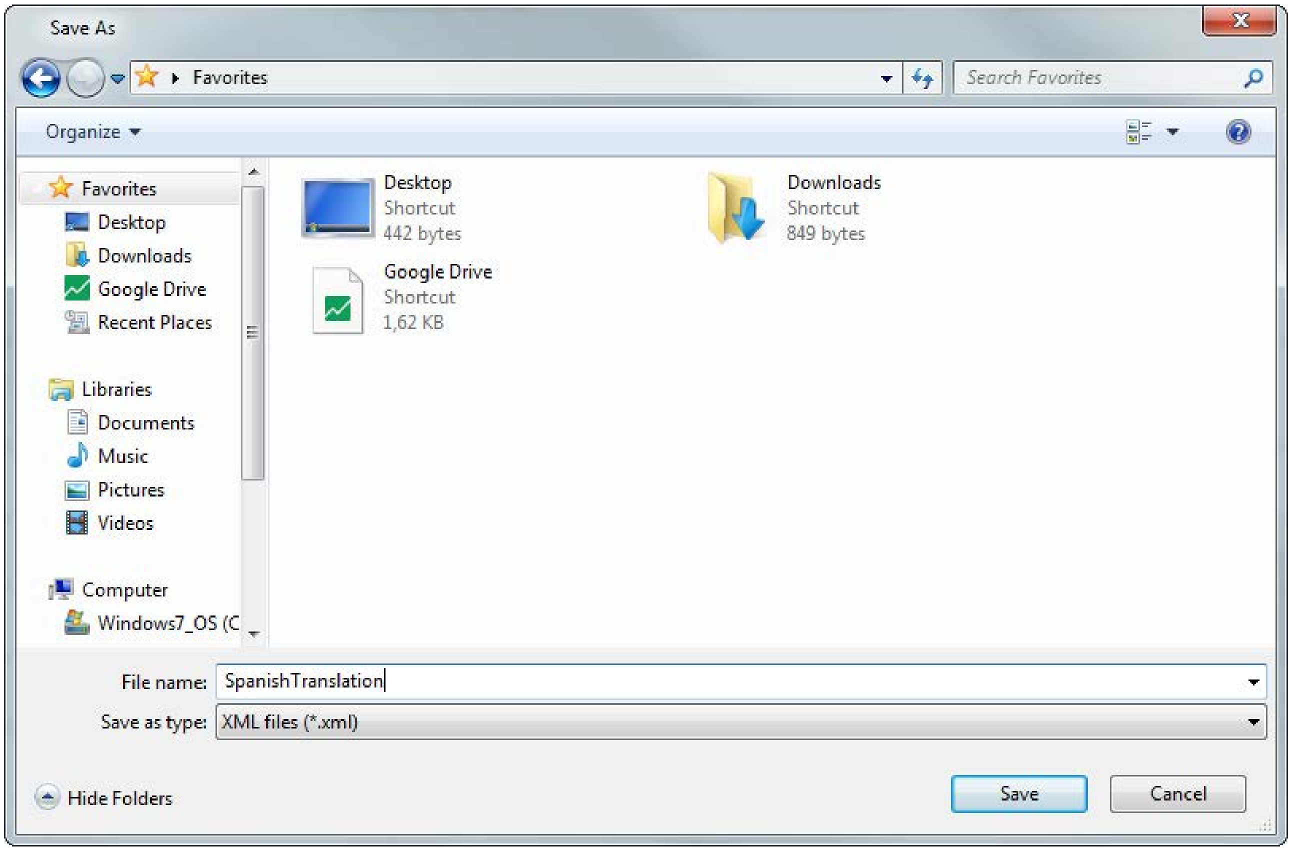
OLAP Translations can be exported and imported. The translations can be imported to a new language in the same projects or can be imported in another project.

To Export the translations, right click the translation you want to export, and select Export:



	Default Language	Translation Property	Spanish (Spanish (Spain))	Dutch (Dutch (Belgi
	Finance	Caption	Finanzas	
		Description		
	Currency Exchange Rate	Caption	Tipo Cambio Divisa	
		Description		
	Exchange Rate	Caption	Tipo Cambio	
		Description		
		Display Folder		

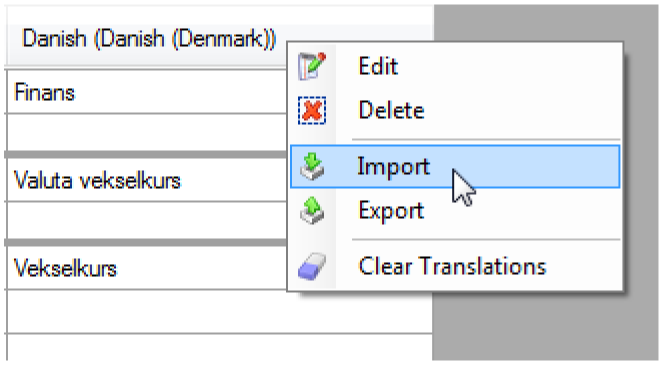
Enter file location and name, and click Save:



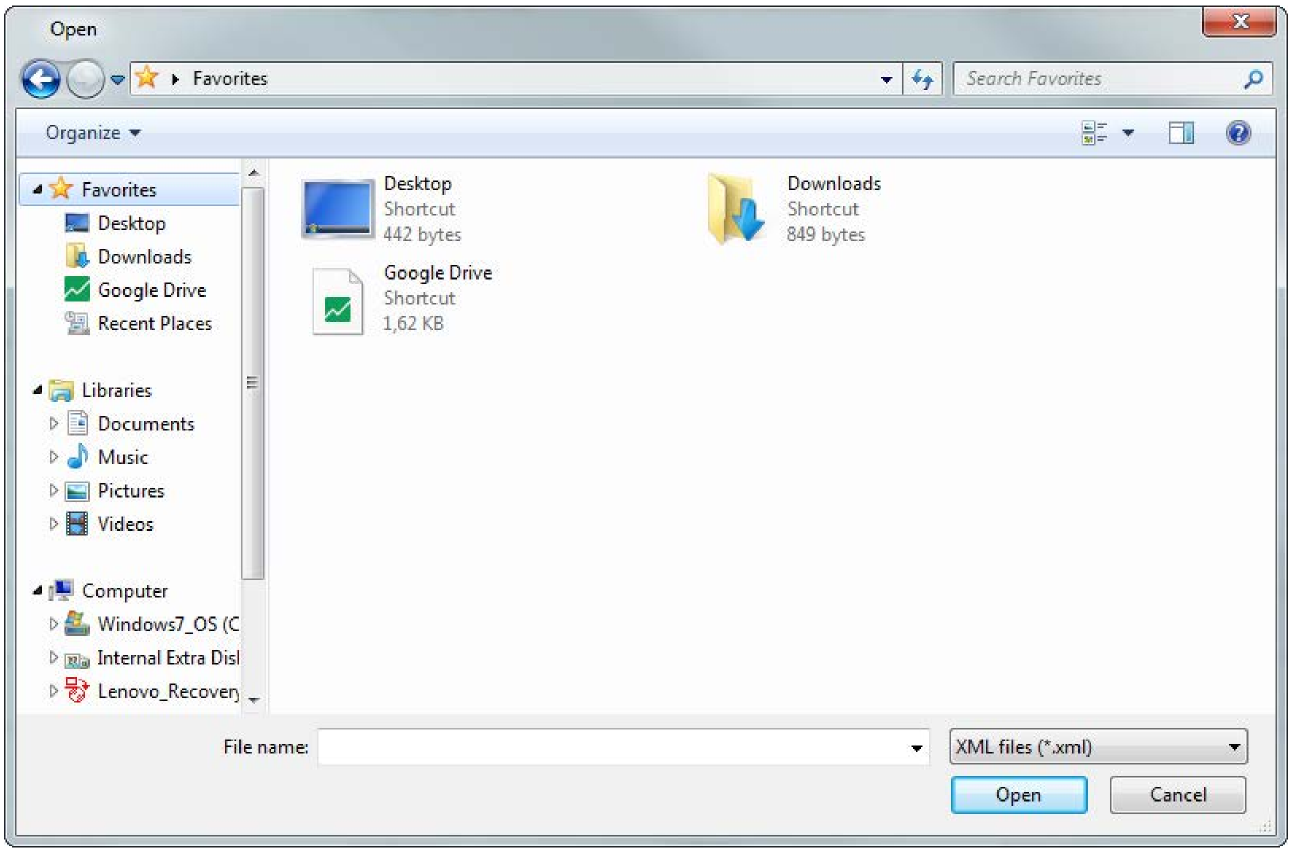
File name: SpanishTranslation

Save as type: XML files (*.xml)

To import a translation, select “Import” on the right click menu on the language you want to import:



Select the file and click Open:



Desktop
Shortcut
442 bytes

Downloads
Shortcut
849 bytes

Google Drive
Shortcut
1,62 KB

File name:

XML files (*.xml)

OLAP Security

Since timeXtender 4.5, we have introduced some major changes to the OLAP Security which will be described in this document along with the general feature documentation.

The latest changes include:

- Define Roles and add users/group to these instead of each user/group becoming a separate role.
- Use Denied or Allowed member sets for dimensions.
- Combine Allowed and Denied Membership on a dimension for a specific cube.
- Support for multiple domains.
- Support for environment specific roles.

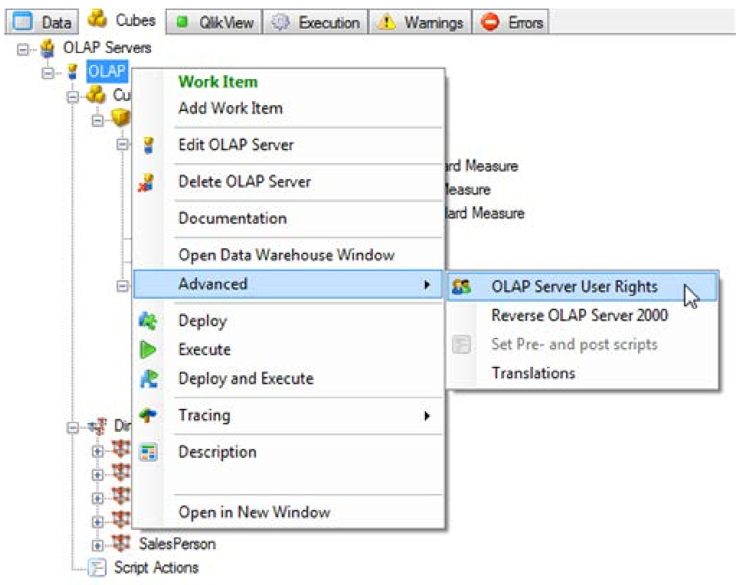
Setting up OLAP Database Security in TX2014

Adding roles

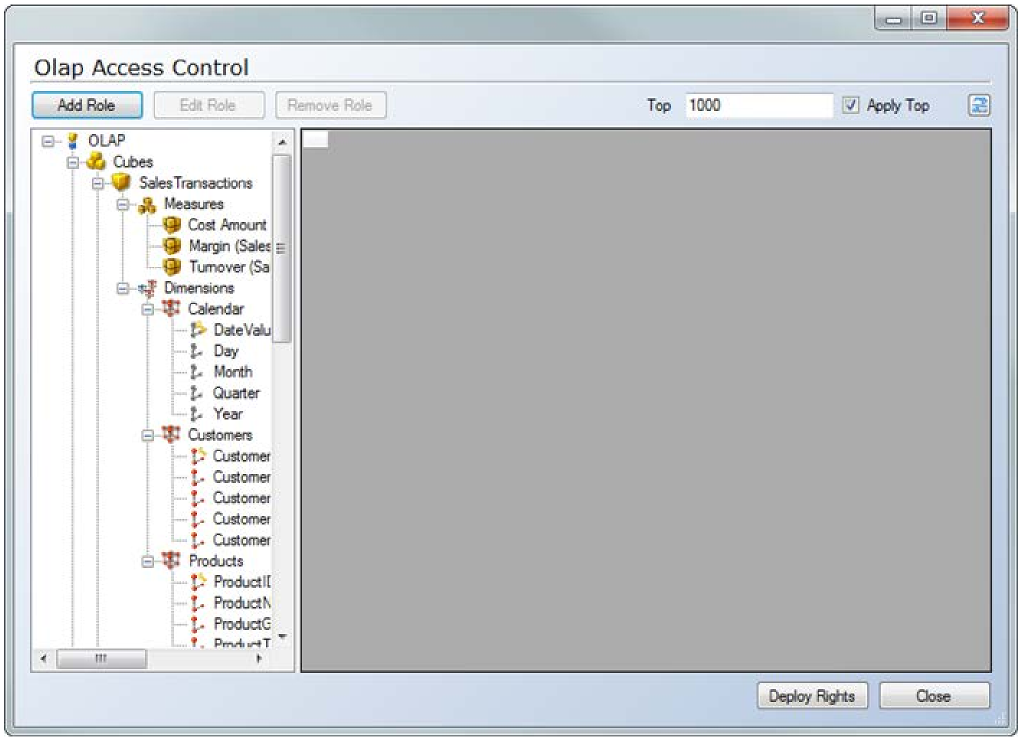
Please note that any user who has Administrator rights on the OLAP server will be able to see anything, no matter the rights you set up. It is a requirement for the users running timeXtender that they have this role.

To set up security in timeXtender, go to the Cubes tab, right click your OLAP server, and select ‘OLAP Server User Rights’

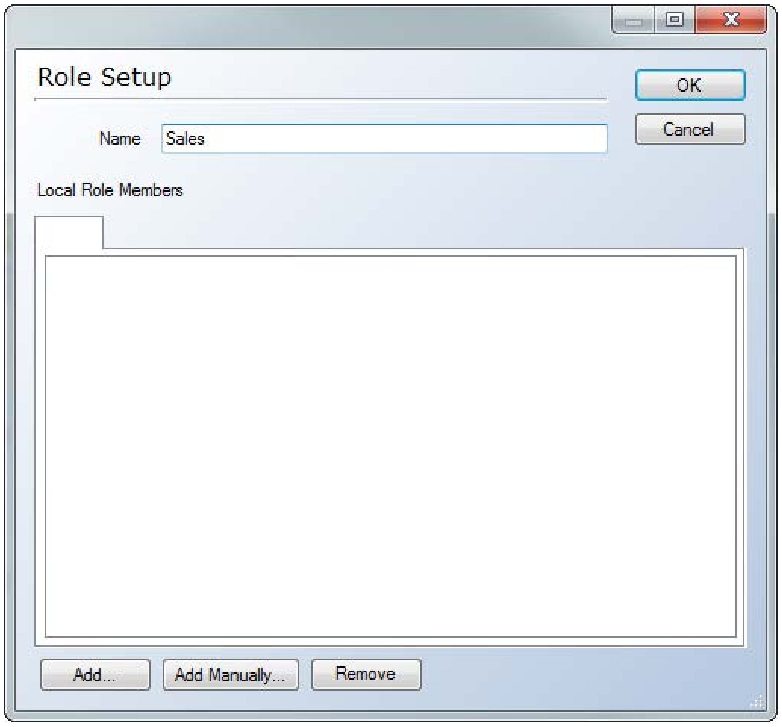
To be able to set up the rights, the cube must have been deployed and executed since the last changes have been made:



In the dialog that opens, you can set up rights for every object and value in the cubes:



To do so, click ‘Add Role,’ and a new dialog will appear from where you can assign a name for the security role:

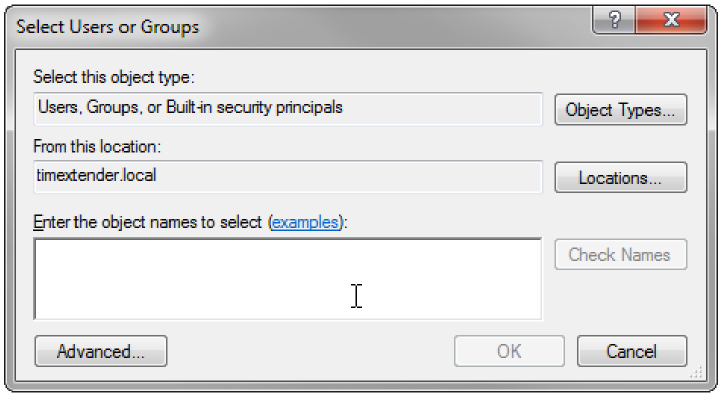


Name: Sales

Local Role Members

You can add users/groups to the role in two ways; either by picking the user/group from the standard windows user/group picker or by entering the users manually:

Click the “Add…” button to pick users/groups:

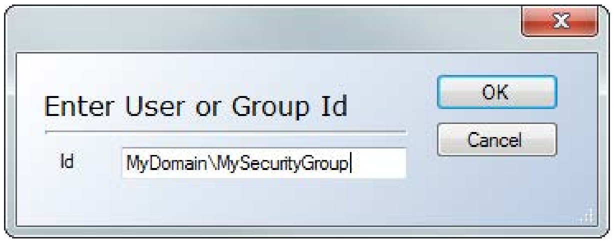


Select this object type: Users, Groups, or Built-in security principals

From this location: timextender.local

Enter the object names to select (examples):

Click the ”Add Manually...” to enter users/groups manually:

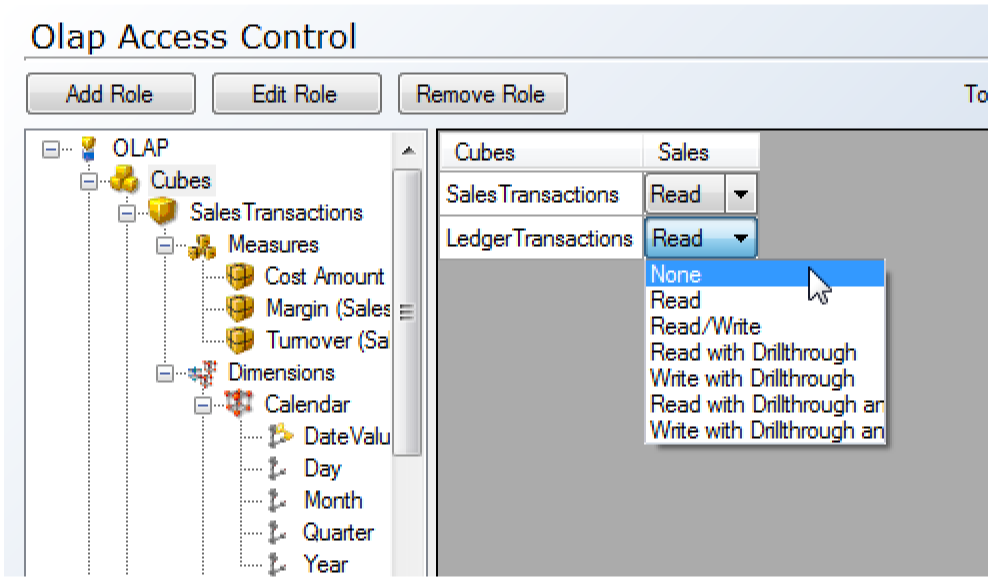


MyDomain\MySecurityGroup

When you add a role, all members of the role will have read access to everything. You can then defer from that setting on any single object in the tree.

On a cube you have the following options:

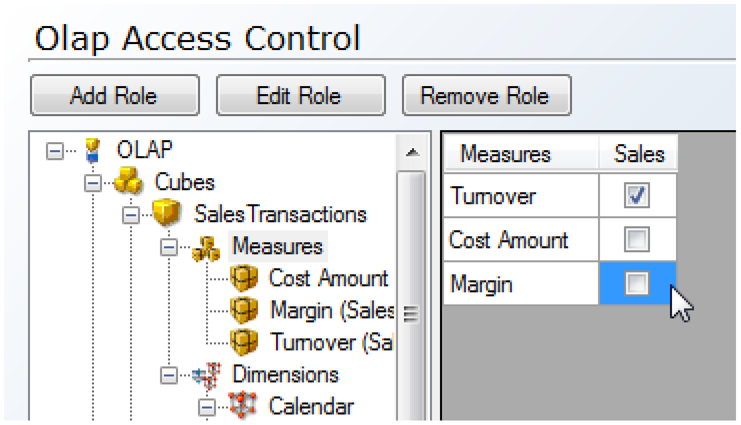
None	No rights on the cube
Read	Read rights on the cube, the cube can be browsed
Read/Write	Read rights, and the users can perform writeback to the cube
Read with Drillthrough	Read access, and on standard measures the user can drillthrough and see the data warehouse records that the value consists of.
Write with Drillthrough	Same as above, with writeback support
Read with drillthrough and Local Cube	Gives the additional ability to save a local copy of the cube.
Write with drillthrough and Local Cube	Same as above, with writeback



Cubes	Sales
SalesTransactions	Read
LedgerTransactions	Read

The access to measures and dimensions are handled in a similar way. Additionally, it is possible to set up rights for any member of the dimensions; for instance, give certain roles access to specific companies, only rights to see specific performance values, and so on.

In this example, members of the Sales Role will be able to see the Turnover Measure but not the Cost Amount and Margin Measures:



Measures	Sales
Turnover	✓
Cost Amount	☐
Margin	☐

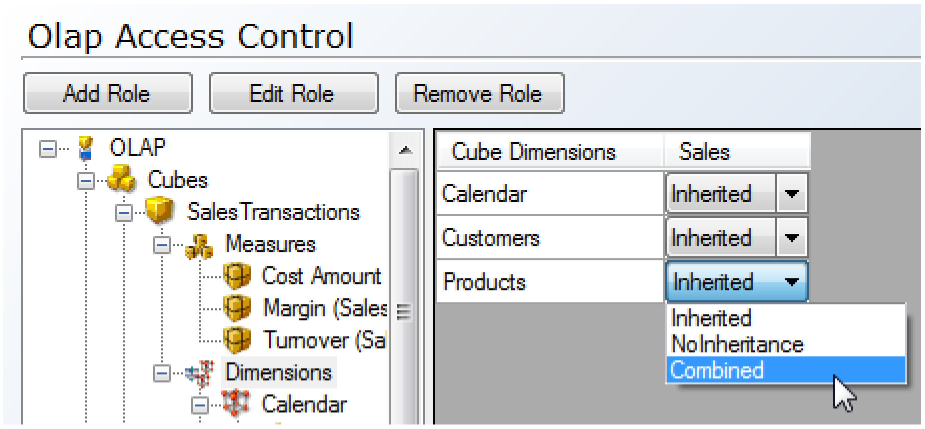
Dimension Security

User permissions on dimension members in tX can be controlled from two different spots in the OLAP Access Control system.

If the permissions are global, meaning they should apply to multiple cubes where the dimension is used, then setup the permission on the dimension itself, and set the permission level to 'Inherited' on the cube. This is the default setting.

If the permissions are specific for a single cube, then define the permissions on the dimension node on the Cube, and set the permission level to 'NonInheritance'.

It is possible to combine the setup on the Global dimension and the local cube dimension. This can be achieved by selecting an allowed member set on the global dimension and setting the local cube dimension to ‘Combined’:



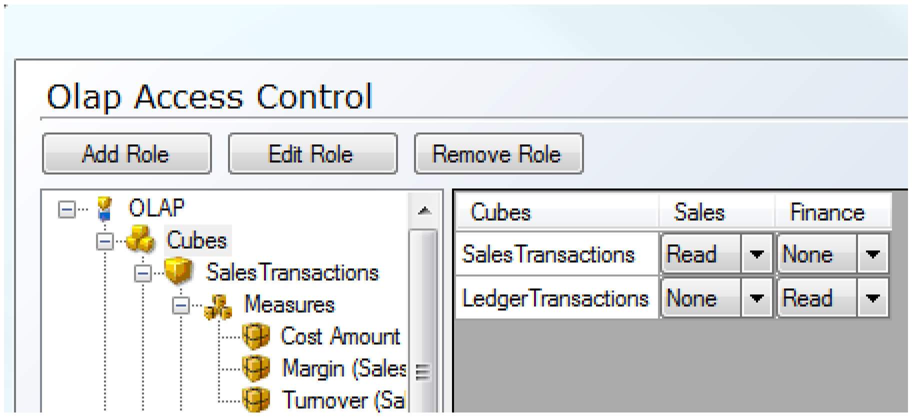
Cube Dimensions	Sales
Calendar	Inherited
Customers	Inherited
Products	Inherited

When using the ‘combined’ setting or when stacking roles (a user being member of multiple groups), please be aware that a “Deny” on a member in dimension security will always have higher priority than an allowed member.

In other words, the new way of handling permissions allows you to set up a default behavior for the dimension that will be inherited on all cubes. You can defer from the default setting by setting the permission level to 'Noninheritance' and setup local permission for the cube/dimension combination. As the third option, you can setup the default setting by setting the permission level to ‘Combined’ and setup additional local permissions for the cube/dimension combination.

Here are a few examples:

In this example, members of the Sales Role will have access to the Sales Transactions Cube, members of the Finance Role will have access to the Finance Transactions Cube. Members of both roles will have access to both roles:



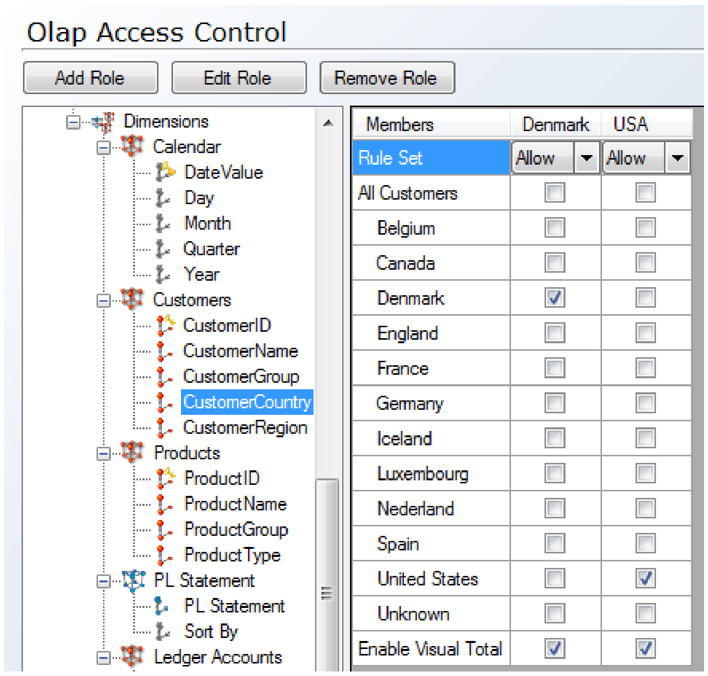
Olap Access Control

Add Role | Edit Role | Remove Role

- OLAP
  - Cubes
    - SalesTransactions
      - Measures
        - Cost Amount
        - Margin (Sales
        - Turnover (Sa

Cubes	Sales	Finance
SalesTransactions	Read	None
LedgerTransactions	None	Read

In this example, members of the Denmark Role will be able to see customers from Denmark, while members of the USA Role will be able to see Customers from United States. Members of both roles will be able to see Customers from Denmark AND United States:



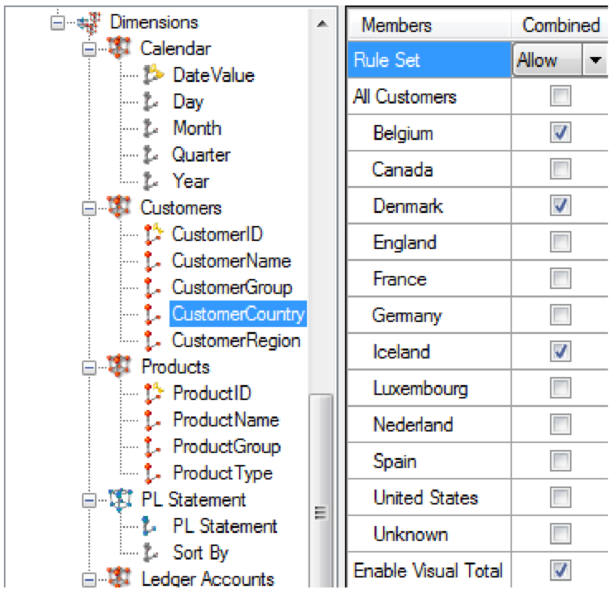
Olap Access Control

Add Role | Edit Role | Remove Role

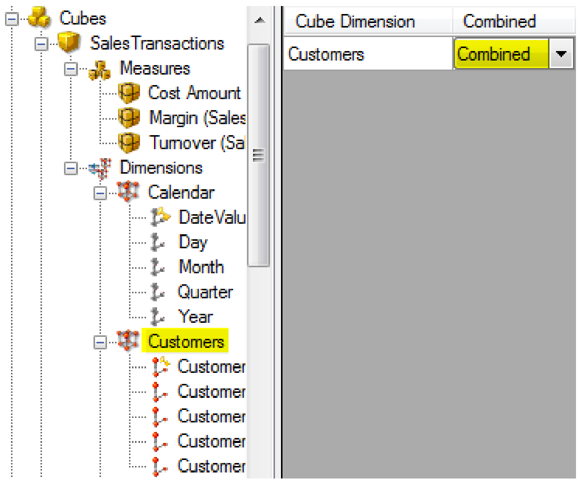
- Dimensions
  - Calendar
    - DateValue
    - Day
    - Month
    - Quarter
    - Year
  - Customers
    - CustomerID
    - CustomerName
    - CustomerGroup
    - CustomerCountry
    - CustomerRegion
  - Products
    - ProductID
    - ProductName
    - ProductGroup
    - ProductType
  - PL Statement
    - PL Statement
    - Sort By
  - Ledger Accounts

Members	Denmark	USA
Rule Set	Allow	Allow
All Customers	☐	☐
Belgium	☐	☐
Canada	☐	☐
Denmark	✓	☐
England	☐	☐
France	☐	☐
Germany	☐	☐
Iceland	☐	☐
Luxembourg	☐	☐
Nederland	☐	☐
Spain	☐	☐
United States	☐	✓
Unknown	☐	☐
Enable Visual Total	✓	✓

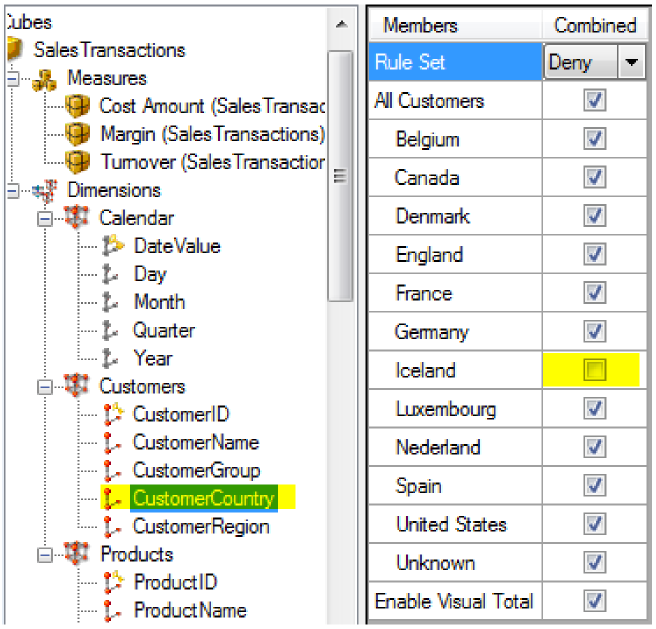
In this example, members of the Combined Role can see Customers from Belgium, Denmark, and Iceland – except on the SalesTransactions Cube where they can not see the Customers from Iceland:



Members	Combined
Rule Set	Allow
All Customers	☐
Belgium	✓
Canada	☐
Denmark	✓
England	☐
France	☐
Germany	☐
Iceland	✓
Luxembourg	☐
Nederland	☐
Spain	☐
United States	☐
Unknown	☐
Enable Visual Total	✓



Cube Dimension	Combined
Customers	Combined



Members	Combined
Rule Set	Deny
All Customers	✓
Belgium	✓
Canada	✓
Denmark	✓
England	✓
France	✓
Germany	✓
Iceland	☐
Luxembourg	✓
Nederland	✓
Spain	✓
United States	✓
Unknown	✓
Enable Visual Total	✓

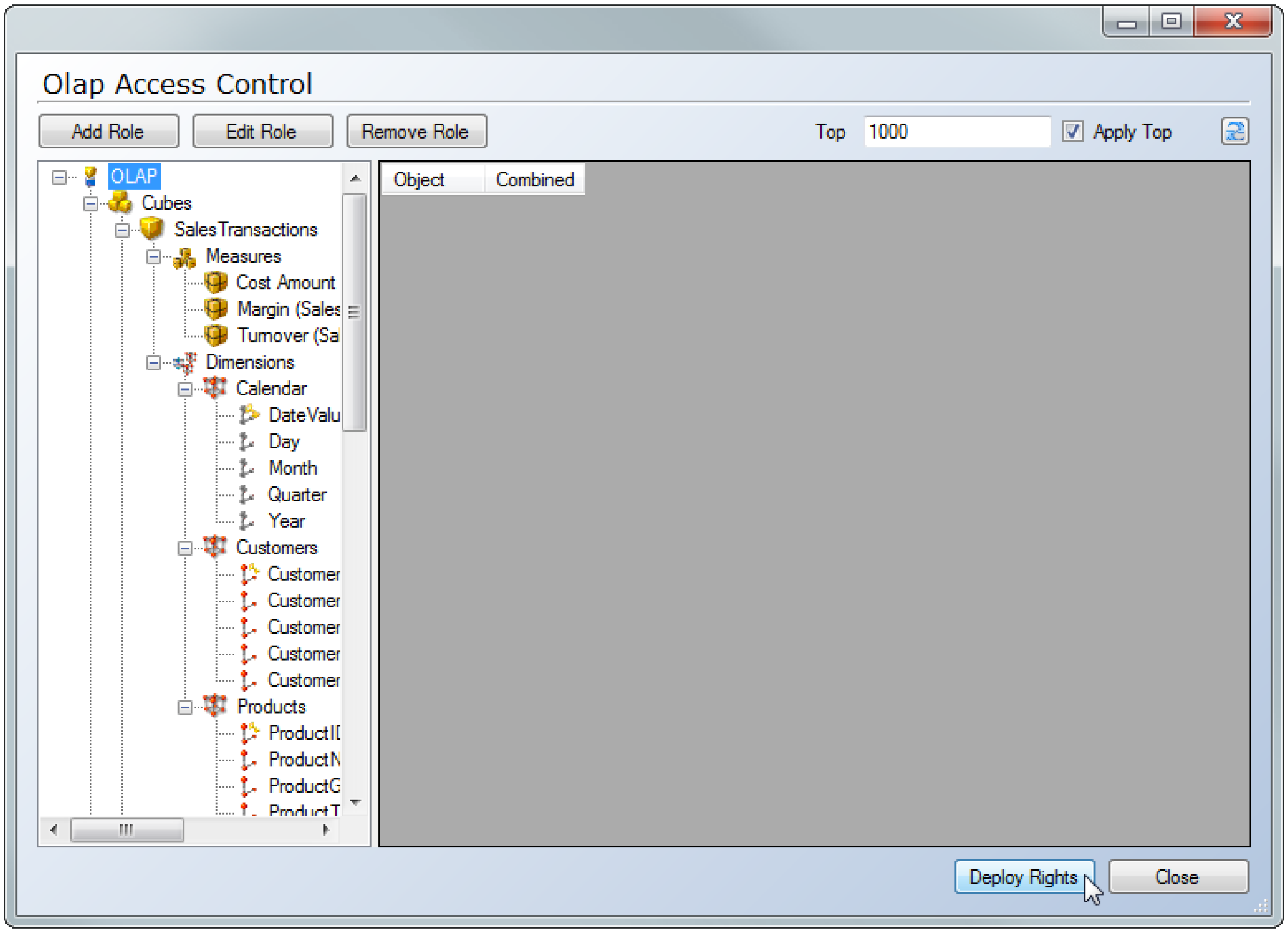
Rule Sets

You can choose between 2 role sets when setting up dimension member security.

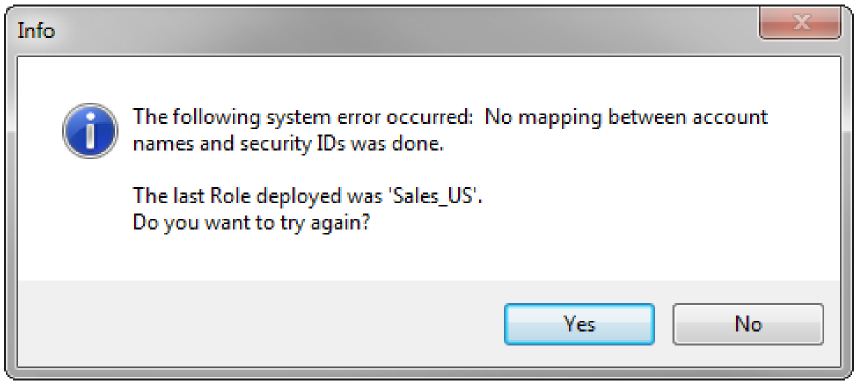
Deny: Manually deselect members to deny. If new members are added to the level, they will by default be allowed.

Allow: Manually select members to allow. If new members are added to the level,they will by default be denied.

To deploy the rights to the OLAP Server, click the ‘Deploy Rights’ button. After this, the rights will be deployed whenever the OLAP Server is deployed.

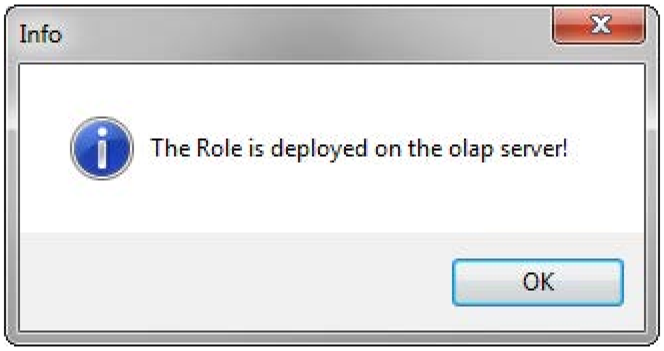


If a user or group does not exist in the active directory, you will receive the following error when deploying rights:


The following system error occurred: No mapping between account names and security IDs was done.

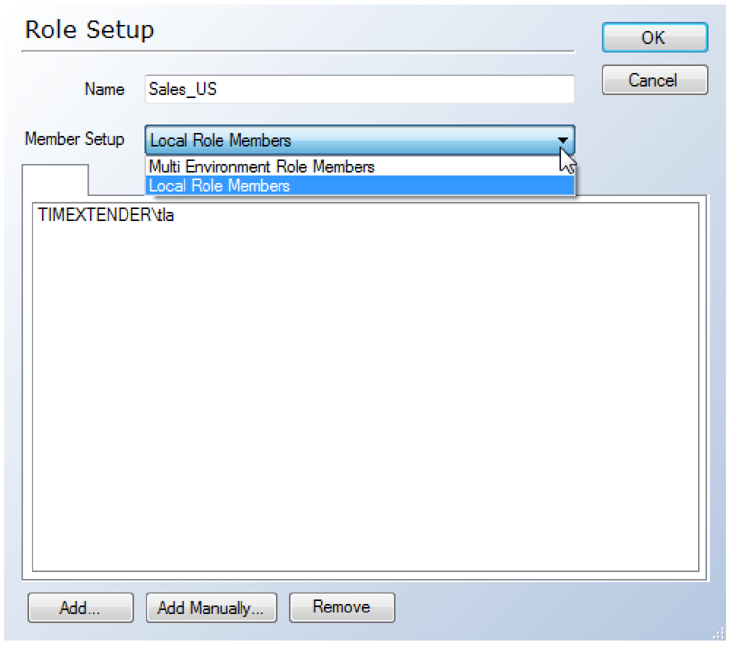
The last Role deployed was 'Sales_US'.
Do you want to try again?

If no error occurs, the following message will appear:


The Role is deployed on the olap server!

Security and Multiple Environments

When using the Multiple Environment Deployment feature of tX, the security setup can be differentiated per environment. When tX detects that it is running in a multiple environment setup, if offers the option to setup users/groups for roles differentiated (Multiple Environment Role Members) or not (Local Role Members):



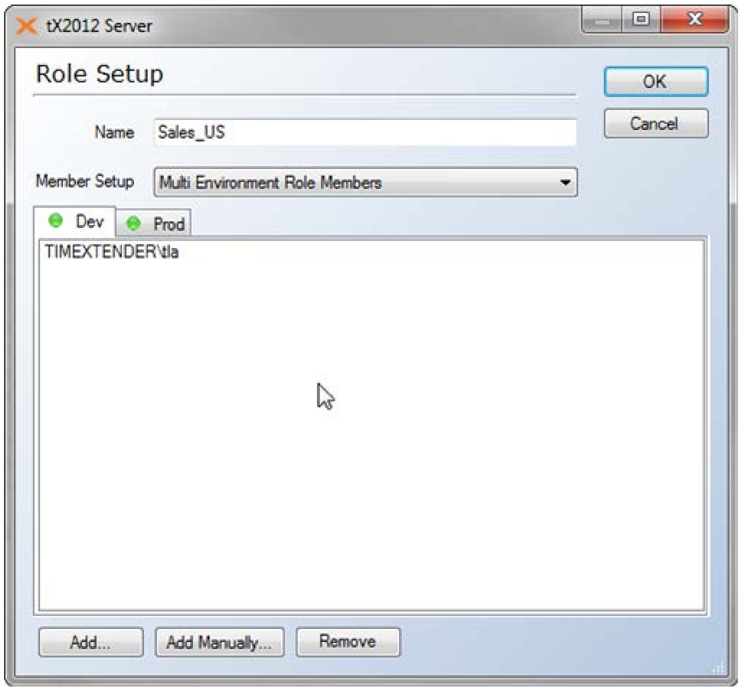
Name: Sales_US

Member Setup: Local Role Members

Multi Environment Role Members
Local Role Members

TIMEXTENDER\tla

If “Multiple Environment Role Members” are selected, then you can select users and groups per environment. The green dot indicates whether the environment can be contacted or not:

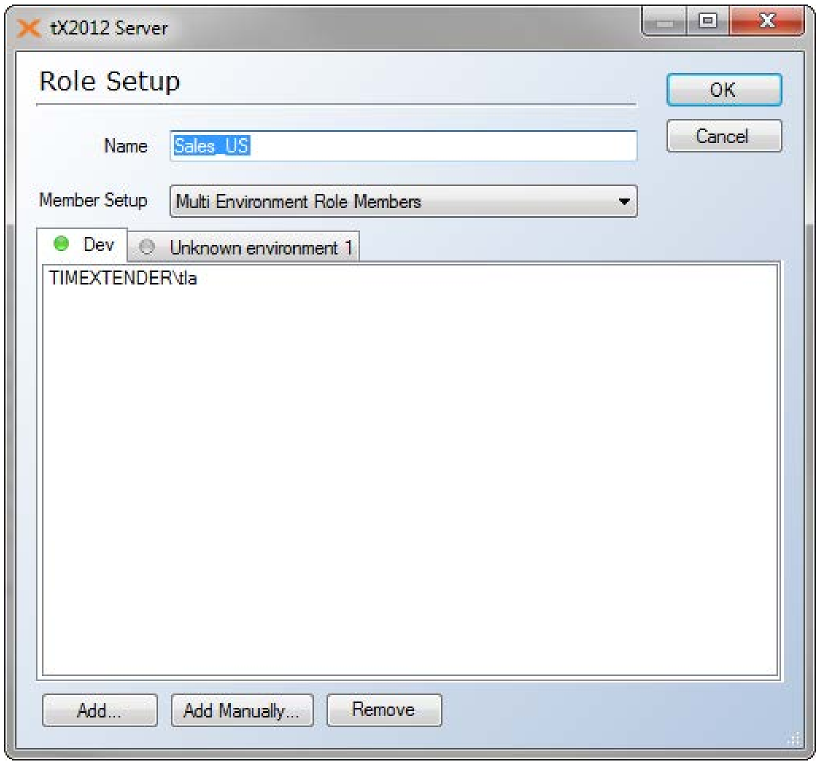


Name: Sales_US

Member Setup: Multi Environment Role Members

TIMEXTENDER\tla

If an environment is deleted, then it will appear as “Unknown Environment”:

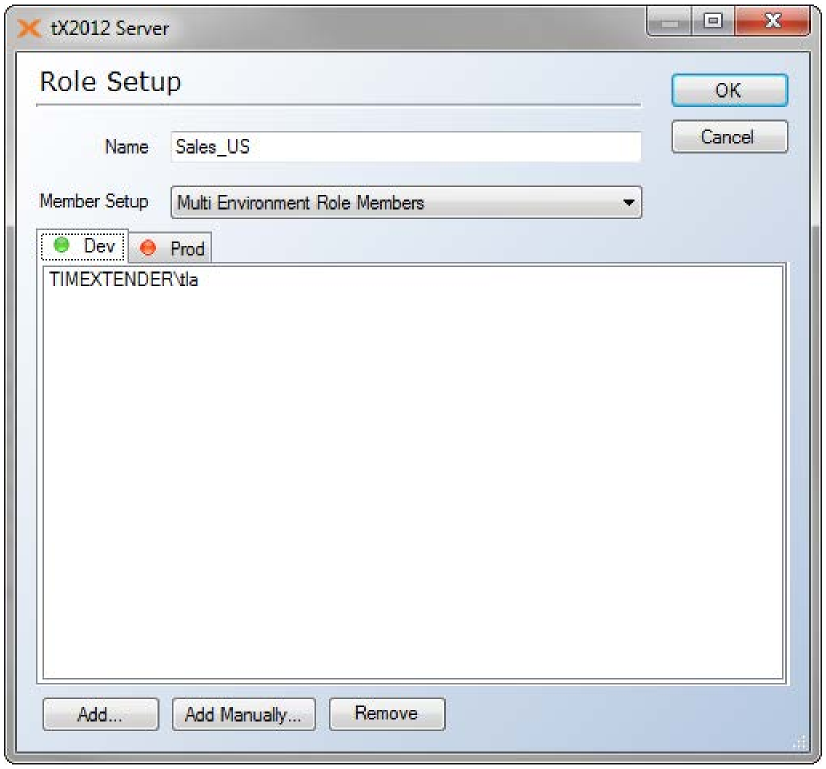


Name: Sales_US

Member Setup: Multi Environment Role Members

TIMEXTENDER\tla

If the environment in known by tX but can not be contacted, then it will appear with a red dot. Users and groups can be added and removed even though the environment cannot be contacted:



Name: Sales_US

Member Setup: Multi Environment Role Members

TIMEXTENDER\tla

SSAS Cube Perspectives

Cubes can be very complex for users to explore.

A single cube can represent the contents of a complete data warehouse with multiple measure groups representing multiple fact tables and multiple dimensions based on multiple dimension tables. Such a cube can be very complex and powerful, but daunting, to users who may only need to interact with a small part of the cube in order to satisfy their business intelligence and reporting requirements.

You can use a perspective to reduce the perceived complexity of a cube. A perspective defines a viewable subset of a cube that provides focused, business-specific or application-specific viewpoints on the cube.

The perspective controls the visibility of objects that are contained by a cube. The following objects can be displayed or hidden in a perspective:

Dimensions

Attributes

Hierarchies

Measure groups

Measures

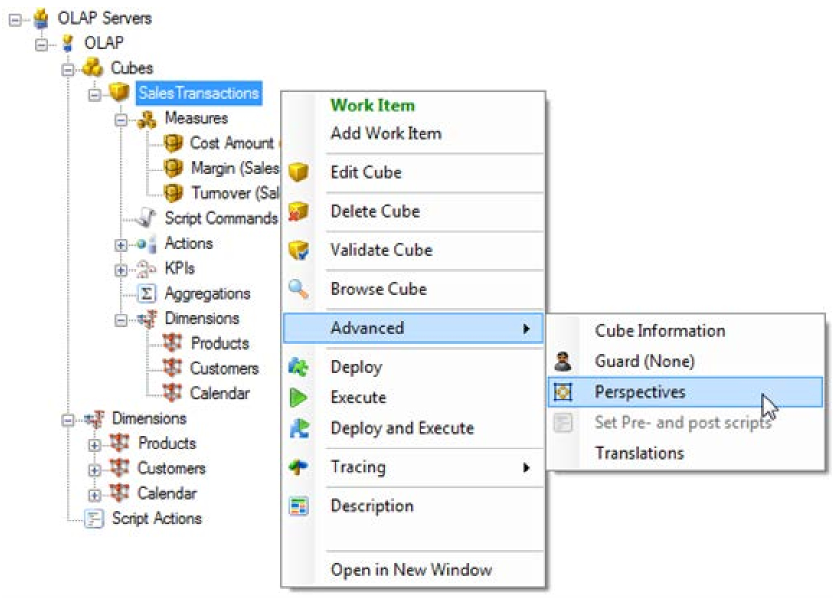
Key Performance Indicators (KPIs)

Actions

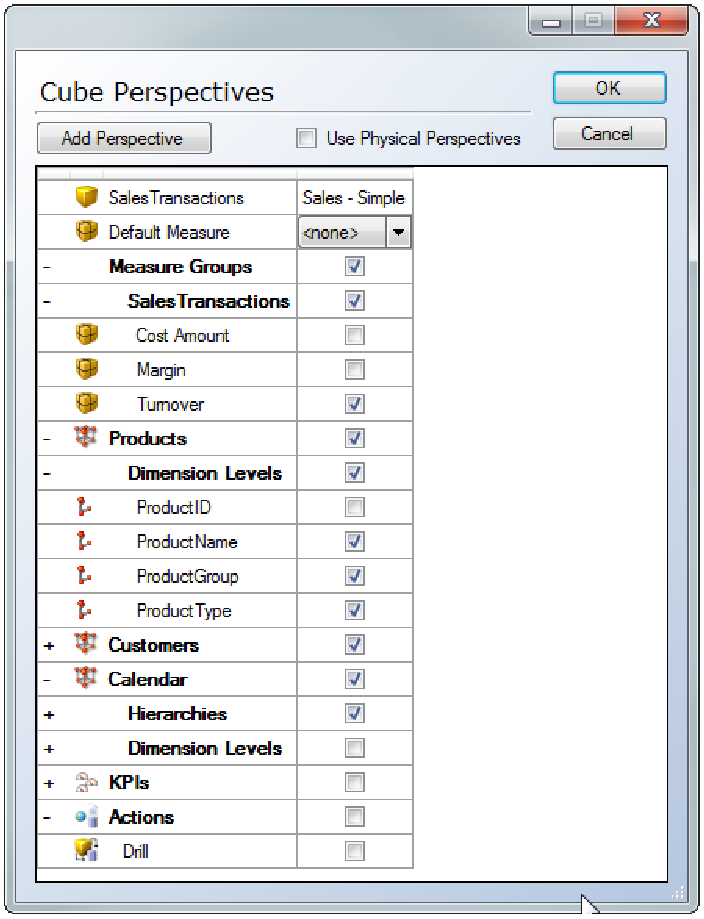
Objects in a cube that are not visible to the user through a perspective can still be directly referenced and retrieved Multidimensional Expressions (MDX). Perspectives do not restrict access to objects in a cube and should not be used as such; instead, perspectives are used to provide a better experience while accessing a cube.

Perspectives are not meant to be used as a security mechanism but as a tool for providing a better user experience in business intelligence applications. All security for a particular perspective is inherited from the underlying cube. For example, perspectives cannot provide access to objects in a cube to which a user does not already have access. - Security for the cube must be resolved before access to objects in the cube can be provided through a perspective.

In TX2014, perspectives can be created and maintained in the Perspectives dialogue on the advanced section of the right-click menu on the cube:



You indicate by a checkmark if the individual object is visible in the perspective or not:

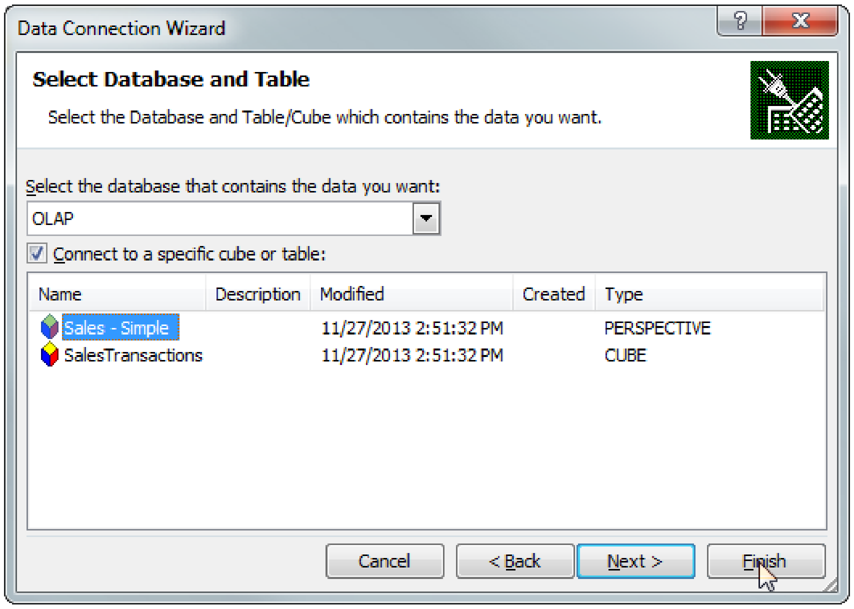


Cube Perspectives

☐ Use Physical Perspectives

	SalesTransactions	Sales - Simple
	Default Measure	<none>
-	**Measure Groups**	✓
-	**SalesTransactions**	✓
	Cost Amount	☐
	Margin	☐
	Turnover	✓
-	**Products**	✓
-	**Dimension Levels**	✓
	ProductID	☐
	ProductName	✓
	ProductGroup	✓
	ProductType	✓
+	**Customers**	✓
-	**Calendar**	✓
+	**Hierarchies**	✓
+	**Dimension Levels**	☐
+	**KPIs**	☐
-	**Actions**	☐
	Drill	☐

When the user connects to the perspective, it is just like connecting to a cube:



Data Connection Wizard

Select Database and Table

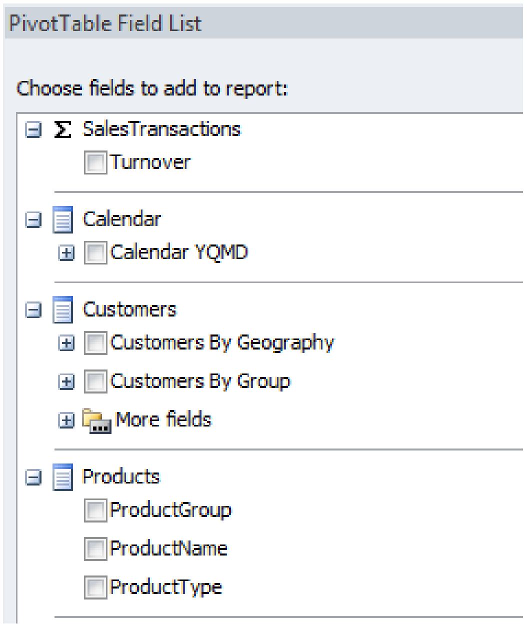
Select the Database and Table/Cube which contains the data you want.

Select the database that contains the data you want: OLAP

✓ Connect to a specific cube or table:

Name	Description	Modified	Created	Type
Sales - Simple		11/27/2013 2:51:32 PM		PERSPECTIVE
SalesTransactions		11/27/2013 2:51:32 PM		CUBE

The end-result as seen in Excel when connecting to the perspective:



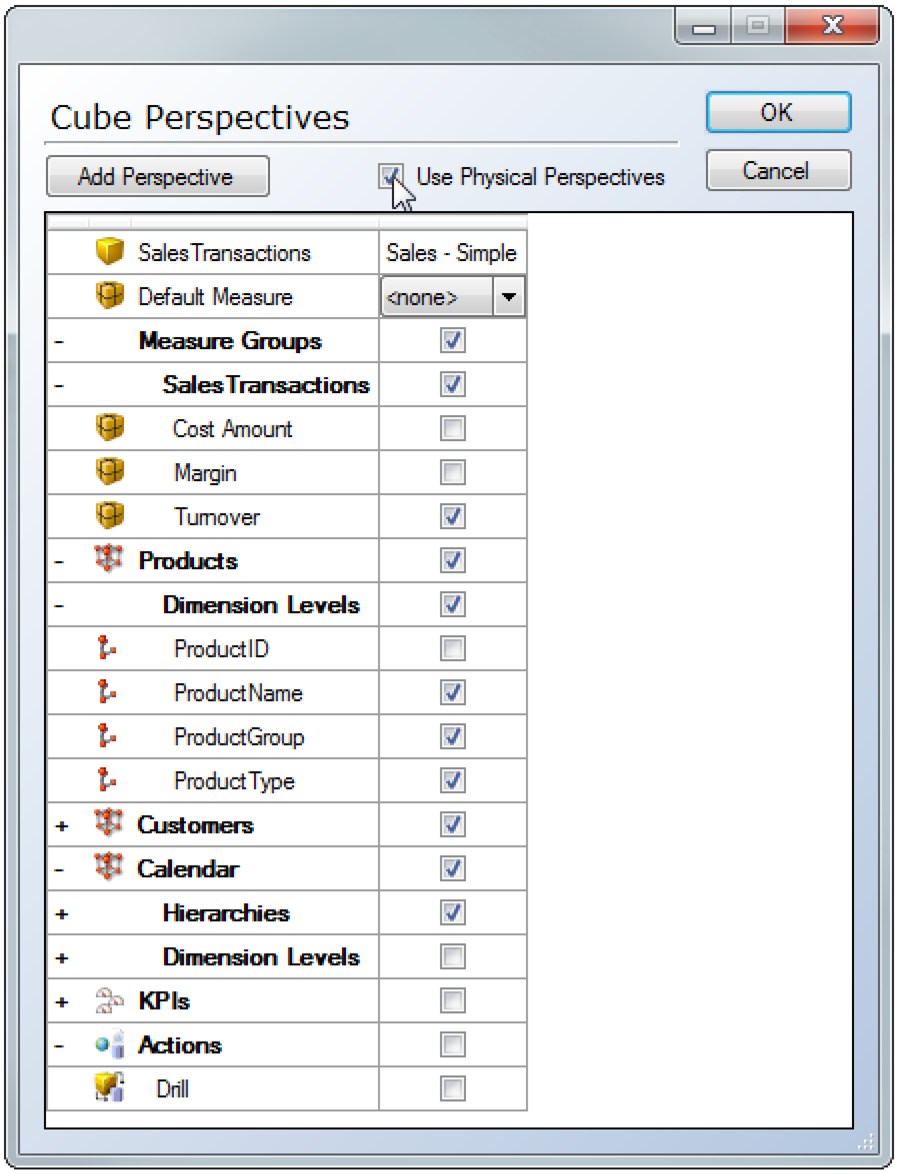
SSAS Cube Perspectives is a Microsoft SQL Server feature available in the Business Intelligence and Enterprise edition only. If you have Standard Edition of Microsoft SQL Server, you can still get the same functionality with the TX2014 exclusive feature: Physical SSAS Cube Perspectives.

Physical SSAS Cube Perspectives

SSAS Cube Perspectives is a Microsoft SQL Server feature available in the Business Intelligence and Enterprise edition only.

But, with the Physical SSAS Cube Perspectives feature, you can get the same functionality on Standard Edition of Microsoft SQL Server.

Simply define your Perspectives as described for the feature “SSAS Cube Perspectives,” and add a checkmark in the “Use Physical Perspectives”:

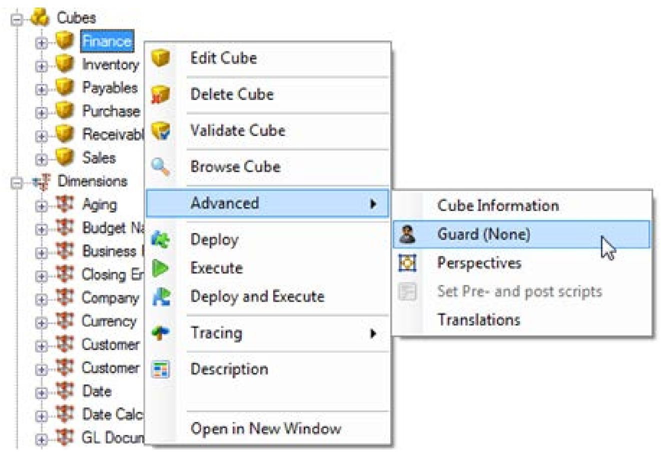


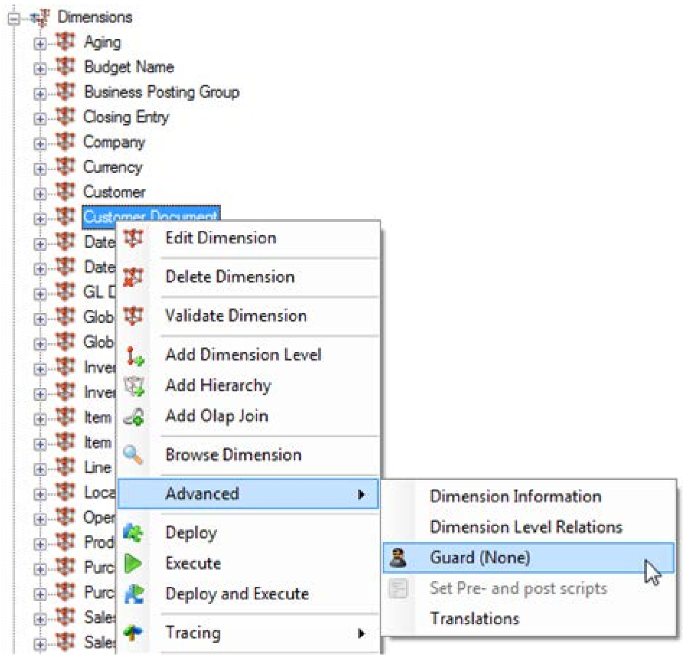
[✓] Use Physical Perspectives

SalesTransactions	Sales - Simple
Default Measure	<none>
- **Measure Groups**	✓
- **SalesTransactions**	✓
Cost Amount	☐
Margin	☐
Turnover	✓
- **Products**	✓
- **Dimension Levels**	✓
ProductID	☐
ProductName	✓
ProductGroup	✓
ProductType	✓
+ **Customers**	✓
- **Calendar**	✓
+ **Hierarchies**	✓
+ **Dimension Levels**	☐
+ **KPIs**	☐
- **Actions**	☐
Drill	☐

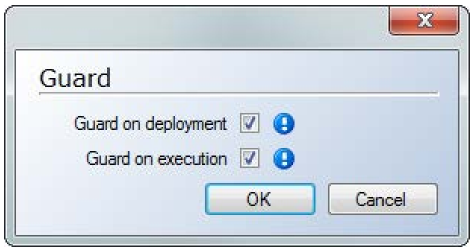
Prevent Deployment and Execution of OLAP Objects

With this feature, you can prevent deployment and execution of specific Cubes or Dimensions. To prevent deployment and execution of a OLAP Object, select Guard on the Advanced menu:





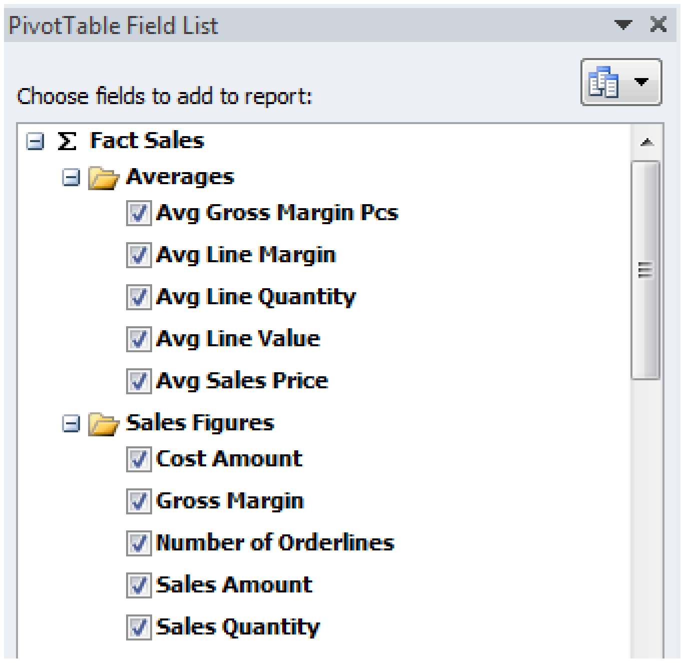
The objects can be guarded on Deployment, on Execution, or on both:



If guarded on deployment, then tX will not deploy the object. If you try and execute an OLAP object that has not been deployed, you will receive an error.

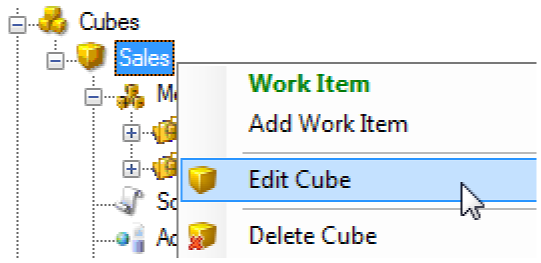
Rename Measuregroup

The name of the measure group is displayed in the cube front end; here it is shown from Excel 2010:

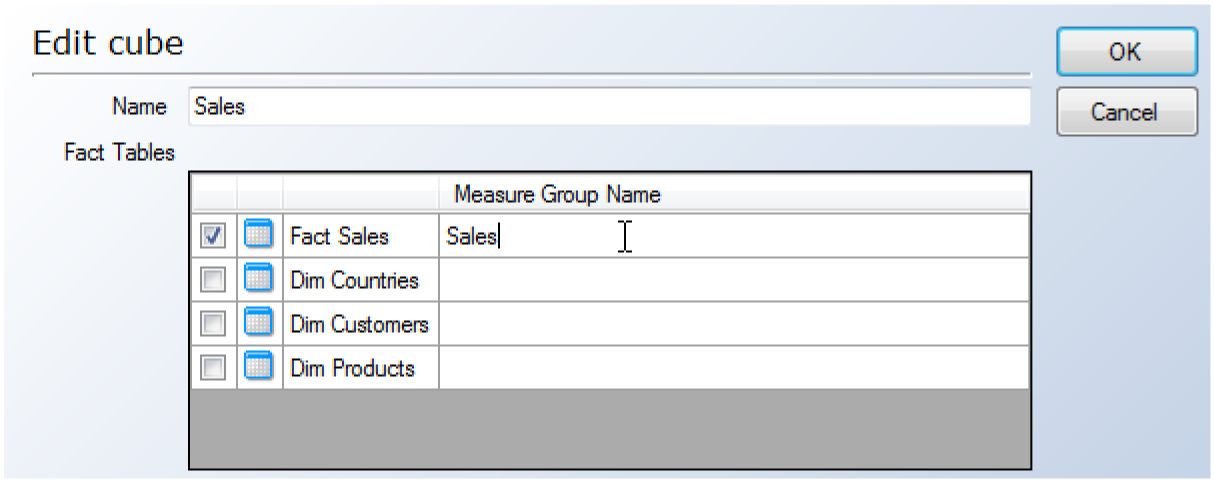


In TX2014 the name of the measuregroup is defaulted to the name of the fact table. This feature makes it possible to rename the measure group without having to rename the fact table in the data warehouse.

To rename a measure group select Edit Cube:



Then enter a new name for the measure group:


Edit cube

Name: Sales

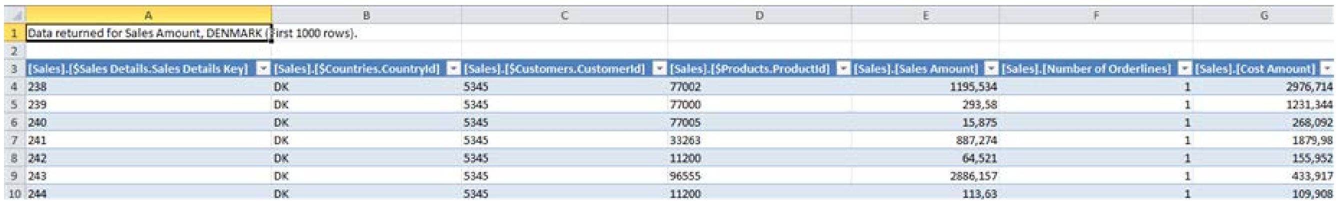
Fact Tables

			Measure Group Name
✓	▦	Fact Sales	Sales
☐	▦	Dim Countries	
☐	▦	Dim Customers	
☐	▦	Dim Products	

Drill through Actions

Drill through actions are very useful when the end users want to dig down behind the numbers in a pivot table. The default drill through action can be used only in Excel shows, measure fields, and dimension key fields. In a drill through action, the user can select exactly what information should be displayed and in which order.

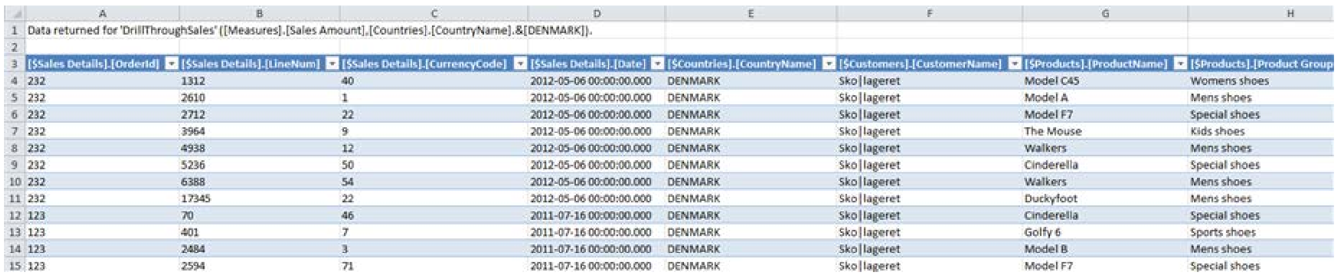
Here is an example from Excel. Notice that only the Surrogate Key is displayed for the Sales Details:



Data returned for Sales Amount, DENMARK (First 1000 rows).

[Sales].[$Sales Details.Sales Details Key]	[Sales].[$Countries.CountryId]	[Sales].[$Customers.CustomerId]	[Sales].[$Products.ProductId]	[Sales].[Sales Amount]	[Sales].[Number of Orderlines]	[Sales].[Cost Amount]
238	DK	5345	77002	1195,534	1	2976,714
239	DK	5345	77000	293,58	1	1231,344
240	DK	5345	77005	15,875	1	268,092
241	DK	5345	33263	887,274	1	1879,98
242	DK	5345	11200	64,521	1	155,952
243	DK	5345	96555	2886,157	1	433,917
244	DK	5345	11200	113,63	1	109,908

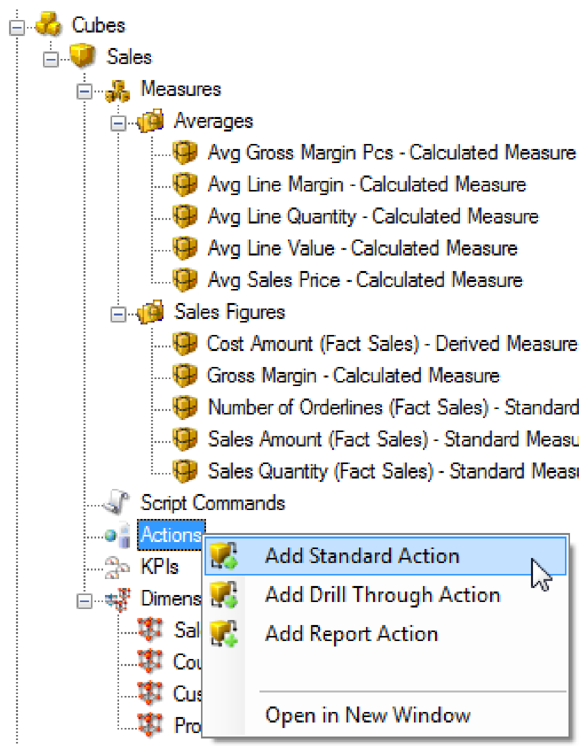
Here is an example from Excel with the drill through action enabled:



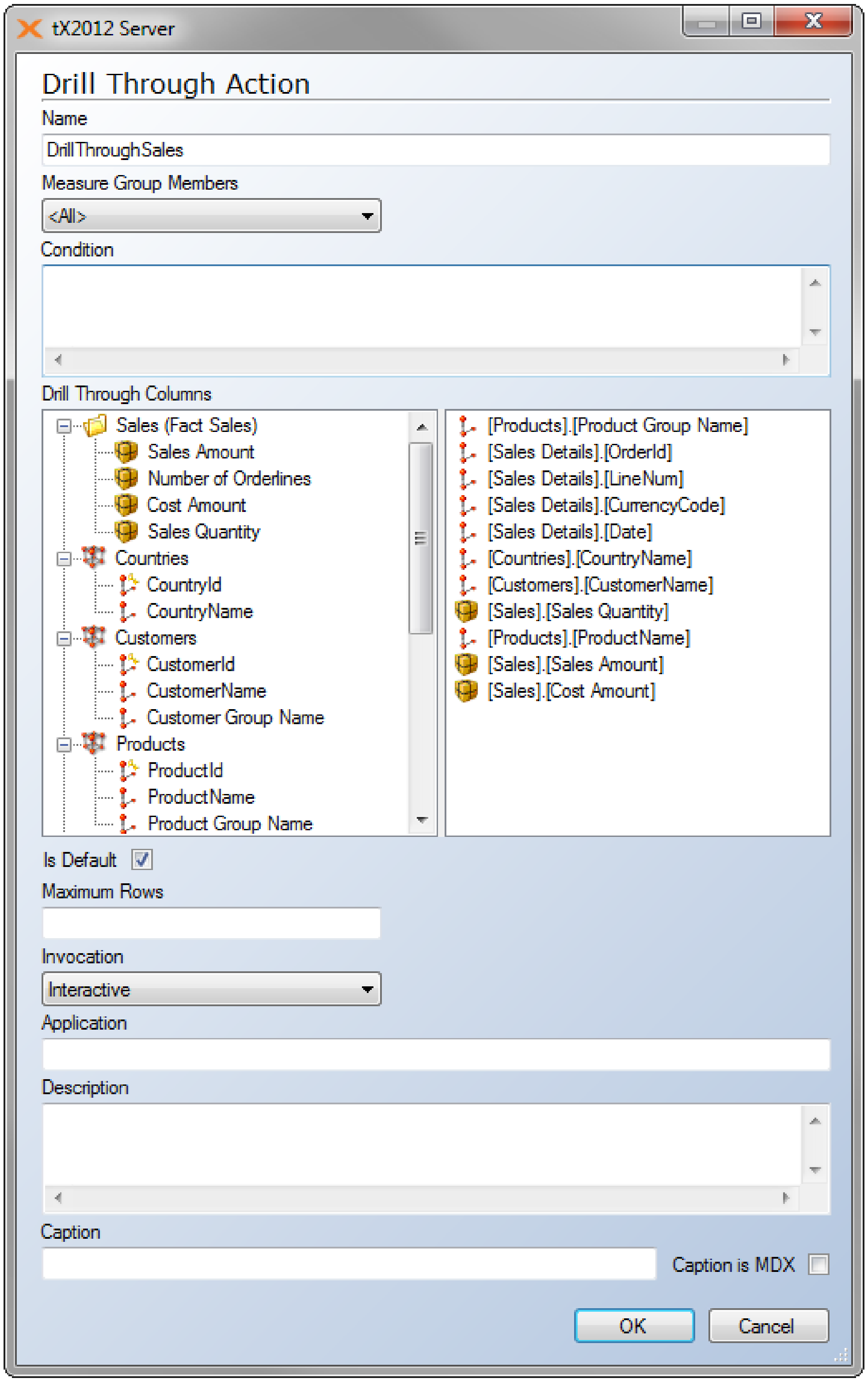
Data returned for 'DrillThroughSales' ([Measures].[Sales Amount],[Countries].[CountryName].&[DENMARK]).

[$Sales Details].[OrderId]	[$Sales Details].[LineNum]	[$Sales Details].[CurrencyCode]	[$Sales Details].[Date]	[$Countries].[CountryName]	[$Customers].[CustomerName]	[$Products].[ProductName]	[$Products].[Product Group
232	1312	40	2012-05-06 00:00:00.000	DENMARK	Sko|lageret	Model C45	Womens shoes
232	2610	1	2012-05-06 00:00:00.000	DENMARK	Sko|lageret	Model A	Mens shoes
232	2712	22	2012-05-06 00:00:00.000	DENMARK	Sko|lageret	Model F7	Special shoes
232	3964	9	2012-05-06 00:00:00.000	DENMARK	Sko|lageret	The Mouse	Kids shoes
232	4938	12	2012-05-06 00:00:00.000	DENMARK	Sko|lageret	Walkers	Mens shoes
232	5236	50	2012-05-06 00:00:00.000	DENMARK	Sko|lageret	Cinderella	Special shoes
232	6388	54	2012-05-06 00:00:00.000	DENMARK	Sko|lageret	Walkers	Mens shoes
232	17345	22	2012-05-06 00:00:00.000	DENMARK	Sko|lageret	Duckyfoot	Mens shoes
123	70	46	2011-07-16 00:00:00.000	DENMARK	Sko|lageret	Cinderella	Special shoes
123	401	7	2011-07-16 00:00:00.000	DENMARK	Sko|lageret	Golfy 6	Sports shoes
123	2484	3	2011-07-16 00:00:00.000	DENMARK	Sko|lageret	Model B	Mens shoes
123	2594	71	2011-07-16 00:00:00.000	DENMARK	Sko|lageret	Model F7	Special shoes

To create a drill through action, simply select Add Drill Through Action:


Cubes
Sales
Measures
Averages
Avg Gross Margin Pcs - Calculated Measure
Avg Line Margin - Calculated Measure
Avg Line Quantity - Calculated Measure
Avg Line Value - Calculated Measure
Avg Sales Price - Calculated Measure
Sales Figures
Cost Amount (Fact Sales) - Derived Measure
Gross Margin - Calculated Measure
Number of Orderlines (Fact Sales) - Standard Measure
Sales Amount (Fact Sales) - Standard Measure
Sales Quantity (Fact Sales) - Standard Measure
Script Commands
Actions
KPIs
Dimens
Sal
Cou
Cus
Pro
Add Standard Action
Add Drill Through Action
Add Report Action
Open in New Window

The drill through actions are defined in this dialogue:



tX2012 Server

Drill Through Action

Name

DrillThroughSales

Measure Group Members

<All>

Condition

Drill Through Columns

- Sales (Fact Sales)
  - Sales Amount
  - Number of Orderlines
  - Cost Amount
  - Sales Quantity
- Countries
  - CountryId
  - CountryName
- Customers
  - CustomerId
  - CustomerName
  - Customer Group Name
- Products
  - ProductId
  - ProductName
  - Product Group Name

[Products].[Product Group Name]
[Sales Details].[OrderId]
[Sales Details].[LineNum]
[Sales Details].[CurrencyCode]
[Sales Details].[Date]
[Countries].[CountryName]
[Customers].[CustomerName]
[Sales].[Sales Quantity]
[Products].[ProductName]
[Sales].[Sales Amount]
[Sales].[Cost Amount]

Is Default ✓

Maximum Rows

Invocation

Interactive

Application

Description

Caption

Caption is MDX ☐

OK Cancel

Fields on the Drill Through Action dialogue:

Name: The name of the action.

Measure Group Members: If the drill through action should be active on measures from all measuregroups, then select <All> - or choose the specific measure group for which you want the action to be functioning.

Condition: An MDX expression that returns True or False. The drill through action will only be active where the condition is True. Example: IIF(<Statement>,True,False)

Drill Through Columns: The columns that will be shown when the drill through action is used. The fields will appear in the same order as shown on the right hand side. You can drag items from the left side to the right side to include the information in the action. To delete information from the action, simple press the Delete key when the information is selected, or use the right click menu.

Please note that there are some limitations in drill through actions. These are: You can only select measures from one measure group, and you can only select dimensions that are related to the measure group that you are showing measures for.

Is Default: Select True to include this drill through action as a default drill through action, otherwise, select False. If more than one drill through actions is set as default, the Microsoft SQL Server Analysis Services instance evaluates all default drill through actions and runs the first default drillt hrough action that returns a non-empty set.

Maximum Rows: The maximum number of rows to be returned by the drill through action. Setting this option to zero or an empty value indicates that the drill through action returns all rows retrieved by the action to the client application.

Invocation: Select the setting that indicates when the action should be carried out. This option only provides a recommendation to a client application as to when to execute an action and does not directly control the invocation of the action.

The following table describes the available settings:

Batch	The action should run as part of a batch operation or an Integration Services task
Interactive	The action runs when the user invokes the action
On Open	The action runs when the cube is first opened

Application: Type the name of the application that can execute the drill through action. You can also use this option to identify which client application most commonly uses this action, as well as to display appropriate icons next to the action in a pop-up menu. This option only provides a recommendation to a client application as to what client application should execute an action, and does not directly control access to the action. Client applications should hide any actions that are associated with other client applications.

Description: An optional description of the action.

Caption: The caption to be displayed for the action in the client application if Caption Is MDX is set to False.

Type the Multidimensional Expressions (MDX) expression that returns a string for the caption if Caption Is MDX is set to True.

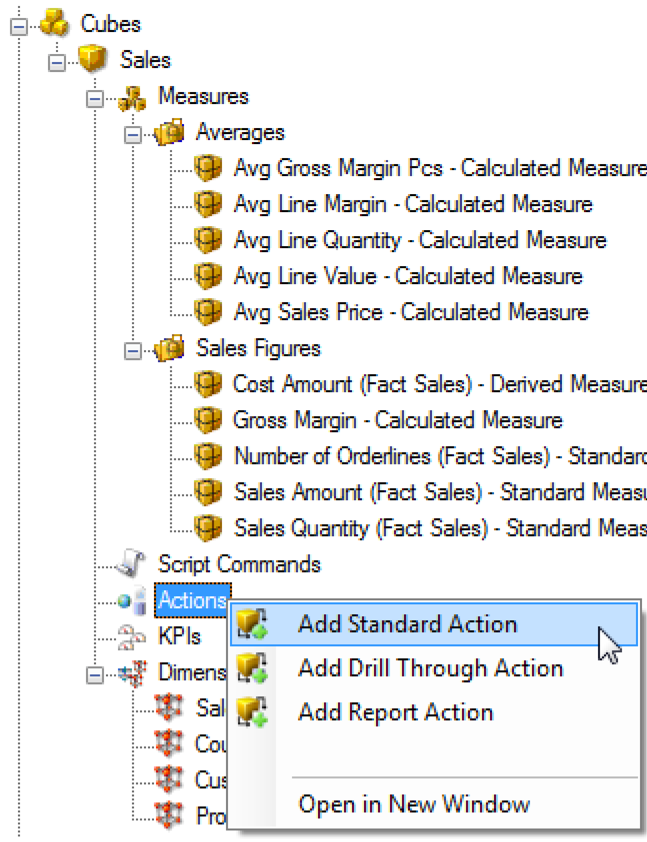
Caption Is MDX: Select False to indicate that Caption contains a literal string representing a caption to be displayed for the action in the client application. Select True to indicate that Caption contains an MDX expression that returns a string representing a caption to be displayed for the action in the client application. The MDX expression must be resolved before the action is returned to the client application.

Standard Actions

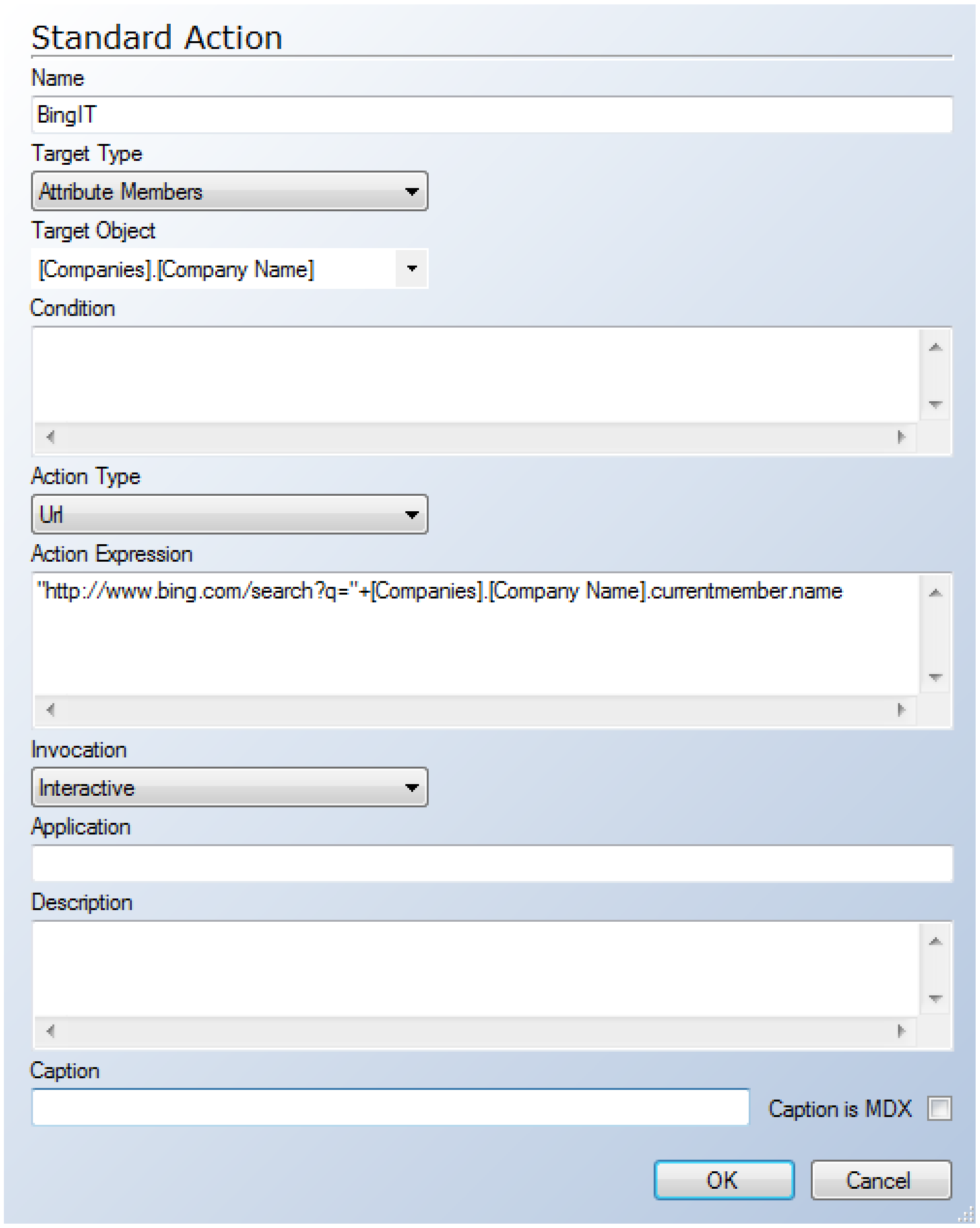
Where Drill Through Actions returns the set of rows that represents the underlying data of the selected cell of the cube where the action occurs, then the Standard actions returns the action element (URL, HTML, DataSet, RowSet, and other elements) that is associated with the selected section of the cube where the action occurs.

The example used in this document creates a Standard Action that will open the browser and do a Bing-search for the dimension attribute value (Company Name on the Company dimension).

To create a Standard action, simply select Add Standard Action:


Cubes
Sales
Measures
Averages
Avg Gross Margin Pcs - Calculated Measure
Avg Line Margin - Calculated Measure
Avg Line Quantity - Calculated Measure
Avg Line Value - Calculated Measure
Avg Sales Price - Calculated Measure
Sales Figures
Cost Amount (Fact Sales) - Derived Measure
Gross Margin - Calculated Measure
Number of Orderlines (Fact Sales) - Standard Measure
Sales Amount (Fact Sales) - Standard Measure
Sales Quantity (Fact Sales) - Standard Measure
Script Commands
Actions
KPIs
Dimens
Sal
Cou
Cus
Pro
Add Standard Action
Add Drill Through Action
Add Report Action
Open in New Window

The action is defined in this dialogue:



Standard Action

Name: BingIT

Target Type: Attribute Members

Target Object: [Companies].[Company Name]

Condition:

Action Type: Url

Action Expression: "http://www.bing.com/search?q="+[Companies].[Company Name].currentmember.name

Invocation: Interactive

Application:

Description:

Caption:

Caption is MDX ☐

Fields on the Standard Action dialogue:

Name: The name of the action.

Target Type / Target Object: The object you select in Target Type determines the objects that are available and the type of selection that you can make in Target Object. The following table lists valid Target Object selections for each target type:

Attribute Members	The only valid selection is a single attribute hierarchy. The target type of the action will be all members of the attribute wherever they appear
Cells	All cells is the only selection available. If you choose Cells as a target type, you can type an expression in Condition to restrict the cells with which the action is associated
Cube	CURRENTCUBE is the only selection available. The action is associated with the current cube
Dimension Members	Select a single dimension. The action will be associated with all members of the dimension
Hierarchy	Select a single hierarchy. The action will be associated with the hierarchy object only
Hierarchy Member	Select a single hierarchy
Level	Select a single level. The action will be associated with the level object only
Level Members	Select a single level. The action will be associated with all members of the selected level

Condition: An MDX expression that returns True or False. The drill through action will only be active where the condition is True. Example: IIF(<Statement>,True,False)

Action Type: The type of action that you want to create. The following table lists the types of actions that are available:

Dataset	Retrieves a dataset
Proprietary	Performs an operation using an interface other than those listed in this table
Rowset	Retrieves a rowset
Statement	Runs an OLE DB command
URL	Display a Web page in an Internet Browser

Action Expression: An expression that defines the action.

Invocation: Select the setting that indicates when the action should be carried out. This option only provides a recommendation to a client application as to when to execute an action and does not directly control the invocation of the action.

The following table describes the available settings:

Batch	The action should run as part of a batch operation or an Integration Services task
Interactive	The action runs when the user invokes the action
On Open	The action runs when the cube is first opened

Application: Type the name of the application that can execute the drill through action. You can also use this option to identify which client application most commonly uses this action, as well as to display appropriate icons next to the action in a pop-up menu. This option only provides a recommendation to a client application as to what client application should execute an action and does not directly control access to the action. Client applications should hide any actions that are associated with other client applications.

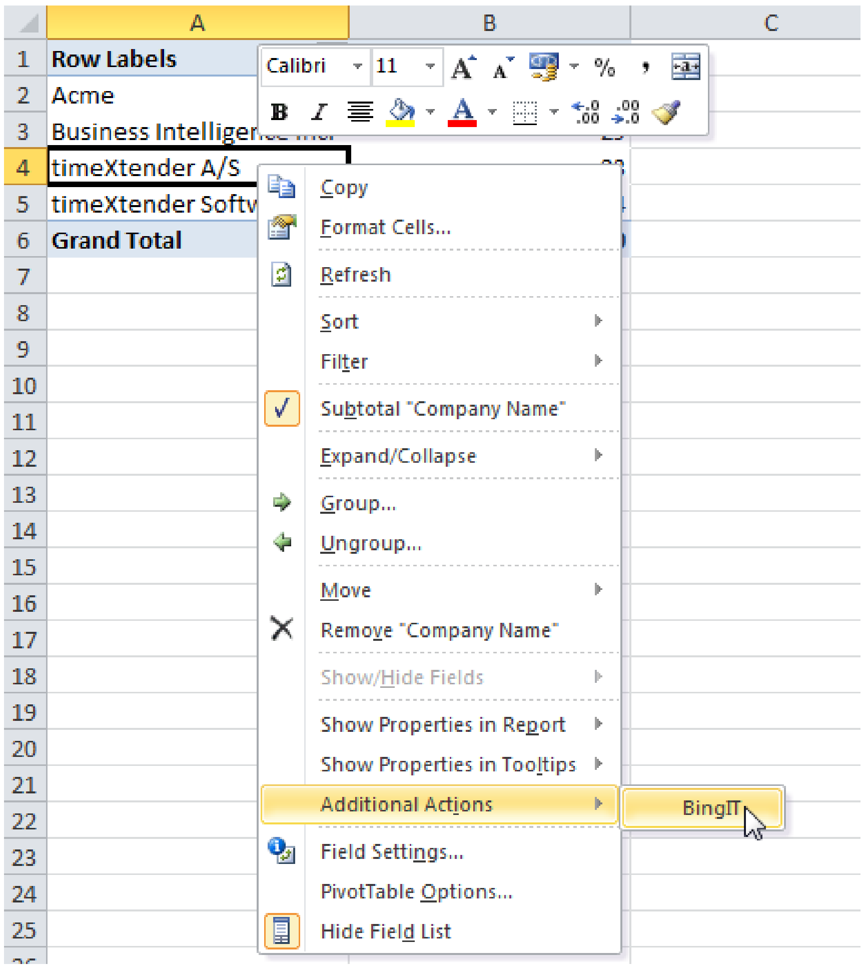
Description: An optional description of the action.

Caption: The caption to be displayed for the action in the client application if Caption Is MDX is set to False.

Type the Multidimensional Expressions (MDX) expression that returns a string for the caption if Caption Is MDX is set to True.

Caption Is MDX: Select False to indicate that Caption contains a literal string representing a caption to be displayed for the action in the client application. Select True to indicate that Caption contains an MDX expression that returns a string representing a caption to be displayed for the action in the client application. The MDX expression must be resolved before the action is returned to the client application.

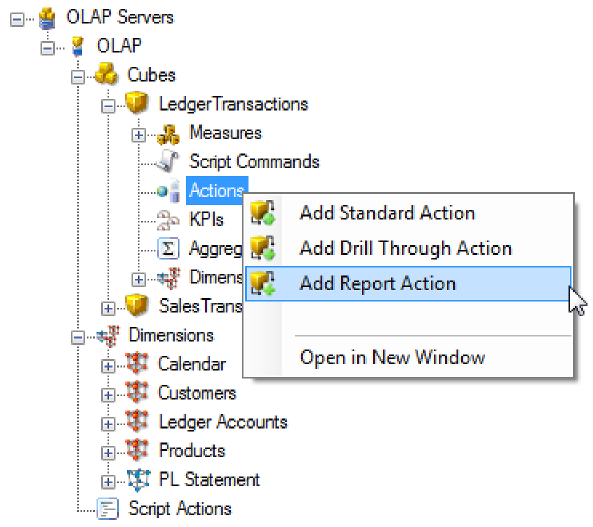
The Action can be used in the Cube front-end. Here it is shown in Excel:


A
B
C
Row Labels
Acme
Business Intelligen
timeXtender A/S
timeXtender Softw
Grand Total
Calibri
11
Copy
Format Cells...
Refresh
Sort
Filter
Subtotal "Company Name"
Expand/Collapse
Group...
Ungroup...
Move
Remove "Company Name"
Show/Hide Fields
Show Properties in Report
Show Properties in Tooltips
Additional Actions
BingIT
Field Settings...
PivotTable Options...
Hide Field List

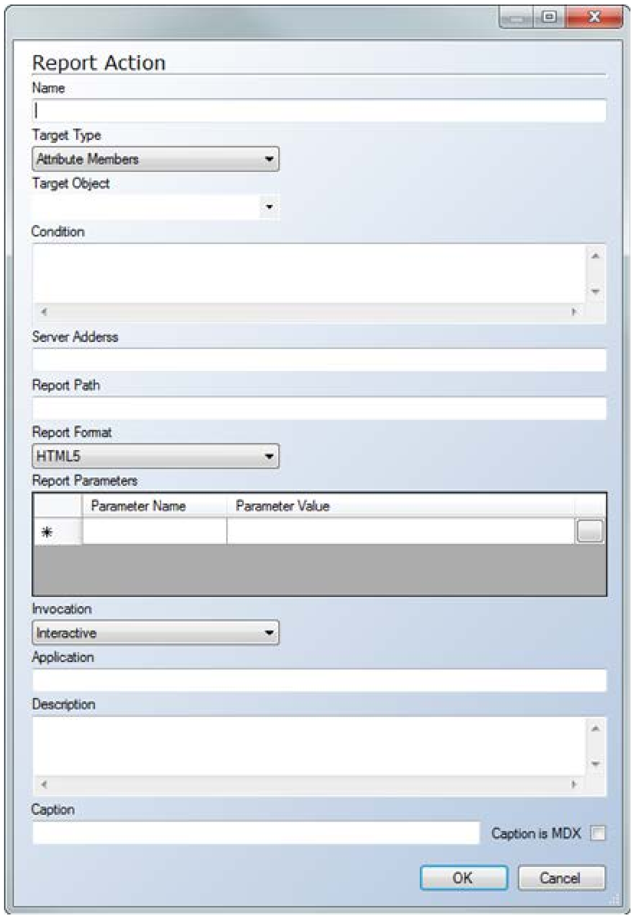
Reporting Actions

Reporting actions are very useful when the end users want to query the cube data at a more detailed level than the cube's granularity. The Reporting action can be used to fire a Pre-defined Reporting Services report while forwarding parameters to the report based on the current location in the cube.

To create a Reporting Action select "Add Report Action":



This screen will open:



Fields on the Reporting Action dialogue:

Name: The name of the action.

Target type: The type of object to which the action is to be associated. The server returns to the client only those actions that apply to the object of the specified type.

The following Target Types are supported:

Attribute Members	Select Attribute in the Target Object
Cells	All Cells is selected in the Target Object Field
Cube	CURRENTCUBE is selected in the Target Object Field
Dimension Members	Select Dimension Member in the Target Object Field
Hierarchy	Select Hierarchy in the Target Object Field
Hierarchy Members	Select Hierarchy Member in the Target Object Field
Level	Select Level in the Target Object Field
Level Members	Select Member in the Target Object field

Target Object: Select the object to which the action is to be associated. The Microsoft SQL Server Analysis Services instance returns to the client only those actions that apply to the selected object. The list of available objects is constrained by the choice of Target type.

Condition: An MDX expression that returns True or False. The drill through action will only be active where the condition is True. Example: IIF(<Statement>,True,False)

Server Address: Type the name of the Microsoft SQL Server Reporting Services instance on which the action runs the report.

Report Path: Type the path to the report on the Reporting Services instance. For example, type Finance/DetailedLedgerTransactions.

Report Format: Select the format in which the report is returned. Choose between HTML5, HTML3, Excel, and PDF.

Report Parameters: Report parameters can be supplied for the report. Specify the name of the parameter and the value of the report parameter to be passed to the report. If the parameter is set to an MDX expression, then the expression is evaluated when the action is run; otherwise, it is passed to the report without modification.

Invocation: Select the setting that indicates when the action should be carried out. This option only provides a recommendation to a client application as to when to execute an action and does not directly control the invocation of the action.

The following table describes the available settings:

Batch	The action should run as part of a batch operation or an Integration Services task
Interactive	The action runs when the user invokes the action
On Open	The action runs when the cube is first opened

Application: Type the name of the application that can execute the drill through action. You can also use this option to identify which client application most commonly uses this action, as well as to display appropriate icons next to the action in a pop-up menu. This option only provides a recommendation to a client application as to what client application should execute an action and does not directly control access to the action. Client applications should hide any actions that are associated with other client applications.

Description: An optional description of the action.

Caption: The caption to be displayed for the action in the client application if Caption Is MDX is set to False.

Type the Multidimensional Expressions (MDX) expression that returns a string for the caption if Caption Is MDX is set to True.

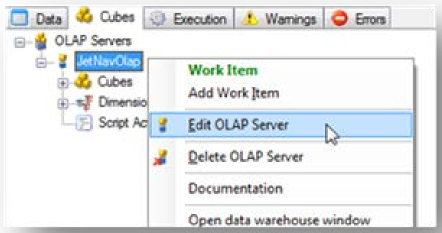
Caption Is MDX: Select False to indicate that Caption contains a literal string representing a caption to be displayed for the action in the client application. Select True to indicate that Caption contains an MDX expression that returns a string representing a caption to be displayed for the action in the client application. The MDX expression must be resolved before the action is returned to the client application.

High Availability Cube Processing

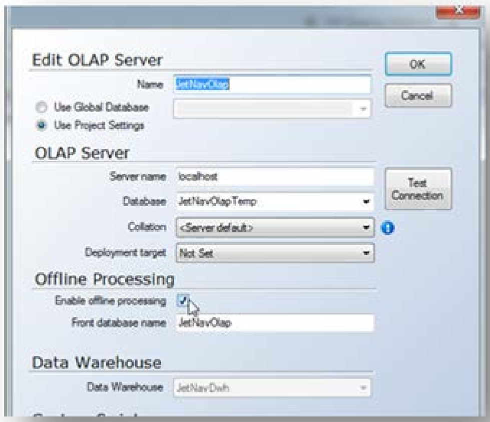
TX2014 supports functionality that allows cubes to be processed without being taken offline. Normally, in SQL Server Analysis Services when a cube is processed and rebuilt, it is taken offline during the duration of the processing and is made unavailable to end-users. There is some functionality in SQL Server Enterprise Edition that allows the cube to be taken offline, but many users do not own this edition of SQL. The High Availability Cube Processing feature in allows TX2014users to take advantage of this without the requirement of owning SQL Enterprise Edition. This way, cubes can update throughout the business day with no impact to end-users. During the processing of the cube, the users will still have access to the original version of the cube, which is then replaced with the new version of the cube, once processing has been completed. **High Availability Cube Processing is an add-on feature that is available for purchase**.

How to Enable High Availability Cube Processing

1. On the **Cubes** tab, right-click on the name of the OLAP database, and select **Edit OLAP Server**.



2. Check the box labeled **Enable Offline Processing**.



Edit OLAP Server

Name: JetNavOlap

Use Global Database

Use Project Settings

OLAP Server

Server name: localhost

Database: JetNavOlapTemp

Collation: <Server default>

Deployment target: Not Set

Offline Processing

Enable offline processing: ✓

Front database name: JetNavOlap

Data Warehouse

Data Warehouse: JetNavDwh

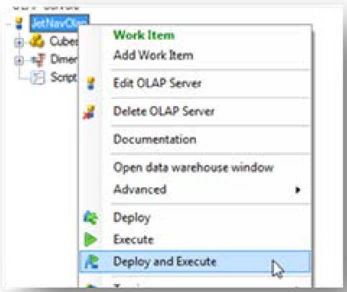
3. Rename the **Database** field in the **OLAP Server** section to represent a temporary OLAP database that will be used during processing. This is *not* the database that will

be used by end-users.

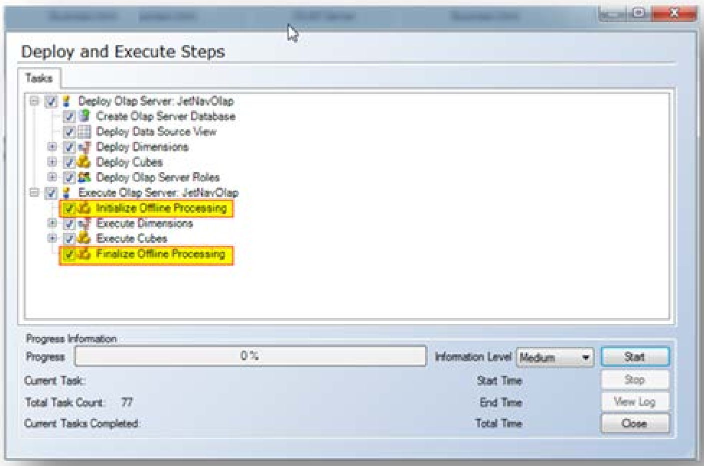
4. Type in the name of the database that will be used by end-users in the **Front Database Name** field in the **Offline Processing** section.

5. Click **OK.**

6. Right-click the OLAP database, and click **Deploy and Execute.**



7. There will now be two additional steps in the **Execute OLAP Server** routine that will handle the Offline Cube Processing. These are **Initialize Offline Processing** and **Finalize Offline Processing.**


Deploy and Execute Steps

Tasks

- Deploy Olap Server: JetNavOlap
  - Create Olap Server Database
  - Deploy Data Source View
  - Deploy Dimensions
  - Deploy Cubes
  - Deploy Olap Server Roles
- Execute Olap Server: JetNavOlap
  - Initialize Offline Processing
  - Execute Dimensions
  - Execute Cubes
  - Finalize Offline Processing

Progress Information

Progress: 0 %

Information Level: Medium

Current Task:

Total Task Count: 77

Current Tasks Completed:

Start Time

End Time

Total Time

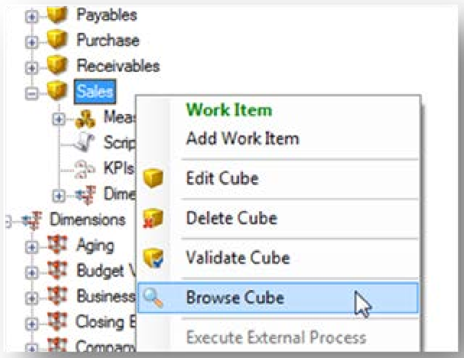
Click the **Start** button to begin processing the cubes. Users will be able to access the cubes as they are being processed.

Cube Browser

The Cube Browser allows aTX2014 user to browse a cube from within TX2014 without first leaving to go into another application such as Excel. The Cube Browser is not meant to replace a proper front-end tool, such as Excel, but is an easy way to browse the cube structure without having to navigate away from TX2014.

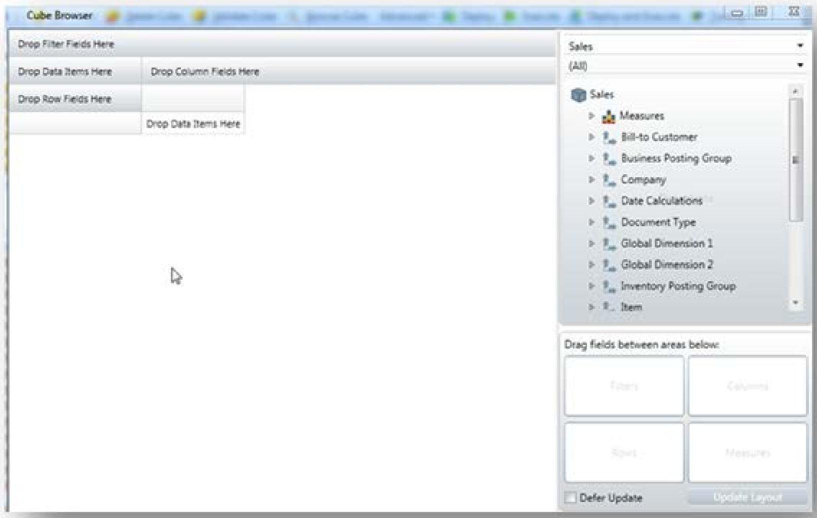
How to Launch the Cube Browser

1. Locate the cube that will be browsed on the **Cube** tab, right-click the cube, and select **Browse Cube.**

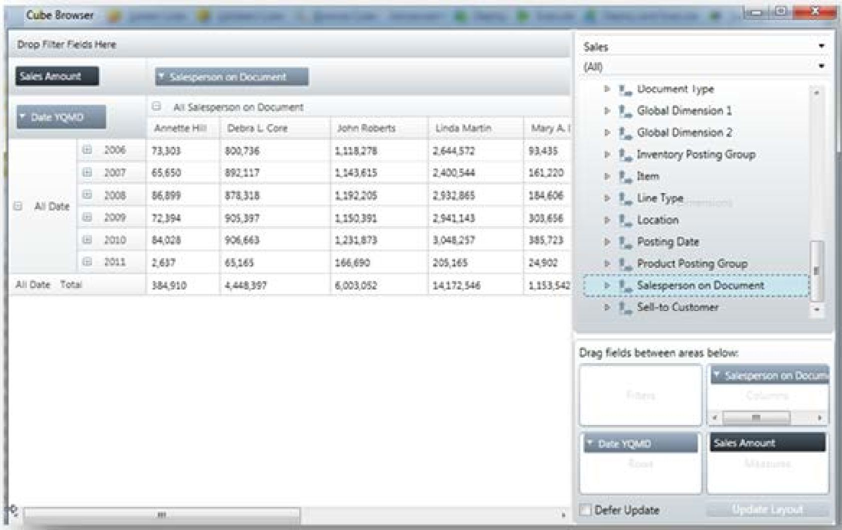


2. The user interface will be used similarly to pivot tables in Excel. Measures and dimensions are dragged from the list on the right and dropped in either the boxes in the lower right-hand corner or directly onto the workspace pane on the left.

How to Use the Cube Browser

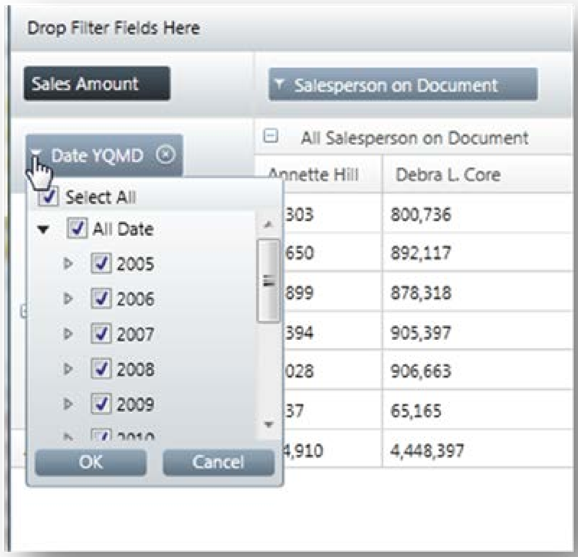


The following report was created from the Sales cube for NAV by dragging the **Date** dimension into the **Rows** box, the**Salesperson on Document** dimension into the **Columns** box, and the **Sales Amount** measure into the **Measures** box.



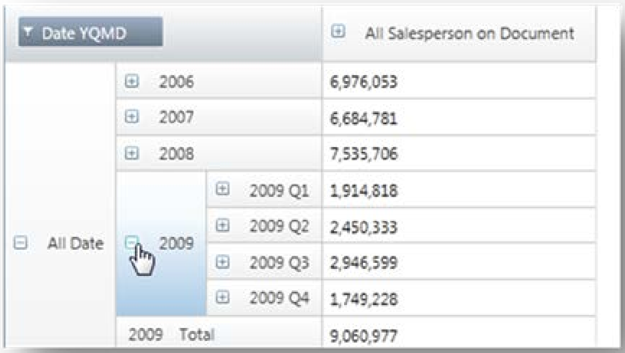
		All Salesperson on Document				
		Annette Hill	Debra L. Core	John Roberts	Linda Martin	Mary A. I
All Date	2006	73,303	800,736	1,118,278	2,644,572	93,435
	2007	65,650	892,117	1,143,615	2,400,544	161,220
	2008	86,899	878,318	1,192,205	2,932,865	184,606
	2009	72,394	905,397	1,150,391	2,941,143	303,656
	2010	84,028	906,663	1,231,873	3,048,257	385,723
	2011	2,637	65,165	166,690	205,165	24,902
All Date Total		384,910	4,448,397	6,003,052	14,172,546	1,153,542

Filters for the rows and columns can be added by clicking the **Filter** icon to the left of the row and the column labels in the workspace pane on the left.



All Salesperson on Document	
Annette Hill	Debra L. Core
303	800,736
650	892,117
899	878,318
394	905,397
028	906,663
37	65,165
4,910	4,448,397

Dimensions can also be expanded and collapsed by clicking the plus and minus signs respectively.



Date YQMD			All Salesperson on Document
All Date	2006		6,976,053
	2007		6,684,781
	2008		7,535,706
	2009	2009 Q1	1,914,818
		2009 Q2	2,450,333
		2009 Q3	2,946,599
		2009 Q4	1,749,228
	2009 Total		9,060,977

Execution Packages

You will typically specify an execution schedule for your entire project which will automatically execute the project accordingly.

It is also possible to specify an execution schedule for only a part of a project. For example, if you want to deploy and execute specific dimensions or cubes on a regular basis you would create an advanced execution package.

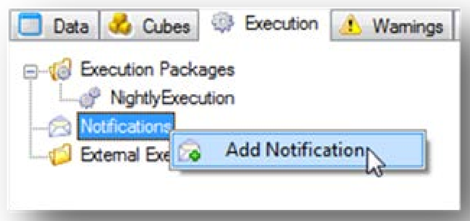
Important: The scheduled execution packages are handled by a service on the machine named “TX2014 Scheduler”. This service is automatically installed duringTX2014packages to run. For optimal performance, identify a service account to be used to start this service as “Log on As”. This account should log in to the server once, open TX2014, and point to the project repository database. This will create a configuration file for the account that will allow the service to start. The Startup Type for the service should always be set to **Automatic** or **Automatic (Delayed Start)**.

Note: An execution package will only execute the objects in the package, not deploy them. If changes have been made to the project, but have not been successfully deployed, then the scheduled execution package will most likely fail.

To Create Notifications

Notifications will alert specified individuals when the execution package was successfully run or in case something caused it to fail. Notifications are most commonly set up as email alerts but can be saved to the Event Log as well.

1. On the **Execution** tab, right-click **Notifications**, and then select **Add Notification**.



2. In the **Name** field, type a name for the notification.

3. In the **Type** field, select the type of notification you want to create. You have the following options:

Mail	Creates an email notification
EventLog	Writes a notification to the event log
Both	Creates both an email notification and writes to the event log

4. In the **Mail Server Name** field, type the name of the mail server you wish to use.

Note: SMTP is currently the only supported email method.

5. In the **From** email field, type the originating email address.

6. In the **To** field, type the destination email address..

7. In the **Cc** field, type the email address you want to send a copy of the notification to.

8. Click **OK**.

The notification can now be selected when you create an execution package.

Email Notifications

It is possible to customize the Subject on notification mails.

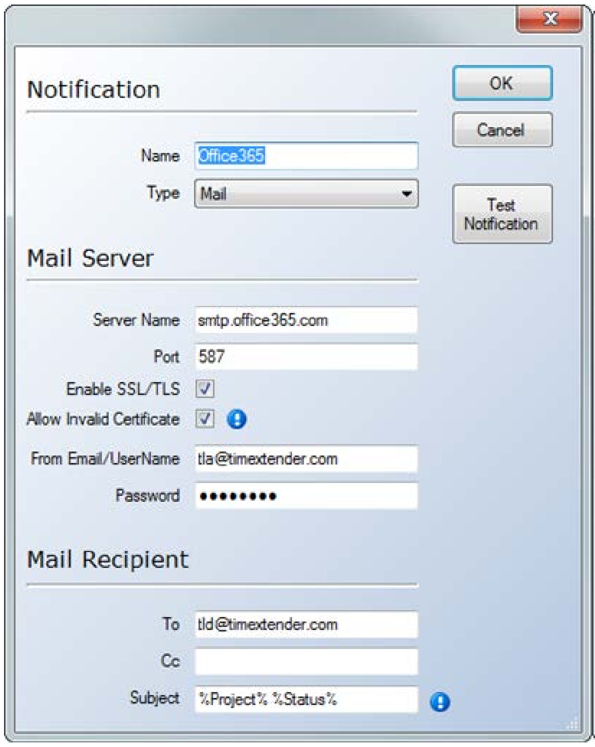
The following parameters are accepted:

%Project% = The name of the project

%Status% = Status of the execution (Success / Fail)

%ExecutionPackage% = Name of the execution package

It is possible to setup the mail server to support SSL/TLS and user authentication. With these options, it is also possible to use office365 as your mail server for sending notifications.


Notification

Name: Office365

Type: Mail

Mail Server

Server Name: smtp.office365.com

Port: 587

Enable SSL/TLS: ✓

Allow Invalid Certificate: ✓

From Email/UserName: tla@timextender.com

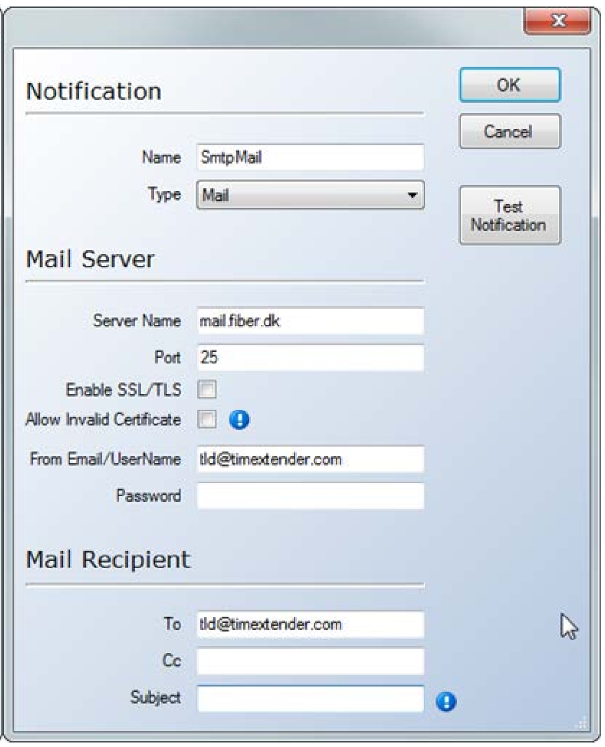
Password: ••••••••

Mail Recipient

To: tld@timextender.com

Cc:

Subject: %Project% %Status%


Notification

Name: SmtpMail

Type: Mail

Mail Server

Server Name: mail.fiber.dk

Port: 25

Enable SSL/TLS: ☐

Allow Invalid Certificate: ☐

From Email/UserName: tld@timextender.com

Password:

Mail Recipient

To: tld@timextender.com

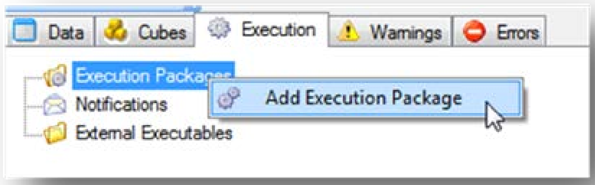
Cc:

Subject:

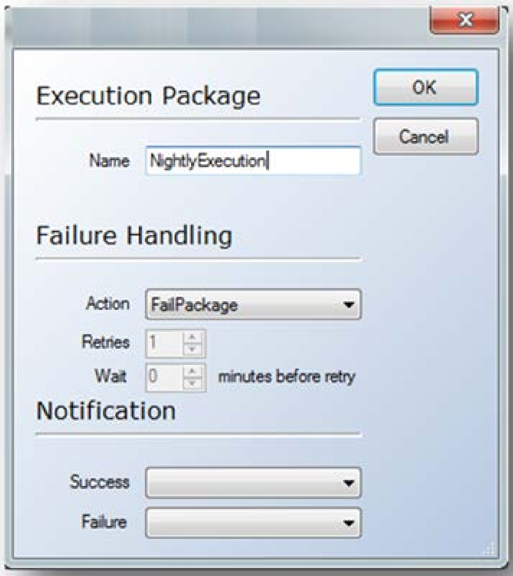
To Add Execution Packages

An execution package determines which elements of a project will be deployed and executed.

1. On the **Execution** tab, right-click **Execution Packages**, and then select **Add Execution Package**.



A window will appear with the following settings:



2. In the **Name** field, type a name for the **Execution Package**.

3. In the **Action** field, specify what should happen if the execution fails. You have the following options:

FailPackage	Cancels the execution in case of failure
RestartPreviousStep	Restarts the execution from the previous step
RestartFirstStep	Restarts the execution process from the beginning

4. If you select **RestartPreviousStep** or **RestartFirstStep**, specify in the **Retries** field how many times you want the execution package to restart. In the **Wait** field, specify the number of minutes to wait before retrying again.

5. In the **Success** list, select the notification to use when the execution package succeeds.

Note: You have to create a notification before it is available from the list. See To Create Notifications

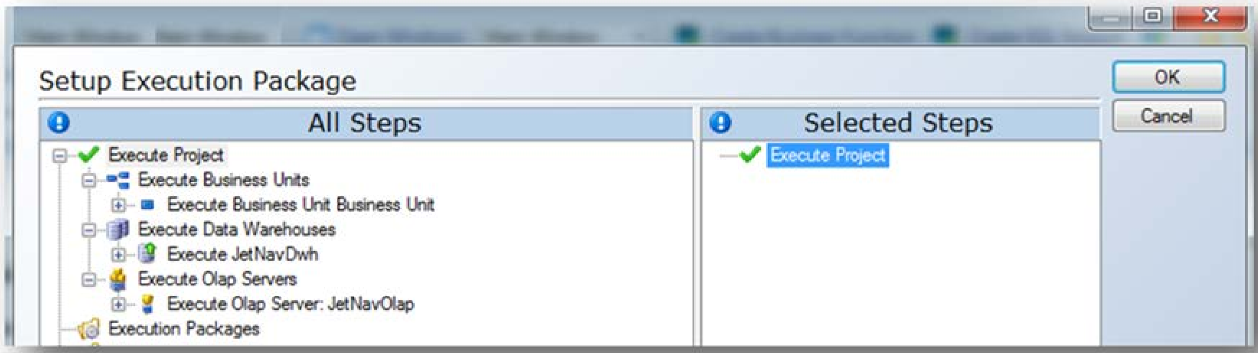
6. In the **Failure** list, select the notification to use when the execution package fails.

Note: You have to create a notification before it is available from the list. See To Create Notifications

7. Click OK.

8. You will be prompted if you want to set up the execution package. Click **Yes**.

9. Click and drag the **Execute Project** level from the **All Steps** tree on the left, and drop it in the **Selected Steps** window on the right. Click OK.



Note: To remove an object from the Selected Steps window, right-click the object, and click **Remove Step**.

The execution package is now set up to execute the entire project.

If you want to create a package that deploys and executes only a part of a project, you can create an advanced execution package. For more information, see To Add Advanced Execution Packages.

Note: You will still need to schedule the execution process to run at specified times. For more information, see To Specify Execution Schedules.

To Add Advanced Execution Packages

An execution package determines which elements of a project will be executed. You can add advanced packages that execute only selected objects in the project. For example, you may want to create packages that execute only certain tables in the staging database, data warehouse, specific dimensions, and particular cubes.

1. On the **Execution** tab, right-click **Execution Packages**, and then select **Add Execution Package.**

2. In the **Name** field, type a name for the **Execution Package.**

3. In the **Action** field, specify what should happen if the execution fails. You have the following options:

FailPackage	Cancels the execution in case of failure
RestartPreviousStep	Restarts the execution from the previous step
RestartFirstStep	Restarts the execution process

4. If you select **RestartPreviousStep** or **RestartFirstStep**, specify in the Retries field, how many times you want the execution package to restart.

5. In the **Success** list, select the notification to use when the execution package succeeds.

Note: You have to create a notification before it is available from the list. See To Create Notifications

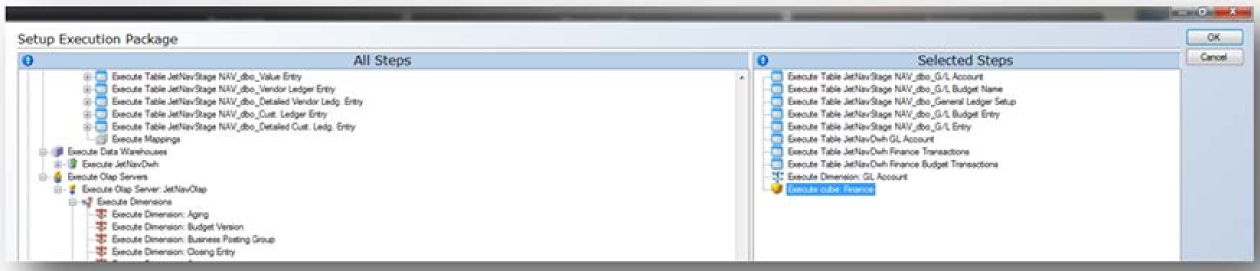
6. In the **Failure** list, select the notification to use when the execution package fails.

Note: You have to create a notification before it is available from the list. See To Create Notifications

7. Click OK.

8. You will be prompted if you want to set up the execution package. Click **Yes**.

A window will appear with the following settings:



All Steps

- Execute Table JetNavStage NAV_dbo_Value Entry
- Execute Table JetNavStage NAV_dbo_Vendor Ledger Entry
- Execute Table JetNavStage NAV_dbo_Detailed Vendor Ledg. Entry
- Execute Table JetNavStage NAV_dbo_Cust. Ledger Entry
- Execute Table JetNavStage NAV_dbo_Detailed Cust. Ledg. Entry
- Execute Mappings
- Execute Data Warehouses
  - Execute JetNavDwh
- Execute Olap Servers
  - Execute Olap Server: JetNavOlap
    - Execute Dimensions
      - Execute Dimension: Aging
      - Execute Dimension: Budget Version
      - Execute Dimension: Business Posting Group
      - Execute Dimension: Closing Entry

Selected Steps

- Execute Table JetNavStage NAV_dbo_G/L Account
- Execute Table JetNavStage NAV_dbo_G/L Budget Name
- Execute Table JetNavStage NAV_dbo_General Ledger Setup
- Execute Table JetNavStage NAV_dbo_G/L Budget Entry
- Execute Table JetNavStage NAV_dbo_G/L Entry
- Execute Table JetNavDwh GL Account
- Execute Table JetNavDwh Finance Transactions
- Execute Table JetNavDwh Finance Budget Transactions
- Execute Dimension: GL Account
- Execute cube: Finance

9. Click and drag the objects that should be executed from the **All Steps** window on the left and drop them in the **Selected Steps** window on the right. You can click and drag objects within the **Selected Window** to adjust the order in which they appear, which in turn will determine the order in which they are executed. Dragging over the Execute Project node will execute all objects in the staging database, data warehouse, and OLAP database. Click OK.

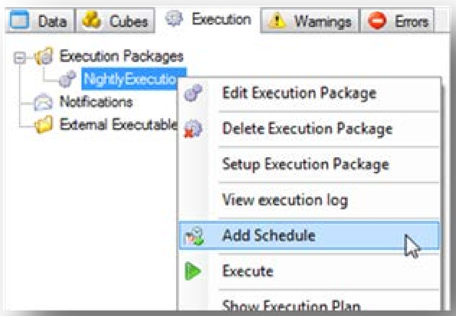
Note: To remove an object from the **Selected Window**, right-click the object and click **Remove Step**.

To Specify Execution Schedules

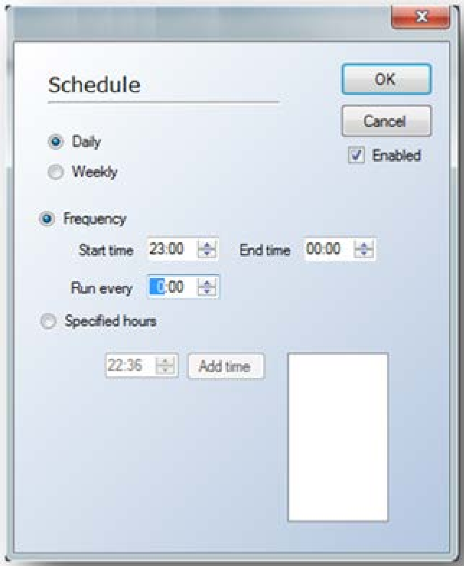
Projects can be executed several times daily, once a night, or several times during the week. When you specify an execution schedule, the execution process is started automatically at the specified time.

To Specify a Daily Execution Schedule

1. Expand **Execution Packages**, right-click the Execution Package to be modified, and select **Add Schedule**.



A window will appear with the following settings:


Schedule

[✓ Daily] Weekly

[✓ Frequency]

Start time 23:00 End time 00:00

Run every 0:00

Specified hours

22:36 Add time

OK

Cancel

[✓ Enabled]

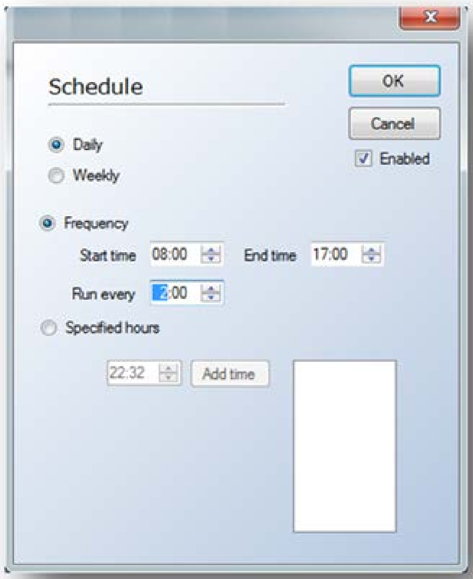
2. Click the **Daily**button.

3. The **Frequency** option will allow the user to specify the desired time interval for project execution. This will run the execution at a specified interval during the specified hours.

4. In the **Start time** field, enter the start time of the time interval.

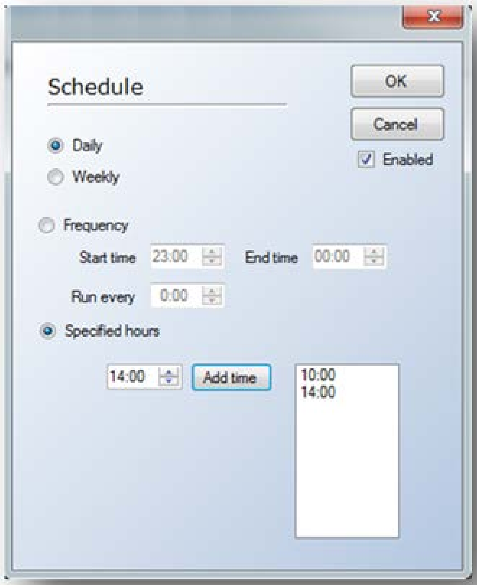
5. In the **End time** field, enter the end time of the interval.

6. In the **Run every** field, specify the number of hours and minutes between each project execution. The example below will run the execution package between 8:00AM and 5:00PM every two hours.



7. The **Specified Hours** option allows the user to select the exact times that the package will be executed.

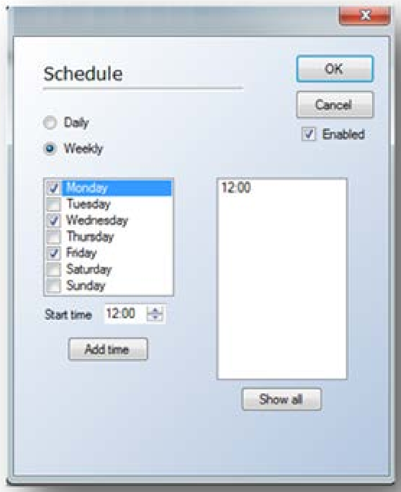
8. If using **Specified Hours**,enter the exact time for the execution to run, and then click **Add Time**. The example below will run the package at 10:00AM and 2:00PM.



9. Select **Enabled**to activate the schedule, and then click **OK**.

To Specify a Weekly Execution Schedule

1. Expand **Execution Packages**, right-click the **Execution Package**for modification, and select **Add Schedule**. In the **Schedule** window, select the **Weekly** button.



2. Select the day(s) when the project should execute.

3. Specify the start time for the execution, and then click **Add Time** to add the time to the schedule. Repeat this step for each day and time that you want to execute the project. A project can be executed several times a day and several times during the week.

4. To view the entire weekly schedule, click **Show All**.

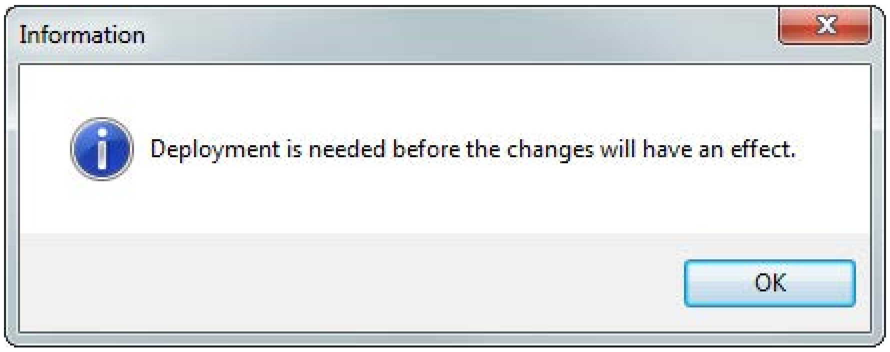
5. Select **Enabled** to activate the schedule, and then click **OK**.

Note: All events that happen during the execution are registered in the Windows Event Viewer if you are running Windows 2000 or Windows XP, or in the Windows Event Log if you are running Windows Vista or later.

Remind users to deploy the project after execution package changes

When adding or changing an execution package, the scheduler will only execute a deployed version of the project. This means that any object in the project needs to be successfully deployed after the execution package has been created.

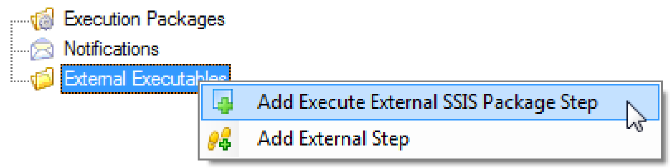
To avoid any mistakes around this issue, the following information that will be presented to the users after adding or changing an execution package:

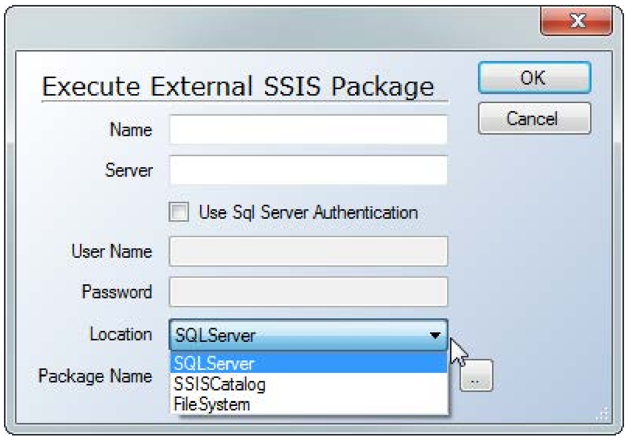


Deployment is needed before the changes will have an effect.

Execution of external SSIS packages

TX2014 supports execution of packages stores in SQL Server, Integration Services Catalogs (SQL 2012) and in the file system.





TX2014 SR1

RELEASE DOCUMENTATION

RELEASE DOCUMENTATION

TX2014 SR1

Contents

Introduction....2

New features....3

Managed Thread Execution....3

Execution Logging and Statistics....8

Index Automation....11

Remote Execution of SSIS Packages....14

Windows Services Management tool....21

Microsoft SQL Server 2014 Support....23

Improved features....24

Set Based Transformations....24

Hashed Primary Key field Control....29

Table and Index Compression Control....30

New Defaults for Conditional Lookup Fields....31

Support for Columnstore Indexes....32

Support for Unique indexes....34

Support for MDX Based OLAP Security....35

Support for extracting data from SQL Server without locks....37

Improved Data Source Synchronization....37

Bug Fixes in 14.2.1....38

Bug fixes in 14.2.2....40

Bug fixes in 14.2.3....40

Bug fixes in 14.2.4....40

Bug fixes in 14.2.5....41

Bug fixes in 14.2.6....41

Introduction

In February 2014, we introduced TX2014 as a serious player in the Data Warehouse Automation Platform segment.

With TX2014 Service Release 1 (SR1), we take another step in this direction. We have a strong commitment to the market and we aim to take every step necessary to maintain our position as the leading Data Warehouse Automation Platform for the Microsoft SQL Server Business Intelligence Platform.

Among the key additions to the software in TX2014 SR1, I will highlight the performance improvements, the Managed Thread execution and the fact that TX2014 SR1 is the first version of TX that supports the newly released Microsoft SQL Server 2014.

In this document, you can find detailed information about all the news and changes in TX2014 SR1.

We hope that you will find the new features useful and as always, we hope that you will find as much joy in using the software as we have in developing it.

Aarhus, September 4 2014

Thomas Duun

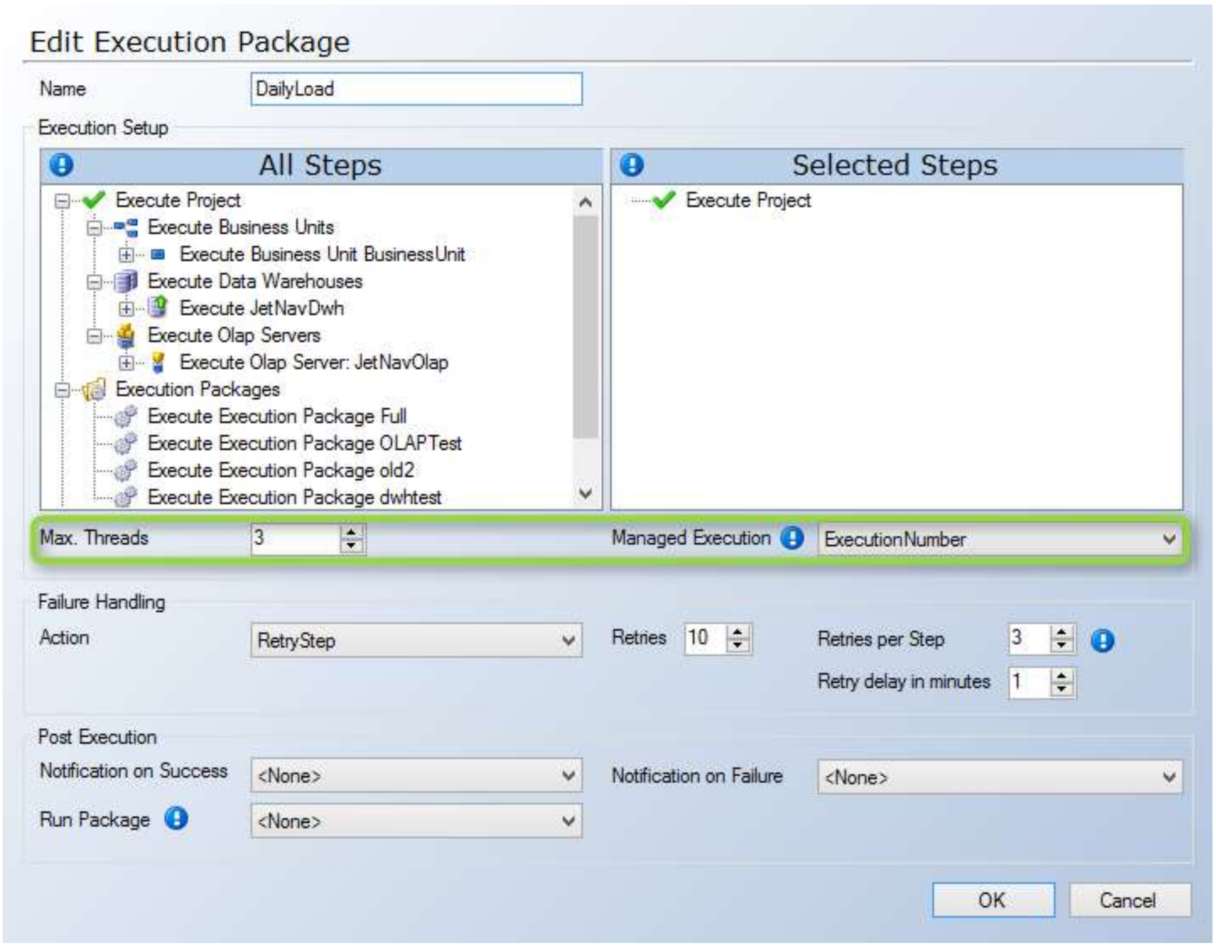
Product Manager
timeXtender Software

New features

Managed Thread Execution

An important task when building a Data Warehouse Solution is to make sure that all the individual ETL tasks are executed in the correct order. Getting the order right is absolute essential to get the correct end-result and since this can be quite complicated to manually keep track of all dependencies, most projects ends up being executed sequentially since this is the simplest way of controlling it. TX2014 is able to automate the execution of the project and can even help you optimize the execution flow by utilizing more threads and hence execute your project in parallel instead of serial.

Managed Thread Execution is controlled on the individual execution packages:



Edit Execution Package

Name: DailyLoad

Execution Setup

All Steps:
- Execute Project
  - Execute Business Units
    - Execute Business Unit BusinessUnit
  - Execute Data Warehouses
    - Execute JetNavDwh
  - Execute Olap Servers
    - Execute Olap Server: JetNavOlap
  - Execution Packages
    - Execute Execution Package Full
    - Execute Execution Package OLAPTest
    - Execute Execution Package old2
    - Execute Execution Package dwhtest

Selected Steps:
- Execute Project

Max. Threads: 3

Managed Execution: ExecutionNumber

Failure Handling

Action: RetryStep

Retries: 10

Retries per Step: 3

Retry delay in minutes: 1

Post Execution

Notification on Success: <None>

Notification on Failure: <None>

Run Package: <None>

Max. Threads: Select the maximum number of tasks that TX can run in parallel.

Managed Execution: When set to anything but Disable, the package will be Managed, meaning that TX will decide the order based on dependencies and the degree of parallelism specified by the Max threads setting.

It is possible to choose how TX should prioritize whenever more than one object is ready to be executed.

ExecutionNumber is the order of the objects in the project tree – the higher in the tree the sooner the execution.

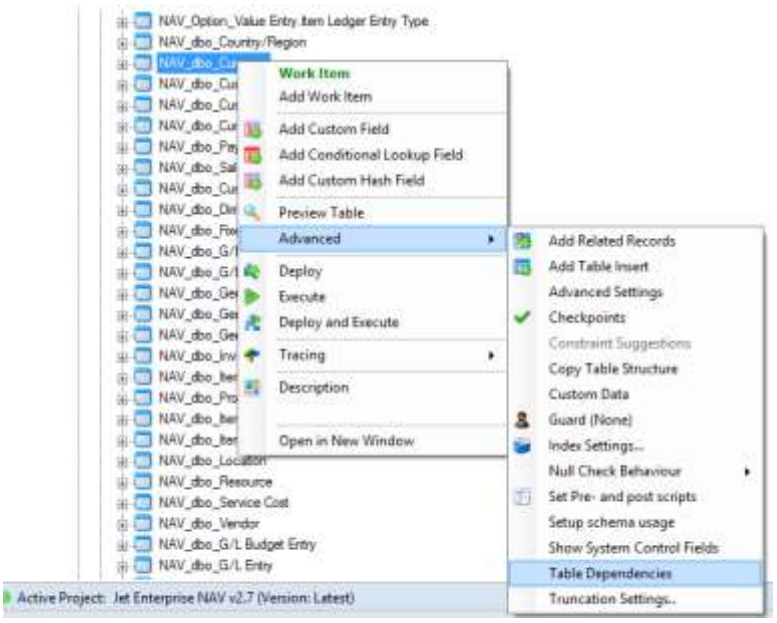
Classification means that TX will choose the order of objects based on their table classification. The order will be "Fact Table – Large", "Dimension Table – Large", "Fact Table", "Dimension Table". If two tables has the same classification TX will use the Execution number as the secondary criteria.

Execution Time means that TX will choose the object based on the average execution time for the object. If two objects has the same average execution time (i.e. new objects), then TX will use Execution Number as the secondary criteria.

The execution order is calculated based on dependencies. The dependency calculation considers the following structures in the project:

- Relations between tables (type Warning and Error)
- Conditional Lookup Fields
- Data Movements to DWH
- Table Inserts
- Add Related Records
- Standard Views
- Custom Views using Parameters

Since dependencies could be hidden in custom views or script commands, we have added the opportunity to define your own dependencies on any table, cube or dimension:



The dependencies are defined by placing a checkmark on the objects that the current object depends on:

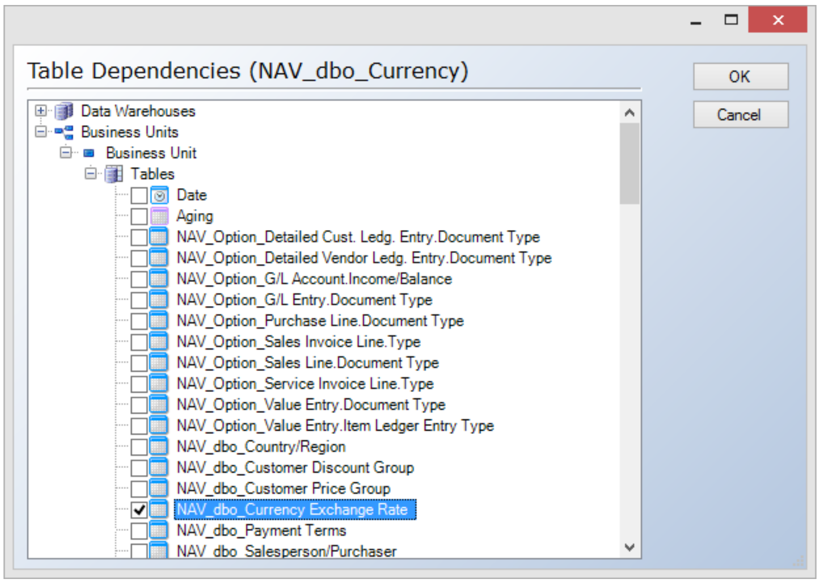
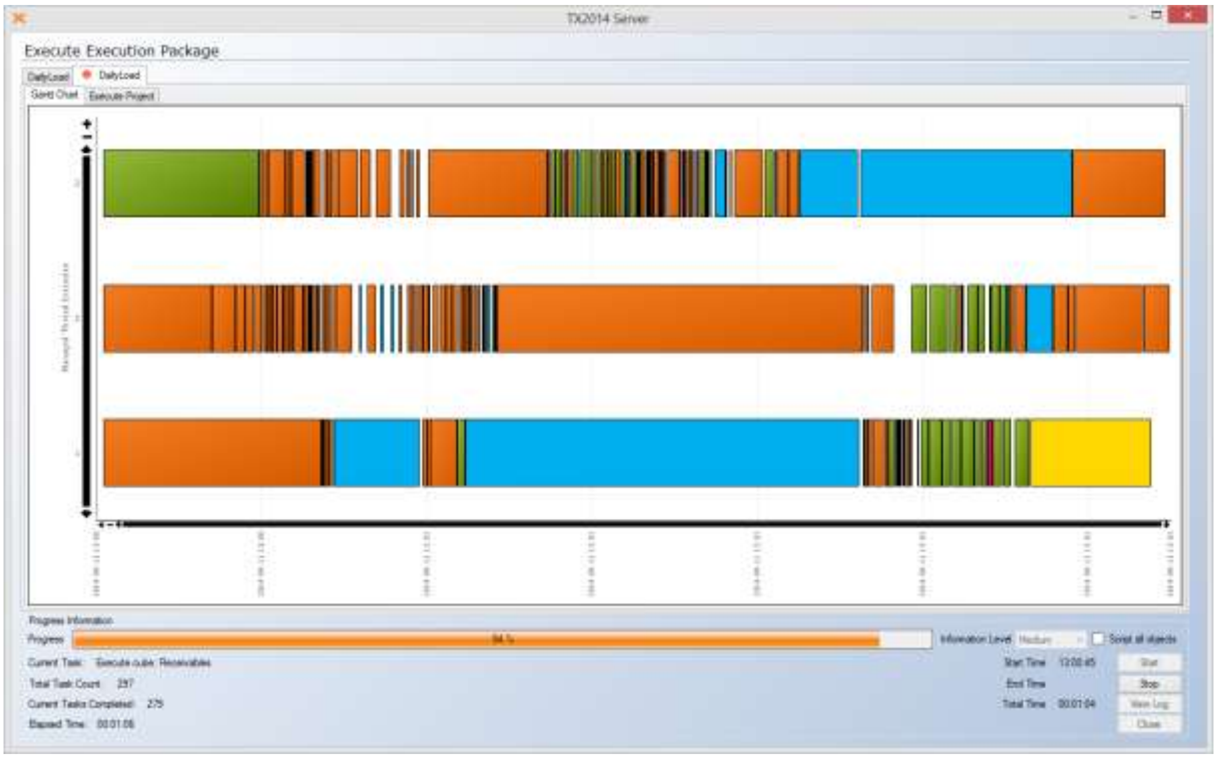


Table Dependencies (NAV_dbo_Currency)

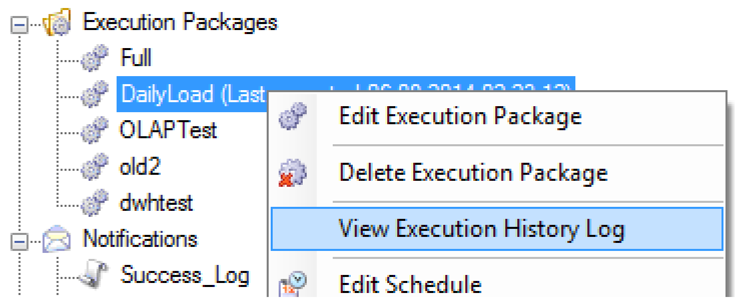
- Data Warehouses
- Business Units
  - Business Unit
    - Tables
      - [ ] Date
      - [ ] Aging
      - [ ] NAV_Option_Detailed Cust. Ledg. Entry.Document Type
      - [ ] NAV_Option_Detailed Vendor Ledg. Entry.Document Type
      - [ ] NAV_Option_G/L Account.Income/Balance
      - [ ] NAV_Option_G/L Entry.Document Type
      - [ ] NAV_Option_Purchase Line.Document Type
      - [ ] NAV_Option_Sales Invoice Line.Type
      - [ ] NAV_Option_Sales Line.Document Type
      - [ ] NAV_Option_Service Invoice Line.Type
      - [ ] NAV_Option_Value Entry.Document Type
      - [ ] NAV_Option_Value Entry.Item Ledger Entry Type
      - [ ] NAV_dbo_Country/Region
      - [ ] NAV_dbo_Customer Discount Group
      - [ ] NAV_dbo_Customer Price Group
      - [x] NAV_dbo_Currency Exchange Rate
      - [ ] NAV_dbo_Payment Terms
      - [ ] NAV_dbo_Salesperson/Purchaser

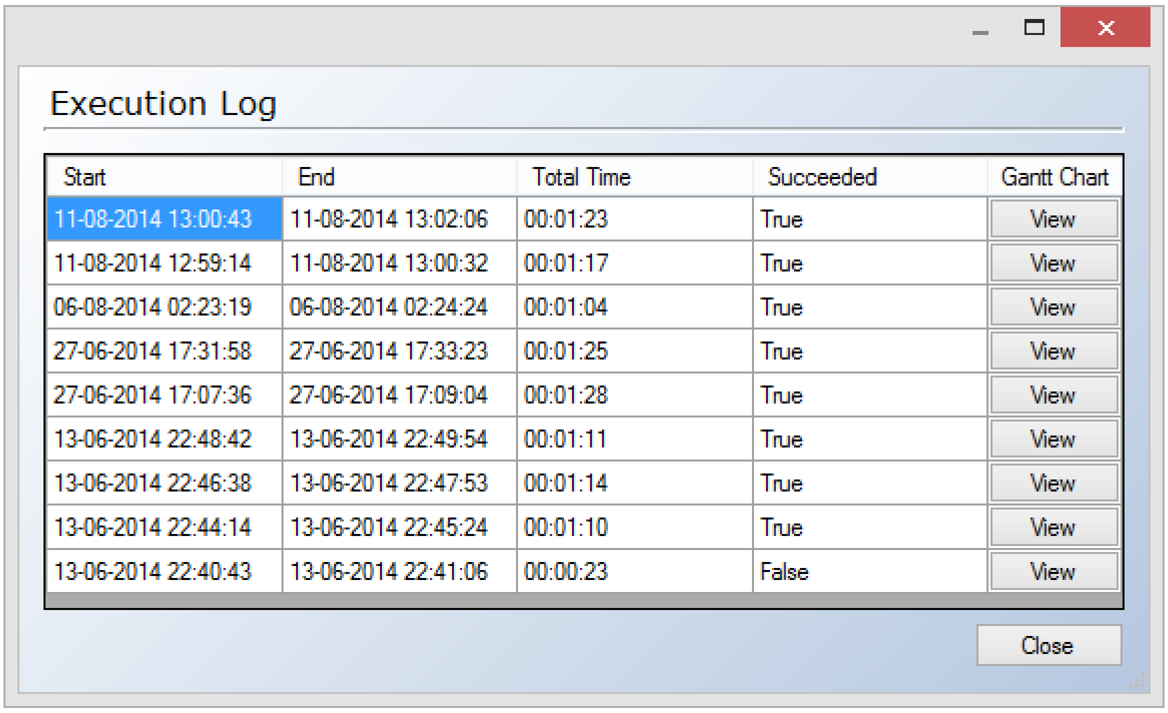
During execution you can monitor the progress and see what the individual threads are doing in a Gantt style chart:



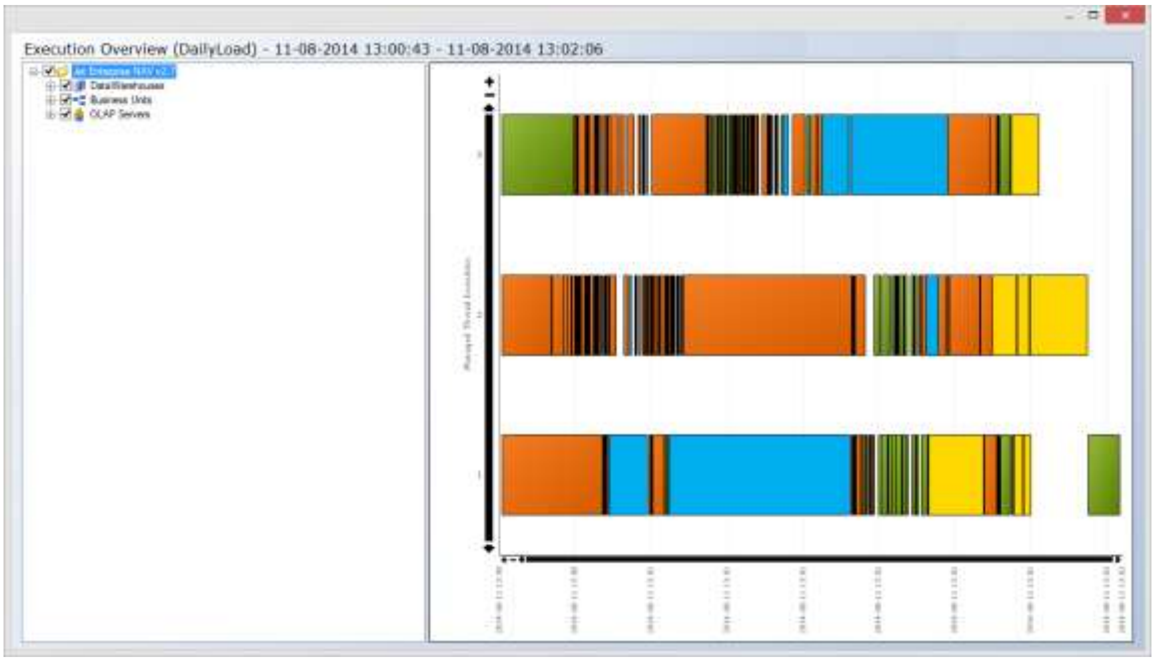
Progress: 94 %
Information Level: Medium
Script all objects
Current Task: Execute cube: Receivables
Total Task Count: 297
Current Tasks Completed: 279
Elapsed Time: 00:01:06
Start Time: 12:00:45
End Time:
Total Time: 00:01:04

During scheduled execution and after execution, the Gantt chart can be displayed through the **View Execution History Log** on the Execution Package:



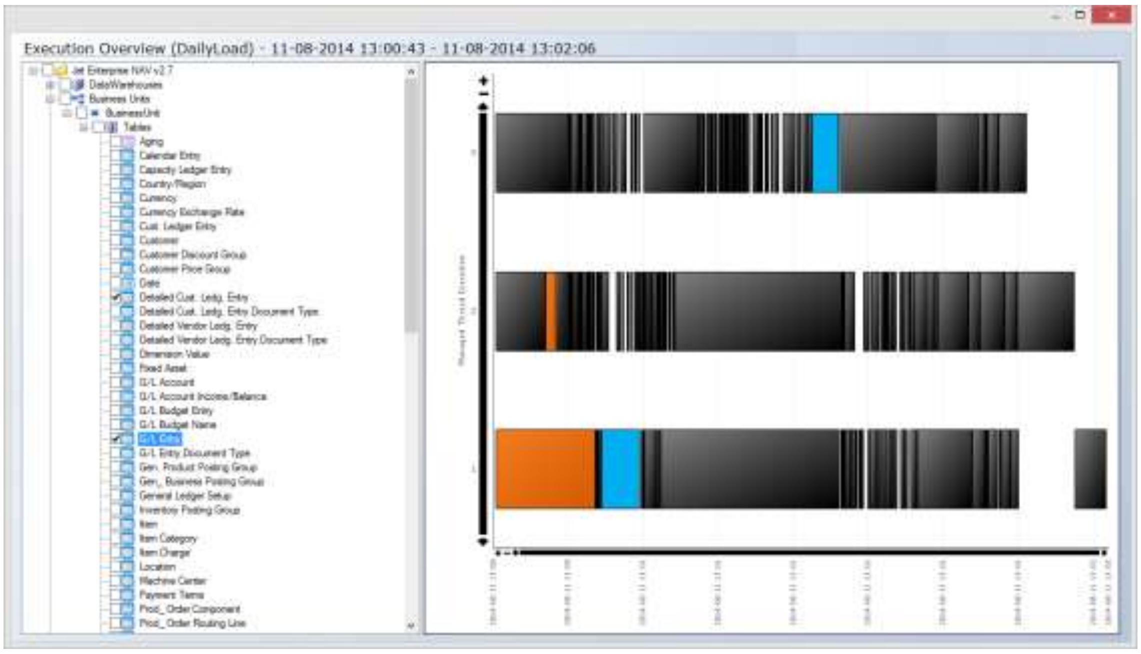

Execution Log

Start	End	Total Time	Succeeded	Gantt Chart
11-08-2014 13:00:43	11-08-2014 13:02:06	00:01:23	True	View
11-08-2014 12:59:14	11-08-2014 13:00:32	00:01:17	True	View
06-08-2014 02:23:19	06-08-2014 02:24:24	00:01:04	True	View
27-06-2014 17:31:58	27-06-2014 17:33:23	00:01:25	True	View
27-06-2014 17:07:36	27-06-2014 17:09:04	00:01:28	True	View
13-06-2014 22:48:42	13-06-2014 22:49:54	00:01:11	True	View
13-06-2014 22:46:38	13-06-2014 22:47:53	00:01:14	True	View
13-06-2014 22:44:14	13-06-2014 22:45:24	00:01:10	True	View
13-06-2014 22:40:43	13-06-2014 22:41:06	00:00:23	False	View


Execution Overview (DailyLoad) - 11-08-2014 13:00:43 - 11-08-2014 13:02:06

- Jet Enterprise NAV v2.7
  - DataWarehouses
  - Business Units
  - OLAP Servers

In the Gantt chart, you can hover the mouse over the individual task to see the details about the object in a tooltip – and if you select one or multiple objects in the right side panel, checked objects will be colored, all other objects will be "Grayed":


Execution Overview (DailyLoad) - 11-08-2014 13:00:43 - 11-08-2014 13:02:06

- Jet Enterprise NAV v2.7
  - DataWarehouses
  - Business Units
    - BusinessUnit
      - Tables
        - Aging
        - Calendar Entry
        - Capacity Ledger Entry
        - Country/Region
        - Currency
        - Currency Exchange Rate
        - Cust. Ledger Entry
        - Customer
        - Customer Discount Group
        - Customer Price Group
        - Date
        - ✓ Detailed Cust. Ledg. Entry
        - Detailed Cust. Ledg. Entry Document Type
        - Detailed Vendor Ledg. Entry
        - Detailed Vendor Ledg. Entry Document Type
        - Dimension Value
        - Fixed Asset
        - G/L Account
        - G/L Account Income/Balance
        - G/L Budget Entry
        - G/L Budget Name
        - ✓ G/L Entry
        - G/L Entry Document Type
        - Gen. Product Posting Group
        - Gen._ Business Posting Group
        - General Ledger Setup
        - Inventory Posting Group
        - Item
        - Item Category
        - Item Charge
        - Location
        - Machine Center
        - Payment Terms
        - Prod._ Order Component
        - Prod._ Order Routing Line

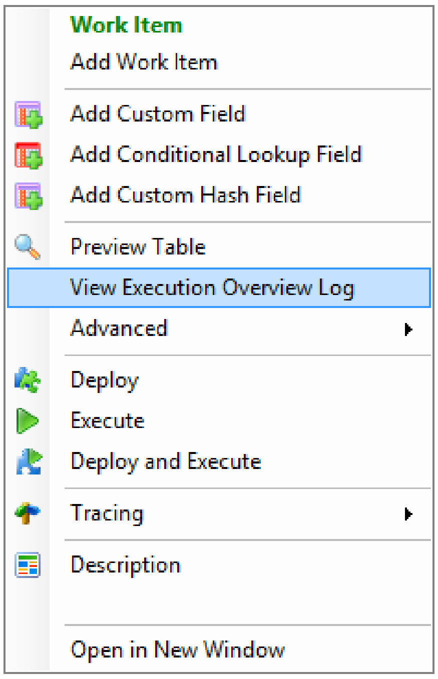
Be aware that execution of tasks in parallel will increase the need for database log file space - do not be worried if the size of the log file starts overgrowing the actual data file size. Do not be tempted to force automatically shrinking of the log file on a regular basis since the file size is actually needed and forcing SQL Server to grow the file on every execution will not do any good for your project execution performance.

Execution Logging and Statistics

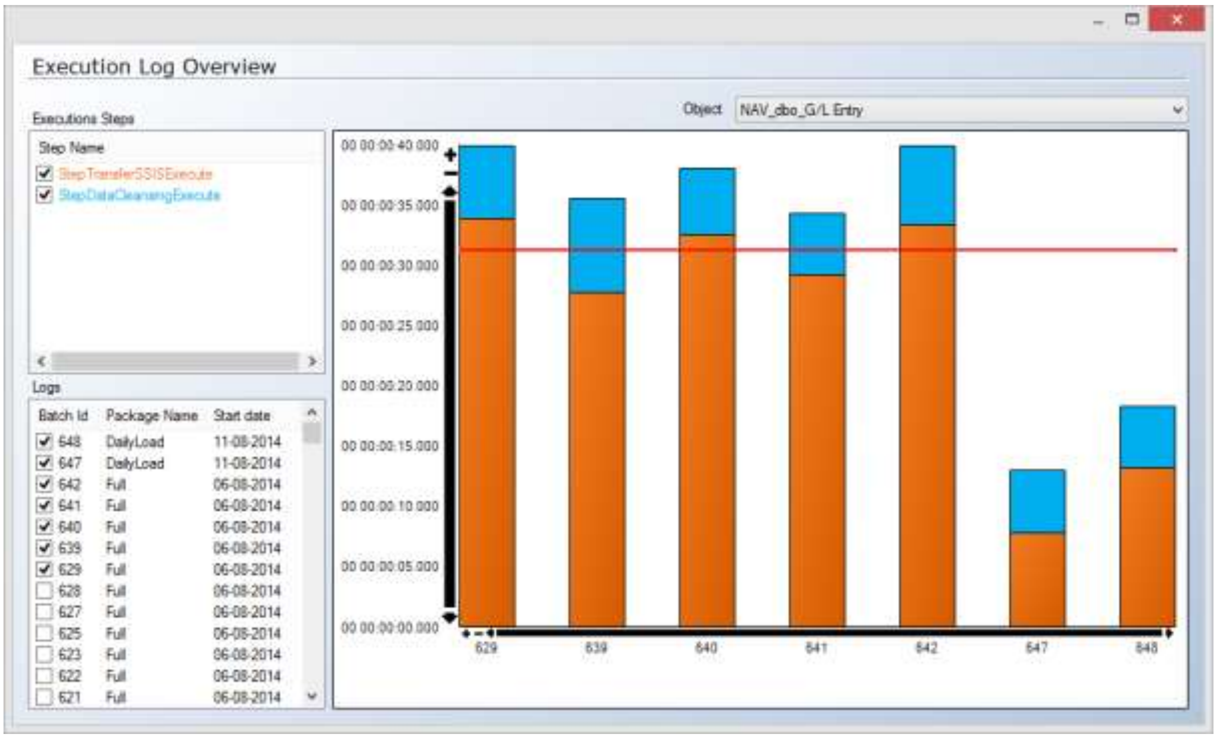
When a table, dimension or cube is executed through an execution package, TX now logs the start- and endtime for each step to a table in the repository.

The log is being utilized by the Managed Thread Execution when using the “Execution Time” method.

We have also added the option to view the execution log overview for the individual objects:



By default the last 7 executions are selected – this number can be changed in Tools / General Settings on the "Execution Log Setting" tab-page.



Execution Log Overview

Executions Steps

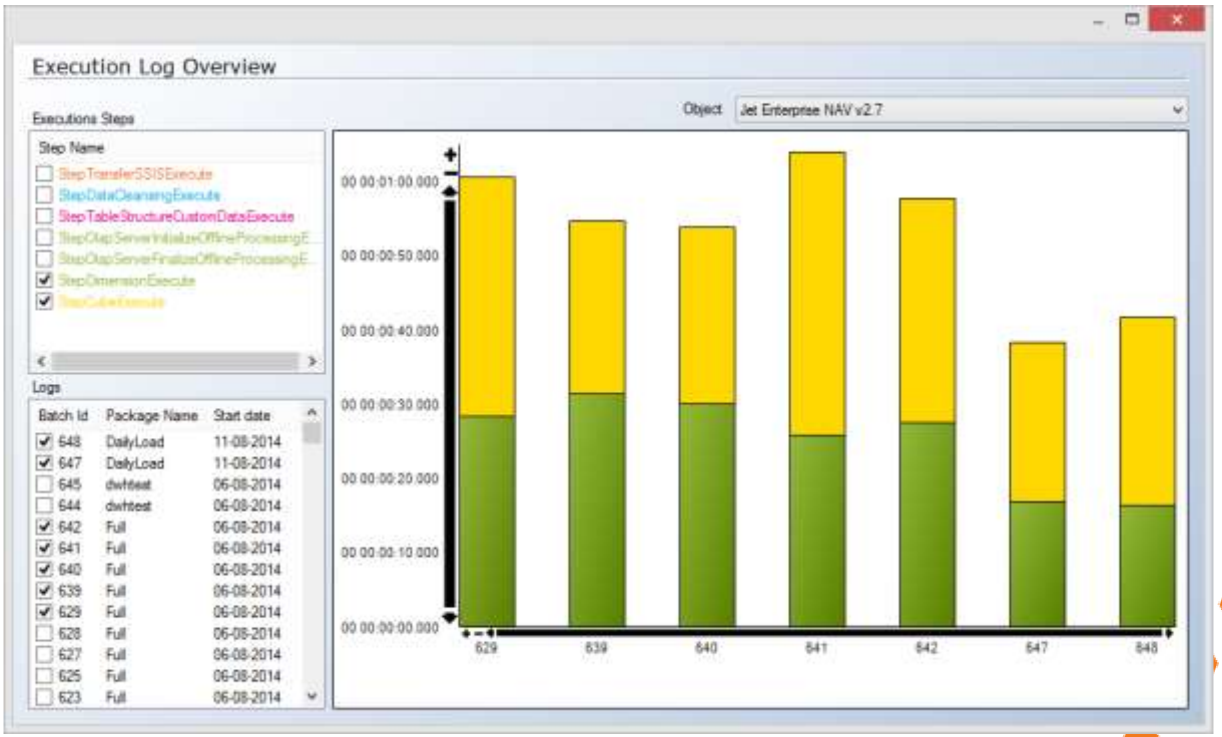
Object: NAV_dbo_G/L Entry

	Step Name
✓	StepTransferSSISExecute
✓	StepDataCleansingExecute

Logs

	Batch Id	Package Name	Start date
✓	648	DailyLoad	11-08-2014
✓	647	DailyLoad	11-08-2014
✓	642	Full	06-08-2014
✓	641	Full	06-08-2014
✓	640	Full	06-08-2014
✓	639	Full	06-08-2014
✓	629	Full	06-08-2014
	628	Full	06-08-2014
	627	Full	06-08-2014
	625	Full	06-08-2014
	623	Full	06-08-2014
	622	Full	06-08-2014
	621	Full	06-08-2014

You can select which steps to see in the bar-chart by selecting / deselecting the individual steps in the Execution Steps dialogue.



Execution Log Overview

Executions Steps

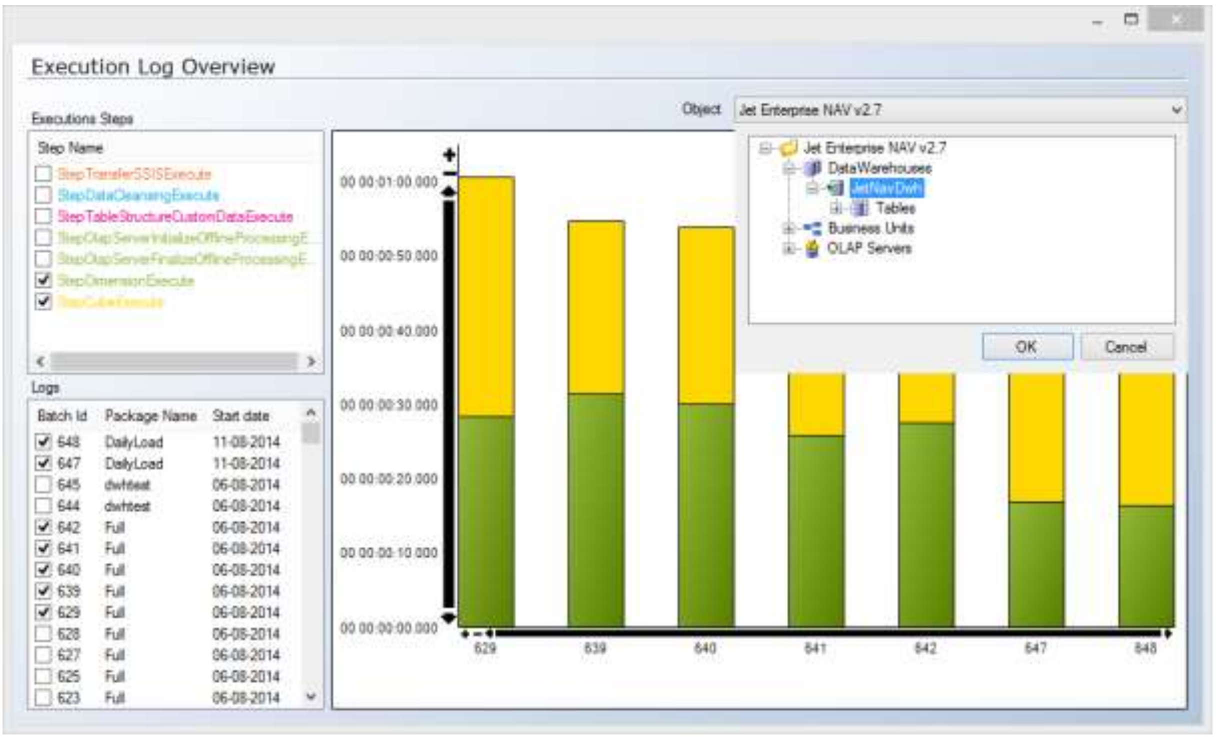
Object: Jet Enterprise NAV v2.7

	Step Name
	StepTransferSSISExecute
	StepDataCleansingExecute
	StepTableStructureCustomDataExecute
	StepOlapServerInitializeOfflineProcessingE
	StepOlapServerFinalizeOfflineProcessingE
✓	StepDimensionExecute
✓	StepCubeExecute

Logs

	Batch Id	Package Name	Start date
✓	648	DailyLoad	11-08-2014
✓	647	DailyLoad	11-08-2014
	645	dwhtest	06-08-2014
	644	dwhtest	06-08-2014
✓	642	Full	06-08-2014
✓	641	Full	06-08-2014
✓	640	Full	06-08-2014
✓	639	Full	06-08-2014
✓	629	Full	06-08-2014
	628	Full	06-08-2014
	627	Full	06-08-2014
	625	Full	06-08-2014
	623	Full	06-08-2014

You can select which object to show statistics for in the "Objects" drop-down:



Step Name
☐ StepTransferSSISExecute
☐ StepDataCleansingExecute
☐ StepTableStructureCustomDataExecute
☐ StepOlapServerInitializeOfflineProcessingE…
☐ StepOlapServerFinalizeOfflineProcessingE…
✓ StepDimensionExecute
✓ StepCubeExecute

Logs

Batch Id	Package Name	Start date
✓ 648	DailyLoad	11-08-2014
✓ 647	DailyLoad	11-08-2014
☐ 645	dwhtest	06-08-2014
☐ 644	dwhtest	06-08-2014
✓ 642	Full	06-08-2014
✓ 641	Full	06-08-2014
✓ 640	Full	06-08-2014
✓ 639	Full	06-08-2014
✓ 629	Full	06-08-2014
☐ 628	Full	06-08-2014
☐ 627	Full	06-08-2014
☐ 625	Full	06-08-2014
☐ 623	Full	06-08-2014

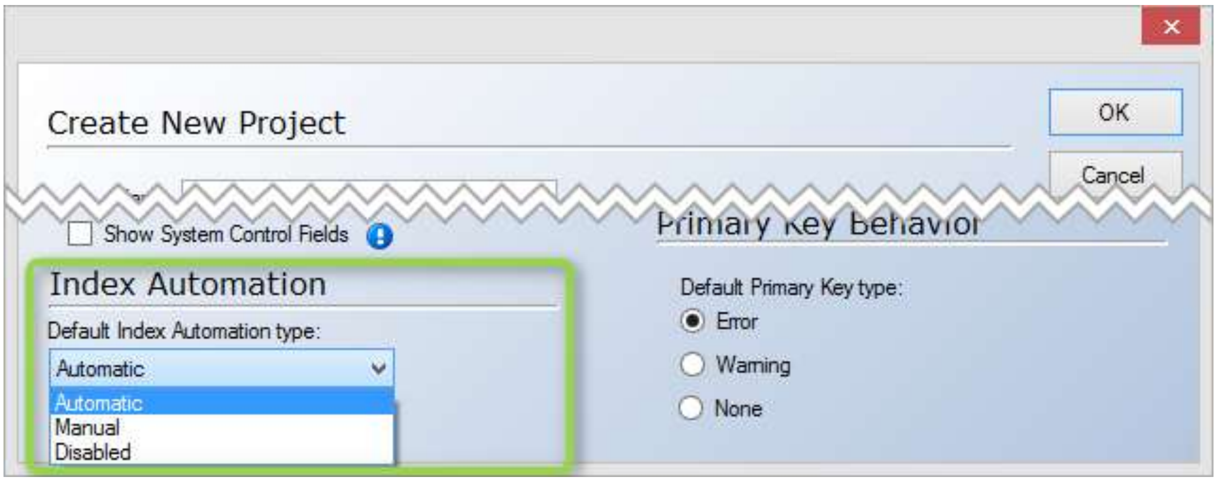
Index Automation

Index Automation is a new way of working with Indexes in TX. In earlier versions of TX, indexes was created in the background, not visible for the end-user and the generation of indexes was not optimized, but instead seen as an isolated need for a specific operation.

Index Automation considers the following when designing indexes for the project:

- Relations between tables with relationship type set to Error or Warning
- Joins on conditional lookup fields
- Primary Key fields (On Raw Table)
- Selection Rules on the Data Warehouse
- Incremental Selection Rule on the Data Warehouse
- Partitioning fields (DW_Partitionkey, DW_TimeStamp)

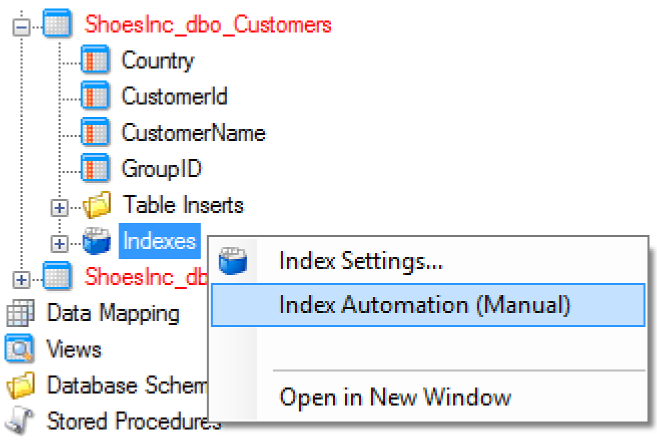
The Index Automation can be set on the Project level when creating a new project or through the Edit Project dialogue:



Index Automation can be enabled in an always-on state (Automatic), a per-request state (Manual) or disabled.

When set to "Automatic", the index automation is updating the indexes whenever the user changes the project in a way that could trigger a new or altered index.

When set to "Manual", the user has to activate the Index Automation for every table where TX should create indexes. The Index automation is activated on the right click menu for the table (if the index-node is not present) – or on the Index Node:



When set to disabled, TX will behave like in earlier versions and create individual indexes in the Data Cleansing Procedures when needed.

The default setting on the project will be inherited on all tables, but you can defer from this by changing the setting on the individual tables, through the Advanced / Advanced Settings:

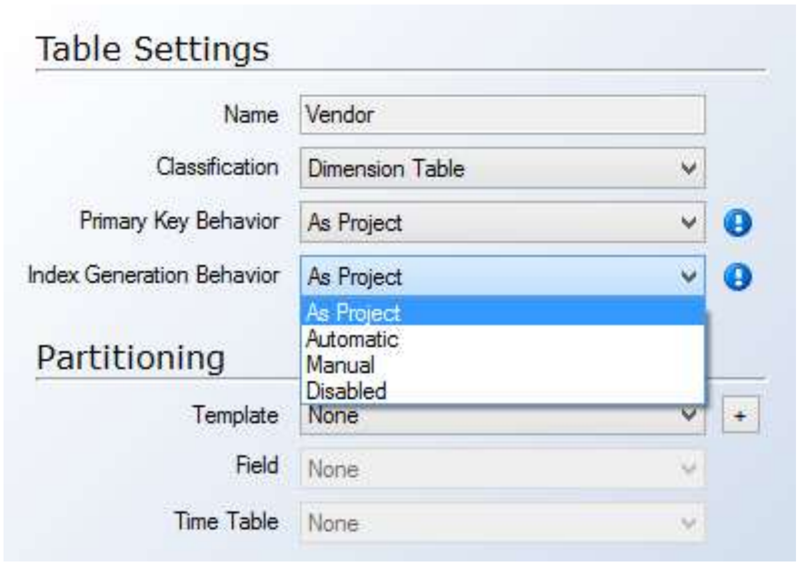


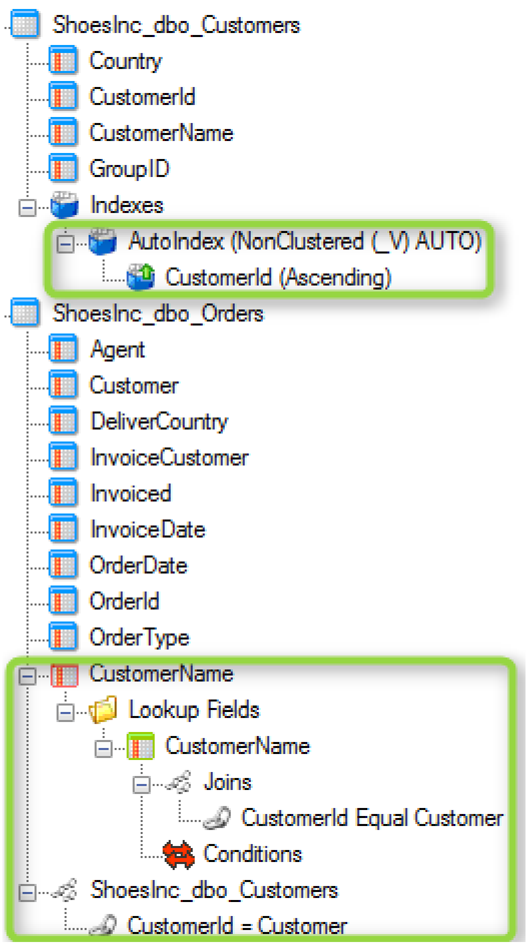
Table Settings

Name: Vendor
Classification: Dimension Table
Primary Key Behavior: As Project
Index Generation Behavior: As Project

Partitioning

Template: None
Field: None
Time Table: None

The indexes created by Index Automation will be named AutoIndex and postfixed with a number for uniqueness within each table.


ShoesInc_dbo_Customers
Country
CustomerId
CustomerName
GroupID
Indexes
AutoIndex (NonClustered (_V) AUTO)
CustomerId (Ascending)
ShoesInc_dbo_Orders
Agent
Customer
DeliverCountry
InvoiceCustomer
Invoiced
InvoiceDate
OrderDate
OrderId
OrderType
CustomerName
Lookup Fields
CustomerName
Joins
CustomerId Equal Customer
Conditions
ShoesInc_dbo_Customers
CustomerId = Customer

Index Automation will try to minimize the number of needed indexes. If two lookups can utilize the same index, TX will take advantage of that.

When calculating which indexes are needed, TX takes your manually created indexes into consideration. It will not change your manually created indexes, but it will use them instead of creating similar indexes.

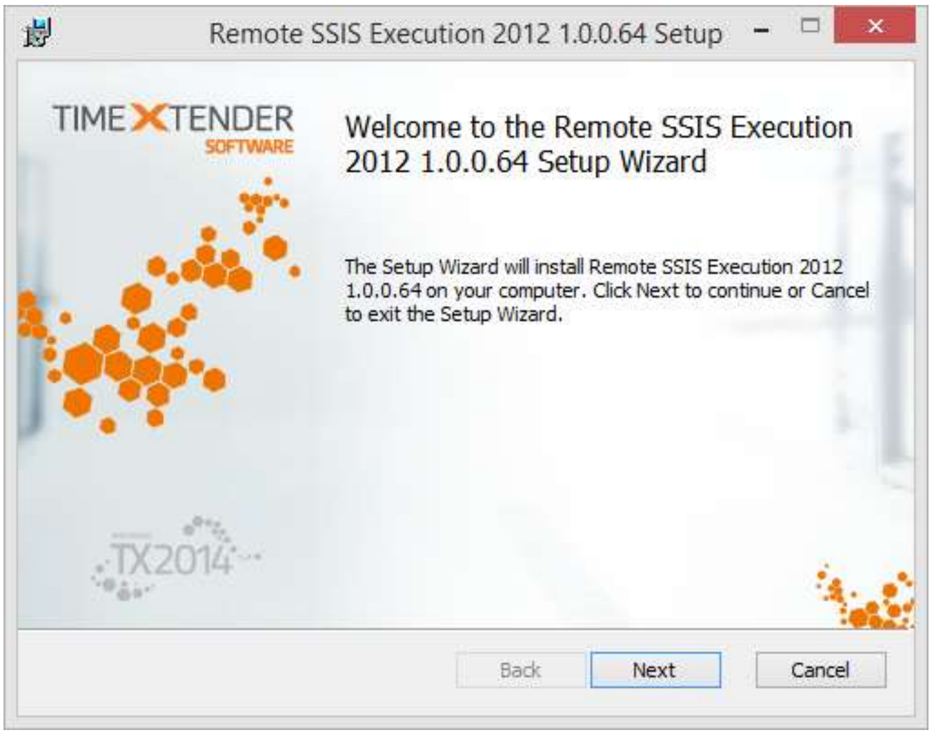
Remote Execution of SSIS Packages

When TX is installed on a separate server and not on the actual BI SQL Server, then all network traffic when transferring data to the staging database(s) and data warehouse(s) is routed through the server where TX is installed. This is because the SSIS package is being opened and executed by TX on the separate server. Even in a high-speed LAN, this is considerable slower than keeping the traffic local on the BI Server, for instance when moving data from staging area(s) to data warehouse(s). With the Remote Execution of SSIS Packages feature, you can force TX to send the SSIS package to another server where it will be executed – typically on the actual BI SQL Server.

To use Remote Execution of SSIS Packages you must first install the Remote SSIS Execution Service in a version matching the Microsoft SQL Server Version (2005, 2008, 2012 or 2014). The installation package can be downloaded from the timeXtender Support Site: http://support.timextender.com/forums/20474987-Downloads.

To Install the service, unzip the installation package and run Setup.exe.

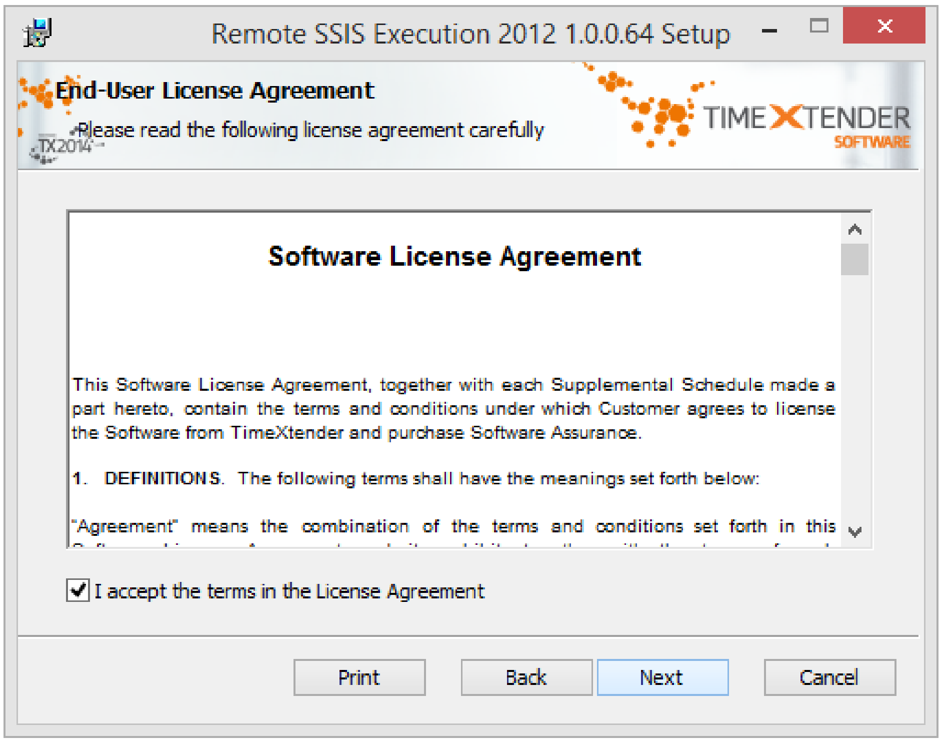
On the Welcome Screen, click Next:


Remote SSIS Execution 2012 1.0.0.64 Setup

Welcome to the Remote SSIS Execution 2012 1.0.0.64 Setup Wizard

The Setup Wizard will install Remote SSIS Execution 2012 1.0.0.64 on your computer. Click Next to continue or Cancel to exit the Setup Wizard.

Accept the Software License Agreement and click Next:


Remote SSIS Execution 2012 1.0.0.64 Setup
End-User License Agreement
Please read the following license agreement carefully
TIMEXTENDER SOFTWARE

Software License Agreement

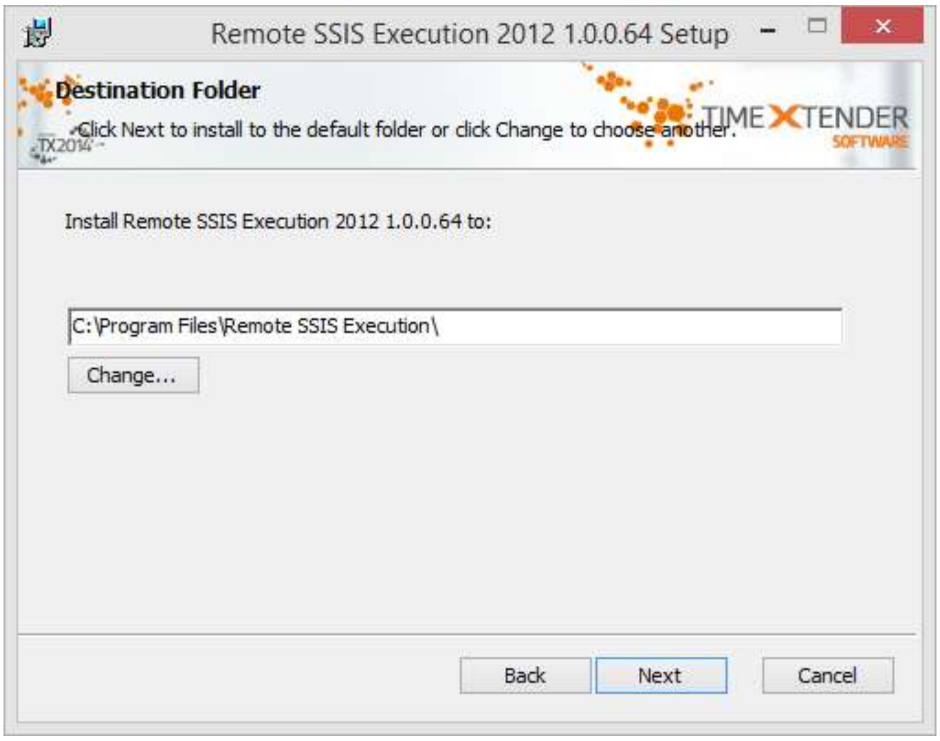
This Software License Agreement, together with each Supplemental Schedule made a part hereto, contain the terms and conditions under which Customer agrees to license the Software from TimeXtender and purchase Software Assurance.

1. **DEFINITIONS.** The following terms shall have the meanings set forth below:

"Agreement" means the combination of the terms and conditions set forth in this

☑ I accept the terms in the License Agreement

Select Installation Path and click Next:

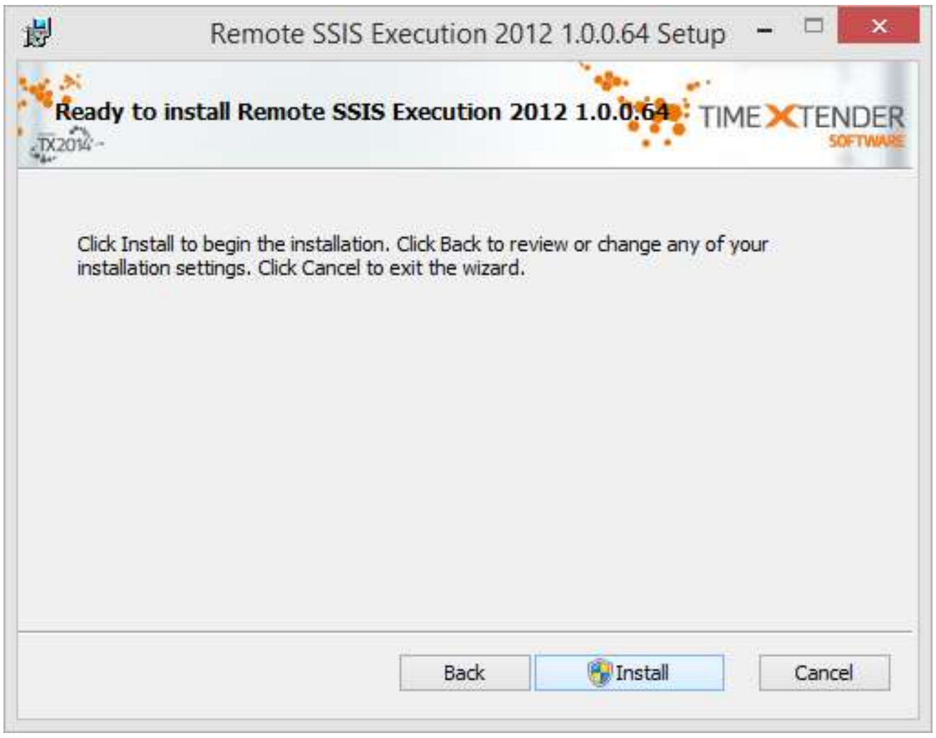

Remote SSIS Execution 2012 1.0.0.64 Setup
Destination Folder
Click Next to install to the default folder or click Change to choose another.
TIMEXTENDER SOFTWARE

Install Remote SSIS Execution 2012 1.0.0.64 to:

C:\Program Files\Remote SSIS Execution\

Change...
Back
Next
Cancel

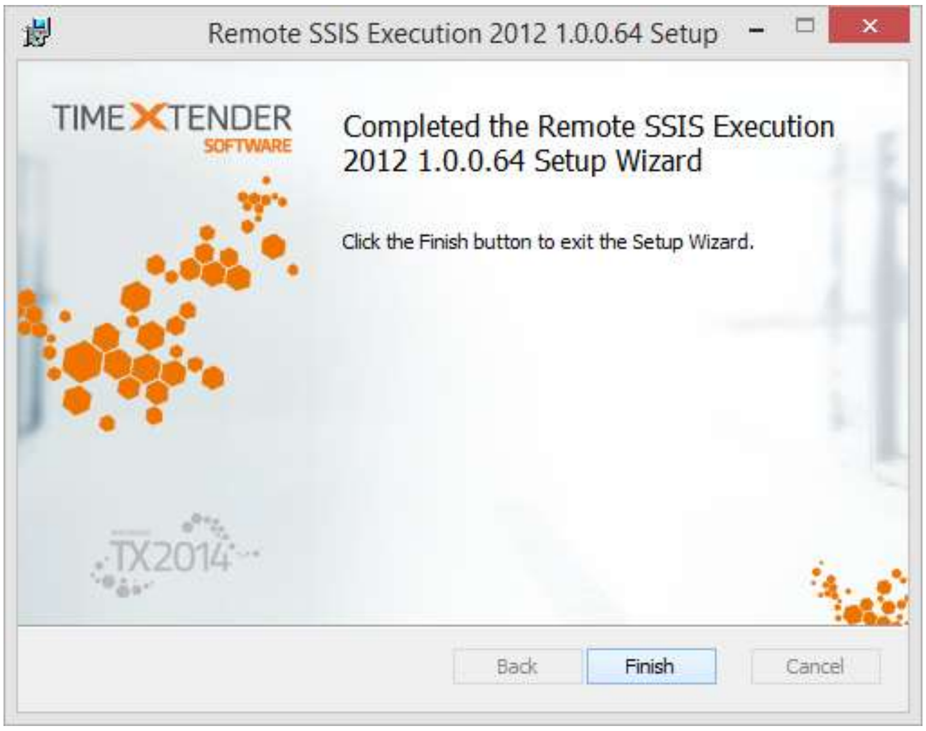
Click Install:



Ready to install Remote SSIS Execution 2012 1.0.0.64

Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.

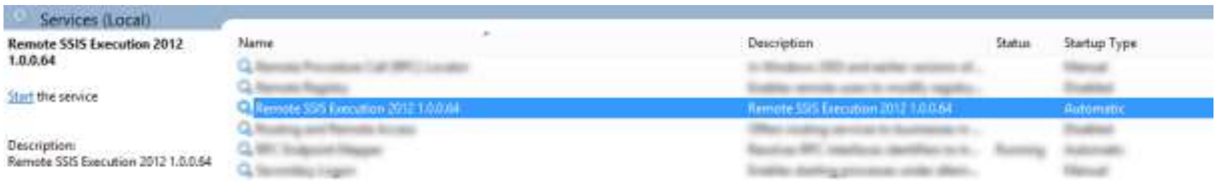
Click Finish:



Completed the Remote SSIS Execution 2012 1.0.0.64 Setup Wizard

Click the Finish button to exit the Setup Wizard.

Start the Windows Services manager (Start -> Run -> Services.msc), Locate the **Remote SSIS Execution Service**, right-click on it and select **Properties**:



Services (Local)

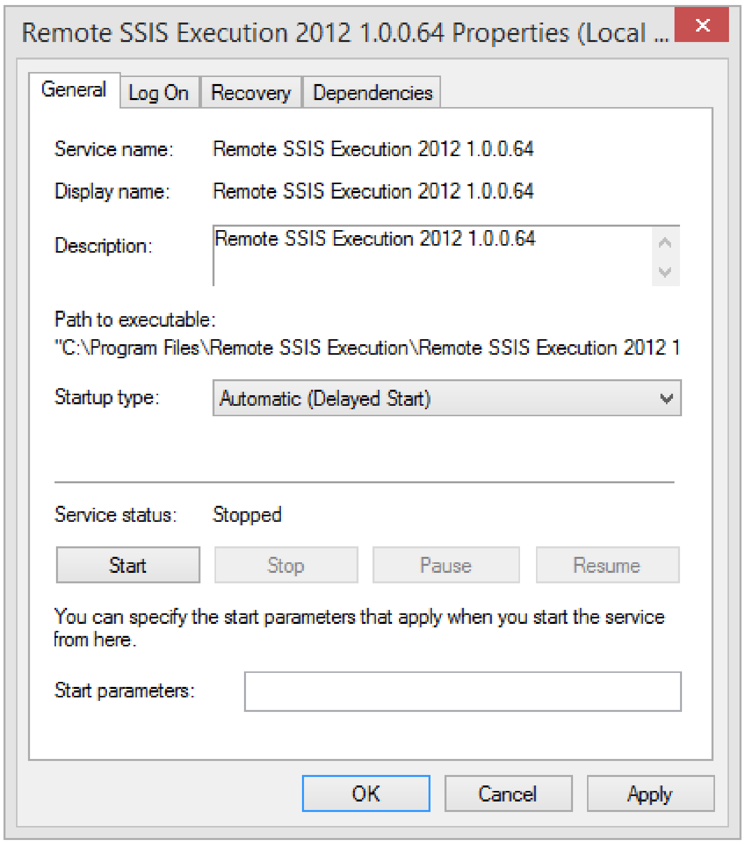
Remote SSIS Execution 2012 1.0.0.64

Start the service

Description:
Remote SSIS Execution 2012 1.0.0.64

Name	Description	Status	Startup Type
Remote SSIS Execution 2012 1.0.0.64	Remote SSIS Execution 2012 1.0.0.64		Automatic

Set the **Startup Type** to "Automatic (Delayed Start)" to ensure that the service is started after the SQL Server:



Remote SSIS Execution 2012 1.0.0.64 Properties (Local ...

Service name: Remote SSIS Execution 2012 1.0.0.64

Display name: Remote SSIS Execution 2012 1.0.0.64

Description: Remote SSIS Execution 2012 1.0.0.64

Path to executable:
"C:\Program Files\Remote SSIS Execution\Remote SSIS Execution 2012 1

Startup type: Automatic (Delayed Start)

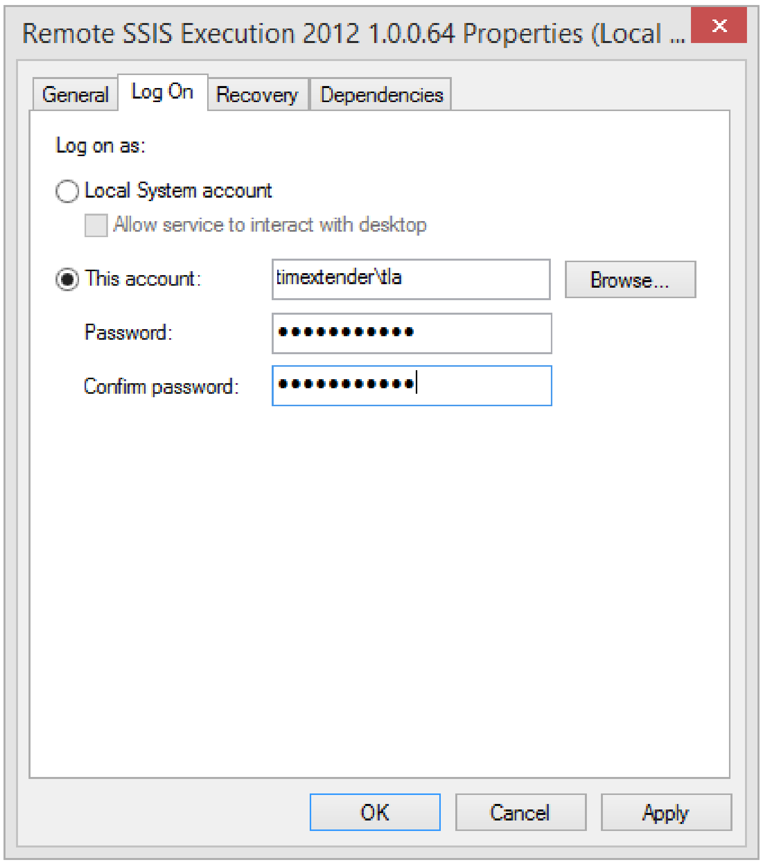
Service status: Stopped

You can specify the start parameters that apply when you start the service from here.

Start parameters:

Open the **Log On** tab and enter a **Username** and a **Password** for the user that should be running the Service.

The service should be using a login that holds permissions to execute SSIS packages and – if using Integrated Security – read permissions on all data sources and read/write permissions on the Staging database(s) and data warehouse(s).


Remote SSIS Execution 2012 1.0.0.64 Properties (Local ...

Log on as:

( ) Local System account

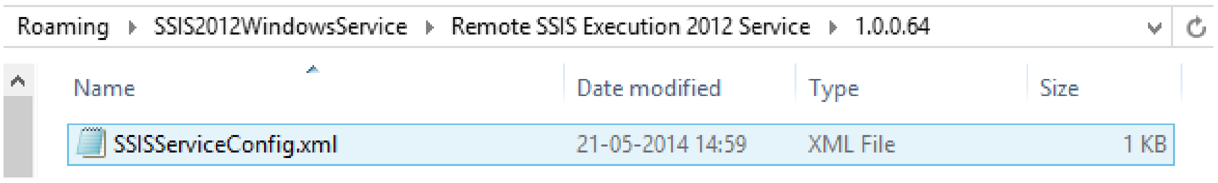
[ ] Allow service to interact with desktop

[● This account:] timextender\tla

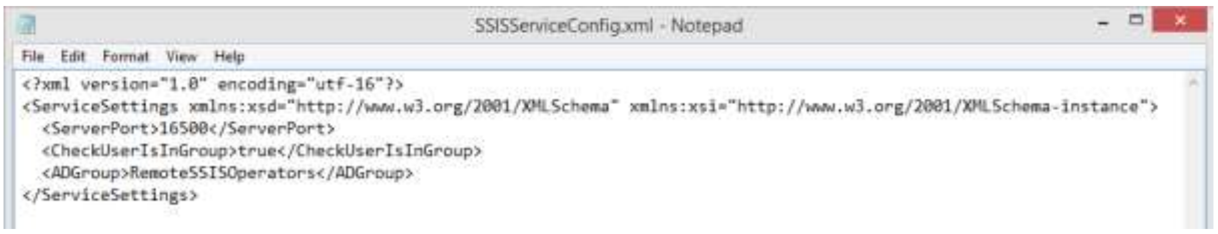
Password: ●●●●●●●●●●●

Confirm password: ●●●●●●●●●●●

By default, the service is configured to listen on TCP port 16500, but this can be changed in the SSISServiceConfig.xml file located in the service account user's %APPDATA%\Roaming\SSISWindowsService\Remote SSIS Execution Service\<Version Number> folder. You need to start the service once to get the config-file created.


Roaming ▸ SSIS2012WindowsService ▸ Remote SSIS Execution 2012 Service ▸ 1.0.0.64

Name	Date modified	Type	Size
SSISServiceConfig.xml	21-05-2014 14:59	XML File	1 KB



```
<?xml version="1.0" encoding="utf-16"?>
<ServiceSettings xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <ServerPort>16500</ServerPort>
  <CheckUserIsInGroup>true</CheckUserIsInGroup>
  <ADGroup>RemoteSSISOperators</ADGroup>
</ServiceSettings>
```

ServerPort: Specify the Port Number that the service should listen on.

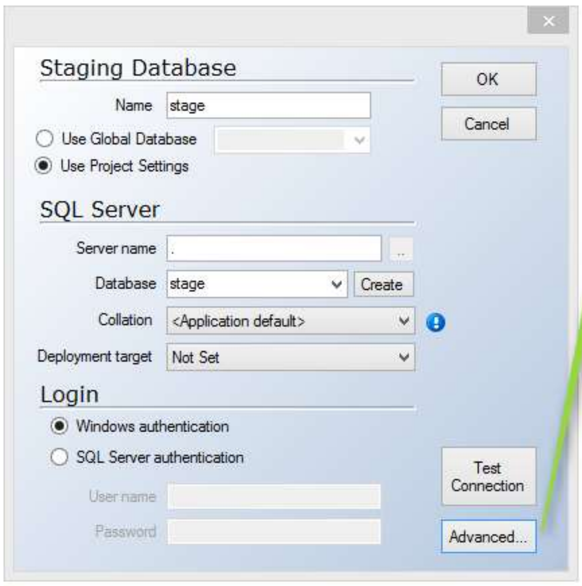
CheckUserIsInGroup: If set to True, the service will only allow members of the AD Security Group specified in <ADGroup> to use the service. If set to False, all users can use the service. Default Setting is True.

ADGroup: Specify which Active Directory Security Group that users has to be members of in order to use the service. Default Setting is RemoteSSISOperators.

Save the file and restart the service to make it run with the new configuration parameters.

By default, the service requires the user calling the service to be member of a specific Active Directory Group. This requirement along with the name of the group can be changed in the services configuration file: SSISServiceConfig.xml.

When the service is installed and configured, remote SSIS execution can be enabled from the individual Staging Database or Data Warehouse:


Staging Database

Name: stage

( ) Use Global Database

(●) Use Project Settings

SQL Server

Server name: .

Database: stage

Collation: <Application default>

Deployment target: Not Set

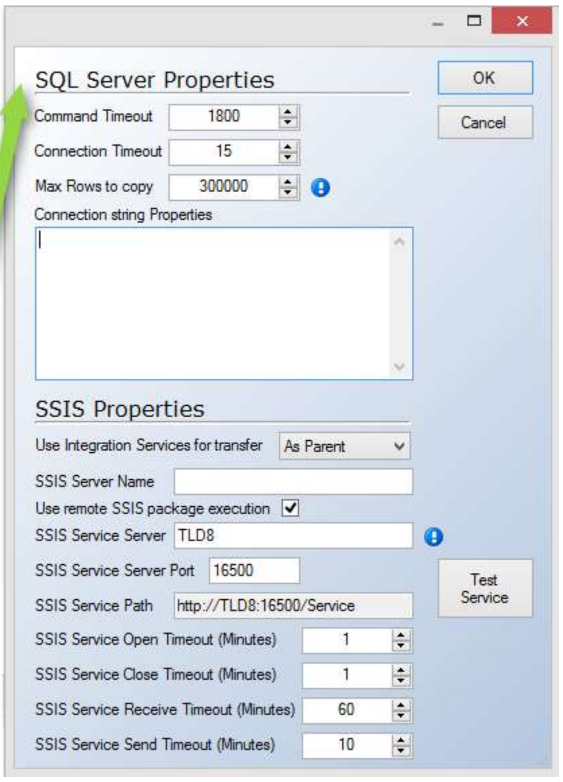
Login

(●) Windows authentication

( ) SQL Server authentication

User name:

Password:


SQL Server Properties

Command Timeout: 1800

Connection Timeout: 15

Max Rows to copy: 300000

Connection string Properties:

SSIS Properties

Use Integration Services for transfer: As Parent

SSIS Server Name:

Use remote SSIS package execution: ✓

SSIS Service Server: TLD8

SSIS Service Server Port: 16500

SSIS Service Path: http://TLD8:16500/Service

SSIS Service Open Timeout (Minutes): 1

SSIS Service Close Timeout (Minutes): 1

SSIS Service Receive Timeout (Minutes): 60

SSIS Service Send Timeout (Minutes): 10

Use Remote SSIS Package Execution: If checked, TX will send the package to the service for remote execution.

SSIS Service Server: Name of the server where the service is running, please write the complete URL including https:// if you want to use https.

SSIS Service Path: Information showing the Path. Can not be changed.

SSIS Service Open Timeout (Minutes): used when opening channels when no explicit timeout value is specified.

SSIS Service Close Timeout (Minutes): used when closing channels when no explicit timeout value is specified.

SSIS Service Receive Timout (Minutes): is not used in the current implementation.

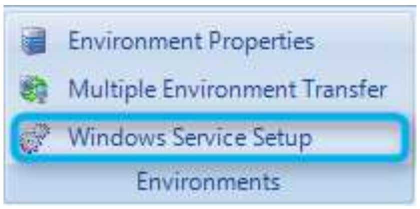
SSIS Service Send Timeout (Minutes): used to initialize the OperationTimeout, which governs the whole process of sending a message, including receiving a reply message for a request/reply service operation. This timeout also applies when sending reply messages from a callback contract method.

Use the **Test Service** button to test for any connectivity and/or permission issues.

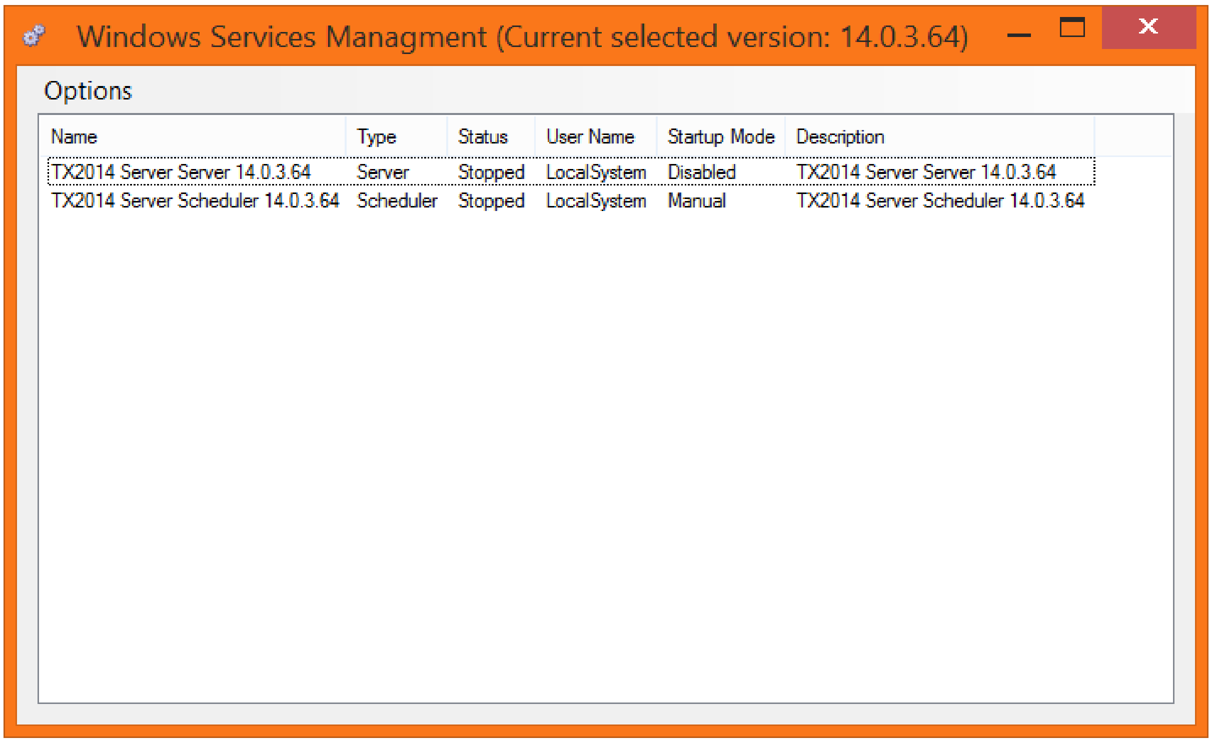
Windows Services Management tool

TX2014 SR1 includes a new tool, **Windows Services Management,** to view, add and delete the Windows services used by TX. The tool is useful in a number of situations, e.g. when configuring the scheduler on initial installation of TX or if you want to add services for another repository on the same server – for instance in a multiple environments setup.

Open the tool from **Tools** -> **Environments** -> **Windows Services Management**.



This will show you the services of the version of TX you are currently using.


Windows Services Managment (Current selected version: 14.0.3.64)

Options

Name	Type	Status	User Name	Startup Mode	Description
TX2014 Server Server 14.0.3.64	Server	Stopped	LocalSystem	Disabled	TX2014 Server Server 14.0.3.64
TX2014 Server Scheduler 14.0.3.64	Scheduler	Stopped	LocalSystem	Manual	TX2014 Server Scheduler 14.0.3.64

To view services from all versions of TX, click **Options** -> **Views** -> **All versions**. This will show you all TX services, including services left over from uninstalled versions of TX.

Right click on an existing service to edit it. You have the following options:

Start service starts the service with the current configuration. A pop-up will inform you if the service was successfully started.

Start mode defines how the service starts. It can be one of the following:

Automatic – should be used for instances of the server service

Automatic (delayed) – should be used for instances of the scheduler service.

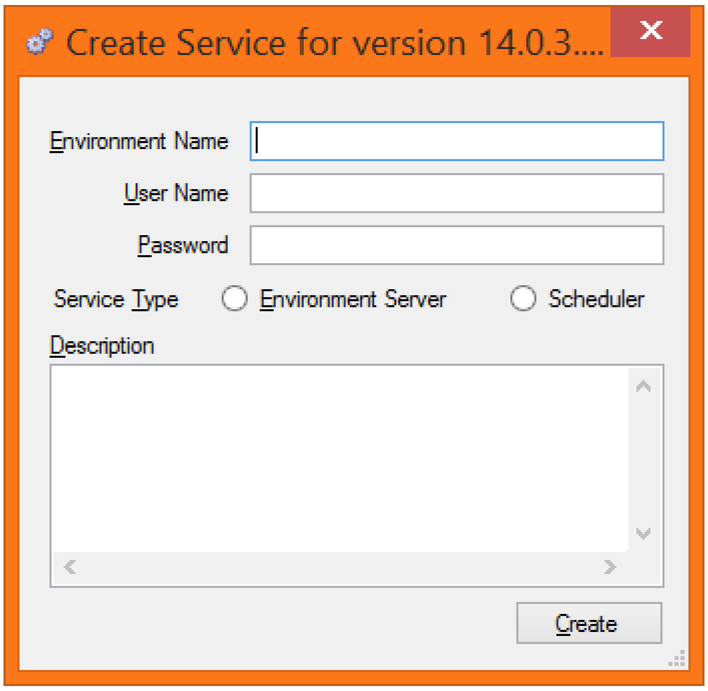
Manual

Disabled

Change username and password lets you set the username and password the service runs under. Note that this determines which repository the service uses as repositories are configured on a per user basis.

Delete service deletes the service. The option is unavailable if the service in question was installed together with a currently installed version of TX.

To add a new service click **Options** -> **Create service for <version>**. The **Create Service** dialogue appears.



Environment Name is a text string added to the name of the service and enables you to identify the service later.

User Name and **Password** determines which repository the service uses as repositories are configured on a per user basis.

Service Type can be "Environment Server" or "Scheduler".

Description is optional and meant for notes about the created service.

Create creates the service. A pop-up will inform you if the service was successfully created. The create service dialogue will remain open to enable you to create another service with the same options.

Microsoft SQL Server 2014 Support

TX 2014 SR1 is the first version of TX with support for Microsoft SQL Server 2014, Standard, BI and Enterprise Edition. This means that your TX Project can now be deployed on the latest SQL Server version from Microsoft.

The list of supported SQL Server versions is now:

- SQL Server 2005 Standard and Enterprise Edition with latest Service Pack
- SQL Server 2008 Standard and Enterprise Edition with latest Service Pack
- SQL Server 2008SR2 Standard and Enterprise Edition with latest Service Pack
- SQL Server 2012 Standard, BI and Enterprise Edition with latest Service Pack
- SQL Server 2014 Standard, BI and Enterprise Edition with latest Service Pack

Improved features

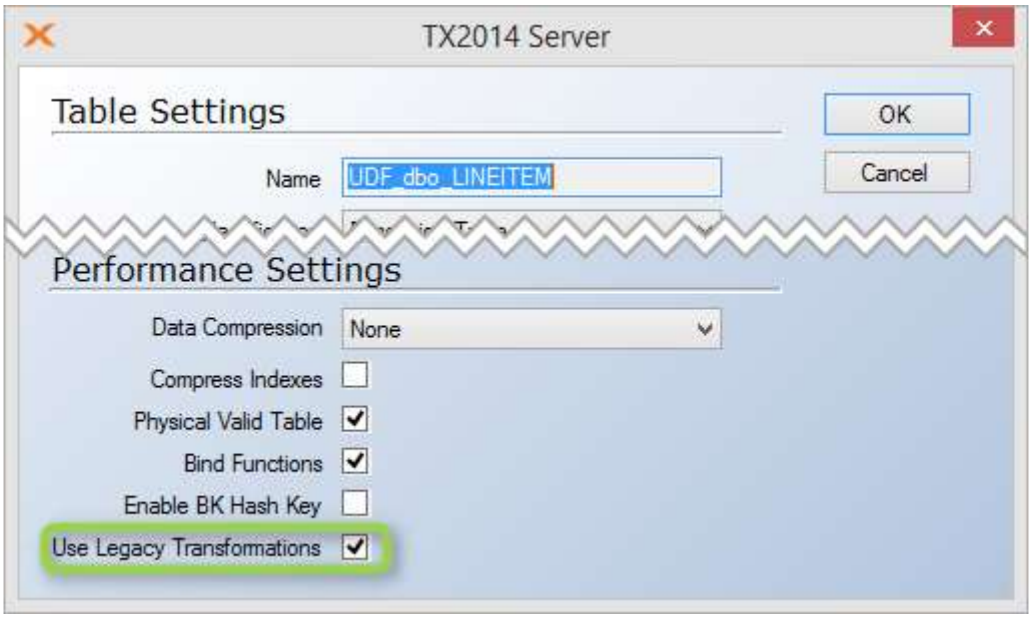
Set Based Transformations

Some transformations do not require to be executed Row-by-Row in a scalar-value user defined function, but can be executed Set-based as part of the transformation view instead. The latter method is significantly faster and TX2014 SR1 takes advantage of that. We have re-programmed the SQL code generator so it will create your transformations as CASE-WHEN statements in the transformation view as much as possible to increase performance.

The only situation where TX will still create Scalar-Valued UDFs is when it would result in a CASE-WHEN statement with more than 10 WHEN's – because that is not supported by SQL Server.

As changing how we generate transformations in SQL is a major change, we have decided to include a way to turn this new functionality off if it – against our best intentions and testing – should generate wrong code in some scenarios.

To change how TX generates the SQL code for transformations, right click on the table and select **Advanced** -> **Advanced Settings**.



Name: UDF_dbo_LINEITEM

Data Compression: None

Compress Indexes: ☐

Physical Valid Table: ☑

Bind Functions: ☑

Enable BK Hash Key: ☐

Use Legacy Transformations: ☑

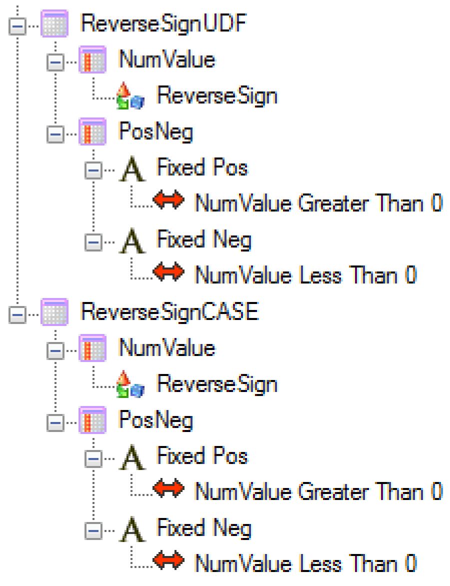
If **Use Legacy Transformations** is checked, TX will revert to the previous SQL Code generation for transformations and create Scalar-Valued User Defined Functions. This is the setting for all tables in upgraded projects – so you need to uncheck it to gain the performance boost.

If **Use Legacy Transformations** is un-checked (Default), then TX will generate CASE-WHEN structures in the Transformation views instead of Scalar-Valued User Defined Functions.

Even though a transformation done in a User Defined Function and in the CASE-WHEN structure gives the same end-result, there are some differences in the design that could cause differences in the end result when using the two different methods.

An example:

I have 2 tables with identical transformations and identical data:


ReverseSignUDF
- NumValue
  - ReverseSign
- PosNeg
  - Fixed Pos
    - NumValue Greater Than 0
  - Fixed Neg
    - NumValue Less Than 0

ReverseSignCASE
- NumValue
  - ReverseSign
- PosNeg
  - Fixed Pos
    - NumValue Greater Than 0
  - Fixed Neg
    - NumValue Less Than 0

Here is the result of the UDF version:

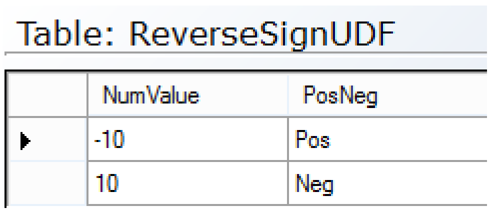


Table: ReverseSignUDF

	NumValue	PosNeg
▶	-10	Pos
	10	Neg

Here is the result of the Set based Transformation:

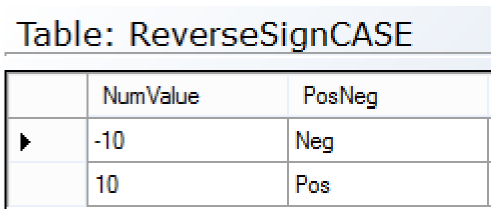


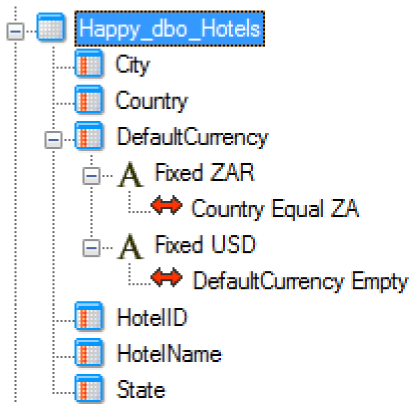
Table: ReverseSignCASE

	NumValue	PosNeg
▶	-10	Neg
	10	Pos

The Set Based Transformation is correct, but since the UDF result has always been used, we decided to make that **the default for upgraded projects**.

Here is an example of how TX generates the underlying SQL Code for a common transformation.

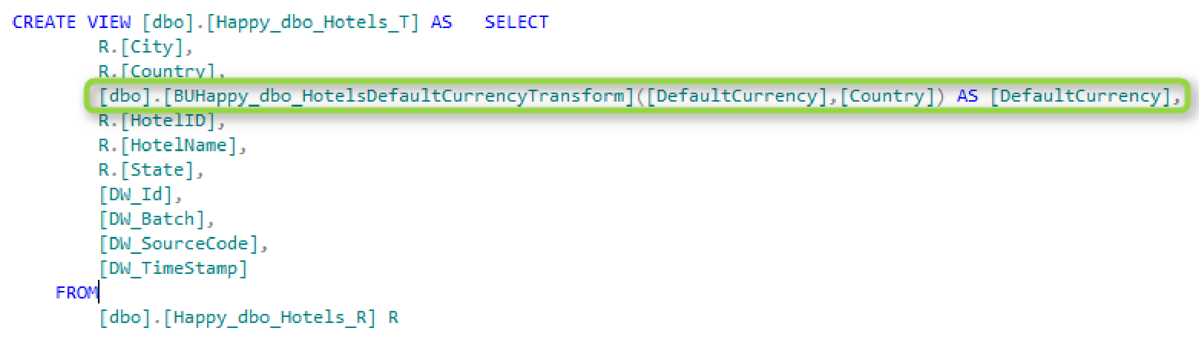
The Transformation in TX:



Previous SQL Implementation was a User Defined Function:

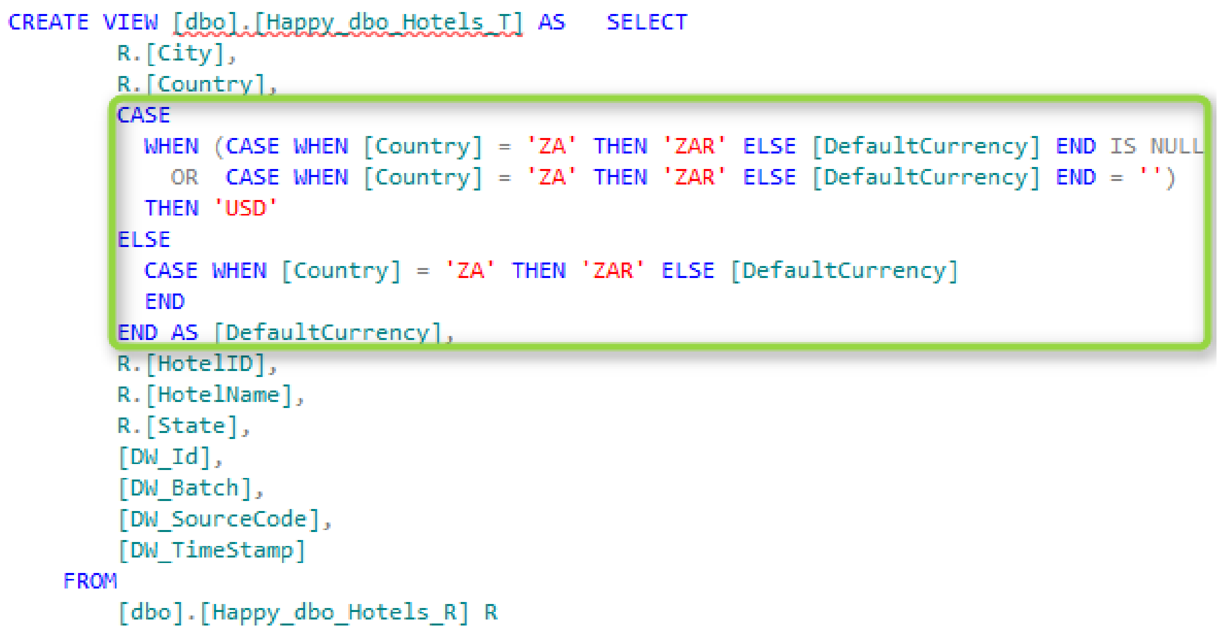
```
CREATE FUNCTION [dbo].[BUHappy_dbo_HotelsDefaultCurrencyTransform](

@DefaultCurrency3 nvarchar(3),
@Country2 varchar(3))

RETURNS nvarchar(3) WITH SCHEMABINDING AS
BEGIN

    IF(@Country2 = 'ZA')
    BEGIN
        SET @DefaultCurrency3 = 'ZAR'
    END


    IF((@DefaultCurrency3 IS NULL OR @DefaultCurrency3 = ''))
    BEGIN
        SET @DefaultCurrency3 = 'USD'
    END

RETURN @DefaultCurrency3
END
```

… and a Transformation View:



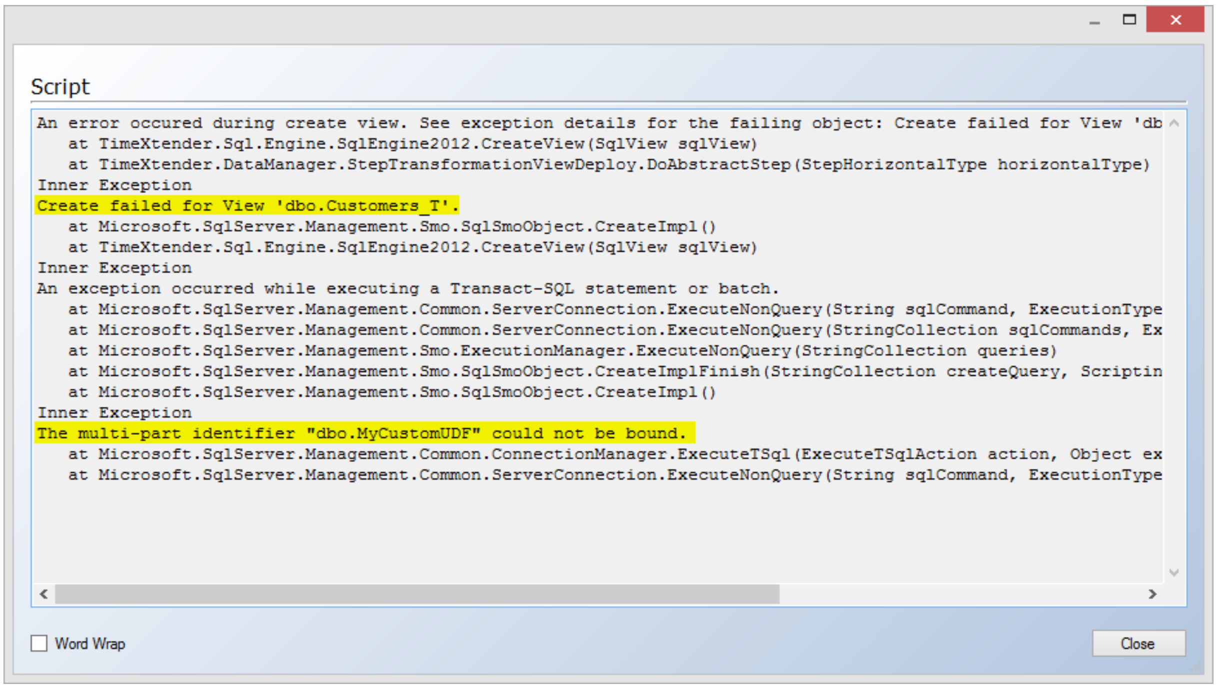
```
CREATE VIEW [dbo].[Happy_dbo_Hotels_T] AS   SELECT
        R.[City],
        R.[Country],
        [dbo].[BUHappy_dbo_HotelsDefaultCurrencyTransform]([DefaultCurrency],[Country]) AS [DefaultCurrency],
        R.[HotelID],
        R.[HotelName],
        R.[State],
        [DW_Id],
        [DW_Batch],
        [DW_SourceCode],
        [DW_TimeStamp]
    FROM
        [dbo].[Happy_dbo_Hotels_R] R
```

The New implementation is this CASE-WHEN structure in the Transformation View:



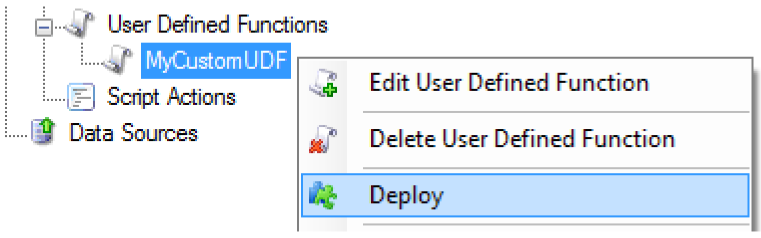
```
CREATE VIEW [dbo].[Happy_dbo_Hotels_T] AS   SELECT
        R.[City],
        R.[Country],
        CASE
          WHEN (CASE WHEN [Country] = 'ZA' THEN 'ZAR' ELSE [DefaultCurrency] END IS NULL
            OR  CASE WHEN [Country] = 'ZA' THEN 'ZAR' ELSE [DefaultCurrency] END = '')
          THEN 'USD'
        ELSE
          CASE WHEN [Country] = 'ZA' THEN 'ZAR' ELSE [DefaultCurrency]
          END
        END AS [DefaultCurrency],
        R.[HotelID],
        R.[HotelName],
        R.[State],
        [DW_Id],
        [DW_Batch],
        [DW_SourceCode],
        [DW_TimeStamp]
    FROM
        [dbo].[Happy_dbo_Hotels_R] R
```

Since the Set Based transformations are deployed in the Transformation View, transformations based on User Defined Functions (Defined by the User in TX) must be deployed before the transformation view is deployed. This can result in some manual deployment of User Defined Functions when deploying to a database where the user defined functions has not deployed earlier. If a Transformation View is deployed before the User Defined Function, a deployment error similar to this is thrown:



```
An error occured during create view. See exception details for the failing object: Create failed for View 'db
    at TimeXtender.Sql.Engine.SqlEngine2012.CreateView(SqlView sqlView)
    at TimeXtender.DataManager.StepTransformationViewDeploy.DoAbstractStep(StepHorizontalType horizontalType)
Inner Exception
Create failed for View 'dbo.Customers_T'.
    at Microsoft.SqlServer.Management.Smo.SqlSmoObject.CreateImpl()
    at TimeXtender.Sql.Engine.SqlEngine2012.CreateView(SqlView sqlView)
Inner Exception
An exception occurred while executing a Transact-SQL statement or batch.
    at Microsoft.SqlServer.Management.Common.ServerConnection.ExecuteNonQuery(String sqlCommand, ExecutionType
    at Microsoft.SqlServer.Management.Common.ServerConnection.ExecuteNonQuery(StringCollection sqlCommands, Ex
    at Microsoft.SqlServer.Management.Smo.ExecutionManager.ExecuteNonQuery(StringCollection queries)
    at Microsoft.SqlServer.Management.Smo.SqlSmoObject.CreateImplFinish(StringCollection createQuery, Scriptin
    at Microsoft.SqlServer.Management.Smo.SqlSmoObject.CreateImpl()
Inner Exception
The multi-part identifier "dbo.MyCustomUDF" could not be bound.
    at Microsoft.SqlServer.Management.Common.ConnectionManager.ExecuteTSql(ExecuteTSqlAction action, Object ex
    at Microsoft.SqlServer.Management.Common.ServerConnection.ExecuteNonQuery(String sqlCommand, ExecutionType
```

To Fix it, manually deploy the user defined function:



When deployed, the Transformation View will deploy without errors.

Hashed Primary Key field Control

Having a Hashed version of the primary key can speed up lookups from other tables, especially if the primary key consists of multiple alphanumerical fields.

On the advanced table settings, it is now possible to select whether TX should generate a Hashed version of the Primary Key (BK_HASH_KEY) or not.

To control this setting on the table, right click it and select **Advanced** -> **Advanced Settings.**

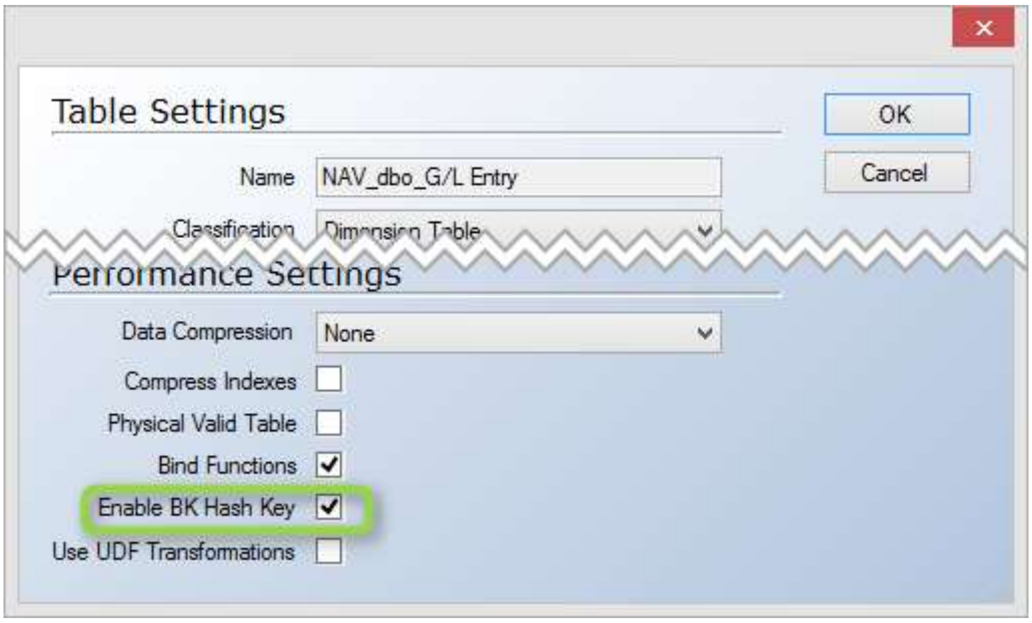


Table Settings

Name: NAV_dbo_G/L Entry

Classification: Dimension Table

Performance Settings

Data Compression: None

Compress Indexes: ☐

Physical Valid Table: ☐

Bind Functions: ✓

Enable BK Hash Key: ✓

Use UDF Transformations: ☐

When **Enable BK Hash Key** is checked, TX will generate a Hashed version of the Primary Key field(s) using the SHA1 Hashing Method. If no primary key is selected, then the BK_HASH_KEY will contain a Hashed version of the DW_ID Field.

In previous versions of TX, The BK_HASH_KEY was controlled by the table classification. Setting a table to "Fact Table – Large" or "Dimension Table – Large" would add the system field BK_HASH_KEY to the table. This new feature removes that functionality and allows the user to be in complete control.

Table and Index Compression Control

On the advanced table settings, it is now possible to control Table Compression for the individual table. You can also add compression to the table's indexes.

In previous versions of TX, this was controlled by the table classification. If the table classification was set to "Fact Table – Large" or "Dimension Table – Large" – AND if the deployment target supported compression, then TX would deploy the table with compression. By default it would use Row Compression on the staging area and Page Compression on the data warehouse.

To control Compression on the table, right click it and select **Advanced** -> **Advanced Settings.** This will display the table settings and in the Performance area, compression options can be set:

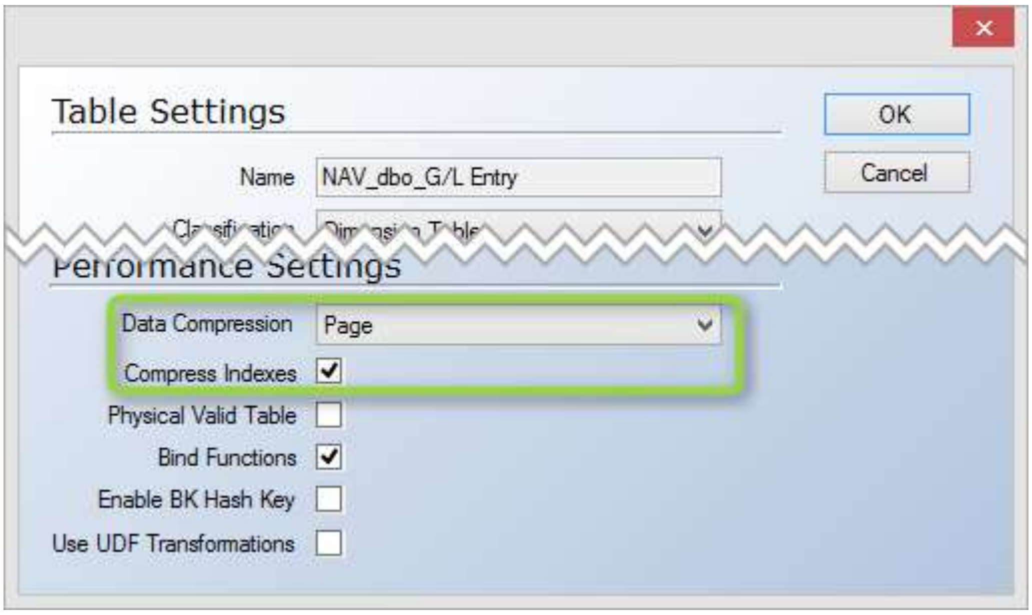


Table Settings

Name: NAV_dbo_G/L Entry

Classification: Dimension Table

Performance Settings

Data Compression: Page

Compress Indexes: ✓

Physical Valid Table: ☐

Bind Functions: ✓

Enable BK Hash Key: ☐

Use UDF Transformations: ☐

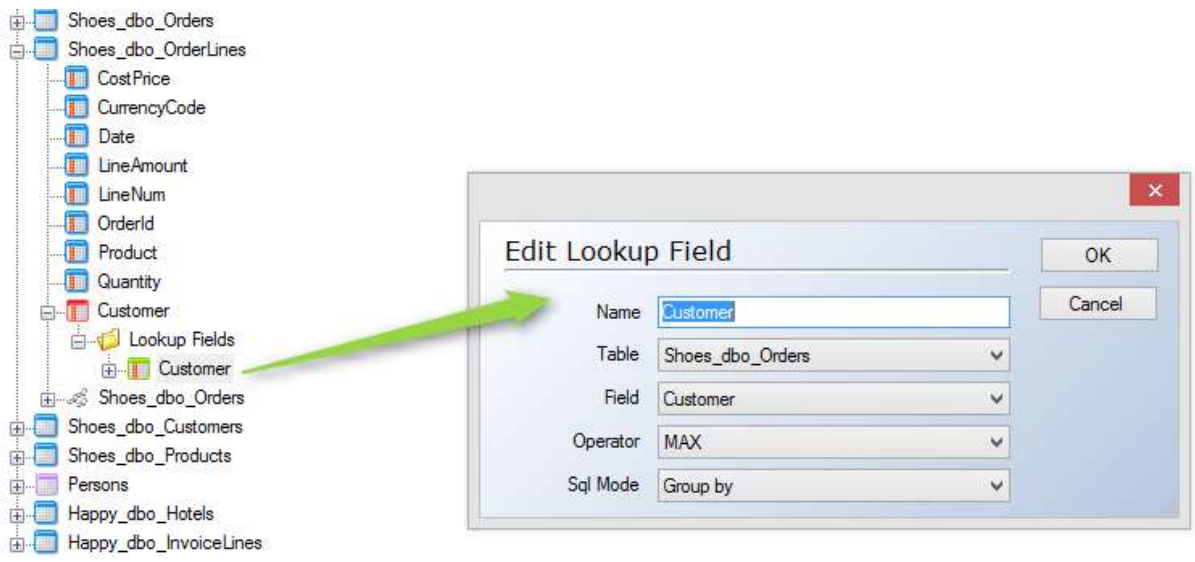
Data Compression: Choose either "None" for no compression, "Row" for row level compression or "Page" for page level compression. Read more about the individual compression types here: http://technet.microsoft.com/en-us/library/cc280449.aspx

Compress Indexes: If checked, TX will apply compression to the indexes on the table as well.

New Defaults for Conditional Lookup Fields

We have changed the default settings for new conditional lookup fields. The default **Operator** is now "MAX", while the default **SQL Mode** is "Group by". In previous versions, the defaults were "TOP" and "Partition Over" respectively.

We have found that using "MAX" and "Group by" is the most efficient method and provides the correct result in most cases. Should you encounter a situation where the lookup returns an incorrect value, then please test with the "Partition Over" **SQL Mode** and report the issue to timeXtender support.



Edit Lookup Field

Name: Customer
Table: Shoes_dbo_Orders
Field: Customer
Operator: MAX
Sql Mode: Group by

Support for Columnstore Indexes

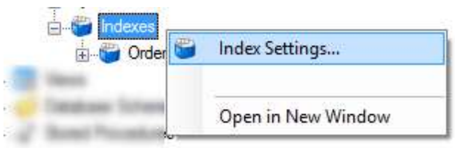
In TX2014 SR1, you can create Columnstore Indexes to significantly improve performance when querying your large fact tables in typical data warehouse scenarios. Please note that you need SQL Server Enterprise Edition to utilize Columnstore indexes and that not all data types are supported.

For a list of supported data types on different SQL Server versions, read more here:

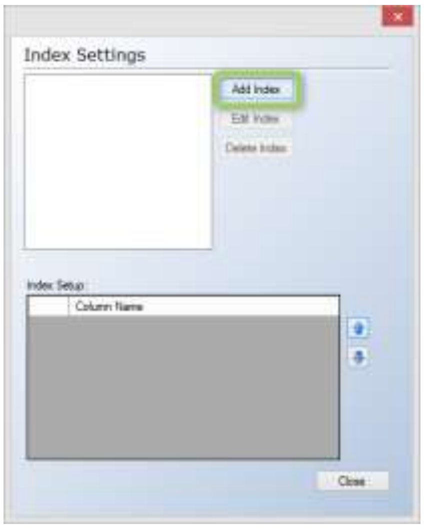
SQL Server 2012: http://msdn.microsoft.com/en-us/library/gg492088(v=sql.110).aspx

SQL Server 2014: http://msdn.microsoft.com/en-us/library/gg492088(v=sql.120).aspx

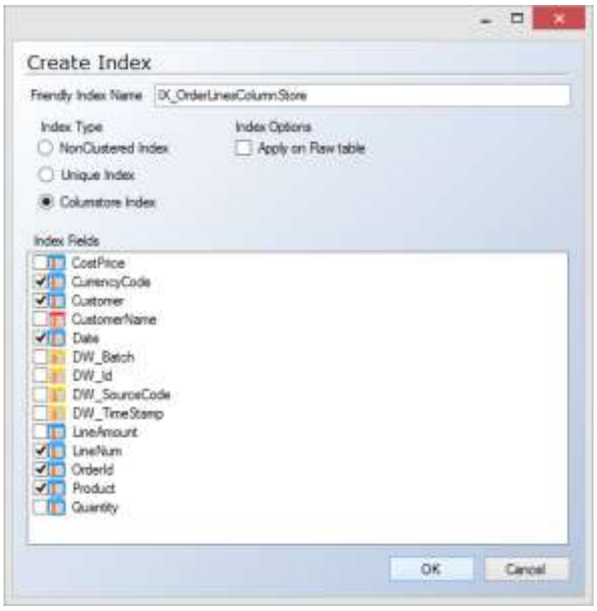
To create a Columnstore index, right click on Indexes and select **Index Settings**…:



Then click "Add Index":



Type an **Friendly Index Name**, select **Columnstore Index** and choose the **Index Fields**:


Create Index

Friendly Index Name: IX_OrderLinesColumnStore

Index Type
- ( ) NonClustered Index
- ( ) Unique Index
- (●) Columnstore Index

Index Options
- ☐ Apply on Raw table

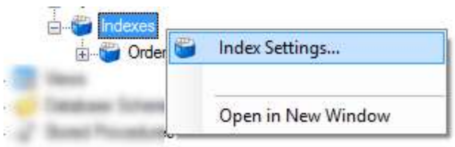
Index Fields
- ☐ CostPrice
- ☑ CurrencyCode
- ☑ Customer
- ☐ CustomerName
- ☑ Date
- ☐ DW_Batch
- ☐ DW_Id
- ☐ DW_SourceCode
- ☐ DW_TimeStamp
- ☐ LineAmount
- ☑ LineNum
- ☑ OrderId
- ☑ Product
- ☐ Quantity

The index will be created as a Non-Clustered Columnstore Index.

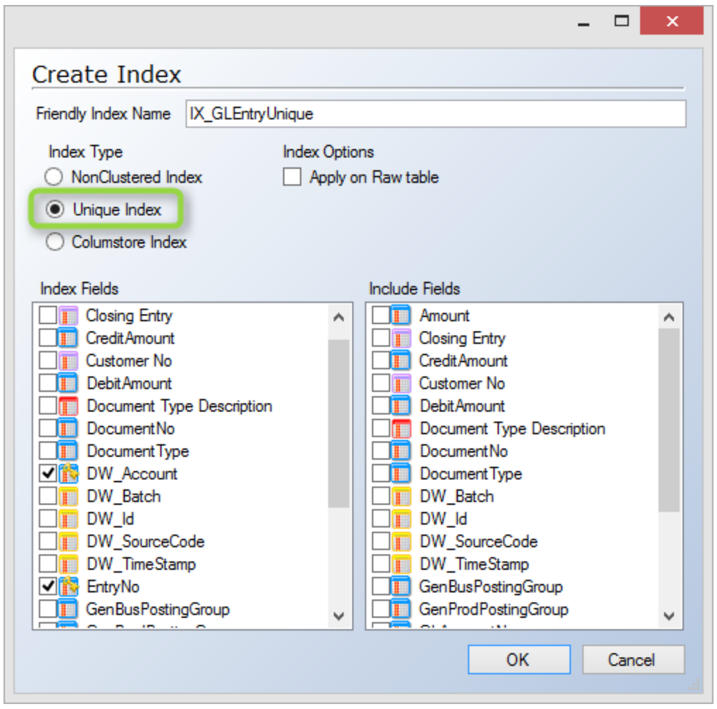
Support for Unique indexes

Due to requests from some of our partners and customers, we have added the option to mark an index as unique, which the Query Optimizer will then take into consideration when determining the best execution plan.

To create a Unique Index, right click on Indexes and choose **Index Settings…**:



Enter a **Friendly Index Name**, select "Unique Index" in **Index Type** and select your Index- and Include columns.


Create Index

Friendly Index Name: IX_GLEntryUnique

Index Type
- NonClustered Index
- [✓ Unique Index]
- Columnstore Index

Index Options
- Apply on Raw table

Index Fields
- Closing Entry
- CreditAmount
- Customer No
- DebitAmount
- Document Type Description
- DocumentNo
- DocumentType
- [✓] DW_Account
- DW_Batch
- DW_Id
- DW_SourceCode
- DW_TimeStamp
- [✓] EntryNo
- GenBusPostingGroup

Include Fields
- Amount
- Closing Entry
- CreditAmount
- Customer No
- DebitAmount
- Document Type Description
- DocumentNo
- DocumentType
- DW_Batch
- DW_Id
- DW_SourceCode
- DW_TimeStamp
- GenBusPostingGroup
- GenProdPostingGroup

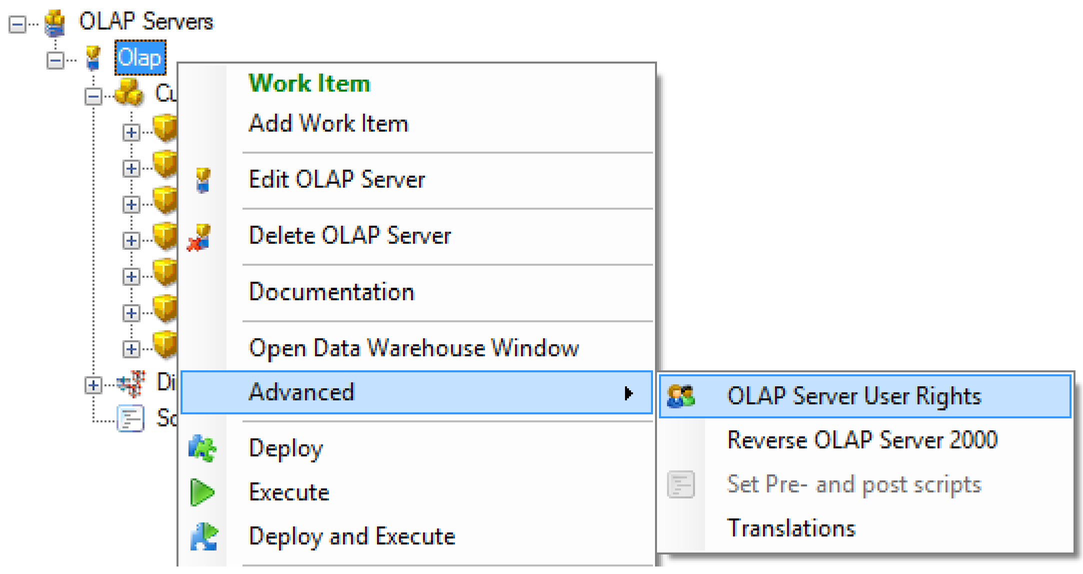
Although a Unique Index cannot be created on a table with duplicates, the index itself does not prevent duplicates to be inserted into the table.

Support for MDX Based OLAP Security

In TX2014 we added new options for setting up OLAP Security. In SR1 we have added the option to use MDX for setting up Allowed or Denied member sets instead of using the graphical user interface.

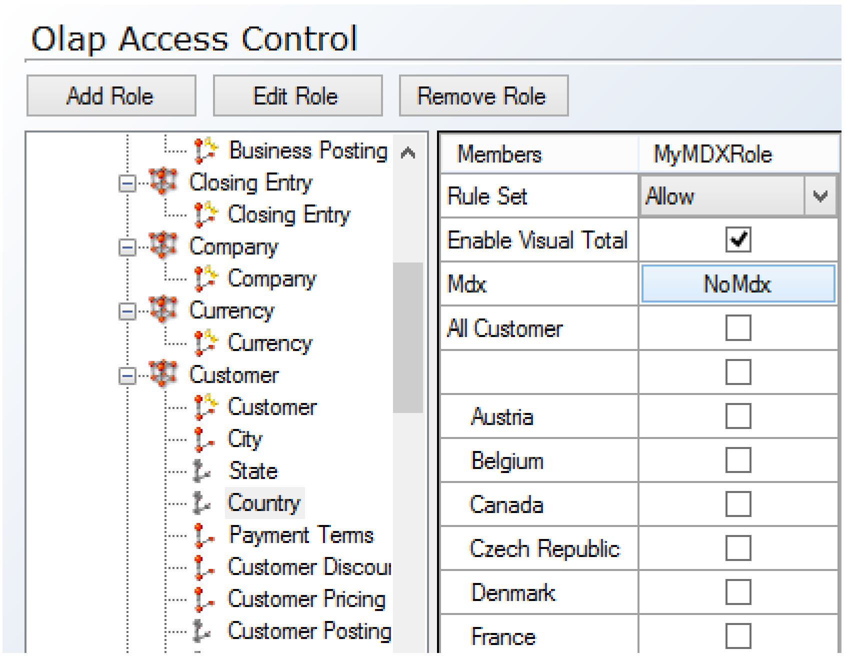
This is very useful if you want to build Dynamic Security where the relations between users and the dimension members they are allowed to see is maintained in a table structure on the data warehouse. A guide for setting up dynamic security on your OLAP database will be available on our support site (http://support.timextender.com) – shortly after the release of TX2014 SR1.

To use an MDX expression in a role, go to **OLAP Server User Rights**:



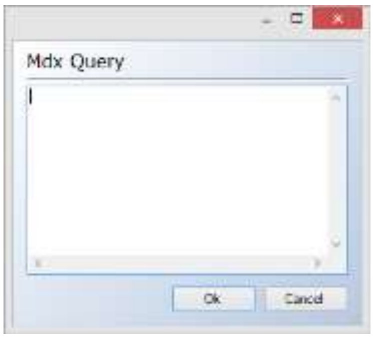
Add or Edit the Role, navigate to the Dimension Member node where you want to define a MDX based allowed or denied Memberset.

Select Either "Allow" or "Deny" Rule Set, and click the **NoMdx** button in the grid:

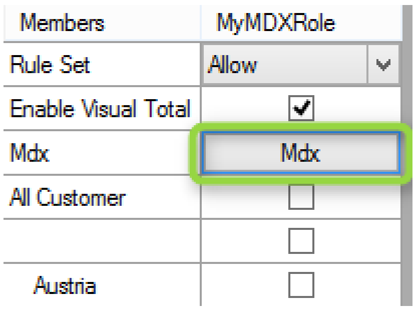


Members	MyMDXRole
Rule Set	Allow
Enable Visual Total	✓
Mdx	NoMdx
All Customer	☐
	☐
Austria	☐
Belgium	☐
Canada	☐
Czech Republic	☐
Denmark	☐
France	☐

Enter your MDX:



The **NoMdx** button will change into **Mdx**.



Members	MyMDXRole
Rule Set	Allow
Enable Visual Total	✓
Mdx	Mdx
All Customer	☐
	☐
Austria	☐

If there is a MDX member set defined, any checkmarks in the individual members on the dimension are ignored – Only the MDX part is deployed to Analysis Services.

Support for extracting data from SQL Server without locks

For SQL Server databases extracting data usually requires a Shared lock on the records extracted to ensure that no dirty reads are made (dirty read = reading not yet committed data). However, in some scenarios, it makes sense to extract the data without the use of a shared lock and you can now control this on your SQL Server Data source(s).

For SQL Server we will specify “WITH NOLOCK” / “WITH READUNCOMITTED” when reading data.

Improved Data Source Synchronization

We have made some improvements to the data source synchronization. First, you can now keep the synchronization result window open while continuing working with the project. This is very useful if you need to handle multiple changes in the data source after a synchronization. If you close the window by accident, then you can reopen the latest synchronization result as long as you have not closed the project.

Secondly, we added groupings to the result so you can easily identify which changes in the data source that is directly affecting the project.

Bug Fixes in 14.2.1

We fixed the following bugs since the last release:

ID	Title
438	"Selection rule" on DWH table is not shown the first time it is selcted in case you like more than one "Selection rule" added.
505	Issue with varbinary(20) on dwh
506	HaskKey on table problem
511	Issue with needless removal of indexes on history table deployment
512	Issue with needless removal of SCD Hash columns on history enabled tables
517	Rename of custom view gives problems
518	Error in Add Related Records when using List operator
525	Change of Version Control Feature Scope
528	Cannot enable sift tables on existing Nav Adapter
529	Not all valid fields on a dw table is shown in selection rule
530	Application chrash when changing transformation to type default
531	Spelling
532	Not Empty Condition on Lookupfields is not implemented correct
536	Data source panel scrolls to the top after deselecting a table
538	Table insert bug
541	MySQL boolean is converted in to binary
542	TX will become non responsive when custom transformations makes a circular reference
543	Timestamp with local timezone does not show on oracle
544	Force Unicode on Global Oracle Database has no effect
546	Text when markup scripts fail to load is wrong
548	Unhandled exception in Table Aggregations dialogue
549	IsDirty not updated when changing lookup method
554	Refresh error when adding dimension levels
555	Issue with deployment of non-dirty Source Based Incremental Table to an empty database
556	Template Data Source parameteres not saved on initial setup
558	Incremental load - more than 2 rules does not work
559	V table alter through smo for decimal is done incorretly

560	Deleting a SQL snippet does not mark table as dirty.
562	Slow saving
563	The dialog for execution does not display full time after 10 hours fo elapsed time
564	Order of Sortings on lookup fields
565	Missing update of Warning-sign on Role Playing Dimensions
566	Sorting of dropdown in Business Function Measure
567	Preview fails on Salesforce adapter if table has custom fields
568	Default transformations can have conditions
569	Project settings not saved on new project
570	SqlExecutionLogger has not been initialized
575	First field of SQL source using ADO transfer can not be custom
576	AX adapter and VARBINARY(max)
577	Single field data movement from stage to separate data warehouse window confuses TX
578	SalesForce login can expire.
579	Problem when execution is done multithreaded with ADO
581	Facttables can be removed when having measures.
586	AX Virtual ref field removed after sync
587	Unable to browse dimension in some cases
588	Change order with drag drop is not working for all objects
589	When deleting external business unit data movements from views are not deleted
590	Data Movements from views from an external business is expecting system fields
597	AX enum label is fixed to 40
599	Insert of data into _r on salesforce and CRM adapert does not use command timeout from staging database
603	DW_TimeStamp on history tables for Type1&2 updates are incorrect
604	Find Unused field not considering aggregation columns on DWH tables
605	Managed Execution and ADO.net can make execution hang
606	Wrong Grouping of conditional lookups
608	Bold Typeface on DWH tables with descriptions can dissappear
609	Problem with conditional lookup and sql2014
610	Deploy & Execute modified tables & views does not work when DM comes from view in ext BU

616	Value Column shows fields from another table and does not save settings
617	GP read object problem when using global database
624	Visual bug in Data Selection Rule feature
633	Load of project list on Open Project can take a long time.
634	Nav Sync - Change in Datatype for Option Value
643	Join problem on conditional lookup field

Bug fixes in 14.2.2

ID	**Title**
653	Advanced section inside the Edit Parent-Child dimension dialogue not accessible
654	Assigned measure group names lost on save / load.
655	Cannot execute dimension using managed thread when dimension is based on a view
656	Salesforce adapter - performance

Bug fixes in 14.2.3

ID	**Title**
661	Concurrency issue on AX adatper / Nav Adapter on sql
663	Slow Multi-threaded execution when having environments
664	Remote SSIS Authentication Issue when running on Windows Server 2012
665	Unable to setup OLAP Server User Rights on front database
666	It is not possible to turn of execution time logging on execution package

Bug fixes in 14.2.4

ID	**Title**
668	Slow table dependency dialogue
669	Execution of text file source fails with DateTime fields and Unicode selected
670	Guard has no effect on table inserts and Add related records in managed execution
671	Slow delete of projects
672	Windows Service Setup is not visible without Multiple Environment Transfer feature

Bug fixes in 14.2.5

ID	Title
674	Wrong error message when processing cube and dimension
675	Synchronization can fail on NAV adapter in very specific scenarios
676	AX Adapter fails synchronizing objects when a table is removed from the Source.

Bug fixes in 14.2.6

ID	Title
677	Table not being marked as dirty if the order of the transformations has changed
678	Issue with Incremental load using TimeStamp fields on SQL Server data sources
679	Upgrade error on tables classified as Large Dimension or Large Fact Table
680	Non SQL 2005 supported Declare statement in Stored Procedures
681	Execution order on lookup fields is not calculated
682	Order of transformations on multiple environment deployment can be wrong
684	OLAP Front Database Name too short

TIME XTENDER

TX2014 SR2

RELEASE DOCUMENTATION

RELEASE DOCUMENTATION

TX2014 SR2

Introduction 2
New features 3
Customized Code 3
Support for External Tables 7
Project Perspectives 10
Query Tool 13
Convert table to custom table 16
Data Movement Improvements 17
Junk Dimension Automation 19
Clustered Columnstore index on SQL 2014 23
Execution Queue 24
Default Execution Package 27
Merge Option on Execution Packages 28
Export Synchronization Result 31
Change provider on adapters and data sources 33
Agresso Adapter - Beta 34
Infor M3 (Movex) Adapter - Beta 35
Improvements 36
Data Lineage and Impact Analysis improvements 36
General User Interface Changes 38
Simplified OLAP Server View 39
Table settings consolidation 40
Improved Usability in the Lookup Join Dialogue 41
Changes in Primary Key Check 42
Changes in truncation settings 43
Improved Execution Logging 44
Force specific text datatype on ODBC 45
NAV Adapter Improvements 46
Fixed Bugs 47

Introduction

We are excited to release TX2014 SR2 with many new features and improvements that makes TX2014 even more complete as a Data Warehouse Automation platform.

In TX2014 SR2 we have focused on two things:

1) Making it easier to integrate traditional coding and hand-coded solutions with our Data Warehouse Automation platform.

2) Giving the user a better overview over his project as well as tools that makes working with the project easier.

We hope that you will find features such as custom code, external tables, project perspective, the execution queue and the query tool useful. In addition to the headline features, TX2014 SR2 includes numerous small improvements and bug fixes.

In this document, you can learn how the new features work and how to use them. It supplements the TX2014 User Guide and the TX2014 SR1 Release Documentation.

We hope you will enjoy TX2014 SR2 just as much as we enjoyed developing it. If you have any feedback, good or bad, on the new release, please do not hesitate to let us know.

On behalf of the development team,

Thomas Duun
Product Manager
TimeXtender Software

New features

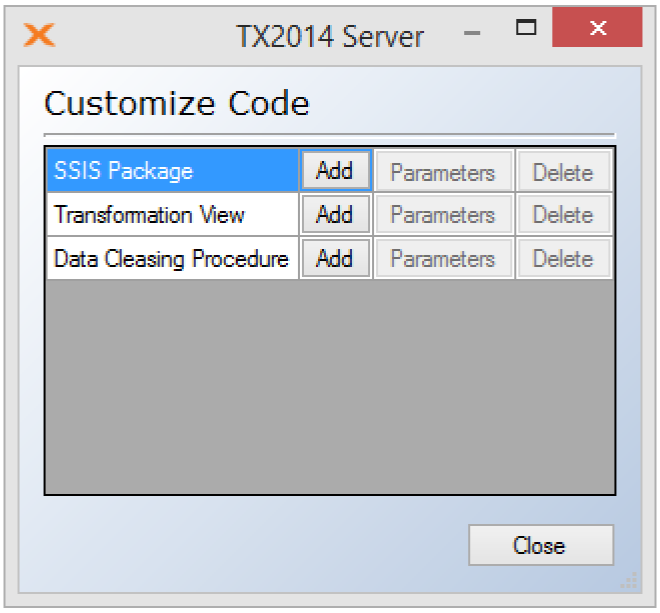
Customized Code

TX enables you to integrate "hand-written" code into a project by customizing the data cleansing procedure, transformation view and SSIS package on a given table. This gives you the flexibility of traditional data warehouse coding together with the immense productivity-boost of Data Warehouse Automation.

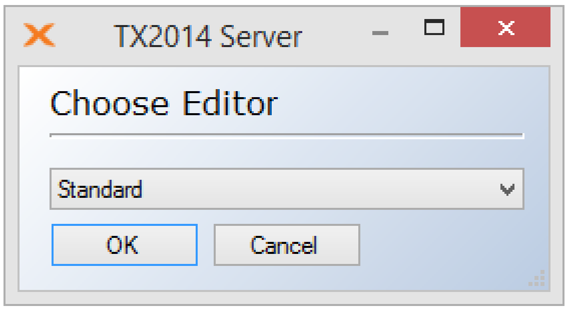
Adding customized code to a table

To customize the code on a given table, follow the steps below:

1. Right click on the table in question, navigate to **Advanced** and click **Customize code**. The **Customize Code** window appears.



2. Click the **Add** button to the right of the step you wish to customize. The **Choose Editor** window appears.

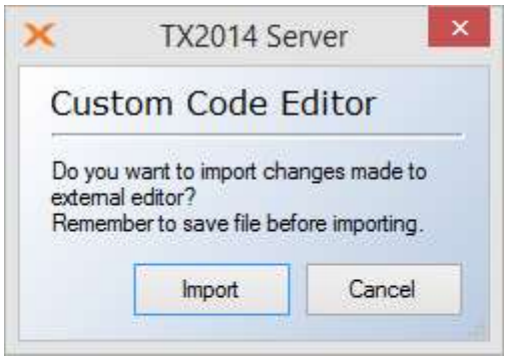


3. In the **Editor Name** list, you have the following options:
    - **Standard** is the basic built-in editor in TX.
    - **Default File Program** is the program that is set to open files of the type in question. For the data cleansing procedure and the transformation view, the filename extension is .sql. For SSIS packages, the filename extension is .dtsx.
    - Any custom editors you have added (see **Managing Custom Editors**).

   If you are adding a SSIS package, the **Custom SSIS** window appears. Chose **Create Default Package** to edit the standard package, **Create Destination Only** to create a package that only contains the destination and **Existing Package** to import an existing package from the file system or an SQL Server.

   Note: Some tables uses multiple SSIS packages. When creating the Default package, TX will create the first SSIS package only! Examples of tables that default will have multiple SSIS packages: Data Warehouse tables that receives data from multiple Staging tables. Data Source tables from NAV adapter with multiple companies. Any Data Source table when Template Data Sources are used.

4. If you chose the **Standard** editor, the **Edit** window opens. When you have finished editing the code, click **OK** to confirm you edits.
   If you chose a custom editor, TX will open the code in editor you chose. When you have finished editing the code, save your changes and close the editor. Back in TX, the **Custom Code Editor** window is open.

   Custom Code Editor

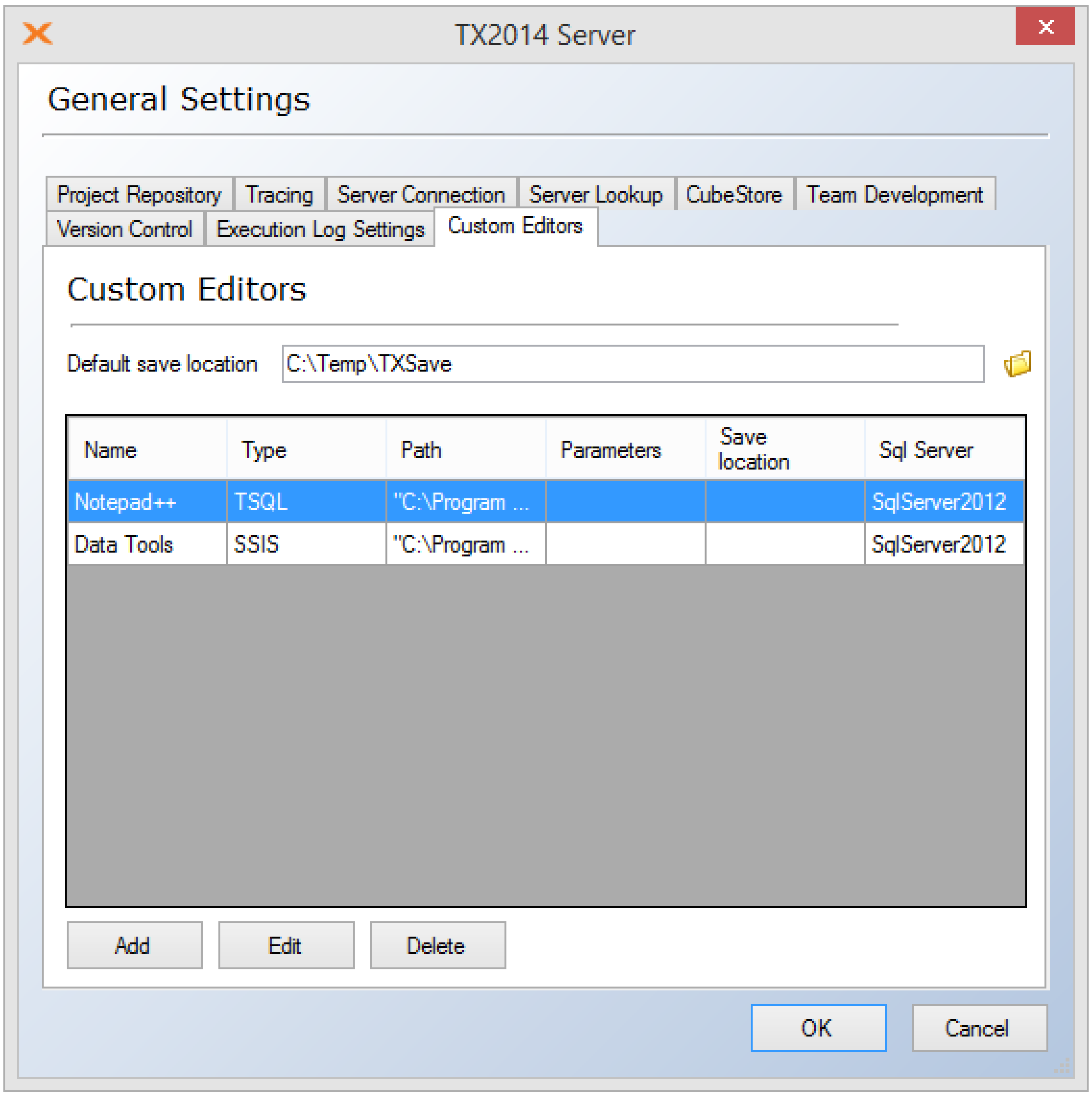
   Do you want to import changes made to external editor?
   Remember to save file before importing.

   Click **Import** to import the changes you have made in the custom editor into your project.
5. When you return to the **Customize Code** window you will notice that you can now click **Parameters** (if applicate) and **Delete**. Click **Delete** to remove the customization and return to having TX generate the code. Click **Parameters** to decide which parameters are sent to the code on execution.
6. Click **Close** to close the window.

Note: When editing the data cleansing procedure or the transformation view, make sure to have a "create procedure" or "create view" declaration in code with the exact name as TX would have given it. This is what is called during execution. To be sure, simply keep the first line of the code generated by TX.

Managing Custom Editors

To add, edit or delete a custom editor, click on the **Tools** tab in the ribbon and then click on **General Settings** in the **Administration** group. The **General Settings** window appears. Click on **Custom Editors**.



Default save location: C:\Temp\TXSave

Name	Type	Path	Parameters	Save location	Sql Server
Notepad++	TSQL	"C:\Program ...			SqlServer2012
Data Tools	SSIS	"C:\Program ...			SqlServer2012

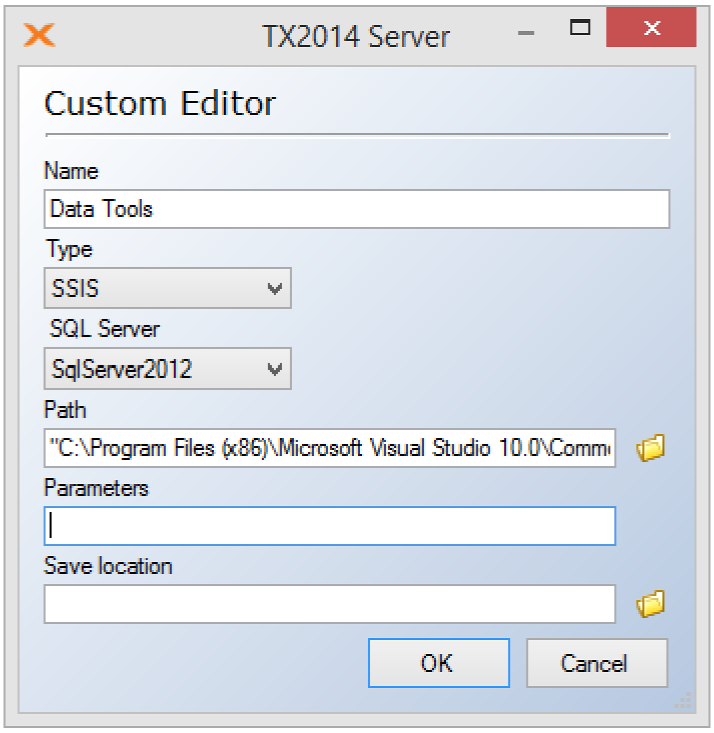
The list of custom editors is displayed. In the **Default save location** box, you can type the path to the folder where the custom code files are temporary stored (or click on the folder icon to open a browse dialog).

To edit the settings for a custom editor, select the editor in the list and click **Edit**.

To remove a custom editor from the list, select the editor and click **Delete**.

To add a custom editor, follow the steps below:

1. Click **Add.** The **Add custom editor** window appears.



Name: Data Tools
Type: SSIS
SQL Server: SqlServer2012
Path: "C:\Program Files (x86)\Microsoft Visual Studio 10.0\Comm
Parameters:
Save location:

2. In the **Name** box, type a name for the editor.
3. In the **Type** list, click on the type of editor you wish to add. Choose TSQL for use with data cleansing procedures and transformation views and SSIS for use with SSIS packages.
4. In the **SQL Server** list, select the SQL Server version that you are using. Currently, this setting is only used for custom editors for SSIS packages. When you want to customize the code for a SSIS package, TX checks what version of SQL Server the table is stored on. You will only be able to select editors that are marked compatible with that version of SQL Server.
5. In the **Path** box, type the path of the program (or click on the folder icon to open a browse dialog)
6. In the **Parameters** box, type any additional parameters for the program.
7. Optionally, in the **Save Location** box, type as save location for the editor (or click on the folder icon to open a browse dialog).
8. Click **OK** to add the custom editor.

Support for External Tables

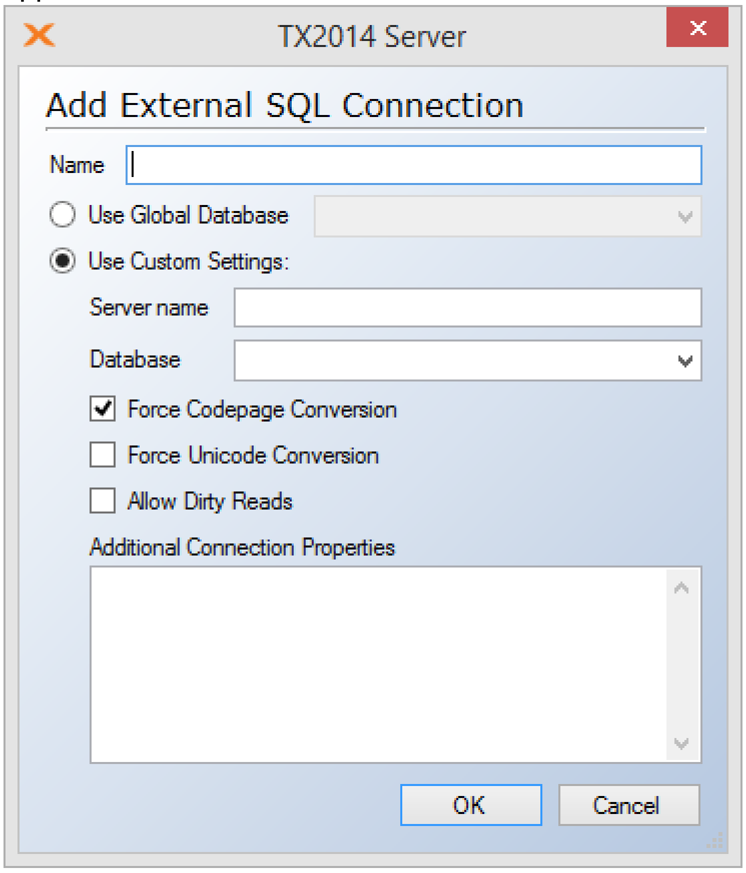
When you already have an established staging area or data warehouse, making the transition to Data Warehouse Automation means that you need to re-model the existing or some of the existing solution in TX2014 to get the full DWA advantage. In some cases, this is a good chance to re-think part of the solution – but it can also be an obstacle for moving to a modern Data Warehouse Automation platform.

External tables is a way to incorporate existing tables into the TX2014 project. An external table will initially not be deployed or executed, but will be available for data movement to data warehouses and data marts, can be used in views and scripts and for cubes and dimensions. Later on in the process, the user can add a custom SSIS package to the table, which can then be executed.

Adding an External SQL Connection

To add an external table, you first need to add an external SQL connection. To add an external SQL connection, follow the steps below:

1. Right click on you data warehouse or a business unit, navigate to **Advanced** and click **Add External SQL Connection**. The **Add External SQL Connection** window appears.



2. Type a name for the connection in the **Name** box.

3. Click on **Use Global Database** and choose a global database in the list
   or
   Click on **Use Custom Settings** and type and select your settings. Type the neame of the server in the **Server Name** box and enter the database you wish to use in the **Database** box. Select **Force Codepage Conversion** to convert all fields to the collation of the data warehouse, **Force Unicode Conversion** to declare all alphanumeric fields as *nvarchar* and **Allow Dirty Reads** to allow reading from the source without locking the table. In **Additional Connection Properties** you can type additional properties.
   Note: The SQL Server needs to be on the same physical SQL Server instance as your data warehouse.

4. Click **OK** to add the source and close the window.

Adding an External Table

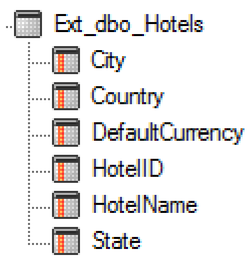
To add an external table, follow the steps below.

1. Navigate to **External SQL Connections** under your data warehouse or business unit in the project tree, right click on the connection you just created and click **Read Objects from Data Source**.

2. When TX has finished reading objects from the data source, the source explorer pane in the right hand side of the window is populated with the objects from the source. Select the tables, views and fields you wish to use in you data warehouse.

Working with External Tables

The external tables in your project are shown in the project tree alongside the standard tables and you can use them in the same way. External tables can be used in dimensions and cubes, for reporting, in Qlikview models etc. You can recognize an external table in the project tree on the black table icon.



Some of the transformations and data cleansing you can do with standard tables, can be done with external tables as well. You can add custom fields, but not lookup fields. For instance, you can add a custom field to the external table and apply a transformation to the field to concatenate two other fields on the table.

You can also add Custom Date to an external table.

Deploying an External Table

To deploy an external table, right click on the table and click **Deploy**. A View will be created that selects from the external Table.

Executing an External Table with an SSIS package

Since an external table is set up outside TX, TX expects it to be executed separately from your project. This means that you initially will not find any execute command on an external table. However, if you have a SSIS Package that is used to populate the table, you can add this package to the table and get the ability to execute the table.

1. Right click on an external table, navigate to advanced and click **Customize code**. The **Customize Code** window appears.
2. Click the **Add** button to the right of **SSIS Package**. The **Custom Editor** window appears.
3. In the **Editor Name** list, click on you editor of choice and click **OK**. The **Custom SSIS** window appears.
4. Make sure **Existing Package** is selected and click **OK**. The **Pick SSIS Package** window appears.
5. Type the server name in the **Server** box. Optionally, you can select **Use SQL Server Authentication** and type your credentials in the **User Name** and **Password** boxes as appropriate. In the **Location** list, click **File system** or **SQL Server** and then click **...** next to the **Package Name** box to browse for the SSIS package. When you have found the package and clicked **Open** in the **Open** window, click **OK** and the editor of your choice opens.
6. Make any changes you wish to make in the editor, save the package and close the editor.
7. While you edit the SSIS package, TX displays the **Custom Code Editor** dialog. When you return to TX, click **Import** to import the changes you made to the SSIS package.
8. In the **Customize Code** window you'll notice that the Add command next to **SSIS Package** has changed to **Edit** and that you can now click **Parameters** and **Delete** as well. Click **Close**.
9. Right click on the table and choose **Execute** to run the SSIS package. You can also execute the table by including it in an execution package, executing the entire project etc.

Project Perspectives

The purpose of Project Perspectives is to make it easier to work with large projects. Working in one big project can make it hard to maintain a good overview and find an individual object quickly.

The idea is that you can create different perspectives on a project. A perspective is a subset of the project objects that relates to a specific area or task. For example, you could create a "finance" perspective that contains all the tables, dimensions and cubes that are related to finance. When this perspective is active, anything else will be hidden in the project tree.

An object can be in any number of perspectives.

Adding a new perspective

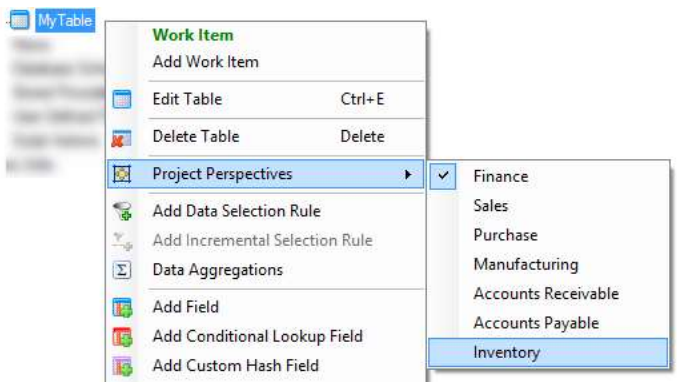
Adding a perspective is done on the project level. To add a new perspective, follow the steps below.

1. Right click on your project in the project tree, navigate to **Advanced** and click **Add Perspective**.
2. The **Add Project Perspective** window appears. Enter a name for the perspective and click **OK**.

Adding objects to and removing objects from a perspective

You can add most objects - tables, fields, dimensions, cubes - to a perspective. You can add the same object to as many perspectives as you need to. To add an object to a perspective or remove an object from a perspective, follow the steps below:

1. Right click on the object you wish to add and navigate to **Project Perspectives**.



   Here, the perspectives that the object is currently a part of have a checkmark next to them.
2. Click on the name of an unchecked perspective to add the object to this perspective
   or
   Click the name of a checked perspective to remove the object from this perspective.

Activating a perspective

Obviously, you can only have one perspective active at a time. There are three ways of activating a perspective.

A. In the project tree, navigate to **Project Perspectives**, right click on the name of the perspective you wish to activate and click **Use Project Perspective**.
B. In the ribbon, navigate to **Tools**. In the **Project Perspectives** group, click on the **Project Perspectives** list and then click on the perspective you wish to activate.
C. In the Quick Access Toolbar, click on the **Project Perspectives** list and then click on the perspective you wish to activate.

   Note: The **Project Perspectives** list is not shown in the Quick Access Toolbar when there are no perspectives in the project.

Deactivating all perspectives

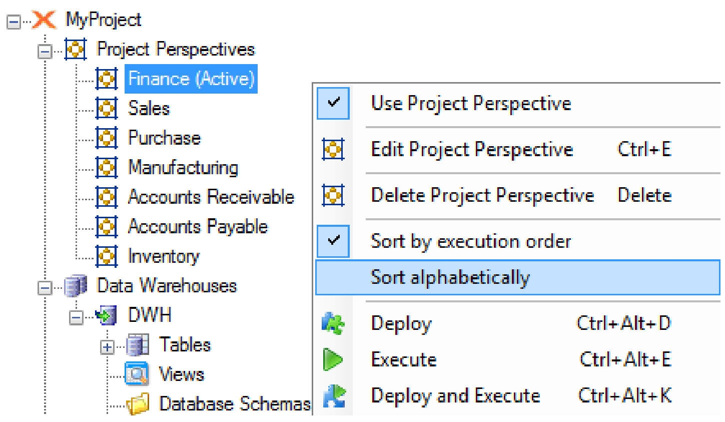
To disable all perspectives and see all objects, you have two options.

A. Click on the **None** perspective in the **Project Perspectives** list in the ribbon or the Quick Access Toolbar as described in "Activation a perspective" above.
B. In the project tree, navigate to **Project Perspectives**, right click on the name of the currently active perspective and click **Use Project Perspective**. This will remove the Checkmark from **Use Project Perspective** and change the current perspective to "None".

Sorting objects in a perspective

Objects within a perspective can be sorted by the execution order or alphabetically. To change the sort order of the active perspective, follow the steps below:

1. Click on **Project Perspectives** in the project tree.



2. Right click on the name of the currently active perspective and select either **Sort by execution** order or **Sort alphabetically.** The chosen sort order will be saved for each perspective.

Note: It is not possible change the order of objects manually while a perspective is active.

Deploying and executing a perspective

Perspectives can be deployed and executed just as other objects. This enables you to easily work with a subset of you project from source to execution. You can deploy and/or execute a perspective in three ways.

A. Right click on your project in the project tree, navigate to **Deploy** and click **Deploy Current Perspective**
or
navigate to **Execute** and click **Execute Current Perspective**
or
navigate to **Deploy and Execute** and click **Deploy and Execute Current Perspective**
B. In the project tree, navigate to **Project Perspectives**, right click on the perspective you wish to deploy and/or execute and click **Deploy**, **Execute** or **Deploy and Execute**.

You can also add a perspective to a Execution Package, for example if you wish to execute the perspective on a schedule.

Note: When deploying a perspective there is a common issue when deploying OLAP Cubes. The last dimension might fail with the error message "No dimension relationships exist within the measure group". This is caused by the way TX deploys dimensions by removing and re-applying the dimensions.

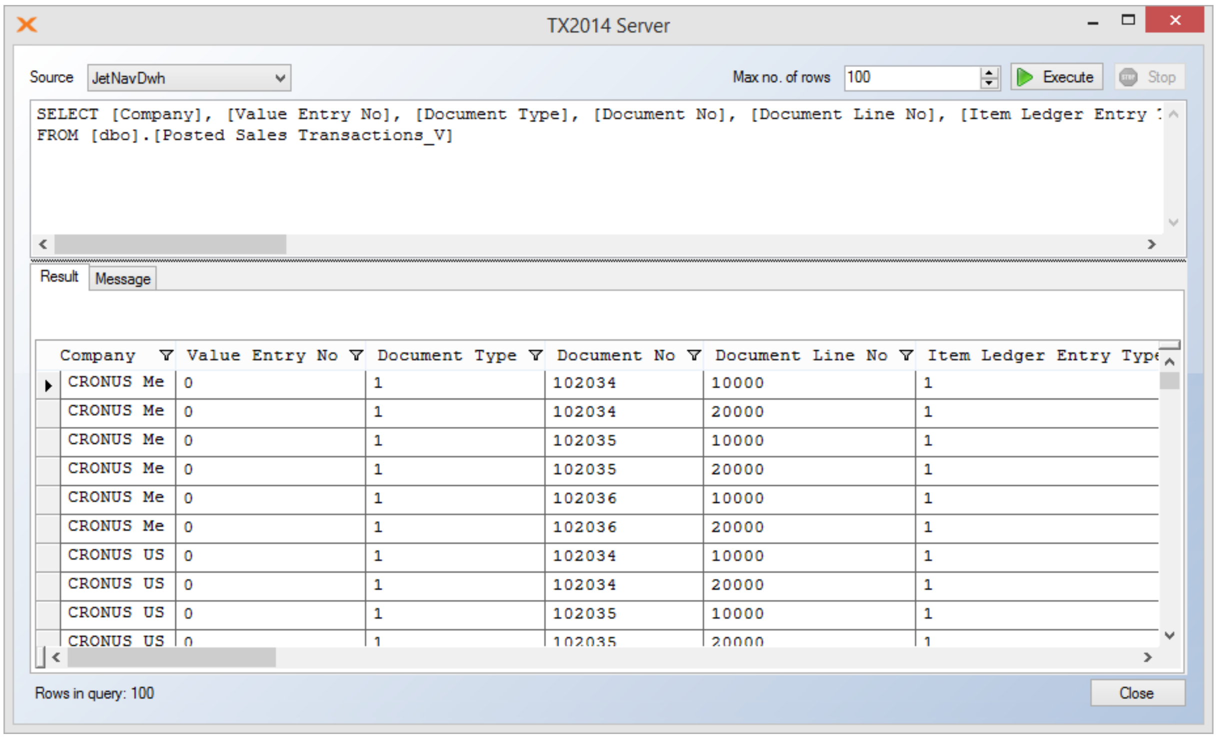
Query Tool

The Query Tool is a powerful supplement to the Preview Table feature of TX that gives you more flexibility in exploring the content of a table. You can execute any SQL query to see the data you wish to see the way you wish to see it.

Opening the Query Tool

You can open the Query Tool in three different ways.

- Right click on a table, click on **Preview Table** and click on Query Tool in the **Table Preview** window.
- Right click on a table, navigate to **Advanced** and click on **Query Tool**.
- Press **F8** on your keyboard.



```
SELECT [Company], [Value Entry No], [Document Type], [Document No], [Document Line No], [Item Ledger Entry T
FROM [dbo].[Posted Sales Transactions_V]
```

Company	Value Entry No	Document Type	Document No	Document Line No	Item Ledger Entry Type
CRONUS Me	0	1	102034	10000	1
CRONUS Me	0	1	102034	20000	1
CRONUS Me	0	1	102035	10000	1
CRONUS Me	0	1	102035	20000	1
CRONUS Me	0	1	102036	10000	1
CRONUS Me	0	1	102036	20000	1
CRONUS US	0	1	102034	10000	1
CRONUS US	0	1	102034	20000	1
CRONUS US	0	1	102035	10000	1
CRONUS US	0	1	102035	20000	1

Rows in query: 100

The Query Tool opens with a query that selects the content of the currently selected table, similar to the query that is executed to get the content for the **Preview Table** window.

Executing queries

To execute a query, follow the steps below.

1. Open the Query Tool using one of the options described above.
2. If available, choose the **Source** and **Account** you wish to query. **Account** is only displayed when using an adapter with multiple possible accounts.
3. Enter your query in the top text box of the **Query Tool** window. You can enter multiple queries that will be executed in sequence by TX.

Adjust **Max no. of rows** to the maximum number of rows you wish to have returned.

4. Click on **Execute**
   or
   Press **F5** on your keyboard.
5. If you wish to stop the query before it completes, click on **Stop**.

When they query is complete, you can see the result in the **Result** tab. If you have entered multiple queries, you can select the query result you wish to see in **Result set**. If your query resulted in a message, for example because of a syntax error, the **Message** tab will display this message.

Drag-and-drop and the Query Tool window

The Query Tool supports drag and drop of tables and fields.

- You can drag a table or a field from the project tree into the query. This places the table name in the query.
- If you drag a table to an empty query, the default query is generated. The default query fetches everything in the table.
- If you drag a table from another source into the window, you will be asked if you want to change connection and generate a default query. If you answer **No**, the name is simply added to the query.

Sorting and filtering data

The Query Tool enables you to sort and filter the results.

Note: Only the rows returned by the query are available for sorting and filtering in the **Results** tab. If you wish to sort or filter all the rows in a table, the most efficient way is to include the conditions in the query, e.g. by using "order by" or "where" clauses. Fetching thousands of rows and sorting them using the tools provided in the **Result** tab can be very slow.

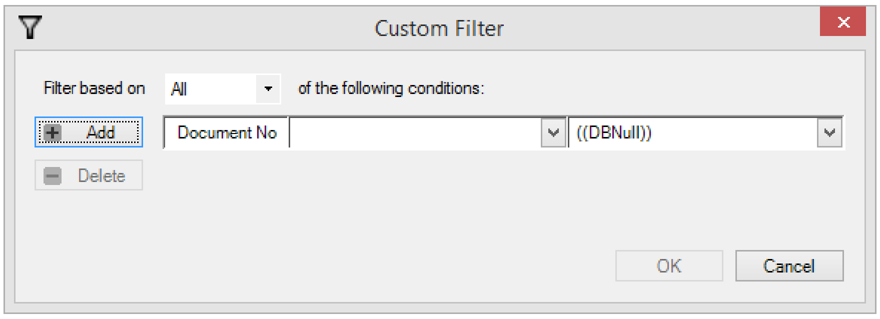
To sort the data follow the steps below:

1. Open the **Query Tool** and execute a query as described above.
2. In the **Result** tab, click on a column heading to sort the rows on the value in that column. Click again to switch between ascending or descending order.

To apply a filter, follow the steps below.

1. Open the **Query Tool** and execute a query as described above.

2. In the **Result** tab, click the filter icon besides the name of the column you wish to filter on. You have five filtering options:
   - **(All)** is equal to no filtering.
   - **(Custom)** opens the **Custom Filter** window, where you can add conditions for filtering.



     Each condition evaluates the value of the row field compared to the possible values in the column. The comparison can be made on **Equals**, **Does not equal**, **Less than**, **Less than or equals to**, **Greater than** and **Greater than or equal to**. Click **Add** to add an additional filter and click **Delete** the currently selected condition. You can choose to filter on **Any** or **All** conditions, i.e. stringing the conditions together with "or" or "and". Click **OK** to activate the filter.
   - **(Blanks)** shows rows where the column in question is blank, i.e. empty.
   - **(NonBlanks)** shows rows where the column in question is not blank.
   - A specific value. All unique values in the column is listed and can be chosen as a filter.

Convert table to custom table

In some situations, it can be useful to convert a table to a custom table.

For instance, you might have migrated some parts of an existing data warehouse into TX using external tables and you now want to consolidate you entire data warehouse in TX. In that case, you do not have to start for scratch. You can simply convert the external tables and add a new source in TX, thus minimizing the changes in the data warehouse.

In other cases, a table you depend on in you solution might not be available when you upgrade your source system to a new version. Instead of redoing a potentially complex mapping with the data warehouse, you can convert the table and keep the data until you are ready to redo that part of your solution.

Converting a table to a custom table

To convert a table to a custom table, right click on the table, navigate to **Advanced** and click **Convert table to custom table**. In the dialog that appears, click **Yes** to confirm your choice.

Data Movement Improvements

Data Movement pane and Smart Synchronization make it easier to create map fields from source to destination and spot unbalanced data movement, especially in big projects.

The Data Movement pane shows the tables that are used for data transfer to the selected table or field. For instance, at table in you data warehouse might combine data from three different tables in the staging database. The Data Movement pane provides you with an overview over these tables and their fields.

The tables and their fields are shown in a tree view. Fields in bold are used by the table or fields, you have selected in the project tree.

Smart Synchronization is a new way of synchronizing a source table with the destination table. When used, TX2014 look on the other source tables on the destination table and adds the fields from the source table that matches fields from the other source tables.

Opening the Data Movement pane

To open the data movement pane, right click on a table and click on **Data Movement**.

You can also set the pane to open automatically when you have selected a table by following the steps below:

1. In the ribbon, click on **Window and Menu settings** in the **Application Settings** group on the **Tools** tab. The **Window and menu settings** window appears.
2. In the **Data Movement pane** box, click **Show as default**.
3. Click **OK** to confirm you choice and close the window.

Using the Data Movement pane to add data movement

The Data Movement pane enables you to add new fields from a source table to the destination table you have selected in the project tree.

You can simply drag and drop fields from a source table in the Data Movement pane to the destination table.

You can also add a new source table using the Data Movement pane. To add a new source table with the Data Movement pane, follow the steps below:

1. Click on the box in **Add Table**, expand the tree view to locate the table you wish to add, click on the table and click **OK**. You can add tables from the staging area and other data warehouses.
2. Choose how to add the table by clicking on the box to the left of **Add** and clicking on the option you would like. You have four options:
    a. **Add without synchronization** simply adds the source table to the **Data Movement** pane so you can drag and drop tables from the pane to the project tree.

b. **Smart synchronize with table** looks on the other source tables on the destination table and adds the fields from the source table that matches fields from the other source tables.

   Note: You can also use smart synchronize when you add a source table by using drag and drop to the destination table in the project tree.

c. **Synchronize with table** adds the fields of the source table to the destination table with data movement.
d. **Synchronize with table (only existing fields)** adds data movement from the source to the destination table for fields that are present in the destination table.

3. Click **Add**.

Junk Dimension Automation

A Junk Dimension combines multiple low-cardinality attributes and indicators into a single dimension table as opposed to adding separate dimension tables. This reduces the size of the fact table and makes the dimensional model easier to work with.

The Junk Dimension contains a row for all distinct combinations of the Junk Dimension attributes along with a key that identifies the specific combination. The Junk dimension attribute fields can be removed from the fact-table and replaced with the single-field reference to the junk dimension table.

Example

In this example, the fact table contains 4 indicators or flags that is suited for a junk dimension (Confirmed, Delivered, Fragile, Delivery Method and Invoiced):

Item	Client	QTY	Amount	Confirmed	Delivered	Fragile	Del. Method	Invoiced
100	A123	150	15000	Confirmed	Not Delivered	Yes	Standard	Not Invoiced
200	A123	341	76000	Not Confirmed	Not Delivered	No	Standard	Not Invoiced
100	B222	140	12500	Confirmed	Delivered	Yes	Express	Invoiced
100	C112	900	85000	Confirmed	Delivered	Yes	Express	Invoiced
200	C112	600	99060	Not Confirmed	Not Delivered	No	Standard	Not Invoiced

Based on these data, the Junk Dimension will contain the following unique combinations of the Junk Dimension Attributes:

Junk ID	Confirmed	Delivered	Fragile	Method	Invoiced
1	Confirmed	Not Delivered	Yes	Standard	Not Invoiced
2	Not Confirmed	Not Delivered	No	Standard	Not Invoiced
3	Confirmed	Delivered	Yes	Express	Invoiced

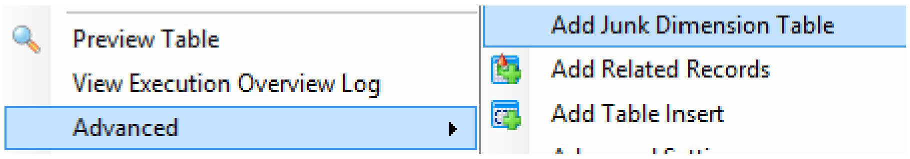
After removal of the Junk Dimension Attributes and addition of the Junk dimension reference to the fact table, it looks like this:

Item	Client	QTY	Amount	Junk Dimension ID
100	A123	150	15000	1
200	A123	341	76000	2
100	B222	140	12500	3
100	C112	900	85000	3
200	C112	600	99060	2

Adding a Junk Dimension

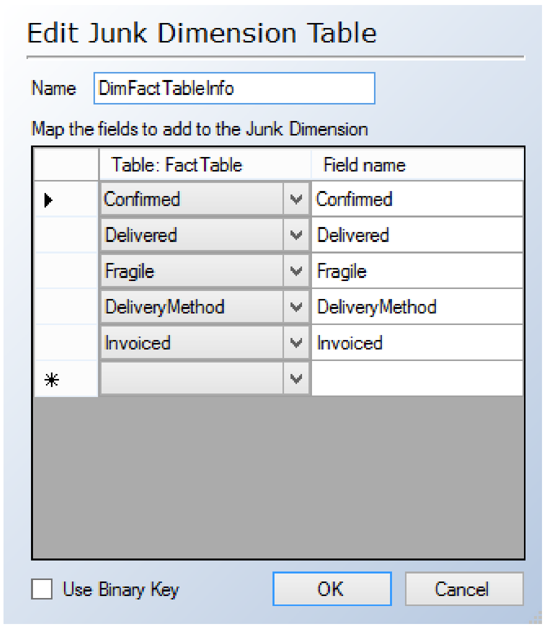
To build a Junk Dimension table for a table in a staging database, follow the steps below:

1. Right click on a table, navigate to **Advanced** and click **Add Junk Dimension Table**.



2. The **Add Junk Dimension** dialog box appears. By default, the table name will be "Dim<SourceTableName>Info". If you wish the table to have another name, type it in the **Name** box.

3. Choose which fields to include in the Junk Dimension in the list under **Map the fields to add to the Junk Dimension**. By default, the **Field Name** in the Junk Dimension table will be the same as the chosen field. I you wish the field to have another name, click on the field in the **Field Name** column and rename it.


Edit Junk Dimension Table

Name: DimFactTableInfo

Map the fields to add to the Junk Dimension

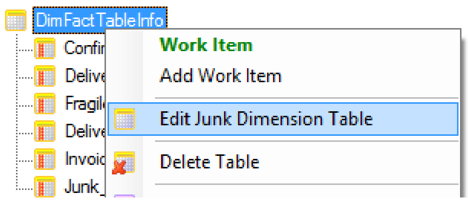
	Table: FactTable	Field name
▶	Confirmed	Confirmed
	Delivered	Delivered
	Fragile	Fragile
	DeliveryMethod	DeliveryMethod
	Invoiced	Invoiced
*		

☐ Use Binary Key

4. Select **Use Binary Key** to use a binary key for unique identification of the Junk Dimension Table.
   Note: The **Use Binary Key** option is not supported by Analysis Services cubes.

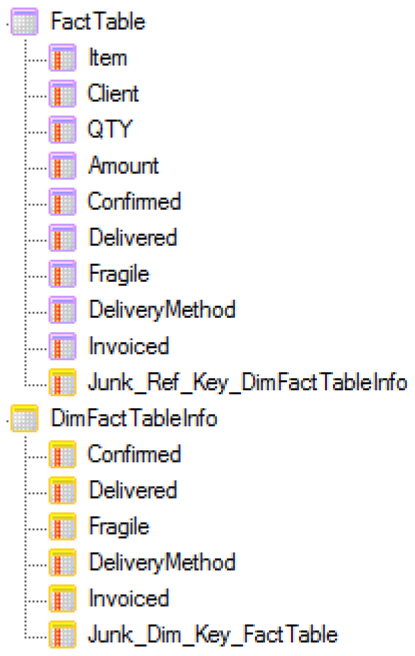
About Junk Dimension Table keys
By default, the key that identifies a specific Junk Dimension combination is a BigInt representing a hashed version of the Junk Dimension attributes. A BigInt is a 8 byte integer. This makes it possible, but unlikely, that 2 specific combinations can be assigned the same ID. If you select **Use Binary Key**, the key will be based on a 20 byte Varbinary datatype which significantly lowers the risk of getting the same value for 2 different combinations. However, Varbinary fields are not supported by Analysis Services cubes.

Adding fields to a Junk Dimension Table
When the junk dimension table has been added, you can add additional fields to the Junk Dimension by right clicking on the Junk Dimension and clicking on **Edit Junk Dimension Table** to bring up the dialog box for choosing fields.



Working with the Junk Dimension Table

When you have added a Junk Dimension Table, it appears in the project tree with a yellow table icon with a "J" on it. You can add fields, lookup fields and transformations to the Junk Dimension Table as well as custom data and data inserts.


```
FactTable
  Item
  Client
  QTY
  Amount
  Confirmed
  Delivered
  Fragile
  DeliveryMethod
  Invoiced
  Junk_Ref_Key_DimFactTableInfo
DimFactTableInfo
  Confirmed
  Delivered
  Fragile
  DeliveryMethod
  Invoiced
  Junk_Dim_Key_FactTable
```

When you have added a Junk Dimension Table to a table in a staging base, the next step is to add the table and corresponding Junk Dimension Table to the data warehouse. You do not need to add the fields on the table that are part of the Junk Dimension. This saves you storage space in the database.

You can add fields to the Junk Dimension Table in the data warehouse. For instance, in the example shown above, you could add a Delivery Instructions field to tell if the delivery should be handled with care. This could be accomplished with a field transformation with conditions that set the field value to "Fragile" if the value of the Fragile field is "Yes" and "Standard" if the value of the Fragile field is not "Yes".

When the Junk Dimension Table is executed, it will insert non-existing junk dimension combinations from the fact table. The Junk dimension table has no truncation of the Raw table.

Clustered Columnstore index on SQL 2014

Clustered columnstore indexes is a SQL Server feature that significantly increases query performance and data compression. The Clustered Columnstore Index accepts DML, so records can be inserted, updated and deleted. This is different from the Non-Clustered Columnstore index that was introduces in SQL Server 2012.

Note: The feature is only available on SQL Server 2014 Enterprise Edition.

To enable clustered columnstore index

You can enable the columnstore index on a table to table basis.

1. Right click on a table, navigate to **Advanced** and click **Indexes**.
2. The **Index Settings** window appears. Click **Add Index**.
3. (Optional) Enter a name for the index in **Friendly Index Name**.
4. Under **Index Type**, choose **Clustered Columnstore Index**.
5. Click **OK** to add the index and close the dialog. Click **Close** to close the **Index Settings** window that appears.

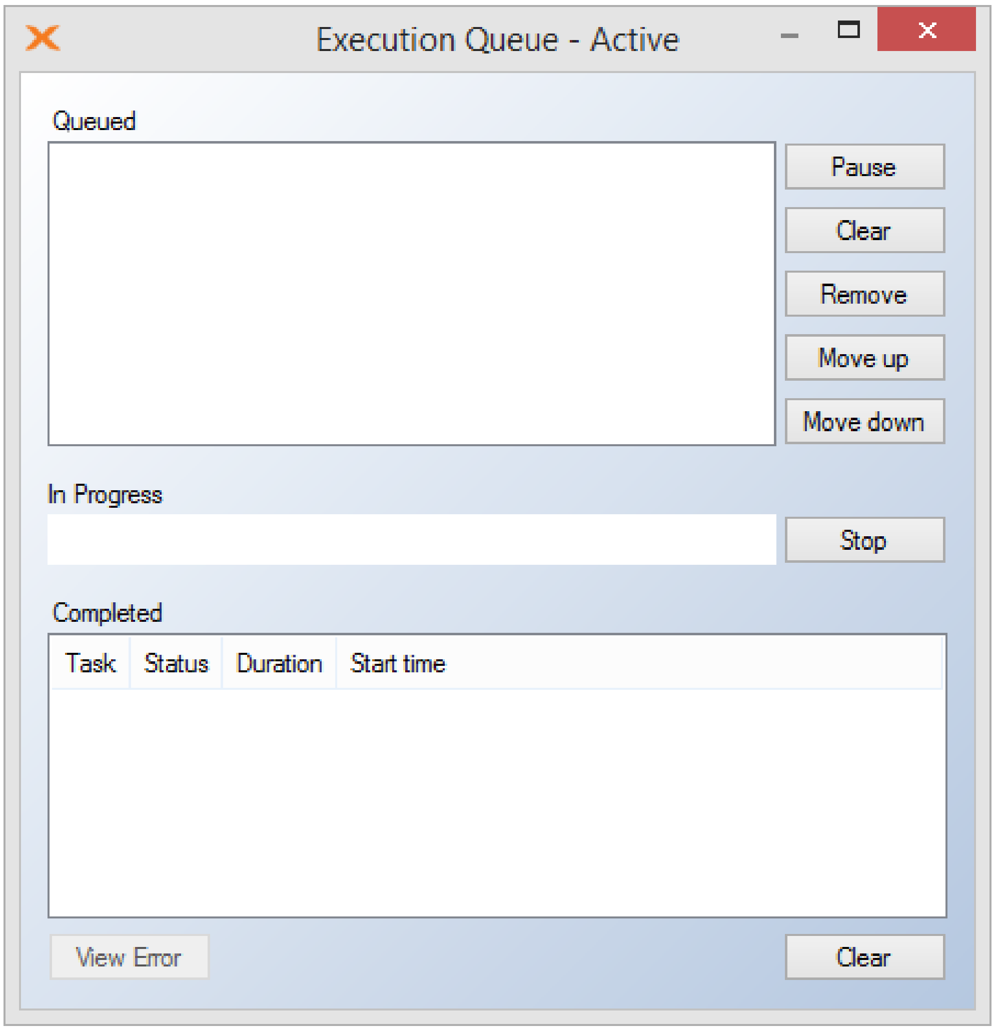
Note: If you build a Clustered Columnstore Index on a table, then no further indexes are allowed.

Execution Queue

Often, when you work with projects, you find yourself waiting for an object to finish execution before you can continue working on your project. With the Execution Queue, you can continue working with your project while TX is silently executing your requested objects in the background.

Opening the Execution Queue window

Navigate to the **Project** ribbon tab, locate the **Development** group and click on **Execution Queue** to open the Execution Queue window.

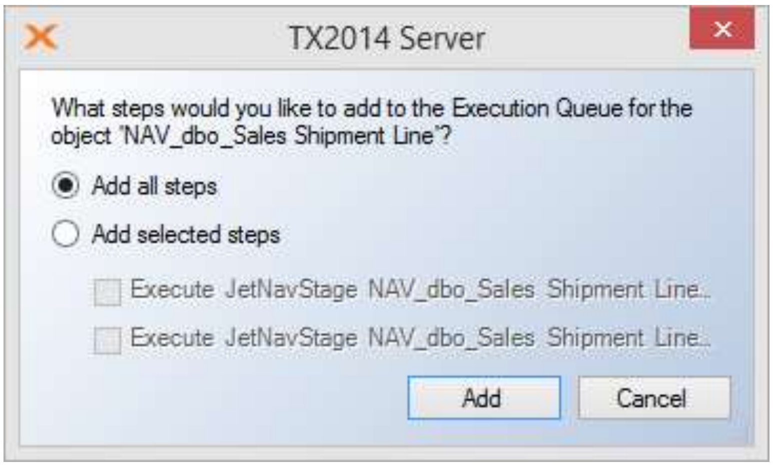


Adding an object to the Execution Queue

Adding an object to the Execution Queue is a simple drag-and-drop operation.

1. Click and hold the left mouse button on a table or an execution package, drag it to the Execution Queue window and let go of the mouse button.

2. A dialog appears to let you choose which execution steps from the object to add to the queue.



What steps would you like to add to the Execution Queue for the object 'NAV_dbo_Sales Shipment Line'?

[✓ Add all steps]

Add selected steps

Execute JetNavStage NAV_dbo_Sales Shipment Line...

Execute JetNavStage NAV_dbo_Sales Shipment Line...

Select **Add all steps** or **Add selected steps** and choose which steps you would like to add to the queue. Click **Add** to add the object to the queue.

3. The object is now listed in the **Queued** list of the Execution Queue. If there is no other items in the queue, the object will be moved to **In Progress** and begin executing immediately.

Working with the Execution Queue

Pausing and running the queue

The Execution Queued mode can de toggled between running and pause using the button in the top right corner of the window. When the queue is running, the button is called **Pause**. Clicking the button prevents further objects from being executed and changes the button text to **Resume**. Pressing **Pause** does not stop an object that is currently in progress. Pressing the **Resume** button resumes executing of the queue.

Moving and removing queued items

The **Queued** list shows the items waiting to be executed.

The queued objects can be moved up and down in the list by selecting the item and using the **Move up** and **Move down** buttons. The top item in the list is the next to be executed.

An object can be removed from list by selecting it and clicking **Remove**. Clicking **Clear** removes all items from the list.

Stopping current execution

In Progress shows the object currently being executed. Pressing **Stop** halts the execution of the object and pauses execution of the queue.

Removing executed items and viewing errors
The **Completed** list shows the objects that have been executed. It lists the **Status** of the individual items, the **Duration** and the **Start Time**. The items can have one of four statuses:

- **Success** means that the object was executed without errors.
- **Failed** means that the execution was ended prematurely by an error.
- **Stopped** means that the execution was stopped by the user before it completed.

You can view error messages for failed objects by selecting it in the list an clicking **View Error**. This brings up a message box displaying the error message.

Closing the Execution Queue window

You can close the Execution Queue window by clicking the X in the top right corner.

Closing the Execution Queue window or closing the entire project does not stop or pause the execution of the queued objects. It only hides the window, while the Execution Queue will continue working in the background. You can open the Execution Queue window again to review the status of the objects in the queue or to add more objects to the queue.

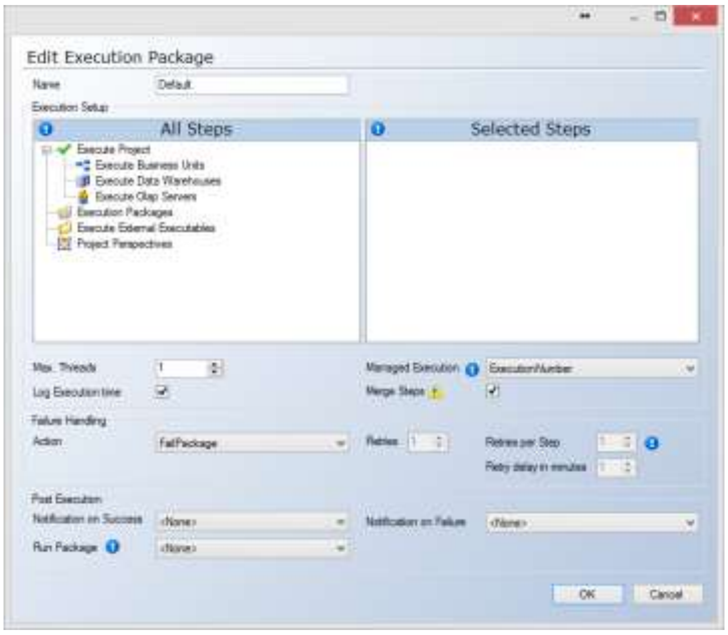
When you close TX, the Execution Queue will be stopped along with the rest of the application.

Default Execution Package

In TX2014 SR1 we introduced Managed Execution where TX will decide the execution order of the individual objects while respecting the implicit and explicit dependencies between objects. However, when you did a manual execution of the project by clicking the **Manual Deployment and Execution** button – or selecting it from the project tree, then TX would use the old sequential one-threaded execution.

With this new feature TX will use a default execution package, where the user can select the number of threads, priority order etc. The package will be dynamically generated based on where the user selects to execute it – i.e. the entire project, data warehouse, business unit or single object.

When a new project is created, a default execution package named "Default" is added to the project. You can make another execution package the default execution package by right clicking on the package and selecting **Set as Default Execution Package**.


Edit Execution Package

Name: Default

Execution Setup

Max. Threads: 1

Log Execution time: ✓

Managed Execution: ExecutionNumber

Merge Steps: ✓

Failure Handling

Action: FailPackage

Retries: 1

Retries per Step: 1

Retry delay in minutes: 1

Post Execution

Notification on Success: <None>

Notification on Failure: <None>

Run Package: <None>

Except for adjusting the **Max Threads** parameter and the **Managed Execution** setting, there is normally no need to make changes to the default execution package.

If you add steps to the **Selected Steps** section of the default execution package, then the steps will always be executed – along with the dynamically added steps.

Merge Option on Execution Packages

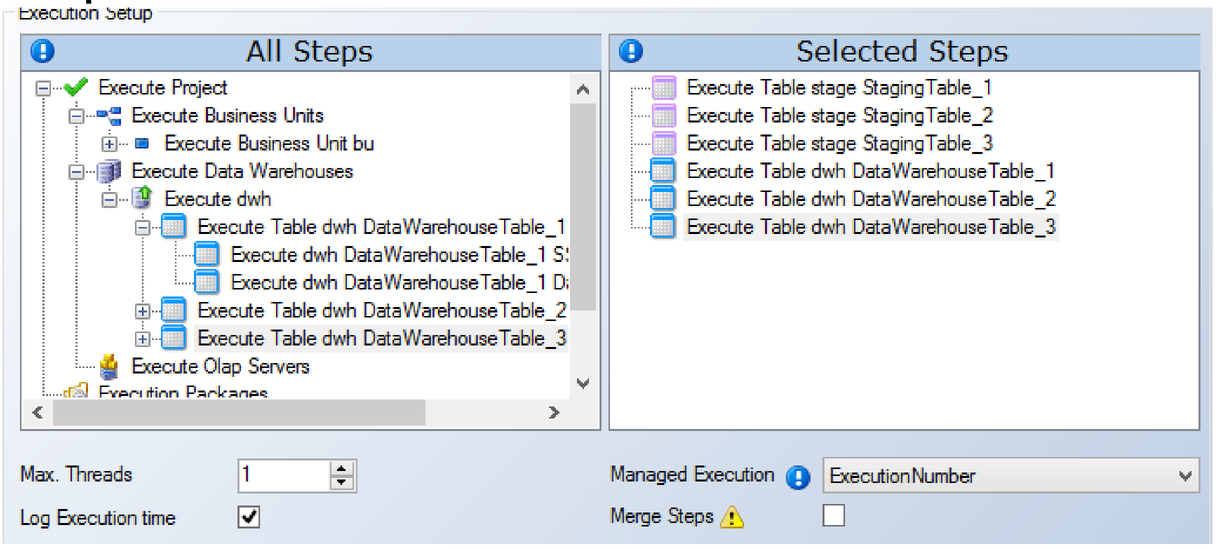
If multiple steps are added in an execution package, then these are treated as an ordered list. The steps will be executed in the order they appear in the list. When using multi-threaded execution, TX2014 will start steps in parallel without considering dependencies between objects.

With the Merge option for execution packages, TX2014 will no longer respect the order in which the steps appear in the list but will instead execute the steps based on the dependencies between the steps.

Examples

Here are some examples to give you a better understanding on how it works:

Example 1

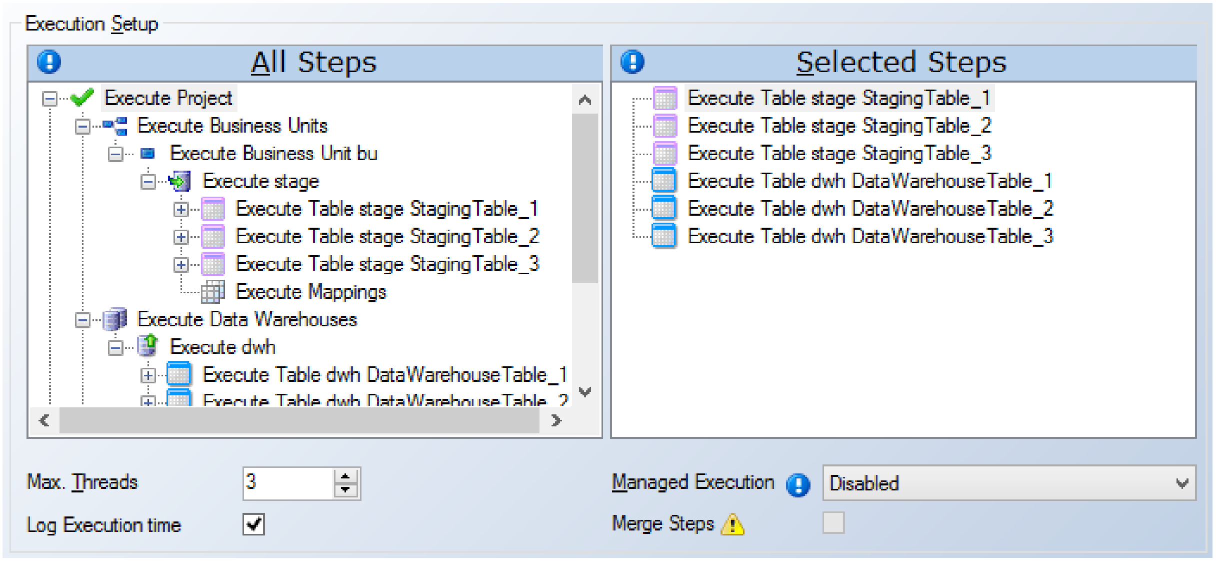

Execution Setup
All Steps
Execute Project
Execute Business Units
Execute Business Unit bu
Execute Data Warehouses
Execute dwh
Execute Table dwh DataWarehouseTable_1
Execute dwh DataWarehouseTable_1 S
Execute dwh DataWarehouseTable_1 D
Execute Table dwh DataWarehouseTable_2
Execute Table dwh DataWarehouseTable_3
Execute Olap Servers
Execution Packages
Selected Steps

Execute Table stage StagingTable_1
Execute Table stage StagingTable_2
Execute Table stage StagingTable_3
Execute Table dwh DataWarehouseTable_1
Execute Table dwh DataWarehouseTable_2
Execute Table dwh DataWarehouseTable_3

Max. Threads: 1
Log Execution time: ✓
Managed Execution: ExecutionNumber
Merge Steps: ☐

Tables will be executed sequentially in the order they appear: StagingTable1, StagingTable2, StagingTable3, DataWarehouseTable1, DataWarehouseTable2, DataWarehouseTable3

Example 2


Execution Setup

All Steps

- Execute Project
  - Execute Business Units
    - Execute Business Unit bu
      - Execute stage
        - Execute Table stage StagingTable_1
        - Execute Table stage StagingTable_2
        - Execute Table stage StagingTable_3
        - Execute Mappings
  - Execute Data Warehouses
    - Execute dwh
      - Execute Table dwh DataWarehouseTable_1
      - Execute Table dwh DataWarehouseTable_2

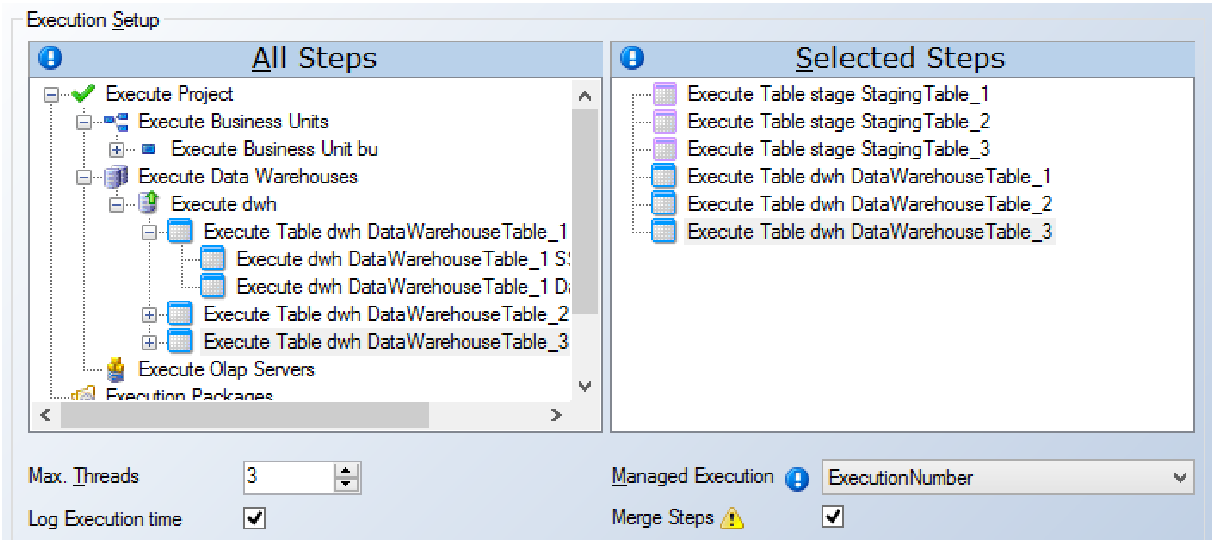
Selected Steps

- Execute Table stage StagingTable_1
- Execute Table stage StagingTable_2
- Execute Table stage StagingTable_3
- Execute Table dwh DataWarehouseTable_1
- Execute Table dwh DataWarehouseTable_2
- Execute Table dwh DataWarehouseTable_3

Max. Threads: 3
Log Execution time: ✓
Managed Execution: Disabled
Merge Steps: ☐

Selected steps will be executed in the order they appear. 3 tasks will be running in parallel, not respecting dependencies. StagingTable_1, StagingTable_2 and StagingTable_3 will start at the same time. When the first table is finished, DataWarehouseTable_1 will start. When the next table is finished, DataWarehouseTable_2 will start, etc.

Example 3


Execution Setup

All Steps

- Execute Project
  - Execute Business Units
    - Execute Business Unit bu
  - Execute Data Warehouses
    - Execute dwh
      - Execute Table dwh DataWarehouseTable_1
        - Execute dwh DataWarehouseTable_1 S:
        - Execute dwh DataWarehouseTable_1 D:
      - Execute Table dwh DataWarehouseTable_2
      - Execute Table dwh DataWarehouseTable_3
  - Execute Olap Servers
- Execution Packages

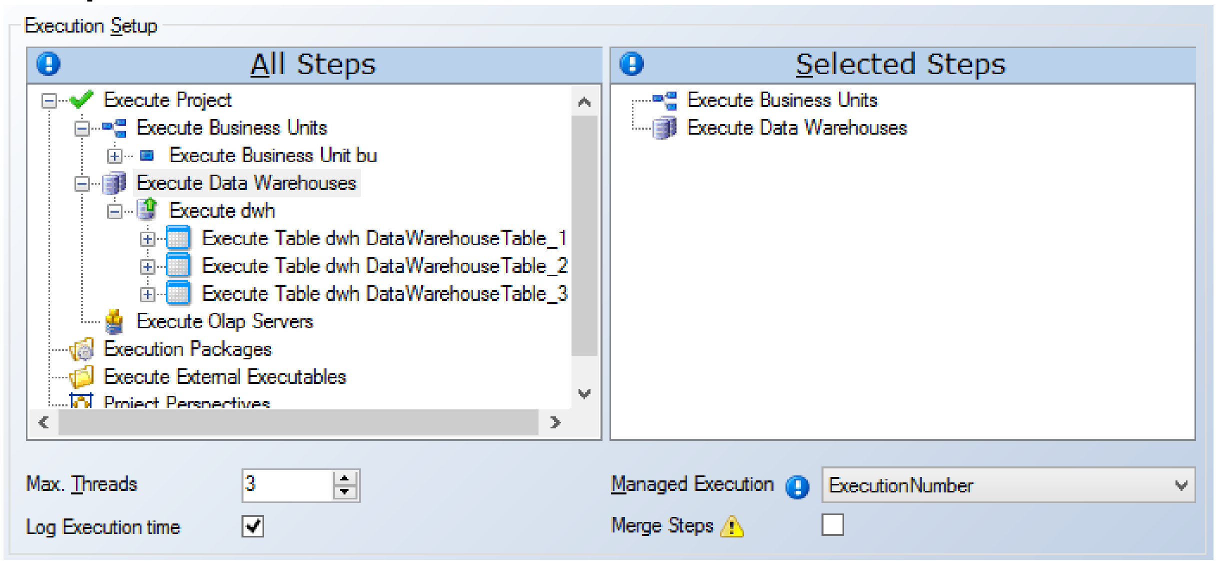
Selected Steps

- Execute Table stage StagingTable_1
- Execute Table stage StagingTable_2
- Execute Table stage StagingTable_3
- Execute Table dwh DataWarehouseTable_1
- Execute Table dwh DataWarehouseTable_2
- Execute Table dwh DataWarehouseTable_3

Max. Threads: 3
Log Execution time: ✓
Managed Execution: ExecutionNumber
Merge Steps: ✓

Selected steps will be executed in parallel using a maximum of 3 threads. The execution order is determined on dependencies. If multiple objects can run at the same time, then TX is using "Execution Number" for prioritization.

Example 4


Execution Setup

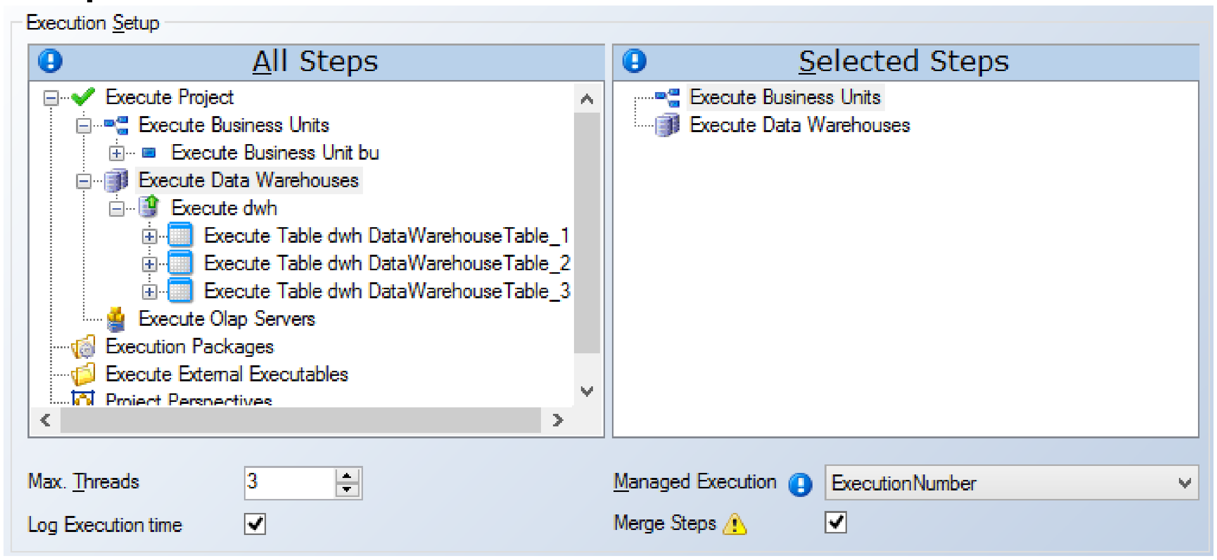
All Steps
- Execute Project
  - Execute Business Units
    - Execute Business Unit bu
  - Execute Data Warehouses
    - Execute dwh
      - Execute Table dwh DataWarehouseTable_1
      - Execute Table dwh DataWarehouseTable_2
      - Execute Table dwh DataWarehouseTable_3
  - Execute Olap Servers
- Execution Packages
- Execute External Executables
- Project Perspectives

Selected Steps
- Execute Business Units
- Execute Data Warehouses

Max. Threads: 3
Log Execution time: ✓
Managed Execution: ExecutionNumber
Merge Steps: ☐

Selected steps will be executed in the order they appear. Each step will be executed in parallel using a maximum of 3 threads with the execution order based on dependencies between objects. In this example the Business Units will be executed using managed thread execution. When all business units have finished execution the execution of the data warehouses will start.

Example 5


Execution Setup

All Steps
- Execute Project
  - Execute Business Units
    - Execute Business Unit bu
  - Execute Data Warehouses
    - Execute dwh
      - Execute Table dwh DataWarehouseTable_1
      - Execute Table dwh DataWarehouseTable_2
      - Execute Table dwh DataWarehouseTable_3
  - Execute Olap Servers
- Execution Packages
- Execute External Executables
- Project Perspectives

Selected Steps
- Execute Business Units
- Execute Data Warehouses

Max. Threads: 3
Log Execution time: ✓
Managed Execution: ExecutionNumber
Merge Steps: ✓

In this example, all tasks from "Execute Business Units" and "Execute Data Warehouses" are collected in a single list and executed in parallel using a maximum of 3 threads. The execution order is determined by dependencies between objects. In this example a data warehouse table can be execute before a business unit object – if the dependencies allows it.

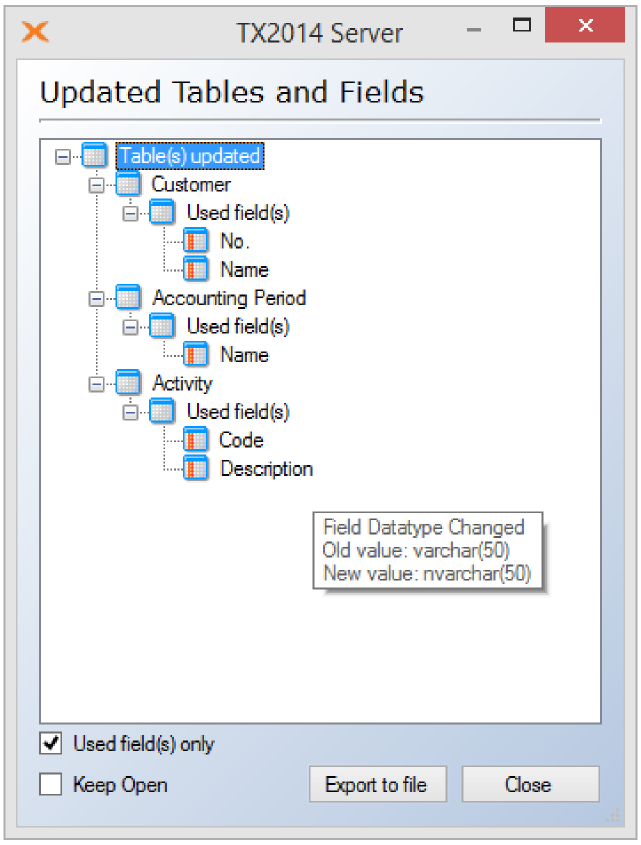
Export Synchronization Result

When you have synchronized a data source, TX tells you which tables and fields have changed compared to the last synchronization. This is useful if you e.g. change databases, but the information is not stored for later reference.

However, you have the ability to export the result of a data source synchronization to a human readable format that can be opened in Excel. This can be used as a reference when "fixing" the project and serve as documentation afterwards.

To export a syncrhonization result, follow the steps below.

1. Make any required changes on the connection settings in TX for the data source in question, e.g. a new database connection.
2. Right click on the data source you wish to synchronize and select **Synchronize Objects**.
3. When the synchronization has finished, the **Updated Tables and Fields** window appears. Click **Export to File**.
4. In the window that appears, choose where to save the result and click **Save**.

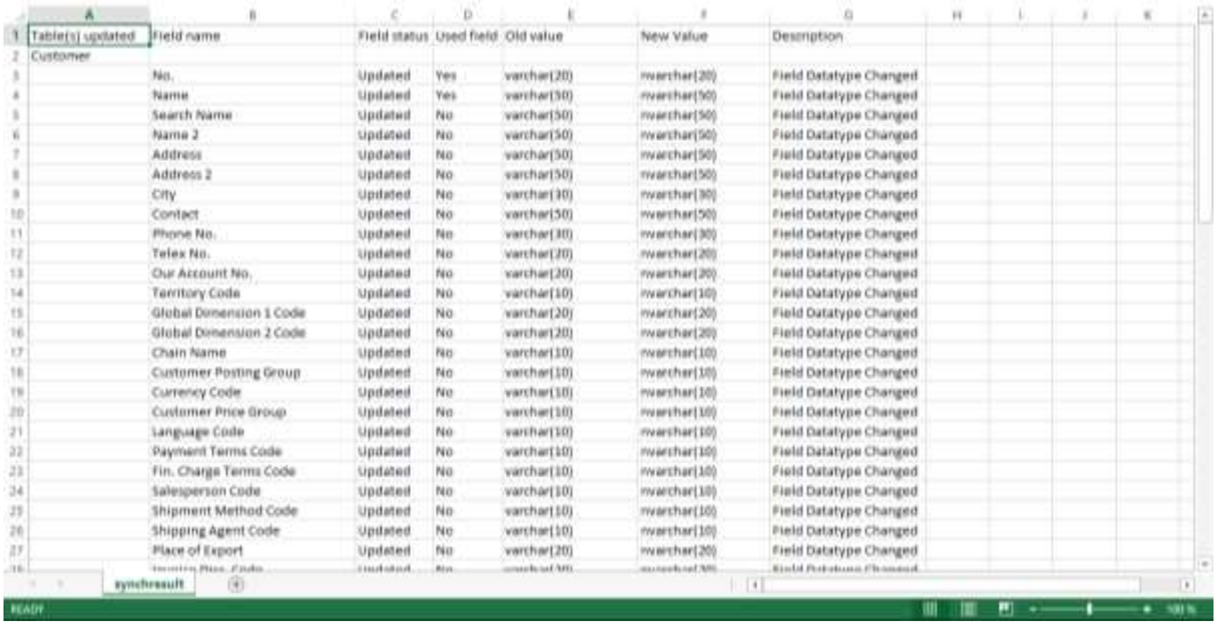


TX2014 Server

Updated Tables and Fields

- Table(s) updated
  - Customer
    - Used field(s)
      - No.
      - Name
  - Accounting Period
    - Used field(s)
      - Name
  - Activity
    - Used field(s)
      - Code
      - Description

Field Datatype Changed
Old value: varchar(50)
New value: nvarchar(50)



Table(s) updated	Field name	Field status	Used field	Old value	New Value	Description
Customer						
	No.	Updated	Yes	varchar(20)	nvarchar(20)	Field Datatype Changed
	Name	Updated	Yes	varchar(50)	nvarchar(50)	Field Datatype Changed
	Search Name	Updated	No	varchar(50)	nvarchar(50)	Field Datatype Changed
	Name 2	Updated	No	varchar(50)	nvarchar(50)	Field Datatype Changed
	Address	Updated	No	varchar(50)	nvarchar(50)	Field Datatype Changed
	Address 2	Updated	No	varchar(50)	nvarchar(50)	Field Datatype Changed
	City	Updated	No	varchar(30)	nvarchar(30)	Field Datatype Changed
	Contact	Updated	No	varchar(50)	nvarchar(50)	Field Datatype Changed
	Phone No.	Updated	No	varchar(30)	nvarchar(30)	Field Datatype Changed
	Telex No.	Updated	No	varchar(20)	nvarchar(20)	Field Datatype Changed
	Our Account No.	Updated	No	varchar(20)	nvarchar(20)	Field Datatype Changed
	Territory Code	Updated	No	varchar(10)	nvarchar(10)	Field Datatype Changed
	Global Dimension 1 Code	Updated	No	varchar(20)	nvarchar(20)	Field Datatype Changed
	Global Dimension 2 Code	Updated	No	varchar(20)	nvarchar(20)	Field Datatype Changed
	Chain Name	Updated	No	varchar(10)	nvarchar(10)	Field Datatype Changed
	Customer Posting Group	Updated	No	varchar(10)	nvarchar(10)	Field Datatype Changed
	Currency Code	Updated	No	varchar(10)	nvarchar(10)	Field Datatype Changed
	Customer Price Group	Updated	No	varchar(10)	nvarchar(10)	Field Datatype Changed
	Language Code	Updated	No	varchar(10)	nvarchar(10)	Field Datatype Changed
	Payment Terms Code	Updated	No	varchar(10)	nvarchar(10)	Field Datatype Changed
	Fin. Charge Terms Code	Updated	No	varchar(10)	nvarchar(10)	Field Datatype Changed
	Salesperson Code	Updated	No	varchar(10)	nvarchar(10)	Field Datatype Changed
	Shipment Method Code	Updated	No	varchar(10)	nvarchar(10)	Field Datatype Changed
	Shipping Agent Code	Updated	No	varchar(10)	nvarchar(10)	Field Datatype Changed
	Place of Export	Updated	No	varchar(20)	nvarchar(20)	Field Datatype Changed

Change provider on adapters and data sources

When you add a new data source, you also have to choose a provider, i.e. SQL, DB2, Oracle etc. It is, however, possible to change the provider of a data source. This enables you to e.g. switch from using an Excel spreadsheet to using a SQL database.

To change the provider for an adapter

For application adapters, you only have the choice of changing between the providers supported by the application.

1. Right click on the adapter data source and choose **Change provider to [provider]**.
2. Enter you connection settings in the **Edit [provider]** window that appears.

Refer to the section on the adapter in question to learn which providers are supported and how to set them up.

To change the provider for a data source

For standard data sources, you can change between almost all providers.

1. Right click on the data source in question and navigate to **Change Provider**
2. Click on **Change to [provider]**
3. Enter you connection settings in the **Edit [provider]** window that appears.
4. Depending on the new provider you have chosen, it might be necessary to rename tables to fit the new provider. To do that, right click on the table, navigate to **Advanced** and choose **Rename Original Table Name**.
5. Right click on the data source and choose **Synchronize Data Source** to finish changing providers.

Note: You cannot change to a Dynamics AX Adapter or a Dynamics NAV Adapter. Furthermore, you cannot change to a text file provider. However, you can change from a text file provider to another provider.

Agresso Adapter - Beta

An adapter for UNIT4 Agresso ERP system.

There is a very specific Dimension setup in Agresso that the adapter is able to handle. Without an adapter for Agresso, it is a quite complex technical challenge to get the dimensional tables constructed.

Note: This feature is in Beta. If you would like to try or test the feature, please contact us.

Infor M3 (Movex) Adapter - Beta

An adapter for Infor M3 - aka Movex ERP system.

According to current knowledge, the adapter should be able to have a central selection of Company and Department which will then be applied as a hidden selection rule on all tables. More or less like the basic operation of our Dynamics AX adapter.

Note: This feature is in Beta. If you would like to try or test the feature, please contact us.

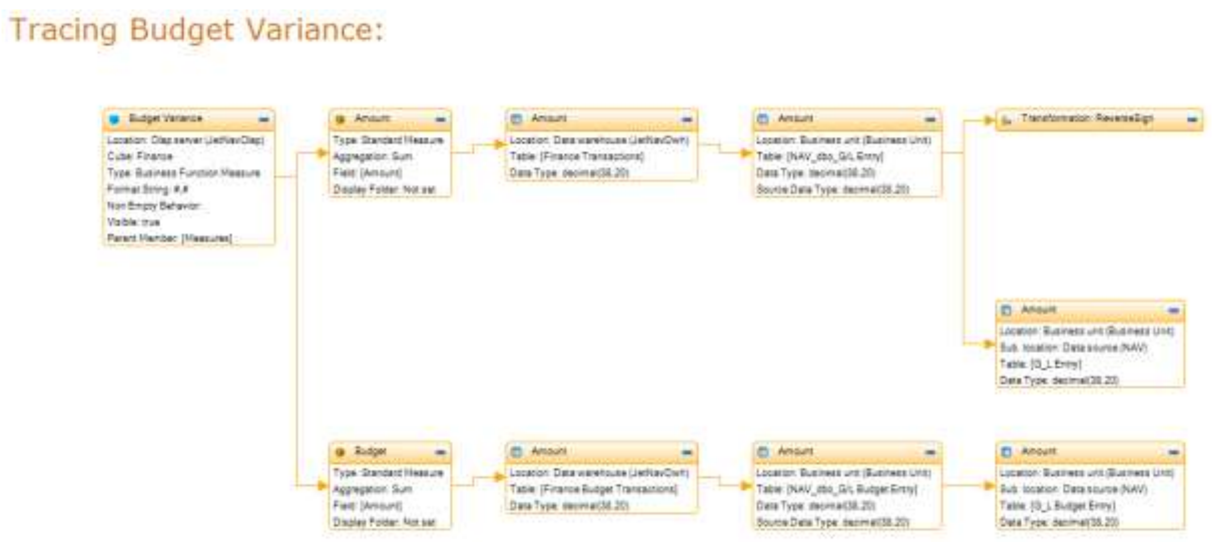
Improvements

Data Lineage and Impact Analysis improvements

The existing Trace Up and Trace Down feature has been given a brush-up to give you better results. As a consequence of this we now think that the feature provides everything you would expect for Data Lineage and Impact Analysis and we have renamed the feature to exactly that.

The Data Lineage trace can answer the question about where the content of an object comes from all the way from the Cube or Data Warehouse down to the Data Source. In other words: "Where do I come from?"

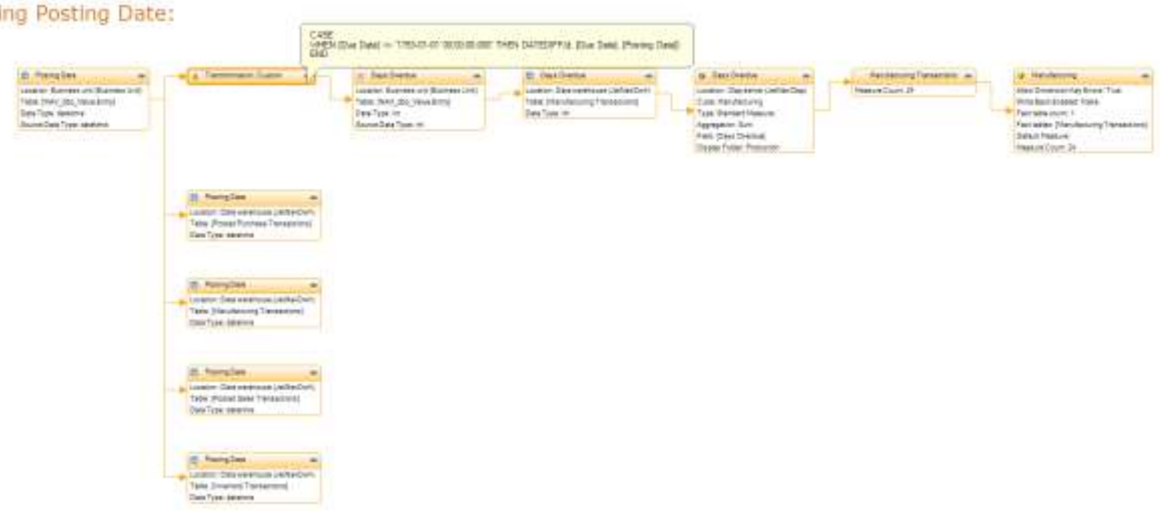
Here is an example of a Data Lineage Trace on a Business Function measure:


Tracing Budget Variance:
Budget Variance
Location: Olap server (JetNavOlap)
Cube: Finance
Type: Business Function Measure
Format String: #,#
Non Empty Behavior:
Visible: true
Parent Member: [Measures]
Amount
Type: Standard Measure
Aggregation: Sum
Field: [Amount]
Display Folder: Not set
Amount
Location: Data warehouse (JetNavDwh)
Table: [Finance Transactions]
Data Type: decimal(38,20)
Amount
Location: Business unit (Business Unit)
Table: [NAV_dbo_G/L Entry]
Data Type: decimal(38,20)
Source Data Type: decimal(38,20)
Transformation: ReverseSign
Amount
Location: Business unit (Business Unit)
Sub. location: Data source (NAV)
Table: [G_L Entry]
Data Type: decimal(38,20)
Budget
Type: Standard Measure
Aggregation: Sum
Field: [Amount]
Display Folder: Not set
Amount
Location: Data warehouse (JetNavDwh)
Table: [Finance Budget Transactions]
Data Type: decimal(38,20)
Amount
Location: Business unit (Business Unit)
Table: [NAV_dbo_G/L Budget Entry]
Data Type: decimal(38,20)
Source Data Type: decimal(38,20)
Amount
Location: Business unit (Business Unit)
Sub. location: Data source (NAV)
Table: [G_L Budget Entry]
Data Type: decimal(38,20)

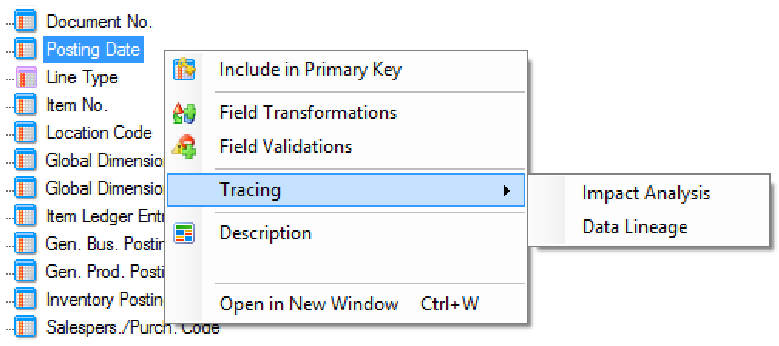
The Impact Analysis trace can answer the question about where an object is being used – all the way from the data source to a data warehouse or a Cube. In other words: "Where do I go?"

Here is an example of an Impact Analysis Trace on a Staging Table:

Tracing Posting Date:

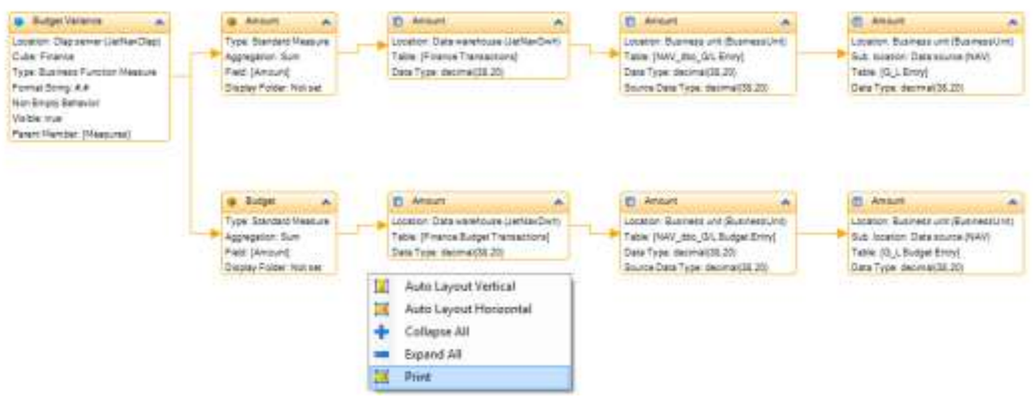


To use Data Lineage or Impact Analysis Trace, right click any traceable object and select **Trace** and then either **Impact Analysis** or **Data Lineage.**



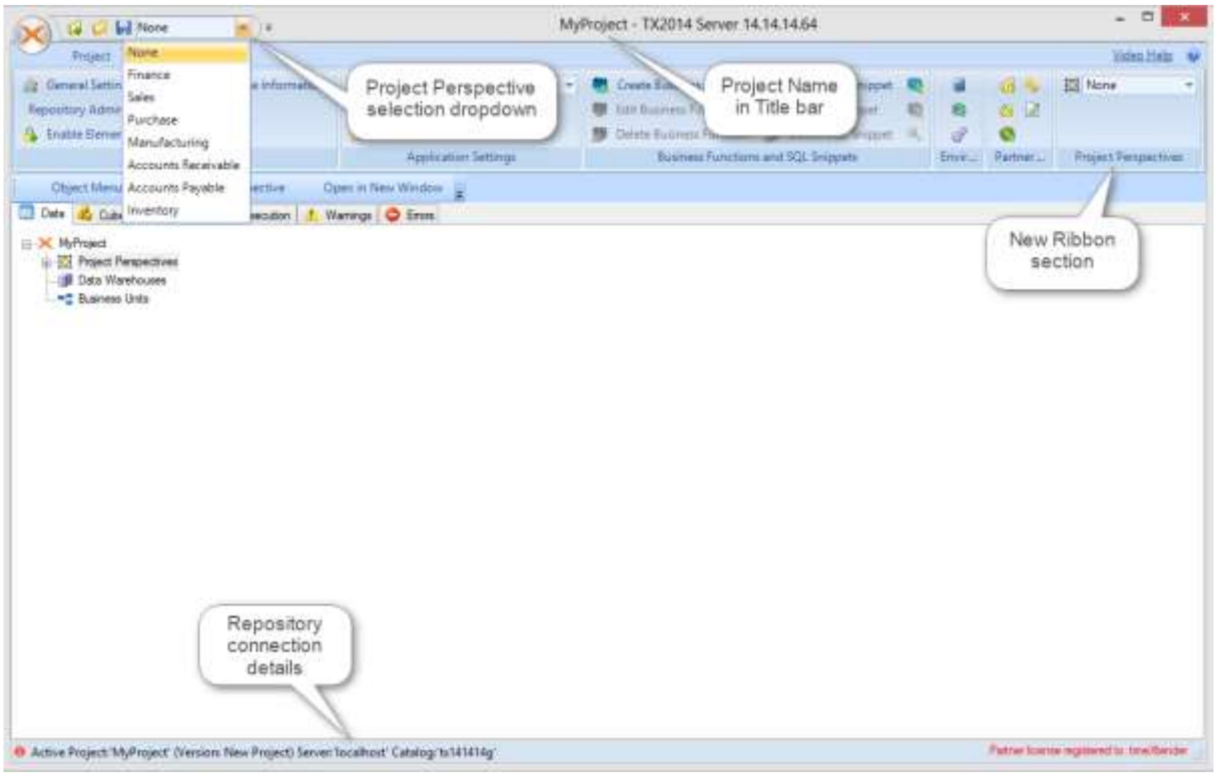
The Impact Analysis and Data Lineage output is now printable. Simply right click and select **Print.**

Tracing Budget Variance:



General User Interface Changes

The Title Bar, the Quick Access Toolbar, the Status Bar and the Ribbon has been improved to accommodate new features and provide you with more information.



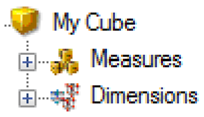
Active Project 'MyProject' (Version: New Project) Server: 'localhost' Catalog: 'ts141414g'

Partner license registered to: timextender

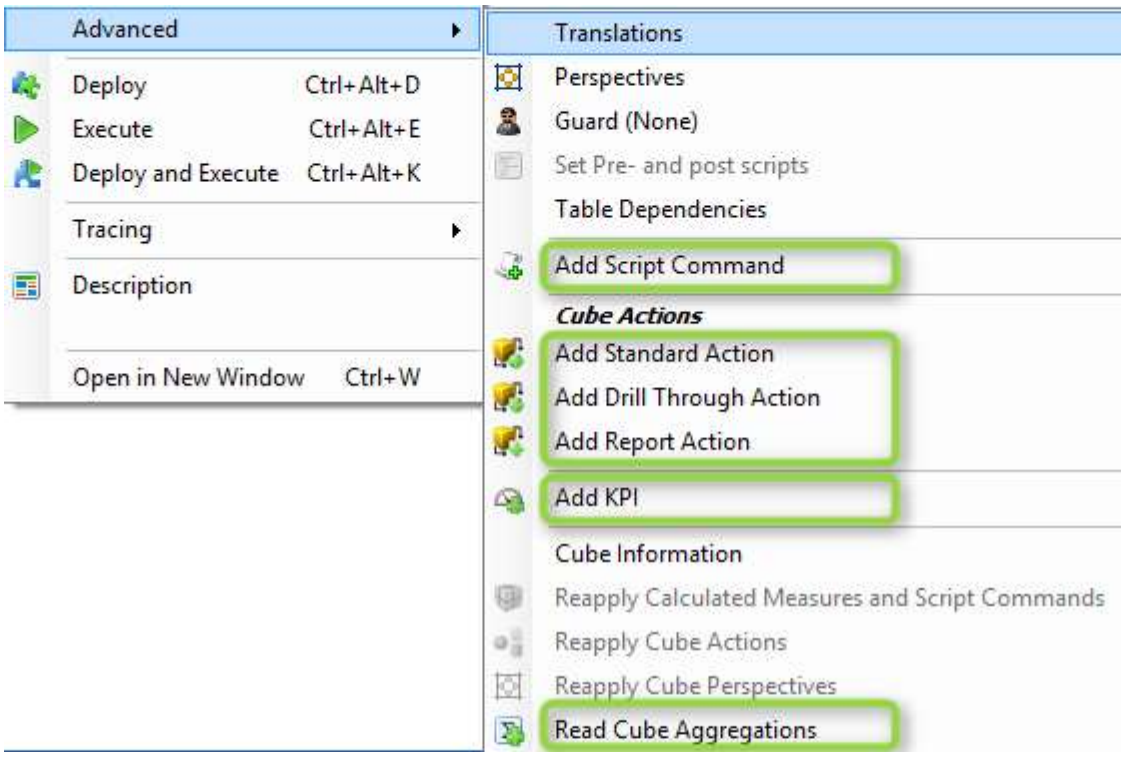
Simplified OLAP Server View

Many users rarely use some of the features on cubes. Each of these features has had its own folder under the cube in the project tree, cluttering the tree without providing value for most users. TX2014 SR2 hides these folders until the corresponding feature is actually used.

As a result, the cubes have less folders by default.



To add one of the more rarely used cube features, right click the cube and use the **Advanced** menu.



Once one of the features are in use, the corresponding folder will appear under the cube:

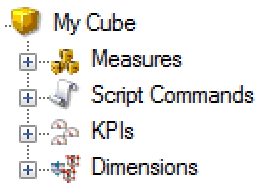
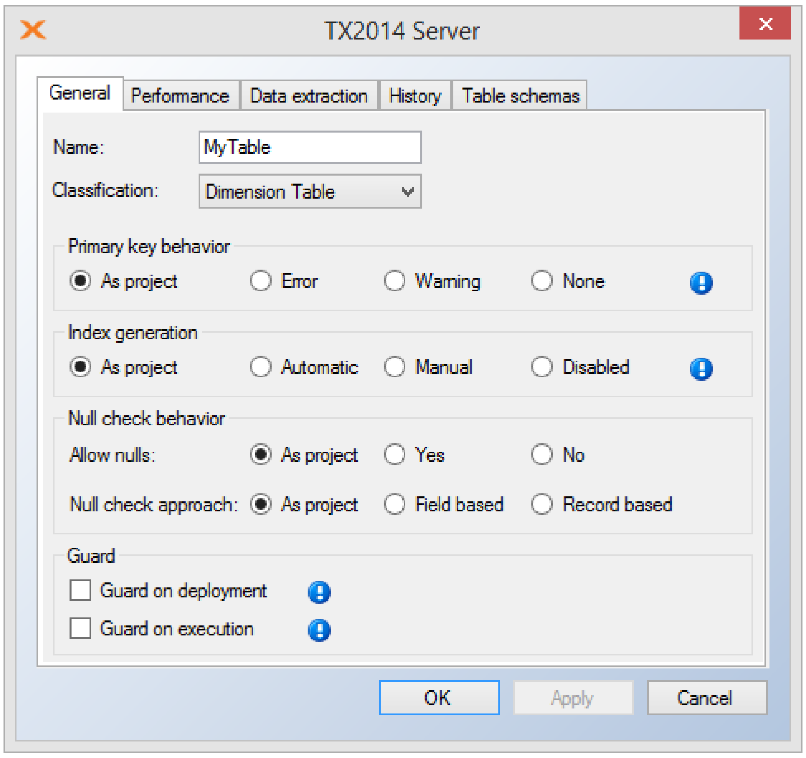


Table settings consolidation

In TX2014 SR2, a number of table settings that used to have their own entry in the context menu, have been moved to a **Table Settings** window. This reduces the clutter in the context menu and makes it easier for you to find the setting you are looking for since many of them can be found in the same location.



Name: MyTable
Classification: Dimension Table

Primary key behavior
(● As project) ○ Error ○ Warning ○ None

Index generation
(● As project) ○ Automatic ○ Manual ○ Disabled

Null check behavior
Allow nulls: (● As project) ○ Yes ○ No
Null check approach: (● As project) ○ Field based ○ Record based

Guard
☐ Guard on deployment
☐ Guard on execution

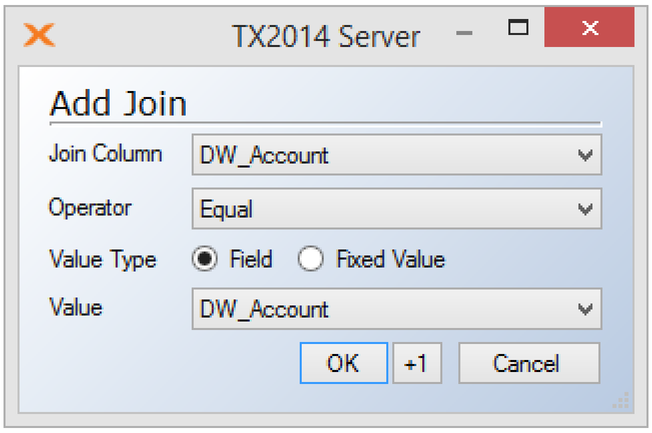
The following settings can be found in the **Table Settings** window:

- General Settings: Name, Classification, Partitioning, Primary key behavior, Index generation, Null check behavior, Guard
- Performance: Enable physical valid table, Bind functions, Enable BK hash key, Use legacy transformations, Compression, Table Partitioning
- Data Extraction: Incremental load, Truncation
- History
- Table Schemas

We have implemented some validations in the window to handle the settings that could conflict. For instance, **History** and **Truncate valid table before data cleansing** cannot be selected at the same time. If they are, error icons will appear in the tab header and by the settings to alert you. If you click **OK** without resolving the errors, an error dialog will appear detailing the conflicts.

Improved Usability in the Lookup Join Dialogue

The Add Join dialog in previous versions of TX2014 could be a bit confusing. In TX2014 SR2, we have simplified it to make it easier to understand.


TX2014 Server

Add Join

Join Column: DW_Account

Operator: Equal

Value Type: [✓ Field] Fixed Value

Value: DW_Account

The new dialog contains four items:

- **Join column** is the column on the source.
- **Operator** is the operator used in the comparison.
- **Value type** decides if a **Field** on the destination table or a **Fixed Value** should be used in the comparison.
- **Value** is either a field on the destination table or a fixed value depending on the chosen value type.

Changes in Primary Key Check

Empty Strings are allowed in primary key fields on SQL server, but TX will mark these records as errors because of the implicit check on PK fields for Null values and Empty strings.

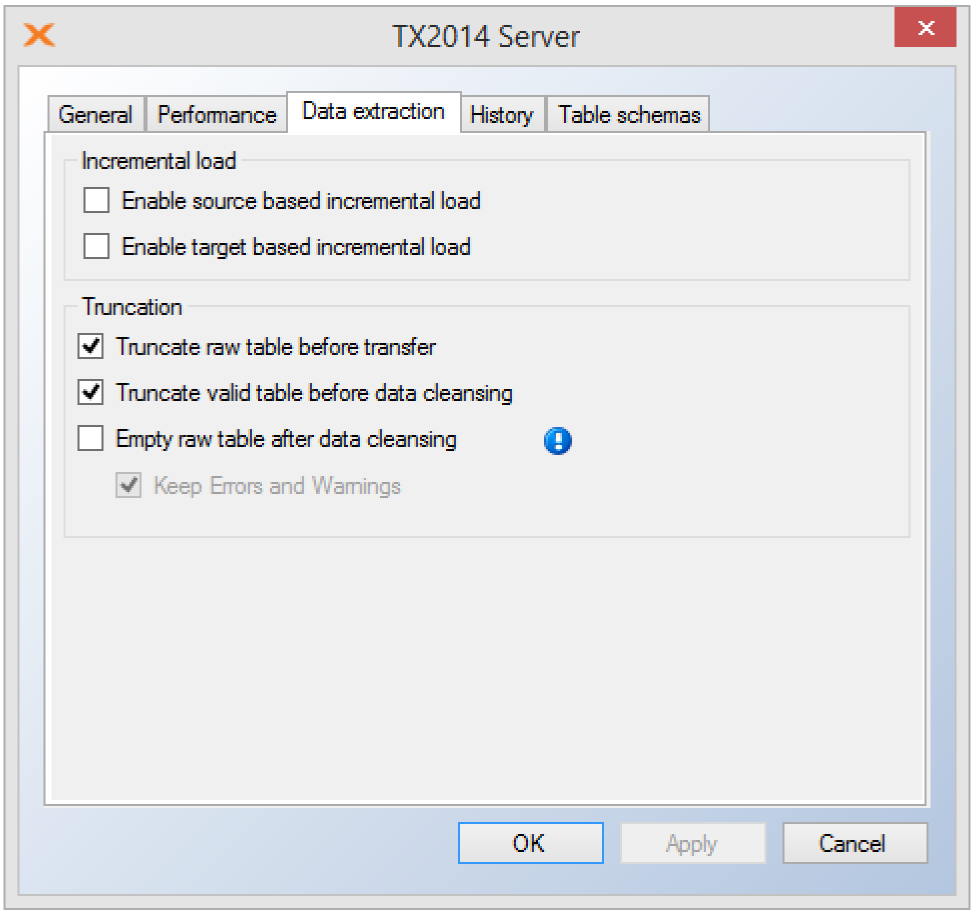
The check has been changed in TX2014 SR2, so it just checks for NULL values, not for Empty strings.

Changes in truncation settings

In TX2014 SR2, the **Truncate raw table after data cleansing** option has been renamed **Empty raw table after data cleansing**. Behind this name, change lies a small improvement. You now have the option to select if you want to **Keep Errors and Warnings** or not, i.e. keep the rows in your raw table that have caused an error or a warning.

If you choose to keep the errors and warnings, TX2014 deletes the rows without errors or warnings from the raw table. If you choose not to keep errors and warnings, TX2014 truncates the raw table, which is faster than deleting. This means that you should not select **Keep Errors and Warnings** if you do not need the info provided by the feature.

To adjust truncation settings, right click on the table, click on **Table Settings** and click on the **Data extraction** tab.



Incremental load

- [ ] Enable source based incremental load
- [ ] Enable target based incremental load

Truncation

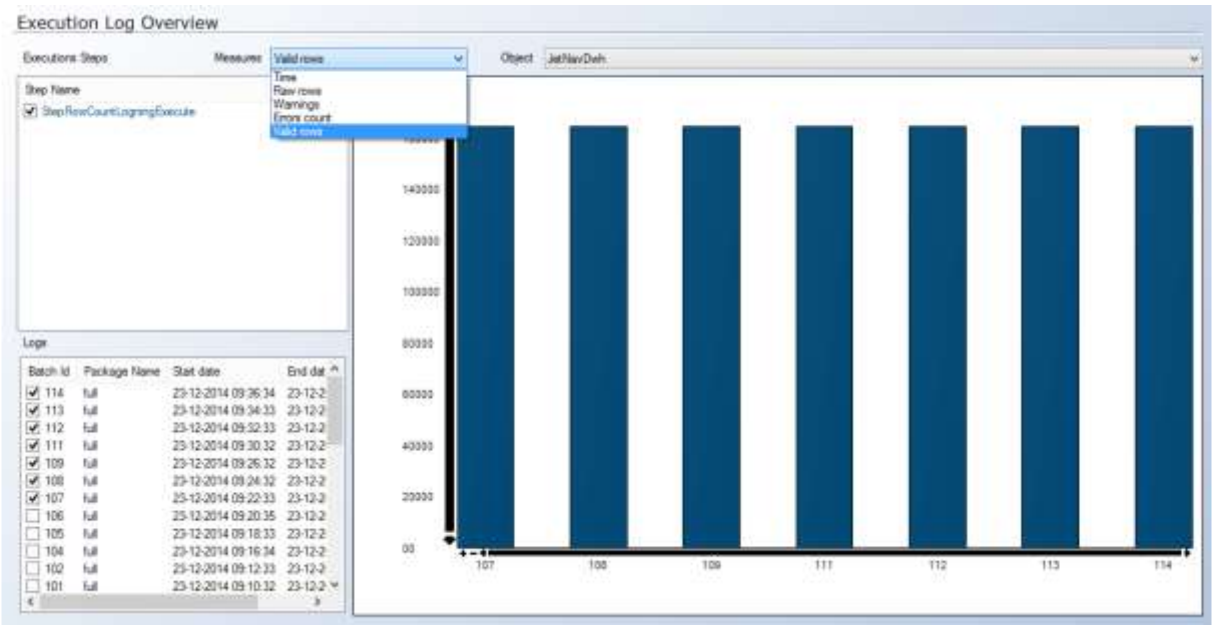
- [x] Truncate raw table before transfer
- [x] Truncate valid table before data cleansing
- [ ] Empty raw table after data cleansing
  - [x] Keep Errors and Warnings

Improved Execution Logging

Before TX2014 SR2, TX2014 only logged start- and endtime for each step during execution. TX2014 SR2 will add more logging information to the execution log:

- Number of Raw Records
- Number of Error rows
- Number of Warning Rows
- Number of Valid Rows

The result of the loggings can be seen when using "View Execution Overview Log" on the Data Warehouse, the Business Unit or the Individual table:

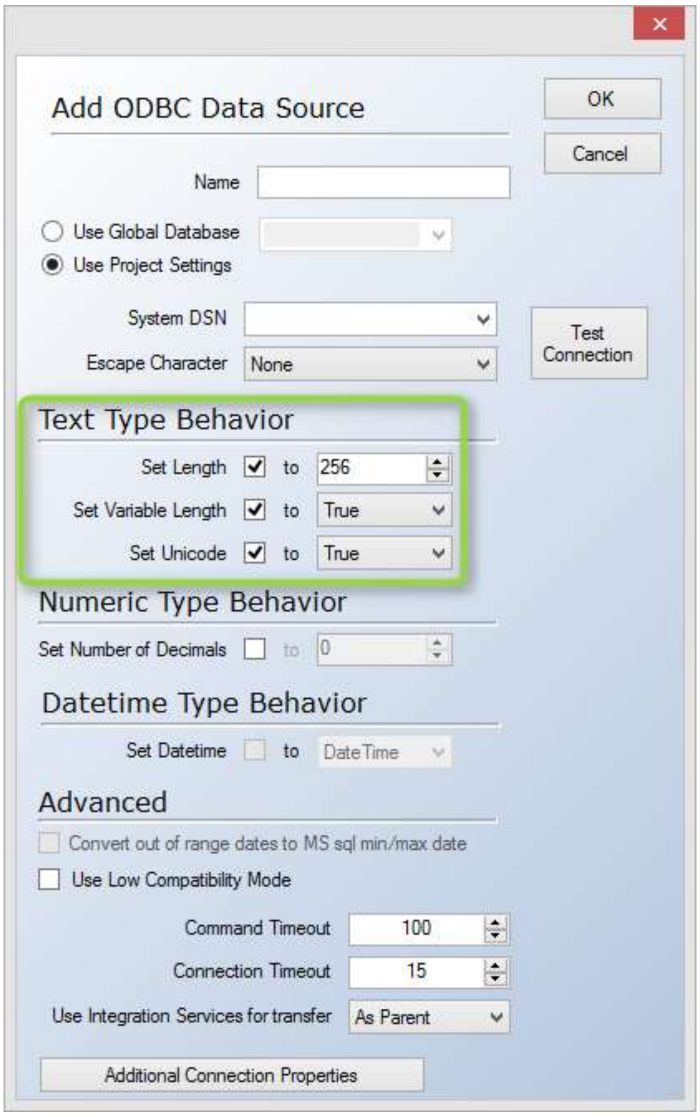


Batch Id	Package Name	Start date	End dat
✓ 114	full	23-12-2014 09:36:34	23-12-2
✓ 113	full	23-12-2014 09:34:33	23-12-2
✓ 112	full	23-12-2014 09:32:33	23-12-2
✓ 111	full	23-12-2014 09:30:32	23-12-2
✓ 109	full	23-12-2014 09:26:32	23-12-2
✓ 108	full	23-12-2014 09:24:32	23-12-2
✓ 107	full	23-12-2014 09:22:33	23-12-2
☐ 106	full	23-12-2014 09:20:35	23-12-2
☐ 105	full	23-12-2014 09:18:33	23-12-2
☐ 104	full	23-12-2014 09:16:34	23-12-2
☐ 102	full	23-12-2014 09:12:33	23-12-2
☐ 101	full	23-12-2014 09:10:32	23-12-2

You can select between the different row counts in the Measures drop-down field.

Force specific text datatype on ODBC

You can force a specific Text datatype on your string fields from ODBC. By default, the text fields will be sized in sync with the actual field size reported from the ODBC provider. With this feature, you can override this and select another (larger) text datatype.


Add ODBC Data Source

Name	
○ Use Global Database	
● Use Project Settings	
System DSN	
Escape Character	None

Text Type Behavior

Set Length	✓	to	256
Set Variable Length	✓	to	True
Set Unicode	✓	to	True

Numeric Type Behavior

Set Number of Decimals	☐	to	0

Datetime Type Behavior

Set Datetime	☐	to	DateTime

Advanced

☐ Convert out of range dates to MS sql min/max date

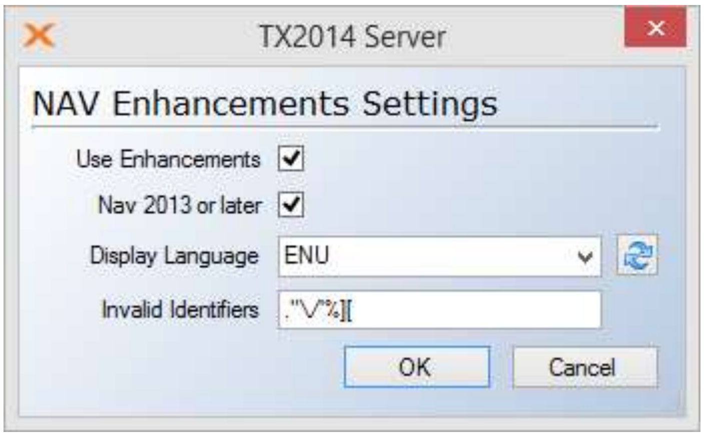
☐ Use Low Compatibility Mode

Command Timeout	100
Connection Timeout	15
Use Integration Services for transfer	As Parent

NAV Adapter Improvements

In TX2014 SR2, we have improved the Dynamics NAV Adapter to remove the hassle of importing and running code on the NAV system to use NAV Enhancements. For technical reasons, this is only an option on Dynamics NAV 2013 or newer. In previous versions, you still have to import a .fob file and run it on your system.

To take advantage of this improvement, make sure that **NAV 2013 or later** is selected in the **NAV Enhancements Settings** window during set up of the source. You can find the setting by right clicking on a NAV source, navigating to Enhancements and clicking **Navision Enhancement Settings (Enabled: xxx)**.


TX2014 Server

NAV Enhancements Settings

Use Enhancements ✓

Nav 2013 or later ✓

Display Language: ENU

Invalid Identifiers: ."\/'%][

Fixed Bugs

14.5.0

ID	Title
18	Incremental Load with Excel Sources fails
19	Manual Execution, Display Current Task issue
20	Renaming a field does not cause a "IsDirty" check on other tables
21	Excel data source cannot change from use global database
22	Custom Hash calculated on Raw values instead of Valid values
24	Error in Additional Data Sources on Oracle
25	Relations from views not loaded
26	Variable overflow in Fixed join Criteria on Lookup fields
27	Lookups under a conditional lookup field can not be sorted.
29	Calculations loses Associated Measure group when using physical perspectives
30	NAV Adapter: Crash when changing Language
31	Test Oracle Connection causes table to become dirty
32	AX Adapter - ENUM Value datatype should be Int and not BigInt
33	Partitioning function is wrong on numeric partition keys
36	When using a SQL snippet in a custom script the documentation of the project fails
40	Moving a lookup field makes the interface focus on another area of the screen
44	History tables deployment error on German SQL Servers
45	Error deleting table with SQL snippet
46	Issue with Not Empty condition on Add Related Records
49	Identity Insert of DW_ID on DWH is ignored when not using SSIS
55	GP Adapter name validation is wrong
57	Error in task count on Execution packages
60	Spelling Mistake
62	Changing a hash field does not make the table modified
63	Validation Rule Not Null cannot be added.
64	Issue with Oracle data source when database kills idle connections.

68	Save as on project loses dimension level relations
70	Add join on conditional look up
75	Documentation: Object not set to an instance of an object
77	German Integration Services problem
78	Issue with Used Fields in Synchronization result
84	Cube Perspectives: Missing Measures if Measures group name is different from Fact Table Name
91	Time table can be deleted without warning when used as partition template timetable
92	A field can be deleted without warning when used as partition template field

14.5.2

ID	Title
34	License owner GUI issue
90	DW_SourceCode default constraint missing after import
118	Merge Execution not checking for Pre- or Post- Scripts on BU or DWH
119	Hierarchy table problem.
120	Joins with fixed value will in most cases be handled wrong after upgrade to SR2
121	AX Adapter: Virtual DataAreaID fields is wrongly reported deleted by synchronization
122	Connect timeout not set on sql conneciton in SSIS package
123	GP adapter company dependant tables problem
126	DW_SouceCode is added in source select in ssis package when table is incremental loaded
127	Slow Olap Role deployment

14.5.3

ID	Title
134	CAST error with Salesforce adapter using global database connection
135	Missing Dependency check on DWH tables based on unparameterized views even though dependencies has been setup

14.5.4

ID	Title
136	Cannot use repository where a business function is named with a ' (quote)