TX DWA

# USER GUIDE

TIMEXTENDER

**TX DWA User Guide**
Version 2017-08-21

Find the latest version at
support.timextender.com

# TABLE OF CONTENTS

**Getting Started with TX DWA** ..... **6**
The User Interface ..... 7
The Workflow ..... 13
**Setting Up Your Project** ..... **16**
Projects ..... 17
Project Repositories ..... 22
Data Warehouses ..... 24
Business Units ..... 29
Staging Databases ..... 31
Team Development ..... 33
**Connecting to Data Sources** ..... **39**
SQL Server Data Source ..... 48
DB2 Data Source ..... 50
Informix Data Source ..... 51
Oracle Data Source ..... 52
MySQL Data Source ..... 54
ODBC Data Sources ..... 56
Excel Data Source ..... 58
Text File Data Source ..... 59
AnySource Data Source ..... 62
Custom Data Source ..... 64
**Connecting with Application Adapters** ..... **65**
Microsoft Dynamics AX Adapter ..... 66
Microsoft Dynamics NAV Adapter ..... 74
Microsoft Dynamics GP Adapter ..... 79
Microsoft Dynamics CRM Online Adapter ..... 81
Salesforce Adapter ..... 83
SAP Application Adapter ..... 86

Sun System adapter ..... 88
Movex/M3 Adapter ..... 89
Agresso Adapter ..... 91
**Designing the Data Warehouse** ..... **100**
Dimensional Modeling ..... 101
Data Lineage and Impact analysis ..... 103
Documentation ..... 105
Moving and Relating Data ..... 108
Selecting, Validating and transforming Data ..... 118
Previewing data ..... 124
Tables ..... 128
Fields ..... 143
Views ..... 150
Indexes ..... 154
History ..... 156
Scripting ..... 159
Database Schemas ..... 172
Data Security ..... 174
**Exporting Data** ..... **180**
Data Export ..... 181
**Online Analytical Processing (OLAP)** ..... **183**
Cubes ..... 185
Dimensions ..... 195
Measures ..... 206
Handling Early Arriving Facts ..... 210
Slowly Changing Dimensions ..... 215
**Deploying and Executing** ..... **224**
Manual Deployment and Execution ..... 226
Scheduled Execution ..... 235
Incremental Loading ..... 243

Multiple Environments ..... 253
**Qlik Modeler** ..... **265**

# GETTING STARTED WITH TX DWA

The purpose of TX DWA is to enable you to create and maintain a complete data warehouse solution with as little effort as possible. To achieve this, TX DWA employs Data Warehouse Automation techniques to automate the tedious parts of the work.

SQL code for the ETL (Extract, transform, load) process, MDX code for OLAP cubes, indexes etc. are generated automatically. Most tasks can be accomplished using drag-and-drop in the graphical user interface, and the amount of code you need to write is minimized. However, if and when you need to customize the code, you can do it in your favorite development environment.

This user guide outlines the features and functionality of TX DWA. Please note that this document does not cover detailed concepts of data modeling or the tables and fields of any particular relational database. If you need help with this, please explore our training options.

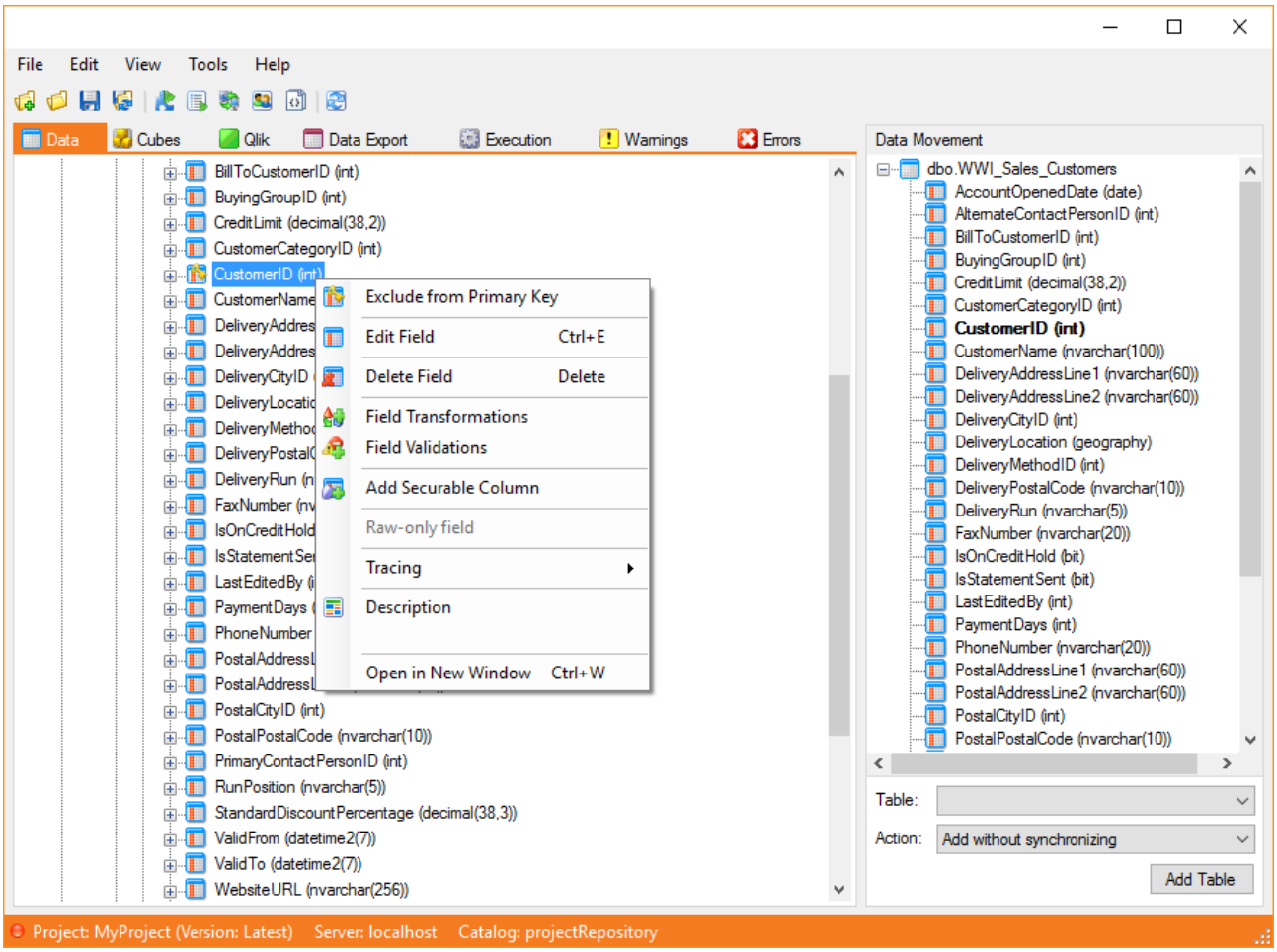
This section provides an overview of the workflow, the components and the user interface you will be working with when you build a data warehouse in TX DWA. Installation instructions for the software is provided on our support site at https://support.timextender.com

# THE USER INTERFACE

The TX DWA user interface is a Windows application based on a menu bar, a toolbar and a number of tabs that contain the different objects you will be working with.

As the number of objects can become quite large, TX DWA includes a feature, project perspectives, that lets you hide the objects you are not currently working on.

This section provides an overview of the interface and instructions for using the project perspectives feature.




## THE MENUS

The commands, settings and tools available in TX DWA are split between two menu locations. The general commands, such as new, open, find, refresh, options, help, can be found in the menu bar. Commands related to a specific object can be found in the context menu that appears when you right click the object.

For easy access to the most used commands in the menu bar, a subset of the commands can also be found in the toolbar just under the menu bar.

A lot of the most used commands have keyboard shortcuts. On all menus, you will notice that the keyboard shortcut to a command is displayed to the right of the command in the menu.

### THE VIEW MENU

This menu contains all the commands related to how the project is displayed.

Almost any object in TX DWA can be opened in a new window by right-clicking it and clicking **Open in New Window**. On the View menu, the open windows are listed under Open Windows to give you an easy overview if you find yourself with a lot of open windows.

On the View menu you can customize the user interface by toggling the display of different elements:

- **Advanced options:** When checked, the Advanced submenu is available in the context menu for some objects.
- **Data types:** When checked, the data type of fields is postfixed to their name in the project tree.
- **Deprecated features:** When checked, deprecated features are shown in the user interface. In projects that do not utilize deprecated features, we recommend that you to keep this disabled for a cleaner interface.
- **Highlight descriptions:** When checked, objects that have a description are displayed with bold text in the tree. Descriptions are added from the context menu on an object.

The View menu is also the home of the project perspectives - see Project Perspectives for more information.

## THE TABS

A project is organized in a number of tabs, where each tab contain the tasks, objects and information that relates to a specific part of the project.

### THE DATA TAB

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

### THE CUBES TAB

On the Cubes tab, you can create OLAP cubes to use for reporting in different presentation and visualization tools. 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.

### THE QLIK TAB

On the Qlik tab, you can build Qlik Sense and QlikView models based on the data in your data warehouse. You can the push these models to Qlik applications for users to see and use.

### THE 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 objects, 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.

### THE 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.

### THE 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.

## COLOR THEME

If you would like to easily tell different installations of TX DWA apart, or if you are simply tired of the default colors, you can choose another color theme for the application. The color theme is saved in the user settings for the specific version of TX DWA.

A color theme in TX DWA consists of a theme - light or dark - in conjunction with an accent color. There are five predefined accent colors available in TX DWA, giving you a total of ten color themes.

To change the color theme

- On the **Tools** menu, click **Options** and then click your prefered theme in the **Theme** list and your prefered accent color in the **Accent Color** list.

If you use multiple environments, you can select a color theme for each environment to make them easier to tell apart. The color themes available for environments are different from the regular color themes to make sure they stand out when you apply them.

## THE STATUS BAR

The status bar in the bottom of the window contains information about the currently open project. It also servers as a useful shortcut to some settings and information. The status bar

contains the following items:

- The name of the currently open project, prefixed with the version in paranthesis. In front of the project name, an icon shows you if all changes have been saved (green) or there are changes waiting to be saved (red). Click this to open the **Open Project** window.
- The name of the repository server and database (catalog). Click this to open the **Options** window on the **Project repository** tab.
- The name of the current environment if any. Click this to open the **Environment properties** window.
- The license type and what company it is registrered to. Click this to open the **License information** window.

## PROJECT PERSPECTIVES

The purpose of Project Perspectives is to make it easier to work with large projects. Working in a 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. You can also chose to make a project perspective dynamic. Any object that depend on an object already in the perspective, will then automatically be included in the perspective.

### ADDING A PROJECT PERSPECTIVE

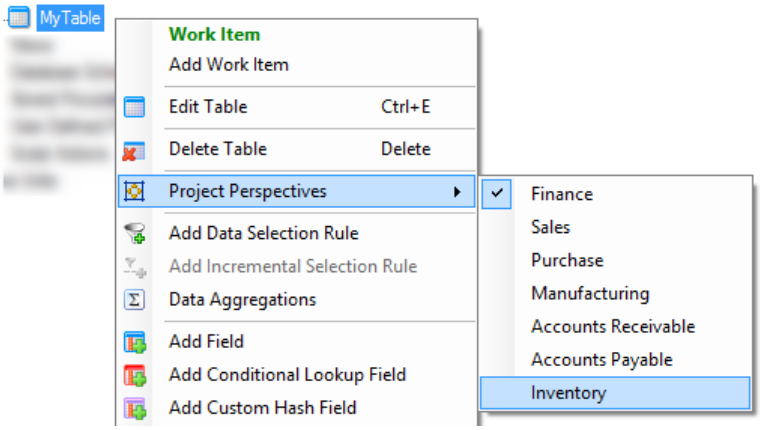
To add a project perspective, follow the steps below.

1. On the **View** menu, click **Add/Edit Project Perspectives**.
2. Click **Add Perspective**. A new column is added to the grid with the name "Perspective" in the **Perspective Name** row. Double-click the name to edit it and click outside the field when you have finished typing. Select **Dynamic Perspective** if you want the new project perspective to be dynamic. You can change this setting at any time.
3. Each Qlik model, OLAP cube, data warehouse and business unit is listed in the first column of the grid. Click the **+** beside one of the objects to show the child objects. Select the Qlik models, cubes, dimensions and tables you want to include in your project perspective. If you add an object to a dynamic perspective, dependent objects will be added automatically. You can recognize dynamically added objects by the checkbox with an indeterminate state.
4. Click **OK**. The new project perspective now appears under **Project Perspectives** in the project tree and on the **View** menu.

### 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 the object you want 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 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

There are two ways of activating a perspective:

- On the **View** menu, point to **Perspectives** and then click the perspective you want to activate.
- In the project tree, expand **Project Perspectives**, right click the perspective you want to activate and click **Use Project Perspective**.

### DEACTIVATING ALL PERSPECTIVES

To disable all perspectives and see all objects, you have two options.

- On the **View** menu, point to **Perspectives** and then click the checked perspective.
- In the project tree, expand **Project Perspectives**, right click the perspective with **(Active)** postfixed to the name click **Use Project Perspective**.

### 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. Expand **Project Perspectives** in the project tree.
2. Right click 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 two ways:

- In the project tree, expand **Project Perspectives**, right click the perspective you want to deploy and/or execute and click **Deploy**, **Execute** or **Deploy and Execute**.
- You can also add a perspective to an execution package, for example if you want to execute the perspective on a schedule.

# THE WORKFLOW

In the following, the steps you typically need to go through when you build a data warehouse solution with TX DWA are briefly described. However, you are in complete control. You can import parts of a legacy data warehouse and built a cube structure on top of that or start from scratch.

## CREATING A TX DWA PROJECT

The project is a container for all of the elements of your solution. Thus, creating or importing a project is always the first step when you build a new 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 Projects.

## DESIGNING THE DATA WAREHOUSE

TX DWA is built to utilize the star and snowflake schemas, the two standard schemas for designing a data warehouse. This means that you'll often find yourself building fact and dimension tables. The fact tables contain the transactional data you want to analyze, while the dimension tables add context to these data. However, you are free to design your data warehouse however you want to.

When you design the data warehouse in the TX DWA, you create relationships between the tables in your data warehouse, assign primary keys, create views and so on.

For more information, see Designing the Data Warehouse.

## CONNECTING TO DATA SOURCES

The TX DWA supports a wide variety of data sources, such as Microsoft SQL Server databases, Oracle Database, Microsoft Excel files and text files. Application adapters simplify access to all major ERP and CRM systems. 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 can 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, Validating and Transforming Data.

## MODELING CUBES

Some front-end presentation tools connect directly to your data warehouse. Often, however, Online Analytical Processing (OLAP) cubes will serve as the basis for your analysis and reporting solution.

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 TX DWA to maintain and change existing cubes.

For more information, see Online Analytical Processing (OLAP).

## 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.

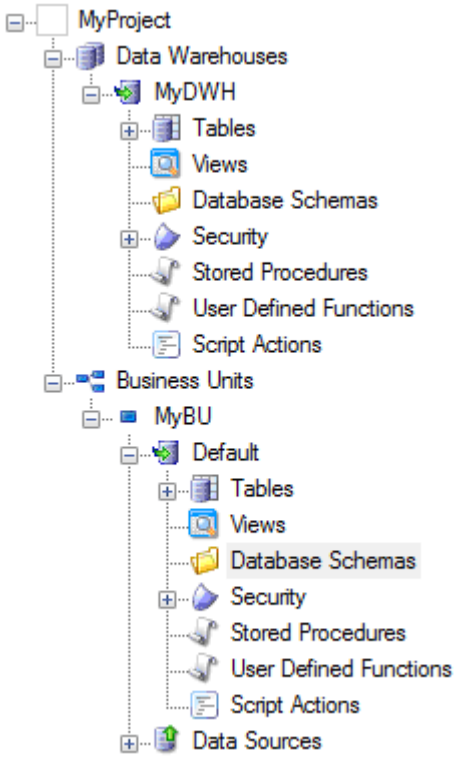
During deployment, the underlying SQL code is automatically generated and stored in the data warehouse.

For more information, see Deploying and Executing Projects.

# SETTING UP YOUR PROJECT

In TX DWA, your work is organized in projects stored in a repository on a SQL Server. Each project is a collection of different objects: data warehouses, tables, fields, business units, execution packages, OLAP servers and so on.

The objects are organized into different tabs in the user interface. In this chapter, we will cover the project itself, data warehouses, business units and staging databases, which you can all find on the Data tab.




Directly below the project in the tree are Data Warehouses and Business Units. Each can contain a number of data warehouse databases or business units. In turn, each business unit contains a staging database and a number of data sources.

You can learn more about data sources for your project in the chapters about Data Sources and Application Adapters. The content of the Cubes tab is covered in the Online Analytical Processing (OLAP) chapter, while the Execution tab is covered in the Deploying and Executing Projects chapter.

# PROJECTS

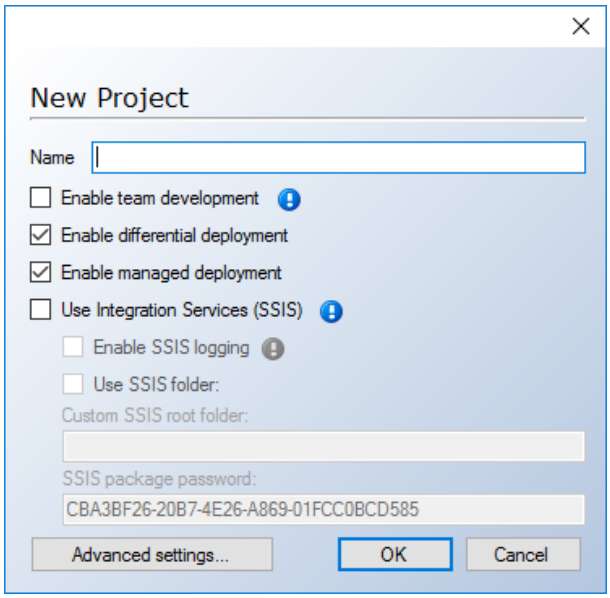
A TX DWA project contains all other elements in the data warehouse.

You can only have one project open at a time. However, if you want to compare different versions of a project, you can open another instance of TX DWA, and load another version of the project to view side-by-side.

## CREATING A PROJECT

To create a projects follow the steps below:

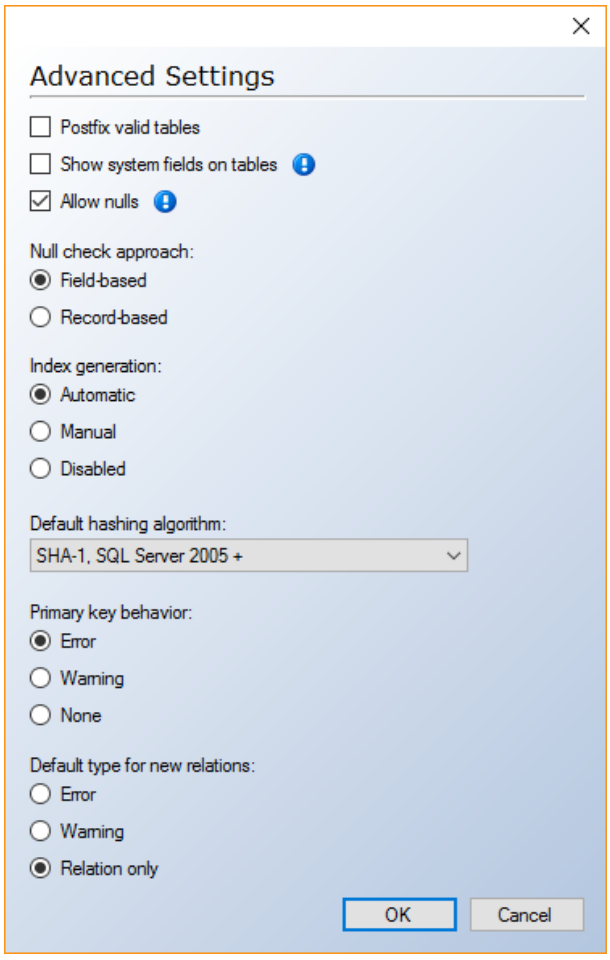
1. Click the application icon and click **New Project...**. The **New Project** window appears.




2. In the **Name** box, type a name for the new project.
3. Select **Enable team development** to enable multiple developers to work on the project simultaneously. Note that this setting cannot be change once the project has been created.
4. Select **Enable differential deployment** to take advantage of TX DWA's differential deployment feature that calculates what steps have changed and need to be deployed and selects only those steps for deployment. When differential deployment is disabled, all steps are deployed.
5. Select **Enable managed deployment** to have calculate dependencies and deploy the objects in the optimal order. There is no difference in performance or otherwise if you are only deploying one object. When managed deployment is disabled, you will have to make sure that objects are deployed in the correct order yourself.
6. Select **Use Integration Services (SSIS)** to use SQL Server Integration Services (SSIS) for data transfer. This SQL Server component Integration Services needs to be installed on the machine that deploys and executes the tables for this to work. TX DWA will use ADO.net if SSIS is not enabled.

- Select **Enable SSIS Logging** to enable SSIS logging.
- Clear **Use SSIS Folder** to not create a folder for the SSIS packages created. It is recommeded to keep this setting enabled to prevent accidental overwriting of the packages by other projects that contain tables with the same names.
- (Optional) Type the name of the SSIS folder in **Custom SSIS Root folder**.
- (Optional) Type a new password for encrypting SSIS packages in **SSIS package password** or leave it as the default. SSIS packages that contain SQL Server logins are encrypted using this password. Please note that this only applies when you are working with sources that use SQL Server authentication. Sources that use Windows authentication do not require encrypted SSIS packages.

7. Click on **Advanced Settings...** to access additional settings for the project. The **Advanced Settings** window appears.




8. Select **Postfix valid tables** to postfix the valid table instance with "_V".
9. Select **Show system control fields** to show system control fields such as DW_ID, DW_Batch, DW_SourceCode and DW_Timestamp.
10. Select **Allow Nulls** to allow null field values instead of moving the row to the error table when encountering null values. Under **Null check approach** click **Field Based** to use a field based check or click **Row Based** to use row based check. A field based

check will tell you exactly where the null value is, while a row based check will only tell you that the record has a a null value.

11. Under **Index Automation**, choose the default setting for automatic index generation. Select **Automatic** to automatically create indexes as needed, **Manual** to enable automatic index generation on an a per-table basis or **Disabled** to create indexes as needed during data cleansing as in earlier versions of TX DWA. This legacy behavior comes with a performance penalty, since indexes are always recreated even when an existing index could be used.
12. In the **Default hashing algorithm** list, click on the the hashing algorithm you want to use for hash fields that are set to "use project default" hashing algorithm if you want to change this from the default. You have the following options:
    1. **SHA-2 512, SQL Server 2016 +:** The safest hashing algorithm in terms of the probability that two different data sets would create identical hashes. The size of the hash is 64 bytes. It is about 40% slower than "SHA-1, SQL Server 2016+" and you should only use this algorithm when extreme safety is required. This algorithm requires SQL Server 2016.
    2. **SHA-1, SQL Server 2016 +:** The fastest hashing algorithm when the amount of data to be hashed is more than 8000 bytes. In those cases, it can be about 30% faster than "SHA-1, SQL Server 2005 +". Othwerwise, the performance for the two algorithms are the same. The size of the hash is 20 bytes. This algorithm requires SQL Server 2016.
    3. **SHA-1, SQL Server 2005 +:** The default algorithm in TX DWA. Slower than "SHA-1, SQL Server 2016 +" when the amount of data to be hashed is more than 8000 bytes. The size of the hash is 20 bytes and it is compatible with all SQL Server versions supported by TX DWA.
    4. **Plain text (debug):** Used for debugging, it will concatenate the fields into a string. This way, you can see what data goes into creating the hash. Because of a limitation in SQL Server, the string is limited to a lenght of 4000 characters.
    5. **Legacy binary:** Provided for compability with earlier versions of TX DWA and should not be used in new projects. It is not typesafe and is limited to a total of 4000 characters for all the fields the hash is calculated from.
    6. **Legacy plain text (debug):** Used for debugging legacy algorithms, it will concatenate the fields into a string with a lenght of up to 4000 characters.
13. Under **Primary key behavior**, choose how TX DWA should treat primary key violations. Click **Error** to move the offending row to the error table, click **Warning** to move the offending row to the warning table or click **None** to ignore the violation.
14. Under **Default type for new relations**, choose how TX DWA should treat foreign key violations. Click **Error** to move the offending row to the error table, click **Warning** to move the offending row to the warning table or click **Relation only** to ignore the violation.
15. Click **OK** to return to the previous window and **OK** again to create the project.

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

**Note:** The first time you start working with TX DWA, you must specify a project repository in which all projects are stored. For more information, see Setting Up a Project Repository.

## SAVING AND OPENING PROJECTS

Saving a project in the repository and opening a project from the repository works much the same as in other Windows programs.

### SAVING A PROJECT

TX DWA includes version control. On every save, a new version of the project is saved, meaning that you can always go back to an earlier version of your project.

To save the project, follow the steps below.

1. From the **File** menu, click **Save** or **Save As**.
2. If you clicked on **Save As**, or you are saving the project for the first time, type a name for the project and click **OK**.

**Note:** The project is automatically saved after a succesful deployment.

### OPENING A PROJECT

1. From the **File** menu, click **Open**.
2. (Optional) Click **Change Version**, click the version you want to open and click **OK** if you want to open a version of the project other than the latest version.
3. In the **Project** list, select the project that you want to open, 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.

### 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**.

# PROJECT REPOSITORIES

Your project is stored in a project repository on a SQL Server or Azure SQL Database. You can have as many or as few projects in a repository as you want. When you open TX DWA for the first time, you'll be prompted to set up your repository. All projects you create in the future will be saved in the specified repository, unless you change the repository.

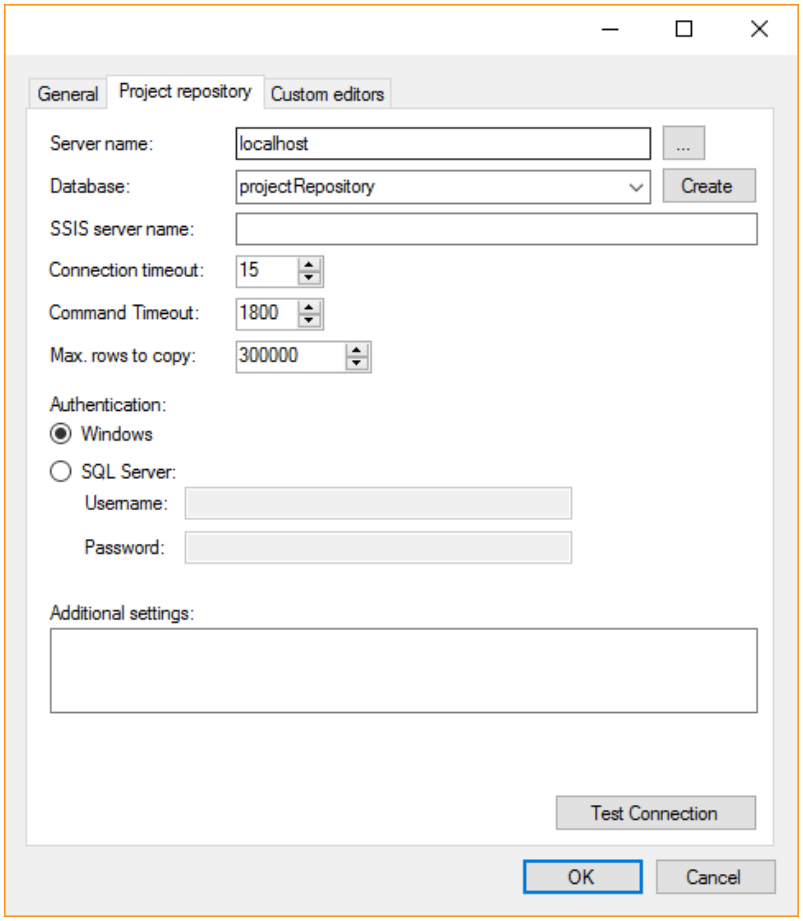
When you install a new version of TX DWA, you will be prompted to run an upgrade script that automatically updates the repository to ensure compatibility with the new software version.

**Note:** The repository settings are saved for the current user and installation of TX DWA. This means that if you open two instances of TX DWA and change the repository settings in one of them, you will change the repository settings for both.

## SETTING UP A PROJECT REPOSITORY

To change the repository settings, follow the steps below:

1. On the **Tools** menu, click **Options** an then click the **Project repository** tab in the window that appears.



2. In the **Server Name** field, enter the name of the database server on which you want to store the project. Click the ellipsis (...) to choose one of the available servers in you local Active Directory, if any.
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. (Optional) 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. (Optional)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 timeout.
6. (Optional) In the **Max. rows to copy** field, specify the batch size when using ADO.net transfer on the repository database.
7. Under **Authentication**, click **SQL Server** and enter username and password to use SQL Server authentication or leave the setting at **Windows** to use Windows authentication.
8. (Optional) Enter any addition connections settings in the **Additional Settings** box.
9. Click **Test Connection** to verify that the connection is working.

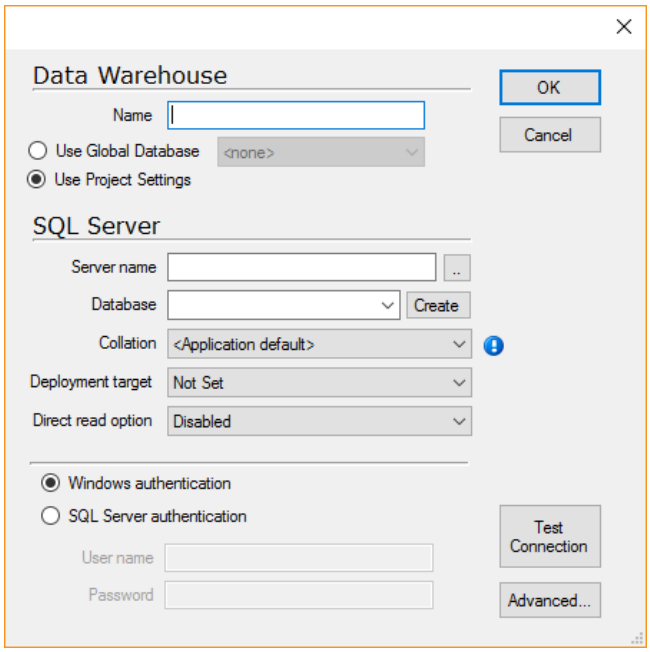
# DATA WAREHOUSES

A data warehouse in TX DWA is a database on either SQL Server or Azure SQL Database where your data is stored for queries and analysis. Most often, a TX DWA project consists of one data warehouse where you consolidate data from one of more staging databases and a number of data sources.

During execution of a project, TX DWA extracts data from the staging database and transfers it to the data warehouse. Initially, the data resides in what is known as raw table instances in the data warehouse. TX DWA applies data transformations and data cleansing rules and saves the resulting data in valid instances of the tables, ready for queries and analysis.

## ADDING A DATA WAREHOUSE

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

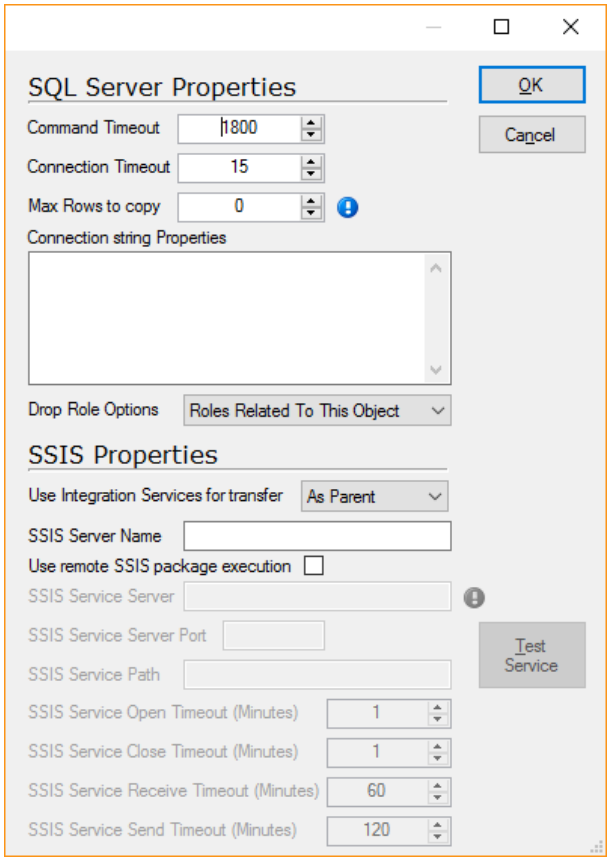



2. In the **Name** field, type a name for the data warehouse. The name cannot exceed 15 characters in length.
3. In the **Server Name** field, type the name of the server that you want to store the database on. If it is a named instance, type the server name and the instance name. Click the ellipsis (...) to choose one of the available servers in you local Active Directory, if any.
4. In the **Database** field, select an existing database, or type the name of a new database, and then click **Create**.
5. In the **Collation** list, click on the collation you want to use. You can choose a specific collation or one of the following:
    - **<Application Default>:** Uses the application default, which is **Latin1_General_CI_AS**.

- **<Server Default>:** Inherits the collation from the specified server.

It is best practice to use the same collation for the staging database, data warehouse database and any OLAP databases.

6. (Optional) In the **Direct read** list, you can enable direct read from the staging database(s) to the data warehouse database. With direct read enabled, data is transferred using a stored procedure containing a simple SELECT statement. This can, especially if TX DWA is not on the same machine as the SQL Server, give better performance than SSIS or ADO.net since transfers using these technologies happen via TimeXtender . For direct read to work, some prerequisites must be met: On SQL Server, the databases need to be on the same server. On Azure SQL Database, the staging and data warehouse databases need to be in the same database. You have the following options for direct read:
   - **Disabled**
   - **Matching Server:** Direct read is used if the data warehouse and staging database server names match.
   - **Matching Server and Database:** Direct read is used if the data warehouse and staging database server names and database names match.
7. Specify the authentication mode. The default setting is **Windows authentication**. Click **SQL Server authentication** and enter username and password to use SQL Server authentication.
8. Click **Test Connection** to verify that the connection is working.
9. Click **Advanced...** to access the advanced settings for the data warehouse. These are all optional.



10. In the **Command Timeout** field, enter the number of seconds to wait before terminating a command.
11. In the **Connection Timeout** field, enter the number of seconds to wait before terminating the attempt to connect to the server. Set it to 0 to wait indefinitely.
12. If you want to add additional connection strings, enter them in the **Connection String Properties** box.
13. See Adding a Database Role for an explanation of the **Drop role options** setting.
14. In the **Use Integration Services for transfer list** click on **Yes** to enable SSIS data transfer, **No** to disable it or leave it at **As Parent** to respect the project setting.
15. If your SSIS Server is installed under a different name than the database, enter the name in the **SSIS Server Name** box.
16. Select **Use Remote SSIS package execution** to enable the execution of SSIS packages on a remote server and enter the details for the remote server below. Note that you will need to install the Remote SSIS Execution service on the remote server - see Installing the Remote SSIS Execution Feature

## CLEANING UP THE DATABASE

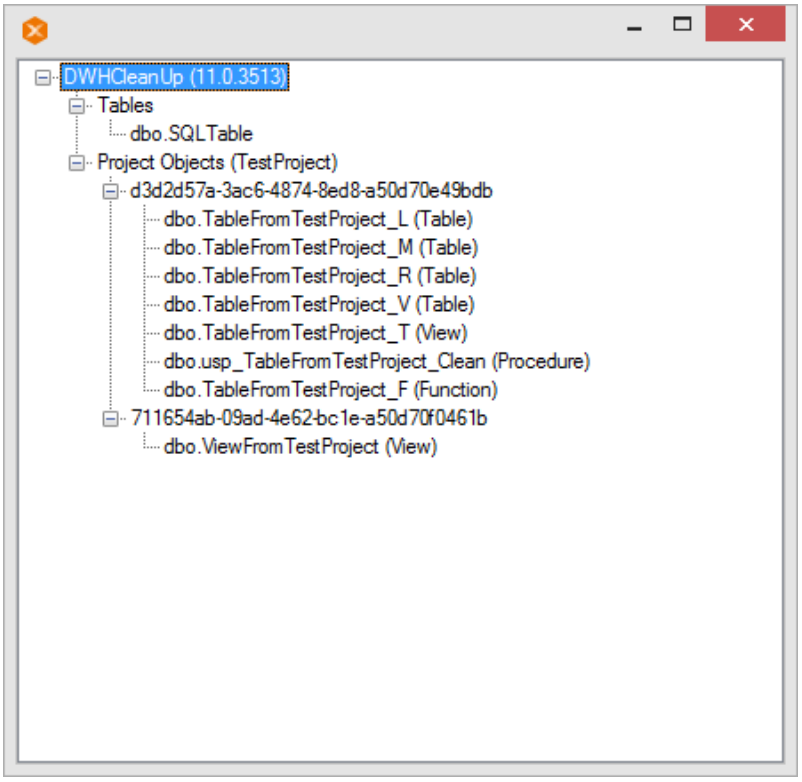
To prevent accidental data loss, deleting a table in TX DWA does not delete the physical table on the SQL Server. The downside is that tables deleted in TX DWA still takes up space in the database. The SQL database cleanup feature enables you to identify tables left behind by TX DWA and delete - drop - them to free up space. Note that database schemas are not deleted from the database. You will need to drop those manually in SQL Server after deleting them in TX DWA.

**Warning**: Tables deleted with the SQL Database Cleanup feature cannot be recovered. Take care when you use the feature.

### IDENTIFYING AND DELETING TABLES WITH SQL DATABASE CLEANUP

To use the database cleanup feature, follow the steps below.

1. In the project tree, right click a data warehouse, click **Advanced** and click **SQL Database Cleanup Wizard**. TX DWA will read the objects from the database and open the **SQL Database Cleanup** window.




2. In the **SQL Database Cleanup** window, the content of the database is listed. Expand the root node, **[database name] ([SQL Server Version])**, to display the categories of content found on the second level. Depending on the content, the following categories are displayed:
    - **Tables**: Tables in the database unrelated to any TX DWA project.
    - **Views**: Views in the database unrelated to any project.
    - **Procedures**: Procedures in the database unrelated to any project.
    - **Functions**: Functions in the database unrelated to any project.
    - **Deleted Project Object**: Objects deleted from the project, but not from the database. These should be safe to drop.
    - **Project Objects (<Unresolved Project: [project id]>)**: Objects in the database related to an unknown project, i.e. a project not in the current repository.
    - **Project Objects ([name of project])**: Objects in the database related to a project in the current repository.
3. Expand Project Objects to display a list of object ids in the project. If you expand an object id, the tables, views etc. related to the object are listed. For example, if the object is a table, the valid, raw and other instances of the table are listed.
4. (Optional) Right click a table, view, procedure or function and click **Script** to display the SQL script behind the object.
5. Right click a table, view, procedure or table and click **Drop** to drop the object from the database. Click Yes, when TX DWA asks you to confirm the drop. A message will tell you if the action succeeded or failed.

6. Right click an object id or a Project Objects item and click **Drop** to drop the object and all objects on the levels below. A window will open with a list of the objects that will be dropped. Clear the selection for any tables you want to keep and click **Drop**.
   Note: TX DWA will automatically clear the selection for any incrementally loaded tables to prevent accidental data loss. TX DWA will ask you to confirm if you want to drop an incrementally loaded table.
7. When you have dropped the all the objects you want to delete from the database, close the window.

# BUSINESS UNITS

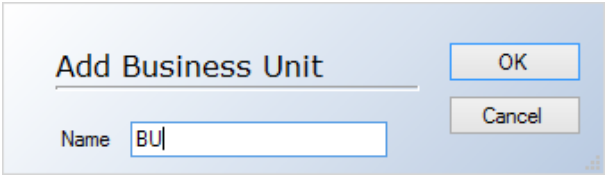
In TX DWA, a business unit is any part of your organization that you want to treat as a separate entity in your project. For example, you may want to treat a company headquarters and each of its subsidiaries as separate business units.

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

## ADDING A BUSINESS UNIT

To add a new business unit, follow the steps below:

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

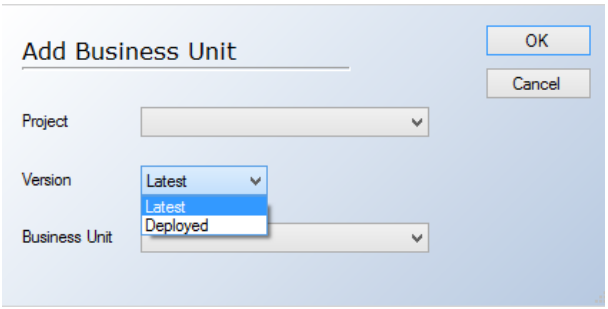



2. Type a name for the business unit and click **OK**.
3. When you create a business unit, you always have to specify a staging database. See Setting Up a Staging Database.

## ADDING AN EXTERNAL BUSINESS UNIT

You can add business units from other projects to your project. This enables reuse of business units across different projects. To add an external business unit, follow the steps below:

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




2. In the **Project** list, select the project that contains the business unit you want to add.
3. In the **Version** list, click on the 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, click on the business unit you want to add, and then click **OK**.
5. Click either **By ID** to synchronize fields by ID or **By Name** to synchronize fields by name. The business unit is add and displayed as a separate entity in the project tree.

If you make changes to the external business unit in its "own" project, you can synchronize the changes to your project.

To synchronize an external business unit

- Right click the external business unit click **Synchronize** and click either **By ID** to synchronize fields by ID or **By Name** to synchronize fields by name.

### ENABLING SIMPLE MODE FOR A BUSINESS UNIT

Simple mode is a setting on tables on business units aimed at maximizing performance when you need to copy large amounts of data into a staging database to create an exact copy. See Simple Mode for more information.

Simple mode can be enabled on the business unit level for all data sourcea and tables on the business unit. It can be overridden on the data source level or on individual tables.

To enable simple mode for a business unit

- Right click on the business unit, click **Business Unit Settings** and select **Enable Simple Mode**.

# STAGING DATABASES

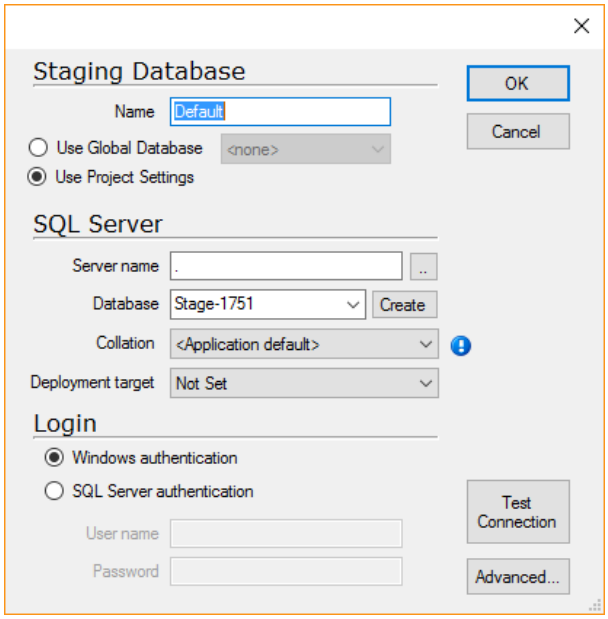
A business unit always contains a staging database, which can be stored on SQL Server or on Azure SQL Database. The staging database stores the selected data from the data sources. Additionally, many of the validation and transformation processes take place in the staging database. This ensures that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

The difference between a staging database and a data warehouse is minimal. You can add custom tables, views, scripts, table relationships, security to a staging database just as if it was a data warehouse database.

## ADDING A STAGING DATABASE

Staging databases are created along with business units.

1. Add a business unit. The **Add Staging Database** window is displayed to let specify a staging database.




2. In the **Name** field, type a name for the staging database. The name cannot exceed 15 characters in length.
3. In the **Server Name** field, type the name of the database server. Click the ellipsis (...) to choose one of the available servers in you local Active Directory, if any.
4. In the Database list, select the preferred database from the drop-down list. If you want to create a new database, then type a name for the new database, and then click **Create**.
5. Specify the authentication mode. The default setting is **Windows authentication**. Click **SQL Server authentication** and fill in the **Username** and **Password** boxes to use SQL Server authentication.
6. Click **Test Connection** to verify that the connection is working.

7. (Optional) Click **Advanced...** to open the advanced settings window. These settings are the same as for the data warehouse. See Adding a Data Warehouse for more information.

# TEAM DEVELOPMENT

TX DWA supports multiple developers working on the same project at the same time. Version notes enable developers to share details about their changes to the project and work items allows developers to see which objects other team members are working on to prevent them from modifying the same objects simultaneously.

When collaborating on a project, developers should avoid working on the same object on the same time. If this happens, the outcome depends on the type of change made:

- If multiple team members add or modify different sub-objects (such as fields) within the same object (such as a table), then TX DWA will save these changes and display them the next time that TX DWA is opened. For example, if one team member adds two fields to a table, and another team member adds another two fields to the same table, all four fields will be visible to all team members the next time they close and re-open the project.
- If multiple team members modify the same sub-object at the same time (for instance renaming a field), the changes deployed and executed last will take precedence, and the project will reflect those changes.

The team development feature of TX DWA depends on the developers to take care when working together as the feature itself does not prevent developers from making conflicting changes.

Follow the rules below to ensure that the project stays consistent:

1. Never work on the same object - table, dimension or cube - as someone else.
    - Always create work items before making any changes to an object.
    - If anyone else has the item marked for their use, wait for them to release the work item.
    - Once you are done working on an object, save the project and release the work item.
    - Keep the work items window open when working.
2. Do not rename tables, cubes or dimensions. The only exceptions are if you just created the table and have not saved yet or if nobody else has the project open. You can see who has a project open in the work items dialog.
3. Always save (CTRL + S) and reload (CTRL + F5) before you start a new development task. This ensures that you have the latest version of all objects before you start taking over somebody else's work.
4. Immediately before creating a new table, always save and reload.

### ENABLING THE TEAM DEVELOPMENT ENVIRONMENT

You need to enable Team Development for any project you want to use the feature in.

1. On the **Tools** menu, click **Repository Administration**.
2. On the **Projects** tab, right-click the relevant project and click **Enable Team Development.**
3. Close the window. Multiple users can now access the project concurrently.

## WORK ITEMS

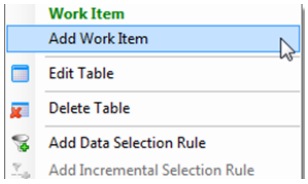
Work items allow the team to know what its users are working on. This will give a visual indication as to what areas of the project are currently under development. Work items are meant to be created immediately prior to starting work on a set of objects and can be either manually deleted or removed during deployment of the object. 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

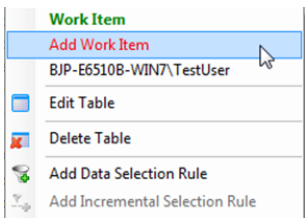
### ADDING WORK ITEMS

To add a work item, follow the steps below.

1. Right-click the relevant object and click **Add Work Item**.




   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.



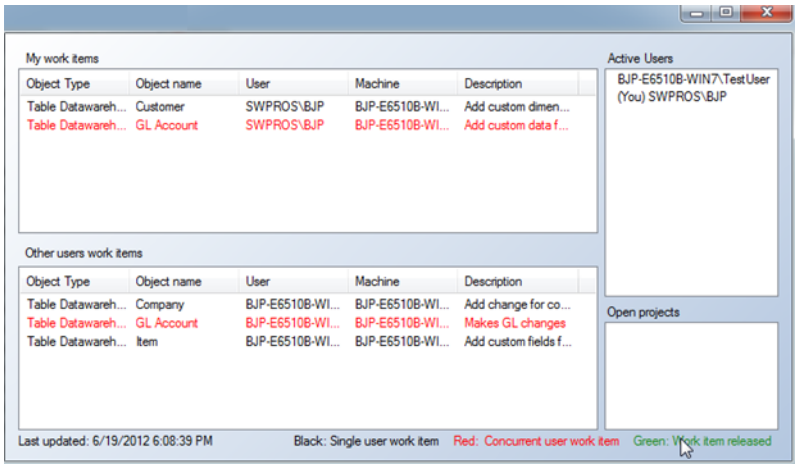

   Otherwise, the Create Work Item window appears.

2. In **Description**type a short description of the work item, e.g. what task the work item is a part of, and click **Create Work Item**.
3. Click **Create Work Item.**

### VIEWING EXISTING WORK ITEMS

To view existing work items, follow the steps below.

1. On the **Tools** menu, click **Work Items**.
2. The **Work Item** window opens. It contains a list of your own work items, **My work items**, and a list of **Other users work items**.




Each list displays the following information:

| Column | Description |
|---|---|
| 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 |

Work items can have three colors depending on their status:

- Black: A work item that is only flagged by a single user.
- Red: Work items that share the same Object Name for different users. This tells you 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.
- Green: Work items that have been completed by another user who has saved or deployed and marked the work item as completed.

In the **Active Users** pane, all users currently accessing the project are shown.

### EDITING A WORK ITEM

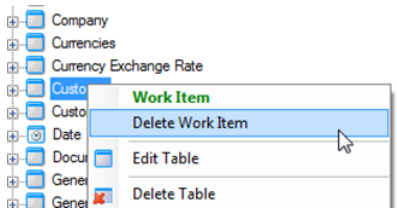
Work items can only be edited by the user that created the work item. To edit a work item, follow the steps below:

1. On the **Tools** menu, click **Work Items**.
2. In **My Work Items**, right-click the work item to edit and select **Edit Work Item**.
3. Update the work item descriptiop and click**Update Work Item** to save the changes.

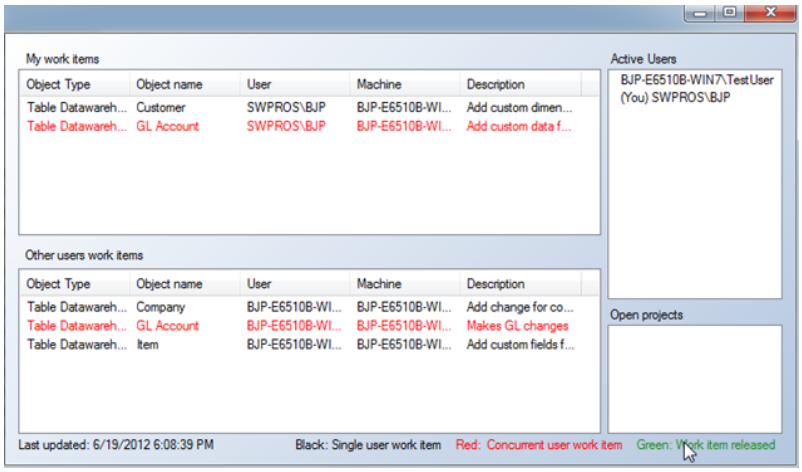
### DELETING WORK ITEMS

Work items can only be edited by the user that created the Work item. You can delete a work item in three different ways:

- Right click the relevant object and click **Delete Work Item**




- On the **Tools** menu, click **Repository Administration**. On the **Work items** tab, click a work item an click **Delete**.
- On the **Tools** menu, click **Work Items**.




In **My work items**, right-click the relevant work item to edit and click **Delete Work Item**. TX DWA will ask you to confirm the deletion. Click **Yes**.

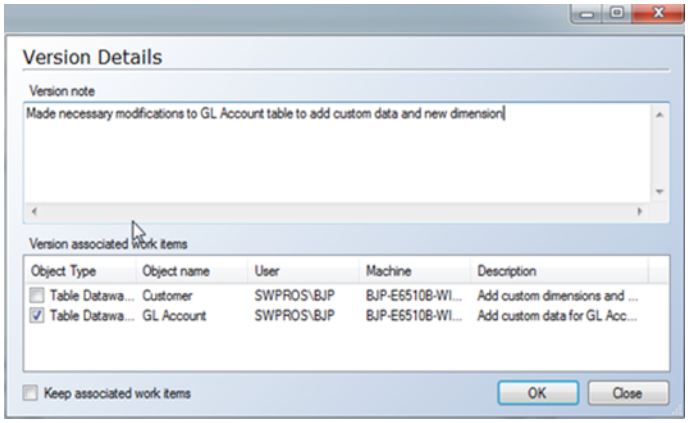
Work items can also be removed when adding version notes, which is discussed below.

## VERSION NOTES

Version notes enables you to add comments about changes that you have made to a project. These comments can then be viewed in the future to reference changes that were made.

## ADDING VERSION NOTES

1. To ensure version notes are enabled, click **Options** on the **Tools** menu. The **Options** window appears.
2. Under **Prompt for version details**, make sure one of the following options is selected:
   - **Everytime the project is saved**: You will be prompted to enter a version note when you deploy an object or when you click the **Save** button.
   - **Everytime the project is saved during deployment**: You will only be prompted to enter a version note when you deploy an object.
3. When you deploy an object in the project, the **Version Details** window will appear just before the deployment begins.

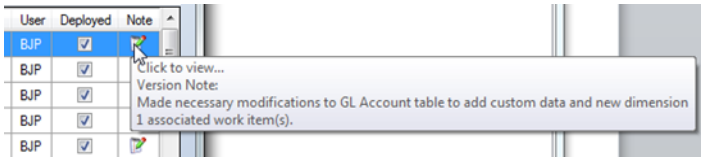



You can type in any details you want to include in the version note. If relevant, you can also select one of your existing work items to associate with the version note. Selecting a work item will also remove the work item from the list of existing work items. If you want to associate a work item with the version note, but not remove the work item, select **Keep Associated Work Items**. If you do not want to save a version note for this deployment, you can click **Close**.This will close the window without saving a version note.

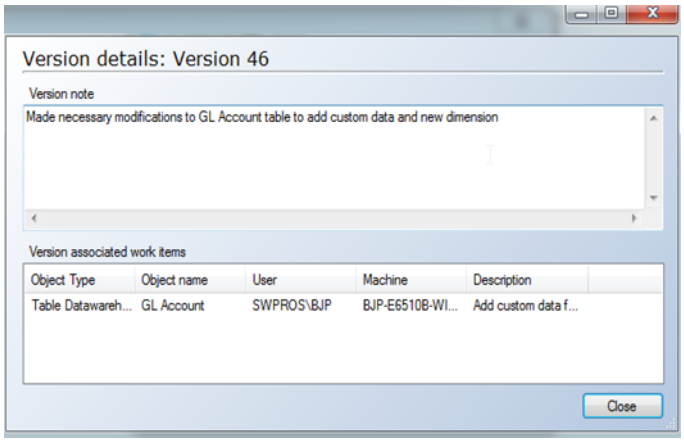
## VIEWING VERSION NOTES

Viewing version notes can be useful when you would like to look back and see when particular changes were made. Follow the steps below to see the list of project versions and associated version notes.

1. On the **Tools** menu, click **Repository Administration**. The Repository Administration window appears.
2. Right click the relevant project and click **Show Project Versions**. The Project Versions window appears.
3. If a notepad icon is displayed in the **Note** column, a version note is available for the version. Hover over the notepad icon to see a quick description of the note.




4. Click the notepad icon to see the full version note as well as any work items associated with the version note.



# CONNECTING TO DATA SOURCES

Data sources contain the data that you want to load into your data warehouse and use for analysis. The current version of TX DWA connects to the following data sources:

- Microsoft SQL Server
- IBM DB2
- IBM Informix
- Oracle
- Oracle MySQL
- ODBC compliant data sources
- Microsoft Excel files
- Plain text files
- AnySource OLE DB/ADO.NET
- Custom Data Sources

In this chapter, you will learn how to connect to the different data sources.

## DATA EXTRACTION

Once you have added and configured a data source, the next step is to select the tables and fields you want to copy into the staging database. For more information, see Moving Data into the Staging Database.

Selection templates offer another way of selecting what data to copy into the staging database. For more information, see Selection Templates.

This section covers the settings for data extraction associated with data sources in general.

### INTEGRATION SERVICES

On most data sources, you will find the setting **Use Integration Services For Transfer** that determines how TX DWA transfers data between a source and a destination table. You have the following options:

- **As Parent**: The setting will be taken from the project setting 'Use Integration Services'.
- **Yes**: A SQL Server Integration Services (SSIS) package is used. This requires that the SQL Server component Integration Services is installed on the machine that deploys and executes the tables.
- **No**: ADO .NET is used. This does not require any SQL components except SQL Server Management Objects.

Using SSIS packages for transferring data is generally considered to be faster for transferring large amounts of data. When transferring data between tables with fewer records, using ADO

.NET can sometimes be faster, as it takes time to load the SSIS packages from the SQL Server where they are stored before data transfer can begin.

It takes significantly longer to deploy SSIS packages than it does to deploy ADO .NET transfer. This setting can have a great impact on the overall deployment time of a project.

SSIS and ADO .NET use different technologies to transfer data. This means that if you get erroneous data through SSIS, you can sometimes get correct data using ADO .NET.

### SIMPLE MODE

Simple mode is a setting on tables on business units aimed at maximizing performance when you need to copy large amounts of data into a staging database to create an exact copy. When a table is in simple mode, everything but the most basic functionality is disabled:

- Tables in simple mode do not support field transformations, field validations, conditional lookup fields.
- Tables in simple mode only have the valid instance of the table unless incremental load is enabled.

Per default, a data source inherits the simple model setting from the business unit, but you can override the setting on the data source and on individual tables on the data source.

To enable simple mode for a data source

- Right click the data source, click **Data Source Settings** and click **Enable** under **Simple mode**.

### GUARDING A DATA SOURCE

You can guard a data source in TX DWA, which prevents tables that get their data from the data source from being deployed, executed or both. In general, the guard feature is useful for tables that contain data that never changes, e.g. from a legacy system. You can also guard a single table. See Guarding a Table for more information.

To guard a data source

- Right click the data source and click **Data Source Settings**. Select **Guard on deployment** to prevent TX DWA from deploying the table and/or **Guard on execution** to prevent TX DWA from executing the table.

### LIMITING CONCURRENT TRANSFERS ON DATA SOURCE

You can put a limit on the number of concurrent transfers from a specific data source. Some data sources can only handle a certain number of transfers before adding more transfers will actually slow down the overall performance of the transfer rather than speed it up.

To limit the amount of concurrent transfers from a data source

- Right click the data source, click **Data Source Settings** and enter the allowed number in the **Max concurrent transfers** box. "0" equals unlimited.

### ALLOWING A DATA SOURCE TO FAIL

If you have some source systems in your solution that are less than critical for your reporting, you can configure your solution so that the entire execution does not stop just because TX DWA cannot reach these noncritical systems. Instead, you can choose to keep the newest data from the system in question until fresh data can be fetched.

Follow the steps below to allow a data source to fail.

1. Right click the data source and click **Data Source Settings**.
2. Under **When transfer fails**, click the option you want to use if the transfer should fail. You have the following options:
    - **Fail and stop the execution**: Stops execution and reports the execution as failed (default setting).
    - **Continue execution without data**: Continues the execution, pretending that the source contains no data.
    - **Continue with existing data**: Continues execution, retaining the existing data from the source.

**Note**: If you use additional connectors, you need to configure this setting on the additional data source as well as the template data source.

If a data source, that you have allowed to fail, fails during execution, the following execution message will be recorded: "Execution was successful, but one or more data sources failed".

## DATA TYPE OVERRIDES

You can override the data type of fields on the data source level. While you can choose the data type for each individual field in the data warehouse it is easier to do on the data source level if you use the same field more than once in the data warehouse.

Data type overides are implemented as rules that can be added and ordered to create a hierachy. For each field, TX DWA goes throught the rules and applies the first rule that matches if there is any that do.

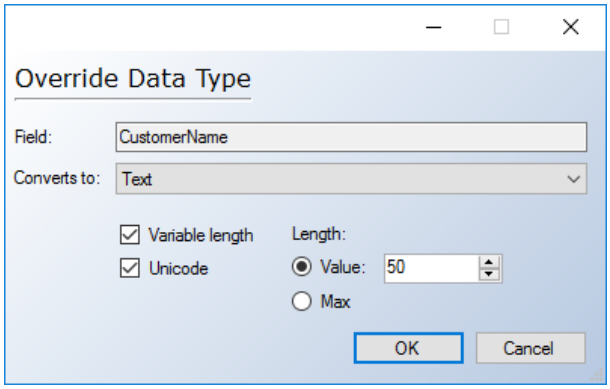
Under the data source in the project tree, a field's original data type is displayed, while the overridden data type is displayed anywhere else.

### ADDING A DATA TYPE OVERRIDE TO A SINGLE FIELD

To add a data type override to a single field on a data source, follow the steps below.

1. Right click the field and click **Override Data Type**. In the **Converts to** list, click on the data type you want to use and then adjust any settings for the data type you have

selected.



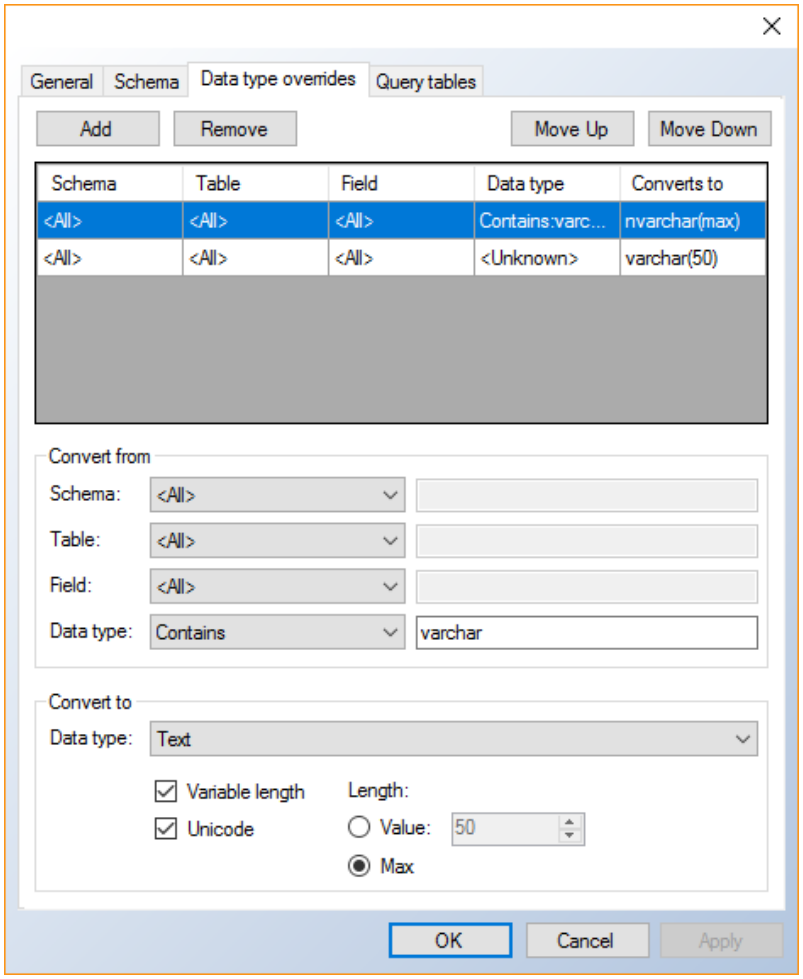

2. Click **OK**.
3. Right click on the data source and click **Synchronize Data Source** to apply the overrides.

**Note:** This creates a rule that matches the schema, table and field name of the field. If one of these names change, the rule will no longer match.

### ADDING A DATA TYPE OVERRIDE

If you want to add a data type override that matches more than a single field, follow the steps below.

1. Right click a data source, click **Data Source Settings** and click the **Data type override** tab.




2. Click **Add**. A new rule is added to the list.
3. Under **Convert from** group, choose what criteria a field should fulfil to match the rule. You can add criteria on schema, table and field name as well as the data type. In the list of operators for the data type, you can click on "<Unknown>" to match data types that are not compatible with SQL Server.
4. Under **Convert to**, click on the data type you want to convert the matching fields to in the **Data type** list. Adjust any settings for the data type you have selected.
5. Click **OK**.
6. Right click on the data source and click **Synchronize Data Source** to apply the overrides.

### ORDERING DATA TYPE OVERRIDES

Data type overrides are processed for each individual table. The first override from the top that matches the field is applied and any other matches are ignored. Follow the steps below to order the overides.

1. Right click a data source, click **Data Source Settings** and click the **Data type override** tab.
2. Click on the override you want to reorder in the list and click on **Move Up** or **Move Down** to move the override up or down respectively.

**Note:** The default rule that matches any unknown data types cannot be reordered. However, you can edit the data type it converts to. The purpose of this rule is to ensure that all fields end up with a data type SQL Server can handle.

## QUERY TABLES

When you connect to a data source in TX DWA, you can simply use the read objects feature to list the content of the source and pick the tables and fields you want to use in your solution.

However, some AnySource providers allows you to connect to a data source, but cannot list the contents of it. In other cases, it is simply useful to be able to create a table, that does not already exist on a data source, from a query. To enable you to use these data sources in TX DWA, we have included the Query Tables feature. While the SQL behind ordinary tables is created by TX DWA, you write the query that brings query tables to life.

The following data sources and adapters support the query tables feature:

- AnySource adapter
- Dynamics AX adapter
- Dynamics NAV adapter
- Oracle data source
- SQL data source

### ADDING QUERY TABLES

To add a query table, follow the steps below.

1. Connect to a data source using one of the supported data sources or adapters.
2. Right click the data source, click **Data Source Settings** and the click the **Query Table**tab.
3. Click **Add**. A new table is added to the list.
4. In the **Name** box, type a name for the table.
5. (Optional) In the **Schema** box, type a schema to use.
6. In **Query**, enter the query you want to use for creating the table. The query should contain a SELECT statement and follow the syntax required by the source.
7. Select **Subquery needed** if you are using an alias in your query. Otherwise, selection rules will fail.
8. Repeat step 3-6 to add the tables you need and click **OK**.
9. Right click the data source and click **Read Objects from Data Source**. The tables are listed in the panel in the right-hand side of TX DWA and can be included in the project like any other table.

### HANDLING ACCOUNTS IN DYNAMICS NAV

When you create query tables for Dynamics NAV, you will have to consider how you handle accounts.

To get data from one account, remember the account in the FROM part of your state-ment:

```
SELECT * FROM [dbo].[MyCompany$MyTable]
```

To get data from multiple accounts, in the same way the Dynamics NAV Adapter does it, you can use placeholders:

```
SELECT * {0} FROM [dbo].[{1}$MyTable]
```

TX DWA will replace the digits in curly brackets during execution to create the following state-ment for each account:

```
SELECT
* ,CAST('MyCompany' AS nvarchar(30)) AS (DW_Account)
FROM
[dbo].[MyCompany$MyTable]
```

## TEMPLATE DATA SOURCES

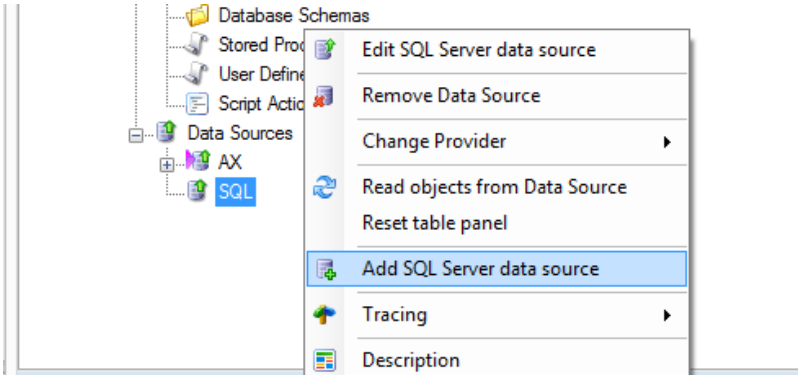
If you have more than one data source with an identical data structure, you can connect all of them together and use them as if they were one and the same. This feature is known as tem-plate data sources.

### ADDING AN ADDITIONAL CONNECTION

Once you have added the first of the indentical data sources as usual, you can add the addi-tional data sources to the first data source. Since all of the data sources will have a similar data structure, it does not matter which one is added first.

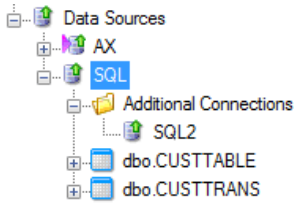
To add an additional connection, follow the steps below.

1. Right-click a data source and click **Add <data source> Data Source.**



An **Add <data source> Data Source** window appears.

2. Configure this second data source with the necessary 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 data source, these changes will seemlessly be applied to all of the data sources that under **Additional Connections**.

The "DW_SourceCode" field on tables on the data source contains the name of the source from which the row was copied. If you have a setup like the one displayed above with two sources, "SQL" and "SQL2", a table on this source would look something like the list below.

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... | Bimamwood | 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 |

## SELECTION TEMPLATES

Once you have selected all the tables you need from a data source, added primary keys and set up incremental load, you can export all this information to a selection template. The template can then be applied to another data source in the same or another project. The data sources do not need to be identical.

Selection templates are stored in XML format.

### EXPORTING A SELECTION TEMPLATE

To export a selection template

- Right click a data source, click **Selection Template** and click **Export**. In the window that appears, chose a file name and folder and click **Save**.

The tables and fields selected, the primary keys chosen and the incremental load setup is saved in the exported file.

### IMPORTING A SELECTION TEMPLATE

To import and apply a selection template to a data source, follow the steps below.

1. Right click the data source, click Selection Template and click **Import**. The **Import Selection Template** window opens.
2. Click **Browse** to find and select the template to import.
3. Under **Apply the following details from the template**, remove the checkmark from any parts of the template you do not want to apply. If you remove the checkmark from **Tables and fields**, but keep **Primary keys** and/or **Incremental Load Setup** selected, these settings will be applied to the tables and fields already selected from the data source.
4. Click **OK**.

## CHANGING DATA SOURCE PROVIDERS

If you have moved your data to a new database, e.g. from Oracle Database to Microsoft SQL Server, you do not have to add a new data source to reflect this. Instead, you can change the type of your existing data source in TX DWA with the change provider feature.

To change the provider for at data source

- Right click the data source, click on **Change Provider** and click on the database you want to change provider to. A connection settings window appears. Refer to the section on the relevant data source in this user guide to learn more about the settings.

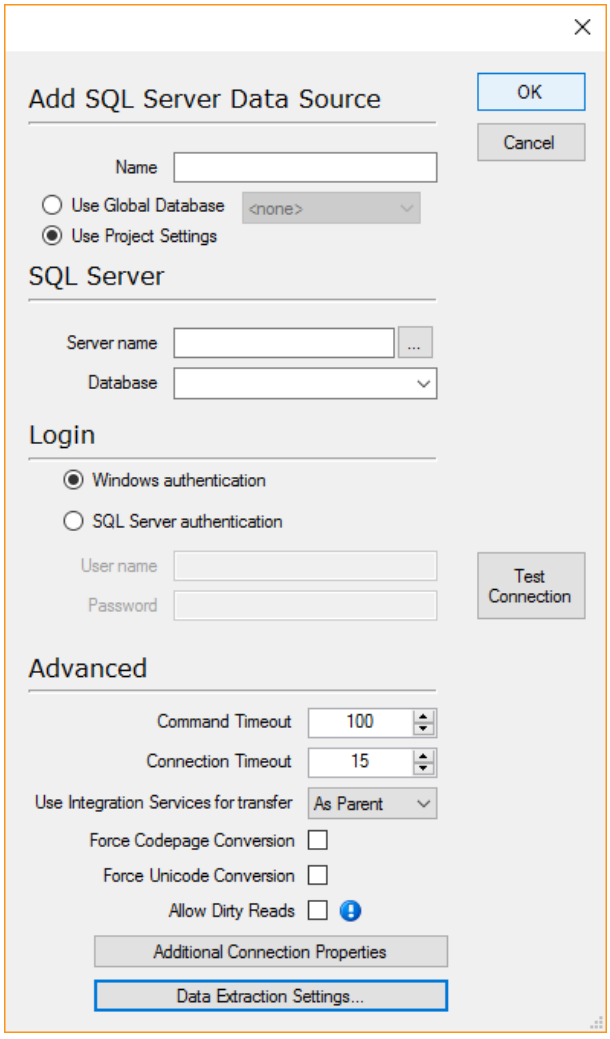
# SQL SERVER DATA SOURCE

TX DWA supports all versions of Microsoft SQL Server as well as Azure SQL Database as a data source.

## ADDING A SQL SERVER DATA SOURCE

To add a new SQL Server data source, follow the steps below:

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**.




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. Click the ellipsis (...) to choose one of the available servers in you local Active Directory, if any.
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**, enter the username and password.

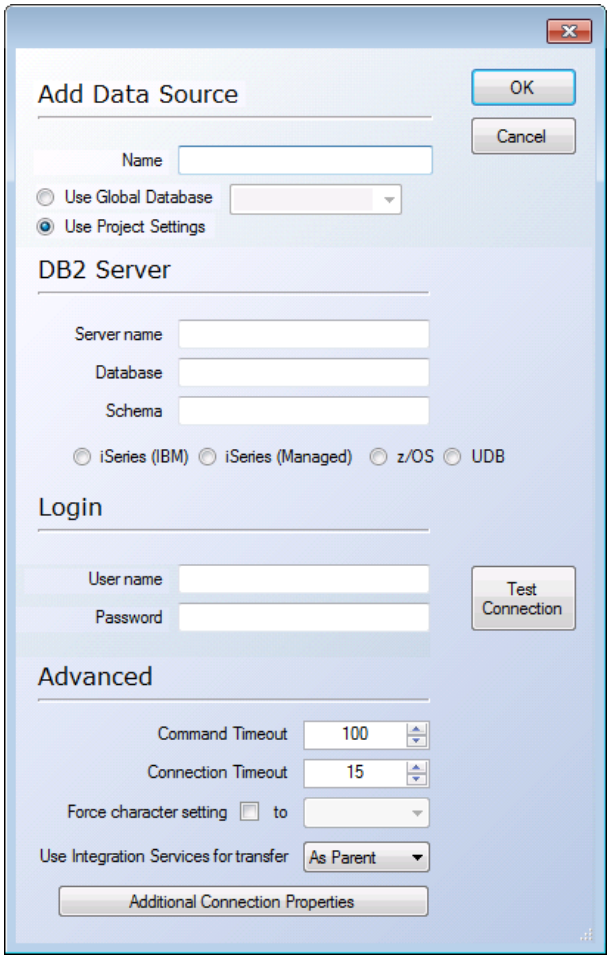
7. Click **Test Connection** to verify that the connection is working, and then click OK. The data source is added to the Data Sources folder in the project tree.
8. In the **Command Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the database.
9. In the **Connection Timeout** field, specify the number of seconds to wait before terminating the attempt to connect to the server.
10. In the **Use Integration Services for transfer** list, click on **Yes** or **No** to explicitly enable or disable SSIS transfer for the source.
11. If you want to add additional connection strings, click **Additional Connection Properties** . In the **Connection String Properties** window, type the preferred connection strings, and then click **OK**.
12. Click **Data Extraction Settings** to open a window with options to limit the objects brough into TimeXtender before the data selection stage. This can be very useful if you need to connect to a very large data source with a lot of tables and views.
13. Click **Test Connection** to test if the connection settings you have specified work and then click **OK** to add the data source.

# DB2 DATA SOURCE

TX DWA can extract data from IBM DB2 databases.

## ADDING A DB2 DATA SOURCE

1. In the project tree, expand **Business Units**, expand the relevant business unit, right click **Data Sources**, click **Data Sources** and click **Add DB2 Data Source**. The **Add DB2 Data Source** window opens.



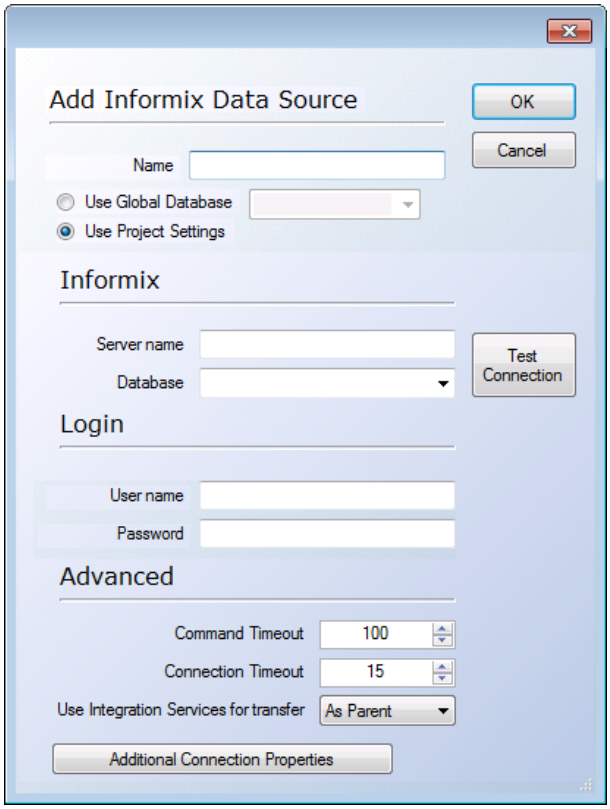

2. Enter the connection information and click **OK**.

# INFORMIX DATA SOURCE

TX DWA can extract data from IBM Informix databases.

## ADDING A INFORMIX DATA SOURCE

1. In the project tree, expand **Business Units**, expand the relevant business unit, right click **Data Sources**, click **Data Sources** and click **Add Informix Data Source**. The **Add Informix Data Source** window opens.



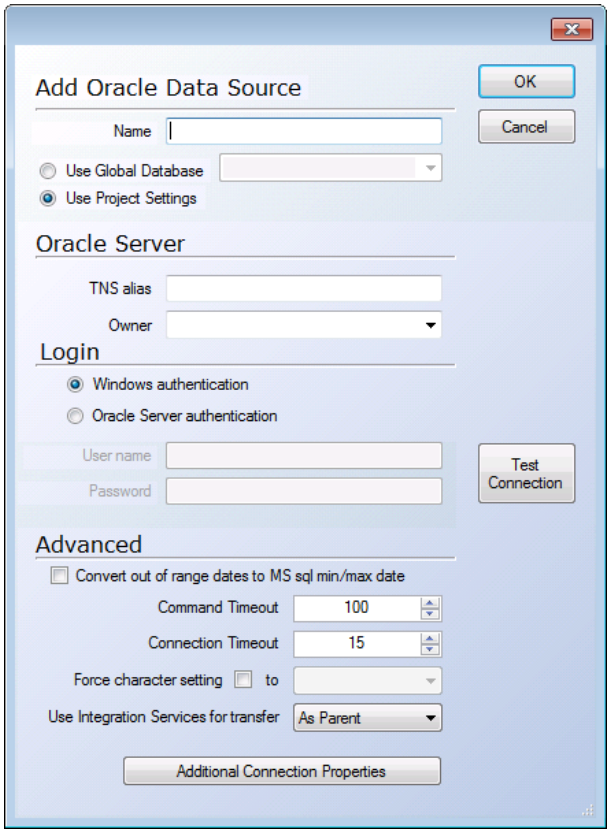

2. Enter the connection information and click **OK**.

# ORACLE DATA SOURCE

TX DWA can extract data from Oracle databases.

## ADDING A ORACLE DATA SOURCE

1. In the project tree, expand **Business Units**, expand the relevant business unit, right click **Data Sources**, click **Data Sources** and click **Add Oracle Data Source**. The **Add Oracle Data Source** window opens.




2. Type a **Name** used to identify the data source in TX DWA.
3. Type **TNS alias**, type the alias that identifies the database.
4. In the **Owner** list, click the owner of the database.
5. Under **Login**, click **Oracle Server authentication** if you want to use this login method and enter **User name** and **password**.
6. Under **Advanced**, 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.
7. (Optional) Enter the number of seconds to wait before terminating the attempt to connect to the database in **Command Timeout**.
8. (Optional) Enter the number of seconds to wait before terminating the attempt to connect to the server in **Connection Timeout**.
9. (Optional) Select **Force Character Setting** if you want to set the character encoding to either Unicode or Non-Unicode and click the preferred character encoding in the list.

**Note:** Forcing character encoding may affect performance.

10. (Optional) In the **Use Integration Services for transfer** list, you can change the default, **As Parent**, by clicking either **Yes** or **No**.
11. (Optional) Add additional connection strings, click **Additional Connection Properties**. In the **Connection String Properties** window, type the connection strings and click **OK**.

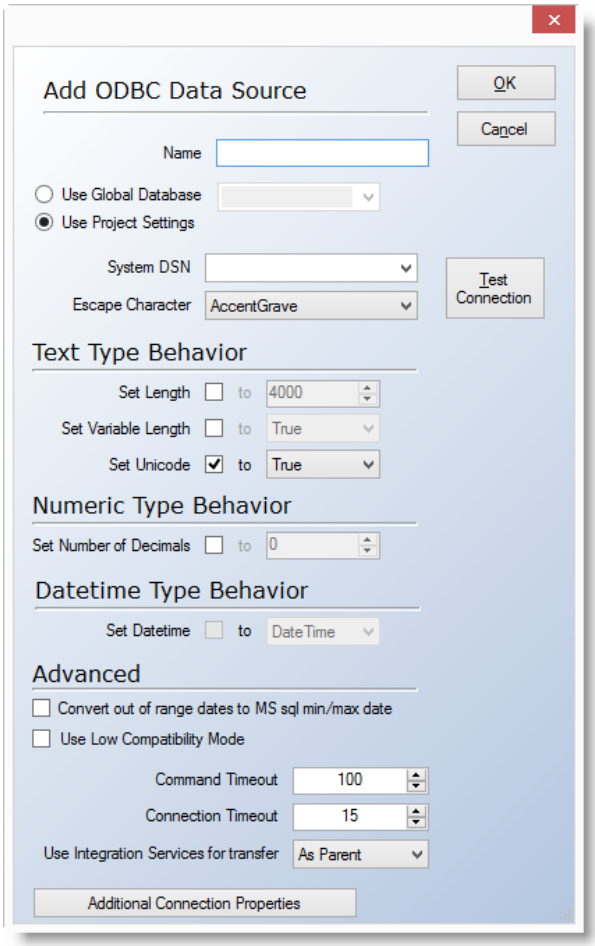
# MYSQL DATA SOURCE

TX DWA supports MySQL data sources through ODBC.

## ADDING A MYSQL DATA SOURCE

To add a MySQL data source, follow the steps below:

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.




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:

| Option | Definition |
|---|---|

| | |
|---|---|
| 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**.

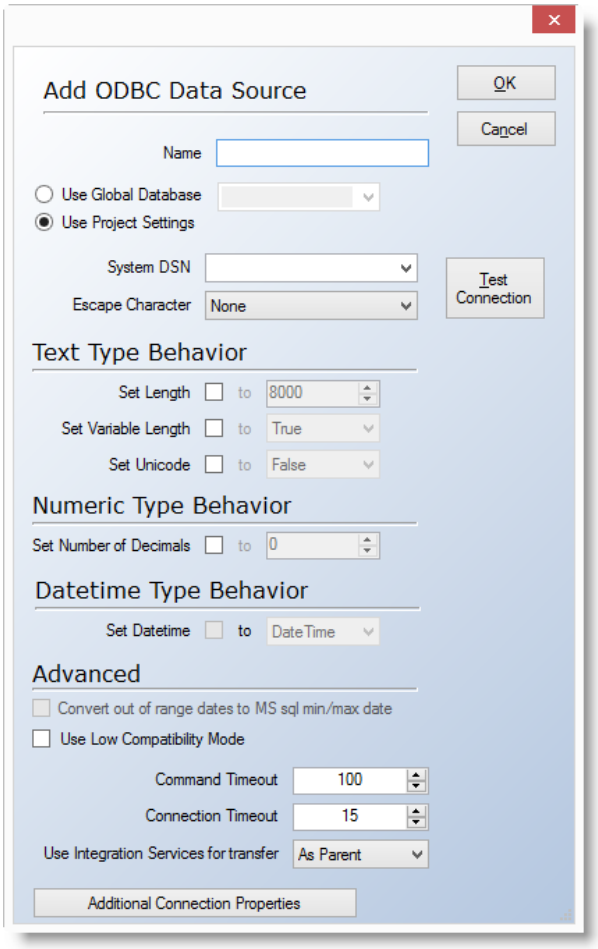
# ODBC DATA SOURCES

TX DWA supports ODBC for retrieving data from a wide range of databases.

## ADDING AN ODBC DATA SOURCE

To add a new ODBC data source, follow the steps below:

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 database.
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:

| Option | Definition |
|---|---|

| | |
|---|---|
| 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**.

# EXCEL DATA SOURCE

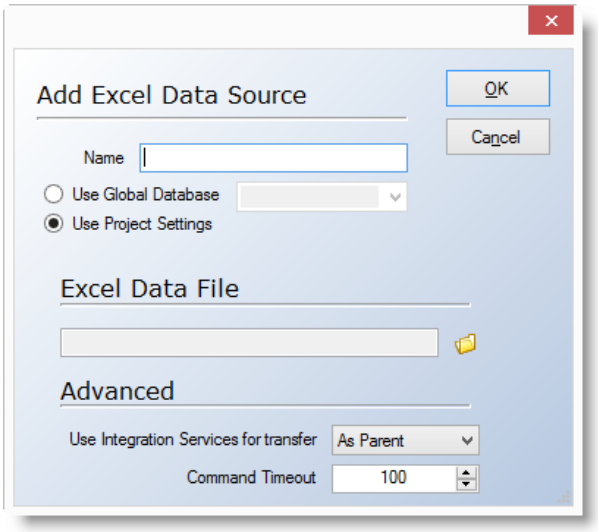
TX supports Microsoft Excel spreadsheets as a data source, both the older .xls format and the newer .xlsx format from Excel 2007 and beyond.

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.

## ADDING AN EXCEL DATA SOURCE

To add a new Excel data source, follow the steps below:

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**. The **Add Excel Data Source** window appears.




3. Click the folder icon under **Excel Data File** to to display the **Find Files or Folders** window. Navigate to and select the Excel file you want to use as a data source, and then click **OK**.

# TEXT FILE DATA SOURCE

One of the simplest ways to transfer data between systems is through the use of text files. For instance, some industrial equipment produce log files in text format that can be used as a data source in TX DWA to include productivity data in the data warehouse.

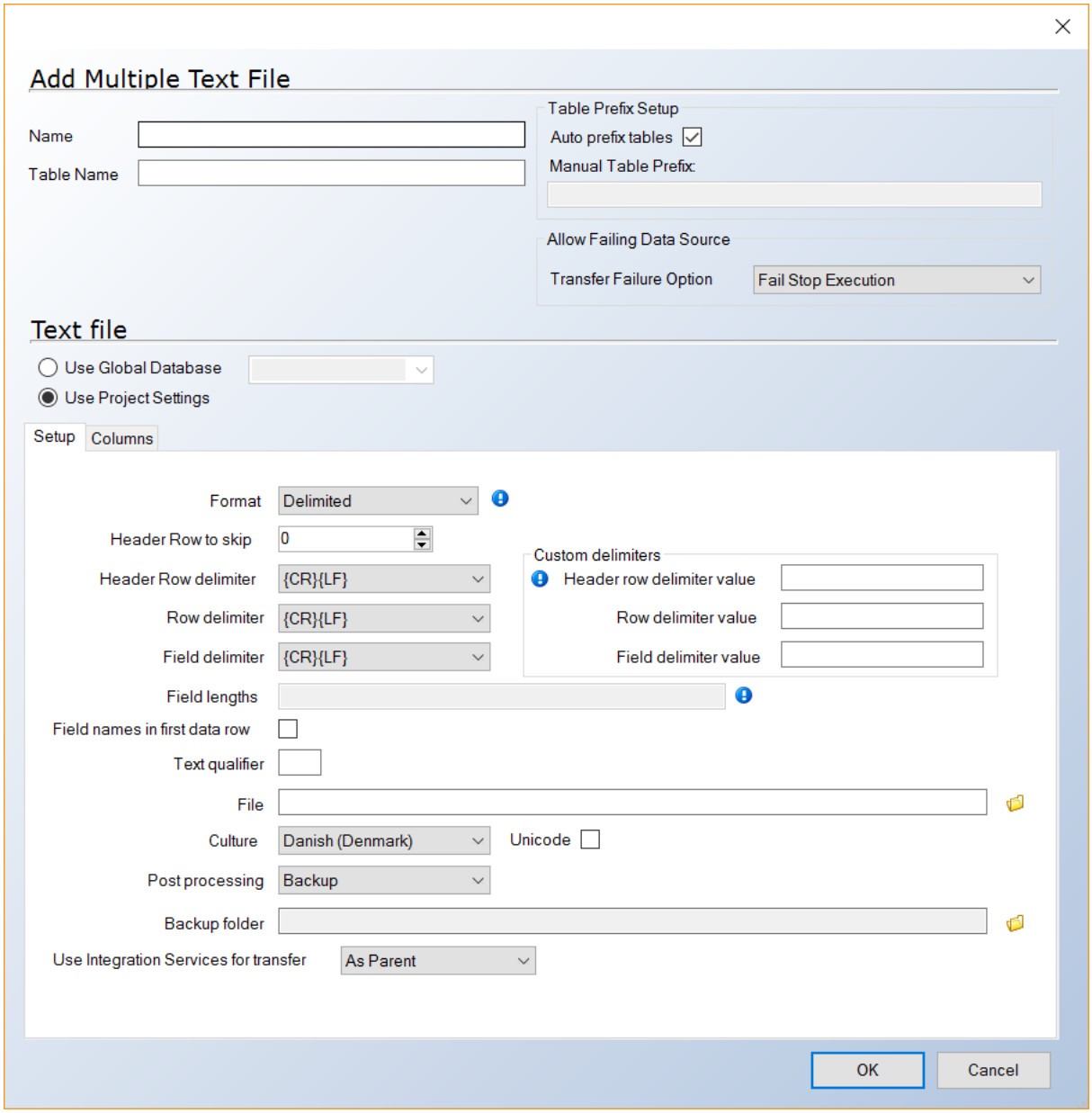
One text file data source corresponds to one table in TX DWA . You can load a single text file or multiple identically formatted text files.

## ADDING A TEXT FILE DATA SOURCE

While virtually identical, you have to choose between the **Single Text File** and **Multiple Text File** data sources when you set up a text file data source. The **Multiple Text File** data source is usually preferable even if you only have one text file since it enables you to add more text files as sources should you need it sometime in the future. To add a text file data source, follow the steps below:

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

The **Add Multiple Text File** or **Add Single Text File** window appears.




3. In the **Name** box, type a name for the data source.
4. In the **Table Name** box, type the name of the table that is created in the staging database. The table is prefixed with the data source name. If you want to set your own prefix, clear **Auto Prefix Tables** and type the prefix you want to use in **Manual Table Prefix** box.
5. (Optional) Click you prefered option in the **Transfer Failure Option** list to change the **Allow Failing Data Source** setting.
6. In the **Format** list, click the format of the text file, e.g. how TX DWA should make sense of the content of the file.
    - Select **Delimited** if rows and fields are seperated by a character and click the relevant characters in the **Header Row delimiter**, **Row delimiter** and **Field delimiter** lists.

- Select **FixedWidth** if the fields have a fixed length and type the lengths in **Field lenghts** in a semicolon-seperated format, e.g. "2;4;8;3".
- Select **RaggedRight** if the last fields is delimited by a character, while the previous fields are fixed width. Click the relevant characters in the **Header Row delimiter**, **Row delimiter** lists and type the lengths in **Field lenghts** in a semicolon-seperated format, e.g. "2;4;8;3".

7. Select **Field names in first data row** if the first row of data contains field name, i.e. not data.
8. Type a **Text Qualifier**, often a quatation mark, if you would like TX DWA to strip from the fields before loading data into the staging data base.
9. If you are adding a multiple text files data source, enter the path to the files you want to process separated by semicolon (;) in the **File** box. You can also use wildcards. Use "*" for any number of characters and "?" for a single character. You can also click the folder icon next to the **File** box to choose the file to process.
   -OR-
   If you are adding a single text file data source, click the folder icon next to the **File** box to choose the file to process.
10. In the **Culture** list, click the language of the text file.
11. Select **Unicode** if TX DWA should treat your file as Unicode.
12. In the **Post processing** list, click the action you want TX DWA to perform when the file has been processes.
    - Select **Backup** to move the file to a backup folder and click the folder icon next to the **Backup folder** field to select the folder.
    - Select **Delete** to delete the file.
    - Select **None** to leave the file as it is.
13. In the **Use Integration Services for transfer** list, you can click Yes or No to change the setting from the default As Parent.
14. Click the **Columns** tab and click **Get Fields**to load the fields, which will then be displayed in a list in the left-hand side of the window. You can select one or more fields in the list and adjust different settings for them:
    - **Column name**
    - **Data type**
    - **Text length**: Enter the maximum number of characters in the field.
    - **Variable length**: Select if you do not want the field to have a fixed length.
    - **Unicode**: Select to convert data to Unicode
    - **Number of decimals**: Enter the maximum number of decimals allowed in the field.
15. (Optional) Clear **Retain null values** to set the value of empty fields to the field's data type's default value instead of null.
16. Click **Update** to show a preview of the data as TX DWA understands it with the settings you have chosen. You have the option of adjusting the **Number of rows** to see more or less rows.
17. Click **OK** to add the data source.

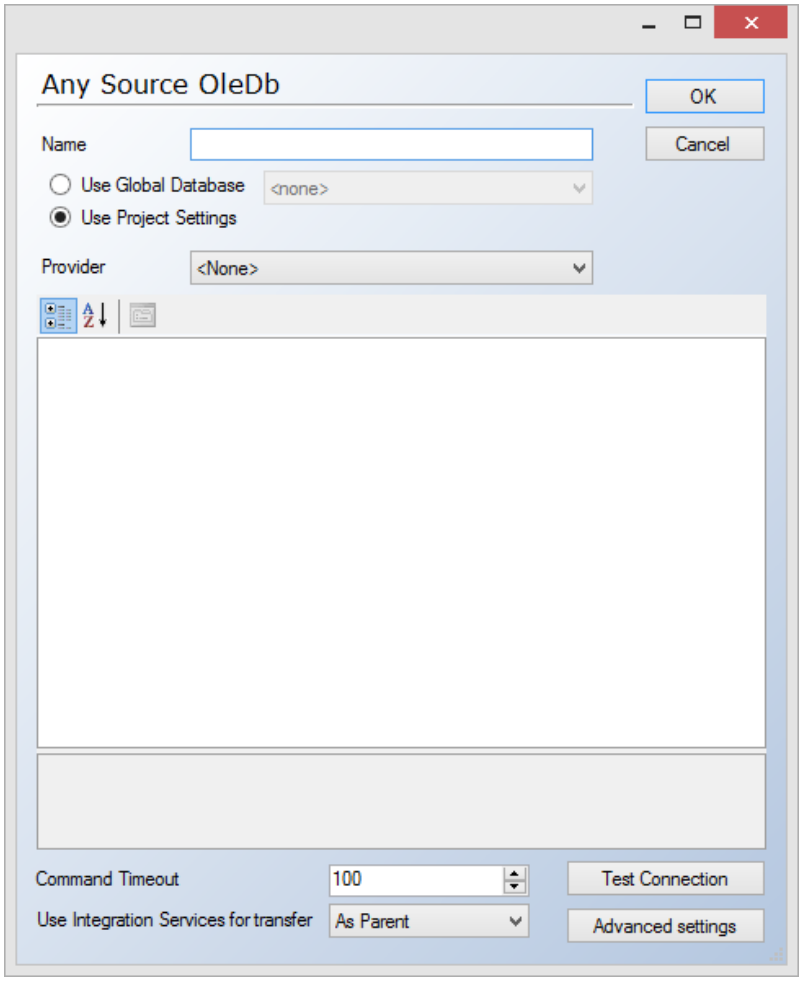
# ANYSOURCE DATA SOURCE

With the AnySource Adapter, you can connect to any data source that you have an OLE DB or ADO provider installed for on your server. Instead of waiting on us to create an adapter that enables you to connect to a specific system, you can now acquire a provider from the system developer or a third party vendor.

## ADDING A ANYSOURCE OLE DB OR ADO DATA SOURCE

The setup for the AnySource adapter is similar for the OLE DB and ADO based adapter. To add an AnySource OLE DB or ADO data source, follow the steps below.

1. On the **Data** tab, right click **Data Source**, click **Data Sources** and click **Add AnySource OLE DB** -OR- **Add AnySource ADO**. The **AnySource OLE DB**-OR-**AnySource ADO** window opens.




2. In the **Name** box, type a name for the data source.
3. In the **Provider** list, click the provider you want to use.
4. In the property sheet, edit the settings for the connection. See the documentation for the provider you are using for more information.

5. (Optional) In **Command Timeout**, enter the number of seconds after which a command should time out.
6. (Optional) In the **Use Integration Services for Transfer** list, click **As Parent** to use the same setting as the parent business unit or **Yes** or **No** to use or not use Integration Services for transfer, respectively.
7. Click **Test Connection** to verify that the connection is working.
8. Click **Advanced settings** to access additional settings. The **Advanced Data Source Properties** window opens.
9. In the **Query Formatting** list, type the **Prefix**, **Suffix** and **Separator** used in the source. Click **Read Value** to fill in the values automatically if possible.
10. In the **Character Replacements** list, you can type a **Replace Character** to replace with the **Replace Value** in data from the data source.
11. In the **Schema Properties** list, type the **Schema Name**, **Table Name**, **Column Name** etc.
12. In the **Object Filtering** list, you can choose to filter the tables you receive from the provider using regular expressions or different string comparisons. Click **Table** or **Schema** in the **Object Type** list, click a filter type in the **Filter Type** list and type a value in the **Filter Value** List.
13. Click **OK** to close the **Advanced Data Source Properties** window.
14. Click **OK** to close the **AnySource OLE DB** -OR- **AnySource ADO** window and add the data source.

**Note**: On other data sources, you can use the preview feature in TX DWA to view the content of a table on the source. This might not work on all AnySource data sources since the TX DWA does not know exactly what the source is and what syntax to use. However, you can use the query tool to explore the content of a table. See Query Tool for more information.

# CUSTOM DATA SOURCE

The Custom Data Source works in conjunction with a separate provider - or driver - to enable access to data sources that are not supported by the core TX DWA product.

## ADDING A CUSTOM DATA SOURCE

The setup for a Custom Data Source depends on the provider you are using. While the general steps to add a Custom Data Source is explained below, you should consult the documentation for the provider to learn more about the specific settings.

1. On the **Data** tab, right click **Data Source**, click **Data Sources** and click **Add Custom Data Source** The **Add Custom Data Source** window appears.
2. In the **Name** box, type a name for the data source.
3. In the **Provider** list, click the provider you want to use.
4. In the **Setup Property** list, click the property you want to edit. Go through all properties in the list and edit the settings for the connection in the property list below.
5. Click **Test Connection** to verify that the connection is working.
6. Click **OK** to close the window and add the data source.

# CONNECTING WITH APPLICATION ADAPTERS

TX DWA supports two different approaches for connecting to source systems: simple data sources and intelligent application adapters.

A data source connection simply connects to the source and enables you to browse the content of the source.

An adapter is a component that enables you to easily extract and synchronize data from different source systems. The adapter knows how a given system organizes and stores data, which enables the adapter to simplify the table structure you see in TX DWA. For instance, data for each company in a Dynamics NAV system is stored in a separate set of tables. The Dynamics NAV adapter merges these tables together and lets you select companies on a global level.

TX DWA includes Application Adapters for the following systems:

- Microsoft Dynamics NAV
- Microsoft Dynamics AX
- Microsoft Dynamics GP
- Microsoft Dynamics CRM
- SAP
- Sun Systems
- Salesforce
- Infor Movex/M3
- UNIT4 Agresso

# MICROSOFT DYNAMICS AX ADAPTER

This adapter simplifies the extraction of data from Microsoft 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 TX DWADynamics 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.

### IMPORTING 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 TX DWA project. For more information, see Add Dynamics AX Adapters.

### ADDING A DYNAMICS AX ADAPTER

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.

### ADDING 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.

### ADDING A DYNAMICS AX ODATA PROVIDER

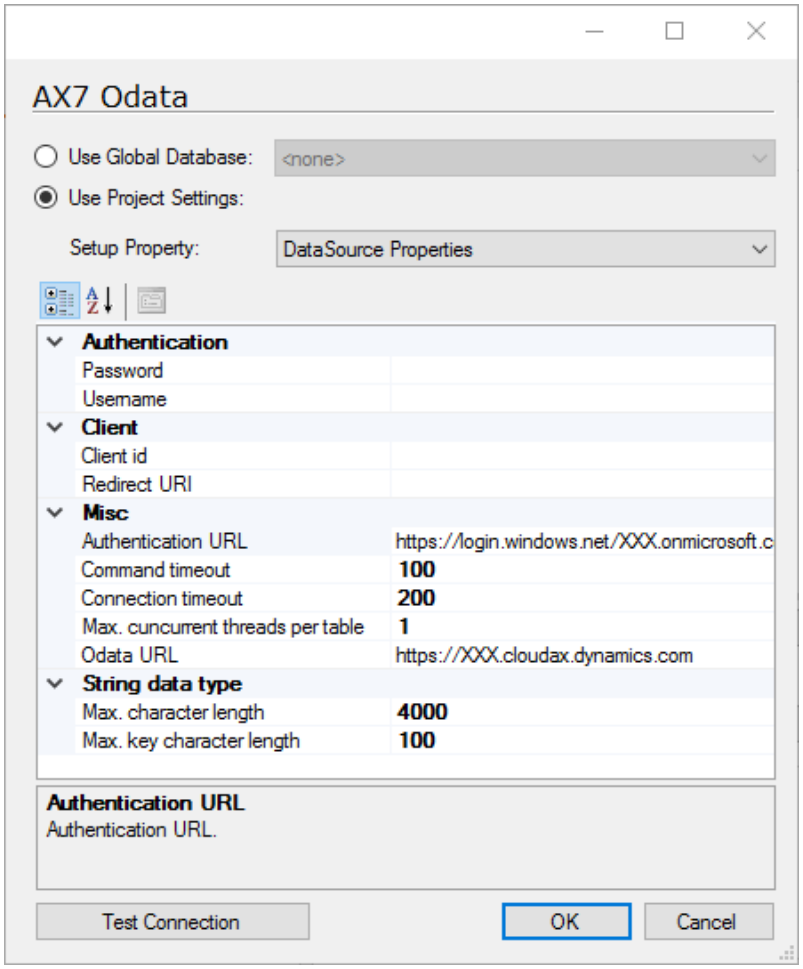
The version of Dynamics AX released in 2016 - known as Dynamics AX 7 - introduced a new Azure-based data storage option. You can access this through the "AX7 Odata" provider.

The provider uses OAuth 2.0 authentication, which means you need to set up a client in Azure AD to connect - please see this article for more information: https://ax.help.dynamics.com/en/wiki/dynamics-ax-7-services-technical-concepts-guide/

Note that this provider cannot use SSIS transfer.

To add a Dynamics AX OData provider, follow the steps below.

1. Right-click the adapter, click **Source Providers** and click **Add AX7 Odata Provider**. The **AX7 Odata** window appears.




2. In the **Password** row, enter the password you use to connect.
3. In the **Username** row, enter the username you use to connect.

4. Under client, enter the **Client Id** and **Return URI** you use to connect.
5. In the **Authentication URL** row, enter the URL used for authentication.
6. (Optional) In the **Connection Timeout** row, modify the number of seconds to wait before terminating an attempt to connect to the server.
7. (Optional) In the **Command Timeout** row, modify the number of seconds to wait before terminating an attempt to connect to the database.
8. (Optional) In the **Max. concurrent threads per table** row, modify the number of connections you will allow to a single table.
9. In the **Odata URL** row, enter the url that you will get the data from.
10. (Optional) In the **Max. character length** row, set the max lenght in characters for fields that are strings.
11. (Optional) In the **Max. key character length** row, set the max lenght in characters for key fields that are strings.
12. If your need to connect through a proxy server, click **Proxy settings** in the **Setup property** list. In the **WebProxyApproach** list you have the following options:
    - **NoProxy**
    - **ApplicationProxy:** Use the proxy settings configured on the application level. To adjust the application settings, click **Options** on the **Tools** menu and then click **Proxy Settings**. Note that using a proxy server for Internet connections can also be turned on and off from here.
    - **SpecificProxy:** Enter the settings - server, port, username and password - to use for the adapter.

### ADDING 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**.

### SETTING THE ACCOUNT TABLE

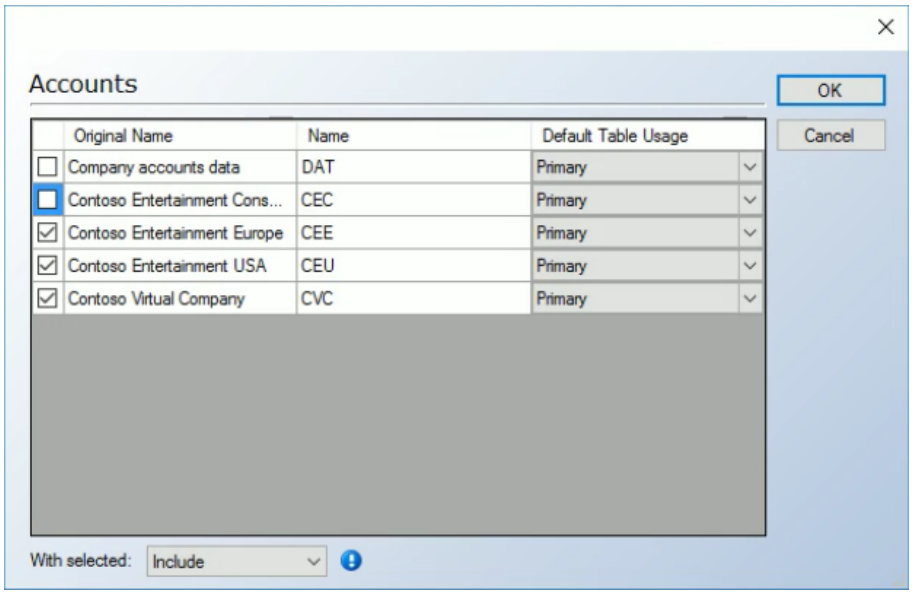
Before you can continue with setting up accounts, you have to verify that the company account table is correct.

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

### SETTING 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 **Set Up Accounts**. An information message is displayed, which lists the accounts that have been added. Click **OK**. The **Accounts** window appears.




2. Select the accounts from which you want to retrieve data. In the **With selected**list, click **Include** to use the selected accounts in the data warehouse or click **Exclude** to exclude the selected accounts and include all other accounts. The last option is useful if you often add new accounts in Dynamics AX and want to make sure that all accounts are included in the data warehouse as soon as they exist in the ERP system. Note that the exclude option isn't available when you use the OData provider.
3. In the **Default Table Usage** list, specify the order in which data from the tables is retrieved and read. You have the following options:

| Option | Definition |
|---|---|
| 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 Modifying Table Usage on Dynamics AX Tables. |

4. Click **OK**.

### LOADING AND SELECTING 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 **Data Selection** pane displays all tables and fields.
3. In the **Data Selection** pane, select the tables and fields you want to extract to your staging database.

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

### ADDING 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**.

### VIEWING DYNAMICS AX TABLE INFORMATION

TX DWA 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:

| Fields Tab | Description |
|---|---|
| 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 |

| Relations Tab | Description |
|---|---|
| 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. **ThisFixed** specifies relation fields to restrict the records in the primary table. **ExternFixed** specifies relation fields that restrict the records in the related table |

| Virtual Company References | Description |
|---|---|
| Company | The name of the company account |
| Virtual Company | The name of the Virtual Company that contains tables shared by several company accounts |

### VIEWING 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.

### CHANGING 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.

## MODIFYING 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 want to change the order of priority on for data retrieval. You have the following options:

| Priority | Definition |
|---|---|
| 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**.

### MODIFYING 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:

| Priority | Definition |
|---|---|
| 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. |

# MICROSOFT DYNAMICS NAV ADAPTER

This 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 only 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 need to include Sum Index Flow Technology (SIFT) tables, and then click **OK**.

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 **TX DWA** 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:

| Option | Definition |
|---|---|
| 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:

| Option | Definition |
|---|---|
| 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:

| Priority | Definition |
|---|---|
| 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:

| Priority | Definition |
|---|---|
| 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**.

# MICROSOFT DYNAMICS GP ADAPTER

This 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. WithTX DWA 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 A DYNAMICS GP ADAPTER

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.
5. In the **Database** field, enter the name of the database (this should be the DYNAMICS database).
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. 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.
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**.
9. 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**.

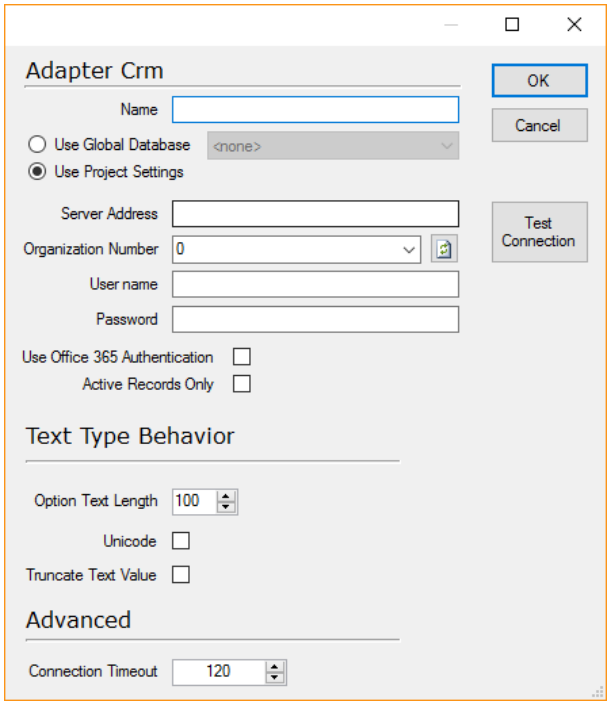
The tables and fields are displayed in the data source tree and in the staging database tree.

# MICROSOFT DYNAMICS CRM ONLINE ADAPTER

This adapter simplifies extraction of data from Microsoft Dynamics CRM Online.

## ADDING A DYNAMICS CRM APPLICATION ADAPTER

1. In the project tree, expand **Business Units**, expand the relevant business unit, right click **Data Sources**, click **Adapter Data Sources** and click **Add Dynamics CRM Adapter**. The **Add CRM Adapter** window opens.




2. In the **Name** box, type a name for your data source.
3. In the **Server address** box, enter the URL of the Dynamics CRM server.
4. In the **Organization number** list, click on or enter the number of the organization you want to see data from. Click the refresh button next to the list to refresh the list of available organization numbers.
5. In the **Username** box, enter the user name you want use for authentication.
6. In the **Password** box, enter the password corresponding to your user name.
7. Check **Use Office365 authentication**to use Office365 for authentication instead of username and password.
8. Checl **Active records only** to fetch only active records.
9. In the **Option text length** box, enter the max length set for data types that does not have a maximum length defined.
10. Check **Unicode** to set text fields to Unicode.
11. Check **Truncate text value** to make the data fit the data types by removing any extra data.
12. In **Connection Timeout**, enter the number of seconds to wait for a connection to be established.

13. Click **Test Connection** to test if the connection properties are correct.
14. Click on **OK** to add the adapter data source.

# SALESFORCE ADAPTER

The Salesforce application adapter enables you to extract data stored in your Salesforce Sales Cloud CRM. You will need an edition of Salesforce Sales Cloud that enables the use of the Salesforce AP. At the time of writing, this means at least the Enterprise edition.

The adapter uses the Salesforce REST API to extract data. The data types of the extracted data is converted from Salesforce data types to their SQL Server equivalents.

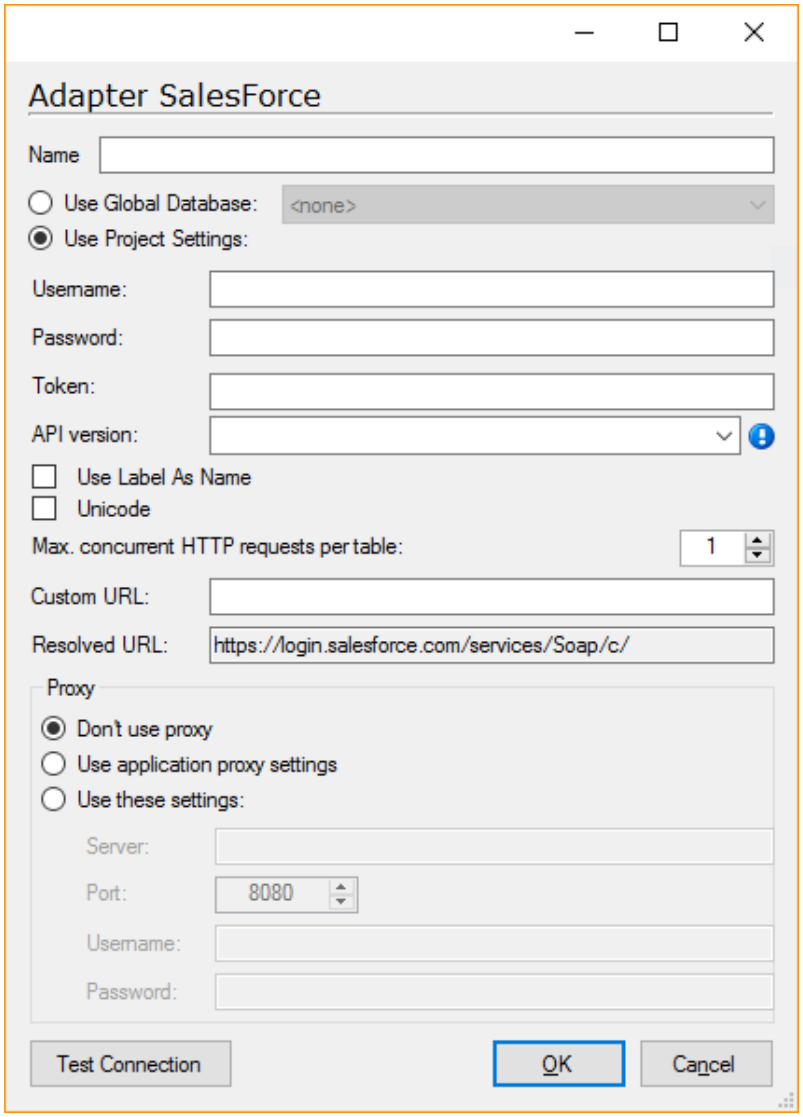
**Note:**The Salesforce adapter requires .NET Framework 4.5 or higher. This is necessary because the Salesforce REST API uses the TLS 1.1 protocol, which is not supported by earlier versions of the .NET Framework. TX DWA only requires .NET Framework 4.0, but if you do not have the .NET Framework 4.5 installed, TX DWA will use TLS 1.0, which will result in connection errors.

## ADDING A SALESFORCE ADAPTER

To connect to a Salesforce data source using the application adapter, follow the steps below.

1. On the **Data** tab, in the project tree, expand **Business Units**, expand the relevant **Business Unit**, right click **Data Source**, click **Adapter Data Sources** and click **Add Salesforce Adapter**.

   The **Add Salesforce Adapter** window opens.




2. Type the **Name** you want to use for the adapter.
3. Enter your **Username**, **Password** and **Token**. The token is provided by Salesforce and will change if the password is changed. Salesforce can be configured not to use security tokens. In that case, simply leave **Token** empty.
4. In the **API Version** list, click the Salesforce API version you want to use. Usually, you should use the newest version. However, if Salesforce makes changes in the API that breaks the adapter, you have the option of using an earlier version.
5. Select **Use Label as Name** to use labels as names instead of the physical system names.
6. Select **Unicode** to make string based fields Unicode ready.
7. (Optional) Enter a **Custom URL** if you want to connect to your Salesforce sandbox instead of the default URL. The **Resolved URL** box shows you the entire URL the adapter will use to connect.

8. In **Max. concurrent HTTP requests per table**, type or select the maximum number of HTTP requests you want to allow per table. For each request, the adapter fetches 1000 rows.
9. If you need to connect through a proxy server, you have the following options:
    1. **Don't use proxy**
    2. **Use application proxy settings:** Use the proxy settings configured on the application level. To adjust the application settings, click **Options** on the **Tools** menu and then click **Proxy Settings**. Note that using a proxy server for Internet connections can also be turned on and off from here.
    3. **Use these settings**: Enter the settings - server, port, username and password - to use for the adapter.

# SAP APPLICATION ADAPTER

The SAP Application Adapter allows you to connect to SAP systems. The Adapter utilizes the **XTract IS** component from Theobald Software. Make sure that the latest version of the component is downloaded and installed on the server where TX DWA is installed. The component can be downloaded from Theobald Software: http://theobald-software.com/en/product-downloads.html

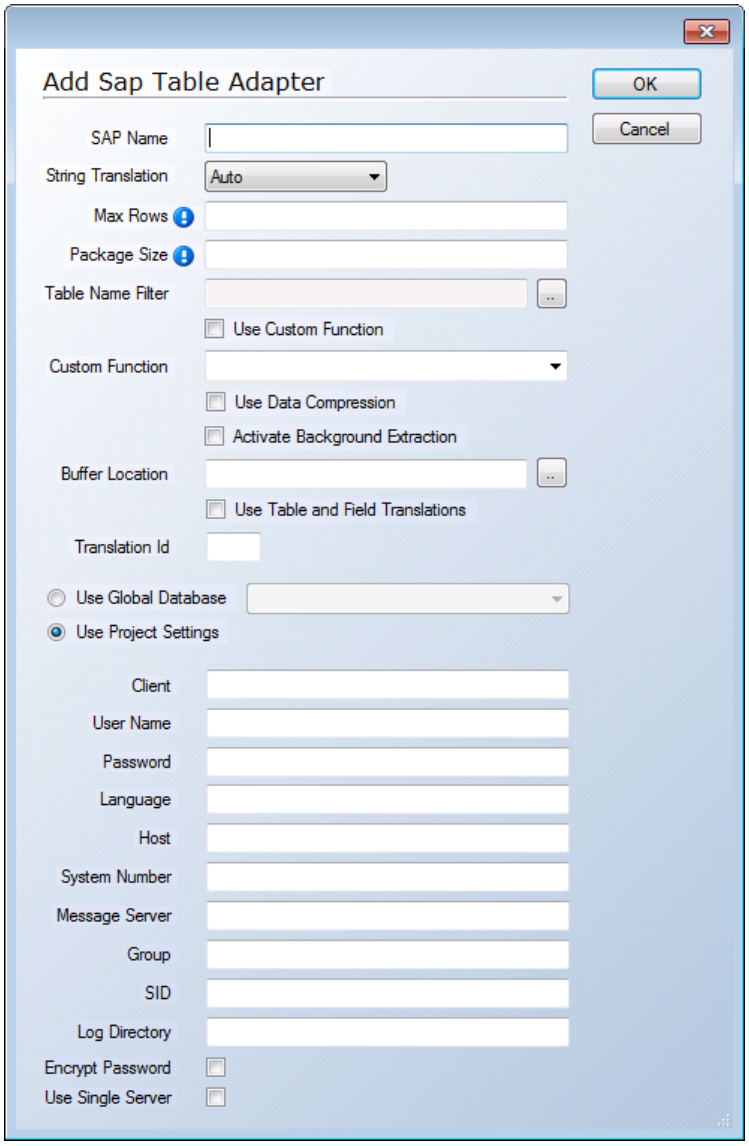
Due to some issues in the standard SAP meta-data layer, you will need to install a function module in SAP. For more information, see https://my.theobald-software.com/index.php?/Knowledgebase/Article/View/56/3/installing-z_xtract_is_table

The function module will make it possible to:

- Extract tables/table columns with an overall width greater than 512.
- Extract tables that contain at least one column of type F (floating point).
- Extract table TCURR which has some meta data problems in the Data Dictionary.

## ADDING A SAP APPLICATION ADAPTER

1. In the project tree, expand **Business Units**, expand the relevant business unit, right click **Data Sources**, click **Adapter Data Sources** and click **Add SAP Table Adapter**. The **Add SAP Table Adapter** window opens.



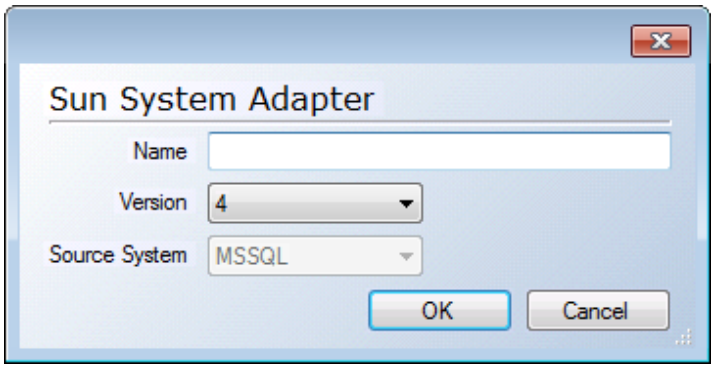

2. Fill in the settings that matches your setup and click **OK**.

# SUN SYSTEM ADAPTER

The Sun System adapter simplifies the extraction of data from a Sun System using a Microsoft SQL Server.

## ADDING A SUN SYSTEM ADAPTER

1. In the project tree, expand **Business Units**, expand the relevant business unit, right click **Data Sources**, click **Adapter Data Sources** and click **Add Sun System Adapter**. The **Add Sun System Adapter** window opens.



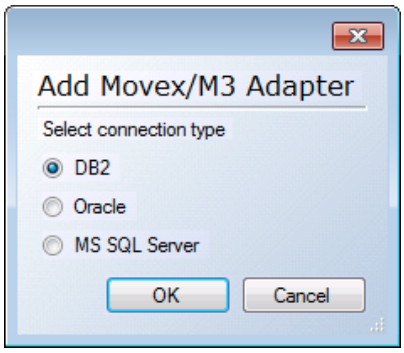

2. Type a **Name** for the adapter to easily identify it in the project tree.
3. In the **Version** list, click the version of Sun System you are connecting to.
4. Click **OK**.
5. The standard window for adding a Microsoft SQL Server data source opens. Enter the connection details for your data source - see Adding a SQL Server Data Source for more information.

# MOVEX/M3 ADAPTER

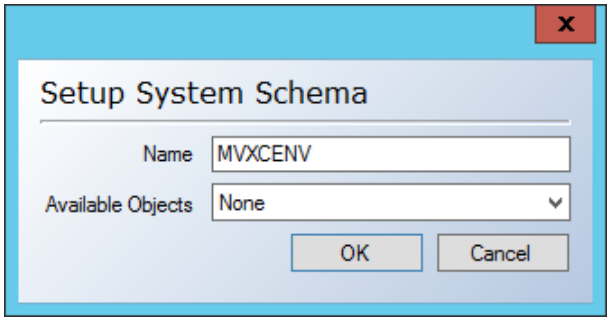
The Movex/M3 adapter simplies extraction of data from Infor M3, an ERP system also known by its previous name Movex.

## ADDING A MOVEX/M3 ADAPTER

1. In the project tree, expand **Business Units**, expand the relevant business unit, right click **Data Sources**, click **Adapter Data Sources** and click **Add Movex/M3 Adapter**. The **Add Movex/M3 Adapter** window opens.

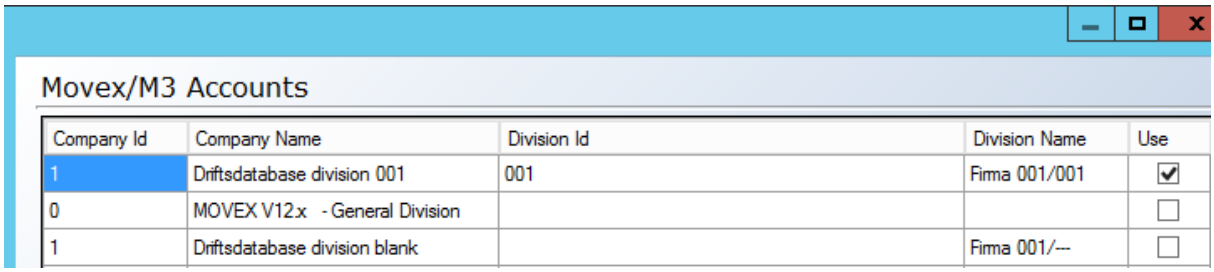



2. Click the database type of your M3 installation. You have the following options:
   - DB2
   - Oracle
   - Microsoft SQL Server
3. Click **OK**.
4. The standard window for adding a data source of the type you chose opens. Enter the connection details. See Adding a DB2 Data Source, Adding a Oracle Data Source and Adding a SQL Server Data Source for more information.
5. In the project tree, right click the M3 data source you just added and click **Change System Schema**. The **Setup System Schema** window opens.




6. Type the environment schema name in the **Name** box.
7. In the **Available Objects** list, click **None** to make no tables available for use in the project, **Company Tables Only** to make only tables from the chosen company available or **All Tables and Views** to make every table and view in the M3 database available. Click**OK**.

8. In the project tree, right click **Setup Accounts**. The **M3/Movex Accounts** windows opens.




9. Select the companies and divisions to include by checking the box in the **Use** column. Click **OK**.

# AGRESSO ADAPTER

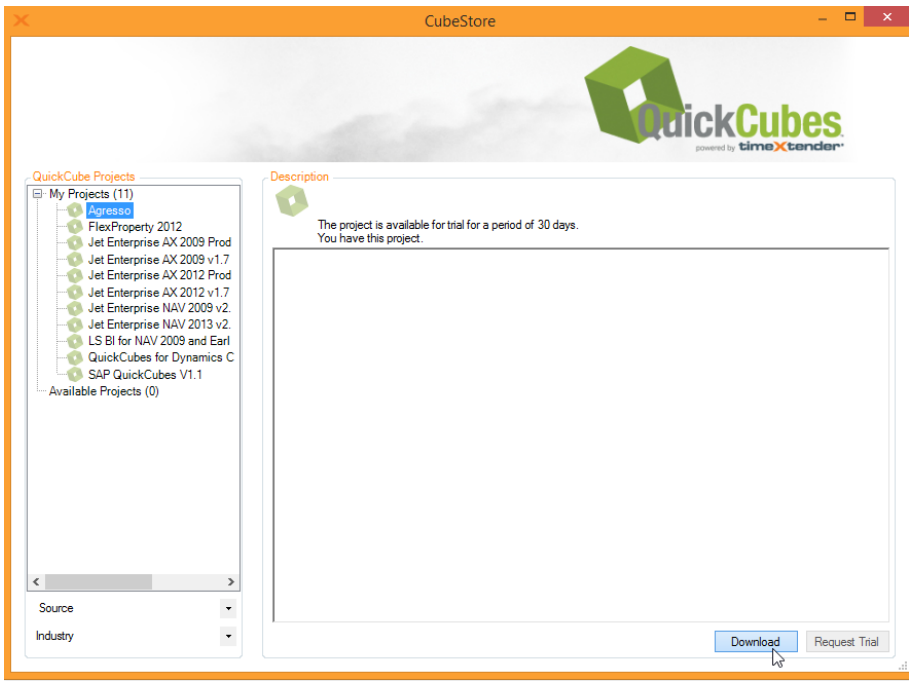
The Agresso Adapter enables the use of the UNIT4 Business World ERP system in TX DWA. Agresso is the former and more well known name of the system and as such, it is used in TX DWA.

## ADDING AN AGRESSO ADAPTER DATA SOURCE

Adding an Agresso Adapter consists of two major parts. Unlike other application adapters, the Agresso Adapter comes with a TX DWA project based on Agressos standard dimensional setup. Downloading and configuring this project is the first part, while the second part is configuring Agresso-specific settings for client, dimensions, translations and flexi-tables.

To download and configure the Agresso project, follow the steps below.

1. Click the application button in the top left corner of TX DWA and click **CubeStore**. The Cubestore window opens.

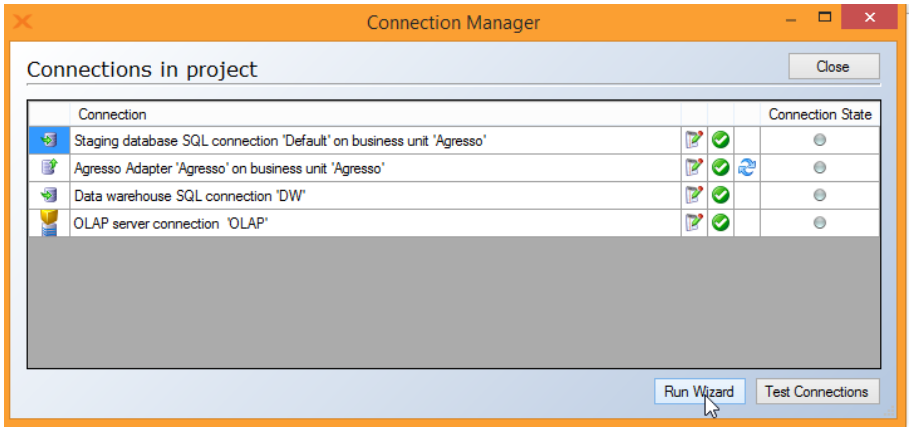



2. In the **QuickCube Projects** list, click **Agresso** and then click **Download**.

   **Note:** The project is only available if you have a license for TX DWA that includes the Agresso Adapter.

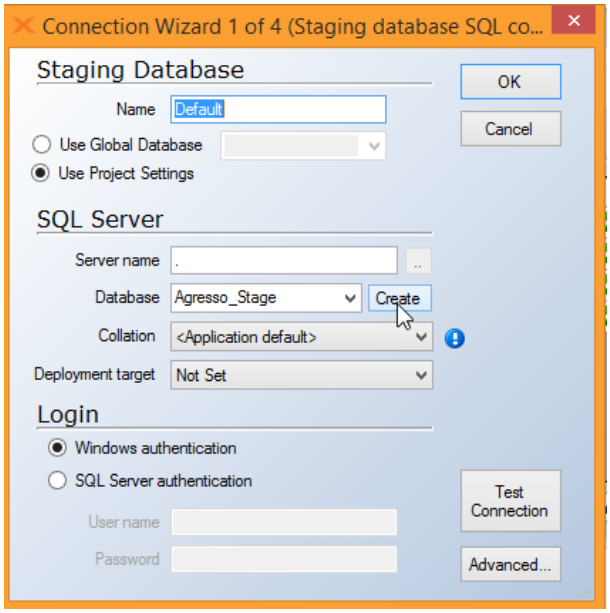
3. One the project has been downloaded, TX DWA will ask you if you want to run the Connection Manager. Click **Yes**.

4. The Connection Manager window opens.

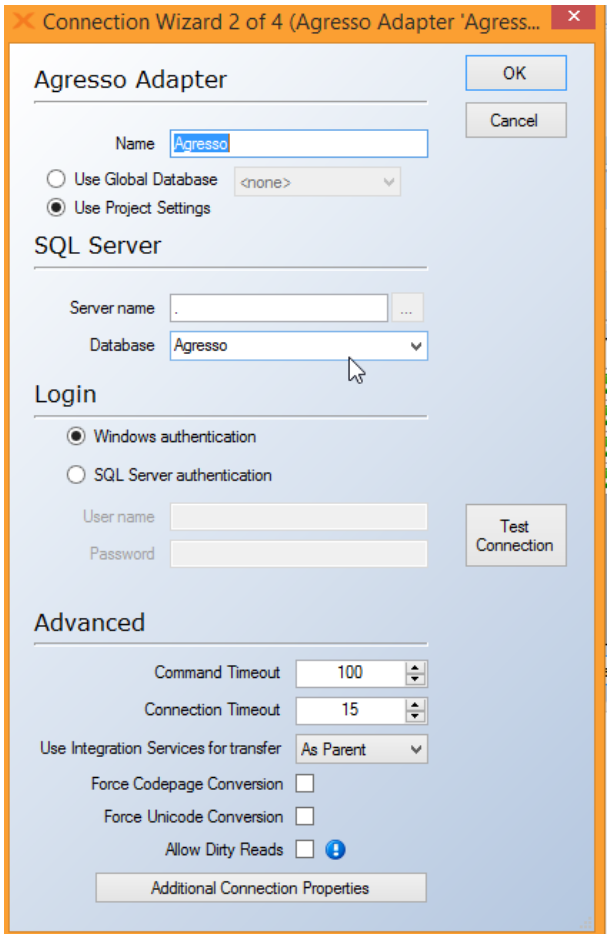



Click **Run Wizard**. Four windows will now open in tour, allowing you to quickly configure the different connections.

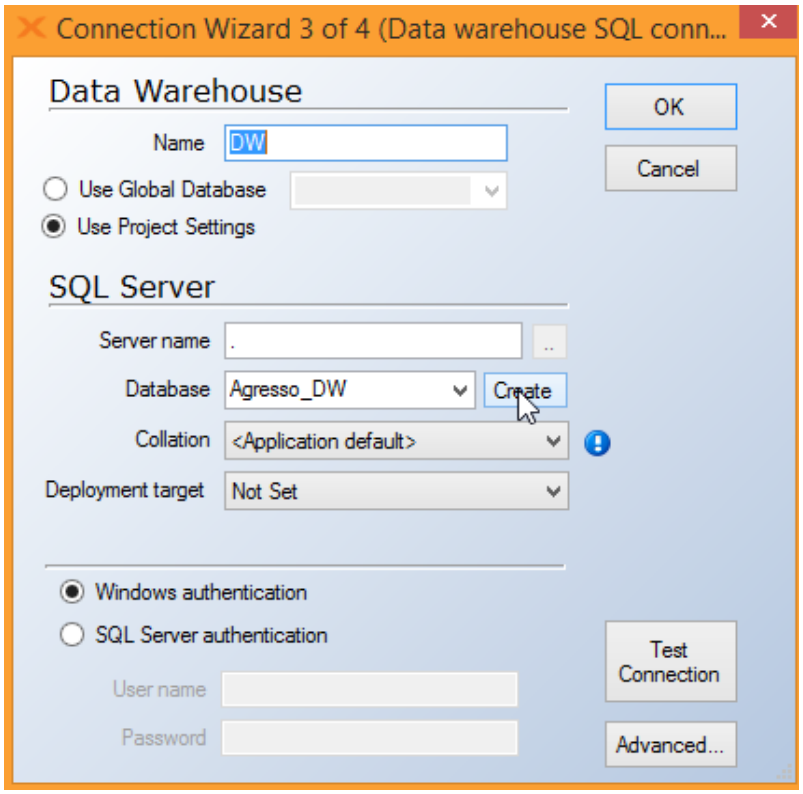
5. Create a staging database for use with the project and click **OK**. For more information, see Setting Up a Staging Database.



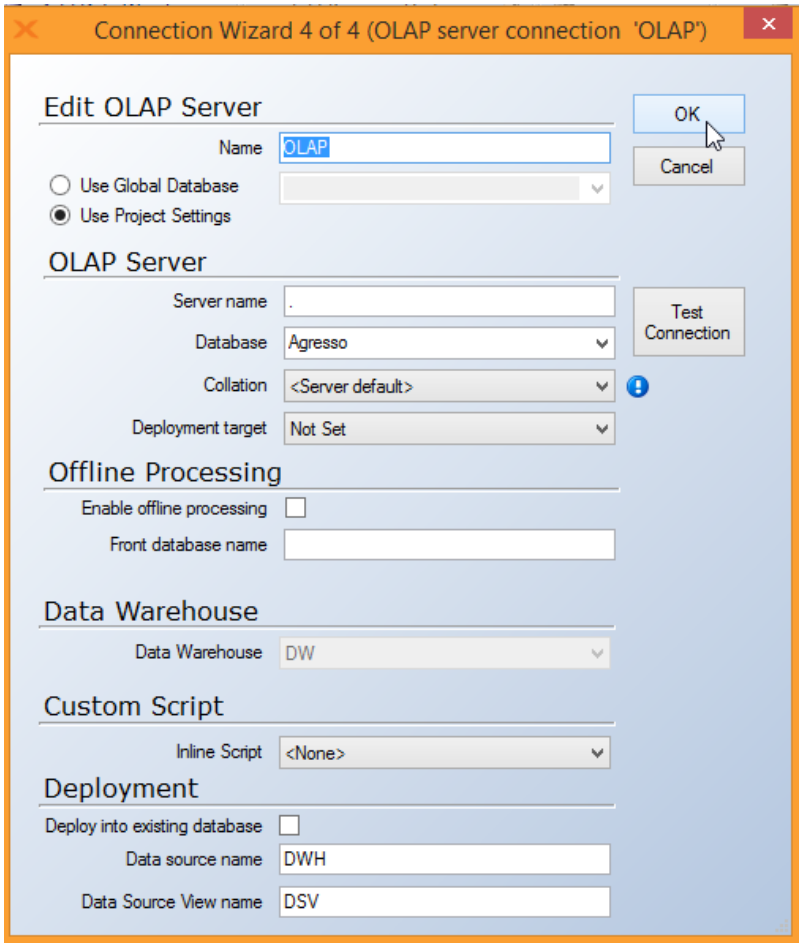
6. Enter the connection information for your Agresso database and click **OK**. For more information, see Adding a SQL Server Data Source.



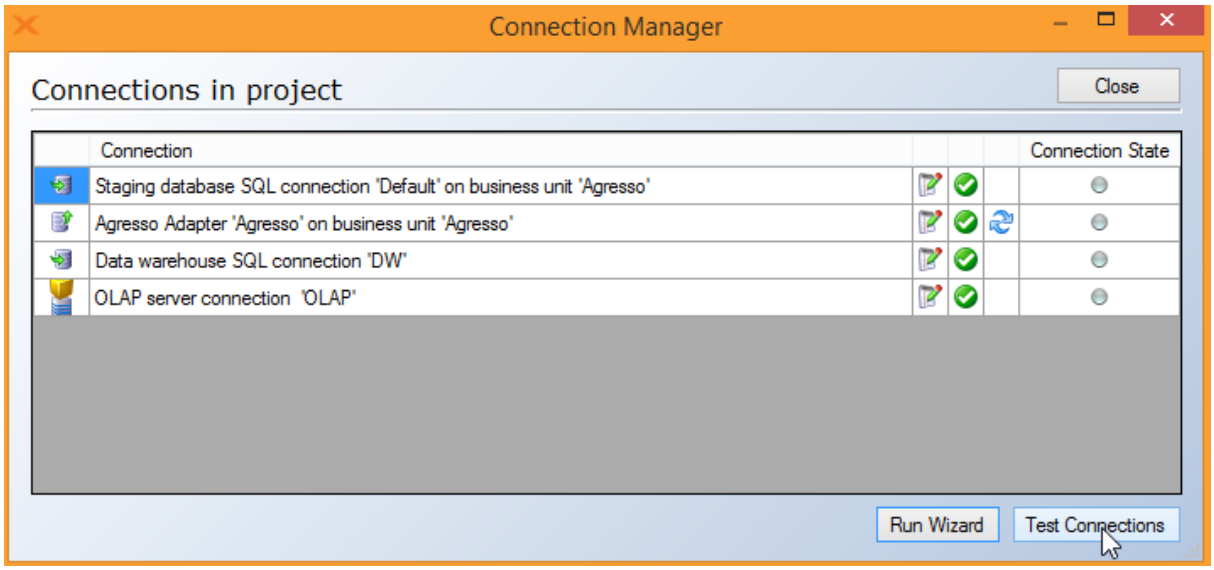
7. Create a data warehouse database and click **OK**. For more information, see Adding a Data Warehouse.




8. Set up an OLAP server for the project. For more information, see Adding an OLAP Server.



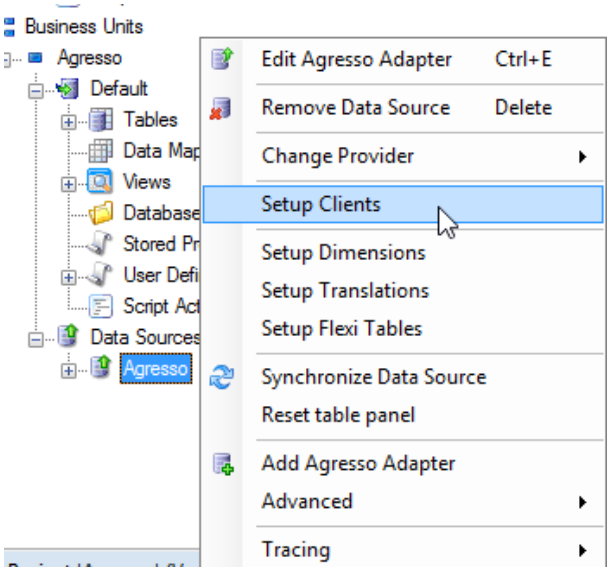

9. Back in the **Connection Manager**, click **Test Connections** to ensure that all connections work and click **Close**.



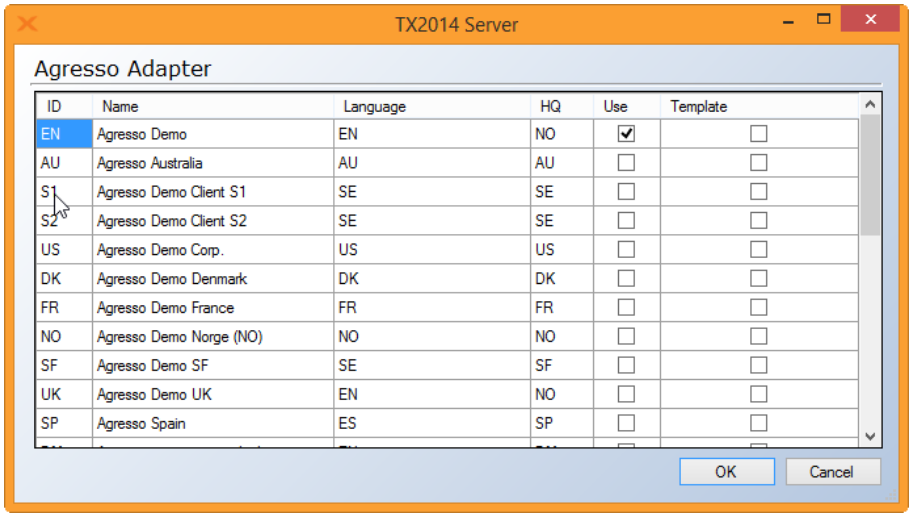

As mentioned earlier, the second part of adding an Agresso Adapter involves setting up the clients, dimensions, flexi-tables and translations that are part of the Agresso system.

To configure the Agresso-specific settings, follow the steps below.

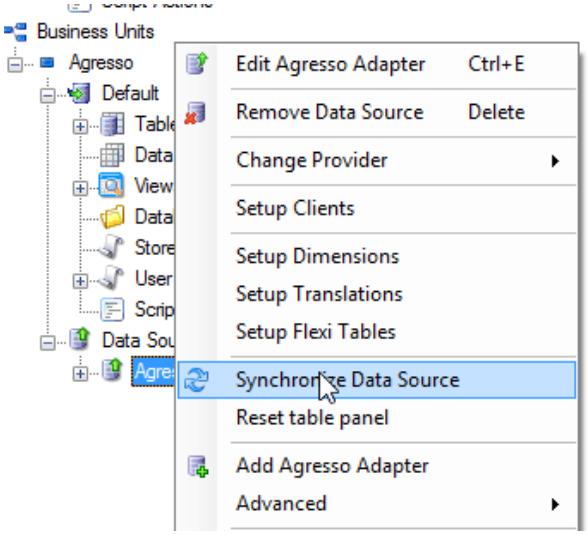
1. In the project tree, right-click the Agresso Adapter you just added and click **Setup Clients**.




A window containing a list of clients opens.

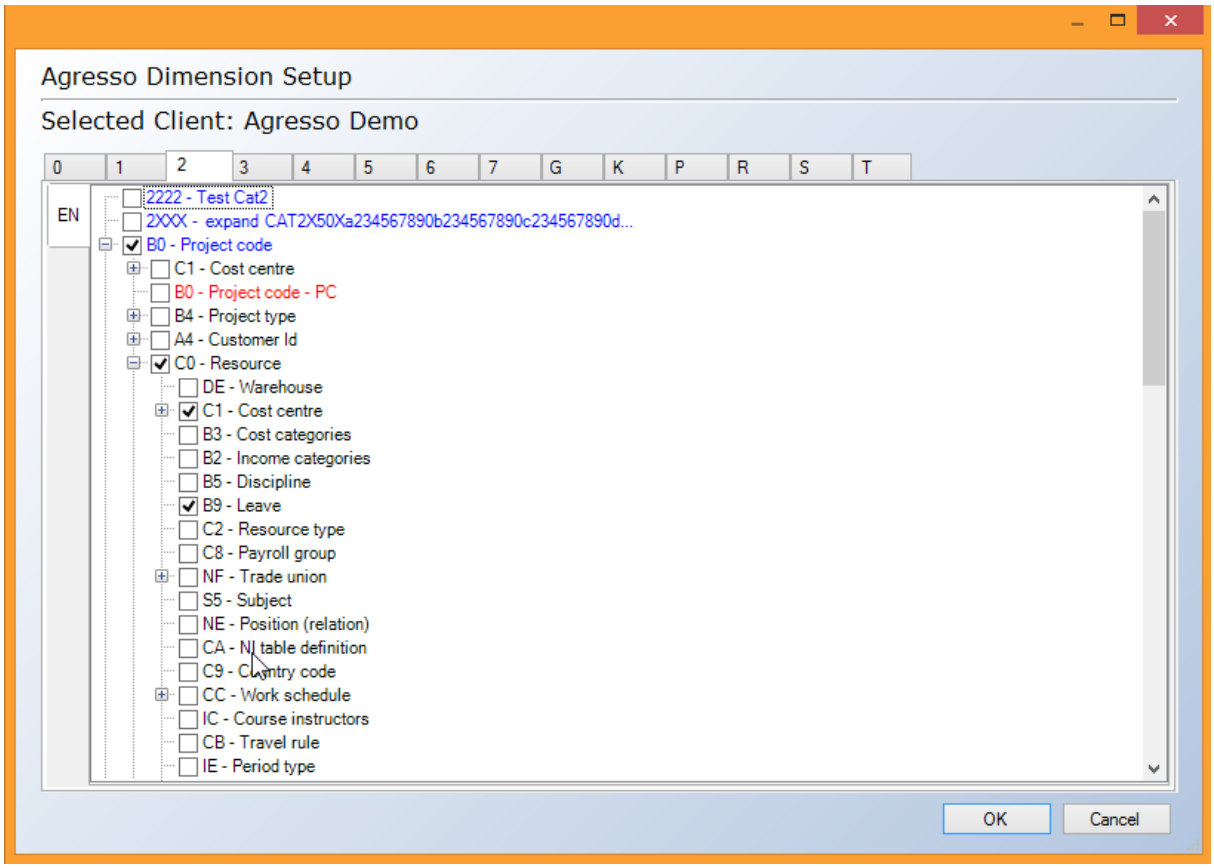



2. Choose the clients you want to extract data from by selecting the check boxes in the **Use** column. If you want a client to serve as a template for the dimensional setup (see below), select the check box in the **Template** column. Click **OK**.
3. In the project tree, right-click the Agresso Adapter and click **Synchronize Data Source**.



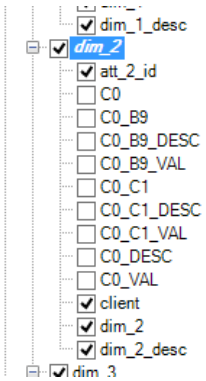

e synchronization window opens. Wait for the process to finish and click **Close**.

4. Before the next step, the project needs to be deployed and executed. On the **Tools** menu, click **Deploy and execute project**. The **Deploy and Execute** window opens. Click **Start**, wait for the process to finish, and click **Close**.
5. In the project tree, right-click the Agresso Adapter and click **Setup Dimensions**. The **Agresso Dimension Setup** window opens.

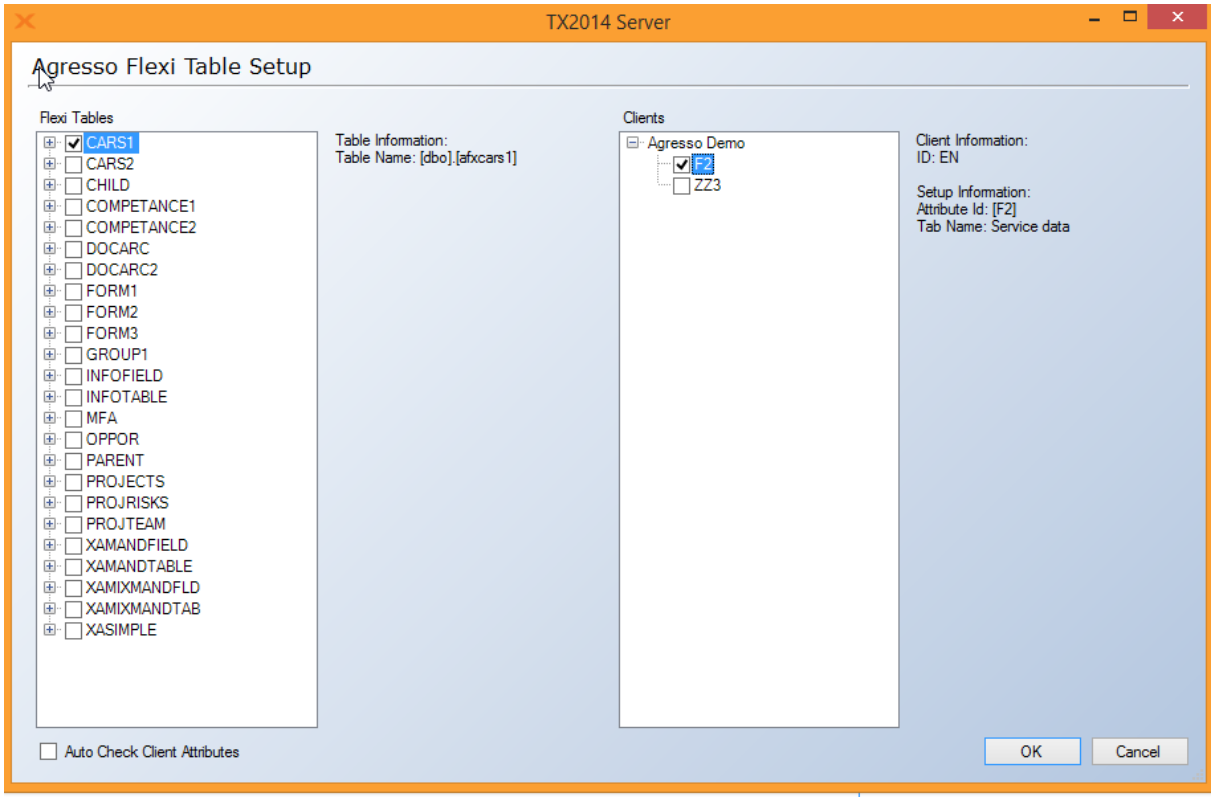



6. Click **Load Dimension Values** (Default setting only shows previously selected dimension attributes)

7. Select the dimension attributes you want to use in the project. If you did not select a template in the client setup, you need to do this for each client.
8. Click **OK**. The selected attributes are automatically added to the dimension table and can then be used in the project.

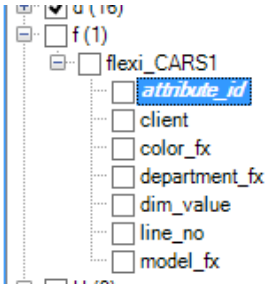



9. In the project tree, right-click the Agresso Adapter and click **Setup Flexi Tables**. The **Agresso Flexi Table Setup** window opens. The content of the **Flexi Tables** and **Clients** list depends on the clients and dimensions you have chosen previously.

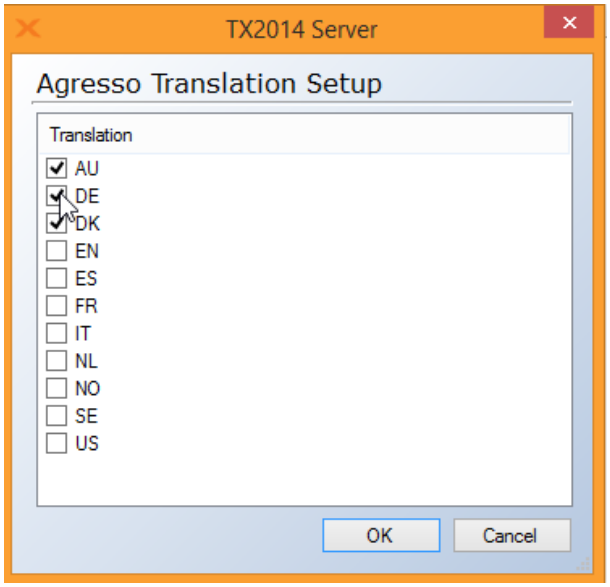



10. Select the combinations of flexi tables and client attributes that you want to use in your project.

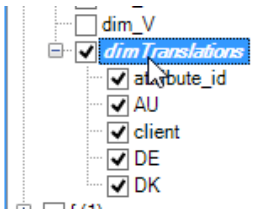
11. Click **OK**. The selected Flexi Tables are automatically added to the source tables and must be integrated into the solution.




12. In the project tree, right-click the Agresso Adapter and click **Setup Translations**. The **Agresso Translation Setup** window opens.




13. Select the translations you want to use in your project from the **Translation** list.
14. Click **OK**. The selected translations are automatically added to the source tables for use in the project.



# DESIGNING THE DATA WAREHOUSE

Once you have set up your data warehouses and business units and established connections to the sources you want to use, it is time to start designing your data warehouse. This involves transferring data from the source systems to the data warehouse(s) via a staging database and applying transformations and data cleansing rules along the way.

# DIMENSIONAL MODELING

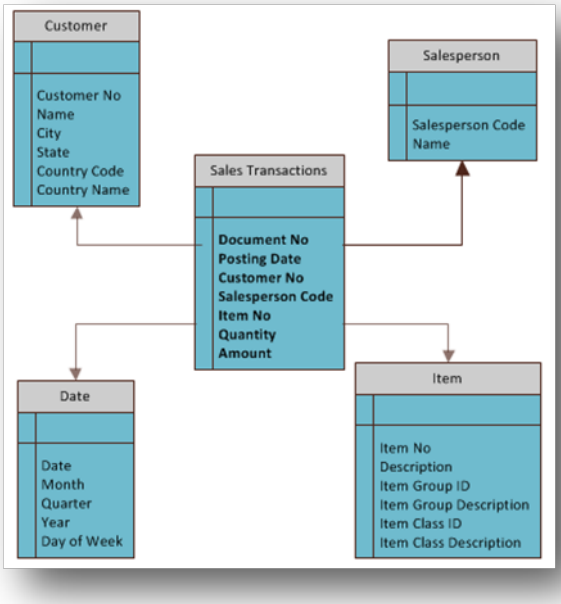
Dimensional modeling is the technique and methodology used in data warehouse design. A dimensional model typically consists of a fact table and a number of dimension tables taking the form of a star schema or a snowflake schema.

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

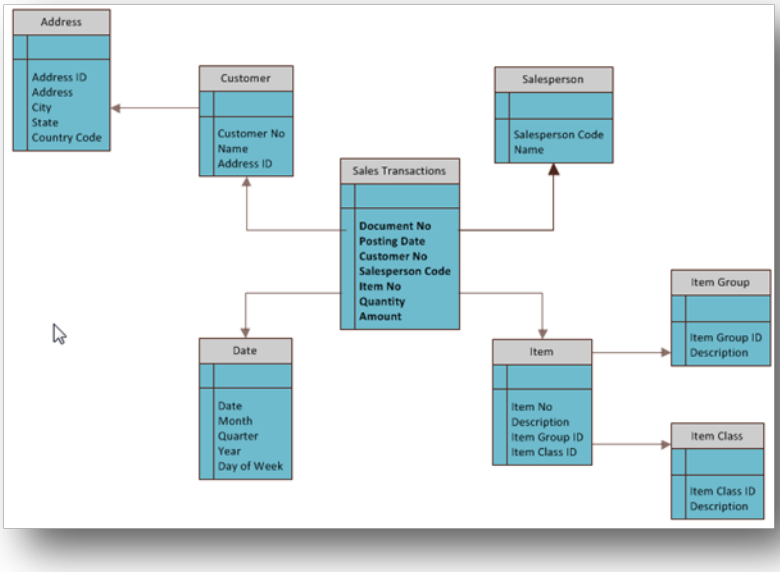
A star schema represents a fact table that is directly joined to all dimension tables. The schema resembles a star with the fact table at the center and the dimension tables as the points of the star. In a star schema, all of the descriptive information for a particular dimension is contained in a single 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.



# DATA LINEAGE AND IMPACT ANALYSIS

As your project grows, the increasing complexity can make it hard to keep track of all objects and their dependencies. To help you with this, TX DWA contains two tracing features, data lineage and impact analysis, that you can use on most objects in TX DWA:

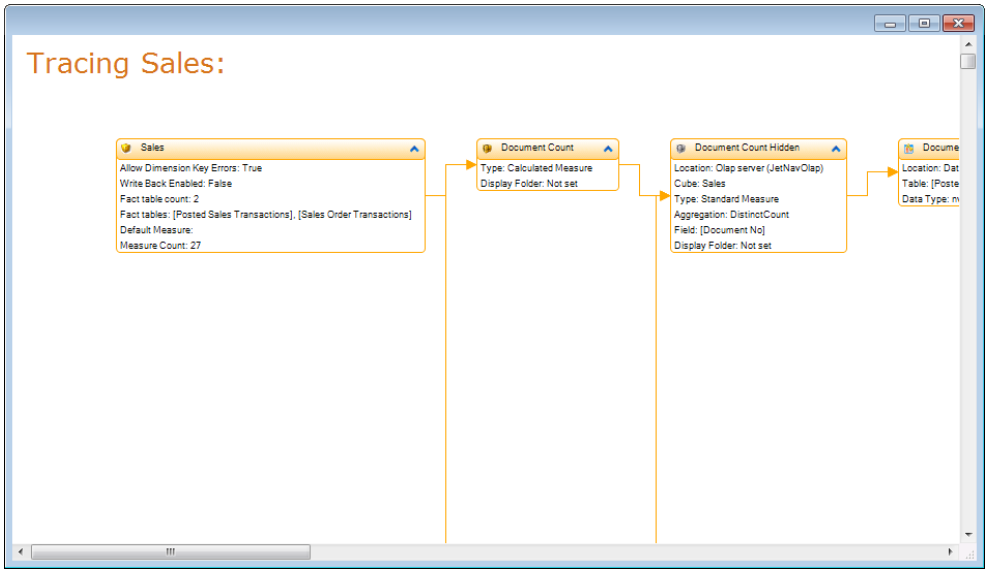
- Data warehouses
- Business units
- Staging databases
- Data sources
- Tables
- Fields
- OLAP servers
- Cubes
- Dimensions

The purpose of data lineage is to show you where the object in question gets its data, while the impact analysis feature shows you where the data is used.

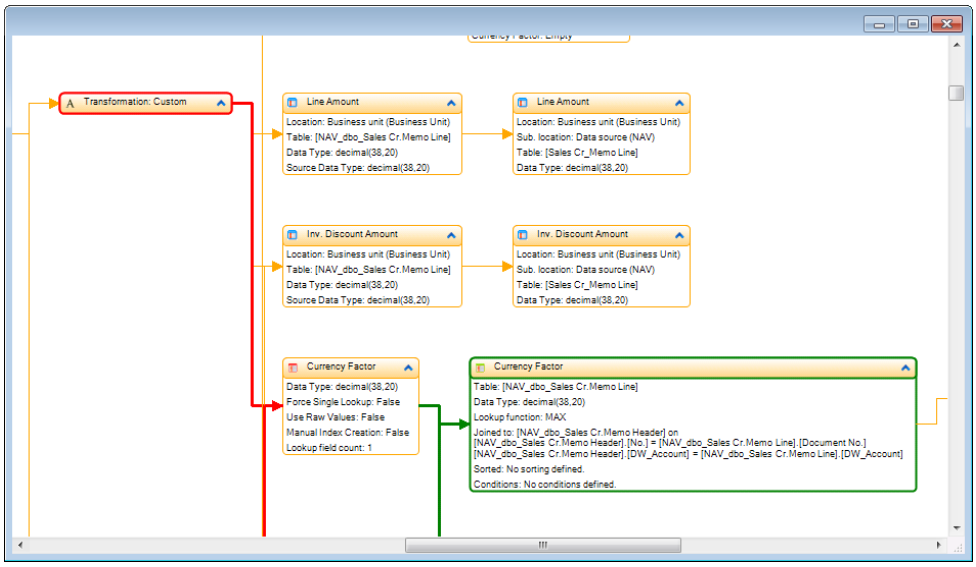
## TRACING AN OBJECT WITH DATA LINEAGE OR IMPACT ANALYSIS

To use the tracing features on an object, follow the steps below.

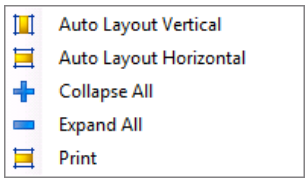
1. Right-click the object, click **Tracing** and click **Data lineage** or **Impact analysis**, depending on what you want to see. The **Tracing** window opens with a tracing diagram:



2. Click an object to see what objects deliver data to the object (marked in green - and what objects it delivers data to (marked in red).




3. Click and hold on a object to move it around in the diagram and release the mouse button to place it.
4. Right-click the background in the **Tracing** window to bring up a menu.




5. Click **Auto Layout Vertical** or **Auto Layout Horizontal** to make TX DWA reorder the objects.
6. Click **Collapse All** or **Expand All** to either collapse the objects to create a better overview, or expand the objects to see more details.
7. Click **Print** to bring up a **Print** window, where you can select print options and print the diagram.

# DOCUMENTATION

For audit and other purposes, you might need a printable documentation of your project. TX DWA lets you create full documentation of a data warehouse, a business unit or an entire project with a few clicks.

The documentation contains names, settings and descriptions for every object as well as the code where applicable. The documentation will always cover the present version of the project, even if it has not been deployed or even saved yet.

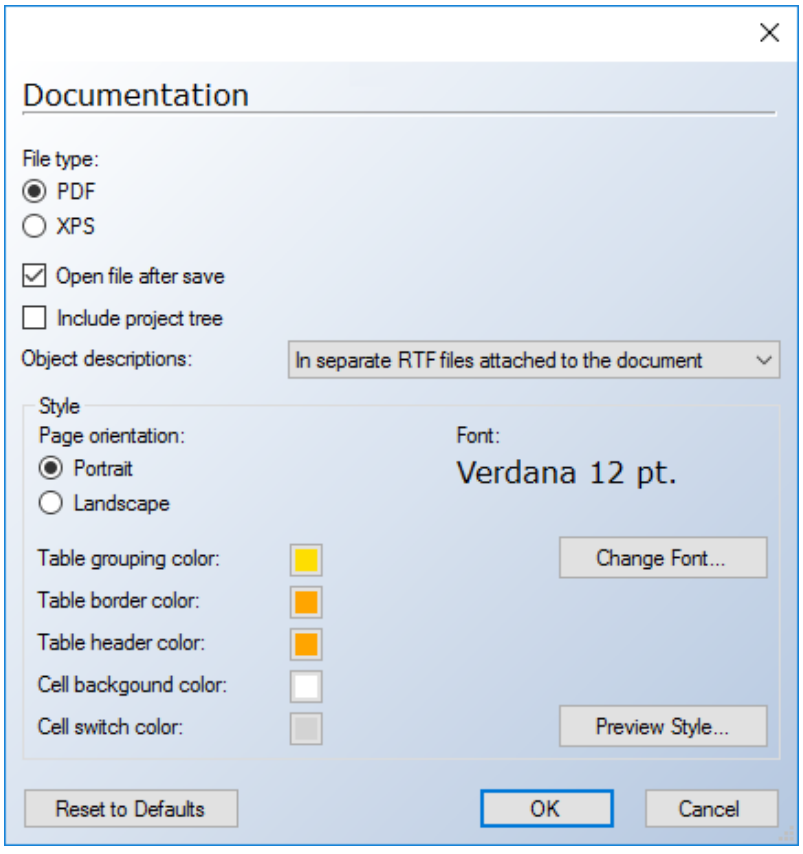
The documentation contains links between the objects to make it easier to locate the information on objects that are referenced from other objects.

## GENERATING DOCUMENTATION

To generate documentation of a part of or your entire project, follow the steps below.

1. Right click on the object you want to create documentation for and click **Documentation**. Documentation is available on the project level, data warehouses, business units, OLAP servers, data exports and Qlik models.

   The **Documentation** window appears.




2. Under **File type**, click the file format you want to use.

3. Select **Open file after save** to view the document when TX DWA has generated it. TX DWA shows the document using external viewers that needs to be present on the machine.
4. Select **Include project tree** to include a rendering of the project tree in the document.
5. In the **Object descriptions** list, you can choose how you want to use include the object descriptions. You have the following options:
    1. **In separate RTF files attached to the document**: The descriptions are attached to the document as RTF files. This option is useful if you have used rich text formatting or added pictures to your descriptions, but only available for the PDF file format.

| Table: | AggrTabRegions | Custom Description in attached file: 2.rtf |
|---|---|---|
| Icon | Σ | |
| Table Type | Aggregation | |
| Allows Nulls | Yes (As Project) | |
| Null Check Approach | Field Based (As Project) | |
| Primary Key Behavior | As Project | |
| Field Count | 6 | |
| Fields | [Name], [CountryRegionCode], [DW_Id], [DW_Batch], [DW_SourceCode], [DW_TimeStamp] | |
| Index Count | 0 | |
| Aggregations | Sum([CountryRegionCode]) | |

    2. **Only text placed in the document**: The descriptions are included in the documentation as plain text.

| Table: | AggrTabRegions |
|---|---|
| Icon | Σ |
| Table Type | Aggregation |
| Allows Nulls | Yes (As Project) |
| Null Check Approach | Field Based (As Project) |
| Primary Key Behavior | As Project |
| Field Count | 6 |
| Fields | [Name], [CountryRegionCode], [DW_Id], [DW_Batch], [DW_SourceCode], [DW_TimeStamp] |
| Index Count | 0 |
| Aggregations | Sum([CountryRegionCode]) |
| Description | This is an example description |

    3. **No descriptions**: The descriptions will not be included in the generated documentation.
6. Under **Style**, you can configure the look of the documentation. The settings are saved on a per-user basis and used for this as well as the files generated by Export Deployment Steps.
7. Under **Page Orientation**, select the page orientation you want the documentation to use.
8. Click the color preview next to the different color details to choose a color.
9. Click **Change font...** to choose a font for the documentation. a preview is displayed under **Font**.
10. Click **Preview Style...** to generate and open a sample file with the colors you have chosen.

11. Click **Reset to Defaults** to reset the style settings to their defaults
12. Click **OK**. In the window that appears, choose a file name and location for the documentation and click **Save**. The default name is the project name followed by "documentation" and a timestamp.

### GENERATE DOCUMENTATION ON REMOTE ENVIRONMENT

In addition to the local project, you can also generate documentation for a project on a remote environment in a multiple environments setup. The documentation will contain information on the last deployed version on the environment.

- To generate documentation for a project on a remote enviroment, right click on the project, click on Multiple Environment Transfer, right click the remote environment and click **Project Documentation**.

### ADDING A DESCRIPTION TO AN OBJECT

You can add an description to most objects in your project. This description can then be included in the documentation you generate. To add a description to an object

- Right click the object and click **Description**.

Per default, objects with a description have a bolded name in the project tree. However, this can be disabled if you find it distracting. To enable or disable the display of described objects in a bold font

- On the **Tools** tab in the **Application Settings** group, click **Window and Menu Settings** and remove the check mark next to **Show descriptions**.

# MOVING AND RELATING DATA

At its core, a TX DWA project is about moving data from one place to another. Often, you simply move data from a few data sources to a data warehouse via a staging database in a business unit. You can also consolidate data from multiple business units in one data warehouse or have multiple data warehouses and move data between those.

In this context, moving data could of cause also be described as copying since no data is deleted from the source. No data is moved before you deploy and execute the tables.

Once the data has been moved, you can set up relations between the tables. Among other things, specifying the relations allows TX DWA to do a referential integrity check during execution of the project.

Moving and relating data makes heavy use of drag-and-drop. Since the project tree can become quite large, it can be useful to open part of the project in a new window.

- To open part of the project in a new window, right click a object and click **Open in new window**.

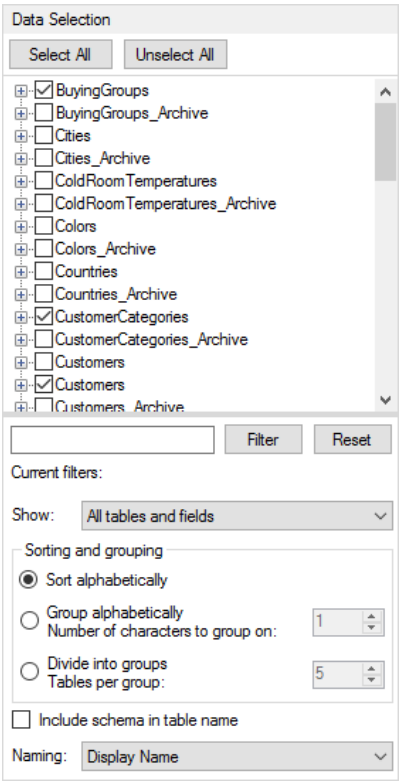
## MOVING DATA TO A STAGING DATABASE

To get data into your staging database, you first have to connect at least one data source or adapter data source. See Connecting to Data Sources and Connecting with Application Adapters for more information.

### SELECTING DATA FROM A DATA SOURCE

To add tables to the staging database, you select the tables and fields that you want to extract from the data source.

1. Right click the data source and click **Read objects from data source**. This will get content for the list of tables in the **Data Selection** pane in the right-hand side of the window.




2. In the bottom of the pane, you will find filter and sorting options:
    - To filter the list by specific terms, type a word in the box and click **Filter**. You can add more than one filter term. The terms you have entered are added to the **Current filters** list. Click **Reset** to remove the current filter.
    - In the **Show** list, you have the following filter options:
        - **All tables and fields:** No filter is applied.
        - **Selected tables and fields:** Only the tables and fields already selected from the data source are displayed. This is useful if you need to find an unselect a table or field.
        - **Unselected tabels and fields:** Only the tables and fields that are not already selected from the data source are displayed.
        - **Only objects in project perspective:** Only the tables included in the currently active project perspective are displayed.
    - Under **Sorting and grouping**, you have the following options:
        - **Sort alphabetically:** Sort the tables in alphabetical order.
        - **Group alphabetically:** Displays the tables in groups based on the number of characters entered in **Number of characters to group on**.
        - **Divide into groups:** Divides the tables into groups with the number of tables by group based on the number entered in **Tables per group**.
    - Select **Include schema in table name** to display schema name along with the table names.
    - In the **Naming** list, you can choose what name you want to show for the tables. You have the following options:

- **Display name**
- **Database name**

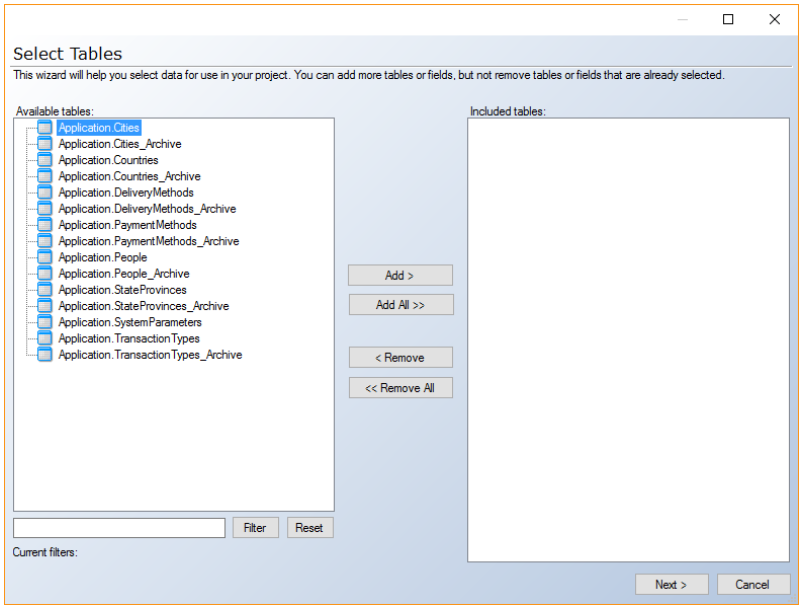
This only has effect on data sources where the two names differ.

3. Select the individual tables and fields that you want to copy from the data source into the staging database or click **Select all** to select all the tables in the source. For each table you select, a table is added to Tables under the staging database.
4. Right click on the staging database and click **Deploy and Execute** to create the table structure in the staging database and copy the data over.

### SELECTING MORE DATA FROM A DATA SOURCE

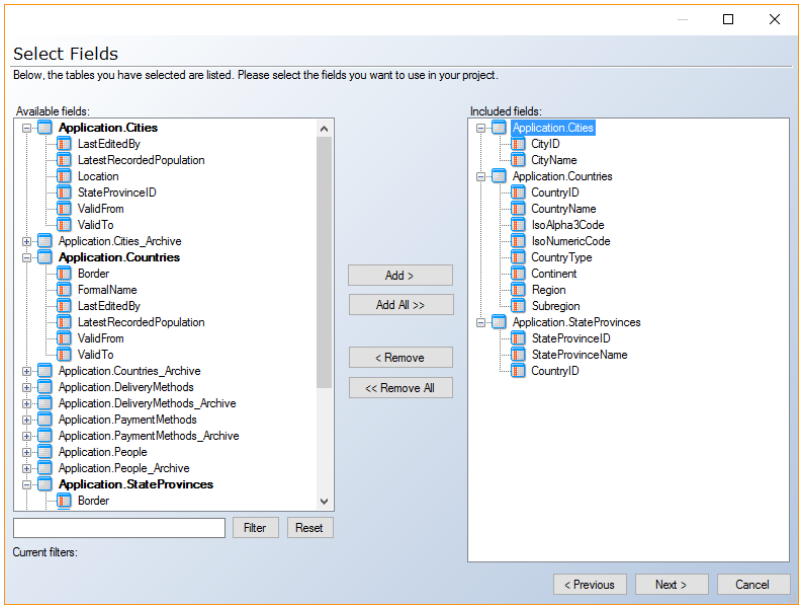
The Select Tables and Fields wizard gives you an alternative way to select more data from a data source. It is especially useful when you have a lot of tables and fields selected on a data source and need to add even more. Since the wizard only allows you to add tables and fields, you can't unselect a table or a field by accident. The select tables and fields from a data source with the wizard, follow the steps below.

1. Right click the data source and click **Read Objects from Data Source**.
2. Right click the data source, click **Automate** and click **Select Tables and Fields**. The wizard appears.




In the **Available tables** list, double-click the tables you want to select. You can also click a table and then click **Add** to add an individual table. Use the filter below the list to filter the list on table name. To add all visible tables, click **Add all**.

3. Click **Next**.

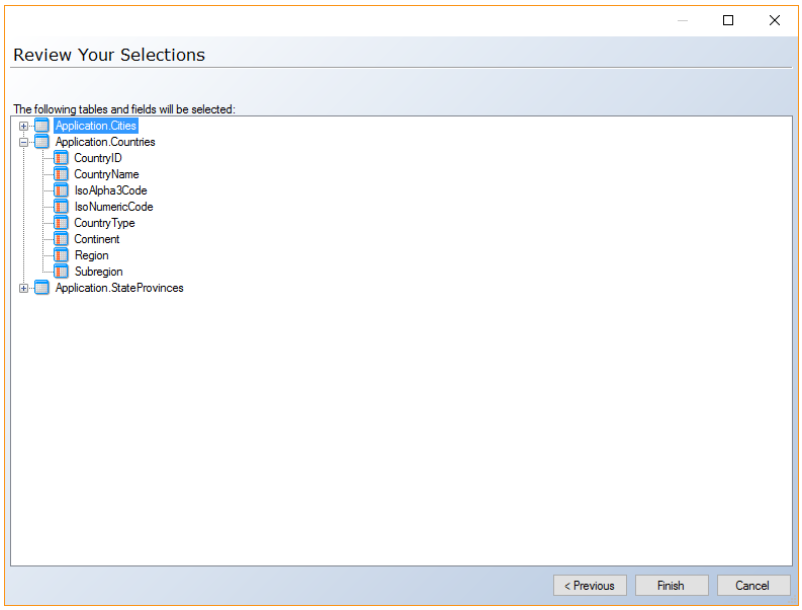



In the **Available fields** list, double-click the fields you want to select from the data source. You can also click a field and then click **Add** to add an individual field.

The tables in the Available fields list that have at least one field selected will be shown in bold.

In the **Included fields** list, any fields that are already selected on the tables you selected in the previous step are listed in grey. You cannot unselected fields on this page, only add new ones.

4. Click **Next**.




Your selections are listed for you to review. Click **Previous** to go back to an earlier

page and adjust the selections there or click **Finish** to apply the changes if they are correct.

5. Right click on the staging database and click **Deploy and Execute** to create the table structure in the staging database and copy the data over.

## MOVING DATA TO A DATA WAREHOUSE

Once you have selected the data to move from your data sources into your staging database, you can move the data further on to the data warehouse. You can also move data between two data warehouses like you move data from a staging database to a data warehouse.

Unlike in the staging database, a table or view in a data warehouse can get data from multiple source tables or views. This makes it possible to consolidate multiple tables or views from the staging database into one in the data warehouse.

You can find the mappings between source and destination tables or views on the fields on the destination table. If you expand a field, the mappings are displayed as "Copy From [schema].[table].[field]".

### MOVING A TABLE TO THE DATA WAREHOUSE

To move a table from a staging database to a data warehouse

- Drag the table from **Tables** under a staging database in a business unit to **Tables** under the data warehouse. If there is already tables in the data warehouse, a menu appears. Click **Add as new table**.

### MOVING ALL TABLES TO THE DATA WAREHOUSE

If you need to move all tables from the staging database, there is an easier way than to move all tables individually.

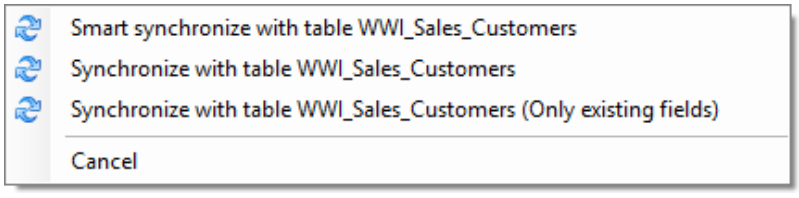
To move all tables from a staging database to a data warehouse

- Drag **Tables** from the staging database to **Tables** under the data warehouse.

### MAPPING A TABLE TO A TABLE ON THE DATA WAREHOUSE

To map a table from a staging database to an existing table in a data warehouse, follow the steps below:

1. Drag a table from a staging database to a table in the data warehouse. A menu appears.




   You have the following options:

2. Click **Smart synchronize with table [table name]** to have TX DWA compare the table you have dragged in with the other source tables on the table and add the fields from the source table that matches fields from the other source tables.
   - OR -
   Click **Synchronize with table [table name]** to add the fields of the source tableto the table. When names are identical, a mapping is added, otherwise a new field is created on the table.
   - OR -
   Click **Synchronize with table [table name] (only existing fields)** to map the fields of the source table to existing fields on the table with the same name.

#### MOVING A FIELD TO A TABLE IN THE DATA WAREHOUSE AS A NEW FIELD

In addtion to moving entire tables, you can also move single fields.

To move a field from a table in a staging database to a table on a data warehouse

- Drag the field from a staging database to a table on the data warehouse.

#### MAPPING A FIELD TO A FIELD IN THE DATA WAREHOUSE

To map a field from a table in a staging database to a table on a data warehouse

- Drag the field from a table in a staging database to a field in a table in the data warehouse.

When you map a field, the data type is automatically converted to match the data type of the data warehouse field if necessary.

## TABLE RELATIONS

For TX DWA to know how tables are related, you have to specify relations. Among other things, the relations between tables you have defined are used for a referential integrity check on execution, for the default join when you create conditional lookup fields and for relating dimensions in an OLAP cube.

Under **Relations** under each table in the project tree, you can see the relations that this particular table has to other tables.

Relations are grouped by the foreign table and the relation name defaults to "[foreign table name]_[foreign field name]".

Each relation contains one or more field relations. On each field relation, the part on the left side of the equals sign is a field on the foreign table, while the part on the right side is a field on the table that has the relation.

#### ADDING A NEW RELATION

To add a new relation, follow the steps below.

1. On the **Data** tab, navigate to the table you want to relate to another table.
2. Click on the field you want to base the relation on and then drag and drop the field on a field on another table. Note that the fields must be of the same data type to create a relation.
3. TX DWA will ask you if you want to create a relation. Click on **Yes**. The relation is created on the table on which you drop the field.

### RENAMING A RELATION

The relation name defaults to "[foreign table name]_[foreign field name]", but you can rename the relation if you want. To rename a relation, follow the steps below.

1. Locate the relation in the project tree and click on it.
2. Press **F2** to make the relation name editable.
3. Type the new name for the relation and press **Enter**.

### SETTING A DEFAULT RELATION

You can set one of the relations between two tables to be the default relation. A default relation is useful if you have more than one relation between two tables, for instance as join for lookup fields that do not have a specific join set and for auto-relation when you add a dimension to an OLAP cube.

- To set a relation as the default relation, right click the relation and click **Set as Default Relation**.

### SETTING THE SEVERITY OF VIOLATING THE REFERENTIAL INTEGRITY CHECK

TX DWA users the relations you have defined to perform a referential integrity check, or foreign key constraint check, when the table is executed. TX DWA checks the value of any field that are part of a relation to see if the value exists in the related field in the related table. If not, TX DWA considers the record invalid.

For instance, a Sales Order table might contain a Customer ID field that is related to a Customer ID field in a Customer table. If a specific sales order record contains a Customer ID that is not in the Customer table, TX DWA considers that record to be invalid.

You can define the severity of the violation on each relation. To set the severity of a violations on a particular relation, follow the steps below.

1. Locate the relation in the project tree.
2. Right click the relation, click **Relation Type** and click your preferred type. You have the following options:
    - **Error**: TX DWA moves the invalid record to the error table. This means data will be missing from the valid instance of the table.
    - **Error with physical relation**: The relation is stored in the database for other database tools to see. The behavior is otherwise the same as error. Note that the

table needs to have a primary key and a unique index set. If Index Automation is disabled on the table, you will have to create the index yourself.

- **Warning**: TX DWA copies the invalid record to the warnings table and the valid instance of the table. You will not be missing data from the valid table. However, you might need to handle the violated rule in some way.
- **Relation only**: TX DWA ignores any violations of the check.

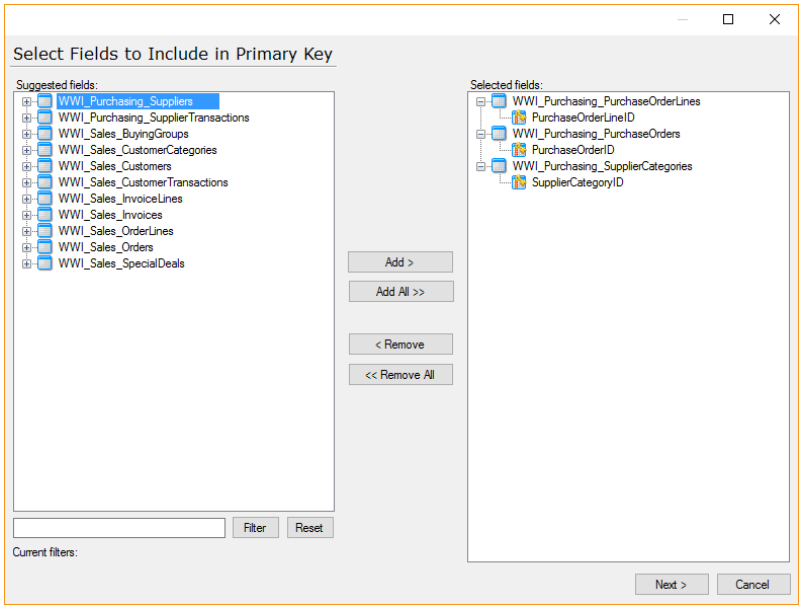
## CONSTRAINT SUGGESTIONS

In addtion to the foreign constraint check, TX DWA can perform a primary key constraint check on execution. Both checks are based on the relations and primary keys you have added to the project. Sometimes information about the tables' primary keys and relations can be found in the data source and read by TX DWA. The Constraint Suggestions wizard surfaces these suggestions and allows you to implement the ones you want to use.

### ADDING RELATIONS AND PRIMARY KEYS FOR MULTIPLE TABLES

To use the Constraint Suggestions wizard to add relations and primary keys to all tables in a staging database or data source, follow the steps below.

1. Right click the staging database or data source, click **Automate** and click **Add Suggested Constraints**. The wizard appears on the page where you can select what fields to include in the primary key on the respective tables.




In the **Suggested fields** list, double-click the fields you want to use as primary keys. You can also click a field and then click **Add** to add an individual field. To add all visible fields, click **Add all**. Use the filter below the list to filter the list on field name. The following wildcards are supported:

- **%**: Any string of zero or more characters.
- **_**: Any single character.
- **[ ]**: Any single character within the specified range ([a-f]) or set ([abcdef]).

- **[^]**: Any single character not within the specified range ([^a-f]) or set ([^abcdef]).

2. Click **Next**. In the **Suggested fields** list, double-click the fields you want to create a relation from. You can also click a field and then click **Add** to select an individual field. In the parentesis after the field name, you can see what table and field the relation will be to. The tables in the Available fields list that have at least one field select will be shown in bold.
3. Click **Next**. Your selections are listed for you to review. Click **Previous** to go back to an earlier page and adjust the selections there or click **Finish** to apply the changes if they are correct.

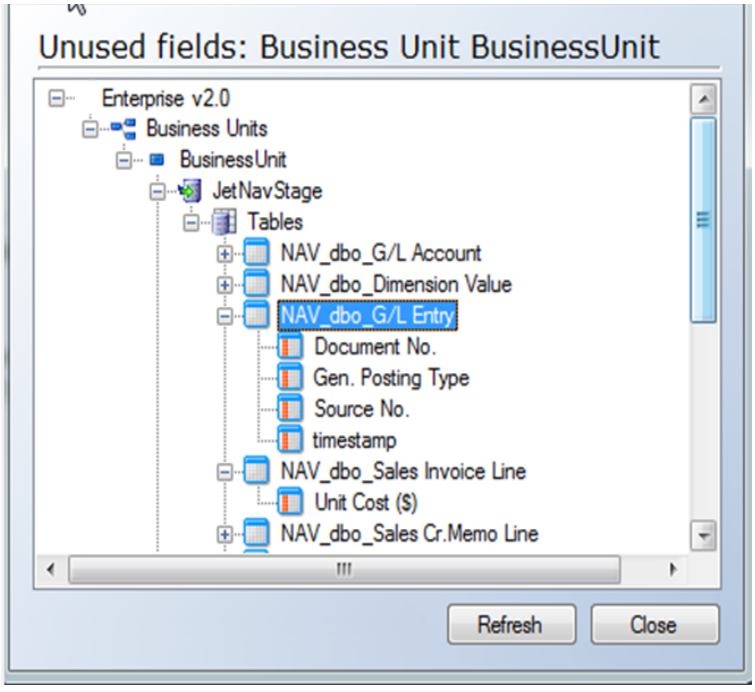
## UNUSED FIELDS

TX DWA has 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.

### DISPLAYING UNUSED FIELDS

To display unused fields

- Right-click the business unit or the data warehouse and click **Find Unused Fields**. The list of unused fields appear.




In the staging database, the list 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, the list 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

# SELECTING, VALIDATING AND TRANSFORMING DATA

Selecting the right data from the source, validating it and transforming the data if needed are central parts of the data warehouse process.

In TX DWA you specify data selection rules to ensure that only the data needed for your analysis is extracted from the data source to the staging database.

On the staging database, you perform data cleansing by applying validation and transformation rules to the data. This ensures that only valid data is loaded into the data warehouse.

However, you can also apply selection, 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.

## OPERATORS FOR SELECTING AND VALIDATING DATA

When defining a data selection or validation rule, you can use the operators listed below.

Values must be either integers or letters. You can also specify a list of values by entering comma-separated values.

| Operator | Definition |
|---|---|
| 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. |

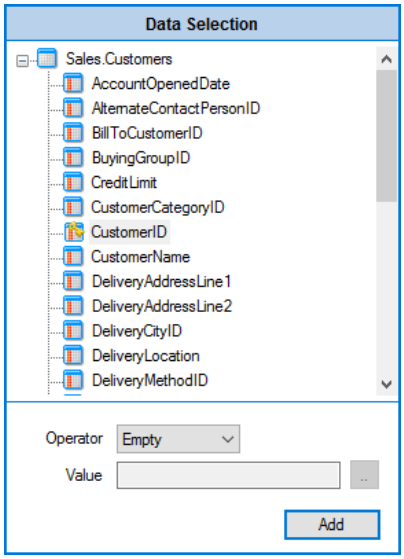
## DATA SELECTION RULES

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. You can also define data selection rules on data warehouse tables to further control the data loaded into the data warehouse.

You can add usage conditions to selection rules based on project parameters. This enables you to e.g. load less data in a development environment than in the production enviroment.

### ADDING A DATA SELECTION RULE

1. In the project tree, expand **Business Units**, expand the business unit that contains the data source you want to apply the selection rule to, expand **Data Sources** and then expand the relevant data source.
   - OR -
   In the project tree, expand **Data Warehouse**, expand the data warehouse that contains the table you want to apply the selection rule to and expand **Tables**.
   - OR -
   On the Qlik tab, expand the Qlik model that contains the table you want to apply the selection rule to and expand **Tables**. If the table is concatenated, expand **Mappings** as well.
2. Right-click the table you want to add the selection rule to and click **Add Data Selection Rule**. The **Data Selection** pane appears in the right hand side of the window.



3. Click the field you want to use in the selection rule.
4. In the **Operator** list, click the operator you want to use. See Operators for Selecting and Validating Data.
5. If applicable, type a value for the operator in the **Value**box.

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

### ADDING A USAGE CONDITION TO A SELECTION RULE

To add a usage condition to a selection rule based on a project variable, follow the steps below.

1. Right click a selection rule and click **Add Usage Condition**. The **Usage Condition** panel is displayed in the right hand side of the appli-cation window.
2. In the **Usage Condition** panel, click the variable you want to use.
3. In the **Operator** list, click the operator you want to use. You have the following options:
   - Equal
   - NotEqual
   - GreaterThan
   - LessThan
   - GreaterEqual (Greater than or Equal to)
   - LessEqual (Less than or Equal to)
4. In the **Comparer** list, click the general data type of the variable, which TX DWA will use in the comparison. You have the following options:
   - String
   - Date
   - Numeric
5. Type the value you want to compare the parameter with in the **Value** box.
6. Click **Add** to add the usage condition.

For more information about project parameters, see Project Variables.

## DATA VALIDATION RULES

Validation rules ensure a high level of accuracy and reliability of the data in the data ware-house 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.

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

| Severity | Definition |
|---|---|
| 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. |

### ADDING DATA VALIDATION RULES

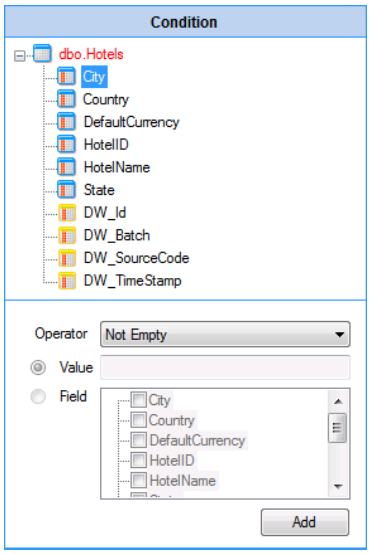
You can add any number of validation rules to a field.

1. In the project tree, expand **Business Units**, expand the business unit that contains the data source you want to apply the validation rule to, expand **Data Sources**, expand the relevant data source and expand the relevant table.
   - OR -
   In the project tree, expand **Data Warehouse**, expand the data warehouse that contains the table you want to apply the selection rule to, expand **Tables** and expand the relevant table.
2. Right-click the field, you want to apply the validation rule to, and click **Field Validations**. The **Field Validations** pane appears in the right-hand side of the window.
3. Click the field you want to use in the validation rule.
4. In the **Operator** list, click the operator you want to use. See Operators for Selecting and Validating Data.
5. If applicate, type a value for the operator in the **Value** box.
6. Click **Error** to specify that as the severity level or leave it at **Warning**.
7. Click **Add** to add the rule.

### ADDING CONDITIONS

You can add any number of conditions to your validation rules. Follow the steps below to add a validation rule.

1. Locate the selection rule you want to modify.
2. Right-click the rule and then click **Add Condition**. The **Condition** pane is displayed.




3. In the **Operator** list, click the operator you want to use. See Operators for Selecting and Validating Data.
4. In the **Value** field, type the value you want to use in the comparison.
   - OR -
   Click **Field** and click the field, you want to use in the comparison.
5. Click **Add** to add the condition to the rule.

The condition is displayed in the project tree below the validation rule or transformation rule it belongs to.

### TO VIEW VALIDATION ERRORS OR WARNINGS

1. Click the **Errors** or the **Warnings** tab.
2. In the **Database** list, click the database that contains the table you want to view errors or warnings for.
3. In the **Table** list, click the relevant table. The **No. of rows** box displays the number of errors or warnings in the table and the rows that violate the rules are displayed in the pane below.
4. Click any row to display the error or warning message in the **Error Message** or **Warning Message** box.

## DATA TRANSFORMATION

Fields transformations lets you modify existing data in a number of ways. You can, for example, easily reverse the sign of numeric values, trim fields or return a specified number of characters from the original field value.

### ADDING FIELD TRANSFORMATION RULES

1. In the project tree, expand **Business Units**, expand the business unit that contains the data source you want to apply the validation rule to, expand **Data Sources**, expand the relevant data source and expand the relevant table.
   - OR -
   In the project tree, expand **Data Warehouse**, expand the data warehouse that contains the table you want to apply the selection rule to, expand **Tables** and expand the relevant table.
2. Right-click the field, you want to add a transformation rule to, and then click **Field Transformations**. The **Field Transformation** pane appears. In the pane, click the field you want to add a transformation to.
3. In the **Operator** list, click the operator, you want to use, and then click **Add**.

| Operator | Description |
|---|---|
| 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 |

4. If you have selected **First** or **Last** as the operator, enter how many characters you want to include in the **Length** field.

### ADDING CONDITIONS

You can add conditions to transformation rules in the same way that you add conditions to validation rules. See " Adding Conditions" on page 121

# PREVIEWING DATA

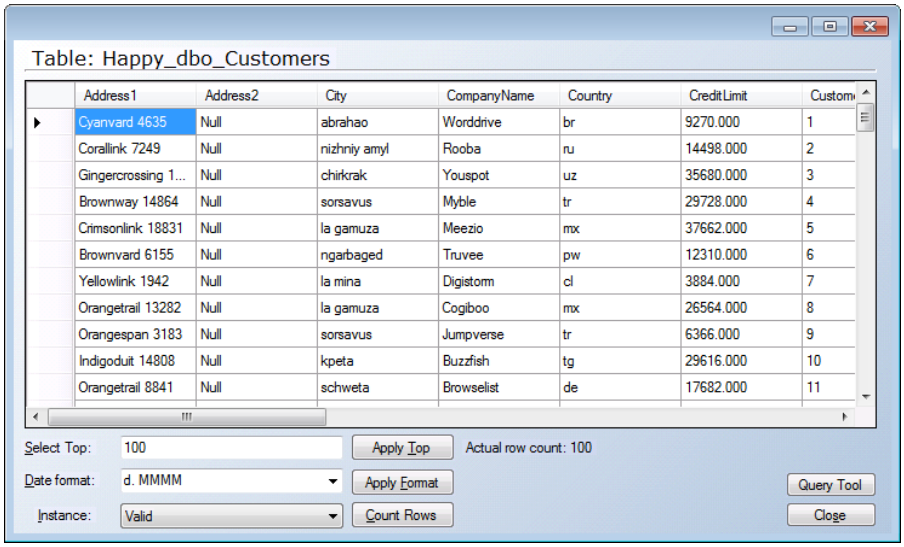
During development, it is often useful to be able to see what data is present in different tables. For instance, you might want to check that a transformation works as intended. TX DWA provides two different ways of viewing the content of a table.

## PREVIEW TABLE

The Preview Table feature gives you a basic overview of the content of a table. To preview the content of a table, follow the step below.

1. On the **Data** tab, expand the relevant business unit, expand **Data Sources**, expand the data source and right-click the table.
   - OR -
   On the **Data** tab, expand **Data Warehouses**, expand the relevant data warehouse, expand **Tables** and right-click the table.

   The Preview Table window opens.




2. You have a number of options for previewing the data:
   - In **Select Top**, type the number of rows you want to fetch and display. Click **Apply Top** to apply the setting. Please notice that the select top is applied before any sorting of data.
   - In the **Date format** list, click the date format you want to use for dates in the data. Click **Apply Format** to apply the settings.
   - In the **Instance** list, click the instance of the table you want to preview.
   - Click **Count Rows** to
3. When you are done, click **Close** to close the window.

## QUERY TOOL

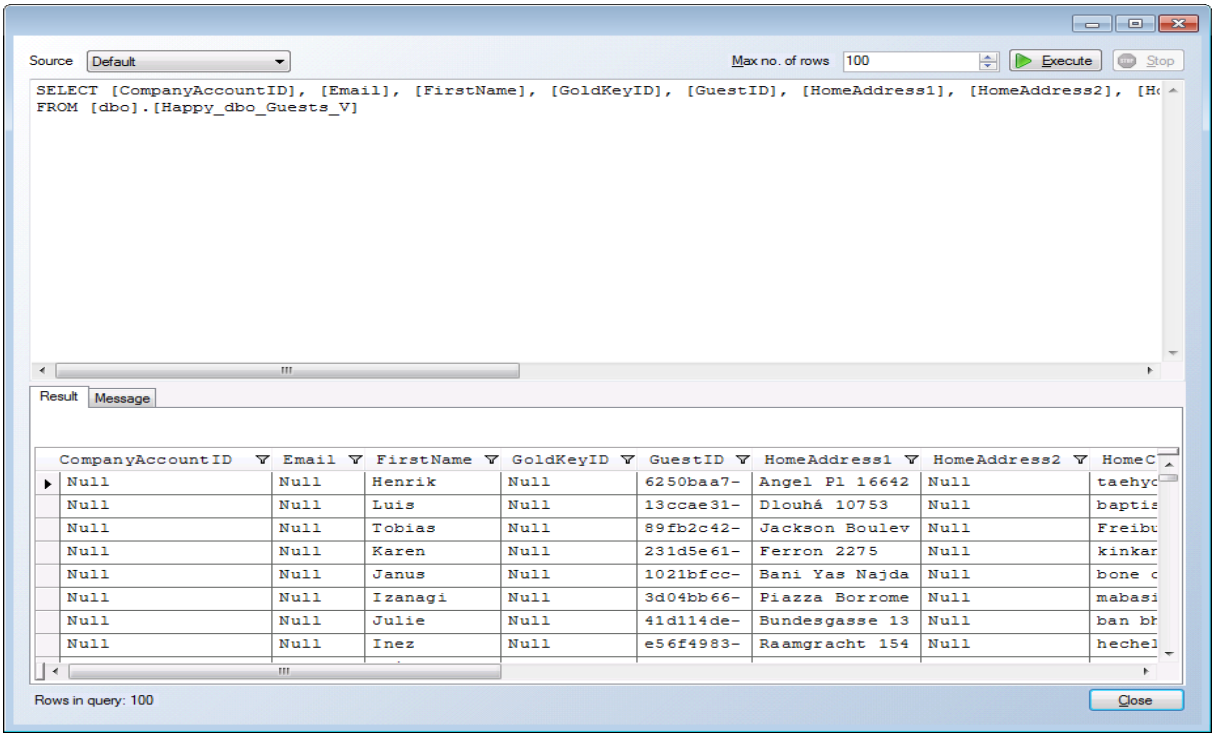
The Query Tool is a powerful supplement to the Preview Table feature that gives you more flexibility in exploring the content of a table. You can execute any SQL query to see the data you want to see the way you want to see it.

### OPENING THE QUERY TOOL

You can open the Query Tool in three different ways.

- Right click a table, click **Preview Table** and click **Query Tool** in the **Preview Table** window.
- Right click a table, click **Advanced** and click **Query Tool**.
- Click table and press **F8** on your keyboard.

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 want 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 DWA.
   Adjust **Max no. of rows** to the maximum number of rows you want to have returned.

4. Click **Execute**
   - OR -
   Press **F5** on your keyboard.
5. If you want to stop the query before it completes, click **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 want 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.

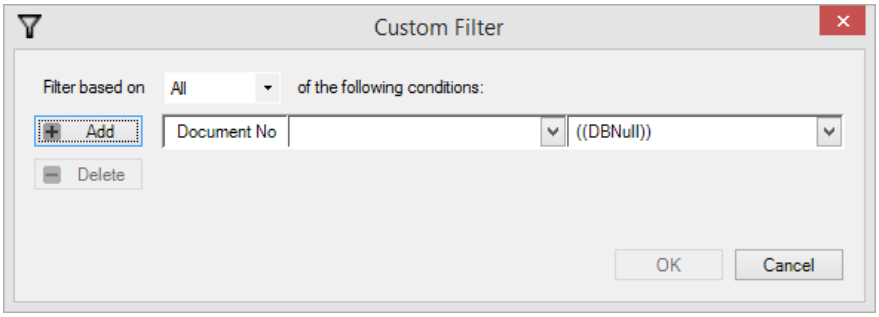
**Note:** Only the rows returned by the query are available for sorting and filtering in the **Results** tab. If you want to sort or filter all the rows in a table, the most efficient way is to include the conditions in the query, e.g. by using "order by" or "where" clauses. Fetching thousands of rows and sorting them using the tools provided in the **Results** 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 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 want 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.

# TABLES

Most tables in TX DWA are brought in from a data source and moved into the date warehouse via a staging database. See Moving Tables from a Staging Database for more information.

In addition to these tables TX DWA has six other table types:

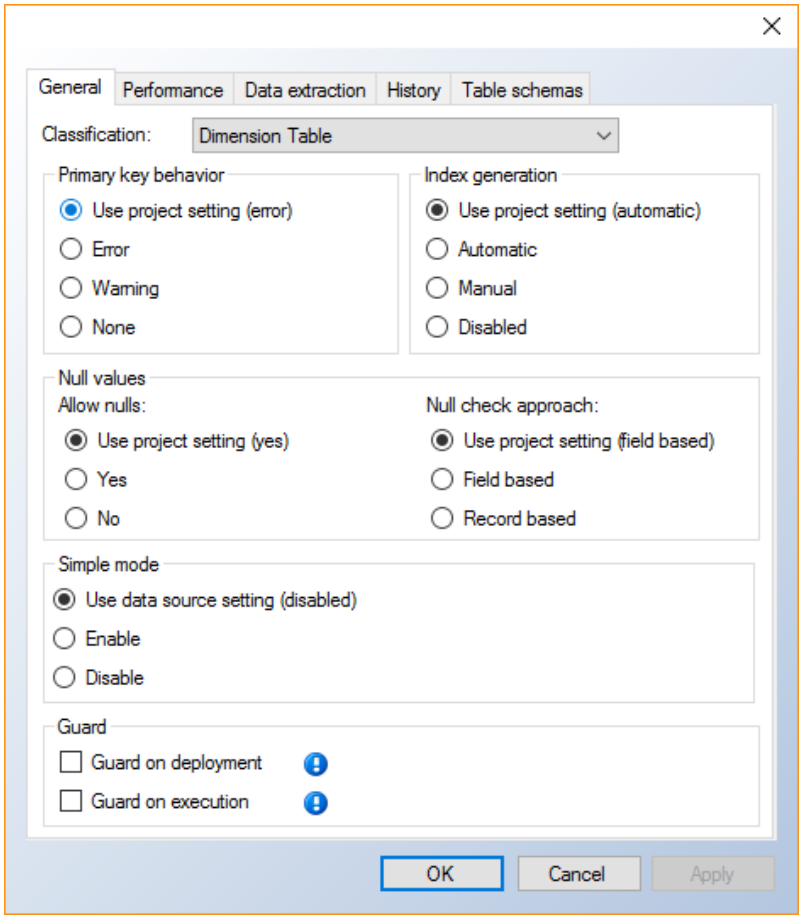
- **Custom tables:** An empty table than can be populated with custom fields.
- **Date tables:** Usually used for creating time dimensions on OLAP cubes.
- **Hierarchy tables:** Used to create special reporting structures based on the other tables in the data warehouse, especially for financial reporting.
- **Junk dimension tables:** A concept in dimensional modeling, junk dimension tables replace multiple fields in a table with a field referencing a row in another table containing the same combination of fields.
- **Aggregate tables:** Creates version of an ordinary table with aggregated data.
- **External tables:** Tables that are "sideloaded" into a data warehouse from another SQL Server database.

## CHANGING SETTINGS FOR A TABLE

Most settings for tables are consolidated in the Table Settings window.

To open the table settings window

- Right click a table and click **Table settings**.




Depending on the table type, not all settings are available. For instance, incremental load does not make sense for date and hierarchy tables, so incremental load settings are disabled for these table types.

### GUARDING A TABLE

Guarding a table tells TX DWA to skip the table on execution or deployment. This is useful if, for instance, the table contains old data from a legacy system that is no longer running.

To guard a table

- Right click the table, click **Table settings** and then select **Guard on deployment** and/or **Guard on execution** under **Guard**.

### SIMPLE MODE

Simple mode is a setting on tables on business units aimed at maximizing performance when you need to copy large amounts of data into a staging database to create an exact copy. See Simple Mode for more information.

Per default, a table inherits the simple mode setting from the data source which in turn inherits the setting from the business unit.

To enable simple mode

- Right click the table, click **Table settings** and click **Enable** under **Simple Mode**.

### ENABLING BATCH DATA CLEANSING TO IMPROVE DATA CLEANSING PERFORMANCE

You can choose to split the INSERT statement up in batches during data cleansing, i.e. when copying data from the transformation view for table to the valid table. This saves log space on the SQL Server which gives you better performance on large tables with 100,000s or millions of rows.

To enable batch data cleansing, follow the steps below.

1. On the Data tab, in a data warehouse, right click the table you want to use batch data cleansing on and click **Table settings**.
2. Click the **Performance** tab and select **Enable batch data cleansing**.
3. (Optional) Enter the number of records you would like each batch to contain in **Batch size**. The default is 100,000.
4. Click **OK**.

## CUSTOM TABLES

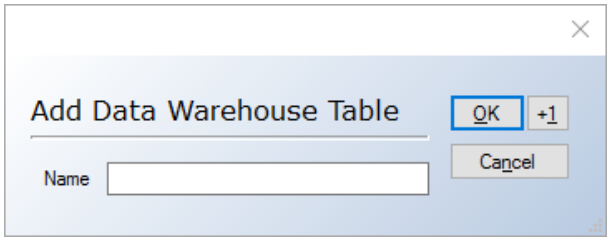
Custom tables are basic tables that you can add to the data warehouse or staging database. While they do not initially contain any fields except the standard system fields, you can add new fields to custom tables.

With custom tables, you can, for instance, build you data warehouse first and then map the data in from the data sources.

### ADDING A CUSTOM TABLE TO THE DATA WAREHOUSE

To add a custom table to a data warehouse, follow the steps below.

1. Expand **Data Warehouses**, and then expand the preferred data warehouse.
2. Right click on **Tables** and click **Add Table**. The **Add Data Warehouse Table window** appears.




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

### ADDING A CUSTOM TABLE TO THE STAGING DATABASE

To add a custom table to a staging database, follow the steps below.

1. Expand **Business Units** and expand the preferred business unit and staging database.
2. Right click **Tables** and click **Add Custom Table**. The **Add Data Warehouse Table window** appears.
3. In the **Name** box, type a name for the table and then click **OK**.

## DATE TABLES

You will typically use date tables when you build OLAP cubes on top of the data warehouse. Most often, the cubes you create will contain a date dimension to make it possible to analyze data over time. For example, you may report data on a daily, weekly, or monthly basis. In TX DWA you use date tables, stored in the data warehouse, as the basis for your time dimensions.

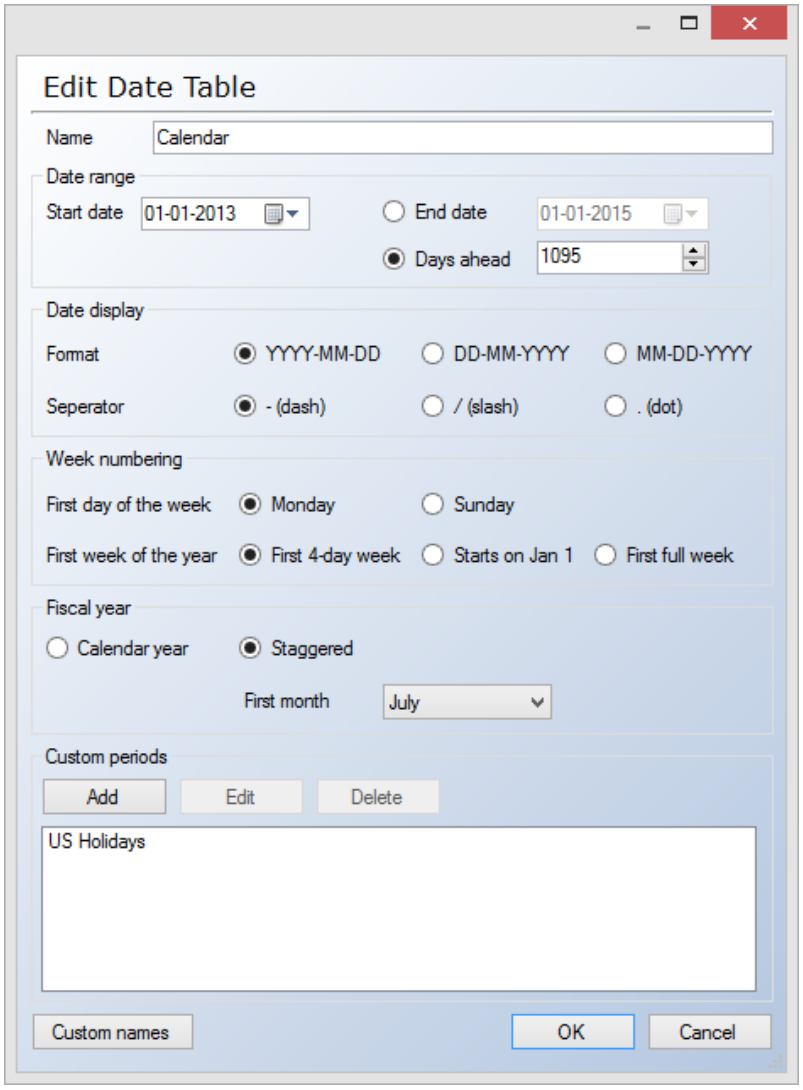
In addition to day of month, day of quarter, week of year and other ordinary information about each date, date tables also contain indexes. An index is a column in the table that tells you something about the date's relation to the current date. Data tables contain year, quarter, month and week indexes. All index values for today's date are 0, while for instance, any day last year would have a year index of -1. This makes it trivial to compare e.g. the same month, day or quarter across years.

Data tables can also contain custom periods, special periods of time, for instance holidays or yearly sales campaigns, that enables you to easily track data across these reoccurring time periods.

### ADDING A DATE TABLE

To add a date table, follow the steps below.

1. On the **Data** tab, on a data warehouse, right click **Tables** and click **Add Date Table**. The **Add Date Table** window opens.




2. In the **Name** box, type a name for the table.
3. Select a **Date range** by entering a **Start date** and an **End date**. Instead of entering an end date, you can enter a number of days to add to the current date in **Days ahead**. This way, your date table will effectively never end.
4. Under **Date display**, select the **Format** you want to use for dates. You have the fol-lowing options:
    - YYYY-MM-DD
    - DD-MM-YYYY
    - MM-DD-YYYY
5. Select which **Separator** to use in the format you chose. You have the following options:
    - - (dash)
    - / (slash)
    - . (dot)
6. Under **Week numbering**, click the **First day of the week**. You have the fol-lowing options:

- Sunday
- Monday

7. Select the how to define the First week of the year. You have the following op-tions:
   - First 4-day week (following the ISO 8601 standard, common in Europe)
   - Starts on Jan 1 (common in North American)
   - First full week
8. Under Fiscal year, click **Staggered** to use a staggered fiscal year and click the first month of the staggered fiscal year in the **First month** list.
9. (Optional) Click **Add** under **Custom periods** if you want to add a custom period. In the custom period window that opens, you can type a **Name** for the custom period and **Name**, **Start date** and **End date** for the included periods. You can also import and export custom periods by clicking **Import** and **Export** respectively. Click **OK** when you are done.
10. (Optional) Click **Custom names** if you want to change the names used for days, quarters and months. In the **Date Table Custom Names** window that opens, you can type the names you want to use. The default is derived from the regional settings on the deploying machine. Click **OK** when you are done.
11. Click **OK** to add the date table.

## HIERARCHY TABLES

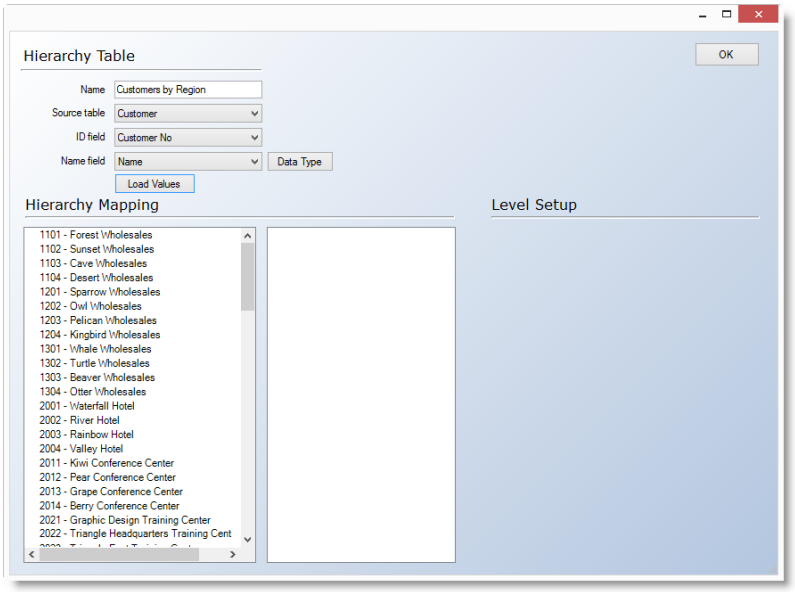
A hierarchy table is used to select data from a table and create a new reporting structure which is different from the structure in the data source. You will typically use a hierarchy table for financial reporting where you want to consolidate data from a number of different accounts, such as ledger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

A hierarchy table is used in conjunction with a parent-child dimension. First, you create the hierarchy table and specify the contents of the table.Then you 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.

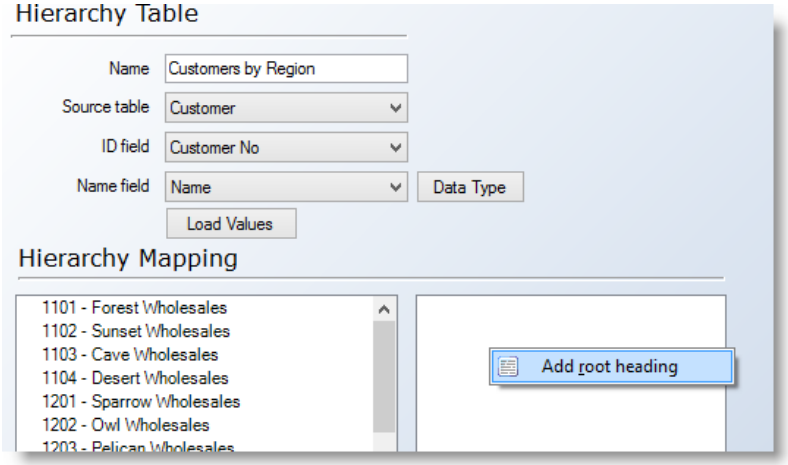
### ADDING A HIERARCHY TABLE

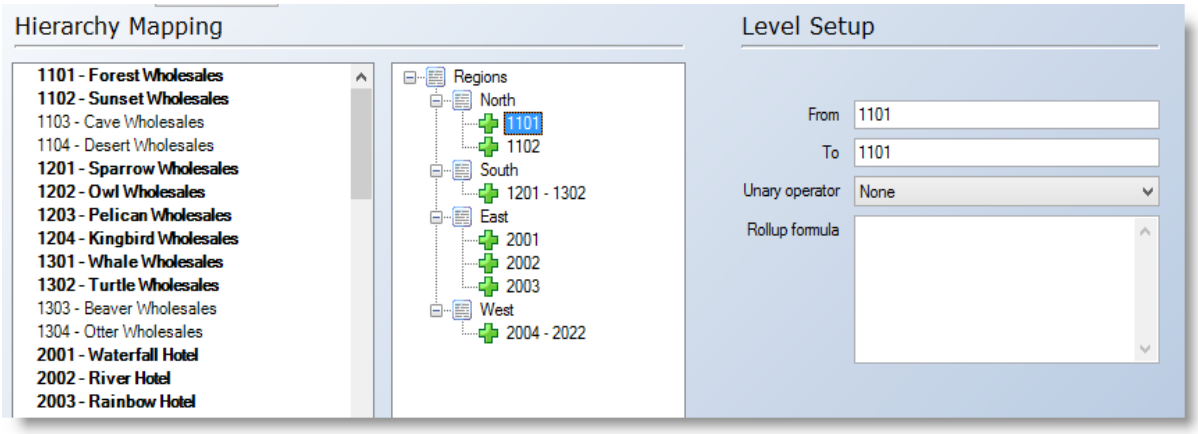
1. On the **Data** tab, expand **Data Warehouses**, and expand the relevant data warehouse.

2. Right-click **Tables** and then click **Add Hierarchy Table**. The **Hierachy Table** window opens:




3. In the **Name** box, type a name for the table.
4. In the **Source table** list, click the table containing the desired data.
5. In the **ID field**, click the field that identifies the individual entries in the table; for example, the customer number. 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.
7. You can now create the report structure. The structure you create corresponds to the structure of the report that is displayed to the end-user.
8. Right-click in the **Blank** pane, and then select **Add Root Heading**. The root headings become root nodes in the final report.




9. In **Level Setup**, type a name for the heading in the **Name** field.
10. Right-click the root heading, and select **Add Sub Heading** to add a child node to the structure.
11. In the **Name** box, type a name for the child node.
12. In the **Unary Operator** list, you 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:

| Operator | Definition |
|---|---|
| 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 8-12 for all root headings and subheadings you want to add.

13. Click and hold an entry in **Hierarchy Mapping** , and drag it to the preferred sub-heading in the **Blank** pane. Alternatively, you can specify a range of entries by typing the relevant numbers in **From** and **To**.
14. To exclude an entry from a given range, right-click the relevant subheading, click **Add Exclude**, and then specify a range by typing the relevant numbers in **From** and **To**. Alternatively, right-click the specific entry and select **Change to Exclude**.
15. 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. Type a formula in the **Roll-up formula**. Formulas are written in MDX.
16. 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.

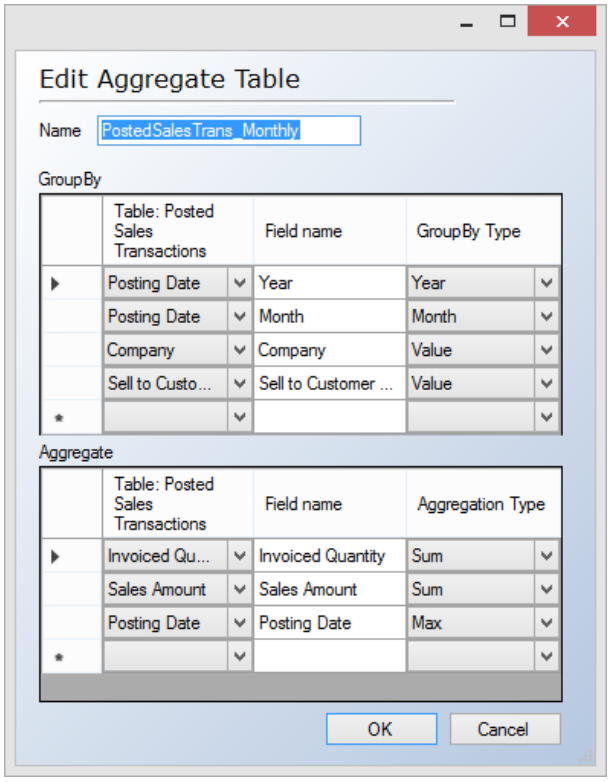
## AGGREGATE TABLES

An aggregated table is an aggregated version of another table in you project. Often, you will not need the transactional level in financial or sales reports, but only data grouped by business unit or sales team. This makes the aggregated tables feature very useful if you are doing reporting directly from you data warehouse as opposed to using, for instance, OLAP cubes.

### ADDING AN AGGREGATED TABLE

To add an aggregated table, follow the steps below.

1. On the **Data** tab, under **Tables** in a data warehouse, right click the table, you want to add an aggregated version of, click **Advanced** and click **Add Aggregate Table**. The **Add Aggregrate Data Table** window opens.




2. Under **GroupBy**, you can choose what columns on the table the aggregated table should use for grouping the aggregated data. Click the column you want to use in the empty list under **Table: [table name]**. Type a name for the field in **Field** name. In the column you have chosen contain date values, click the list under **GroupBy Type** to adjust the granularity of the grouping. You can choose second, minute, hour and all the way up to year. Note that you can use the same date column multiple times with different **GroupBy** types. For other data types, the **GroupBy** type will always be **Value**.
3. Under **Aggregate**, you can choose what columns from the table you want to have aggregated. Click the column you want to use in the empty list under **Table: [table name]**. Type a name for the field in **Field** name. Click the list in the **Aggregation Type** column and click the method you want to use for calculating the aggregation. You have the following options:
    - **Min**: The lowest value of the field in question.
    - **Max**: The highest value of the field in question.
    - **Count**: The number of rows.
    - **Count_Big**: Same as count, but is able to count higher than 2^31, be-cause it uses the bigint data type instead of the int data type.
    - **DistinctCount**: The number of unique values in the field.
    - **Sum**: The sum of all row values.
    - **Average**: The average of all row values.
4. Click **OK**. The aggregated table is added under **Tables** and can be recognized by its yellow icon.

## JUNK DIMENSION TABLES

A junk dimension is a concept in dimensional modeling. It 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 table 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.

Multiple tables can utilize the same junk dimension table.

### ADDING A JUNK DIMENSION TABLE

Junk dimensions can be added to tables in both data warehouses and staging databases. To add Junk Dimension table for a table, follow the steps below.

1. Right click on a table, click **Advanced** and click **Add Junk Dimension Table**.
2. In the window that appears, select the fields you want to include in you junk dimension table and click **OK**. A new window appears that allow you to customize you junk dimension table.
3. Enter a name for the table in the **Name** box or leave the default ("Dim[table name]Info").
4. (Optional) Click on a table in the **Available tables** list and click add to add it to the junk dimension table. A message will appear to ask you if you want to map fields automatically. The auto mapping algorithm maps a field on the table you are adding to a field on the junk dimension table if one of the following conditions is true: It has the same name as the junk dimension table field or it has the same name as another field that is mapped to the junk dimension table field.
5. Map the fields in the junk dimension table with the fields in the included tables. Each row in the table represents one field in the junk dimension table while each column represents a table.
    - To add a new field to the junk dimension table, click the empty field in the bottom row of the second column and type a name.
    - To remove a field from the junk dimension table, right click the first column of the corresponding row and click **Remove**.
    - To remove a table from the junk dimension table, right click the header row of the corresponding column - the table name - and click **Remove**.
    - To change the order the tables are loaded in, click on the header row of the corresponding column - the table name -and drag it to the desired location.
6. (Optional) In the **Hashing algorithm**list, click on the hashing algorithm you want to use for the dimension table key. See Default hashing algorithm under Creating a Project for information about the different algorithms. Junk dimensions have a special

hashing algorithm available, "Legacy integer", for compability with older versions of Analysis Services. You should avoid this algorithm if possible since it is not very safe. It is only 8 bytes which means that the risk of two different data sets giving you the same hash is much highed than with any of the other algorithms.

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.

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 in the database.

When the junk dimension table is executed, it will insert non-existing junk dimension combinations from the included table. The junk dimension table has no truncation of the raw instance of the table.

## EXTERNAL TABLES

External tables is a way to incorporate tables from an existing data warehouse into a TX DWA project. This is useful if you, for instance, have a legacy data warehouse humming along that you would like to use data from without remodeling it in TX DWA.

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. By default TX DWA will create views and read data from the external connection, but you can also chose to move the data into you TX DWA data warehouse. You can also add a custom SSIS package to the table, which can then be executed.

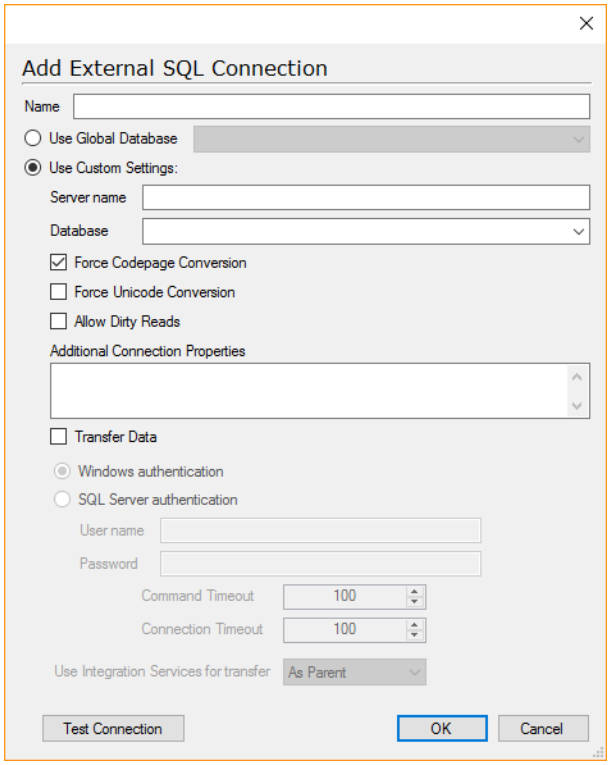
### ADDING AN EXTERNAL SQL CONNECTION

To add an external table, you first need to add an external SQL connection.

**Note:** The SQL Server needs to be on the same physical SQL Server instance as your data warehouse or on a linked SQL Server, unless you enable the Transfer Data option.

To add an external SQL connection, follow the steps below:

1. Right click your 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.
3. Click **Use Global Database** and choose a global database in the list
   - OR -
   Click **Use Custom Settings** and customize the settings:
   - Type the name of the server in the **Server Name** box.
   - Type the name of the database you want to use in the **Database** box.
   - Select **Force Codepage Conversion** to convert all fields to the collation of the data warehouse.
   - Select **Force Unicode Conversion** to declare all alphanumeric fields as **nvarchar**.
   - Select **Allow Dirty Reads** to allow reading from the source without locking the table.
   - (Optional) Enter additional connection properties in the **Additional Connection Properties** box.
   - Select **Transfer Data** to move the data from the external SQL Server to your local data warehouse, much like a regular SQL data source.
4. Click **OK** to add the source and close the window.

### ADDING AN EXTERNAL TABLE

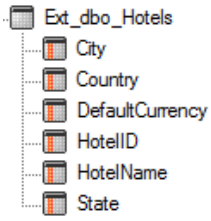
To add an external table, follow the steps below.

1. Navigate to **External SQL Connections** under your data warehouse or business unit in the project tree, right click the connection you just created and click **Read Objects from Data Source**.

2. When TX DWA 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 want 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 Qlik 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 data to an external table.

### DEPLOYING AN EXTERNAL TABLE

To deploy an external table, right click 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 DWA, TX DWA 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 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 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 want to make in the editor, save the package and close the editor.
7. While you edit the SSIS package, TX DWAdisplays the **Custom Code Editor** dialog. When you return to TX DWA, 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 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.

# FIELDS

Most fields in TX DWA products usually stems from data sources, but you can also add new fields to tables on data warehouse or staging databases. You can add custom fields - simply called "fields" on data warehouse databases - conditional lookup fields and custom hash fields.

## INCLUDING FIELDS IN THE PRIMARY KEY

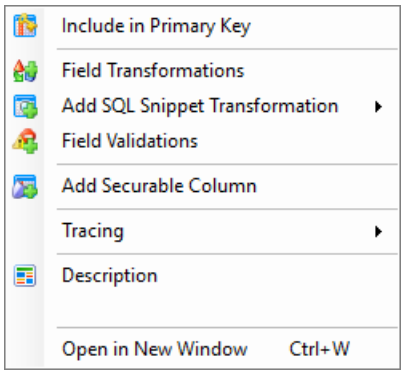
All tables in your project can have a primary key that uniquely identifies every row in the table. The key can consist of one or more fields.

There are multiple advantages to and uses for primary keys:

- TX DWA can enforce the primary key constraint, i.e. that all the primary keys are unique.
- You avoid duplicate values in your dimensions when you consolidating data from different business units.

To include a field in the primary key for at table

- Right click the field and click **Include in Primary Key**.




## HIDING A FIELD FROM THE VALID INSTANCE OF A TABLE (RAW-ONLY FIELDS)

A raw-only field is a field that only exists in the raw instance of the table and not the valid instance. This way, the field won't show up in OLAP cubes, Qlik models and other front-ends that use the valid instance of the table. This is very useful if you have fields one the table with no other purpose than to be part of other fields, e.g. a surrogate key when you do dimensional modeling.

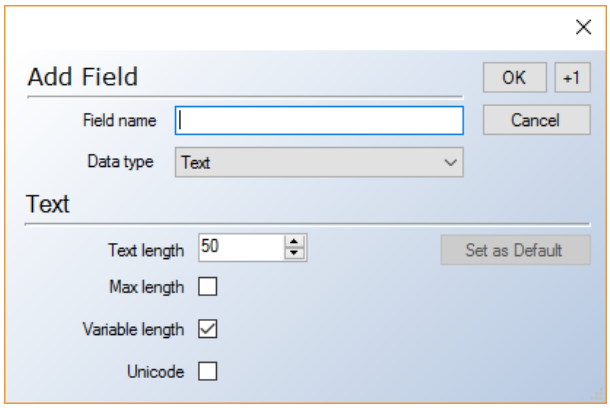
To make a field raw-only

- Right click the field and click **Raw-only field**.

## CUSTOM FIELDS

Custom fields - or just "field" in the data warehouse - is basically an empty shell that you need to add data to through transformations, scripting or data copy. It has many uses - for instance, you can build a model of your data warehouse before bringing data in. Follow the steps below to add a custom field.

1. Right click a table and click **Add Field/Add Custom Field**. The **Add Field/Add Custom Fields** window appear.




2. In the **Field name** box, type a name for the field.
3. In the **Data type** list, click on the data type you want to use for the field.
4. Define the attributes of the selected data type. You have the following options:

| Data Type | Attribute | Description |
|---|---|---|
| Text | Text length | Specifies the maximum number of characters the field can contain. |
| | Max length | Specifies that the field can contain any amount of characters up to a storage size of 2 GB. |
| | Variable length | Specifies that the field can be of variable length. |
| | Unicode | Enables Unicode character-encoding. |
| Integer | Type | The size of the integer: bigint, int, smallint, tiny-int. |
| Numeric | Number of decimals | Specifies the number of decimals in the field. |
| | Mantissa bits | Sets the precision of the floating point number if floating point is selected. 1-24 bits equals single precision, while 25-53 bits equals double pre-cision. |
| | Floating point | Specifies that the number should be saved using floating point notation. |

| | | |
|---|---|---|
| Binary | Length | The length of the binary field. |
| | Max length | Specifies that the field can contain any amount of characters up to a storage size of 2 GB. |

5. (Optional) If you want to use the same attributes the next time you add a field of the same type, click **Set as Default**. The defaIf the button is disabled, the current settings are the default settings.
6. Click **OK** to add the field and close the window or **+1** to save the field and add another field.

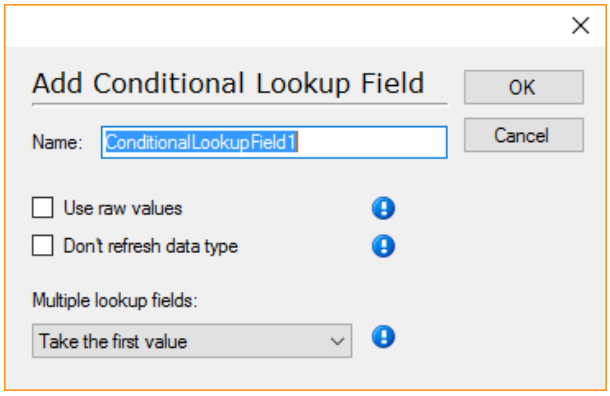
## CONDITIONAL LOOKUP FIELDS

Conditional lookup fields are used to add a field to a table in order to retrieve the value of the field in another table. A conditional lookup field contains a number of lookup fields and conditions that

### ADDING A CONDITIONAL LOOKUP FIELD

Follow the steps below to add a conditional lookup field.

1. Right click on a table in a staging database or data warehouse and click on **Add Conditional Lookup Field.**



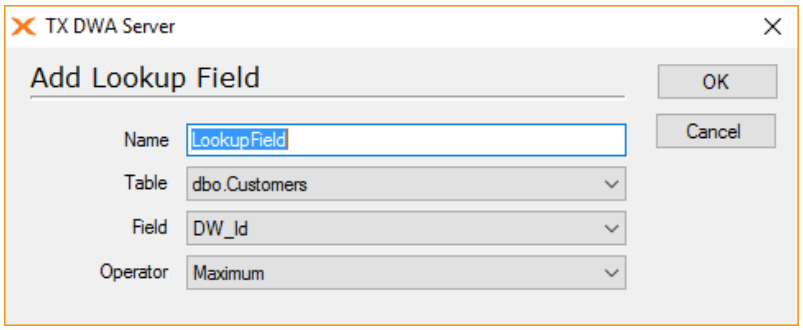

2. In the **Name** field, type a name for the lookup field.
3. Select **Use raw values** to perform the lookup on the raw values of the source table instead of the valid values, i.e. before any transformations or other cleansing tasks are performed. Lookups are always inserted into the raw destination table, and this setting does not affect that.
4. Select **Don't refresh data type** to set the data type of the lookup field manually. Per default, TX DWA will set the data type of the conditional lookup field to the data type of the source field of the first lookup field. If you enable this option, you can right-click the conditional lookup field, when it has been added, and click **Edit Data Type**. If your conditional lookup field contains lookup fields with different data types, this option is useful to set a data type that can contain all possible values.

5. Under **Multiple lookup fields**, select what value is used when a the field contains more than one lookup field. In any case, the lookup fields are evaluated in the same order as they appear in the tree. What happens when there is a match depends on this setting, which can be one of the following:
    - **Take the first value:** The value of the conditional lookup field will be the value of the first lookup field with a condi-tion that evaluates to true.
    - **Take the first non-empty value:** The value of the conditional lookup field will be the value of the first lookup field with a condition that evaluates to true and is not empty.
6. Click **OK**. The field is added to the project tree under the table.

### ADDING A LOOKUP FIELD TO A CONDITIONAL LOOKUP FIELD

After creating a conditional lookup field, the next step is to specify the lookup fields that contain the values that can end up in the field. You can add multiple lookup fields to one conditional lookup field.

1. Expand the conditional lookup field, right-click **Lookup Fields** and click **Add Lookup Field**.
   - OR -
   Drag a field from a table on a data warehouse to **Lookup Fields** under the conditional lookup field. This pre-fills some of the field in the **Add Lookup Field** window that appears.




2. In the **Name** field, type a name for the field.
3. In the **Table** list, select the table containing the field you want to use.
4. In the **Field** list, select the field you want to use.
5. In the **Operator** list, specify how to return the values. You have the following options:
    - **Top:** Returns the value from the first record that matches the join criteria. When you select this operator, a **Sorting** node will be added to the project tree under the lookup field. Right click this and click **Add Sorting** to define how the matching values are sorted before they are retrieved from the source table.
    - **Sum:** Returns a sum of all the values that match the join criteria. This will only work on numeric values. Null values are ignored.
    - **Count:** Returns a count of all the values that match the join criteria. Null values are ignored.

- **Maximum:** Returns the highest value of the values that match the join criteria. For strings, it will find the highest value in the collating sequence. Null values are ignored.
- **Minimum:** Returns the lowest value of the values that match the join criteria. For strings, it will find the lowest value in the collating sequence. Null values are ignored.
- **Average:** Returns the average value of the values that matches the join criteria. This will only work on numeric values. Null values are ignored.

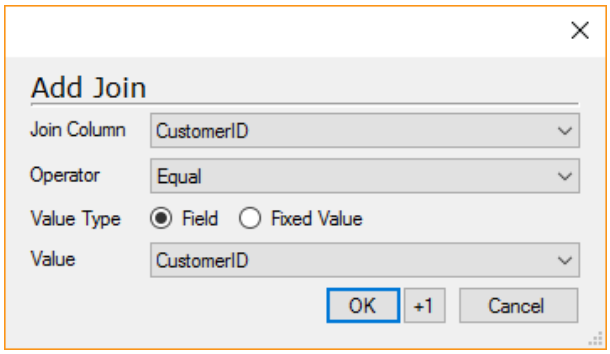
6. Click **OK**.

**Note:** You can 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.

### ADDING A JOIN TO A CONDITIONAL LOOKUP FIELD

Next you have to add a join that specifies which join criteria must be met in the source table. Less complex joins will make the lookup perform faster. Complexity is a combination of the number of fields in the join and the data type. To get the best performance, use one single numeric field for the join.

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




2. In the **Join Column** list, select the field that uses the lookup.
3. In the **Operator** field, specify when to look up a value.
4. Click **Field** or **Fixed Value** to specify if you want to compare the field selected in the join colum list to a field on the destination table or a fixed value. The **Value** box changes to fit your choice.
5. Depending on the value type, click the relevant field in the **Value** list or enter a value in the **Value** box.
6. Click **OK**.

**Note:** If you do not specify a join for a lookup field, it will use the default join between the source and destination tables if such a join exists.

### SPECIFYING CONDITIONS

You can now specify conditions for when to lookup. The lookup will only be performed when the condition evaluates to true. For example, if you can determine that the lookup will only find related values when a certain field in the destination table has a certain value or apply a condition to avoid the lookup being performed on many records without finding a matching record. Conditions must also be used when having multiple lookup fields within one conditional lookup field to determine which lookup field to use. Per default, the first lookup field where the condition evaluates to true will be used, even if it returns a NULL value or finds no matching records. If no conditions are specified, the first lookup field will always be used and any subsequent lookup fields will be ignored.

1. Expand the lookup field andclick**Conditions**. The **Conditions** task pane appears.
2. Click on a **Field** in the pane.
3. In the **Operator** list, click the operator you want to use.
4. Click **Value** and enter a value to use in the comparison in the box
   - OR -
   Click **Fields** and select a field to use for the comparison in the list.
5. Click **OK** to add the condition.

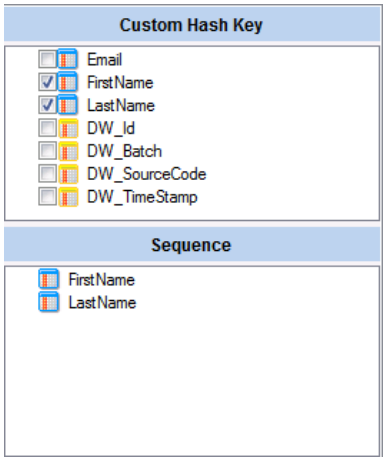
**Note:** If you select the Custom operator, you can script your own condition. When you drag a field from the list to the right in the script editor, remember to specify if you refer to the field in the raw or the transformation view instance of the table by prefixing the column with "R.[FieldName]" or "T.[FieldName]". If you don't do this, TX DWA will not be able to tell what field to use and the deployment of the data cleansing script will fail.

## CUSTOM HASH FIELDS

In scenarios with multiple fields making up the primary key, hashing the values of those fields into a single field can improve lookup performance. You can also use the field to easily investigate whether changes has been made to a record or not. In TX DWA, such a field is called a custom hash field.

### ADDING A CUSTOM HASH FIELD

1. Right-click a table and click **Add Custom Hash Field**. A custom hash field is added to the table and selected in the project tree. The **Custom Hash Key** pane appear in the right hand side of the window.




2. In the **Custom Hash Key** pane, select the fields you want to include in the custom hash field.
3. (Optional) Under **Sequence**, you can reorder the fields using drag-and-drop or by selecting a field and pressing **ALT** + **Up** or **Down**. If you are comparing two fields, it is important that the sequence is the same on both custom hash fields. Otherwise, the hash value will be different even if the values of the individual fields are the same.

### CHANGING THE HASHING ALGORITHM FOR A FIELD

For compability reasons, we offer a number of different algorithms for hashing fields in TX DWA. These hashing algorithms are available on all hashed fields in your project. Apart from custom hash fields, there can be the following hashed fields:

- Junk dimension key
- BK hash key
- Incremental hash key and incremental value key (used for target-based incremental load)
- Surrogate hash key, type I hash key and type II hash key (used for history/slowly changing dimensions)

There is usually no reason to change the hashing algorithm for an individual field. The default setting, "use project default", ensures that all hash fields use the same algorithm which in turn makes it possible to compare the values of the individual fields. The most common exceptions are debugging purposes and "upgrading" the hashing algorithm of a field that was created in an earlier version of TX DWA.

To change the hashing algorithm of a hash field

- Right click the field, click **Hashing algorithm** and click the hashing algorithm you want to use.

# 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 TX DWAfollows the same methodology as creating standard SQL views.

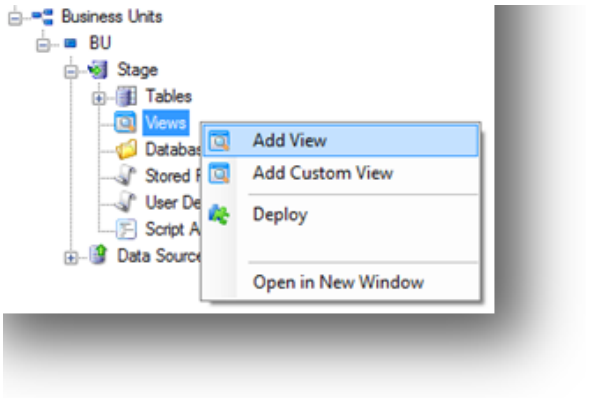
In TX DWA you can create two types of views: views that consist of a subset of columns or rows in one or more tables and views that are joins of one or more tables.

## CREATING VIEWS BASED ON LOOKUP FIELDS

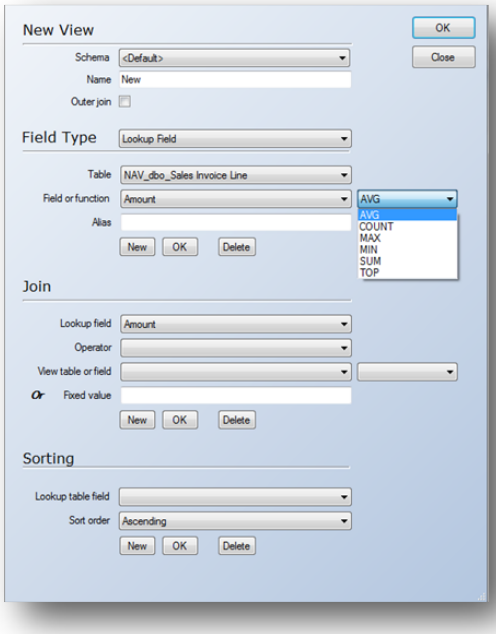
Creating a view based on a lookup field consists of the following steps. The steps are all carried out in the View dialog.

### CREATING A LOOKUP FIELD

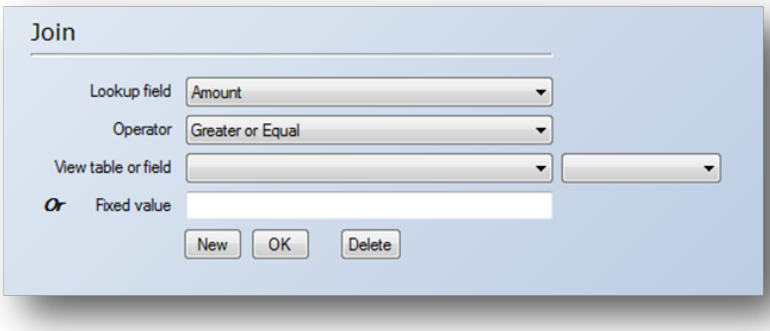
1. On the **Data** tab, expand the chosen data warehouse or staging database. Right-click**Views** and then click **Add View**.



2. The Add View window appears.




3. In the **Name** field, type a name for the view. You can also click **Schema** to select a schema and **Outer Join** to use outer join in 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:
    - Click the **Table** list and select the table that holds the lookup field.
    - Click the **Field/Function** list, select the field or function you want to use 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.
    - In the **Alias** field, type a name for the lookup field if you want the name to be different from the source field name.
5. Click **OK**. The field is displayed in the **View** pane of the window.
6. Click **New** if you want to create another field.
7. You have to specify a join between the view table and the lookup tables.

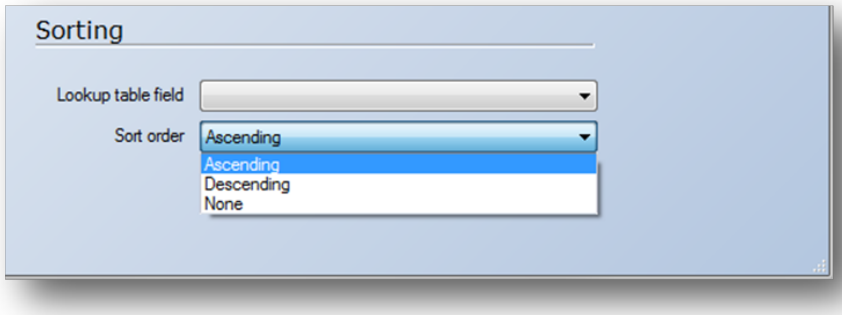


8. Click the **Lookup field** list and select the field to look up.
9. In the Operator field, select the operator that determines how you want the columns compared.

| Operator | Description |
|---|---|
| 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.

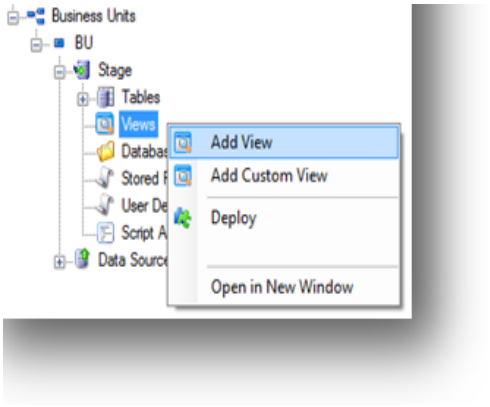
10. Click **OK**, and then click **New** if you want to specify a new join.
11. You also have to define the sort order.



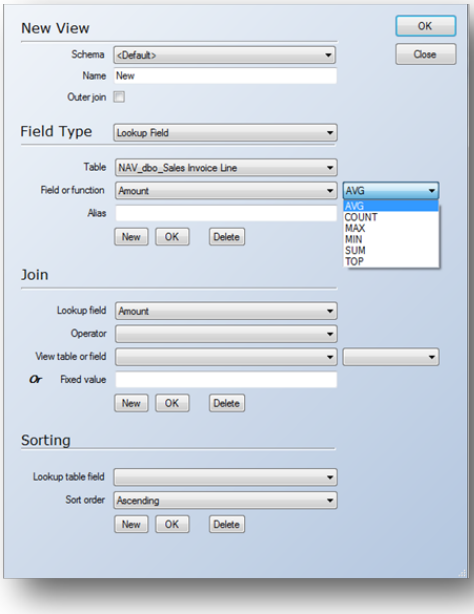

12. In **Lookup Table Field** list, click the field you prefer.
13. In the **Sort Order** list, select how you want the results sorted. Results can be sorted in either ascending or descending order.
14. Click **OK**, and then click **New** if you want to specify a sort order for another field in the view.
15. 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. 1. On the **Data** tab, expand the chosen data warehouse or staging database , right-click Views, and then select **Add View**.




The **New View** windows opens.




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.

# INDEXES

To achieve the optimal performance on your data warehouse, it is important to have the right indexes on your tables. TX DWA can generate the necessary indexes automatically or assist you in creating indexes manually. You can also choose to use a legacy approach to indexes instead.

With the Index Automation feature, you can let TX DWA handle all index creation and maintenance. 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)

Index Automation will try to minimize the number of indexes. If two lookups can use the same index, TX DWA will take advantage of that. In addition to that, TX DWA takes any manually created indexes into consideration. It will not change your manually created indexes, but it will use them instead of creating similar indexes. The indexes created by Index Automation will be named AutoIndex and postfixed with a number for uniqueness within each table.

## SETTING UP INDEX AUTOMATION

Index Automation is configured on the project level, but can be overwritten on the individual table. The following options are available:

- **Automatic** (default): Index automation updates the indexes whenever the user changes the project in a way that could trigger a new or altered index.
- **Manual**: The user can have TX DWA create indexes on selected tables. However, these indexes are not managed by TX DWA. Nothing happens automatically if the table is changed in a way that impacts the indexes.
- **Disabled**: TX DWA will use the legacy index generation behavior. Indexes will be generated during execution when needed by a data cleansing procedure. However, the same index might be created multiple times, since the index generation behavior is not tuned for performance. In addition to that, these auto-generated indexes are not visible for the end user in the project tree.

### CONFIGURING INDEX AUTOMATION FOR THE PROJECT

To configure the Index Automation setting on the entire project, follow the steps below.

1. On the **Data** tab, right click the project node, and click **Edit Project**. The **Edit Project** window appears.

2. In the **Default Index Automation type** list, click the option you want to use.
3. Click **OK**.

### CONFIGURING INDEX AUTOMATION FOR A TABLE

To configure the Index Automation setting on a specific table, follow the steps below.

1. On the **Data** tab, right click the table and click **Table Settings**. The table settings window appears.
2. On the **General**tab, in the **Index Automation** group, click the option you want to use.
3. Click **OK**.

## MANUAL INDEX GENERATION

Setting the index automation setting to manual, makes it possible for you to use the index generation features of TX DWA while maintaining complete control over the indexes in your project. When you run manual index generation on a table, data warehouse, staging database or the project, TX DWA creates any indexes Index Automation finds necessary. However, you can delete and edit indexes as you see fit. TX DWA will not create new indexes on the tables unless you run manual index generation again.

### GENERATE INDEXES MANUALLY ON THE PROJECT, A DATA WAREHOUSE OR A STAGING DATABASE

To generate indexes on a data warehouse, staging database or the entire project, follow the steps below.

1. Set or make sure Index Automation is set to manual. See Configuring Index Automation for the Project.
2. Right-click the project, data warehouse or staging database you want to use manual index generation on, click **Advanced** and click **Index Automation(manual)**.
3. A message will appear to tell you how many tables TX DWA checked. Click **OK**.

### GENERATE INDEXES MANUALLY ON A TABLE

To generate indexes on a specific table, follow the steps below.

1. Set or make sure Index Automation is set to manual. See Configuring Index Automation for the Project.
2. Right-click the table you want to use manual index generation on, click **Advanced** and click **Index Automation(manual)**.

   Note: If the table you want to add an automatic index to already has one or more indexes, the **Index Automation (manual)** option is not available in the **Advanced** menu. Instead, expand the table, right click **Indexes** and click **Index Automation (manual)**.

# HISTORY

The history feature is like "track changes" for tables. It allows you to record how data in a table changes over time so you can create reports that show how data looked at a particular moment in time.

The principle behind history on tables is quite simple. Instead of simply storing every unique record once, a version of the record is stored every time an important change is made to the record. To keep track of what record is the currently valid one, meta data fields are added to the table. Two of these, "valid from" and "valid to", can be combined to get the time span the table was valid in.

When a table has history enabled, TX DWA compares the records loaded in from the data source with the records already in the data warehouse. If the record is new, it is added to the data warehouse. If there is a record with the same key already, hashed versions of the two records are compared to each other to find changes. If no changes are detected, nothing is updated in the data warehouse. If one or more changes are detected, what happens depend on the type of the field or fields that have changed. TX DWA support the following types:

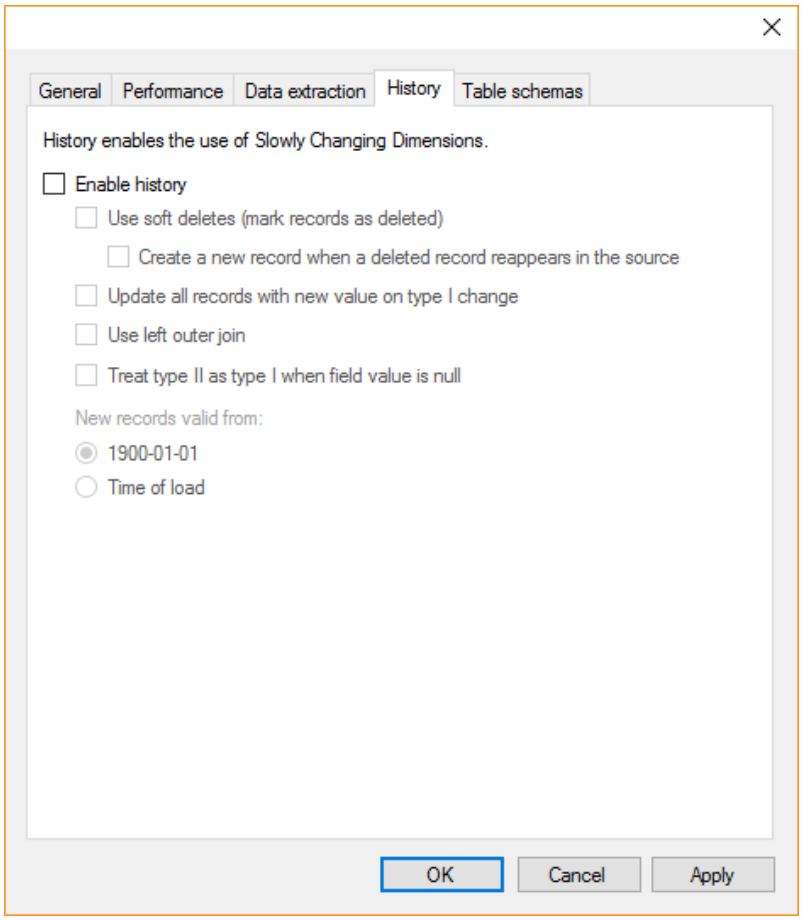
- **Type 0:** Fields that are basically ignored by the history logic. They are inserted together with the rest of the record when it is created – either on the initial load or on a type II change - but they do not trigger an update. Type 0 fields give you flexibility in your history-enabled tables. For instance, you might have a type 0 field with values that are calculated by custom code inside or outside TX DWA.
- **Type I (default):** Fields that are overwritten with new data when data changes in the source. This means that there will be no history of the change. When a type I field changes, the current version of the record is updated with the new field value. You can also configure TX DWA to overwrite all instances of the record with the new value. If, for instance, a customer changes name from ABC Consulting to Acme Consulting, the default behavior is to update the current record with the new name. Previous records will still contain the old name, ABC Consulting. If you enable the option, all instances will contain the new name, Acme Consulting. This is often useful in reporting where the purpose is to have a recognizable name for the customer, not the correct name at a specific time.
- **Type II:** Fields that will cause a new record to be created, thus creating history about the change. TX DWA will create a new record if one of the type II fields on a record has changed, not a new record for each change. The new record will be a copy of the record from the data source. This means that any type 0 fields will be updated.

In addition to these three types, a field can be part of the natural key that uniquely identifies a record to the history logic. The natural key is usually identical with the primary key on the table.

### ENABLING HISTORY ON A TABLE

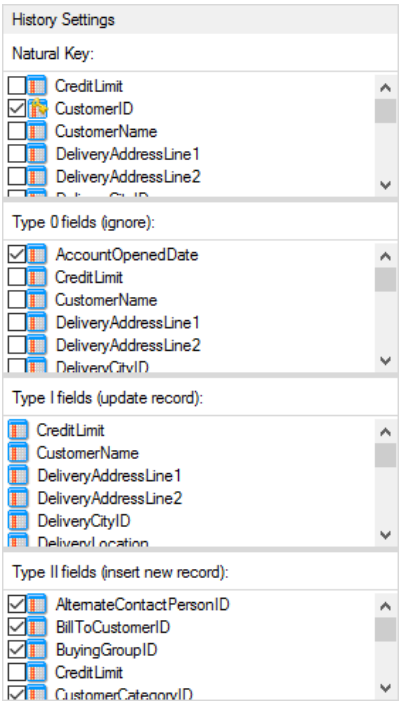
Follow the steps below to enable history on a table.

1. Right click the table and click **Table Setting**. The Table Settings window appears. Click the **History** tab.




2. Click **Enable History**.
3. Check **Use soft deletes** to mark records that have been deleted in the source system as deleted in the table. This is done by setting the "Is Tombstone" system field on the record to 1. Check **Create a new record when a deleted record reappears in the source** to keep track of history when a record is deleted and later restored in the source system. When this option is disabled, the status of the record will simply change between deleted and not deleted with no information saved about the fact that the record was missing from the source system for a while.
4. Check **Update all records with new value on type I change** to update all versions of a record with the new value when a type I change is detected. The default behavior is to only change the value in the currently valid record.
5. Check **Use left outer join** to deploy the table with slightly different SQL code (that includes a left outer join, hence the name). In rare cases, SQL Server will create a very inefficient execution plan for a history-enabled table. The change in the code will cause SQL Server generate a new execution plan for the table.

6. Check **Treat type II as type I when field value is null** to not insert a new record when a type II field changes from null to a non-null value. Enable this when you are not interested in keep track in this kind of change, e.g. when you have added a new field to the table, thus creating a field where all values are null.
7. Under **New columns valid from**, click **Time of load** to have TX DWA insert the time of load in the "valid from" field when a new record is added. Depending on you reporting needs, this might make more sense than the default, **1900-01-01**.
8. Resolve any conflicting table settings (marked with error icons) and the click **OK**.
9. Locate the table in the tree. The table icon will be overlaid with an "H" to make it easy to identify as a history-enabled table. Expand the node and click on **History Settings**. The **History settings** pane appears.




10. Under **Natural Key**, select the fields you want to use to uniquely identify the records. The primary key fields are selected per default. Note that if you add another field to the primary key, you have to add it to the natural key as well to have it work with the history logic.
11. Select the fields you want to be either type 0 or type II under the respective headings - **Type 0 fields (ignore)** and **Type II fields (insert new record)**. Any fields that you do not select as any of these types will be type I.
12. Deploy the table to have the settings take effect.

**Note:** You might see a **Clean Up Tombstone Field** button under history settings. This refers to the "Is Tombstone" field used for keeping track of records that have been deleted in the source. Earlier versions of TX DWA would create an "Is Tombstone" field on history-enabled tables even if deletes was not enabled. Click the button to remove the uneccesary field from the table.

# SCRIPTING

TX DWA generates most of the code you need, but you can extend the functionality of TX DWA by writing your own scripts. When you need to include custom SQL code in your project, you have different options depending on what you need to do.

- **User Defined Functions** and **Stored Procedures** are used to create reuseable code on SQL Server. TX DWA uses them when it generates the code for executing your project. You can create you own User Defined Functions and Stored Procedures and call them from Execution Packages or Script Actions.
- **Script Actions** enables you to add snippets of SQL code to be run before or after each step in the deployment or execution of a table.
- **Snippets** are small snippets of parameterized code you can use on the field level. In addition to the SQL snippets you use in the data warehouse, snippets come in OLAP and Qlik flavors.
- **Custom Code** lets you replace the code generated by TX DWA for deployment and execution with your own code written in your favorite IDE.

You can also create global **Project Variables** for use in your scripts.

## USER DEFINED FUNCTIONS AND STORED PROCEDURES

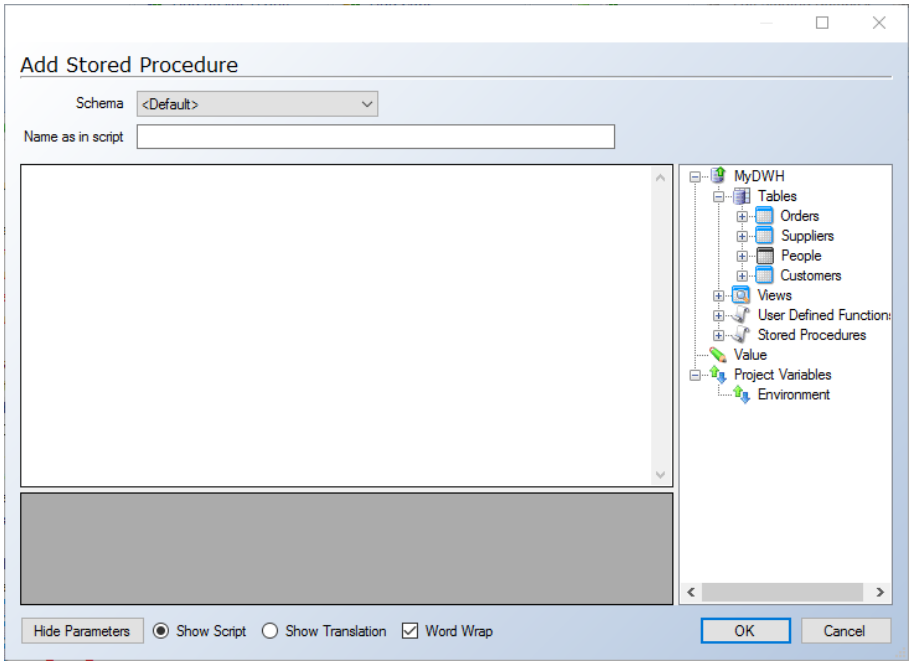
User Defined Functions allow you to define your own Transact SQL functions. A user defined function returns a table or a single data value, also known as a scalar value. You can, for example, create a function that can be used to perform complex calculations.

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.

### ADDING A USER DEFINED FUNCTION OR STORED PROCEDURE

You can add user defined functions and store procedures to both the staging database and to the data warehouse. The steps for adding either one are similar.

1. Under a data warehouse or a staging database in the project tree, right-click **User Defined Functions** and click **Add User Defined Function**.
   - OR -
   Under a data warehouse or a staging database in the project tree, right-click **Stored Procedures** and then click **Add Stored Procedure**.
   The **Add User Defined Function** or **Add Stored Procedure** window will appear depending on your previous selection.



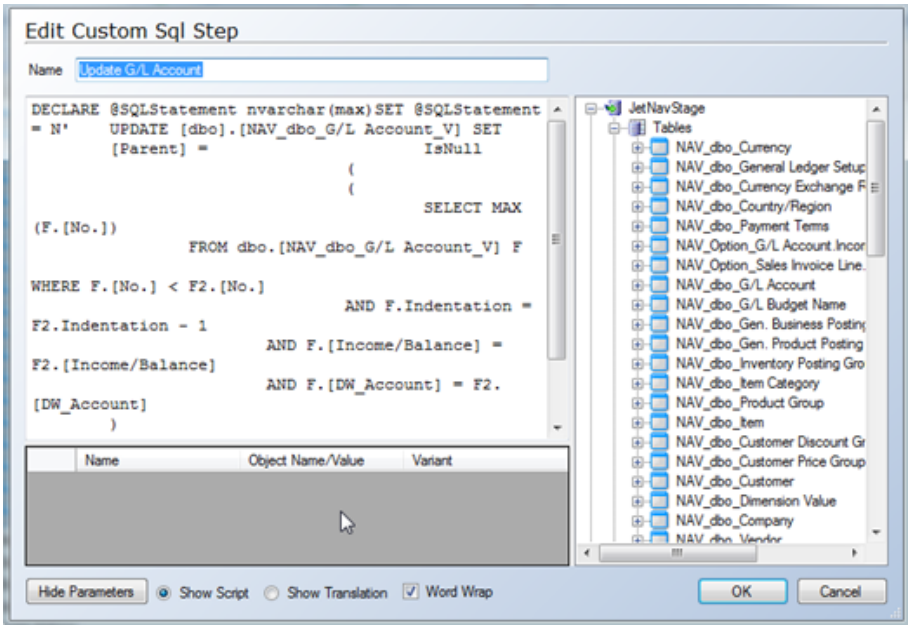

2. In the **Name as in script** box, type a name for the function/procedure.
3. In the main ttext box, enter the SQL code for the user function/procedure. You can drag in tables, fields, views, stored procedures, user defined functions and project variables from the list to the right to use in your function/procedure. Drag in "Value" to create your own custom variable.
4. Click **OK** to save the function/procedure. A **Script Action** can then be created, if necessary, to call the function/procedure.

## SCRIPT ACTIONS

Script Actions are SQL scripts that can be executed along with deployment or execution of a table to complete a number of different tasks. A Script Action can utilize the User Defined Functions and Stored Procedures that you have already created.

### ADDING A SCRIPT ACTION

1. Under a **Data Warehouse** or a **Staging Database** in the project tree, right-click **Script Actions** and click **Add Custom Step.** The **Edit Custom SQL Script** window appears.

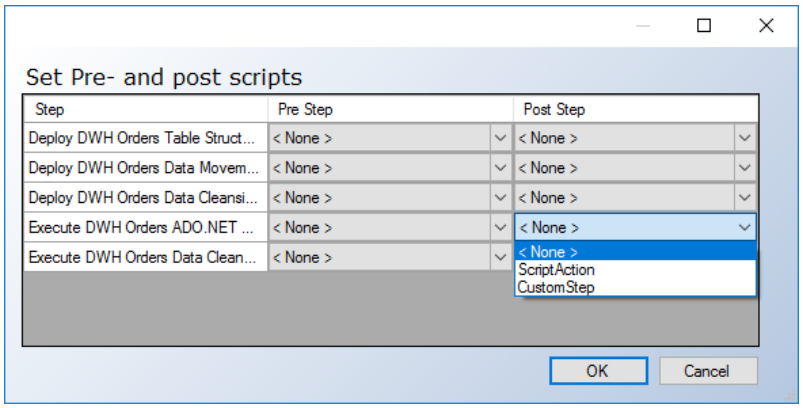



2. In the **Name** box, type a name for your script action.
3. Enter your SQL script in the text box. You can drag in tables, fields, views, stored procedures, user defined functions and project variables from the list to the right to use in your script action. Drag in "Value" to create your own custom variable.
4. Click **OK** to save the script action.

### ADDING A SCRIPT ACTION TO A TABLE AS PRE- OR POSTSCRIPT

Script actions are used in the project as pre- or postscripts for a table. As the names suggest pre- and postscripts are executed before - pre - or after - post - the execution steps for the table.

1. Right-click the table you want to associate with the **Script Action**, click **Advanced** and click **Set Pre- and Post Scripts**. The **Set Pre- and Post Scripts** window appears.




2. The deployment and execution steps on the table in question are listed in the window. The steps at which you can call the script are:
   - **Deploy Table Structure**
   - **Deploy Data Movement View**

- **Deploy Data Cleansing Rules**
- **Execute 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.
- **Execute 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
- **Execute 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
- **Execute Data Cleansing (post-step):** This will cause the script to be called after the data cleansing process has completed.

Click **<None>** in the Pre Step or Post Step column of the step you want to add you script to and click the name of the script.

3. Click **OK**.

## SNIPPETS

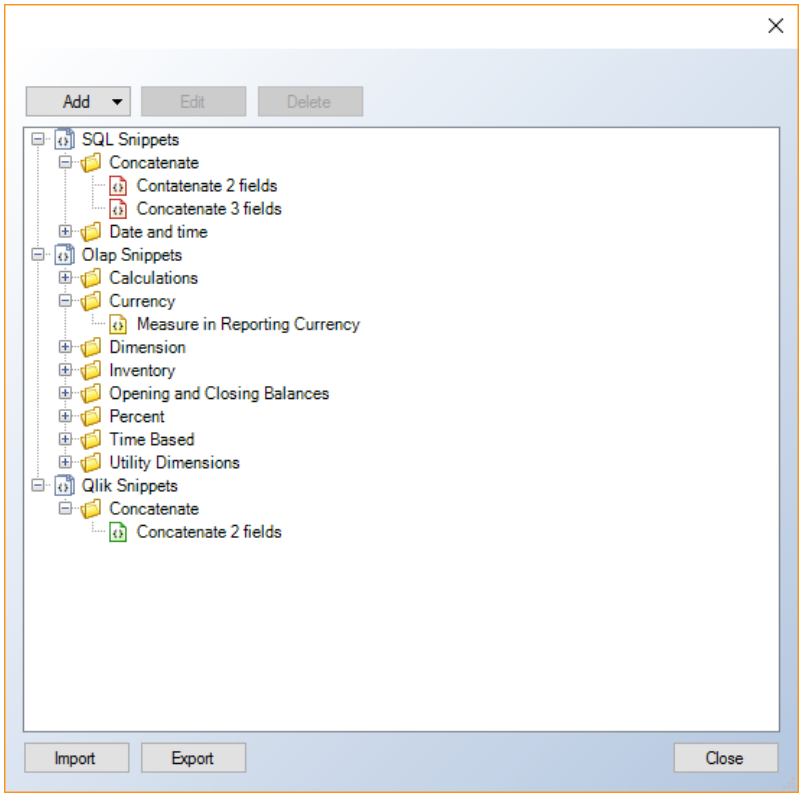
Snippets are reusable, parameterized pieces of code for use in field transformations and other parts of your project. This enables you to write the code once and use it in multiple places, saving you the trouble of maintaining the same functionality on a many fields.

Snippets come in three flavors: SQL, OLAP and Qlik. When you build a data warehouse, you will be using SQL snippets.

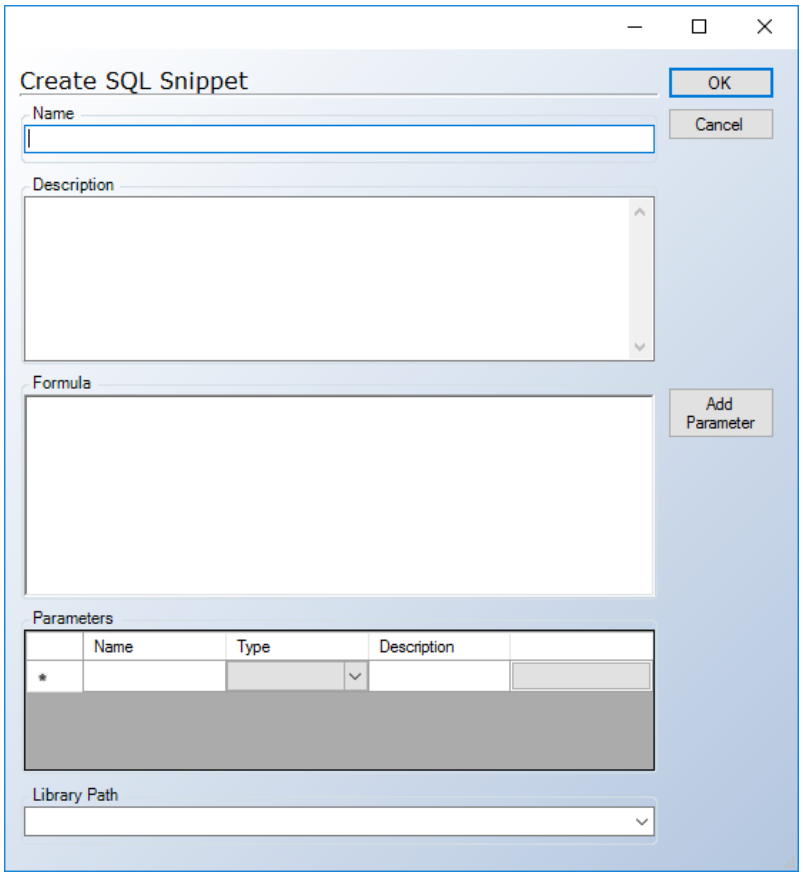
### ADDING A SNIPPET

Whether you want to create a SQL, OLAP or Qlik snippet, the steps are the same. Follow the steps below to create a snippet and substitute OLAP or Qlik for SQL if that is the type of snippet you want to create.

1. On the **Tools** tab, in the **Administration** group, click **Snippets**. The **Snippets** window appears.

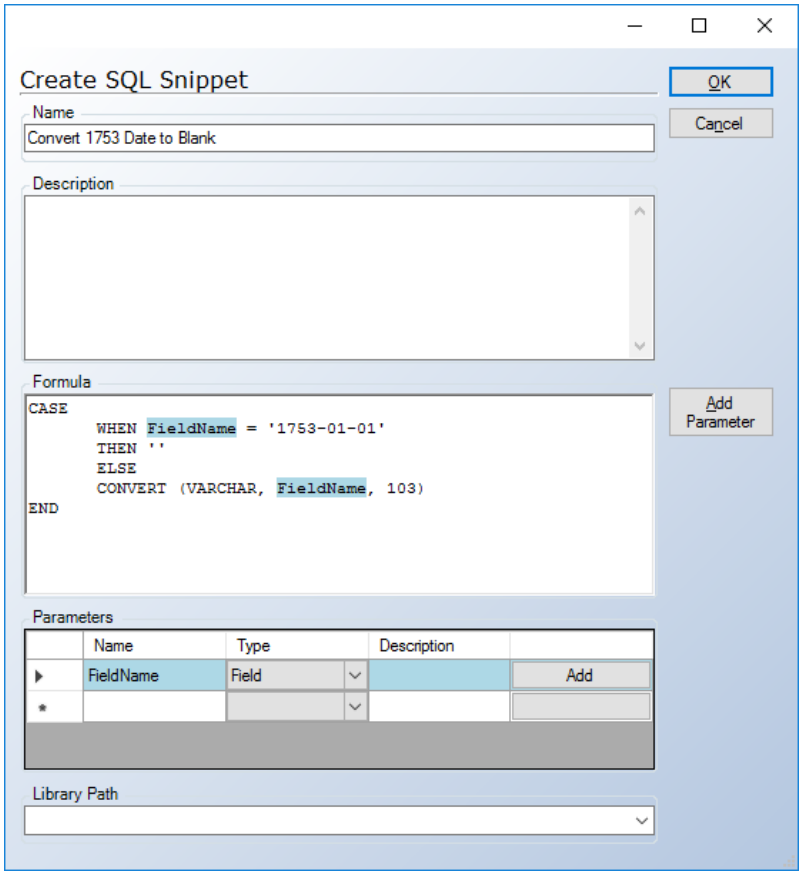



2. Click **Add** and click **SQL Snippet**. The **Create SQL Snippet** window appears.




3. In the **Name** box, type a name for your snippet.
4. (Optional) In the **Description** box, type a description of what the snippet does.

5. Enter the script in the **Formula** box. For any variables ('FieldName' in the example below), highlight the variable and click **Add Parameter**. This will add the highlighted text as a parameter name under **Parameters**.




6. Under **Parameters**, change the **Type** to match what the variable represents. You have the following options:
   - Table
   - Field
   - Database
   - User Defined Function
   - Stored Procedure
   - Value
7. Click **OK** to save the snippet.

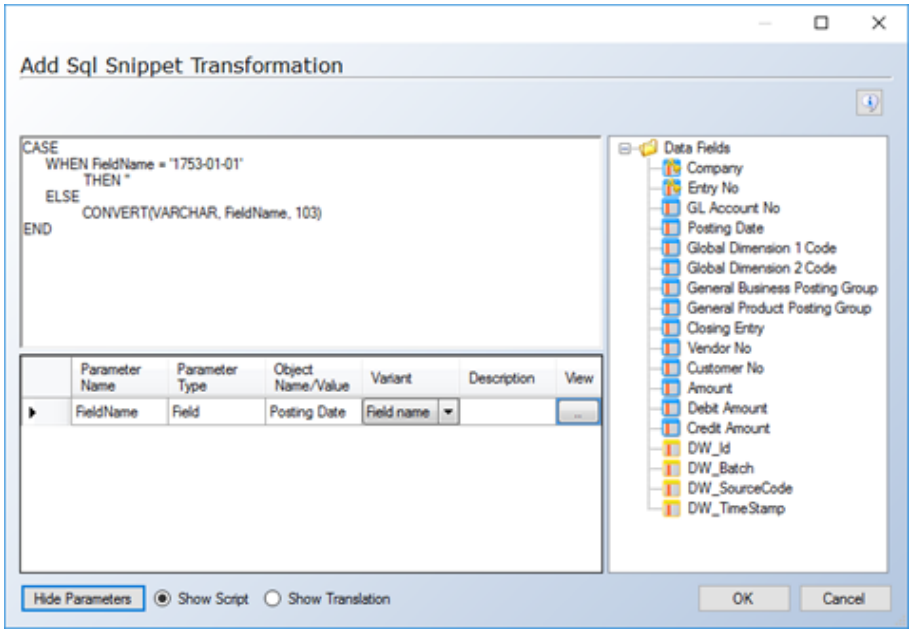
### USING A SQL SNIPPET

SQL snippets can be used in a number of situations.

1. To use a SQL snippet in a field transformation, right-click the field, click **Add SQL Snippet Transformation** and click on the SQL snippet you want to use.
   - OR -
   To use a SQL snippet as a stored procedure , right-click **Stored Procedures**, click **Add Snippet Stored Procedure** and click on the SQL snippet you want to use.
   - OR -
   To use a SQL snippet as a used defined function, right-click **User Defined**

**Functions**, click **Add Snippet User Defined Function** and click on the SQL snippet you want to use.
- OR -
To use a SQL snippet as a script action custom step, right-click **Script Actions**, click **Add Snippet Custom Step** and click on the SQL snippet you want to use.

2. In the window that appears, map the available fields to the parameters in the snippet. Drag the field(s) from the list on the right and drop the field on the **Object Name/Value** column for the relevant variable. The **Object Name/Value** column and **Variant** column will populate automatically.




3. Click **OK**.

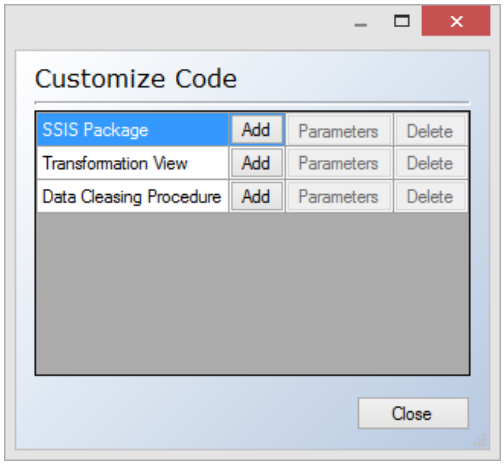
## CUSTOMIZED CODE

TX DWAenables you to integrate "hand-written" code into a project by customizing the data cleansing procedure, transformation view and SSIS package on a given table.

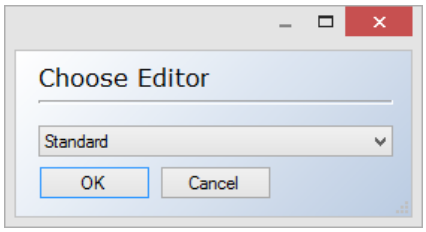
### ADDING CUSTOMIZED CODE TO A TABLE

To customize the code on a given table, follow the steps below:

1. Right click the table in question, navigate to **Advanced** and click **Customize code**. The **Customize Code** window opens.




2. Click the **Add** button to the right of the step you want to customize. The **Choose Editor** window opens.

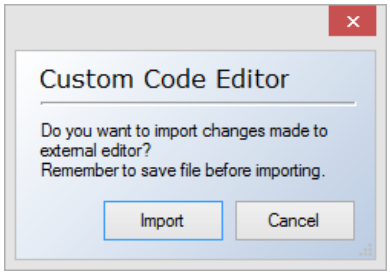



3. In the **Editor Name** list, you have the following options:
   - **Standard** is the basic built-in editor in TX.
   - **Default File Program** is the program that is set to open files of the type in question. For the data cleansing procedure and the transformation view, the file-name extension is .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 use multiple SSIS packages. When creating the Default package, TX DWA will create the first SSIS package only. Examples of tables that will have multiple SSIS packages as default: Data warehouse tables that receive data from multiple staging tables, data source tables from NAV adapters 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 DWA 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 DWA, the **Custom Code Editor** window is open.




Click **Import** to import the changes you have made in the custom editor into your project.

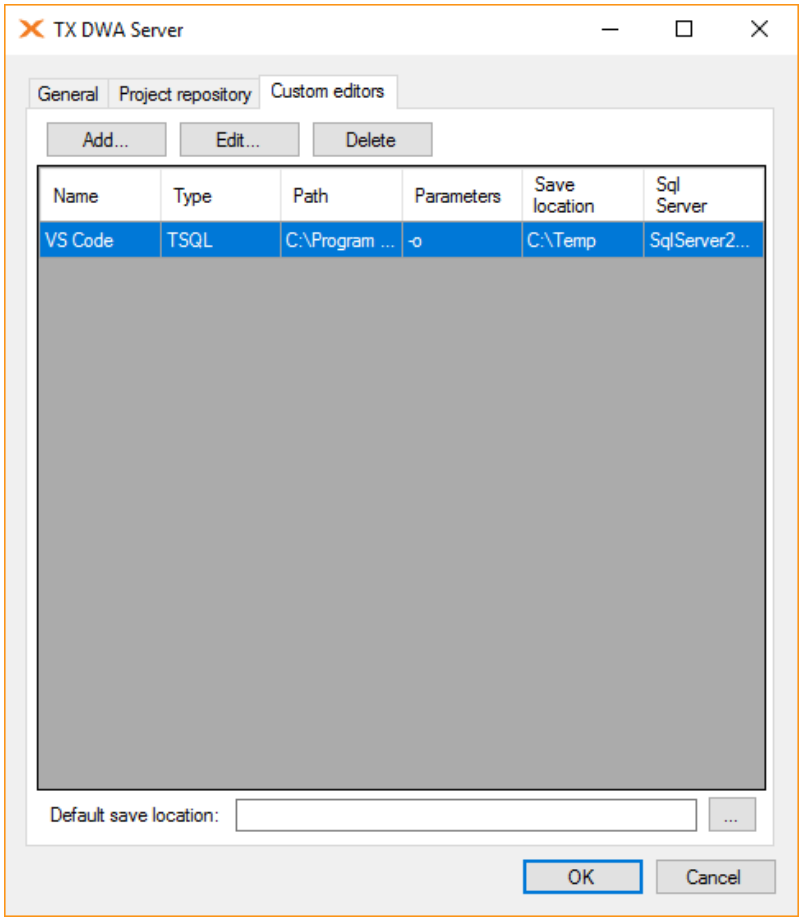
5. When you return to the **Customize Code** window you will notice that you can now click **Parameters** (if applicate) and **Delete**. Click **Delete** to remove the customization and return to having TX DWA generate the code. Click **Parameters** to decide which parameters are sent to the code on 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 the code with the exact same name as TX DWA would have sued. This is what is called during execution. To be sure, simply keep the first line of the code generated by TX DWA.

### MANAGING CUSTOM EDITORS

To add, edit or delete a custom editor

- Click **Options** on the **Tools** menu. Click the **Custom editors** tab in the **Options** window that appears.



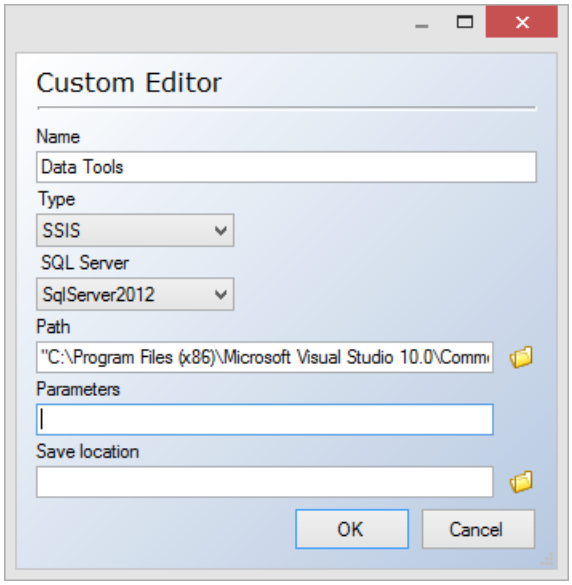

The list of custom editors is displayed. In the **Default save location** box, you can type the path to the folder where the custom code files are temporary stored (or click the folder icon to open a browse dialog).

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.




2. In the **Name** box, type a name for the editor.
3. In the **Type** list, click the type of editor you want to add. Choose TSQL if you want to use the editor with data cleansing procedures and transformation views and SSIS if you want to use it 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 DWAchecks 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 that TX DWA should call to start the program (or click 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 the folder icon to open a browse dialog).
8. Click **OK** to add the custom editor.

## PROJECT VARIABLES

Project variables allows you to save information in project-wide variables. This is useful when you need to distinguish different environments in a script or

The value of a given variable is determined when you deploy the object that uses the variable. As such, when you have changed a variable it is important to deploy the objects that uses this variable. The exception is when you use project variables with customized code. Here, the value of the variables is determined on execution.

The variable does not have a specific data type. If, for instance, you want to use the variable as a string, make sure to enclose the variable in quotes in the script.

### ADDING A PROJECT VARIABLE

To add a new project variable, follow the steps below.

1. In the project tree, right-click your project and click **Project Variables**. The **Project Variables** window opens.
2. Click **Add**. The **Add New Variable** window appear.
3. In the **Name** column, type a name for the variable.
4. In the **Type** list, click the variable type you want to use. You have the following options:
    - **Fixed:** A fixed string.
    - **System:** One of the following system properties:
        - MachineName
        - EnvironmentName
        - UserName
        - UserDomainName
    - **Source Scope:** A property of the source of the current object. For instance, if you use a source scope variable in a custom transformation rule on a table in the data warehouse, the variable will have the value of property on the relevant staging database. Since different possible sources have different properties, the variable might not always have a value. Examples of properties include Database Name, API Version, Host, File Name.
    - **Destination Scope:** A property of the destination of the current object, similar to source scope..
    - **Contextual Scope:** A property of one specific element in the project, such as database name on a particular staging database.
    - **Dynamic:** The value of the variable is generated by a custom script you written.
5. If you are adding a dynamic variable, in the **Resolve Type** list, select when you want to resolve the value of the variable. You have the following options:
    - **Every Time:** Resolve the value every time the value is used.
    - **One Time:** Resolve the value when the variable is used for the first time and reuse the resolved value for the following uses until the project is closed.
    - **Each Batch:** Resolve the value once for each batch, e.g. an execution.
6. If you are creating a Contextual Scope variable, click the object you want to use in the **Context** list.
7. If you are adding a Source or Destination Scope variable, click the Value Filter list and click the type of object you want to see available properties for in the **Value** field.
8. If you creating a fixed variable, enter the value of the variable in the **Value** field.
   - OR -
   If you are creating a dynamic variable, click **Script Editor** to open the standard script editor in TX DWA and write the script that generates the value.
   - OR -
   If you are creating a variable of a type other than fixed or dynamic, in the **Value** list, click the property you want to use as a value for the variable.
9. Click **OK**.

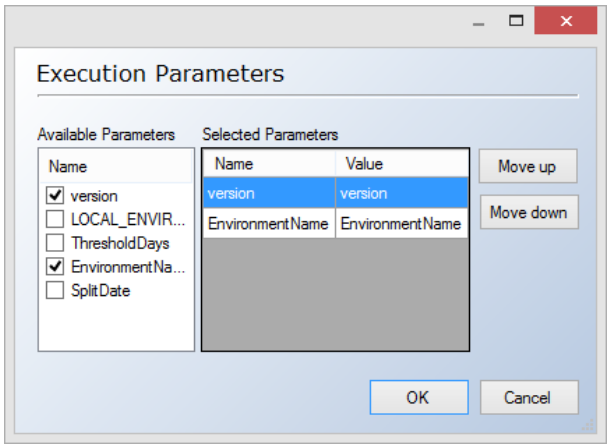
### USING A PROJECT VARIABLE IN A SCRIPT ACTION, CUSTOM TRANSFORMATIONS AND CUSTOM VIEWS

Project variables are available for use when writing script actions as well as custom transformation rules and custom views. The available variables are listed in the tree view in the right hand side of the editor window. Simply drag the variable in from the tree to use it.

### USING A PROJECT VARIABLE WITH CUSTOMIZED CODE

Project variables are available as parameters when using the customized code feature. To make a project parameter available in a customized step, follow the steps below.

1. On the **Data** tab, right click a table with customized code. Click **Parameters** next to the step in which you would like to use a project parameter. The **Execution Parameters** window opens.




2. In the **Available Parameters** list, select the variables you want to have available.
3. Click **OK**.

# DATABASE SCHEMAS

Database schemas allow you to apply a certain schema to a table or a group of tables. You can use schemas to e.g. restrict access to tables that report designers do not need, thereby making reporting off of the data warehouse easier.

Schemas can be set on staging databases and data warehouses as well as data sources and individual tables. The schema settings are applied as follows: Table level settings take precedence over data source settings which in turn take precedence over business unit/data warehouse settings.

## ADDING A DATABASE SCHEMA TO A DATA WAREHOUSE OR BUSINESS UNIT

To create a database schema, follow the steps below.

1. On a data warehouse or business unit, right-click **Database Schemas** and click **Add Database Schema**.
2. In the **Name** box, enter a name for the new schema. In the **Owner** box, you can enter the owning role for the schema. The default is "dbo". Click **OK** to create the schema.
3. Assign a Schema Behavior by right-clicking on the newly created schema. You have the following options:
   - **None**: The schema will be applied to the tables you manually assign it to.
   - **Main default schema**: The schema will be applied to all tables and views in the region (data warehouse or staging).
   - **Main Raw default schema**: The schema will be applied to all Raw (_R postfix) tables in the region (data warehouse or staging).
   - **Main Transfer default schema**: The schema will be applied to all Transfer (_T postfix) views in the region (data warehouse or staging).
   - **Main Valid default schema**: The schema will be applied to all Valid (_V postfix) tables and views in the region (data warehouse or staging).
   - **Main Error/Warning default schema**: The schema will be applied to all Link and Message (_L and _M postfix) tables in the region (data warehouse or staging).
4. If you have selected **None** as the **Schema Behavior**, you need to assign the schema manually. Right-click the table, click **Table Settings** and click the **Table Schemas** tab. Here, you can then select a schema as **Default** (all instances of this table), **Raw**, **Transformation**, **Valid** or **Error/Warning**.
5. Assign user rights to the schema. This can be done through SQL Server Management Studio or T-SQL. See this article on the Microsoft website for details on how to grant user rights using T-SQL: http://msdn.microsoft.com/en-us/library/ms187940.aspx

## CONFIGURING SCHEMAS FOR TABLES AND DATA SOURCES

Schemas for tables and data sources are configured in the settings for the respective objects.

1. Right click a table and click **Table Settings**.
   *OR*

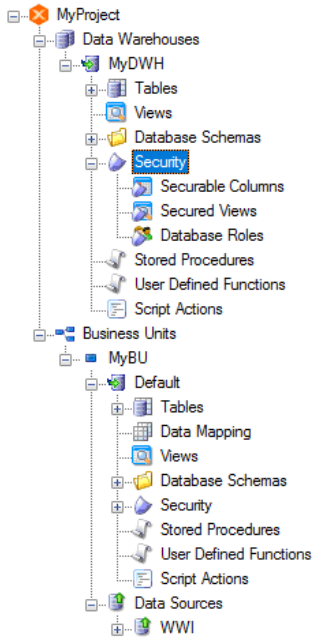
Right click a data source and click **Data Source Settings**.
The settings window for the table or data source appears.

2. Click the **Schema** tab. Here, you can choose the schemas to use for the different instances of the table(s). The lists contain the schemas added to the data warehouse or business unit the table or data source belongs to. Click **Add new schema...** to add a new schema for use on the table or data source.
3. Click **OK**.

# DATA SECURITY

In TX DWA, you can control access directly on the data warehouse or staging database. You can restrict access to specific views, schemas, tables and columns on tables - object level permissions - or specific data in a table - data level permissions.

The access control features can be found under **Security** under any data warehouse or staging database in the project tree on the **Data** tab.




## ADDING A DATABASE ROLE

Object level security is based on SQL Server database roles. A user has access to an object if he is a member of a database role that has access to that object. To add a database role, follow the steps below.

1. Under **Security**, right click **Database Roles** and click **Add Database Role**. The **Database Role Setup** window opens.
2. In **Name**, type a name for the role.
3. If you are using the Multiple Environments feature: In the **Member** setup list, click **Environment Specific Role Members** if you want to have a different setup for different environments. The different environments will then each have a tab in the list below.
4. Next, you should add users to the role. Click **Add login** to role to add Active Directory or SQL Server logins that are known to the SQL Server that the data warehouse resides on. The **Select Login(s)** window opens. Select the logins you want to add in the list and click **OK** to add the user(s).
5. Click **Add manually** to add Active Directory or SQL Server users or groups that the SQL Server does not know, e.g. users on a production server. The **Enter User or**

**Group ID** window opens. In **ID**, type the user or group id. Under **Type**, click **AD user/group** or **SQL user/group** depending on the type of ID you entered. Click **OK** to add the user.

6. Click **OK** to close the window and add the database role, which is listed under **Database Roles** in the project tree.

**Note:** Roles are limited to the data warehouse or business unit it was created on, i.e. you cannot use a role created on one data warehouse on another data warehouse or business unit.

If you need to add a new login on the SQL Server, you can right click **Security** in the project tree and click **SQL Server Logins**. Here, you can add logins if you have the necessary permissions on the SQL Server. However, for safety reasons, you cannot delete users here.

On each deployment, TX DWA drops existing roles on the database before recreating them. By default, TX DWA only drops database roles related to the data warehouse or staging database being deployed. However, you can also set TX DWA to drop more database roles with a setting on the data warehouse or staging database. To access the setting, right click the data warehouse or staging database, click **Edit Data Warehouse** or **Edit Staging Database** as applicable and click **Advanced...** In the **Drop Role Option** list, click **Roles Created by Application** to drop all roles created by TX DWA or **All Roles on Database** to drop all roles altogether.

### ASSIGNING OBJECT LEVEL PERMISSIONS TO DATABASE ROLES

You can assign permissions to database roles on the object level. TX DWA uses the same allow/deny concept as SQL Server with three possible states:

- **Not set** (grey dot): The database role is not allowed to access the object, but are not explicitly denied access.
- **Grant** (green with white checkmark): The database role is granted access to the object. However, if a user is a member of another database role that is denied access, he will not be able to access the object.
- **Deny** (red with white bar): The database role is denied access to the object. Even if a user is a member of another database role that is allowed access, he will still be denied access.

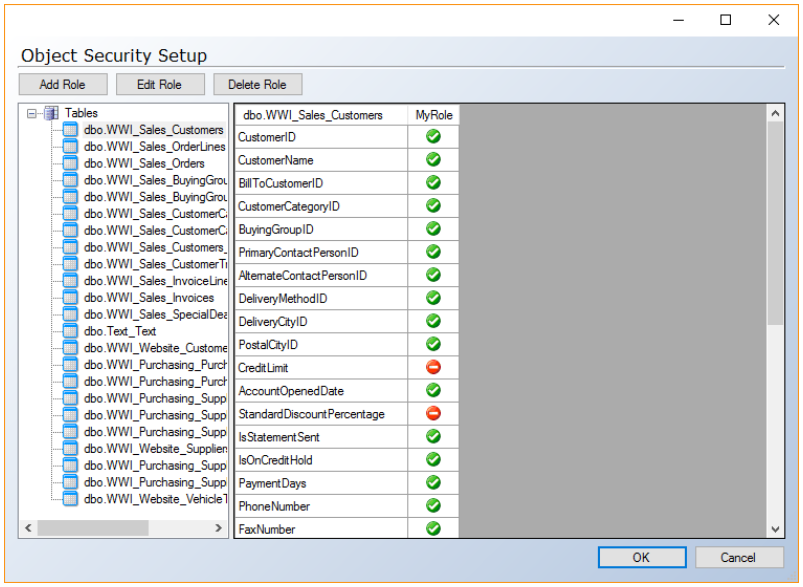
In addition to the three states described above, a table can have different mixed states depending on the column level permissions set on the table. The mixed states are:

- **Partially Granted** (green and grey icon). The database role is granted access to some columns on the table. Note that you will also see this icon if the database role is granted access to all columns on a table since this will not automatically set Allow on the table level.

- **Partially Denied** (red and grey icon): The database role is denied access to some columns on the table. Note that you will also see this icon if the database role is denied access to all columns on a table since this will not automatically set Deny on the table level.
- **Mixed Grant/Deny** (red and green icon): The database role is granted access to some columns and denied access to other columns on the table.

To assign object level permissions, or column level permissions on tables, to database roles, follow the steps below.

1. On the **Data** tab, in the project tree, under **Data Warehouses** and the relevant data warehouse, right click **Security** and click **Object Security Setup**. The **Object Security Setup** window opens.




2. Click **Tables**, **Views** or **Schemas** in the left-hand column to choose the type of object you want to set up access for. Expand **Tables** and click an individual table to assign column level permissions for that table.
3. In the right-hand column, the table shows object names in the left-most column and database roles in the following columns. Click icon in the intersection between the object name and the database role to change the permission for the database role on that object. If you set column level permissions on a table, this will overwrite any current object level permissions set and vice versa.
4. (Optional) Click **Add Role**, **Edit Role** or **Delete Role** to add, edit or delete database roles as needed.
5. Click **OK** to save changes and close the window.

### ASSIGNING DATA LEVEL PERMISSIONS

In addition to configuring access on the object level, you can filter the data available to individual Active Directory users or SQL Server database roles. You might, for instance, want a

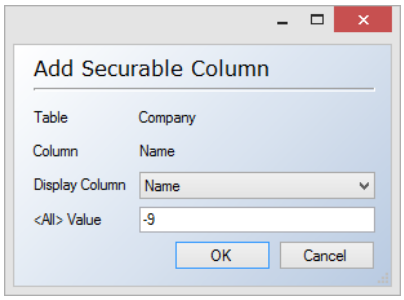
sales person to be able to see all sales data, but only in his or her own region.

Data level security in TX DWA is based on the concepts of securable columns, securable column setups, secured columns and secured views. This design allows you to create one security model and reuse it on any number of tables.

- A **securable column** contains the values that we want to use in a filter. Continuing the example above, it could be "sales region id" in a "sales regions" table.
- A **securable column setup** is a mapping between securable column values and users or database roles, e.g. what "sales region id" does the sales person have access to. Each securable column can have multiple securable column setups.
- A **secured column** is a column on the table containing the data we want to filter. This could be a "sales region id" column on a "sales transactions" table.
- A **secured view** is a view where all the data the user does not have access to is filtered out. For instance, all the "sales transactions" rows where the "sales region id" does not match the "sales region id" the sales person has access to. When using data level security, the secured view should be used for reporting instead of the table it secures.

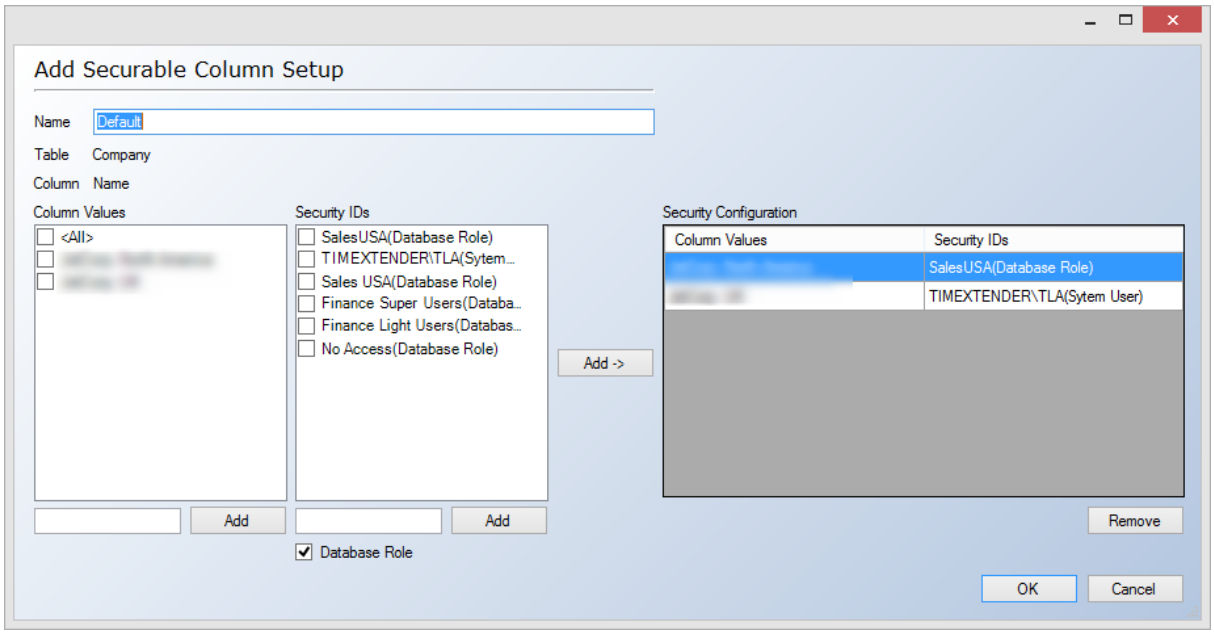
To assign data level security to a table, follow the steps below.

1. On the **Data** tab, navigate to and expand the table that contains the column you want base the permissions on, right click the field and click **Add Securable Column**. The **Add Securable Column** window opens.




2. (Optional) In the **Display Column** list, click the column value you want to display instead of the column you are adding as a securable column. If the securable column contains e.g. an ID, it might be helpful to choose something that is easier to understand, e.g. a name, as the display column.
3. (Optional) In the **<All> Value** box, type a value that will be used to indicate all values. This value should be a value that is guaranteed not to be among the values in the securable column.

4. Click **OK**. The **Add Securable Column Setup** window opens to let you add your first **Securable Column Setup**.



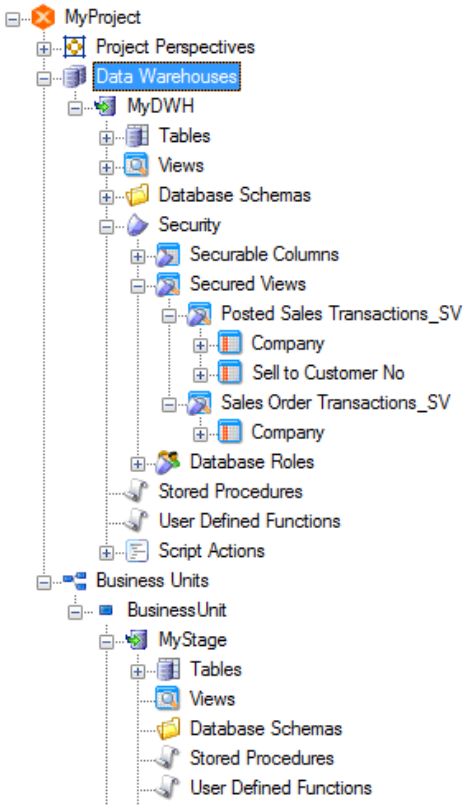

5. In **Name**, type a name for the securable column setup.
6. In the left-hand side of the window, you combine the values in the securable column with the users or database roles that should have permission to access the data. Select a number of values in the **Column Values** list and one or more users or database roles in the **Security ID** list and click **Add->**. The resulting pairs are displayed in the **Security Configuration** list.
7. (Optional) If you need to assign permissions to a value or a security ID that is not in either list, type the value or name in the box under the appropriate list and click **Add**. Select **Database Role** if you want to add a database role as opposed to an Active Directory user. For more information on database roles, see Adding a Database Role.
8. Click **OK** when you have finished configuring the securable column setup. The securable column setup can be found in the project tree under **Security**, **Securable Columns**, **[table name]**, **[securable column name]**.

### APPLYING DATA LEVEL PERMISSIONS

When you have created a securable column setup, you are ready to use it to apply data level permissions to a table. To do so, follow the steps below.

1. Drag and drop a securable column setup on a field in the table in the data warehouse that you want to secure. A secured view is created and can be found under **Security**, **Secured Views**, **[table name]_SV**.
2. If you want to add further permissions to the view, you can drag and drop a securable column setup on the view. The **Add Field** window opens.
3. In the **Field Name** list, select the field that contains the values you want to use in the filter with the securable column setup.

4. Click **OK**. The field is added to the secured view.



# EXPORTING DATA

The nature of a data warehouse is to be a means to an end, for instance getting up-to-date sales numbers every morning or the data needed for financial reporting.

TX DWA supports these scenarios through OLAP cubes or data visualization tools such as QlikView or Qlik Sense. Through the Data Export feature, however, you also have the ability to push parts of - or the entire data warehouse - to another destination, such as an Oracle database or text files.

The feature uses the same external provider concept as the custom data source, which means that we have a framework that makes it possible to add new providers without releasing a new version of TX DWA.

# DATA EXPORT

Data export allows you to push all or some of the content of a data warehouse to another destination, for instance another database.

For each type of destination, you will need a data export provider that can be downloaded and installed through the Custom Component Setup application. Please see the support site for more information: https://support.timextender.com/hc/en-us/articles/209604866

At the time of writing, the following data export destinations are supported:

- Oracle
- SQL Server
- Text files

A data export, or data export destination, and the tables and fields it contains, supports a subset of the features found on regular data warehouses.

Table features:

- Selection rules
- Guard table
- Preview table and the query tool
- Tracing
- Pre- and postscripts
- Description

Field features:

- Include in primary key
- Edit name and data type
- Tracing
- Description

## ADDING A NEW DATA EXPORT

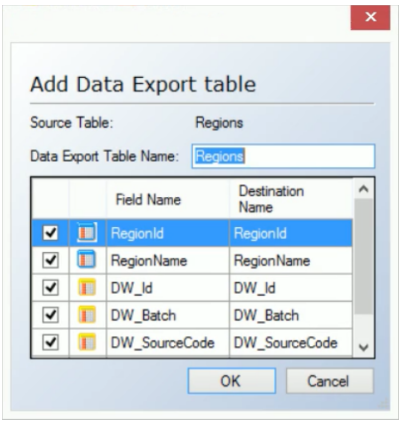
To add a new data export, follow the steps below.

1. Click the **Data Export** tab, right click **Data Exports** and click **Add Data Export**.
2. In **Name**, type a name for the data export destination.
3. In the **Provider** list, click the provider you want to use.
4. In the **Setup Property** list, click **Setup Properties**. In the grid below, enter the setup properties required by the selected data export destination.
5. (Optional) Click **Test connection** to test that the connection properties you entered allow you to connect.

## ADDING A TABLE TO A DATA EXPORT

To add a table to a data export, follow the steps below.

1. On the **Data Export** tab, expand **Data Warehouses** until you have located the table you want to add. Drag the table from **Tables** under **Data Warehouses** and drop it on **Tables** under the relevant data export under **Data Exports**.
   The **Add Data Export Table** window opens.




2. (Optional) In **Data Export Table Name** type a name for the table when used in the data export.
3. Clear the selection for fields you do not want to include in the table on the data export. Click a name in the **Destination Name** to edit the name as it appears on the **Data Export**.
4. Click **OK**.

### ADDING ALL TABLES AND/OR VIEWS FROM A DATA WAREHOUSE TO A DATA EXPORT

To add all tables and/or views from a data warehouse to a data export, do as follows. On the **Data Export** tab, expand **Data Warehouses** until you have located the data ware-house that contains the tables or views you want to add. Drag **Tables** or **Views** under the data warehouse – or the entire data warehouse – and drop it on the relevant data export under **Data Exports**.

TX DWA will add all views and all tables with all fields to the data export with the following exceptions:

- Tables or views that have already been added to the data export will be ignored.
- Tables or views that are not visible on the data warehouse in the currently selected pro-ject perspective will not be added.
- Tables or views that are not visible on the data export in the currently selected per-spective, will be ignored if you add them again.

### PREVIEWING A TABLE

Once the data export has been executed, the regular Preview Table command can be used to view the content of the table.

Click on **Destination** in the **Instance** list to see the data stored in the data export and **Source** to view the data in the table in the data warehouse.

# ONLINE ANALYTICAL PROCESSING (OLAP)

TX DWA includes support for designing Online Analytical Processing (OLAP) cubes that end users can browse through Excel or specialized business intelligence front-ends. OLAP utilizes the fact- and dimension tables you have created in your data warehouse. See Dimensional Modeling to learn more about how you structure a data warehouse.

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.

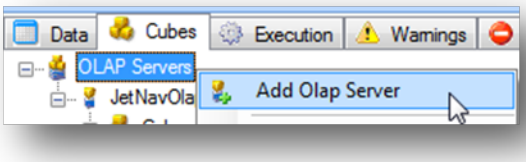
A cube consists of a number of dimensions and measures. The dimensions determine the structure of the cube, and the measures represent the numerical values. You can, furthermore, define hierarchies within a dimension by using dimension levels.

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.

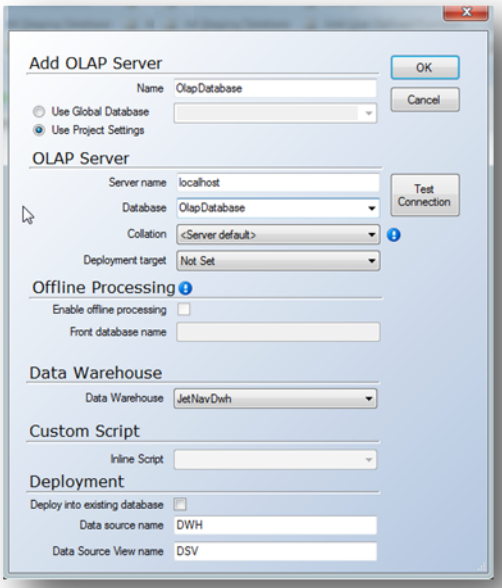
## ADDING AN OLAP SERVER

To use OLAP, you will first need to add an OLAP Server.

1. On the **Cubes** tab, right-click **OLAP Servers**, and then select **Add OLAP Server**.




A window with the following selections will appear:




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** box, type the name of the OLAP database server.
4. In the **Database** box, type a name for the database.
5. In the **Collation** list, click 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.

**Note:** If you choose to delete the OLAP database later, it 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.

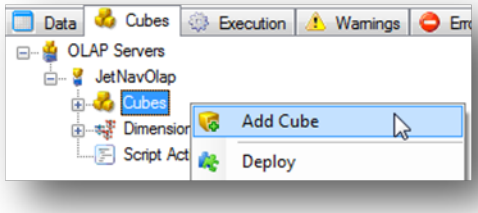
# CUBES

Cubes are the cornerstone when presenting data through OLAP. Cubes are typically built around central functions in the company, such as sales, finance, inventory etc.

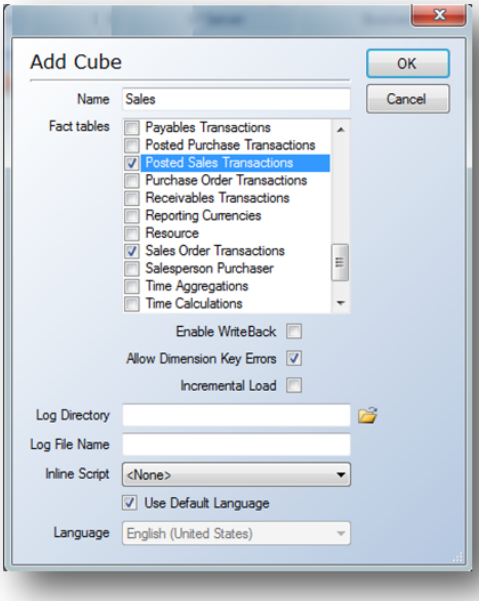
## ADDING OLAP CUBES

To add a cube, follow the steps below.

1. On the **Cubes** tab, expand the OLAP server, right-click **Cubes** and click **Add Cube**.




The **Add Cube** window opens.




2. In the **Name** field, type a name for the cube.
3. In the **Fact Table** list, select on the table(s) from the data warehouse you want to use for the cube.
4. If you want end users to be able to change cube data while they browse it, select **Enable Writeback**. Any changes the end users make are saved in the write-back table.

> **Note:** You can only utilize write-back if the front-end application supports it.

5. If you want to continue processing the cube even if dimension key errors occur, select **Allow Dimension Key Errors**. When you allow dimension key errors, all errors are reported to a log and you need to specify where you want to store the log. Click the folder icon besides **Log Directory**. This opens a new window. Navigate to the folder you want to store the log in and click **OK**. In the **Log File Name** box, type a name for the log file.
6. Click**OK** to add the cube.

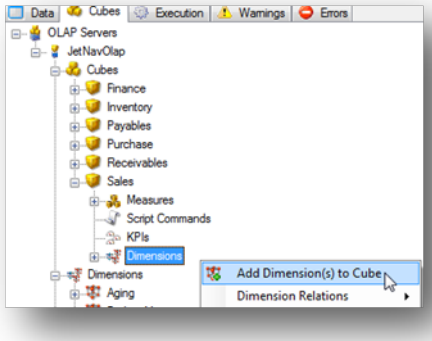
### ADDING A SINGLE DIMENSION TO A CUBE

1. Expand **Dimensions** and drag-and-drop the dimension to the cube.
2. Set the relationship to the fact table in the cube. See Adding Dimension Relationships.

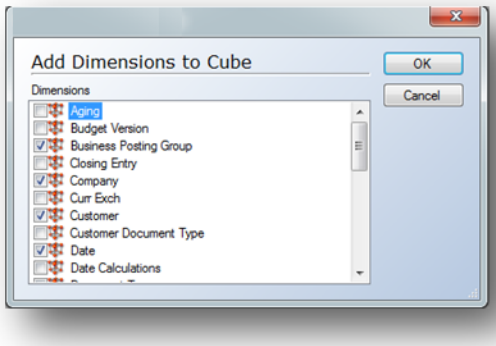
### ADDING MULTIPLE DIMENSIONS TO A CUBE

You can also add multiple dimensions to a cube in one operation.

1. On the **Cubes** tab, expand the relevant OLAP server, expand **Cubes**, and then expand the cube you want to add dimensions to.
2. Right-click**Dimensions**, and then select **Add Dimension to Cube**.




The **Add Dimensions to Cube**window opens with all the dimensions in your project listed.




3. Select the dimension or dimensions you want to add, and click **OK**.

You must then set the relationship to the fact table in the cube. See Adding Dimension Relationships.

### ADDING 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.

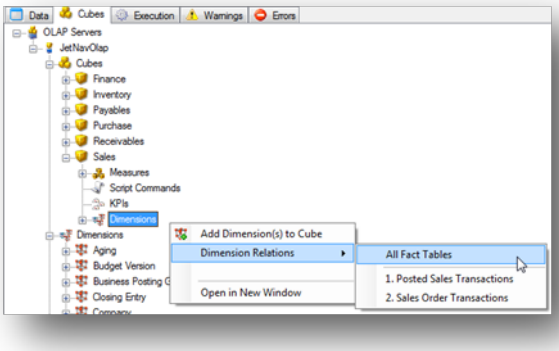
When you have added a role-playing dimension to a cube, you should follow the steps described above to add the dimension and then specify a new name for the dimension to distinguish it from the other dimensions of the same type. Then define the relationship to the proper field in the fact table.

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.

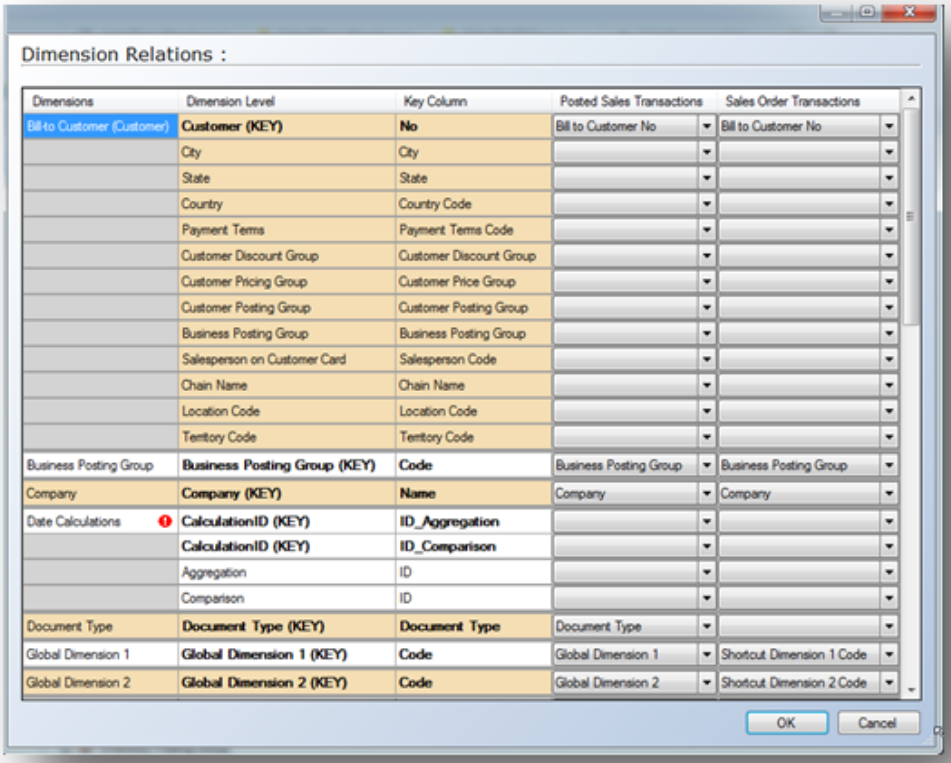
### ADDING DIMENSION RELATIONSHIPS

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 the relevant cube. Expand the cube, right-click **Dimensions**, click **Dimension Relations** and click **All Fact Tables**.



2. The **Dimension Relations** window opens.




The table contains the following columns:

| Column | Description |
|---|---|
| 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 |

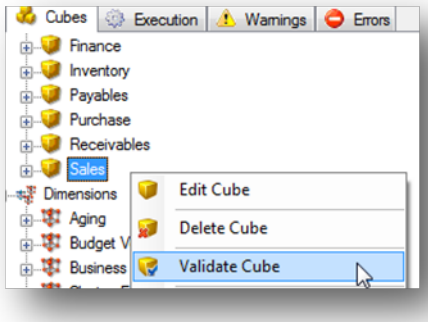
3. In the Fact Table column(s), select the field where you will join the dimension key with the dimension. The dimension levels in **Bold** are the required values to be set for each dimension.
4. Click **OK** when you have created all of the necessary relationships.

### VALIDATING CUBE AND DIMENSIONS

To validate a cube or a dimension, follow the steps below.

1. On the **Cubes** tab, expand the OLAP server that contains the cubes you want to validate, and then expand **Cubes**.
   - OR -
   On the **Cubes** tab, expand the OLAP server that contains the dimensions you want to validate and then expand **Dimensions**

2. Right-click the cube or dimensions you wishto validate, and then select **Validate Cube/Dimension**.



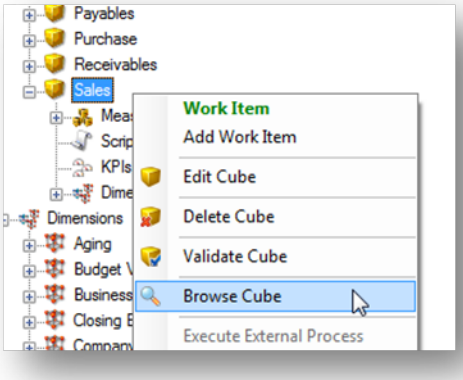

If the cube or dimension is valid, an OK message is displayed. Click **OK** to close the message dialog. If the cube or dimension is invalid, a message is displayed outlining the changes that you need to make.

## THE CUBE BROWSER

The Cube Browser allows aTX DWA user to browse a cube from within TX DWA 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 TX DWA.

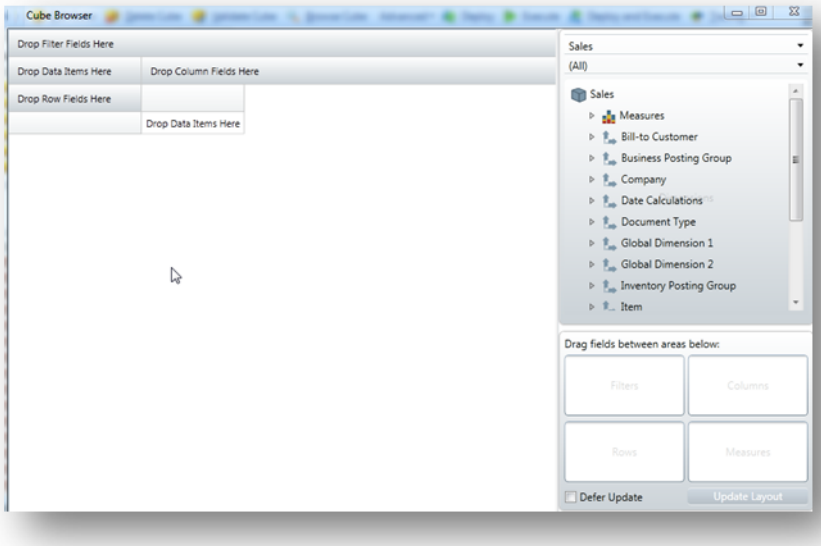
### LAUNCHING THE CUBE BROWSER

Locate the cube that will be browsed on the **Cube** tab, right-click the cube, and select **Browse Cube**.

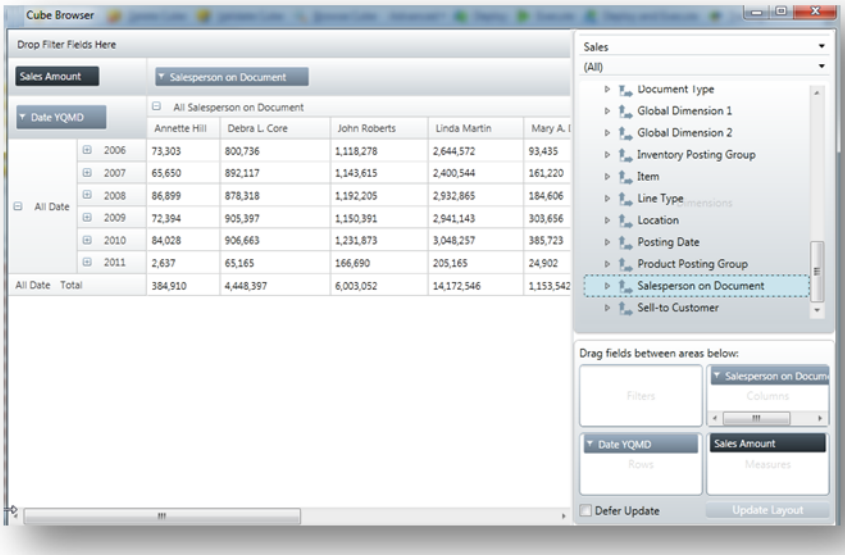



### USING THE CUBE BROWSER

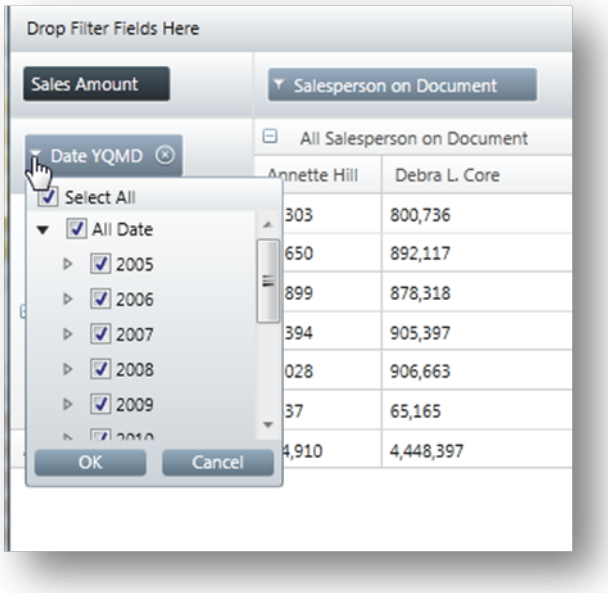
The user interface is similar to pivot tables in Excel. Measures and dimensions are dragged from the list on the right and dropped in either the boxes in the lower right-hand corner or directly onto the workspace pane on the left.



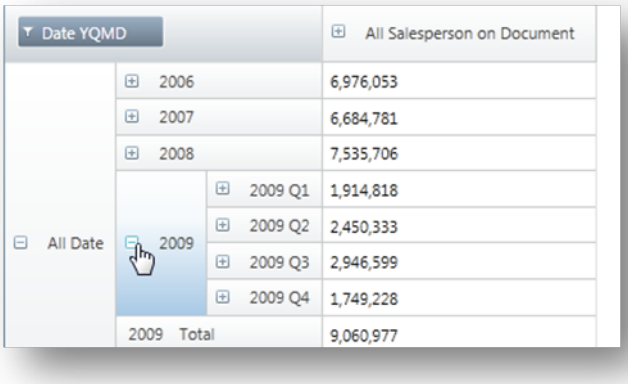

The following report was created from the Sales cube for NAV by dragging the **Date** dimension into the **Rows** box, the**Salesperson on Document** dimension into the **Columns** box, and the **Sales Amount** measure into the **Measures** box.




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.




Dimensions can also be expanded and collapsed by clicking the plus and minus signs respectively.




## CUBE WRITEBACK

Cube Writeback is a feature that allows users to update or add data to the cubes through the front-end.

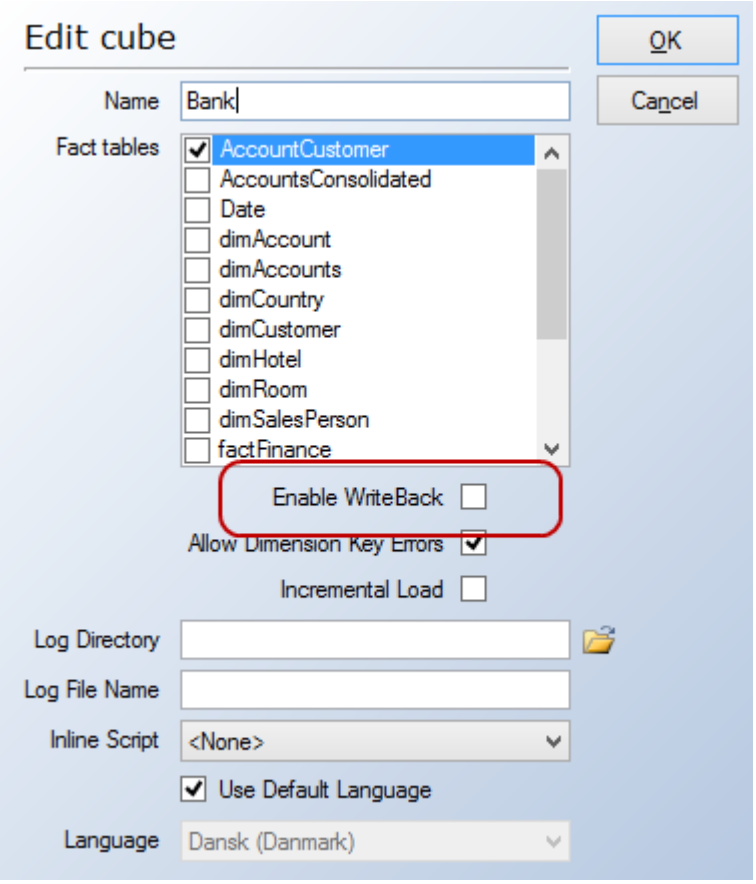
When Writeback is enabled for a cube, it will be enabled for all measure groups in that cube.

The structure of the fact tables will be maintained and updated when the structure of the cube changes.

It is important to understand how writeback works before implementing it. While this document does not target its full scope, it warrants a thorough understanding. Otherwise, users will find themselves performing actions that are incomplete.

To enable this feature, right-click the cube you want to utilize writeback on, and click **Edit Cube**.

In the dialog that opens, check **Enable WriteBack**, and click **OK** to close the dialog.




When writeback has been enabled, a writeback table will be created in the data warehouse on the next deployment. Whenever anything is written to a measure, it will be added to the writeback table. Data will never be written directly to the original fact table.

For the best results, design a separate cube specifically targeted for cube writeback.

## OFFLINE CUBE PROCESSING

TX DWA supports functionality that allows cubes to be processed without being taken offline. Normally, when a cube is processed and rebuilt in SQL Server Analysis Services, it is taken offline during the duration of the processing and is made unavailable to end-users.

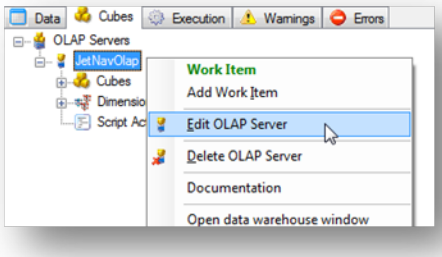
SQL Server Enterprise Edition allows the cube to be kept online, but if you do not run this edition of SQL Server, the Offline Cube Processing feature of TX DWA allows you achieve the same result.

Enabling Offline Cube Processing means that the cubes can be updated throughout the business day without disturbing end-users. While the cube is being processed, the users will

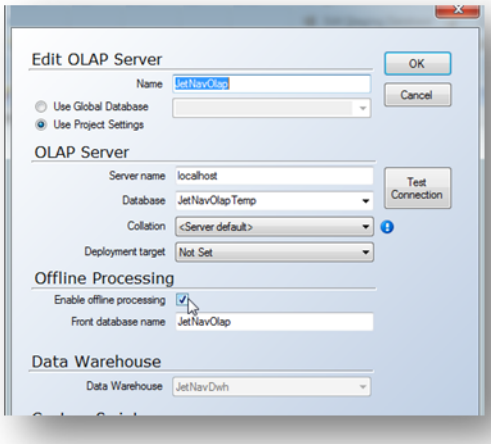
have access to the original version of the cube, which is replaced with the new version of the cube once processing has been completed.

### ENABLING HIGH AVAILABILITY CUBE PROCESSING

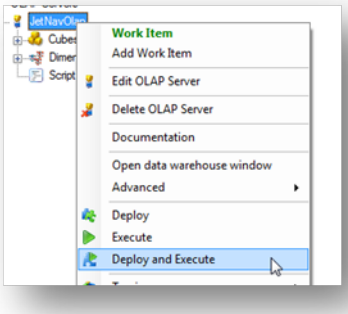
1. On the **Cubes** tab, right-click the name of the OLAP database, and select **Edit OLAP Server**.



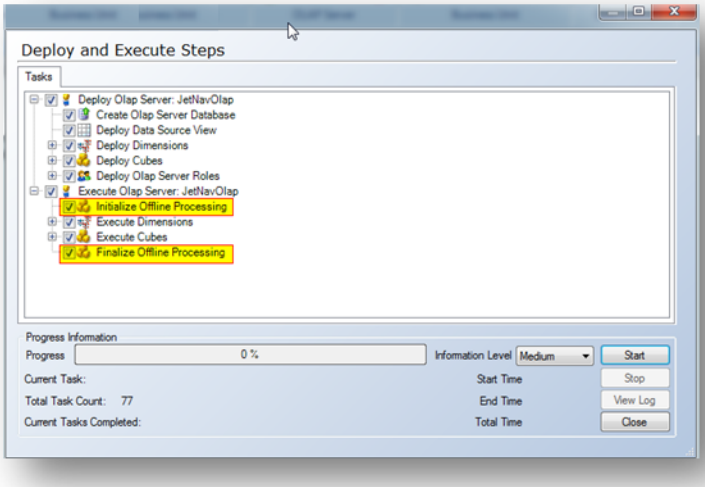

The Edit OLAP Server window opens.




3. Rename the **Database** 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. Select **Enable Offline Processing**.
5. Type in the name of the database that will be used by end-users in **Front Database Name**.
6. Click **OK.**
7. Right-click the OLAP database and click **Deploy and Execute**.




7. You will notice two new steps under **Executed OLAP Server** that handles Offline Cube Processing. These are **Initialize Offline Processing** and **Finalize Offline Processing.**




8. Click the **Start** button to begin processing the cubes. Users will now be able to access the cubes as they are being processed.

# DIMENSIONS

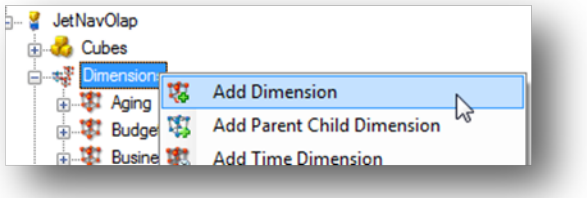
Dimensions define how a user looks at data in a cube. Dimensions are created independently of a particular cube and can be used within several cubes at the same time. The same dimension can also be utilized several times within the same cube, which is referred to as a role-playing dimension. A common 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 or ragged hierarchies.

### CREATING REGULAR DIMENSIONS

1. On the Cubes tab, expand the relevant OLAP server, right-click**Dimensions**, and then select **Add Dimension**.




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


3. In the **Unknown Member** list, select **Visible** to apply an Unknown Member to dimension keys in the fact table with no matching dimension members. 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 setting in TX DWA. An example could be a Salesperson Code that exists in the sales transactions fact table but does not exist in the Salesperson dimension table.

4. Next to the **Type** box, click the ellipsis **(...)** and select the type of dimension you want to create. You can leave the **Type** blank for regular dimension types, which are the most common type.

| Type | Description |
|---|---|
| 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** box, type a name for the **All** Member. This is left blank by default, which means that Analysis Services creates 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. Optional: In the **Description** box, type a describtion for the dimension.
7. Click **OK** to add the dimension.

When you have added a dimension, you also have to add at least one dimension level. The **Add Dimension Level** dialog is displayed when you click **OK** to add a dimension. For more information, see Adding Dimension Levels.

Once you have created a dimension, you can use the dimension in several cubes at the same time. Dimensions that are used in more than one cubes at a time are known as role playing dimensions.

## PARENT-CHILD DIMENSIONS

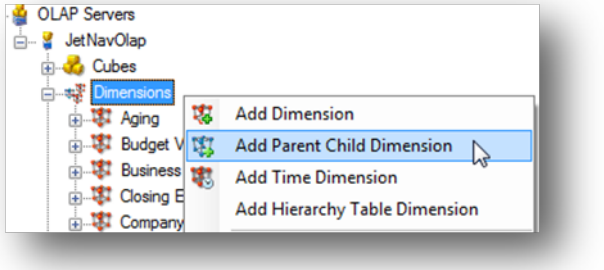
Parent-child dimensions are used to create unbalanced hierarchies where the branches descend to different levels, and where the parent and the child exist in the same table. Typically, parent-child hierarchies are used for creating organizational hierarchies.

### 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.

To create a parent-child dimension, follow the steps below.

1. Expand **OLAP Servers**, expand the relevant OLAP server, right-click **Dimensions** and click **Add Parent-Child Dimension**.




   The **Add Parent-Child Dimension** window opens.


2. In the **Name** box, type a name for the dimension.
3. In the **Unknown Member** list, select **Visible** to apply an Unknown Member to dimension keys in the fact table with no matching dimension members.
4. In the **Table** list, select the main table of the dimension.
5. In the **Key Column** list, select the key column of the child table. This column identifies each member of the dimension.
6. In the **Parent Column** list, select the key column of the parent field. This column identifies the parent of each member.
7. In the **Lay-out** list, select how you want the dimension level displayed to the end user. The following options are available:

| Setting | Description |
|---|---|
| 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 |

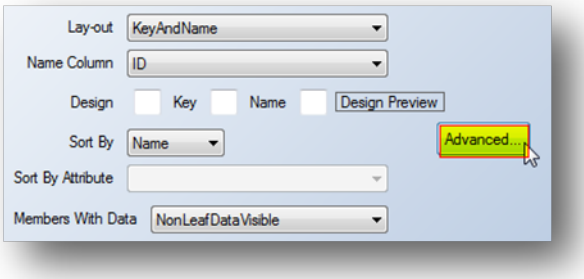
8. In the **Name column** list, select the column that provides a meaningful value to the user. This field is only available if you have selected the **Name**,**KeyAndName**,or**NameAndKey** layout.
9. In the **Design** fields, specify which separator to use in the front-end application to separate Key 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 **Design Preview**.
10. In the **Sort By**list, select whether you want the values sorted by Key or Name.

11. 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**.

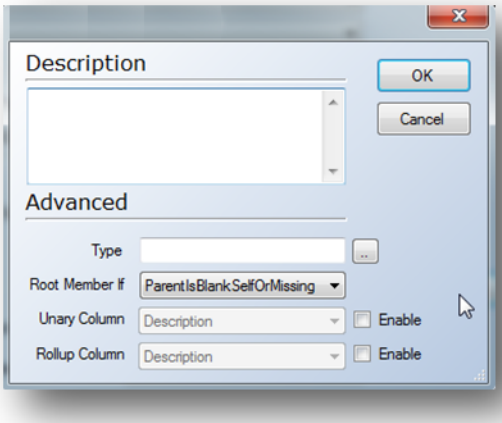
**Note:** When you create the parent-child dimension based on a consolidation table, you will typically use a **Sort By Attribute**. You therefore need to create a Sort order dimension level where the key column is **Sort Order**. Then, you must enable Unary column and Roll up column on the dimension. You can then set the parent-child dimension to **Sort By Attribute**.

### DEFINING ADVANCED PARENT-CHILD DIMENSION SETTINGS

1. To access advanced settings for Parent-Child dimensions, click **Advanced...** in the **Add/Edit Parent-Child Dimension** window.




The **Advanced** window opens.




2. Next to the **Type** box, click the ellipsis **(...)** and select the type of dimension you want to create. You can leave the **Type** blank for regular dimension types, which are the most common type. For a list of possible dimension types, see Creating Regular Dimensions.
3. In the **Root Member If** list, select one of the following options that controls when the dimension is the root member:

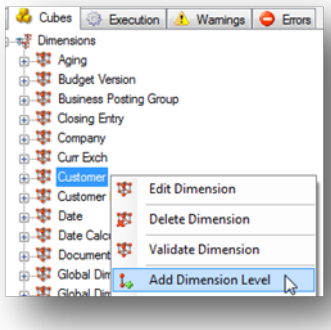
| Option | Description |
| --- | --- |
| ParentIsBlank | Hides the root if the member is a null, a zero or an empty string |
| ParentIsBlankSelfOrMissing | Hides the root if the member is a null, a zero, an empty string, if the parent is missing, and if the member itself is a parent |
| ParentIsMissing | Hides the root if the parent is missing |
| ParentIsSelf | Hides the root if the member itself is a parent |

4. In the **Unary Column** list, select the column that contains the unary operators that are used in this dimension level. If you You have to select **Enable** to select from this list.
5. In the **Roll-up Column**list, select the column that contains the roll-up values used in this dimension level. You have to select **Enable** to select from this list.

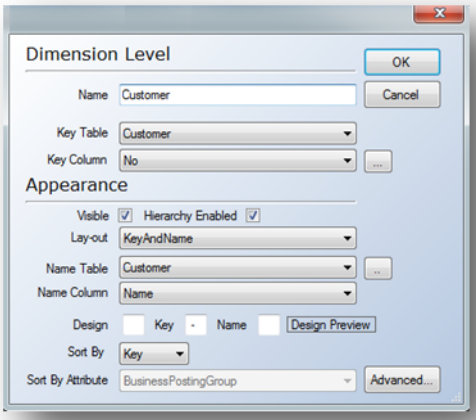
### ADDING DIMENSION LEVELS

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 want to add a level to. Then expand **Dimensions**, right-click the dimension and click **Add Dimension Level**.




The **Dimension Level** window opens.




2. In the **Name** box, type a name for the dimension level.
3. In the **Key Table** list, select the table in the data warehouse to add the dimension from.
4. 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.
5. Select **Visible** if you want the level to be displayed in the front-end application.
6. In the **Layout** drop-down, select the way that the dimension level should be displayed to users. The options are:

| Setting | Description |
|---|---|
| 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 |

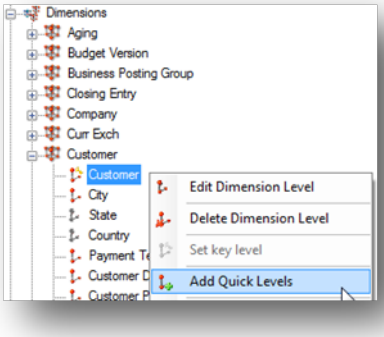
7. In the **Name Table** list, select the table that provides a meaningful name to the user. To set the Name Table value to the same value as the Key Table value, click the ellipsis button.
8. In the **Name Column** list, select the column that provides a meaningful value to the user.
9. In the **Design** fields, type the separators 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 review the design of the layout, move the pointer over **Design Preview**.
10. 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**.

11. 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 AttributeKey or AttributeName.
12. 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 before this change takes effect in the front-end.

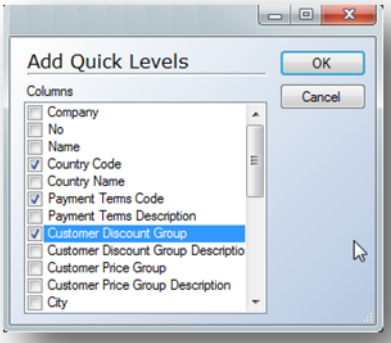
### ADDING QUICK LEVELS

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 relevant OLAP server, expand **Dimensions**, right-click the parent-child dimension or key level on the dimension to which you want to add a level and click **Add Quick Levels**.




2. Select the columns you want to use as levels from the source table in the data warehouse, and then click **OK**.

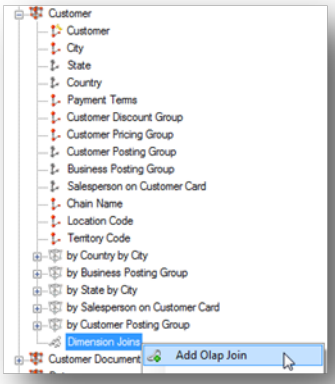



The levels you have added are now available when you create a hierarchy. The OLAP database must be deployed and executed before this change takes effect in the front-end.

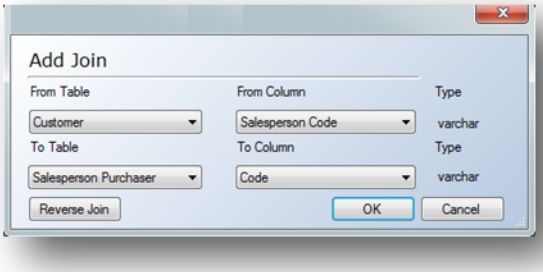
## ADDING DIMENSION JOINS

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 want to modify, expand **Dimensions**, expand the dimension to which you want to add a join, right-click**Dimension Joins**, and then select **Add Olap join**.




   The **Add Join** window opens.

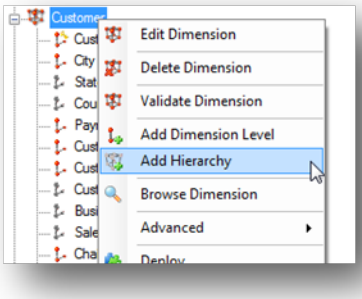



2. In the **From Table**list, select the table *from* which you want to create a join.
3. In the **To Table** list, select the table *to* which you want to create a join.
4. In the **From Column** list, select the column from which you want to create a join. The column's data type is displayed next to the list. You can only make joins between fields with compatible data types.
5. 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.
6. If you want to reverse the direction of the join, click the **Reverse Join** button.
7. Click **OK**. The dimension join is displayed in the **Dimension Joins** folder in the **Dimensions** tree.

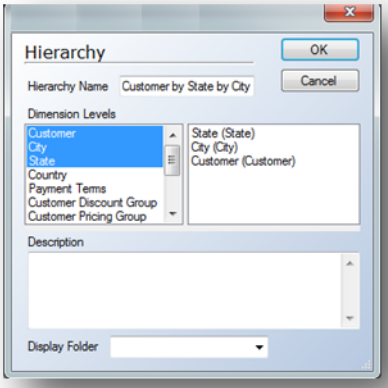
## ADDING DIMENSION HIERARCHIES

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 relevant OLAP server, expand **Dimensions**, right-click the dimension to which you want to add a hierarchy and then select **Add Hierarchy**.




   The **Hierarchy** window opens.




2. In the **Hierarchy Name** field, enter a name for the hierarchy. The name cannot be the same as the name of a dimension level.
3. 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
4. In the **Description** field, type a description of the hierarchy. This field is optional.
5. In the **Display Folder** list, select the folder where the hierarchy is displayed by the front-end application. This is optional.
6. 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.

### ADDING A TIME DIMENSION

Date dimensions are based on time or date tables, so you have to create a time or date table in the data warehouse before you can create a time dimension. For more information, see Adding Date Tables.

To add a date dimension on your OLAP server based on a date table, follow the steps below:

1. On the cubes tab, on you OLAP server, right click **Dimensions** and click **Add Time Dimension**. The **Add Date Dimension** window opens.
2. Type a **Name** for your Date Dimension, click the date table you want to use as a basis for the date dimension in the Table list and click **OK**.
3. The date dimension will then be created based on the data table and appear in the OLAP tree under **Dimensions**. It includes date, week, month, quarter, half year and year dimension levels, in both fiscal year and non-fiscal year variations, as well as any custom periods and **Calendar** and **Fiscal Calendar** hierarchies.
4. (Optional) It might be useful to configure any custom period dimension levels to use the name of the custom period as a key. If, for instance, you have defined a number of national holidays across many years, these will then be grouped as op-posed to a hav-ing "unique" yearly holidays for each year. To do so, right click any custom period level and click **Edit Dimension Level** and click **[custom period name]Name** in the **Key Column list**.

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. You can add hier-archies as well.

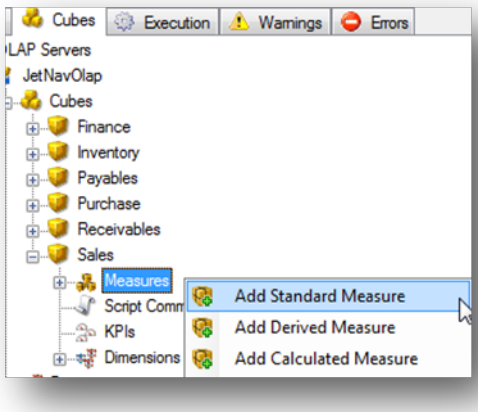
# MEASURES

Measures determine the numerical values of a cube and a cube must contain at least one measure. You can define the following three types of measures:

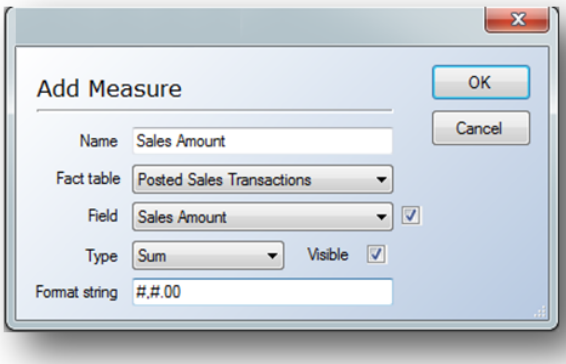
- **Standard measures** obtain their values directly from a column in a source fact table.
- **Derived measures** are derived before aggregation or summing of columns. This means that they are calculated when the cube is processed and are stored in the fact table. You can use standard arithmetic operators 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.

## ADDING STANDARD MEASURES

1. On the **Cubes** tab, expand **OLAP Servers**, expand the OLAP server that contains the cube to which you want to add a measure, expand the cube, right-click **Measures** and click **Add Standard Measure**.




The Add Measure window opens.



2. In the **Name** box, type a name for the measure.
3. In the **Fact Table**list, select the fact table that you want to use for the measure.
4. In the **Field** list, select the field that you want as the measure. Disable this field by clicking the box next to it.
5. In the **Type** list, select the preferred aggregation method. You have the following options:

| Aggregation Method | Description |
|---|---|
| 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 |

6. Select **Visible** if you want the measure to be displayed in the front-end application.
7. In the Format string field, specify how you want the numeric results displayed. You have the following options:

| Format String | Description |
|---|---|
| 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 |

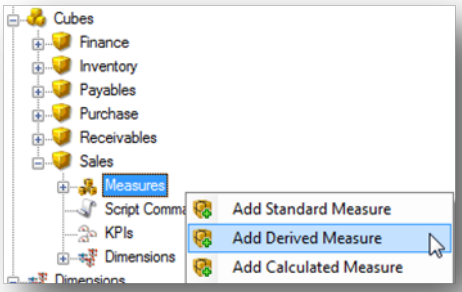
Below are examples of what the output will look like for various combinations of the format strings:

| Format String | Output |
|---|---|
| None | 1234567.89 |
| #,# | 1,234,567 |

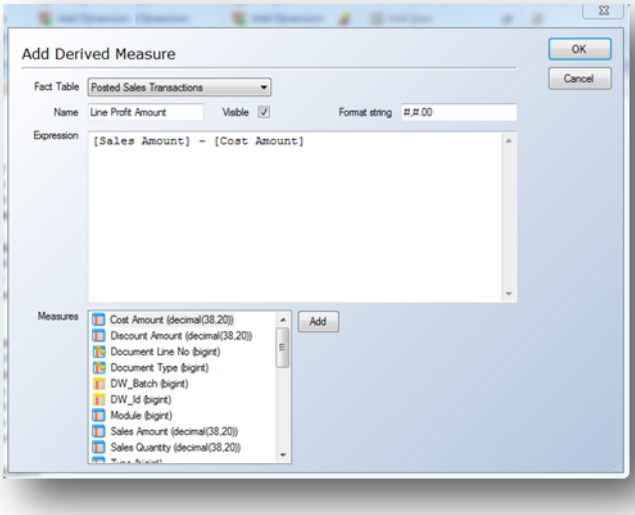
| | |
|---|---|
| #,#.00 | 1,234,567.89 |
| #,#% | 1% |
| #,#.0% | 1.2% |
| Percent | 1.23% |

## ADDING DERIVED MEASURES

1. On the **Cubes** tab, expand **OLAP Servers**, expand the OLAP server that contains the cube to which you want to add a measure, expand the cube, right-click **Measures** and




click **Add Derived Measure**.
The **Add Derived Measure** window will appear.



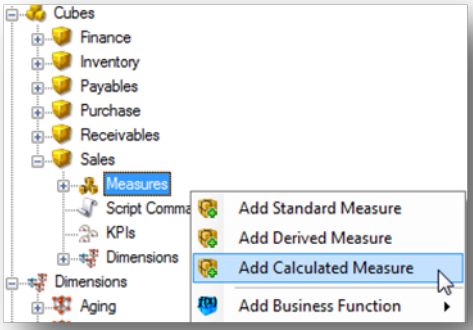

2. In the **Fact Table** list, select the fact table that you want to use for the measure.
3. In the **Name** box, type a name for the derived measure.
4. Select **Visible** if you want the measure to be displayed in the front-end application.
5. In the **Format String** box, type how you want the numeric results displayed. The format is the same as for standard measures. See Adding Standard Measures.
6. 7. In the **Expression** box, enter 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 click**Add**. Your selections will be added to the expression, where they can be combined with mathematical operators to achieve the outcome you desire.

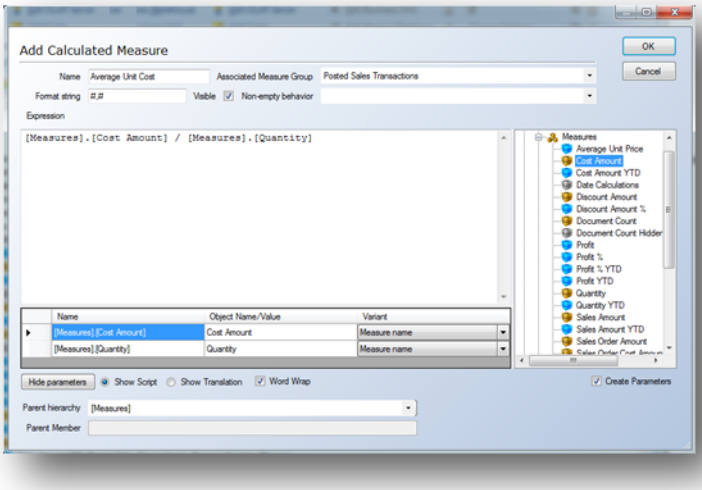
7. Click **OK** to add the derived measure.

## ADDING CALCULATED MEASURES

1. On the **Cubes** tab, expand **OLAP Servers**, expand the OLAP server that contains the cube to which you want to add a measure, expand the cube, right-click **Measures** and click **Add Calculated Measure**.




The Add Calculated Measure will open.




2. In the **Name** box, type a name for the calculated measure.
3. Select **Visible** 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. The format is the same as for standard measures. See Adding Standard Measures.
5. Optional: In the **Non-empty Behavior** list, select the measure or measures used to resolve NON EMPTY queries in MDX.
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.

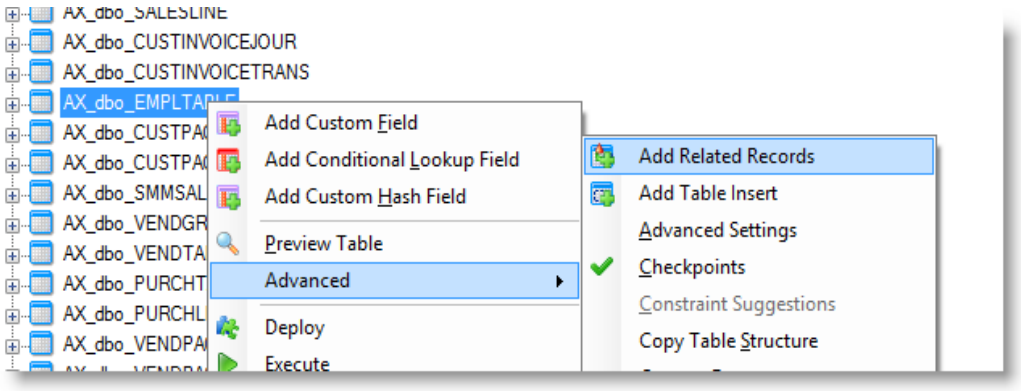
# HANDLING EARLY ARRIVING FACTS

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.

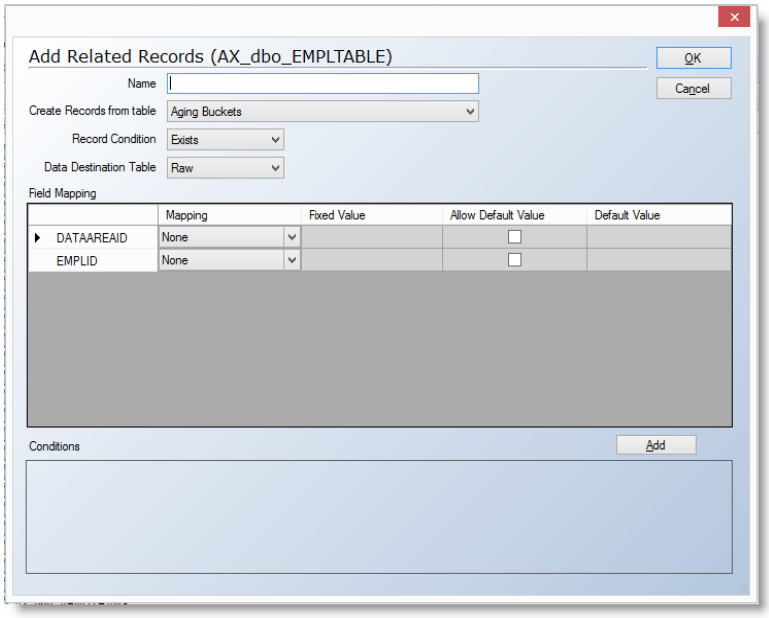
In TX DWA,it is possible to handle these "early arriving facts" in such a manner that they will show at least partial information until the data source is properly updated with all of the normal dimension information. This prevents information from being placed into the "Unknown" member when the data 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

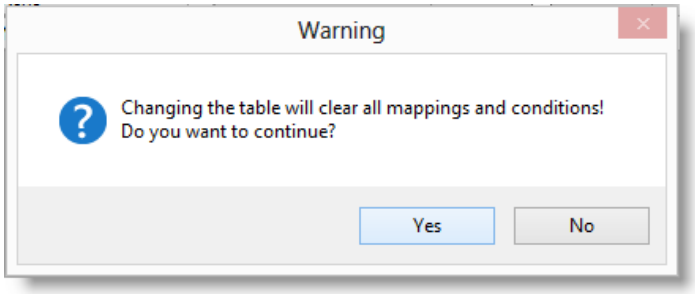
1. On the **Data** tab, identify the dimension table to which relevant values from the transaction table should be added, right-click the table name, and go to **Advanced -> Add Related Records.**



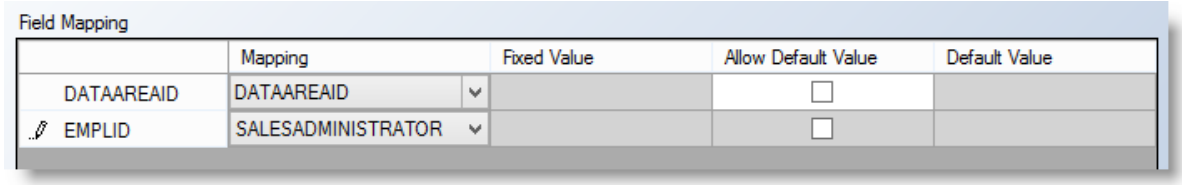

The **Add Related Records** window opens.



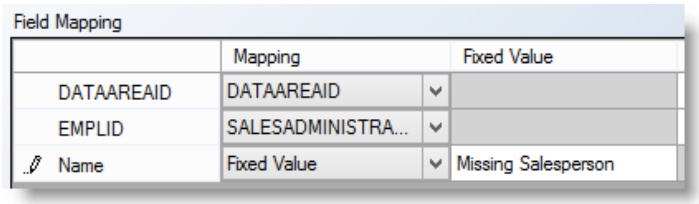

2. In **Name**, type a descriptive name ofr the **Add Related Records** rule that is currently being created.
3. In the **Create Records from Table**list, select the transaction table that will identify the table from which to bring in potential new values. A window may appear stating that all mappings and conditions will be cleared. Click **Yes**.



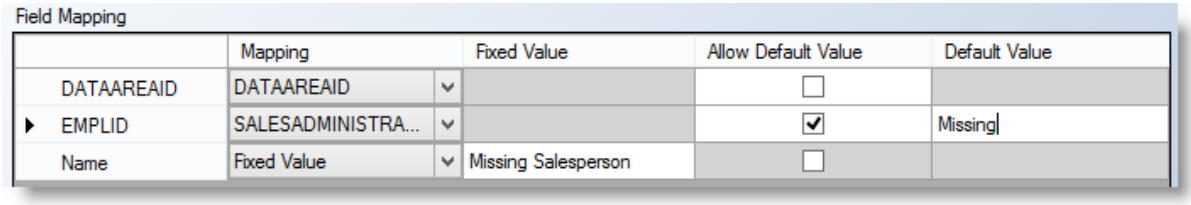

4. In the **Record Condition** list, select the option to determine when data will be inserted into the dimension table if new values are found in the transaction table. The most common option is **Not Exist**, which will add in values that do not currently exist in the dimension table.
5. Select the **Data Destination Table** to insert the values into. The default option is the Raw table.
6. In the **Field Mapping** table, specify 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.



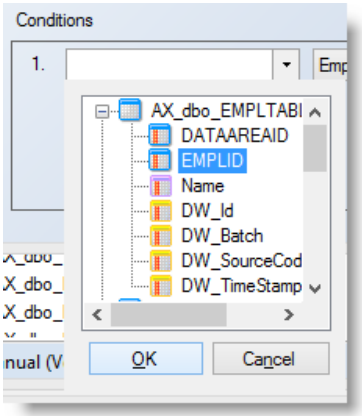
7. It is possible to add in fixed values for fields in the dimension table that the transaction may not have data for. In the example below, the fixed value "Missing Salesperson" will be added in the **Name** field for all Salesperson Codes added from the transaction table. This is achieved by selecting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



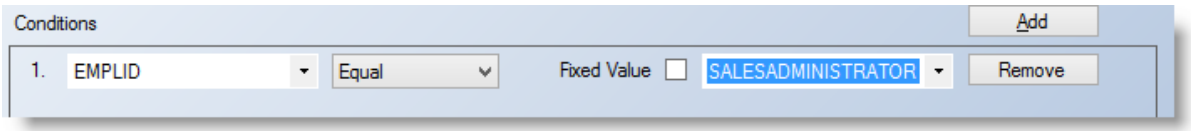

8. If desired, a default value can be inserted instead of bringing in the values that exist in the transaction table. This could be used to assign fixed values to all data brought in for early arriving facts. This is achieved by clicking the checkbox in the **Allow Default Value** column and typing the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



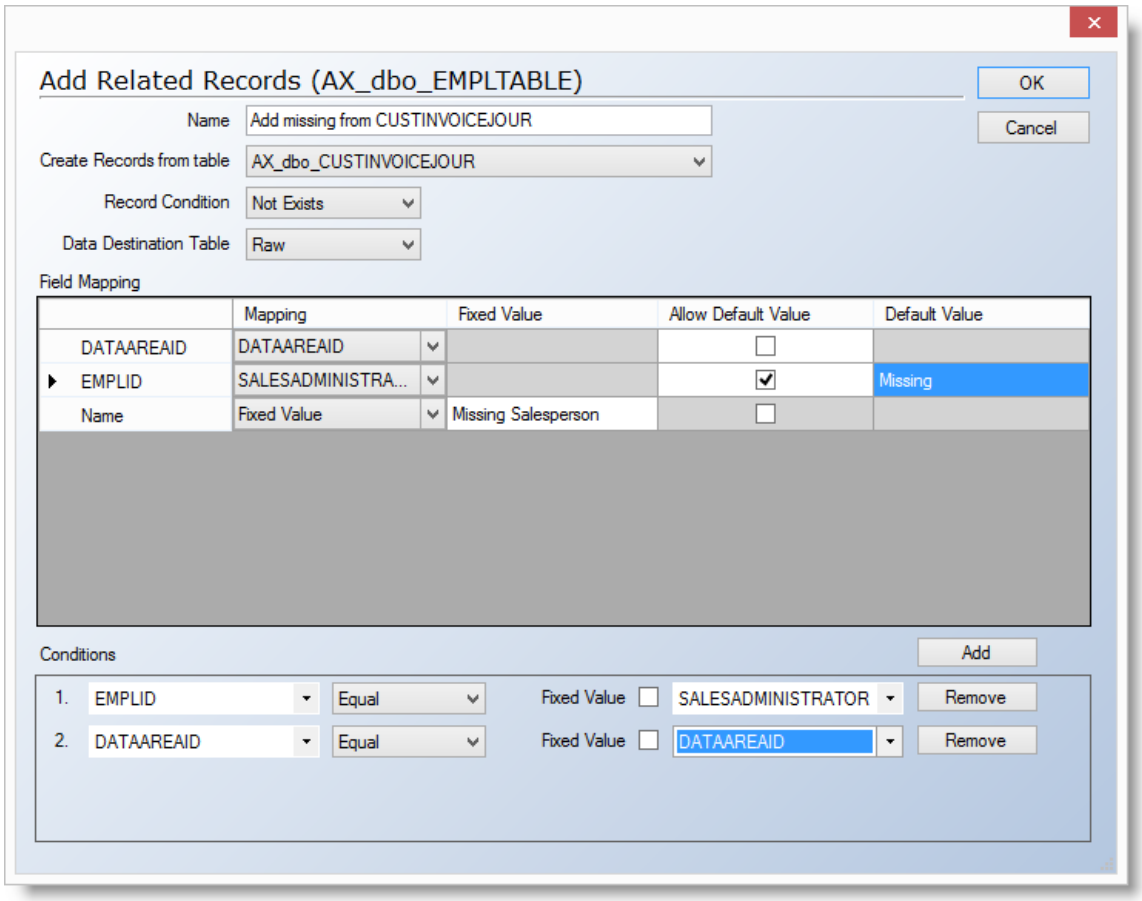

9. The last step is to define the relationship between the two tables. Click the **Add** button in the **Conditions** section.
10. Select the first field to join in the dimension table (**Code**), and click **OK.**




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.**

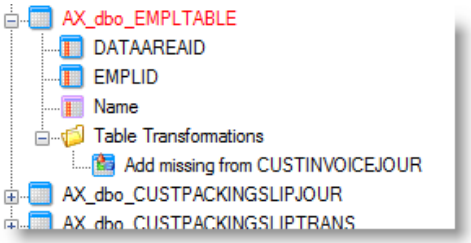


13. Repeat steps 9 through 12 for any additional joins that need to be made (such as Company). The final result will look similar to the screenshot below.




Click **OK** when finished to save the settings, and close the **Add Related Records** window.

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**.




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.

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 |

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."

# 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. TX DWA will 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 above was tracked using Type II functionality.

| Customer No | Name | City | State | Version | From Date | To Date |
|---|---|---|---|---|---|---|
| 123 | ABC Consulting | Portland | OR | 1 | 1/1/1900 | 9/18/2012 |
| 123 | Acme Consulting | Portland | OR | 2 | 9/18/2012 | 12/31/9999 |

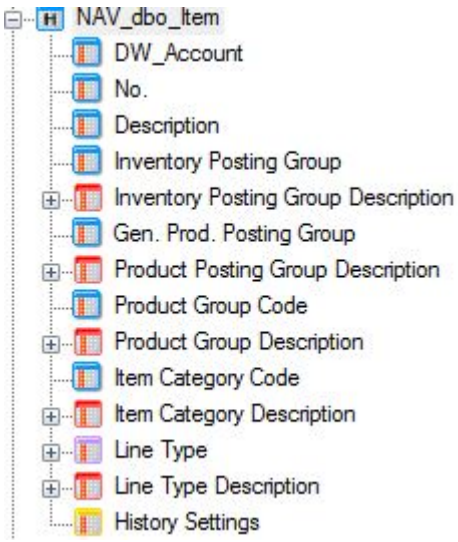
## IMPLEMENTING TYPE II SLOWLY CHANGING DIMENSIONS

The steps below will explain how to utilize slowly changing dimensions - also known as History in TX DWA - in a project. 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

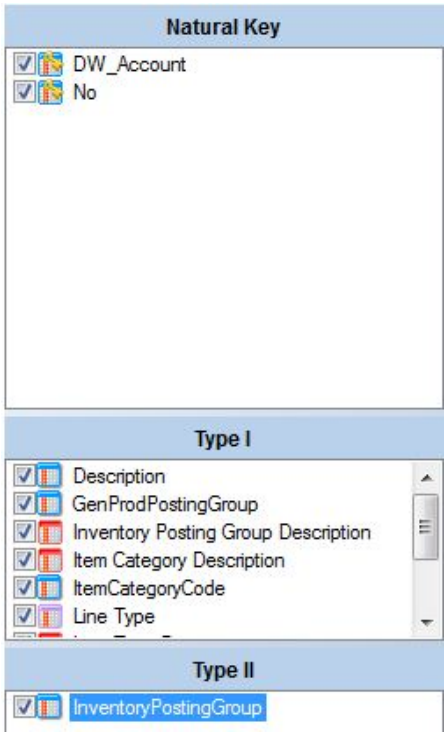
organization wishes to track sales for this item under both the historical Inventory Posting Group as well as the new one.

### ENABLING SLOWLY CHANGING DIMENSIONS ON THE DIMENSION TABLE

1. Identify the table where historical changes need to be tracked. Right-click the table, click **Table Settings**.
2. Click the **History** tab and select **Enable history**. If an error icon appears next to the setting, it means that another setting needs to be changed to enable SCD. Move you mouse over the error icon to see the error message.
3. Click **OK**.
4. The table icon will now be overlaid with an "H" to make it easy for you to identify it as a table with history/slowly changing dimensions enabled. Expanding the table will show a node named **History Settings**.



5. Click **History Settings**. The Slowly Changing Dimensions configuration options appear in the right-hand side of the screen.




6. This screen is broken down into three sections:
   - **Natural Key:** Selected all fields that represent the natural key of the table. Any primary keys defined on the table is automatically selected. In the example this is the "DW_Account" and "No." fields
   - **Type I:** Here, all fields that should not have history tracking enabled are listed. 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:** Select all fields that should have history tracking enabled.
7. Deploy and execute the table to have the initial history data stored.

**Note:** 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.

### EXAMPLE

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 | Product Group | Product Group | Item Category | Item Category | Line Type | Line Type | DW_Id |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CRONUS EXT U... | 1000 | Bicycle | FINISHED | FINISHED | RETAIL | Retail | | Null | | Null | 2 | Item | 1 |
| CRONUS EXT U... | 1000 | Bicycle | RESALE | RESALE | RETAIL | Retail | | Null | | Null | 2 | Item | 145 |

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 screenshot 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.

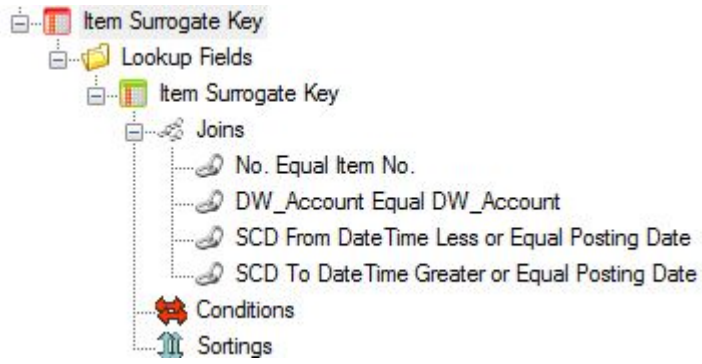
### BRINGING THE SURROGATE KEY TO THE TRANSACTION TABLE

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 screenshot 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 TX DWA, 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 TX DWA, follow the steps below.

1. Right-click the table, click **Advanced** and click **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.




6. Deploy and execute the transaction table to have the new field added and populated.

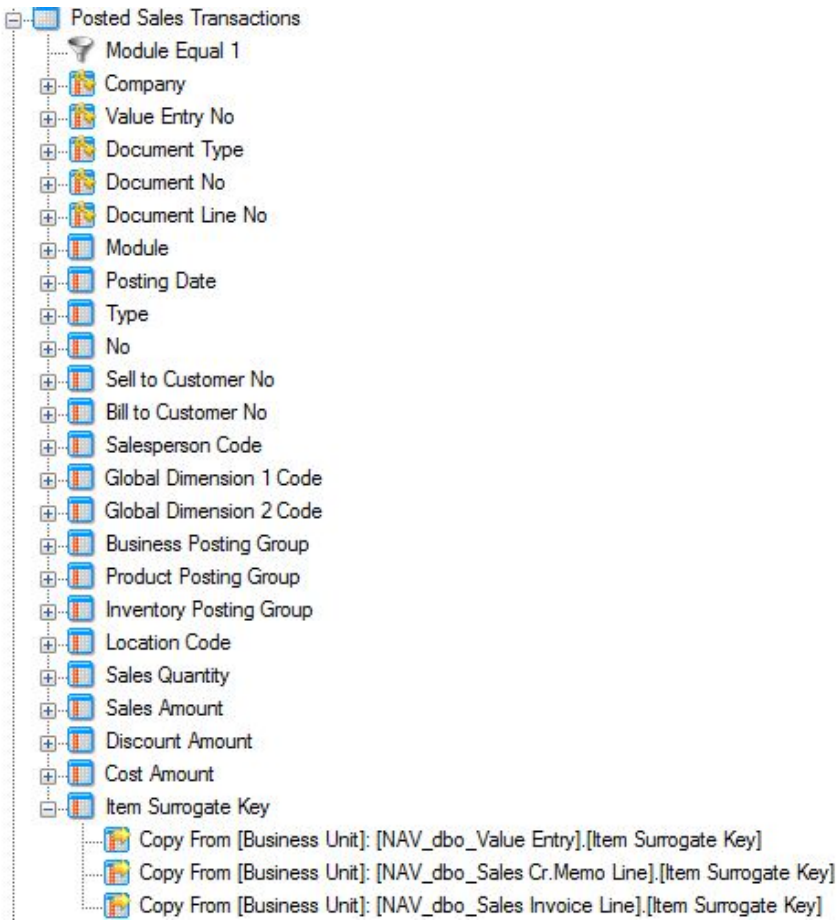
The process above should be repeated for any additional transaction tables where history needs to be tracked.

### MOVING THE SURROGATE KEY FROM THE TRANSACTION TABLE IN THE STAGING DATABASE TO THE 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.

To accomplish this, follow the steps below.

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 the changes to take effect.

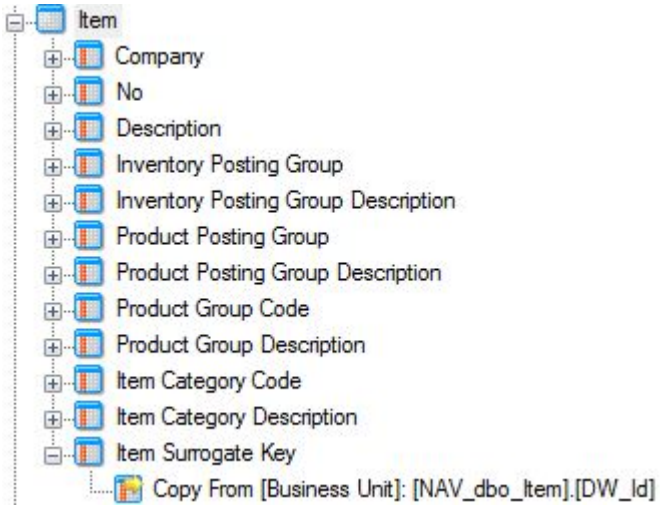



### ADDING THE SURROGATE KEY TO THE DIMENSION TABLE IN THE DATA WAREHOUSE

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 accomplish this, follow the steps below.

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 take effect.



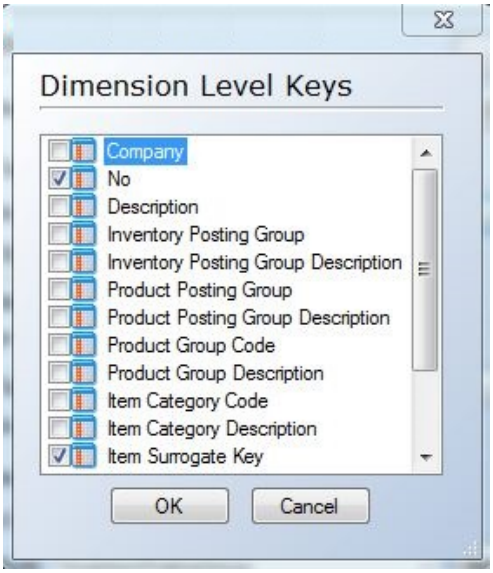

## UPDATING THE DIMENSION KEY

The dimension key should 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.

To accomplish this, follow the steps below.

1. On the Cubes tab, expand **Dimensions**, expand the dimension, and edit the key level (in this example "Item").


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 "Item")




### UPDATING THE DIMENSION RELATIONSHIPS IN THE CUBE

The relationships between the dimension and the transaction table should 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), click **Dimension Relations** and click **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 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 |

# DEPLOYING AND EXECUTING

Before a TX DWA project is deployed and executed, it is simply a meta data model of your data warehouse. Deployment and execution generates and runs the code for extracting, transferring and loading your data as well as creating any OLAP cubes in the project. During development it can also be a good idea to deploy and execute the project to see if everything works as expected.

## DEPLOYING A PROJECT

Deploying a project, or a part of a project, is the process of generating the structure of the staging database and the data warehouse, processing cubes and 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.

Deployment in TX DWA is optimized in two ways: It is managed, i.e. objects are deployed after any objects they depend on, and differential, i.e. only the steps that have changed since the last deployment are deployed again.

## EXECUTING A PROJECT

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

Executing a project involves the following steps:

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.

TX DWA supports managed and threaded execution. This means that TX DWA can execute a project in multiple threads while managing dependencies between objects and optimizing the execution to take the shortest amount of time.

# MANUAL DEPLOYMENT AND EXECUTION

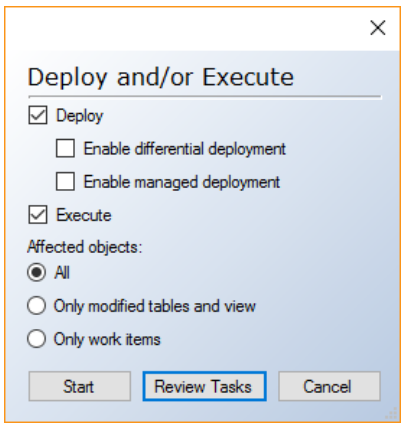
While you are developing and maintaining your project, you may want to deploy or execute the entire project or individual objects to confirm that everything works as expected.

Manual execution is configured in the default execution package, that can be configured just like any other execution package. See Execution Packages for more information.

## DEPLOYING AND/OR EXECUTING INDIVIDUAL OBJECTS

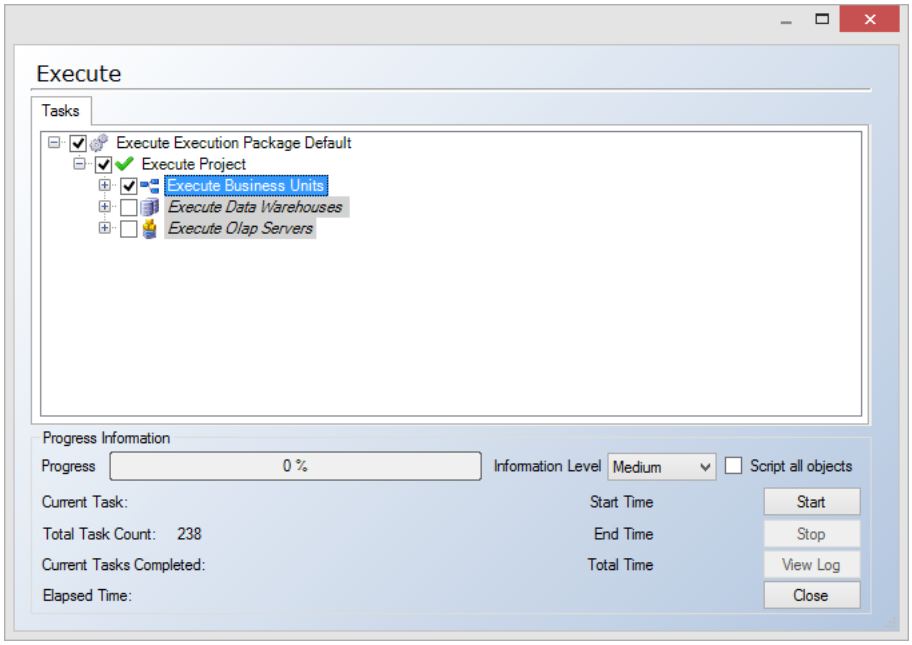
Whether you want to deploy, execute or deploy and execute an object, the steps are similar.

1. Right-click the project or the project element you want to deploy and/or execute, and click **Deploy**, **Execute** or **Deploy and Execute**
   - OR -
   Click the object at press **CTRL + ALT+ D** to deploy, **CTRL + ALT + E** to execute and **CTRL + ALT + K** to deploy and execute. The **Deploy and/or Execute** window appears.




   The settings in the window depends on how you initialized the window – choosing deploy, execute or deploy and execute – and your default settings for deployment.
2. Select **Deploy** to deploy the selected objects. You have two deployment options. Their initial setting is based on your project settings.
   1. Select **Enable differential deployment** to take advantage of TX DWA's differential deployment feature that calculates what steps have changed and need to be deployed and selects only those steps for deployment. When differential deployment is disabled, all steps are deployed.
   2. 
   3. Select **Execute** to execute the selected objects. The execution settings are governed by the default execution package.
   4. Under **Affected objects**, select which of the objects you selected for deployment and/or execution you want to deploy and/or execute. Your options are **All**, **Only modified tables and views** and **Only work items**.
3. Click **Start** to begin the deployment and/or execution process as soon as TX DWA is ready or **Preview Tasks** to review the tasks and settings before you start the process.

4. If you use the differential deployment method, it will take TX DWA a few moments to calculate what steps need to be deployed. If TX DWA does not find any steps that needs to be deployed, you will be asked if you want to save the project as the deployed version. This has to do with the way the scheduler works. It will execute the last deployed version of a project, i.e. if you want the current version to be the one that is executed by the scheduler, it needs to be marked at such.
5. The **Deploy**, **Execute** or **Deploy and Execute** window opens. If you clicked on **Start** earlier, TX DWA will begin the deployment and/or execution process immediately. Otherwise, review the tasks below b




6. Clear the selection for any objects you do not want to deploy, execute or deploy and execute.
7. (Optional) In the **Information level** list, click your preferred level of information during the completion of the tasks. The following options are available:
    1. **None:** Displays no progress information.
    2. **Low:** Displays current task, the total task count, start time, end time, and total time.
    3. **Medium (default):** 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.
    4. **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.
8. (Optional) Select **Script all objects** to make all parts of the SQL script available in the log for debug purposes. This option disables the use of Shared Management Objects to create tables instead of executing "CREATE TABLE" statements against SQL Server since statements executed through Shared Management Objects are not in the log.

9. Click **Start** to deploy, execute or deploy and execute the objects you have chosen. Click **Stop**if you want to halt the processing prematurely.

If there are any errors during deployment or execution, the **Error** window is displayed. Click **Yes** to view the log. The object on which the deployment or execution failed, has an **Error Information** node. Double-click **Error Information** to view an error description.

You will have to wait for TX DWA to finish the tasks before you can continue work on the projects. However, with the Execution Queue, you can execute objects in the background which enables you to continue working on your project while objects are executed. See Executing Objects with the Execution Queue below.

### Customized Code Warnings

TX DWA will display a warning message if you try to deploy a table with customized code where the table has changed in TX DWA since you last customized the code. Often, you need to update the customized code when you have made other changes to the table and the purpose of this feature is to help you catch some errors earlier.

TX DWA displays the warning message when the deployment window opens. Click **OK** to close the message. On the **Customize Code Warning** tab, you can see what tables might have outdated customized code.

If you click **Start** after closing the message, TX DWA will assume that you have things under control and not display the message again for the same issue.

#### REVIEWING THE EXECUTION LOG

On each execution, the execution diagram, message and setup is saved to the execution log.

- To view the execution log for an execution package, click the **Execution** tab, right click the execution package and then click **View Execution History Log**.

#### RESUMING A FAILED EXECUTION

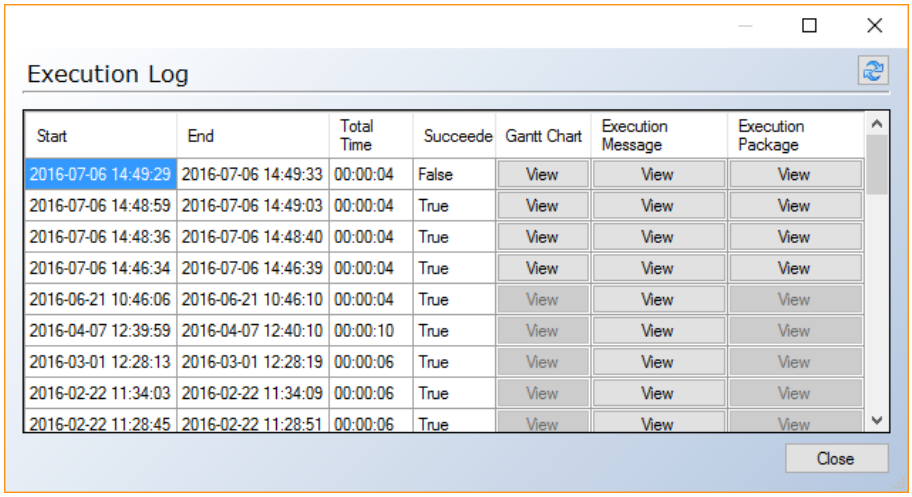
Executions that fail are a fact of life, but restarting an execution that has failed can be very time consuming if the error occured two hours into the execution. Therefore, TX DWA enables you to resume a failed execution from the point of failure.

This is useful since, with managed execution, you cannot always determine in what order TX DWA will execute tables. This means that if an execution fails, a complete restart of the execution is usually the only way to ensure that everything is executed correctly.

You can also configure TX DWA to allow some non-essential data sources to fail without failing the entire execution at the same time. See Allowing a Data Source to Fail

When an execution fails, resume the execution by following the steps below.

1. First, you need to identify the error. On the **Execution** tab, right click the failed execution package, and click **View Execution History Log**. The **Execution Log** window opens.




2. Click **View** in the **Execution Message** column next to the failed execution to display the error message. Close the message and the Execution Log window.
3. Second, you need to solve the error you identified in the execution message. Remember to deploy any changed objects as necessary.
4. Now you are ready to resume the execution. Open the **Execution Log** again as described in step 1.
5. Right click the execution you want to resume and click **Resume Execution**. The **Execute** window opens.
6. Click **Start** to begin the execution.
7. Repeat steps 1-6 if the execution package fails again.

## DEPLOYMENT STATUS REPORT

You can generate a deployment status report that contains a list of the objects to need to be deployed. This can be done on the project level, data warehouses, business units and OLAP servers as well as a remote environment.

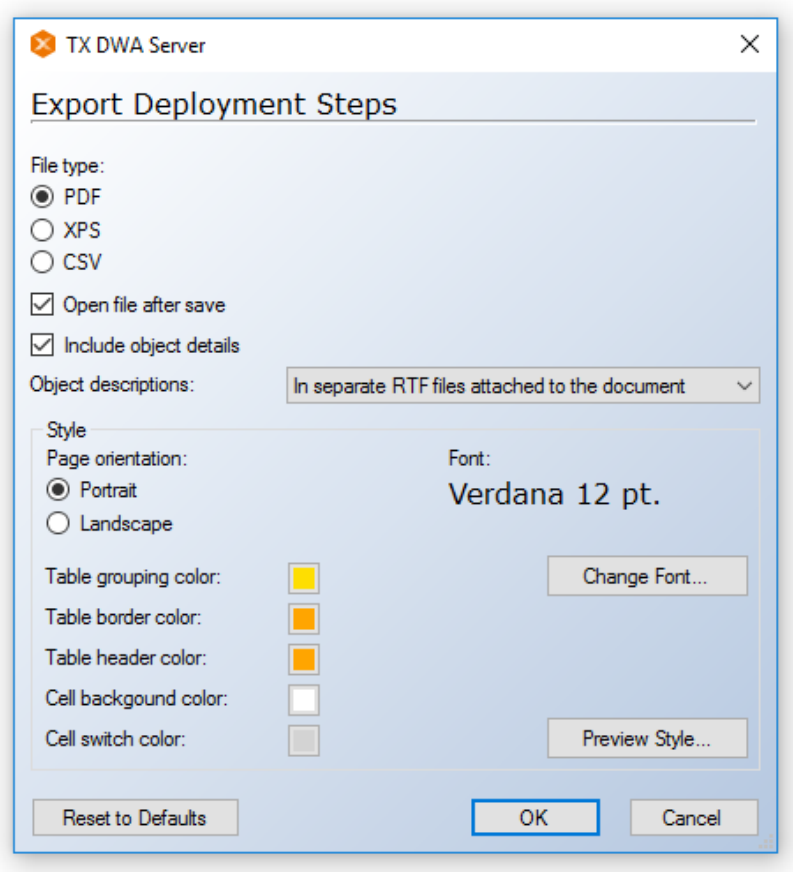
The process is very similar to generating documentation, however, a deployment status report can also be generated in the CSV file format in addition to PDF and XPS.

### GENERATING A DEPLOYMENT STATUS REPORT

To generate a deployment status report for a part of or your entire project, follow the steps below.

1. Right click on the object you want to create the report for and click **Export Deployment Steps**. The option is available on the project level, data warehouses, business units, OLAP servers, data exports and Qlik models.

   The **Export Deployment Steps** window appears.




2. Under **File type**, click the file format you want to use. If you choose CSV,
3. Select **Open file after save** to view the document when TX DWA has generated it. TX DWA shows the document using external viewers that needs to be present on the machine.
4. Select **Include object details** to include details of the objects that the steps that need to be deployed are part of.
5. In the **Object descriptions** list, you can choose how you want to use include the object descriptions. You have the following options:
    1. **In separate RTF files attached to the document**: The descriptions are attached to the document as RTF files. This option is useful if you have used rich text formatting or added pictures to your descriptions, but only available for the PDF file format.

| Table: | AggrTabRegions | Custom Description in attached file: 2.rtf |
|---|---|---|
| Icon | | |
| Table Type | Aggregation | |
| Allows Nulls | Yes (As Project) | |
| Null Check Approach | Field Based (As Project) | |
| Primary Key Behavior | As Project | |
| Field Count | 6 | |
| Fields | [Name], [CountryRegionCode], [DW_Id], [DW_Batch], [DW_SourceCode], [DW_TimeStamp] | |
| Index Count | 0 | |
| Aggregations | Sum([CountryRegionCode]) | |

2. **Only text placed in the document**: The descriptions are included in the documentation as plain text.

| Table: | AggrTabRegions |
|---|---|
| Icon | Σ |
| Table Type | Aggregation |
| Allows Nulls | Yes (As Project) |
| Null Check Approach | Field Based (As Project) |
| Primary Key Behavior | As Project |
| Field Count | 6 |
| Fields | [Name], [CountryRegionCode], [DW_Id], [DW_Batch], [DW_SourceCode], [DW_TimeStamp] |
| Index Count | 0 |
| Aggregations | Sum([CountryRegionCode]) |
| Description | This is an example description |

3. **No descriptions**: The descriptions will not be included in the generated documentation.

6. Under **Style**, you can configure the look of the status report. The settings are saved on a per-user basis and used for this as well as the files generated by the documentation feature.
7. Under **Page Orientation**, select the page orientation you want the documentation to use.
8. Click the color preview next to the different color details to choose a color.
9. Click **Change font...** to choose a font for the documentation. a preview is displayed under **Font**.
10. Click **Preview Style...** to generate and open a sample file with the colors you have chosen.
11. Click **Reset to Defaults** to reset the style settings to their defaults
12. Click **OK**. Wait while the steps are calculated. In the window that then appears, choose a file name and location for the report and click **Save**. The default name is the project name followed by "deployment steps" and a timestamp.

Please note that you will need to have working connections to the relevant databases when you generate the status report. Otherwise, TX DWA won't be able to calculate the differences that indicate that a deployment is necessary.

### GENERATE DEPLOYMENT STATUS REPORT ON REMOTE ENVIRONMENT

In addition to the local project, you can also generate a deployment status report for a project on a remote environment in a multiple environments setup. The report will contain information on the last available version on the environment.

- To generate documentation for a project on a remote enviroment, right click on the project, click on Multiple Environment Transfer, right click the remote environment and click **Export Deployment Status**.

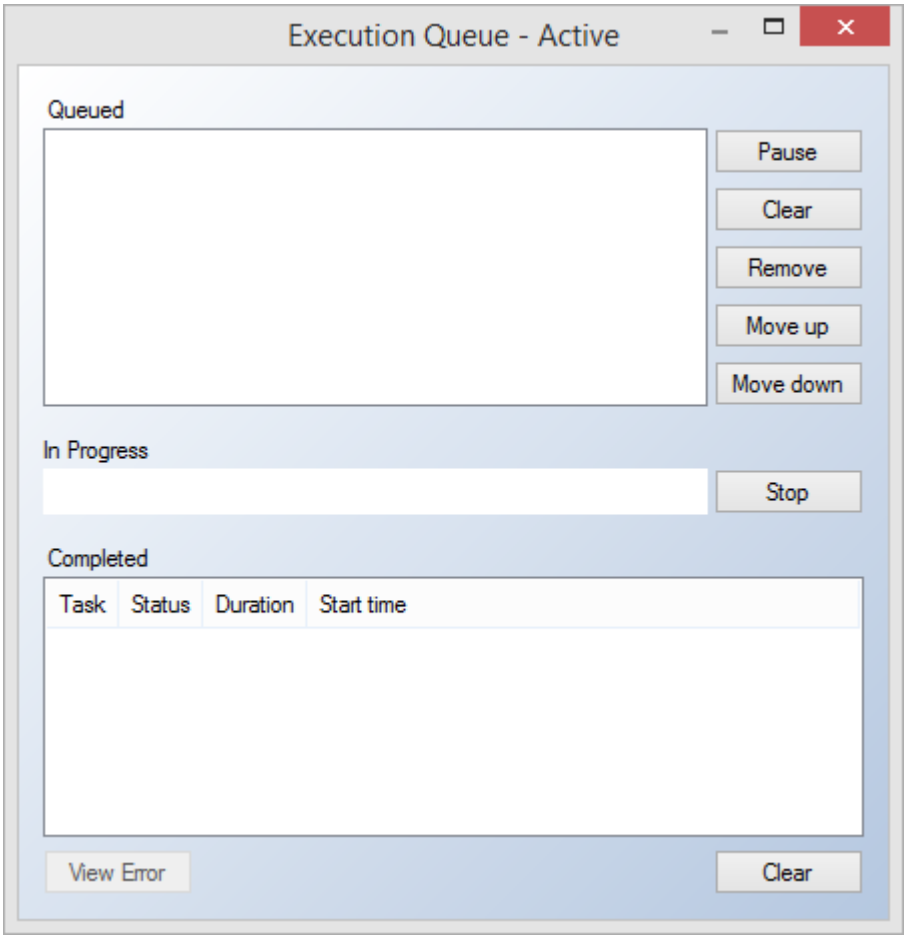
## EXECUTING OBJECTS WITH THE EXECUTION QUEUE

The Execution Queue enables you to continue working while tables or you entire project is executed in the background.

### OPENING THE EXECUTION QUEUE WINDOW

To open the Execution Queue window

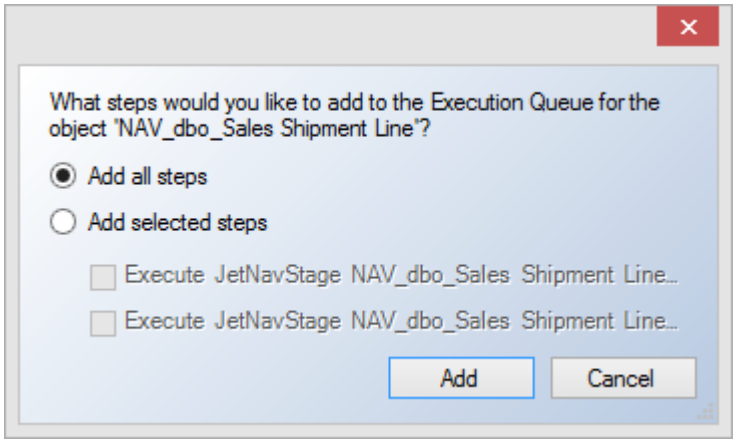
- On the **Tools** menu, click **Execution Queue**.




### ADDING AN OBJECT TO THE EXECUTION QUEUE

Adding an object to the Execution Queue is a simple drag-and-drop operation.

1. Drag-and-drop a table, a perspective, a data warehouse, a business unit, an execution package or another executable object to the Execution Queue window. A window appears to let you choose which execution steps from the object to add to the queue.




2. 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 queued in the Execution Queue. If there is no other items in the queue, the object will be moved to **In Progress** and begin executing immediately.

### 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 pro-gress. 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 execut-ed.

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**: The object was executed without errors.
- **Failed**: The execution was ended prematurely by an error.
- **Stopped**: The execution was stopped by the user before it completed.

You can view error messages for failed objects by selecting the object in the list and 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 DWA, the Execution Queue will be stopped along with the rest of the application.

# SCHEDULED EXECUTION

When your project is ready, you would typically like the entire project, or part of it, to be executed according to a schedule. One common scenario is to execute the project every night to ensure that the business users have updated numbers in the morning.

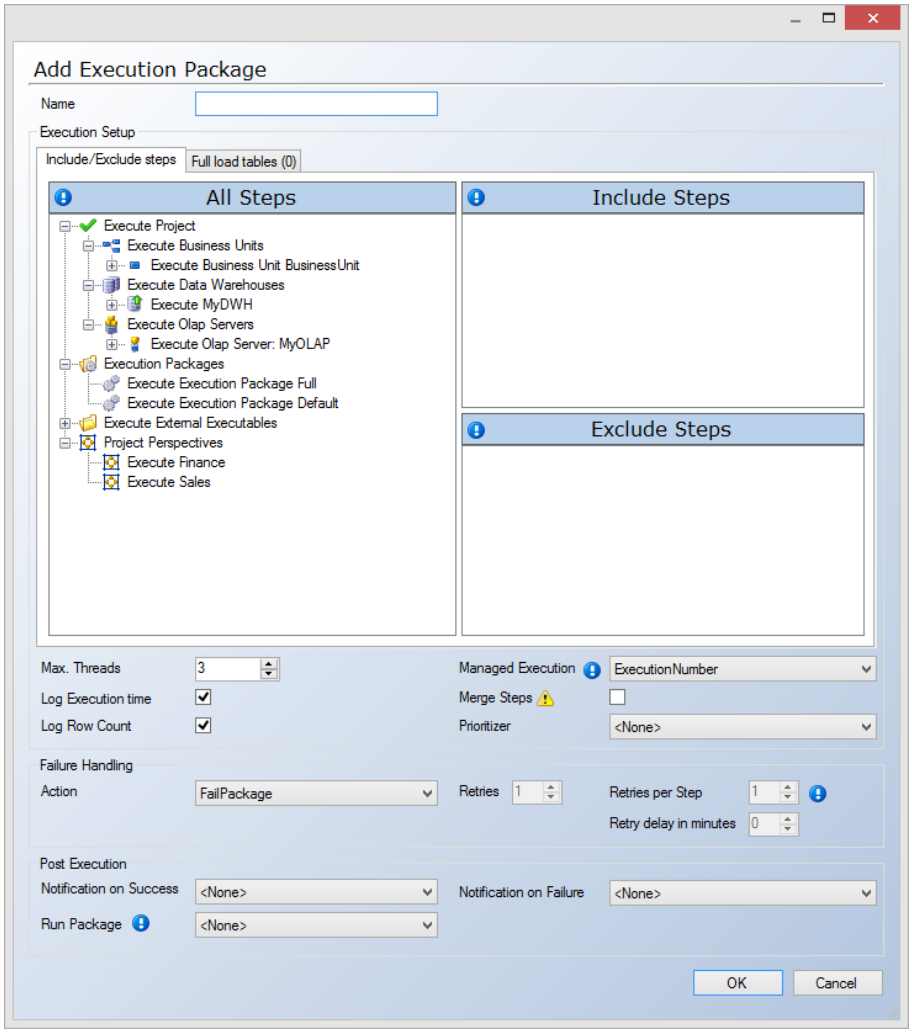
No matter what you needs are, the answer in TX DWA is the execution package. An execution package contains tables, cubes and other objects that will be executed when the package is run. You can schedule the package to be run at specific times and set up notifications to e.g. alert you if a scheduled execution fails. You can also set up conditions that need to be fulfilled for a execution package to run.

**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, the scheduled execution package will most likely fail.

## ADDING AN EXECUTION PACKAGE

An execution package determines which objects in a project will be executed and how. To create an execution package, follow the steps below.

1. On the **Execution** tab, right-click **Execution Packages** and then click **Add Execution Package**. The **Add Execution Package** window appears.




2. Type a **Name** for the execution package.
3. In the **Include/Exclude Steps** tab, choose the steps you want to include in or exclude from the execution package by dragging objects from **All Steps** to **Include Steps** or **Exclude Steps**, respectively. For instance, simply drag the **Execute Project** step to **Included Steps** if you want to execute the entire project when the package is executed. If you want to exclude the entire project except one or more steps, simply drag those steps to **Exclude Steps**. Right click an object and click **Remove Step** to remove a step from the **Include Steps** or **Exclude Steps**.
4. (Optional) In the **Full load tables** tab, drag any incrementally loaded tables, you want to have full loaded in this execution package, from **All Tables** to **Full Load Tables**. You can also drag business units, data warehouses, OLAP servers or the entire project.
5. Enter the maximal number of threads you want to utilize during execution in **Max. Threads**.
6. Clear the **Log Execution time** and/or the **Log Row Count**check boxes if you do not want to log this information.

7. Select a setting for **Managed Execution**. You have the following options:
   - **Disabled**: Managed execution is disabled. Objects will be executed in the order specified in the project tree.

     **Warning:** If you disable managed execution, tables are executed in the order in which they appear in the tree on the **Data** tab. To avoid errors during execution, you must ensure that tables are executed in a logical order. For example, an Order table must be executed before the related Order Detail table. Tables can be moved up and down the tree using drag-and-drop or the keyboard combination ALT+Up/Down.
   - **Execution Number**: When more than one object is ready to executed, TX DWA prioritizes the objects based on their position in the project tree from top to bottom.
   - **Classification**: When more than one object is ready to executed, TX DWA prioritizes the objects based on their table classification. The order will be "Fact Table – Large", "Dimension Table – Large", "Fact Table", "Dimension Table". If two tables have the same classification TX DWA will use the execution number as the secondary criteria.
   - **Execution Time**: When more than one object is ready to executed, TX DWA prioritizes the objects based on their average execution time so that the object with the longest execution time is executed first. If two tables have the same execution time (e.g. in case of new objects), TX DWA will use the execution number as the secondary criteria. When execution time of the objects in the project are known, this option will result in the shortest execution time in most cases.
8. Check **Merge Steps** if you want to treat all individual sub-steps of the chosen steps as one big collection. This can speed up execution.
9. (Optional) In the **Prioritizer** list, click the prioritization you want the execution package to use. For more information on prioritization, see Adding a Prioritization.
10. Under **Failure Handling**, select what **Action**TX DWA should perform if the execution fails. You have the following options:
    - **Fail Package**: When a step fails, the execution is stopped and the package is declared failed.
    - **Retry Step**: When a step fails, the step will be retried until the maximum number of retries for the entire package or the individual step is reached. Enter the maximum number of retries allowed for the package in the **Retries** box and the maximum number of retries allowed for an individual package in **Retries per Step**. Enter the amount of time to wait between retries in **Retry delay in minutes**.
11. Under **Post Execution**, select a Notification on Success and a Notification on Failure. You have to create a notification before it is available from the list. See Adding Notifications below.
12. If you want to run a package after the execution, select the package in **Run Package**.
13. Click **OK** to add the execution package.

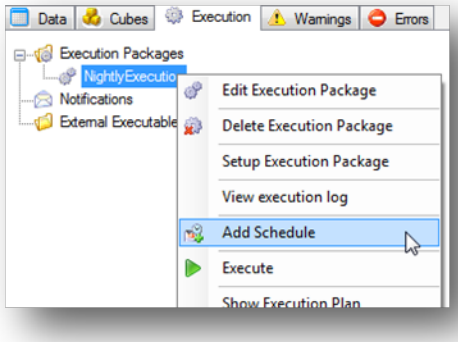
### ADDING AN EXECUTION SCHEDULE

When you specify an execution schedule, the execution process is started automatically at the specified time.

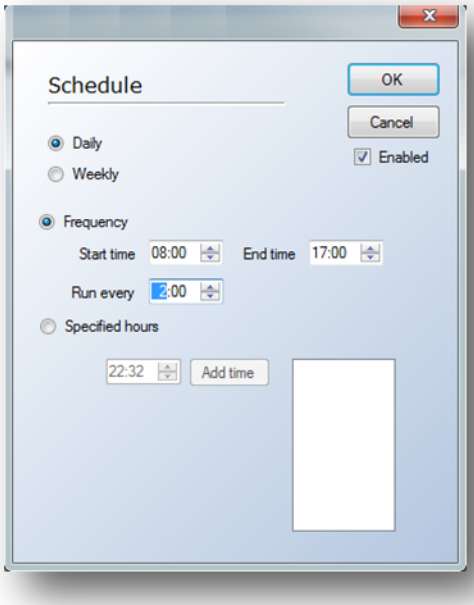
**Note:** Because of the way the scheduler service works, you should make sure to schedule packages at least two minutes apart. If you schedule two packages to be executed at the same time, only one of the packages will be executed.

To add an execution schedule, follow the steps below.

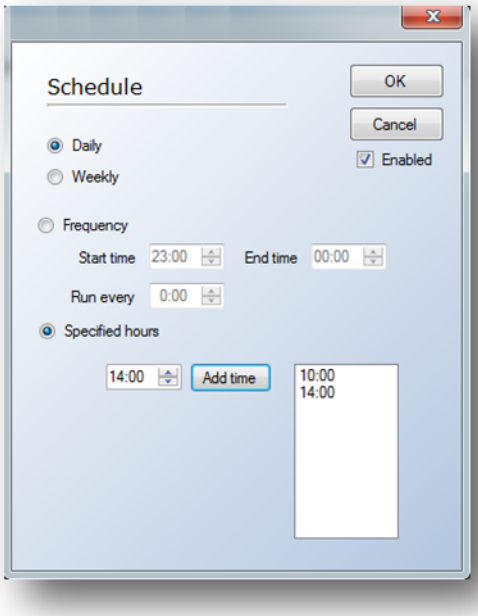
1. On the **Execution** tab, expand **Execution Packages**, right-click the relevant execution package, and click **Add Schedule**.



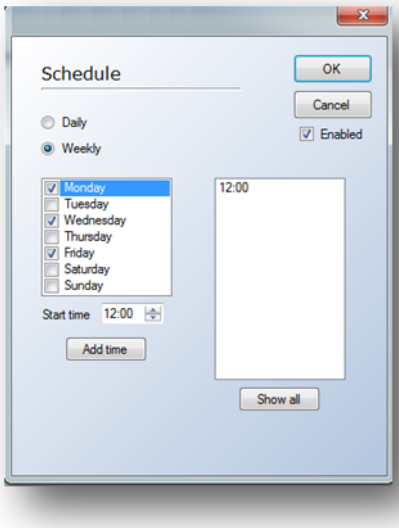

2. You have three different options for specifying the schedule:
   A. To set up multiple daily executions in a specific time frame, click **Daily** and then click **Frequency**. In the **Start time** field, enter the start time of the time interval. In the **End time** field, enter the end time of the interval. In the **Run every** field, specify the number of hours and minutes between each project execution.



B. To set up one or more daily executions on specified hours, click **Daily** and then click **Specified Hours**. Enter the exact time for the execution to run and then click **Add Time**.




C. To set up weekly executions, click **Weekly**. Select the day(s) when the project should execute, enter 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. To view the entire weekly schedule, click **Show All**.



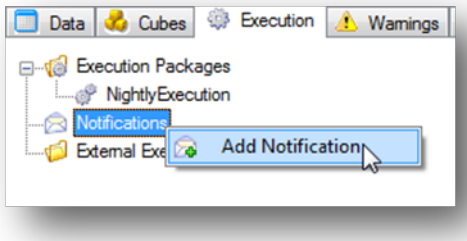

3. Select **Enabled**to activate the schedule, and then click **OK**.

**Note:** You can get an overview of the previous executions of an object by right-clicking on it an clicking on **View Execution overview Log**. Additionally, all events that happen during execution are registered in the Windows Event Log.

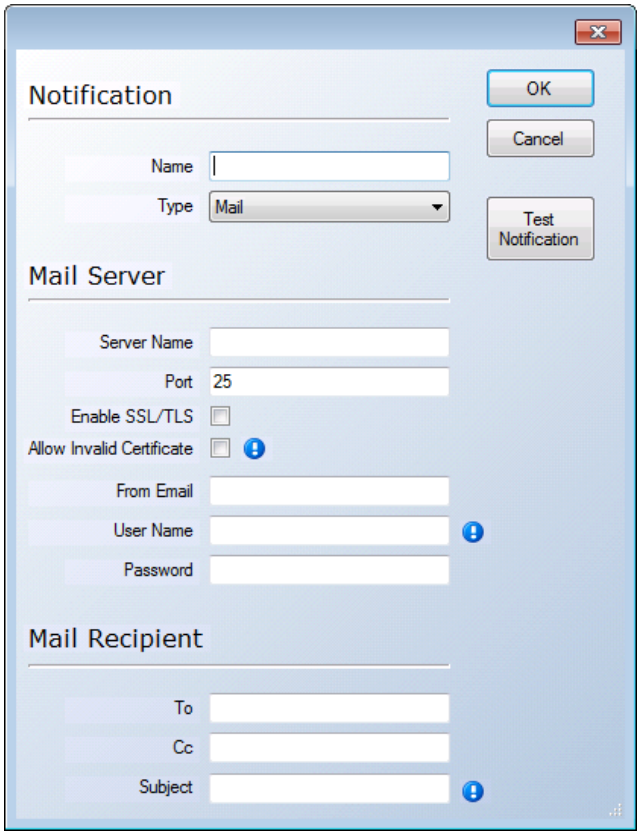
### ADDING A NOTIFICATION

Notifications can be used to 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 click **Add Notification**.




   A window opens to let you add a notification.




2. Enter a **Name** for the notification and select the **Type** the type of notification you want to create. You have the following options:

| Option | Description |
|---|---|
| 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 |

3. Enter the information for the SMTP server you want to use under **Mail Server**.
4. Enter the e-mail addresses of the recipients and a **Subject** under **Mail Recipient**. In the subject, you can use the following variables:
   - **%Project%**: The name of the project .
   - **%Status%**: The status of the execution (Success / Fail).
   - **%ExecutionPackage%**: The name of the execution package.
5. Click **OK**.

The notification can now be selected when you create an execution package.

### ADDING A PRIORITIZATION

You can choose what objects to prioritize during a managed execution. This is useful if you, for instance, only have a small timeslot for extracting data from a source, or I you would like to have a certain cube ready for the users as early as possible during execution.

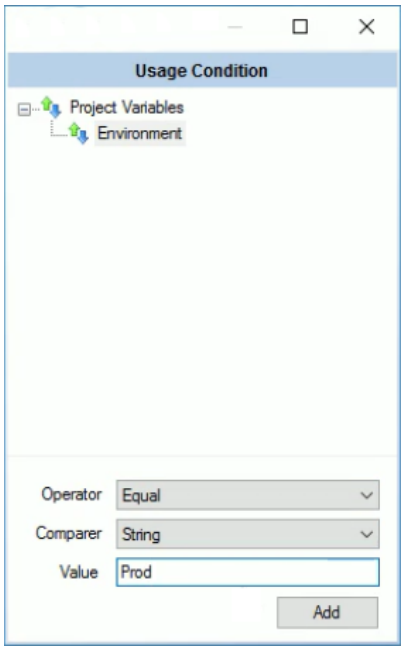
To set up a new prioritization for use in an execution package, follow the steps below.

1. On the **Execution** tab, right click **Prioritizations** and click **Add Prioritization**. The **Add Prioritization** window opens.
2. In **Name**, type a name for the prioritization.
3. Drag and drop an object from the **Available Objects** tree to **Selected Objects** to add this object to prioritized objects.
4. Click an object in the **Selected Objects** list to have the execution steps for that object displayed under **Object Settings**. Select or clear the individual steps under **Selected steps** to configure what steps will be prioritized in the execution. For instance, you might only want to prioritize the transfer of data from a specific table because your priority is to have the transfer completed as soon as possible.
5. Select **Blocking** if you want for the execution to halt all other execution tasks until the selected steps for the selected object has been completed.
6. Click **OK** to close the window and create the new prioritization, which will be added to the tree.

### ADDING A USAGE CONDITION

If you want to execute an execution package only under certain conditions, you can add a usage condition to the execution package. For instance, if you use multiple environments, you could have an execution package execute only in your production environment. To add a usage condition to an execution package, follow the steps below.

1. Add the project variable you want to use in your usage condition. See Adding a Project Variable.
2. On the **Execution** tab, expand **Execution Packages**, right-click the relevant execution package, and click **Add Usage Condition**. The **Usage Condition** window appears:




3. The available project variables are listed in the window. Click the variable you want to use. In the example above, the "environment" variable has been selected.
4. In the **Operator** list, click the operator you want to use.
5. In the **Comparer** list, click the data type you want to use when comparing the variable to the value.
6. In the **Value** box, type the value you wishg to compare the variable against.
7. Click **Add** to add the usage condition.

If try to manually execute a package with a usage condition that evaluates to "true", a warning message will pop up. The same message will be written to the log if you try to execute the package in a scheduled execution.

# INCREMENTAL LOADING

Incremental loading facilitates faster load times by enabling you to load only new transactional data into the data warehouse and staging databases. This is useful when the volume of transactional data in the data sources causes scheduled execution to take too long.

You should consider enabling incremental loading if your scheduled executions run longer than your execution strategy allows. For instance, if you load data during the night when business is closed to have fresh data in the morning, the load obviously needs to be completed before the users begin their day. If you want to transfer data from sources that are in use, you might also like to use incremental loading to minimize the impact on performance data transfer can have on users.

The default load plan in TX DWA is a full load. During a full load, the existing tables in the staging database and data warehouse are truncated, which removes all of the existing data, and new data is subsequently loaded from the data sources.

While it can be slow, the advantage of full loading is that you can be sure that any changes made in the source systems are carried over to the data warehouse. Therefore, it is common strategy to schedule periodical full loads while otherwise utilizing incremental loading. For instance, you could set up a full load during the weekend to make absolutely sure that your data warehouse is up to date, and schedule incremental loads during the workweek.

Incremental loading can be either source-based or target-based. Source-based incremental loading is the most efficient form, but it requires data that contains fields that can be used to identify which records are new and should be transferred. If the data do not contain such fields, target-based incremental loading offers a less powerful alternative.

## SOURCE-BASED INCREMENTAL LOADING

The purpose of incremental loading is to minimize the amount of data that needs to be loaded into the data warehouse, thus speeding up the loading process and minimizing any issues caused by putting a load on source systems. The key is to avoid transferring data that is already stored in the data warehouse. If your data contains one ore more fields that can define what records are new, you can utilize source-based incremental loading.

During the first deployment after incremental loading has been enabled, TX DWA will create additional tables in the staging database and data warehouse that have an _INCR or _I suffix. TX DWA will then do a full load. During subsequent executions of the project, truncation is disabled so that the data from the full load is not removed. Using the field, or number of fields, that you have chosen, TX DWA determines which records have been added to the data source since the last load and only transfers new records to the staging database and data warehouse.

Incremental load is enabled on the table level. Naturally, you will get the greatest increase in performance by enabling it on tables with a large amount of transactional data, such as those

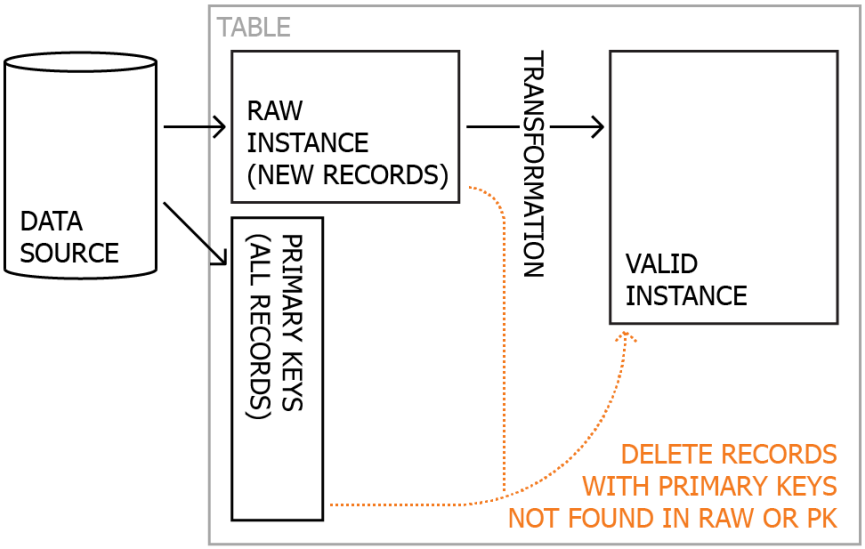
that contain large volumes of general ledger and inventory transactions. On the other hand, smaller tables with relatively few records, such as a Customer or Item table, usually do not take long to execute in the first place and enabling incremental load on them will only give you marginally better overall performance.

### Handling Records Deleted in the Data Source

An issue with incremental load is when records are deleted in the data source and cause the data in the data warehouse to come out of sync with the data source. You can handle this by scheduling a periodical full load, which is also useful if your incremental selection rule do not catch records that have been changed.

For larger data sources, a full load might not be feasible or you might want a better sync between source and data warehouse between full loads. For these situations, you can enable hard deletes, where the records deleted in the data source are also deleted in the data ware-house, or soft deletes, where the records are just marked as deleted in the data warehouse, on the incrementally loaded table.

The feature is implemented as shown in the figure below. On each incremental load, data is loaded from the source and into two instances of the table. The raw instance contains all the new records, while the primary key ("PK") instance contains the content of the primary key columns for all rec-ords. The "magic" happens once data has been transferred from raw to valid through the transfor-mation view. TX DWA will then delete any record in the valid instance that has a primary key that cannot be found in either the raw or the PK instance. If soft delete is enabled, TX DWAwill add a column, "Is tombstone" to the valid instance and updated deleted records with the value "true".



The implementation is designed to be fast and simple. To handle more advanced use cases, the feature can be used in conjunction with, or replaced by, scripts. Among the cases you should be aware of are the following:

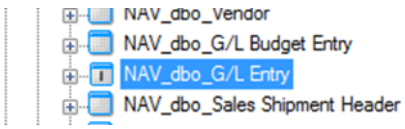
- If a record is undeleted/restored in the source, TX DWA will not undelete it in the valid in-stance unless it is loaded into the raw table again. TX DWA will not do anything with primary keys that are in the PK instance, but not in the valid instance. However, with a good incremental selection rule that loads both new and modified records or loads the new records and e.g. 100 more, this should not be an issue.
- If the primary key is transformed from raw to valid, TX DWA will not be able to match them to the primary keys stored in the raw and PK instance. All records in valid will therefore be deleted.

### ENABLING SOURCE-BASED INCREMENTAL LOADING

To utilize source-based incremental loading on a table, the table must contain a field that represents new data. This could be an identifier field, an entry number or a date. I addition to that, the table must have a primary key defined. To define a field or fields to be used as the primary key for the table, right-click a field and click **Include in Primary Key**.

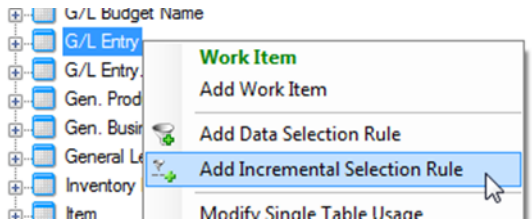
To enable source based incremental loading, follow the steps below.

1. Right click the table for which you want to enable incremental loading and click **Table Settings**.
2. Click the **Data Extraction** tab and select **Enable Source Based Incremental Load**.
3. Click on the option that represents how you would like to handle records deleted in the source. For more information, see Handling Records Deleted in the Data Source. You have the following options:
   - **Don't handle deletes**
   - **Use hard deletes**
   - **Use soft deletes**
4. If an error icon appears next to the setting, it means that another setting needs to be changed to enable source based incremental loading. Move you mouse over the error icon to see the error message. Once any settings have been changed as required, click **OK**. The table icon will now be overlaid with an "I" to make it easy for you to identify it as an incrementally loaded table.

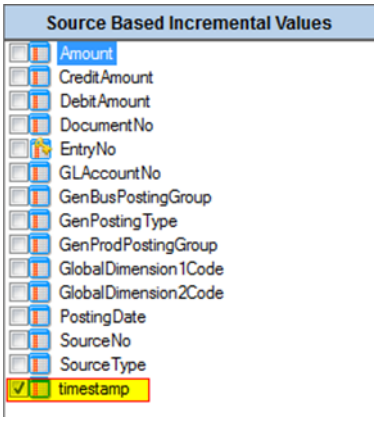



5. If the table belongs to the staging database, right click the corresponding source table under **Data Sources** in the project tree and click **Add Incremental Selection Rule.**
   - OR -
   If the table belongs to the data warehouse, right click the table and click **Add**

**Incremental Selection Rule**.




6. The **Source Based Incremental Values** pane appears. Select the fields identifying which records have been added or changed since the last incremental load.




   The fields you choose should ideally be fields that are generated by the system and incremented sequentially when new records are added. The table below lists recommended choices in some popular systems. Here are some recommended fields for the Microsoft Dynamics ERP systems:
   - **Dynamics NAV:** timestamp
   - **Dynamics AX:** RECID
   - **Dynamics GP:** DEX_ROW_ID
7. Repeat steps 1-5 for all tables you want to enable incremental loading on.
8. Right-click the **Business Unit**, click **Deploy and Execute**, click **Only modified tables and views** and click **Start**. TX DWA will begin the first full load of the tables with source based incremental loading now enabled. The necessary incremental tables will be automatically added to the staging database and populated with the latest incremental values. The next time the table is executed, TX DWAwill query these tables to determine the last record that was previously loaded and will only extract the data from the data source that was added after the last execution.

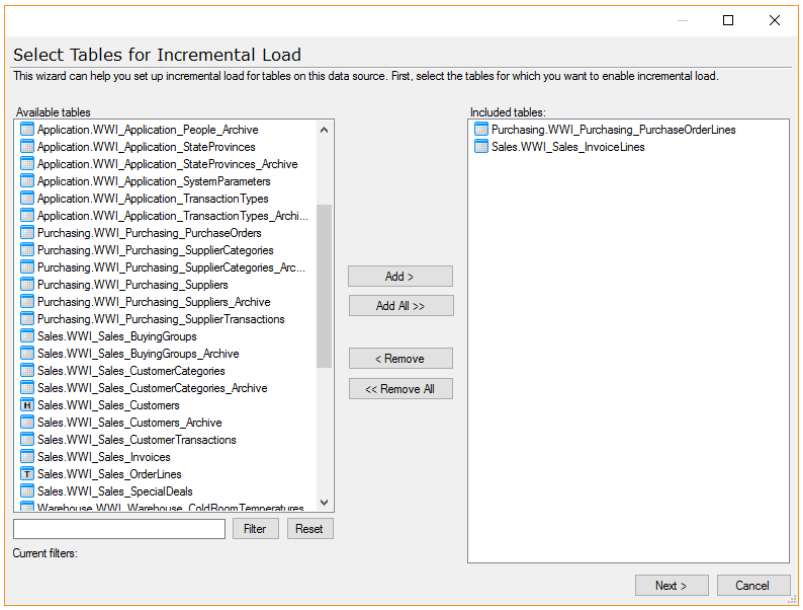
### ENABLING SOURCE-BASED INCREMENTAL LOADING FOR MULTIPLE TABLES

With the Set Up Incremental Load wizard, you can automate the set-up of source-based incremental load to enable it for a number of tables in one go. You still have to choose primary keys and what fields to use for the incremental load rule, but you selections are applied automatically. Apart from enabling source-based incremental load on the tables you select, any settings that are incompatible with source-based incremental load will also be changed.

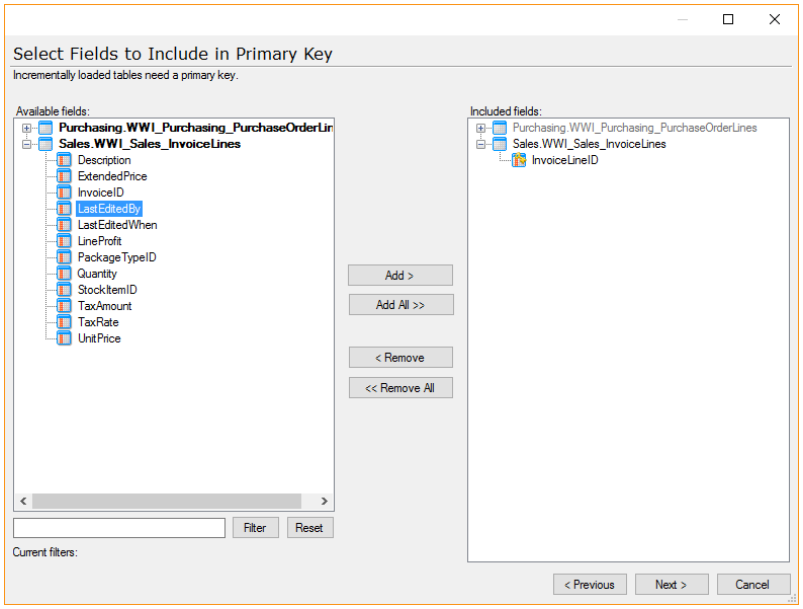
**Note:** When you run the wizard on a data warehouse, additional help will be available in the form of button, Auto Suggest. It will suggest primary keys based on the primary keys set on the staging database. For selection rules, the suggestion is a rule based on a custom field created on the table and mapped to the DW_TimeStamp system field of the source table. This rule is displayed in the wizard as "IncrementalTimeStamp" and has a gear icon.

To use the Set Up Incremental Load wizard, follow the steps below.

1. Right click a data source, staging database or data warehouse, click **Automate** and click **Set Up Incremental Load**.
2. The wizard appears.




   In the **Available tables** list, double-click the tables you want to enable source-based incremental load for. You can also click a table and then click **Add** to add an individual table. To add all visible tables, click **Add all**. Use the filter below the list to filter the list on table name. The following wildcards are supported:
   - **%**: Any string of zero or more characters.
   - **_**: Any single character.
   - **[ ]**: Any single character within the specified range ([a-f]) or set ([abcdef]).
   - **[^]**: Any single character not within the specified range ([^a-f]) or set ([^abcdef]).
3. Click **Next**. All incrementally loaded tables need to have a primary key, which you can add on this page. Any tables that do not have a primary key when you proceed to the next page, will be removed from the wizard.

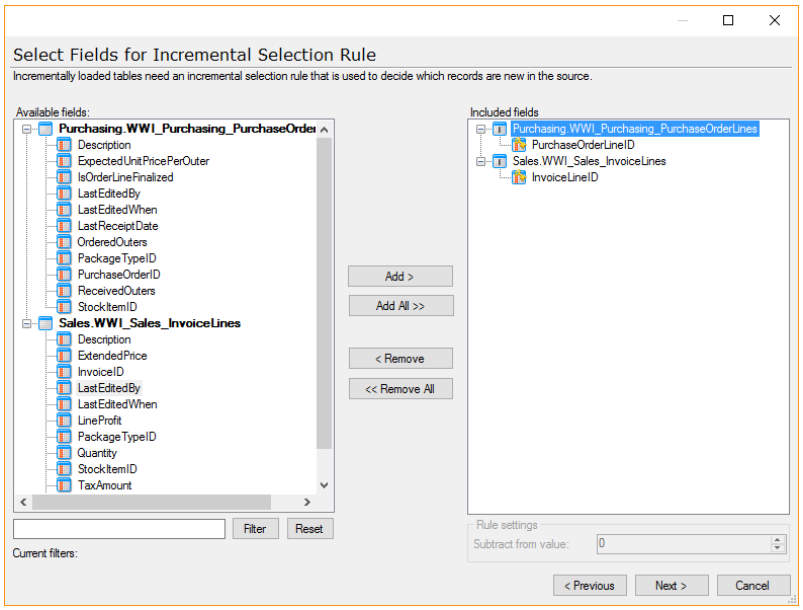



In the **Available fields** list, double-click the fields you want to use as primary keys on the tables. You can also click a field and then click **Add** to add an individual field.

The tables in the Available fields list that have at least one primary key field will be shown in bold.

In the **Included fields** list, any fields that are already primary keys on the tables you selected in the previous step are listed in grey. You cannot remove existing primary keys on this page, only add new ones.

4. Click **Next**. All incrementally loaded tables need an incremental selection rule. You select those on this page.

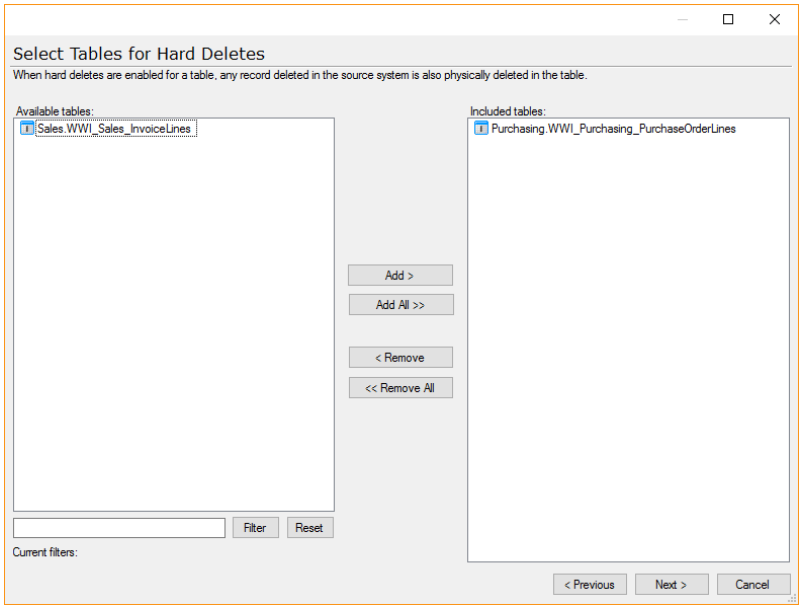



Select the fields you want to use as primary keys on the tables. Under **Rule Settings**, you can enter a subtraction value for a field if you want the incremental load to overlap

the previous load.

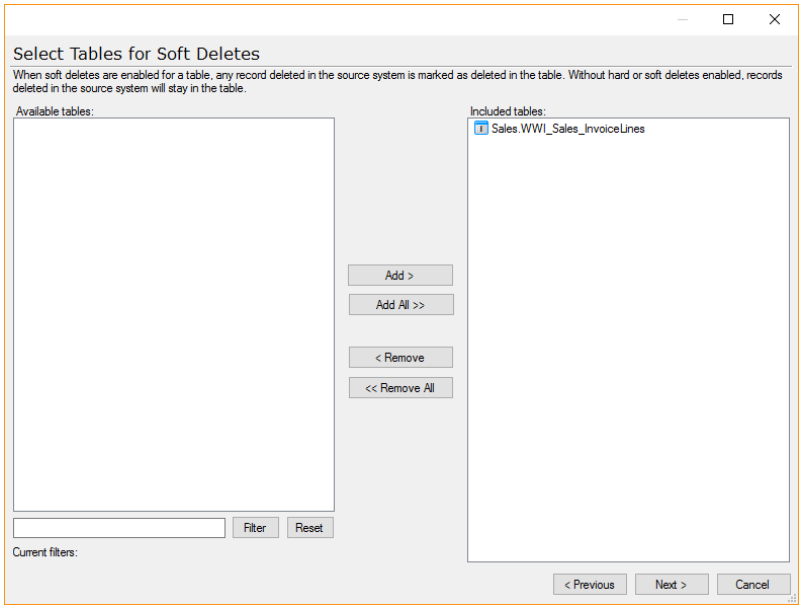
The tables in the Available fields list that have a field selected for incremental selection rule will be shown in bold.

5. Click **Next**. TX DWA can handle records deleted in the data source for you. On tables with hard delete enabled, records deleted in the data source is also deleted from the data warehouse. For more information about delete handling, see Handling Records Deleted in the Data Source.



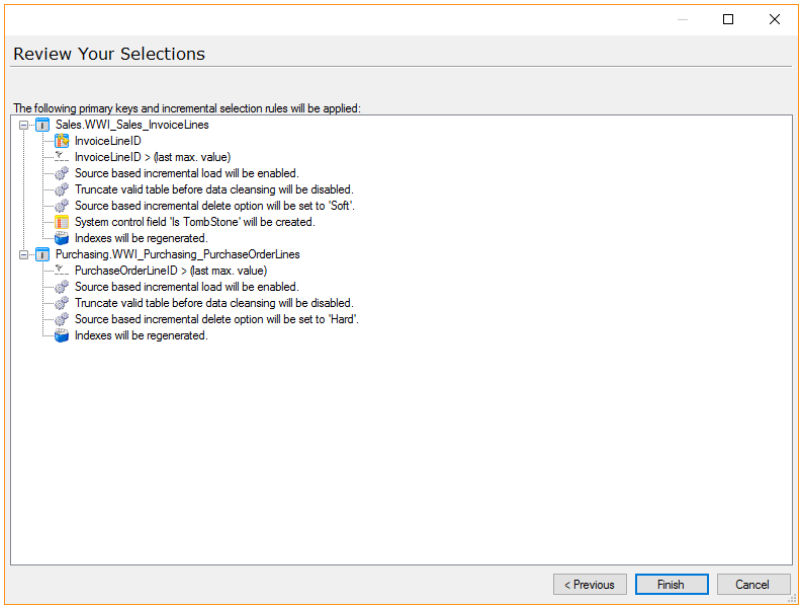

Select the tables you want to enable hard deletes for.

6. Click **Next**.




Select tables you want to enable soft deletes for. The tables you do not select for either hard and soft deletes, will not have any delete handling enabled.

7. Click **Next**.




Click **Previous** to go back to an earlier page and adjust the selections there or click **Finish** to apply the changes if they are correct.

## TARGET-BASED INCREMENTAL LOADING

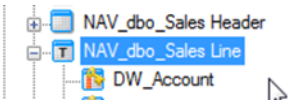
Target-based incremental loading is primarily used when there are no identifying fields that determine which records have been added since the last incremental update. With target based incremental loading, all of the data is transfered from the data source. Records are then compared against the existing records in the table and only new, updated, or deleted records are added to the staging database or data warehouse.

Target-based incremental loading is not as fast as source-based incremental loading, but is faster than a full load strategy.

### ENABLING TARGET-BASED INCREMENTAL LOADING

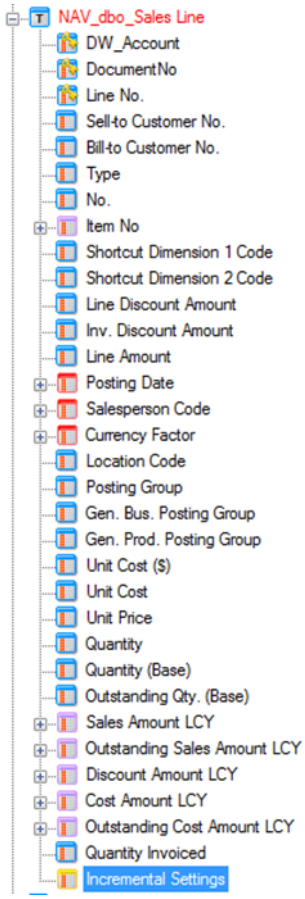
To enable source-based incremental loading for at table, follow the steps below.

1. Right click the table for which you want to enable incremental loading and click **Table Settings**.
2. Click the **Data Extraction** tab and select **Enable Target Based Incremental Load**. If an error icon appears next to the setting, it means that another setting needs to be changed to enable target based incremental loading. Move you mouse over the error icon to see the error message.
3. Click **OK**. The table icon will now be overlaid with an "T" to make it easy for you to identify it as an target-based incrementally loaded table.

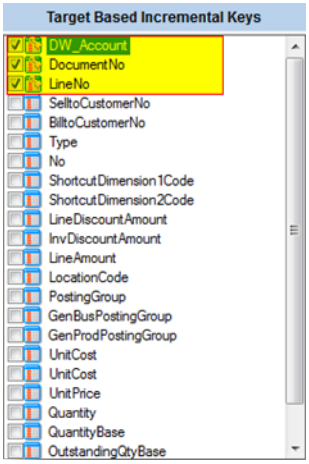



In the project tree, a new item, **Incremental Settings**, is added below the list of

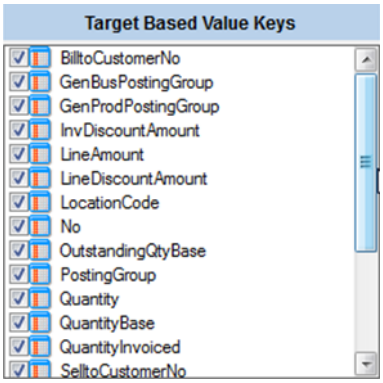
fields.



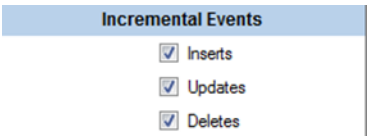

4. Click **Incremental Settings**. Three pane appear in the right-hand side of the application. In the first pane, **Target Based Incremental Keys**, the field or fields that represent the primary key for the table should be checked.




In the second pane, **Target Based Value Keys**, you should select the fields that would indicate a change in this table. TX DWA will create a hash key field based on the values of the fields you have selected and use this to determine if a record has been updated. In this example, all fields have been selected.




In the third pane, **Incremental Events**, you should select the events on the data source TX DWA will take into consideration during execution.




- **Inserts** are any new records that have been added to the data source, based on the primary key on the table.
- **Updates** are any records that have modified or been changed since the last incremental load. This is determined based on the **Target Based Value Keys** selected.
- **Deletes** are 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.

5. Right-click the table and click **Deploy and Execute** to perform the initial load of the table. The **Deploy and/or Execute** window opens.
6. Click **Start** to begin the deployment and execution process. During this process, all of the data will be loaded, and the hash keys for the target-based incremental load will be generated by TX DWA. Subsequent executions of the table will only load records that, depending on your settings, have been added, modified, or deleted from the data source since the last execution.

# MULTIPLE ENVIRONMENTS

With multiple environments, you can have a dedicated development environment with automatic transfer of the latest version of your project to the production environment. This ensures that the production environment is always online and available for end-users.

A dedicated development environment enables you 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 TX DWA.

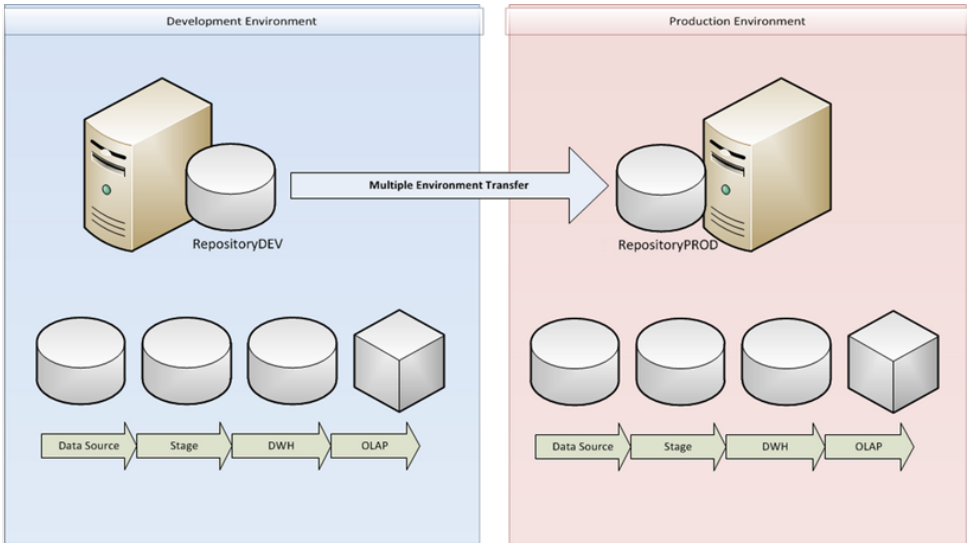
## PREREQUISITES

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 TX DWAinstalled. This applies to bit version(32-bit or 64-bit) as well.
- Ensure that TX DWA service is installed and started on the server(s) you want to deploy on. 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.



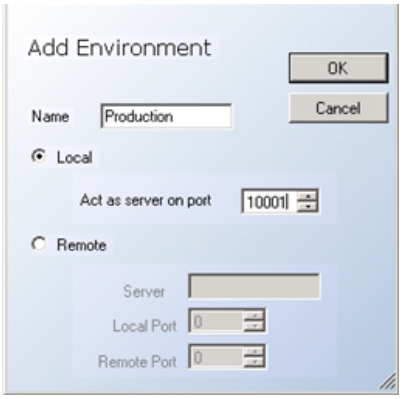

**Note:** You can set up as many environments as you need. Setup of additional environments follows the same steps as listed below.

### SETTING UP THE PRODUCTION SERVER

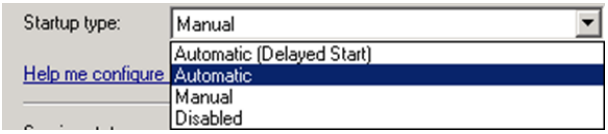
The first step is to set up the **Production** environment on the production server.

1. Log on to the production server and open TX DWA.
2. On the **Tools** menu, click **Environment Properties**
3. The Environment Properties window will open. Click**New Environment**.

4. The **Add Environment** window will open:

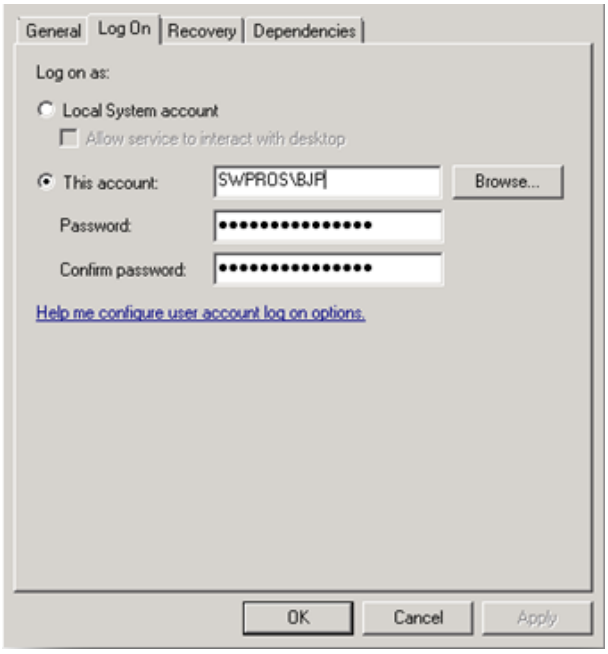



   - Type a **Name**. In this example, the name is "Production".
   - Select **Local**, as deployment will only be done into this environment and not from it.
   - Enter a port to use in the **Act As Server On Port** box. Make sure the port is free to avoid any conflicts on the network.
5. Close TX DWA.
6. Next you need to make sure that the TX DWA erver service is correctly set up. Click-/Right-click **Start**, click **Run**, type **Services.msc** and click **OK**. Locate the server service in the list. It will be named TX DWA **Server [version]**.
7. Right click the service and click **Properties**.
8. On the **General** tab, in the **Startup type** list, click **Automatic**.




9. On the **Log On** tab, click **This account**. In **This Account**, enter the account that was used for setting up the production environment, i.e. the account you are logged in with. Type the password for the account in the **Password** and **Confirm password**

boxes. Click **OK**.

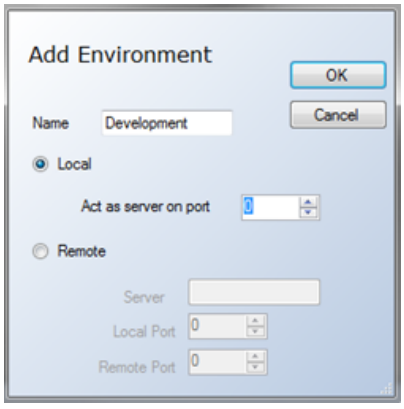



10. Right click the service in the list an click **Start**.
11. Log off from the production server.

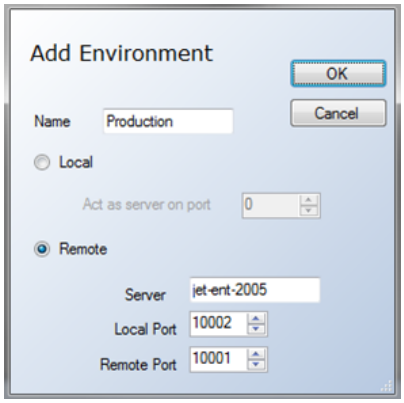
## SETTING UP THE DEVELOPMENT SERVER

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

1. Log on to the development server and open TX DWA.
2. On the **Tools** menu, click **Environment Properties**
3. The Environment Properties window will open. Click **New Environment** to create the development environment.
4. The **Add Environment** window will open:




   - Type a **Name**. In this example, the name is "Development".
   - Select **Local**, leave **Act as Server on Port** set to 0 and click **OK**.
5. 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.



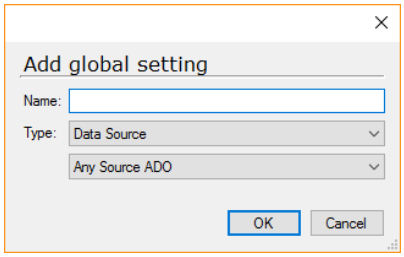

- Type a **Name**.
- Click **Remote** and fill in the remote server information:
  - **Server**: Type the server name or IP address of the production server.
  - **Local Port:** Enter any open port that is not being used by another application.
  - **Remote Port:**Enter the port you selected when setting up the production environment on the production server.

6. Click **OK**.

### CREATING GLOBAL DATABASES

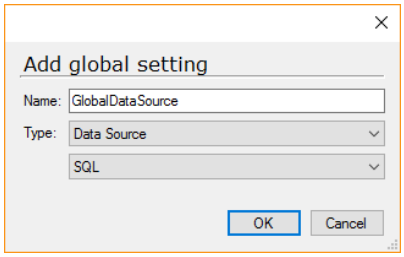
Global databases allow TX DWA to know where the related databases reside for the Production and Development environments. For example, the location of the Staging Database for both the Production and Development environments will be specified.

1. Click **New Global Database** from the Environment Dialog window. The **Add Global Setting** window appears.



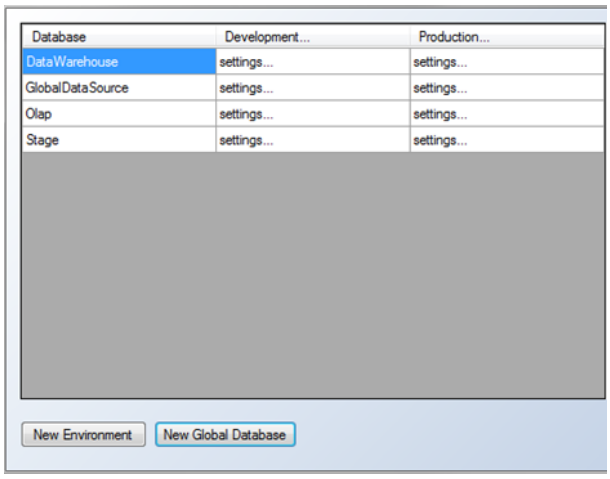

2. Using this window, you will be creating a series of databases that will be used in your project. You will create the following global databases:
   - Data source
   - Staging database
   - Data warehouse
   - OLAP
3. Assign a name to your data source, select **Data Source** in the **Type** section and select the relevant **Provider Type**. In this example, we will name our global database "Glob-

alDataSource" and use a provider type of Microsoft SQL.



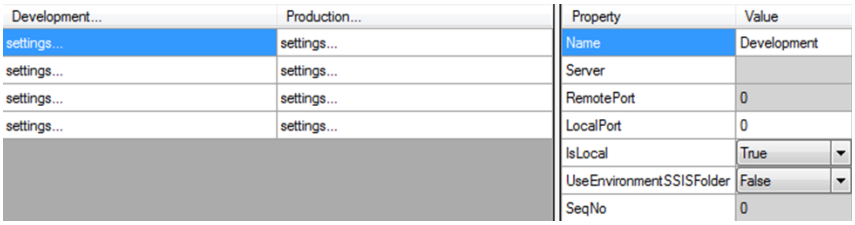

4. Repeat the previous step for all databases in the project (staging, data warehouse, and OLAP).

   Your results should look similar to those shown below:




## CONFIGURING GLOBAL DATABASES

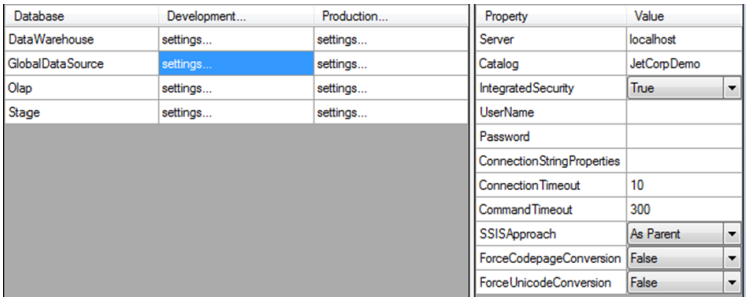
Within the **EnvironmentProperty**window you should have a data source, data warehouse, OLAP, and staging database. Each environment, Development and Production, has a "**Settings..."** section for each database. You can also click the environment name to access additional settings.




### CONFIGURING THE DATA SOURCE

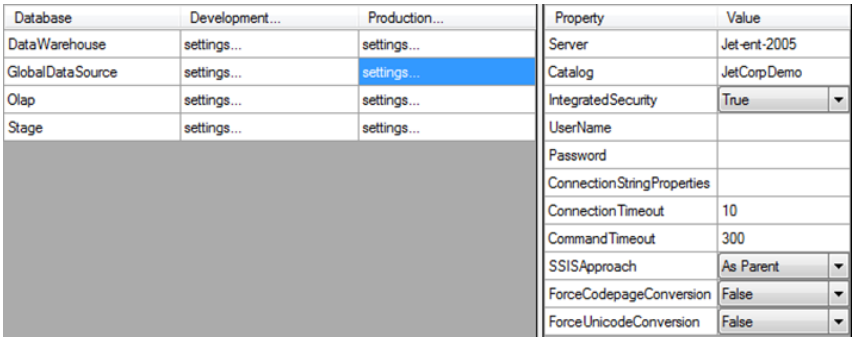
1. Select the **"Settings..."** field from the data source row in the Development column from the **Environment Properties** window. This will display the Settings pane to the right.

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.




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, for instance your ERP system. In our example, this is "JetCorpDemo".

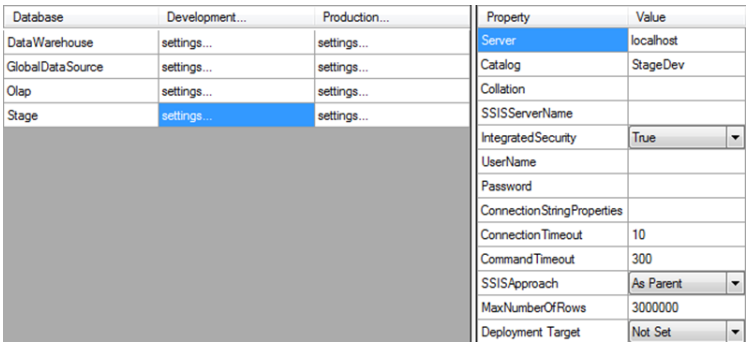
**Note:** If you are using your live ERP database for extracting data in both the development and production environments, then the server name and catalog in both the Development and Production columns will be the same.



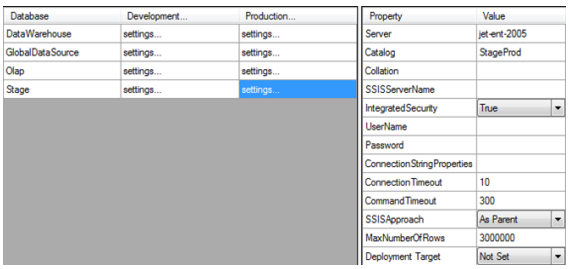

## CONFIGURING THE STAGING DATABASE

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.



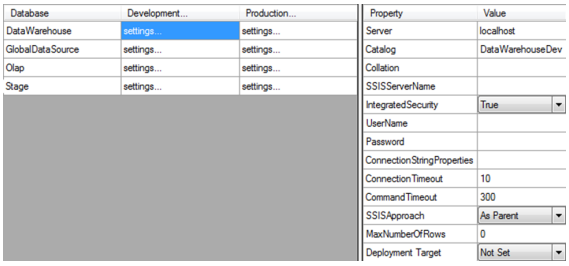

3. Next click "**Settings...**" on the **Stage** row of the **Production** column.
4. Enter the following configuration:
   - **Server:** This will be the name of the server for the Production Environment. In our example, the server name is "jet-ent-2005".
   - **Catalog:** This will be the name associated with your staging database in the production environment. In our example, we use StageProd.




## CONFIGURING THE DATA WAREHOUSE

Next you will need to configure another Global Database for the data warehouse.

1. 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.



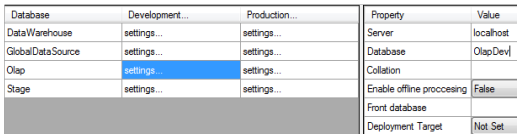

3. Next Click "Settings..." on the data warehouse row of the **Production** column.

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

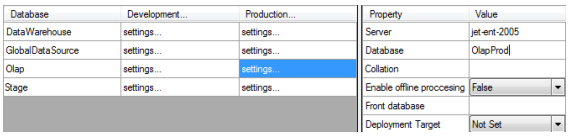
Next you will need to configure another Global Database for the OLAP cubes.

1. 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".



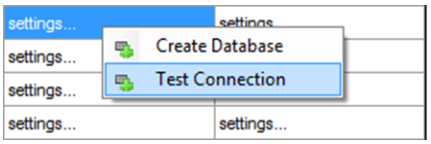

3. Next, Click "Settings..." on the OLAP row of the **Production** column.

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".




### 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 "Settings..." and select Test Connection. If you get an error message, it generally means that the database has not been created yet. Right-click "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.

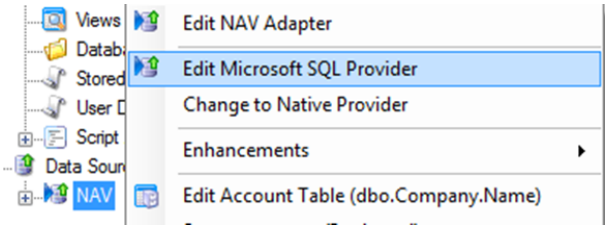
**Note:** The data source does not have the option to Create Database. This database represents the data source that TX DWA is extracting from and will already exist in your infrastructure. An example of this will be your Dynamics NAV, GP, or AX database.

### CONFIGURE PROJECT CONNECTIONS

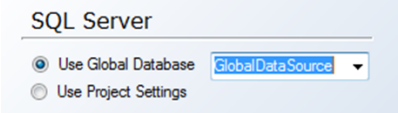
The environments have now been set up, and the global databases have been configured. The next step is to configure the connections in the project to utilize these Global Databases.

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**.

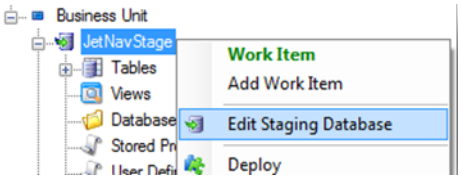
   **Note:** This will vary depending on your data source type.




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.

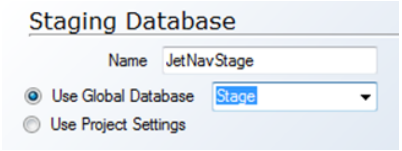



3. Navigate to your Staging Database, right-click the database, and select Edit Staging Database.

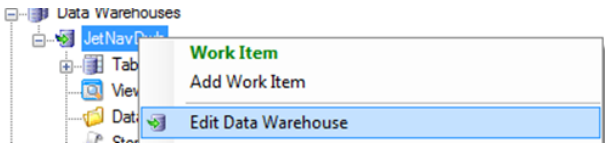



4. Select **Use Global Database** for the data source, choose the Global Database that represents your staging database, and click OK. There will generally be only one Global

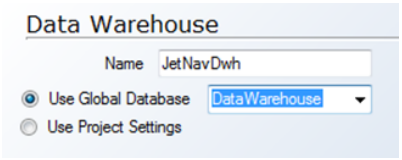
Database displayed in the drop-down list.



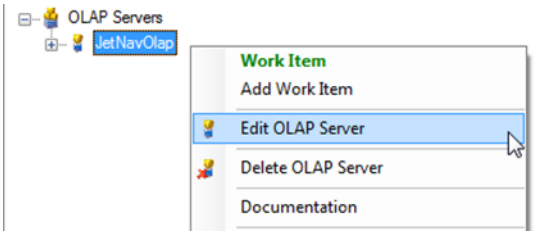

5. Navigate to your **Data warehouse**, right-click the database, and select Edit Data Warehouse.




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.




7. Navigate to the OLAP Database on the **Cubes** tab, right-click the OLAP Database, and select **Edit OLAP Server**.

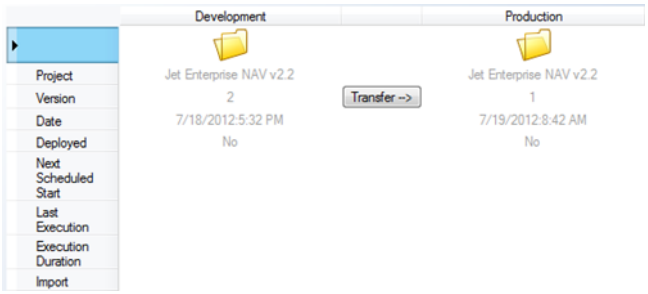



8. Select **Use Global Database** for the data source, choose the Global Database that represents your OLAP database, and click OK. There will generally be only one Global Database displayed in the drop-down list.
9. The final step is to deploy and execute the project to ensure your project is properly configured and ready for transfer. On the **Tools** menu, click **Deploy and Execute Project** and then click **Start**.

## TRANSFER THE PROJECT FROM DEVELOPMENT TO PRODUCTION

You are now ready to transfer the project from the development to the production environment.

1. Log in to the **Development Environment** server, and open TX DWA.
2. On the **Tools** menu, 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**.

4. Deploy the project on the production environment.
    - If you are transfering the project from development to production for the first time, or if you simply want to deploy all objects, right click the **Production Environment** folder and click **Deploy**. When the **Deployed** line in the **Production** column changes to "Yes", the process has finished.
    - If only some objects were changed in the development environment and are in need of deployment, right click the **Production Environment** folder and click **Partial Deployment**. The **Remote Deployment Window** opens. A list of deployable objects is displayed. Select the objects you want to deploy and click **Deploy**. While deployment is under way, the **Partial Deploy** window displays the deployment status. When deployment has finished, a window opens with a list of the deployment tasks completed. Click **Close** to close the window
5. Click **Close** to close the **Multiple Environment Transfer** 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 "TX DWA Server Scheduler" service is disabled on the machine hosting the development environment. For more information regarding the configuration of Execution Packages, see Execution Packages.

# QLIK MODELER

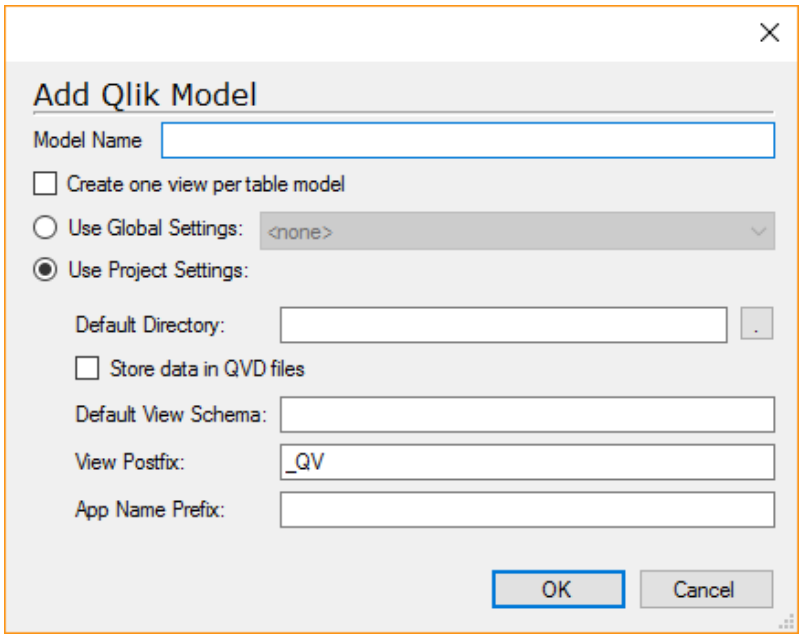
The Qlik Modeler in TX DWA creates data structures that can be used by both QlikView and Qlik Sense. For Qlik Sense, TX DWA can create an app directly in Qlik Sense Desktop and deploy an app to Qlik Sense Enterprise. For QlikView, the script generated by TX DWA can be imported or included in the application.

## BUILDING A QLIK MODEL

### ADDING A QLIK MODEL

To add a new Qlik model, follow the steps below.

1. On the **Qlik** tab, right click **Qlik Models** and click **Add Qlik model**. The **Add Click Model** window appears.




2. In the **Model name** box, type a name for the model.
3. In the **Default directory** box, enter the directory where you would like to store the generated Qlik script files.
4. Select **Store data in QVD files** if you would like to store the retrieved data in QVD files for later reuse by the Qlik apps.

   **Note:** If you want to deploy and execute to Qlik Sense Enterprise, using QVD files require additional setup. You also need to make sure that the **Default directory** path can be reached by the Qlik Sense server. See Deploying to Qlik Sense Enterprise for more information.

5. (Optional) In the **Default view schema** box, type a schema name if you would like the view generated by the Qlik Adapter to belong to a special schema for an easier over-view.

6. (Optional) In the **View postfix** box, type a postfix for the generated views. Default is "_QV" (for "Qlik Valid").
7. Select **Create one view per table model** if you would like TX DWA to generate one view per table model in your solution, even if the same table with the same fields selected is used multiple times. The default behavior is to create one view per unique table model. This means that if two table models have the same fields, TX DWA will only create one view.
8. Click **OK**.

### CLONING A QLIK MODEL

Instead of creating a Qlik model from scratch, you can clone an existing Qlik model to create a new Qlik model that is identical to the one it was cloned from. Naturally, this is useful if you need to create a Qlik model that is very similar to an existing Qlik model.

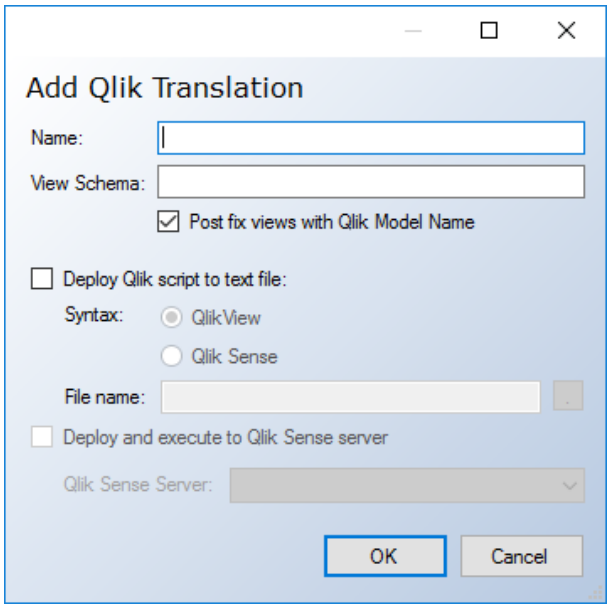
- To clone a Qlik model, right click the model, click **Advanced** and click **Clone Qlik model**.

### ADDING A QLIK MODEL TRANSLATION

Each Qlik model can have a number of translations that you can use for internationalizing your solution. You need at least one translation, so when you create a Qlik model, TX DWA will automatically create a translation for you. Translations are more than just your data labeled in a different language, however. Deployment of a Qlik Model is also done on the translation level.

Translations can be found under Model Translations under each Qlik model. To create a new translation, follow the steps below.

1. Right click **Model Translations** under the relevant Qlik model and click **Setup Qlik Model Translations**. The **Qlik Model Translations** window appears.
2. Click **Add Translation**. The **Add Translation** window appears.



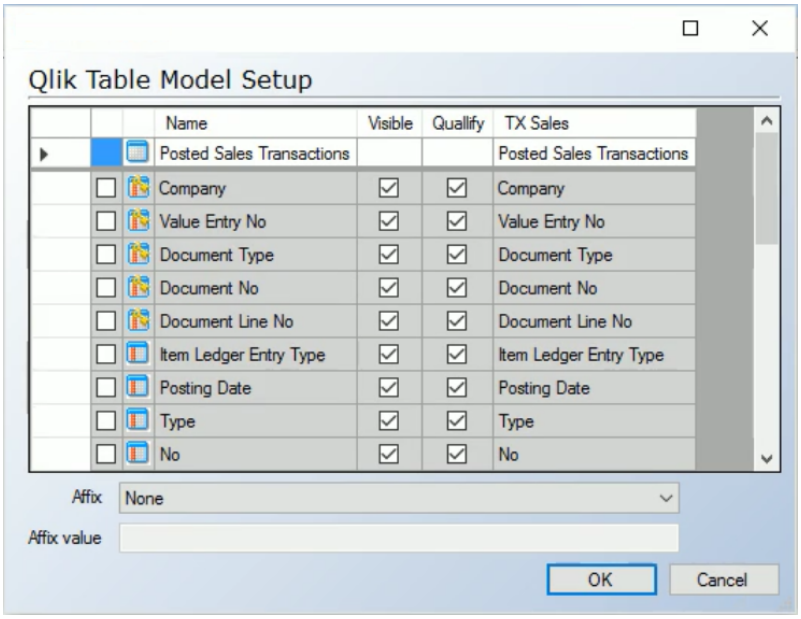
3. In the **Name** box, type a name for the translation, for instance the language of the translation.
4. (Optional) In **View Schema**, type a name for the schema the related views should use.
5. Clear **Post fix views with Qlik model name**, if you do not want TX DWA to postfix the views with the model name. This is not recommended since it means that some views might end up with the same name. In that case, that last view to be deployed will be the only view there is of that name.
6. Select **Deploy Script to Text file** if you want to output the generated Qlik script to a text file. Under **Syntax**, make sure that the application, you will be using the script with, is selected. Type a path and name for the file in the **File name** box.
   **Note:** If you want to deploy and execute to Qlik Sense Enterprise, you need to make sure that the **File name** path can be reached by the Qlik Sense server.
7. Select **Deploy and execute to Qlik Sense server** and select a server in the **Qlik Sense server** list to deploy the Qlik model to Qlik Sense Enterprise.
8. Click **OK**. The Add Translations window closes and a new column is added in the Qlik Model Translations window.
9. Type the translated names in the column under the name of your newly created translation and click **OK** when you are done.

### ADDING A QLIK TABLE MODEL TO A MODEL

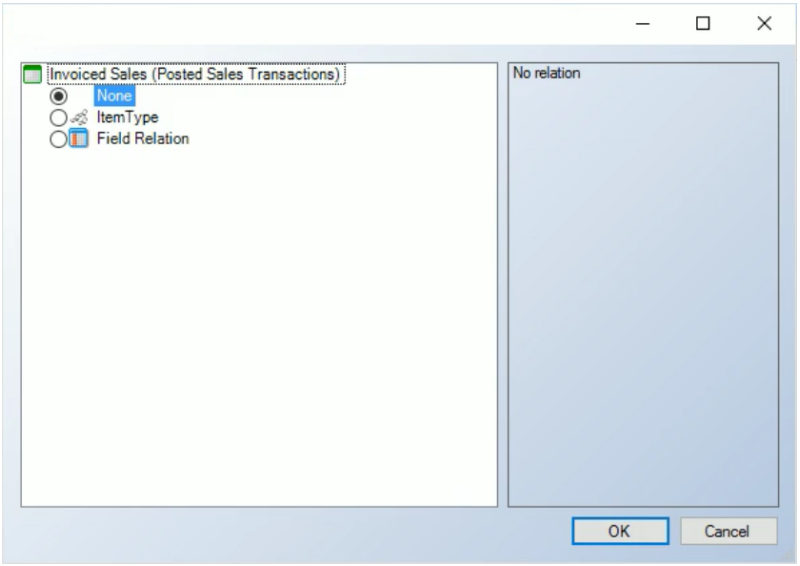
A Qlik model should contain a number of table models. A table model is simply the term we use for a table that is part of a Qlik model. You can add the same table multiple times, for instance if your database contains a Sales table with both a Sell-to-customer and a Bill-to-customer. To add a table model, follow the steps below.

1. On the **Data** tab, find the data source, or database that contains the tables you want to add to your Qlik model, right click it and click **Open in New Window**.

2. Switch to the **Qlik** tab. Drag a table from the window you opened to a Qlik model. The **Qlik Table Model Setup** window appears.




3. Select the fields you want to include in your table model. Clear the checkbox in the visible column if you do not want to show the field in QlikViev/Qlik Sense, but need to include the field to e.g. create a relation.
4. If the Qlik model already contains other table models, the **Setup Qlik Table Model Relations** window appears.




Here, you can set how the table you are adding is related to existing table models in the Qlik model. For each existing table model, you have the following options:

- None: No relations to that table model.
- An existing relation defined in the data warehouse (recommended).
- Field relation: Relate using identical field names on both tables.

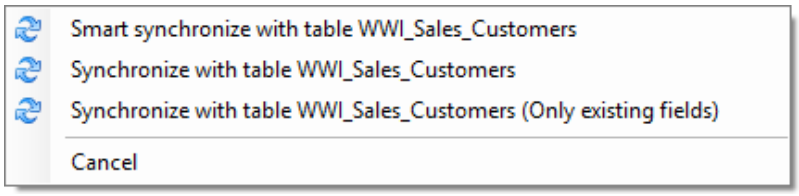
5. Click **OK**.

### ADDING A CONCATENATED QLIK TABLE MODEL

A concatenated table is a table that combines data from multiple source tables. In QlikView or Qlik Sense, concatenated tables are useful for e.g. avoiding circular references. Concatenated tables can be created in the data warehouse, but creating them in the Qlik Modeler allows you to have one architecture in the data warehouse and another in Qlik without duplicating a lot of data.

To create a concatenated table model, follow the steps below.

1. On the Data tab, find the data source, or database that contains the tables you want to add to your Qlik model, right click it and click **Open in New Window**.
2. Switch to the **Qlik** tab. Drag a table from the window you opened to an existing Qlik table model. A menu appears.

   You have the following options:
   - **Smart synchronize with table [table name]:** TX DWA looks on the other source tables on the Qlik table model and adds the fields from the source table that matches fields from the other source tables.
   - **Synchronize with table [table name]:** Adds the fields of the source table to the Qlik table model.
   - **Synchronize with table [table name] (only existing fields):** Adds the fields of the source table that have the same name as a field already on the Qlik table model.

   Click on the option you want to use.

Under **Mappings**, you can see the tables that delivers data to the table model. If you expand a field on the concatenated table model, you can see the fields from the tables that are mapped to that particular field.

You can also add new tables to a concatenated table model by dragging fields from a data warehouse or business unit opened in a new window to the Qlik table model.

1. On the **Data** tab, right click the relevant table and click on **Open in New Window**.
2. Click the **Qlik** tab, expand the concatenated table model and drag fields from the window containing you data warehouse or business unit table to the fields on the concatenated table model. You will see that new fields are added under the table model fields and a new table is added under **Mappings** if the table was not already there.

When the Qlik model is deployed, the tables are set to be loaded in the order they appear under mappings.

- To switch the order in which the tables are loaded, click on a table under **Mappings** and press **Alt + Up/Down** to change the order.

### MANUALLY RELATING QLIK TABLE MODELS

If you do not have any relations between table models defined in the data warehouse, you can add them manually. To add a new relation between two table models, follow the steps below.

1. On the **Qlik** tab, navigate to the table model you want to relate to another table model.
2. Click the field you want to base the relation on and then drag the field on a field on another table model.
3. TX DWA will ask you if you want to create a relation. Click **Yes**.
4. (Optional) TX DWA will manage the relation automatically and make sure that only the relations defined in the TX DWA project will be used in Qlik. However, you can choose to an unmanaged approach to let the Qlik app create the relation. The Qlik behavior is to relate similarly named fields. To use this approach, right click on the newly created relation, point to **Relation Type** and click **By name**

### ADDING SELECTION RULES TO A QLIK TABLE MODEL

Like tables on data sources and in data warehouses, Qlik table models support data selection rules. See Data Selection Rules for more information.

### RENAMING A QLIK MODEL, MODEL TRANSLATION OR TABLE MODEL

It can be very useful to rename a Qlik table model, especially if you use the same table multiple times in your solution. In addition to table models, you can also rename Qlik models and model translations. To rename an object, follow the steps below.

1. On the **Qlik** tab, locate the model, model translation or table model and click it.
2. Press F2 to make the name editable.
3. Type the new name for the object and press Enter.

The original name of the table model will be displayed in parenthesis after the name you type.

### DEBUGGING RELATIONS WITH PLAIN TEXT KEYS

TX DWA utilizes concatenated key fields for creating relations between Qlik table models. For performance reasons, the value of these fields are hashed to create a value that uniquely identifies each record in the Qlik table model, but are shorter than the key would otherwise be.

For debugging purposes, it can be useful to see the values before the hashing is applied. In TX DWA, that can be accomplished by changing the hashing algorithm. To change the hashing algorithm for a Qlik model to plain text

- Right click the Qlik model, click **Hashing Algorithm** and click **Plain Text (Debug)**.

You can then preview the data in a Qlik application.

Since the plain text algorithm is very slow, remember to change the hashing algortihm back when you have finished debugging.

## SCRIPTING IN QLIK MODELS

The Qlik Modeler is powerful, but sometimes there are things that call for code. Like other parts of TX DWA, the Qlik Modeler contains features that enable you to write your own scripts to supplement the scripts generated by TX DWA .

- You can use Qlik snippets to create reusable pieces of code.
- You can create custom Qlik fields, where you write the code that controls what the field will contain.
- You can use pre- and postscripts to add code before and after the code generated by TX DWA respectively.

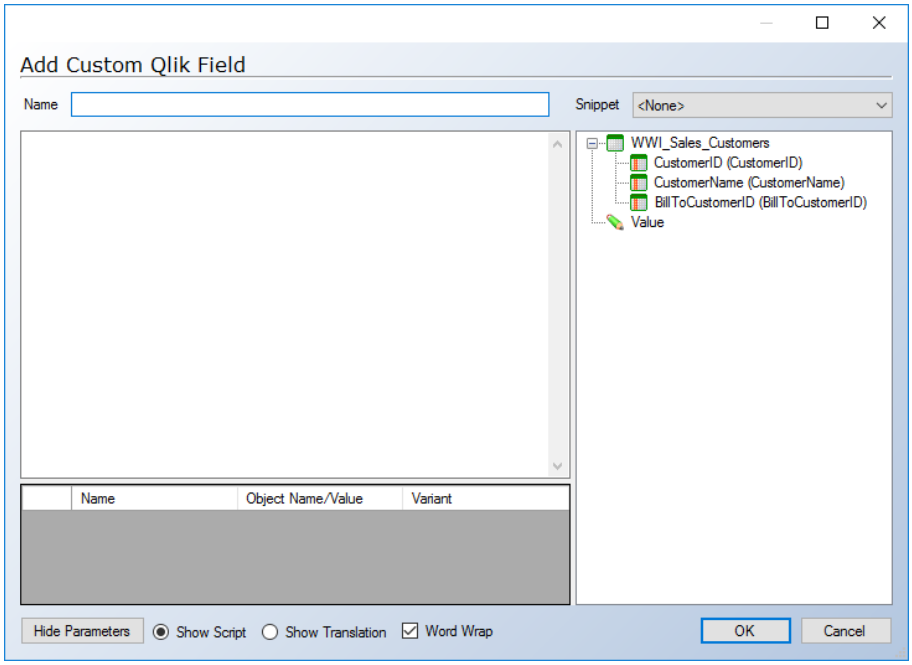
### ADDING A QLIK SNIPPET

Like SQL snippets on data warehouses and OLAP snippets on cubes, you can create Qlik snippets. These can be used in custom Qlik fields. See Snippets for more information.

### ADDING A CUSTOM QLIK FIELD

Custom fields are fields where the value is calculated by a Qlik script that you write.

1. To add a custom field, right click a table model, click **Add Custom Qlik Field**. The **Add Custom Qlik Field** window appears.




2. Type a name for the custom field in the **Name** box.

3. If you want to use a Qlik snippet in your custom field, click on the Qlik snippet you want to use in the **Snippet** list.
4. Enter your script in the main text box. You can drag in fields from the list to the right to use in your script. Drag in "Value" to create your own custom variable.
5. Click **OK** to save the script.

### ADDING PRE- AND POSTSCRIPTS

TX DWA generates the code the creates the Qlik model you build in the Qlik modeler. You can supplement that code with pre- and postscripts that are added before or after the generated code.

1. Right click **Prescripts**, click **Add Snippet Prescript** and click the snippet you want to use.
   - OR -
   Right click **Postscript**, click **Add Snippet Postscript** and click the snippet you want to use.
2. Type a name for the script in the **Name** box.
3. In the main text area, enter the code for the Qlik snippet. You can drag in tables, fields, and project variables from the list to the right to use in your function/procedure. Drag in "Value" to create your own custom variable.
4. Click **OK** to save the script.

### ADDING PRE- AND POSTSCRIPTS BASED ON A QLIK SNIPPET

In addition to writing it from scratch, you can add a pre- or postscript based on a Qlik Snippet.

1. Right click **Prescripts** and click **Add Prescript**
   - OR -
   Right click **Postscript**and click **Add Postscript**
2. Type a name for the script in the **Name** box.
3. In the window that appears, map the available fields to the parameters in the snippet. Drag the field(s) from the list on the right and drop the field on the **Object Name/Value** column for the relevant variable. The **Object Name/Value** column and **Variant** column will populate automatically.
4. Click **OK** to save the script.

### ADDING A CUSTOM FORMAT SECTION

The custom format section - you can only have one - is added as a seperate script before the script containing prescripts, postscripts and the script generated by TX DWA. To keep things tidy in the Qlik application, any default formatting settings you want to override should be added to this script.

- To add a formatting script, right click **Script**, click **Add Custom Format Section**, type a name for the script in the **Name** box, type your script in the main text area and click **OK**.

## DEPLOYING AND EXECUTING QLIK MODELS

Just like data warehouses, Qlik models needs to be deployed before you can use them in QlikView/Qlik Sense. Deployment generates the script that can be used in one of the Qlik applications. You can deploy a Qlik model, which deploys all underlying Qlik model translations, or the individual Qlik model translation.

- To deploy a Qlik model or Qlik model translation, right click it and click **Deploy**.

Depending on the Qlik application you are targeting and your setup, there are different ways to deploy and get data into the Qlik application:

- Deploy the Qlik model and copy and paste the Qlik script from TX DWA to the Qlik application.
- Deploy the Qlik script to a text file and set up the Qlik application to read the file.
- Deploy to Qlik Sense Desktop. TX DWA can create an Qlik Sense app and load the data.
- Deploy to a Qlik Sense Enterprise server. TX DWA can create a Qlik Sense app and execute the Qlik Sense app to load data.

### REVIEWING SCRIPTS AND DEPLOYING TO QLIK WITH COPY AND PASTE

You can always view the latest script generated for a Qlik model or Qlik model translation. To get the model into your Qlik application, you can copy the script and paste it to your Qlik application.

- To review the scripts generated by TX DWA, right click a Qlik model translation and click **Qlik Sense App script** or **QlikView Scripts**.

### DEPLOYING TO A TEXT FILE

To set up a Qlik model translation to deploy to a text file, follow the steps below..

1. On the Qlik tab, right click on the Qlik model translation and click **Edit Translation**. The **Edit Translation** window appears.
2. Select **Deploy Qlik script to text file**.
3. Under **Syntax**, make sure that the application you will be using the script with, is selected.
4. Type a path and name for the file in the **File name** box.
5. Click OK.

### DEPLOYING TO QLIK SENSE DESKTOP

TX DWA can create a Qlik Sense app and make it available in Qlik Sense Desktop. To create a Qlik Sense app based on a Qlik model, follow the steps below.

1. Deploy the Qlik model translation if you have not already done so.
2. On the Qlik tab, right click the relevant Qlik model translation and click **Create Qlik Sense App**. The **Create Qlik Sense App** window appears.
3. In **App** name, type a name for the app.
4. Click **Create**. TX DWA will start Qlik Sense. If an app with the same name already exists, TX DWA will ask you if you want to update the app.
5. When the app has been successfully created or updated, TX DWA will ask you if you want to load data.
6. Lastly, TX DWA will ask you if you would like to close Qlik Sense.

### DEPLOYING TO QLIK SENSE ENTERPRISE

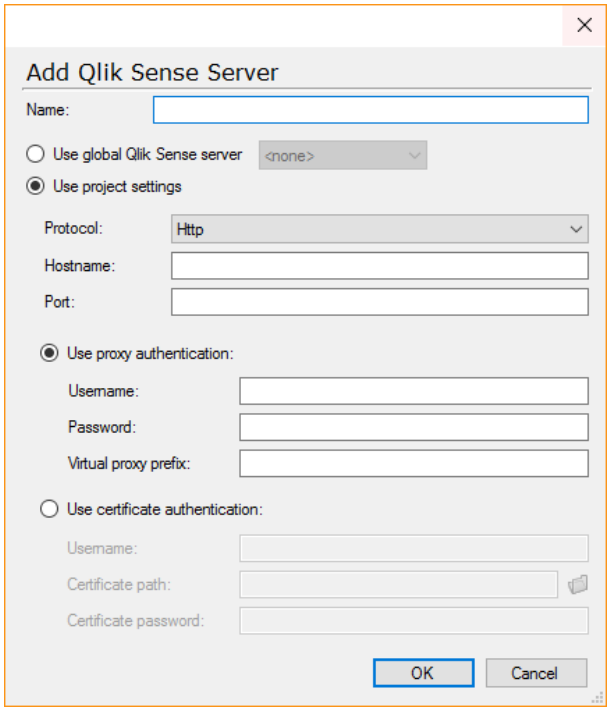
TX DWA can deploy a Qlik Sense app to Qlik Sense Enterprise and make the app update data each time the project is executed. The setup consists of the following steps:

- Add a Qlik Sense Enterprise server to TX DWA.
- Set up the Qlik model translation to deploy to the server.
- (Optional) Configure TX DWA to save data as QVD files and Qlik Sense Enterprise to use these files.

**Note:** The Qlik Sense Server needs to be able to connect to the SQL Server that contains the data used in the Qlik model. Each staging database or data warehouse database that contain tables used in the Qlik model should have the server name set to a publicly accessible IP. If you use Windows authentication, the user running the Qlik services needs access to database.

To add a Qlik Sense Enterprise server, follow the steps below.

1. On the Qlik tab, right click **Qlik Sense Servers** and click **Add Qlik Sense server**. The **Add Qlik Sense Server** window appears.




2. Type a name for the server in the **Name** box.
3. In the **Protocol** list, click the protocol you want to use.
4. Type the your server's hostname in the **Hostname** box.
5. Type the port to connect to in the **Port** box if it is different from the default. The defaults are 4747 if you use certificate authentication, 80 if you use proxy authentication with HTTP and 443 if you use proxy authentication with HTTPS.
6. Click **Use proxy authentication** if you are using the proxy authentication method to authenticate with the Qlik Sense Enterprise server. Type your username in the **Username** box and your password in the **Password** box. Write the prefix from the virtual proxy in Qlik Sense in the **Virtual Proxy Prefix** box.
7. Click **Use certificate authentication** if you are using the certificate authentication method for authenticating with Qlik. Type your username in the **Username** box, enter the path to the certificate in the **Certificate path** box and the associated password in the **Certificate password** box.
8. Clik **Ok**.

For more information about adding a virtual proxy to Qlik Sense Enterprise or exporting a certificate for use in TX DWA, please see the Qlik Sense documentation.

To set up a Qlik Model Translation to deploy to Qlik Sense Enterprise, follow the steps below.

1. On the Qlik tab, right click on the Qlik model translation and click **Edit Translation**. The **Edit Translation** window appears.
2. Select **Deploy and execute to Qlik Sense server**.
3. Select a server to deploy to in the **Qlik Sense server** list.
4. Click OK.

To set up Qlik Sense Enterprise to be able to save data from TX DWA as QVD files, follow the steps below:

1. On the Qlik tab, right click on the **Qlik model** and click on **Edit Qlik Model**. The **Edit Qlik Model** window appears.
2. Select **Store data in QVD files** and click **OK**.
3. If the Qlik model translation has not been deployed before, right click on it and click **Deploy**. The deployment will fail since the QVD files cannot be created, but it will create an app on the Qlik Sense Enterprise server.
4. Open Qlik Sense in your browser, open your Qlik Sense App, and click **Data Load Editor** to go to the data load editor.
5. Click **Create New Connection** and then click on **Folder**. Choose a folder on your machine. In the **Name** box, type "File Storage <Qlik model translation name>", where <Qlik model translation name> is the name of the Qlik Model translation you are deploying. Click **Create**.
6. Go to the Qlik Sense management console and click **Data Connections**. In the list, click on the data connection you just created to open it. In the **Name** box, delete the text in the brackets, so the name is simply "File Storage <Qlik model translation name>". Click **Apply**.
7. You should now be able to deploy your Qlik model translation without errors.

With the setup complete, you are ready to deploy and execute. The deployment generates the script and pushes the app to the server, while execute makes the app load new data.

- To deploy and execute all Qlik models in your project, right click the Qlik root node and click **Deploy and Execute**.
- To deploy and execute a Qlik model, right click the Qlik model and click **Deploy and Execute**.
- To deploy and execute a Qlik model translation, right click the Qlik model translation and click **Deploy and Execute**.

**Warning:** When a Qlik Sense app is deployed to a server, a data connection is created. This connection uses the names of the staging databases and data warehouses in the project. If you have another project that deploys to the same Qlik Sense Enterprise server, the names for these objects in this project have to be different to avoid overrides.

**Thank you for using TX DWA!**

Have questions not answered in this user guide?

Need to get in touch with support?

Please visit our support site at

support.timextender.com