Embarcadero® Change Manager™ 5.0.1 User Guide

Contents

Welcome to Change Manager

Embarcadero® Change Manager™ offers database administrators and developers a powerful tool to simplify and automate the database change management lifecycle. The schema compare and alter, data compare and synchronization, and configuration auditing capabilities let you report on database changes, roll out new releases, and troubleshoot database performance problems that result from both planned and unplanned changes. Change Manager supports IBM® DB2® for LUW, Microsoft® SQL Server, Oracle®, and Sybase®. Change Manager is available as a stand-alone application or as an Eclipse plug-in.

Technical Requirements

Before installing Change Manager, verify that your environment meets the following requirements.

Hardware Requirements

The following minimum hardware requirements are required to run Change Manager:

- 1.2 GHz processor
- 1 GB of RAM
- 500 MB of hard disk space (more space is recommended for large data source comparisons)
- High resolution monitor (1024 x 768), 16-bit display (or greater)
- Internet Explorer 6 Service Pack 1 or later (Windows) or Mozilla (Linux)

Operating System Requirements

Change Manager supports the following operating systems:

- Microsoft Windows XP (x86-32, Win32)
- Microsoft Windows Server 2003
- Microsoft Windows Vista

DBMS Support

Change Manager supports the following platforms:

- IBM DB2 LUW v8 and v9
- Oracle 8i - 11g
- Microsoft SQL Server 2000, 2005, and 2008
- Sybase ASE 12.5 and 15, 15.0.1, and 15.0.2

Installation Notes

Change Manager can be installed as a standalone application (RCP installation) or as an Eclipse plug-in (Plug-in Installation). Eclipse is an open source development framework that supports application plug-ins to provide additional functionality.

The Eclipse plug-in version of Change Manager requires Eclipse version 3.3 or higher, and Sun Java Standard Edition 5.0 Update 11 or later, in addition to regular system requirements.

Before installing the plug-in version of Change Manager, ensure that Eclipse and Java are installed on your machine. You can download Eclipse from the following Web site: http://www.eclipse.org/downloads.

Additional Product Resources

The Embarcadero Web site is an excellent source for additional product information, including white papers, articles, FAQs, discussion groups, and the Embarcadero Knowledge Base.

Go to www.embarcadero.com/support, or click any of the links below, to find:

- Documentation
- Online Demos
- Technical Papers
- Discussion Forums
- Knowledge Base

Embarcadero Technologies Technical Support

If you have a valid maintenance contract with Embarcadero Technologies, the Embarcadero Technical Support team is available to assist you with any problems you have with our applications. Our maintenance contract also entitles registered users of Embarcadero Technologies' products to download free software upgrades during the active contract period.

To save you time, Embarcadero Technologies maintains a Knowledge Base of commonly-encountered issues and hosts Discussion Forums that allow users to discuss their experiences using our products and any quirks they may have discovered.

To speak directly with Embarcadero Technical Support, see Contacting Embarcadero Technologies Technical Support below.

NOTE: Evaluators receive free technical support for the term of their evaluation (14 days).

Contacting Embarcadero Technologies Technical Support

When contacting Embarcadero Technologies Technical Support please provide the following to ensure swift and accurate service:

Personal Information

- Name
- Company name and address
- Telephone number
- Fax number

- Email address

Product and System Information

- Embarcadero product name and version number. This information is found under Help, About.
- Your client operation system and version number.
- Your database and version number.

Problem Description

A succinct but complete description of the problem is required. If you are contacting us by telephone, please have the above information, including any error messages, available so that an Embarcadero Technical Support Engineer can reproduce the error and clearly understand the problem.

There are three ways to contact Embarcadero's Technical Support department:

- Via the Web
- Via Phone
- Via Email

Via the Web

Embarcadero Technical Support provides an online form that lets you open a Support case via the Web. To access this form, go to http://www.embarcadero.com/support/open_case.jsp.

We normally acknowledge the receipt of every case on the same day, depending on the time of submission.

Via Phone

United States

Embarcadero Technologies Technical Support phone number is (415) 834-3131 option 2 and then follow the prompts. The hours are Monday through Friday, 6:00 A.M. to 6:00 P.M. Pacific time.

For licensing issues, including Product Unlock Codes, call (415) 834-3131 option 2 and then follow the prompts. The hours are Monday through Friday, 6:00 A.M. to 6:00 P.M. Pacific time.

The Embarcadero Technologies Technical Support fax number is (415) 495-4418.

EMEA

Embarcadero Technologies Technical Support phone number is +44 (0)1628 684 499. The hours are Monday to Friday, 9 A.M. to 5:30 P.M. U.K. time.

For licensing issues, including Product Unlock Codes, call +44 (0)1628-684 494. The hours are Monday to Friday, 9 A.M. to 5:30 P.M. U.K. time

The Embarcadero Technologies Technical Support fax number is +44 (0)1628 684 401.

Via Email

United States

Depending on your needs, send your email to one of the following:

- support@embarcadero.com - Get technical support for users and evaluators
- upgrade@embarcadero.com - Request upgrade information
- key@embarcadero.com - Request a product key
- wish@embarcadero.com - Make a suggestion about one of our products

EMEA

Depending on your needs, send your email to one of the following:

- uk.support@embarcadero.com- Get technical support for users and evaluators
- uk.upgrade@embarcadero.com - Request upgrade information
- uk.key@embarcadero.com - Request a product key
- uk.wish@embarcadero.com - Make a suggestion about one of our products

Change Manager Overview

Embarcadero Change Manager offers database administrators and developers a powerful tool to simplify and automate the database change management life cycle. The schema compare and alter, data compare and synchronization, and configuration auditing capabilities let you report on database changes, roll out new releases, and troubleshoot database performance problems that result from both planned and unplanned changes. Change Manager supports IBM DB2 for LUW, Microsoft SQL Server, Oracle, and Sybase. Change Manager is available as a stand-alone application or as an Eclipse plug-in.

Key Features

Change Manager enables you to:

- Bundle and roll changes forward between development, testing, and production environments.
- Track database change over periods of time, create reports, and roll back schema, security, and configuration changes.
- Understand how vendor patches affect a database.
- Set automatic notifications that enable you to monitor changes taking place during production.

Using Change Manager

This section contains information on how to implement Change Manager on a new machine and additional, basic functionality and procedures that will enable you to begin using the product almost immediately upon installation.

This section contains the following topics:

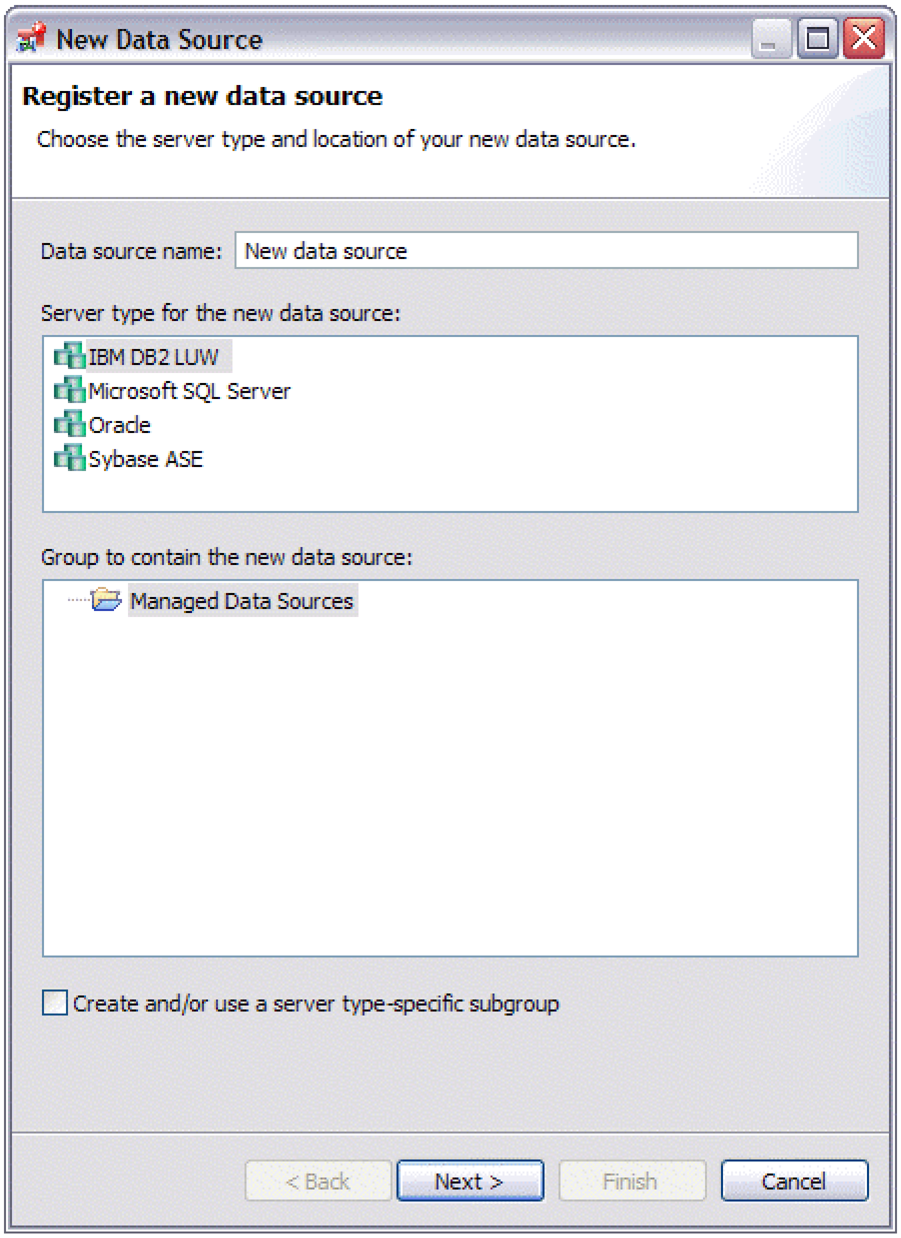
Registering Data Sources

When you execute comparison jobs, Data Source Explorer provides a tree view of data sources to use in the jobs. Data sources must be registered with the application prior to using them in jobs.

If a data source is not present in the registry, but you need to add it to **Data Source Explorer**, you can manually add it by right-clicking on the **Managed Data Sources** node in **Data Source Explorer** and selecting **New > Data Source**.

To register a new data source

1 Right-click the **Managed Data Sources** node in **Data Source Explorer** and select **New > Data Source**. The Data Source Wizard appears.



Register a new data source

Choose the server type and location of your new data source.

Data source name: New data source

Server type for the new data source:

- IBM DB2 LUW
- Microsoft SQL Server
- Oracle
- Sybase ASE

Group to contain the new data source:

- Managed Data Sources

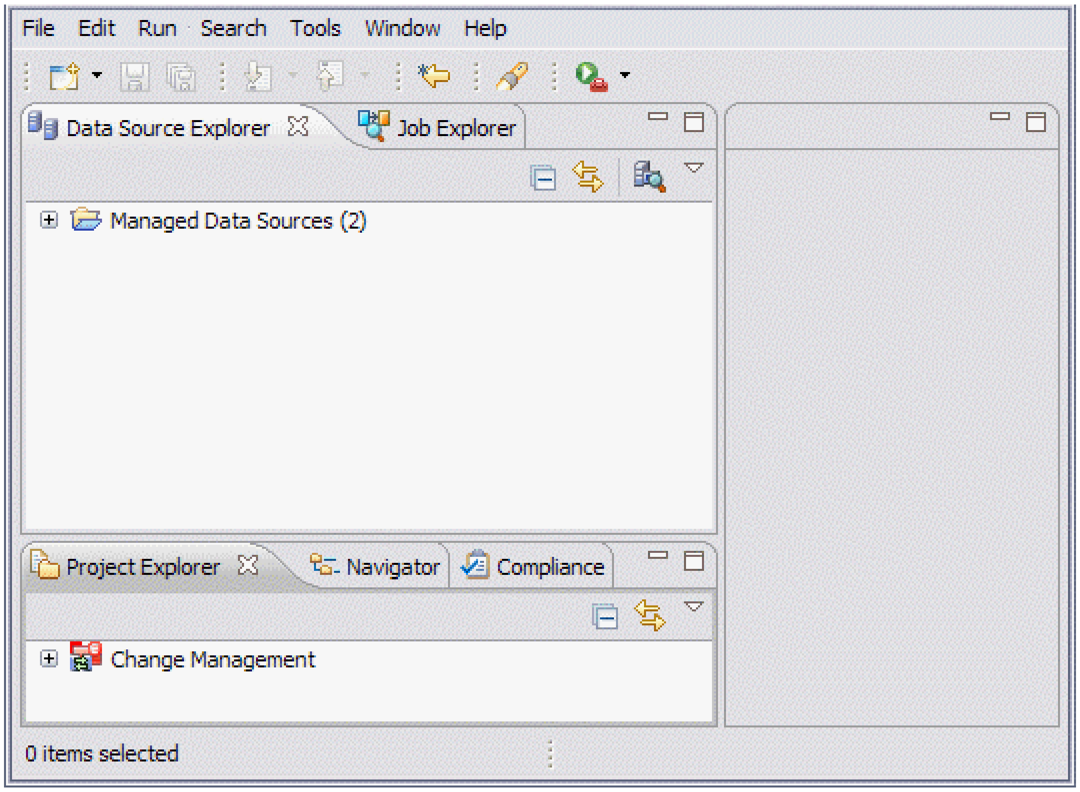
☐ Create and/or use a server type-specific subgroup

2 Follow the steps in the Data Source Wizard by selecting the data source platform and entering the appropriate connectivity parameters in the fields provided. When you have completed this task click OK.

3 The new data source appears under the Data Source Group you specified and can be used immediately in comparison jobs.

Navigating the Workbench

The Workbench is the Change Manager development environment. It provides users with the interface to create, manage, and navigate comparison jobs through interaction with Views, Editors, and menus.



The Change Manager environment is composed of a number of common interface elements to help you navigate and work inside the application environment. The individual parts of the Workbench share certain elements in common and provide you with a uniform system for working in Change Manager. Additionally, most of the interface components synchronize with one another, where one interface component on its own is dependant on other interface components to function as a whole.

It is important to understand how these components work individually as well as together. A basic understanding of how these parts function will enable a DBA to work faster and more efficiently within the product and will aid in system customization to tailor the application to individual needs and development requirements.

The following interface components compose the Change Manager work environment:

- Welcome Page
- Workbench Space
- Views
- Job Explorer
- Data Source Explorer
- Project Explorer
- Navigator

- Compliance
- SQL Errors View
- Data Diff View
- SQL Log
- SQL Errors Log

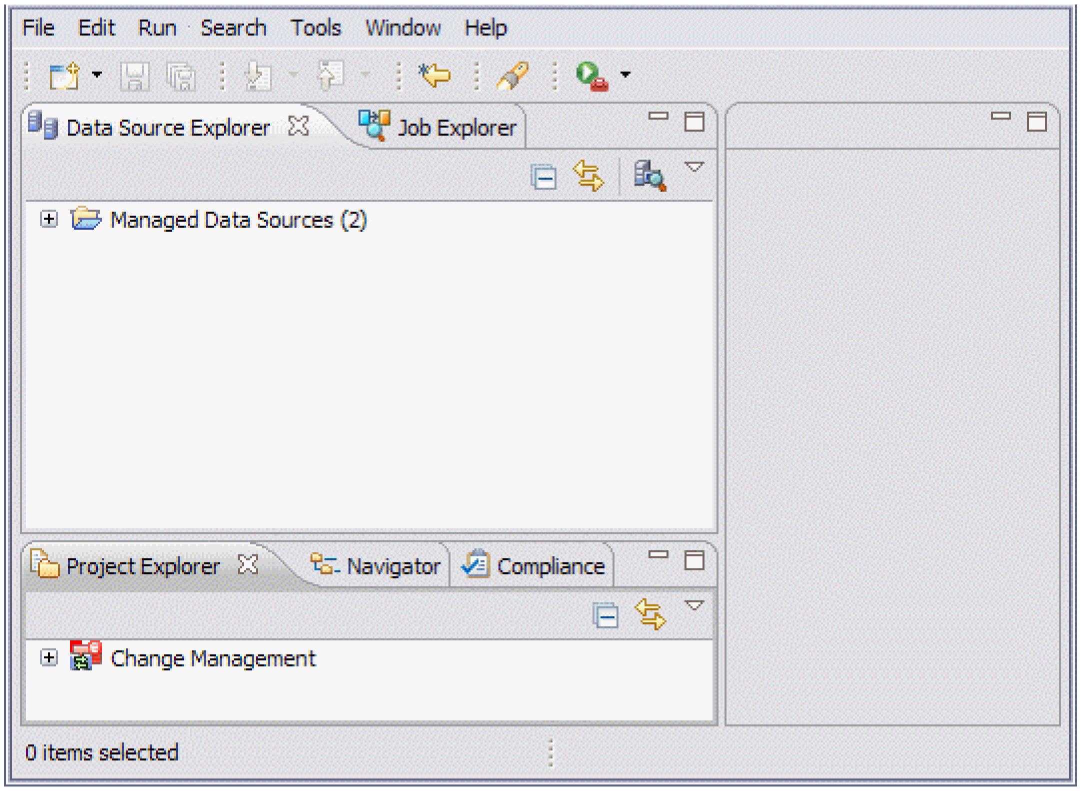
Welcome Page

The Welcome page is the first screen a user encounters when Change Manager is launched. It is composed of icons that are clicked to access tutorials and other information to help you get started. The Welcome page is intended to provide easy access to information that may be of value to new users.

Each module can be accessed directly via the **Welcome** page by clicking on its name. Additionally, a **Workbench** link is offered, which takes you to the Workbench space rather than opening one of the specified jobs. You can choose to open the Welcome page each time you start Change Manager by selecting the **Show on Startup** check box.

Workbench Space

The Workbench is the Change Manager desktop development framework. It provides users with an interface to register, manage, and compare data sources through interaction with Views and Editors.



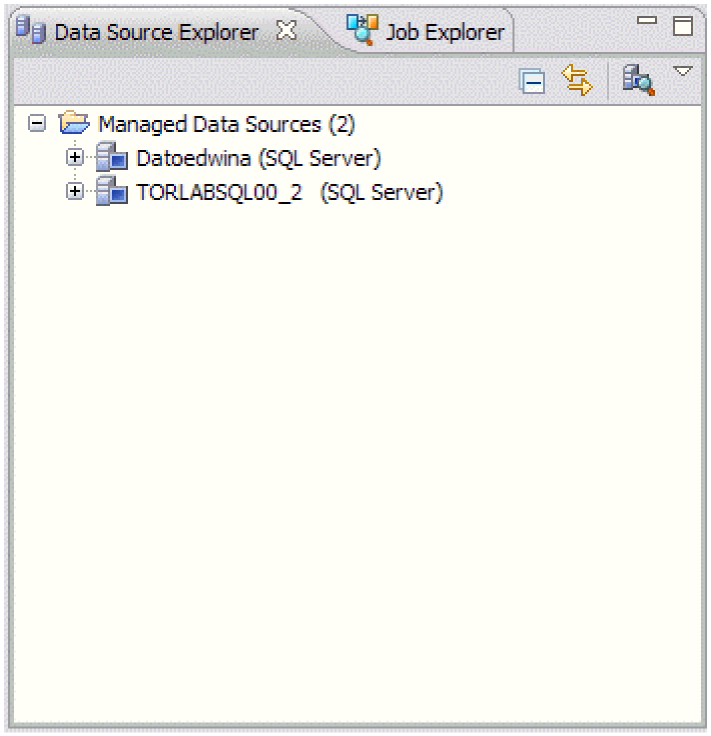
The Workbench Space is composed of Views and Editors (described below), as well as the Menu Bar and Command Toolbar.

- The **Menu Bar** contains command menus. These commands enable users to control Change Manager functionality and processes, and are either specific to the whole application, or individual Views and Editors, depending on the currently-selected frame in the Workbench Space.
- The **Command Toolbar** provides icons that issue common Menu Bar commands.

Views

A View is a Workbench interface component typically used to navigate a hierarchy of information (such as resources in the Workbench), open Editors for data source comparison purposes, or display the properties of various elements.

Views support Data and Configuration Comparison Editors and provide alternative presentations or navigations of the information in the Workbench. For example, Data Source Explorer provides a list of all registered data sources in Change Manager, organized by group folder.



Views can be active or inactive, but only one view can be active at any one time. An active view will have its title highlighted and is the target for common commands such as cut, copy, and paste. Views can be stacked with other views to create tabbed notebooks and all views contain two command menus -- one accessed by right-clicking the View tab, and the second by clicking the icon in the View Toolbar. These menus will contain Menu Bar commands pertinent to the selected View.

The following navigation actions can be performed on Views:

Move a View

Click the tab of the View and drag your mouse. A frame indicates where the view will fit in the Workbench Space. Release the button to move the View to its new location.

Toggle a ‘Floating’ View

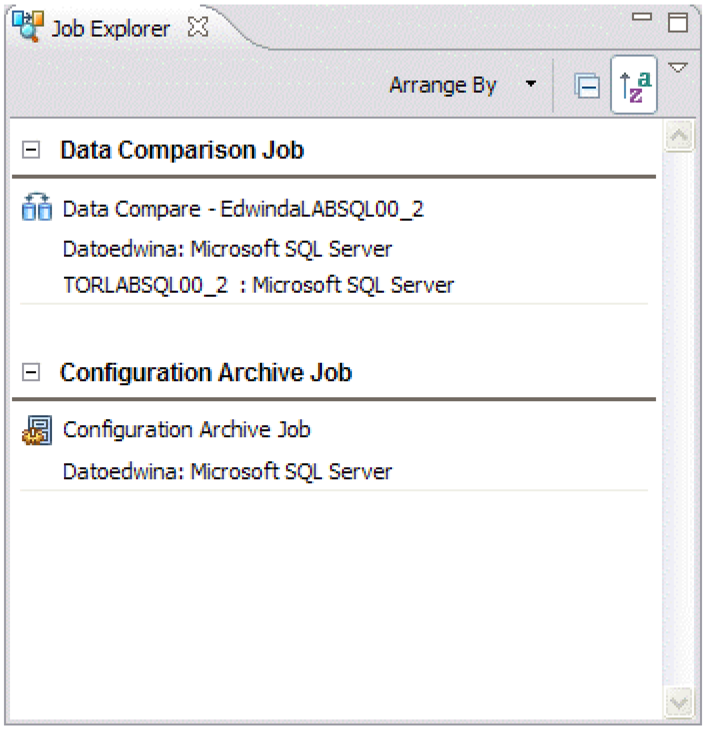
Right-click the tab of the View and select Detached. The View switches to a ‘floating’ window format that can be used to position it overtop of ‘embedded’ views in the Workbench Space, as needed.

Build a Stacked View

Drag the View over another View and release the mouse button when the frames meet. Both Views overlap and create a tabbed ‘book’ within the same frame. Multiple Views can be combined in this format.

Job Explorer

Job Explorer provides an interface for managing existing jobs in Change Manager.



Data Comparison Job

Data Compare - EdwindaLABSQL00_2
Datoedwina: Microsoft SQL Server
TORLABSQL00_2 : Microsoft SQL Server

Configuration Archive Job

Configuration Archive Job
Datoedwina: Microsoft SQL Server

Jobs are organized by date created, source and target platform types, last modified, last executed, job name, data source name, job type and module, job type, and module.

Use Job Explorer when searching for jobs of a specific type or name.

To view jobs in Job Explorer

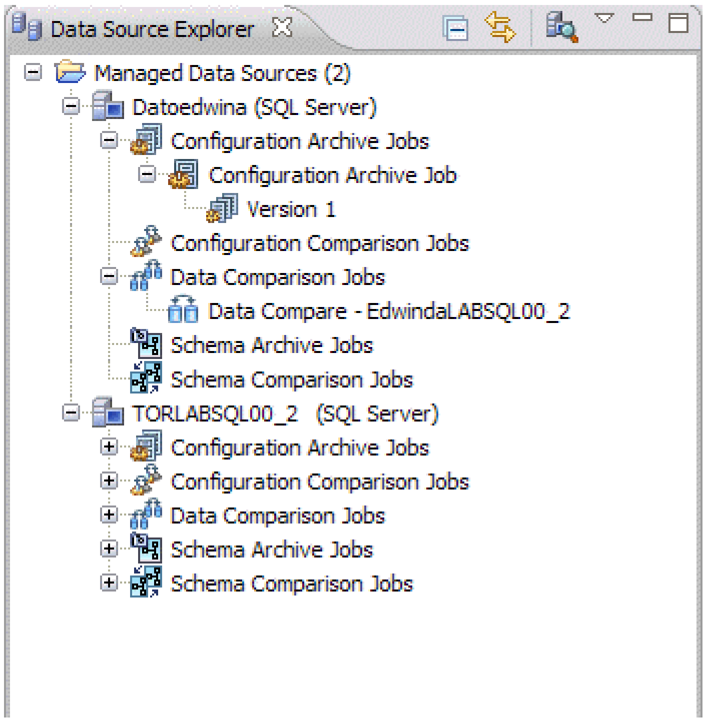
1 Navigate to Job Explorer and click **Arrange By** to view a list of categories by which you can organize the View.

2 Select a category. Job Explorer displays all jobs based on your new selection.

3 Double-click a job to open it in its respective Editor.

Data Source Explorer

Data Source Explorer provides an organizational tree of registered data sources and the comparison jobs and archives associated with them.



Data sources are registered and managed in **Data Source Explorer**. Each time a new data source is registered, its connection information must be specified, and it should be organized in the View, as needed.

Once a data source has been registered, it remains stored in a catalog and does not need to be re-registered each time you open Change Manager. It can be used in multiple jobs, archived, or otherwise 'shared' with regards to the general functionality of the application.

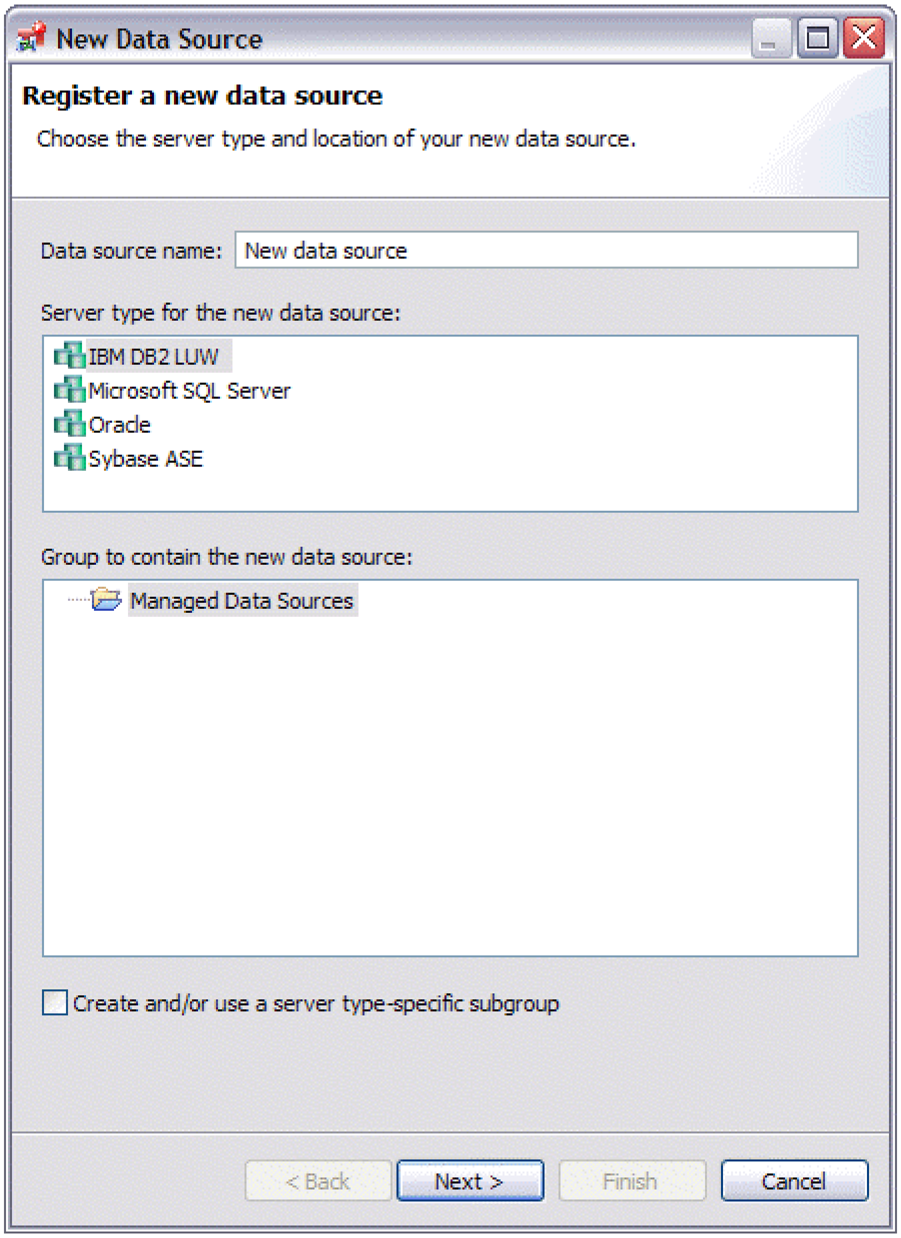
In addition to displaying registered data sources, Data Source Explorer also organizes the Change Manager jobs associated with a data source, and any configuration archives that have been generated. These jobs appear as nodes beneath the specified data source.

Additionally, you can filter the view in the tree based on working sets. A working set is a collection of user-defined data sources that can be organized, as needed, in Change Manager. Click the menu icon in the **Data Source Explorer** toolbar and choose **Select Working Set**. In the **Select Working Set** dialog, choose a working set from the list or click **New** to define one.

When you select a working set, **Data Source Explorer** only shows those data sources you specified for the set. To change the view back to the default tree, click the menu icon and choose **Deselect Working Set**.

To add a data source

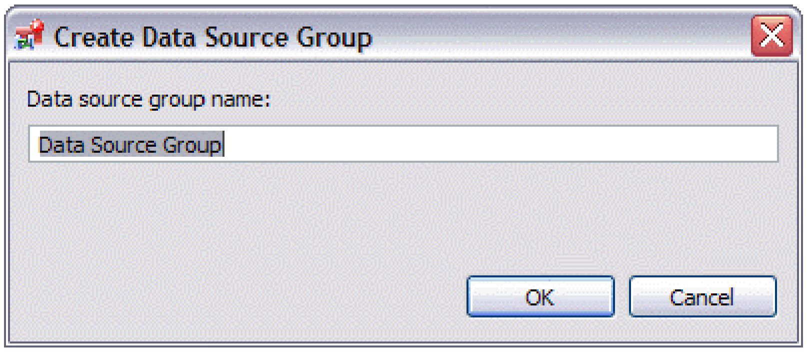
1 Right-click on the **Managed Data Sources** node in **Data Source Explorer** and select **New > Data Source**. The **Add Data Source** dialog appears.


New Data Source
Register a new data source
Choose the server type and location of your new data source.
Data source name:
New data source
Server type for the new data source:
IBM DB2 LUW
Microsoft SQL Server
Oracle
Sybase ASE
Group to contain the new data source:
Managed Data Sources
Create and/or use a server type-specific subgroup
< Back
Next >
Finish
Cancel

2 Follow the steps in the Data Source Wizard by selecting the data source platform and entering the appropriate connectivity parameters in the fields provided. Click **OK** when you are finished. The new data source appears in Data Source Explorer.

To create Data Source Group folders and organize Data Source Explorer objects

1 Right-click on the **Managed Data Sources** node in **Data Source Explorer** and select **New > Data Source Group**. The **Create Data Source Group** dialog appears.



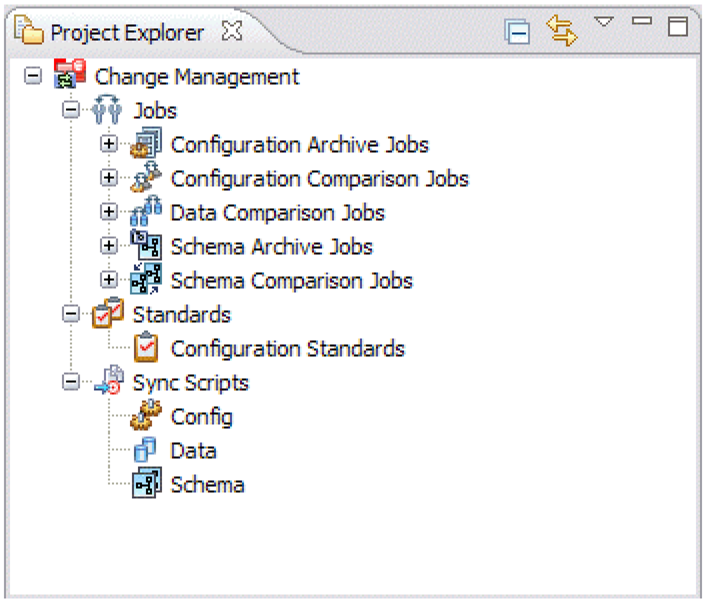
Data source group name: Data Source Group

2 Enter a meaningful name for the new group in the **Name** field and choose a parent folder, if any.

3 Click **Finish**. The **Data Source Group** folder is added to **Data Source Explorer**. You can drag and drop data sources in the folder or add new data sources to the folder, as required.

Project Explorer

Project Explorer provides a view that organizes jobs, standards, and synchronization scripts into a tree-structured project folders for management and source control purposes.



Each branch of the tree is organized by **Jobs**, **Standards**, and **Sync Scripts**. respectively. Elements can be accessed by expanding the specified branch and double-clicking on the it.

For example, if you wanted to open an existing **Data Comparison Job**, you would navigate to the **Jobs > Data Comparison Jobs** node and double-click the job to open it in the **Data Comparison Job Editor**.

Files contained in a project can be shared by a group of administrators by connecting it to a source code control and then accessed via the **Navigator**.

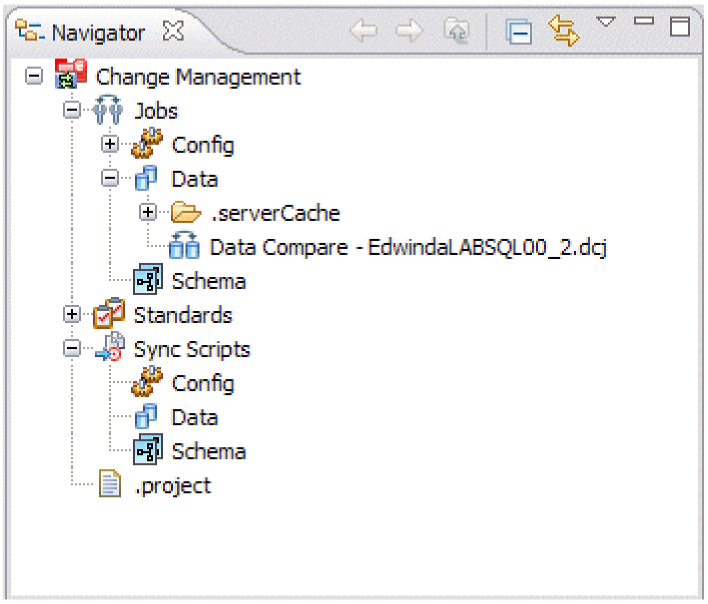
Standards are created via the Configuration Standard Editor and provide a customized list of configuration properties to use in configuration comparison jobs. Once a Standard has been defined, it automatically appears Project Explorer and can be assigned to configuration jobs, or modified, as required.

Sync Scripts are generated by the system whenever a synchronization job is run. Synchronization scripts are created in the Data Comparison Job Editor and Schema Comparison Job Editor, respectively.

Navigator

The **Navigator** provides a tree of all files in each of your projects.

You use the Navigator to share files via a source control system for jobs, results output, standards, and synchronization scripts.



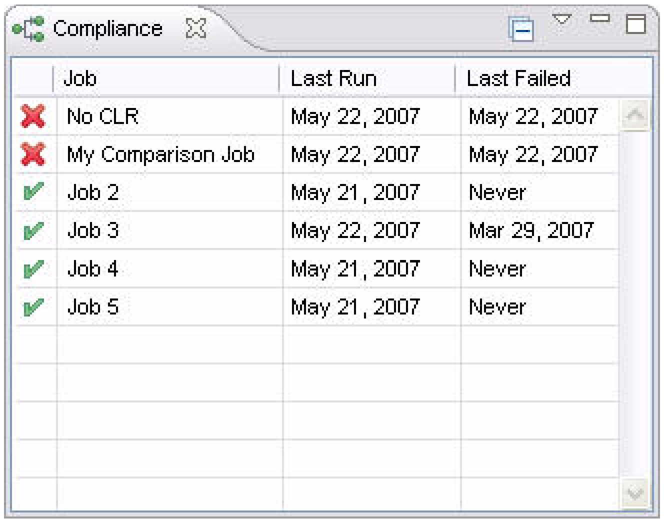
Whenever a job, Standard, or synchronization script is generated by the system, its output is saved to a file that is stored in a directory and displayed in the **Navigator** tree. These files can then be shared in a team environment for the purposes of administering and developing the data source enterprise.

All jobs, synchronization scripts, and archive versions are stored in XML format, so that objects can be merged and differentiated between different versions of the same object.

Before you can implement source code control, you need to install a third-party plug-in and the appropriate client tools. See Using Source Control Systems for more information on this process and what it involves.

Compliance

The **Compliance** view enables users to quickly reference all configuration comparison jobs and schema comparison jobs that have been saved in Change Manager, and determines if those jobs are currently in compliance with organization standards, or contain modified properties.



	Job	Last Run	Last Failed
✗	No CLR	May 22, 2007	May 22, 2007
✗	My Comparison Job	May 22, 2007	May 22, 2007
✓	Job 2	May 21, 2007	Never
✓	Job 3	May 22, 2007	Mar 29, 2007
✓	Job 4	May 21, 2007	Never
✓	Job 5	May 21, 2007	Never

The **Compliance** view contains a list of any configuration or schema comparison jobs that have been run and specifies:

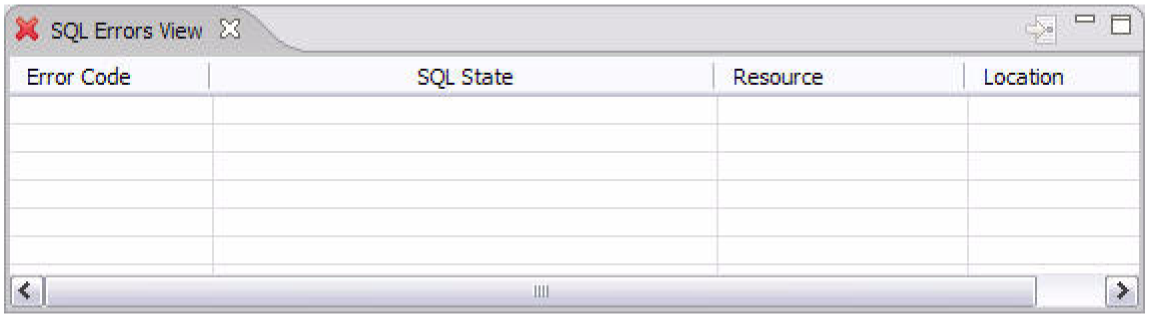
- the last time the job was executed.
- the last time the job failed, if applicable.

To enable a job in the **Compliance** view, you must enable the **Track Results in Compliance Editor** check box on the **Overview** tab in Configuration Comparison Job Editor or Schema Comparison Job Editor,respectively, per selected job.

SQL Errors View

When comparison jobs are executed in Change Manager, a SQL script can be generated that provides synchronization code to balance the data repositories of the two comparison data sources. A synchronization script can be saved and retrieved at any time, and executed within the application.

The **SQL Errors** view logs any errors that may occur when executing SQL scripts within Change Manager.

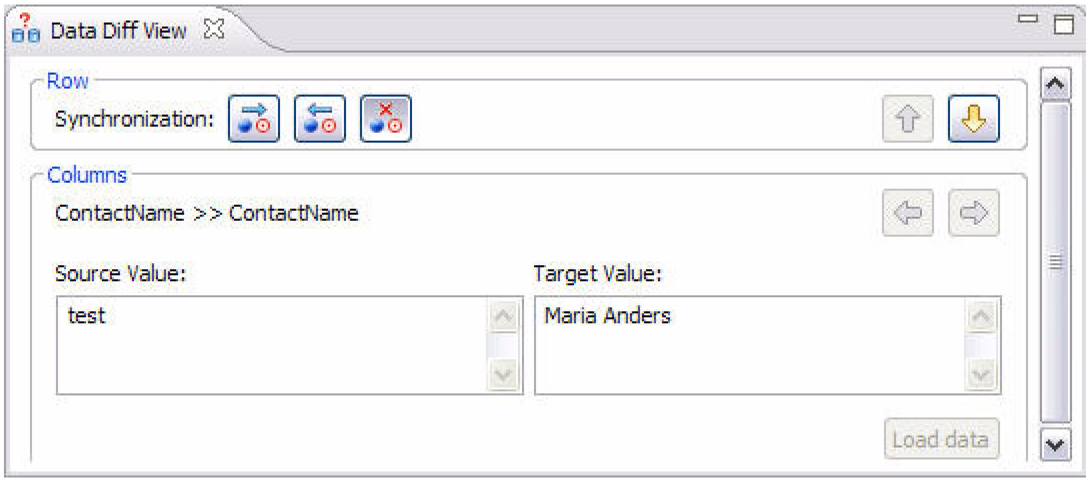


Error Code	SQL State	Resource	Location

SQL Errors are listed by SQL **Error Code**, **SQL State**, **Resource**, and **Location**.

Data Diff View

The **Data Diff** View displays the values of a mismatched row pairs when viewing the results of a data comparison job. The view contains row synchronization icons and the specific data values of the selected pair.



You can specify how data is resolved in the **Data Diff** view by selecting one of the appropriate icons beside the **Synchronization** prompt. By row, you can choose to **Make Target Match Source for Selected Row**, **Make Source Match Target for Selected Row**, or **Do Not Resolve Selected Row**, respectively.

This view enables you to specify changes without resolving the repositories of both data sources via synchronization scripts.

SQL Log

The **SQL Log** captures all SQL commands executed by SQL Editor and the system. **SQL Log** entries are listed by **SQL Statement** name, **Date** issued, **Host/Server**, **Service**, **User**, **Source**, and the **Time** (in milliseconds) it took to execute the command.

SQL Errors Log

The **SQL Errors** log automatically logs all SQL errors encountered when SQL commands are executed through Change Manager. Errors are listed by **Error Code**, **SQL State**, error **Details**, **Resource**, and the **Location** of the error in the SQL file.

Editors

Editors are workbench interface components that enable Change Manager processes. That is, data, configuration, and schema comparison jobs. Editors are treated differently than views because as stand-alone components, they function at an individual level rather than a supportive one.

- Data Comparison Job Editor
- Configuration Comparison Job Editor
- Configuration Standard Editor
- Configuration Archive Editor
- Schema Comparison Job Editor
- Schema Archive Editor
- SQL Editor

The following navigational-type actions can be performed on all Editors in general:

Move an Editor

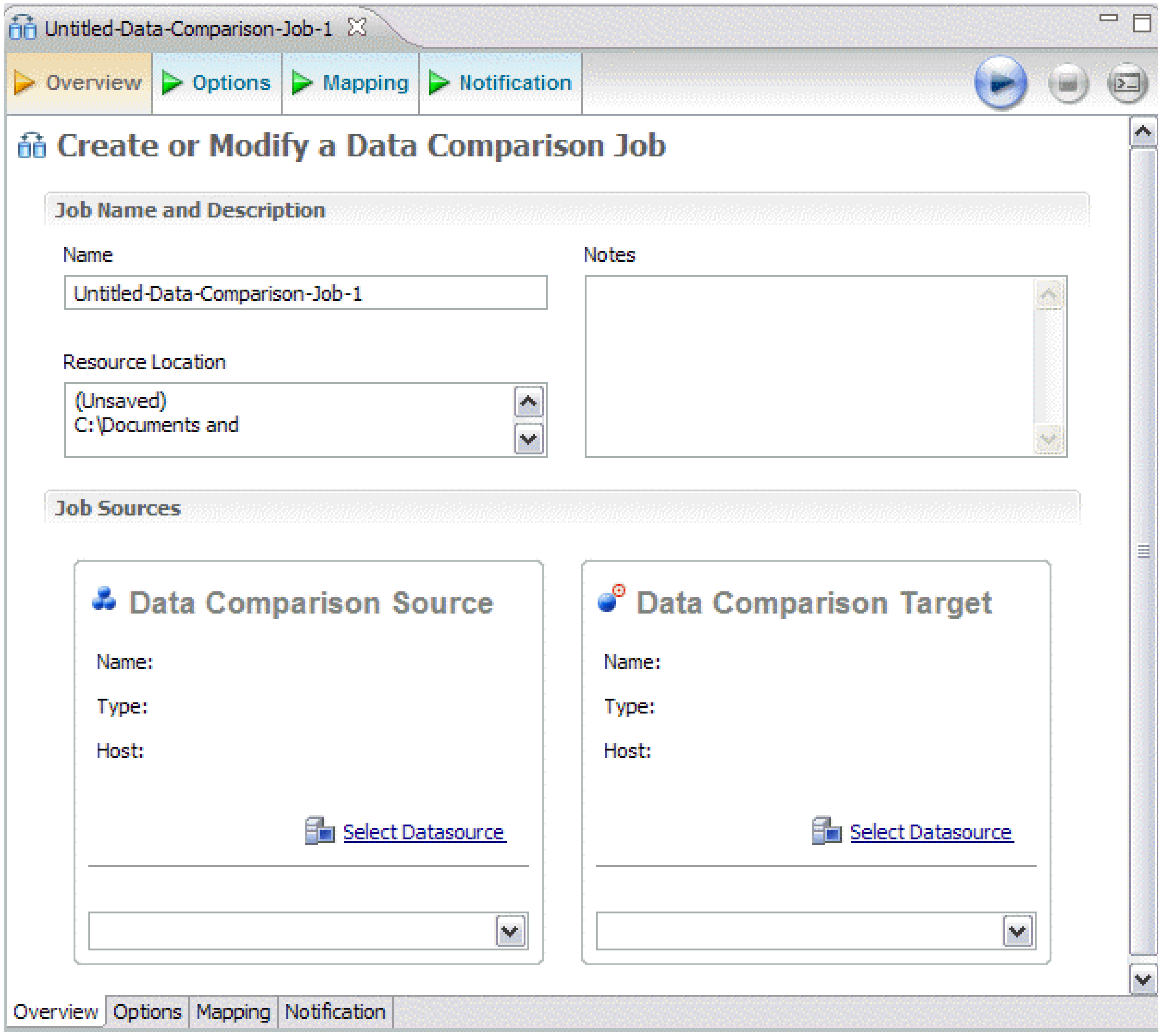
Click the tab of the Editor and drag your mouse. A frame indicates where the Editor will fit in the Workbench Space. Release the button to move the Editor to its new location.

Build a Tabbed Notebook

Drag the Editor over another Editor and release the mouse button when the frames meet. Both Editors overlap and create a tabbed ‘book’ within the same frame. Multiple Editors can be combined in this format.

Data Comparison Job Editor

The Data Comparison Job Editor compares the data repositories of two specified data sources at the table and row levels. Additionally, the editor provides the ability to specify table mappings and further manipulate data via matching functions that synchronize it at the global, table, and row level, as specified by the user, and provides high-level mapping/refinement options for a job, based on data source (database, schema, or owner selection).



Create or Modify a Data Comparison Job

Job Name and Description

Name: Untitled-Data-Comparison-Job-1

Notes

Resource Location: (Unsaved) C:\Documents and

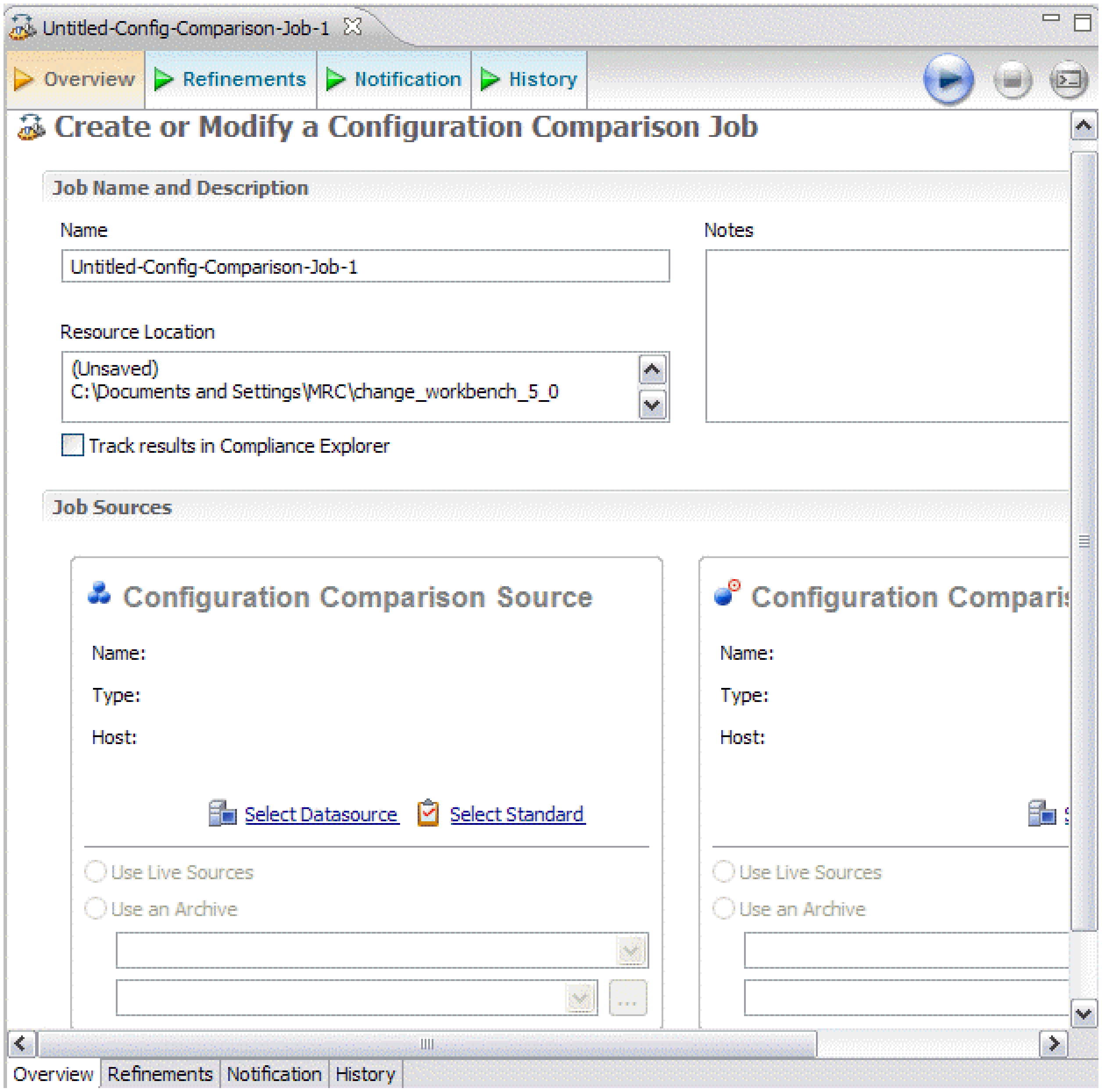
Job Sources

Data Comparison Source
Name:
Type:
Host:
Select Datasource

Data Comparison Target
Name:
Type:
Host:
Select Datasource

Configuration Comparison Job Editor

The Configuration Comparison Job Editor compares the configuration settings of two specified data sources. Additional, the Editor provides the ability to define standard settings templates and archive individual data source properties to be used in comparison jobs.



Create or Modify a Configuration Comparison Job

Job Name and Description

Name: Untitled-Config-Comparison-Job-1

Notes

Resource Location: (Unsaved) C:\Documents and Settings\MRC\change_workbench_5_0

Track results in Compliance Explorer

Job Sources

Configuration Comparison Source

Name:

Type:

Host:

Select Datasource

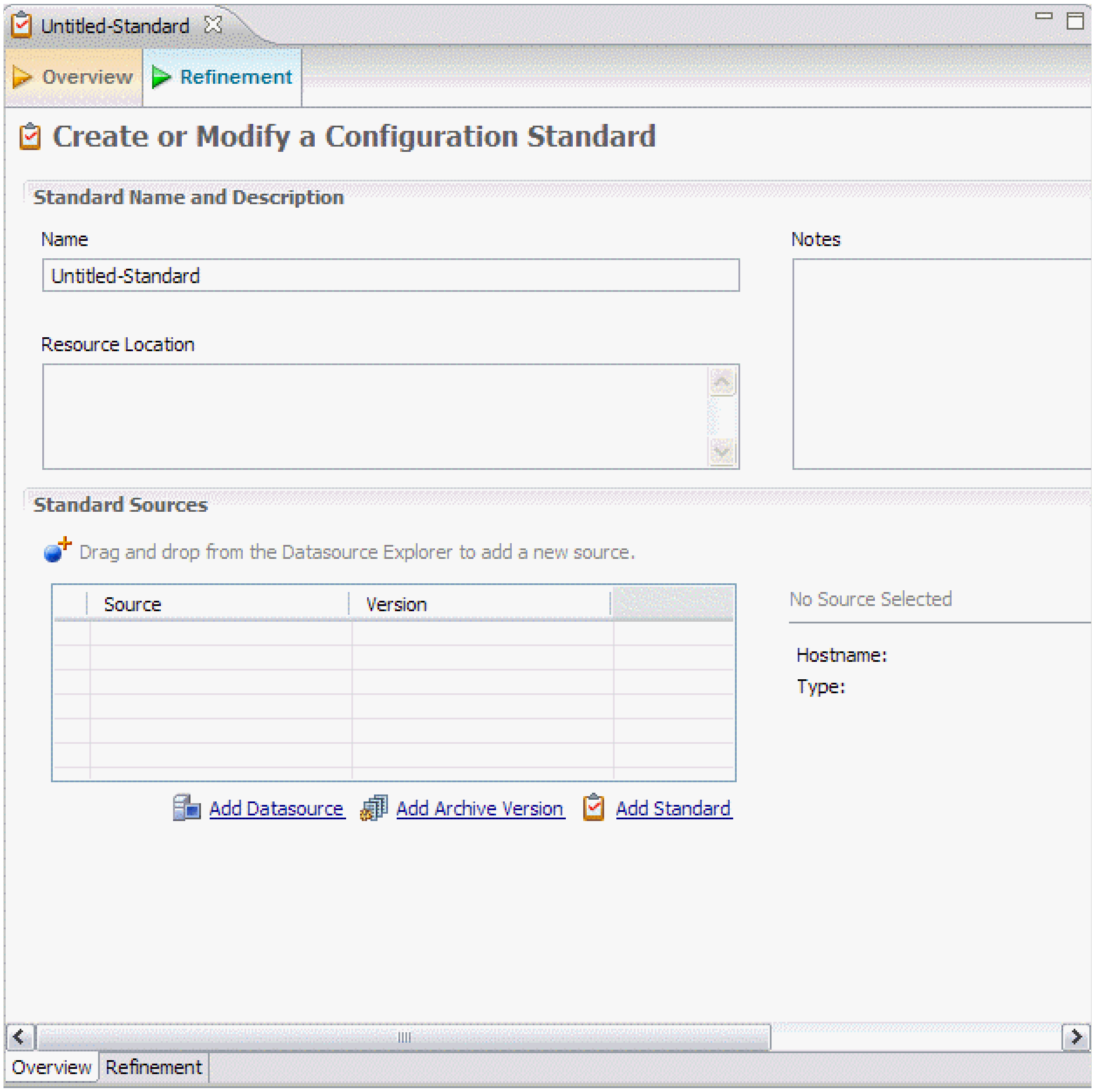
Select Standard

Use Live Sources

Use an Archive

Configuration Standard Editor

The Configuration Standard Editor provides an interface for creating a configuration Standard in Change Manager. Configuration Standards are used to provide customized data source configuration properties, and, once defined, are used in configuration comparison jobs against data sources and archives to provide a specific set of values and threshold operators that ‘check’ enterprise settings and ensure all network system configuration values fall within acceptable ranges.



Create or Modify a Configuration Standard

Standard Name and Description

Name: Untitled-Standard

Resource Location

Notes

Standard Sources

Drag and drop from the Datasource Explorer to add a new source.

Source	Version

No Source Selected

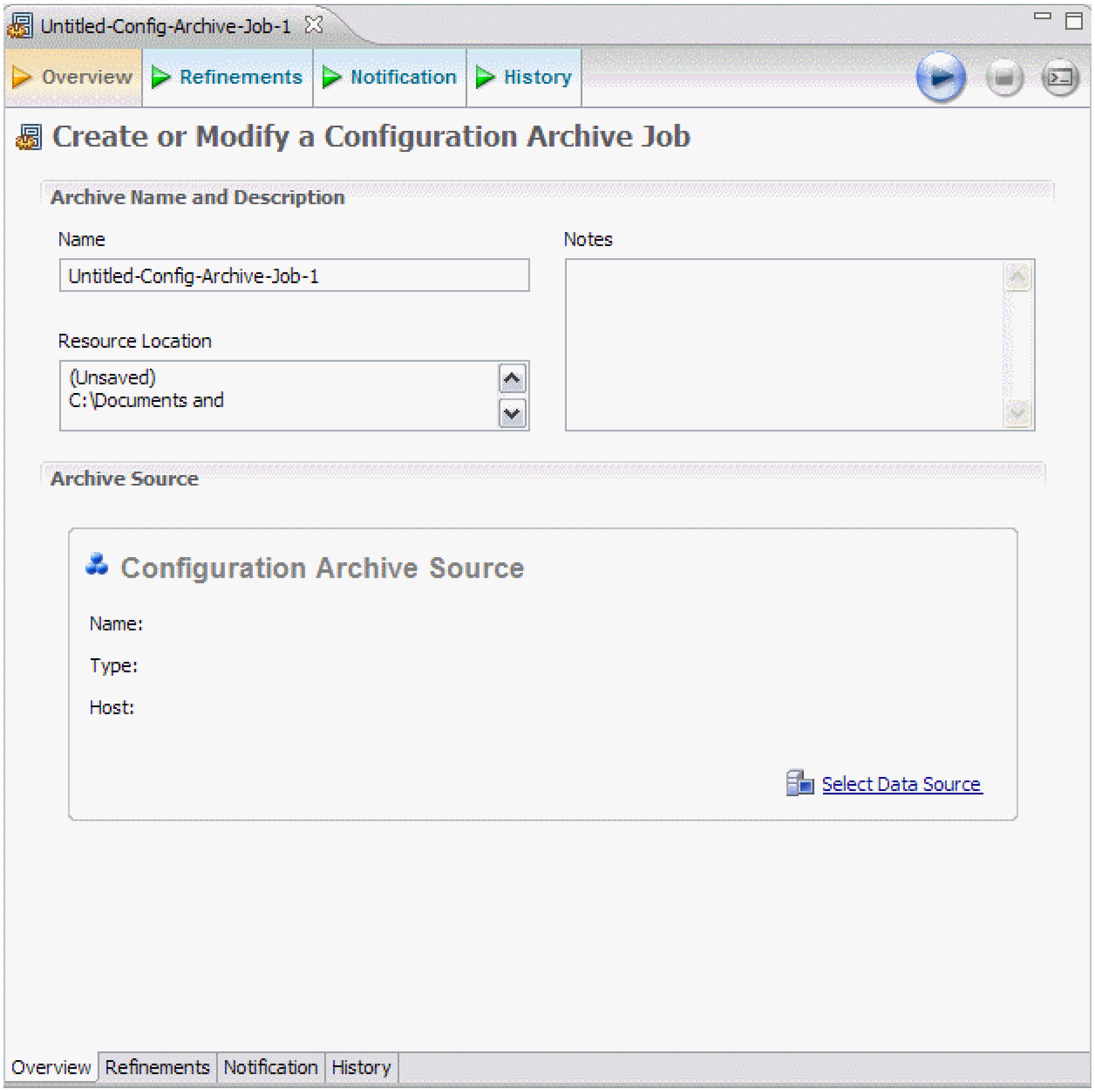
Hostname:

Type:

Configuration Archive Editor

The Configuration Archive Editor provides the ability to capture a ‘snapshot’ of existing data source configuration values. It provides the ability to version the same archive multiple times, so users can store and utilize archive configuration values from different points in time.

Configuration archives are snapshots of existing data source configuration values. Each data source in your environment can contain multiple archive versions. These versions can be used in configuration comparison jobs against other data sources, archives, or standards.



Create or Modify a Configuration Archive Job

Archive Name and Description

Name: Untitled-Config-Archive-Job-1

Notes:

Resource Location: (Unsaved) C:\Documents and

Archive Source

Configuration Archive Source

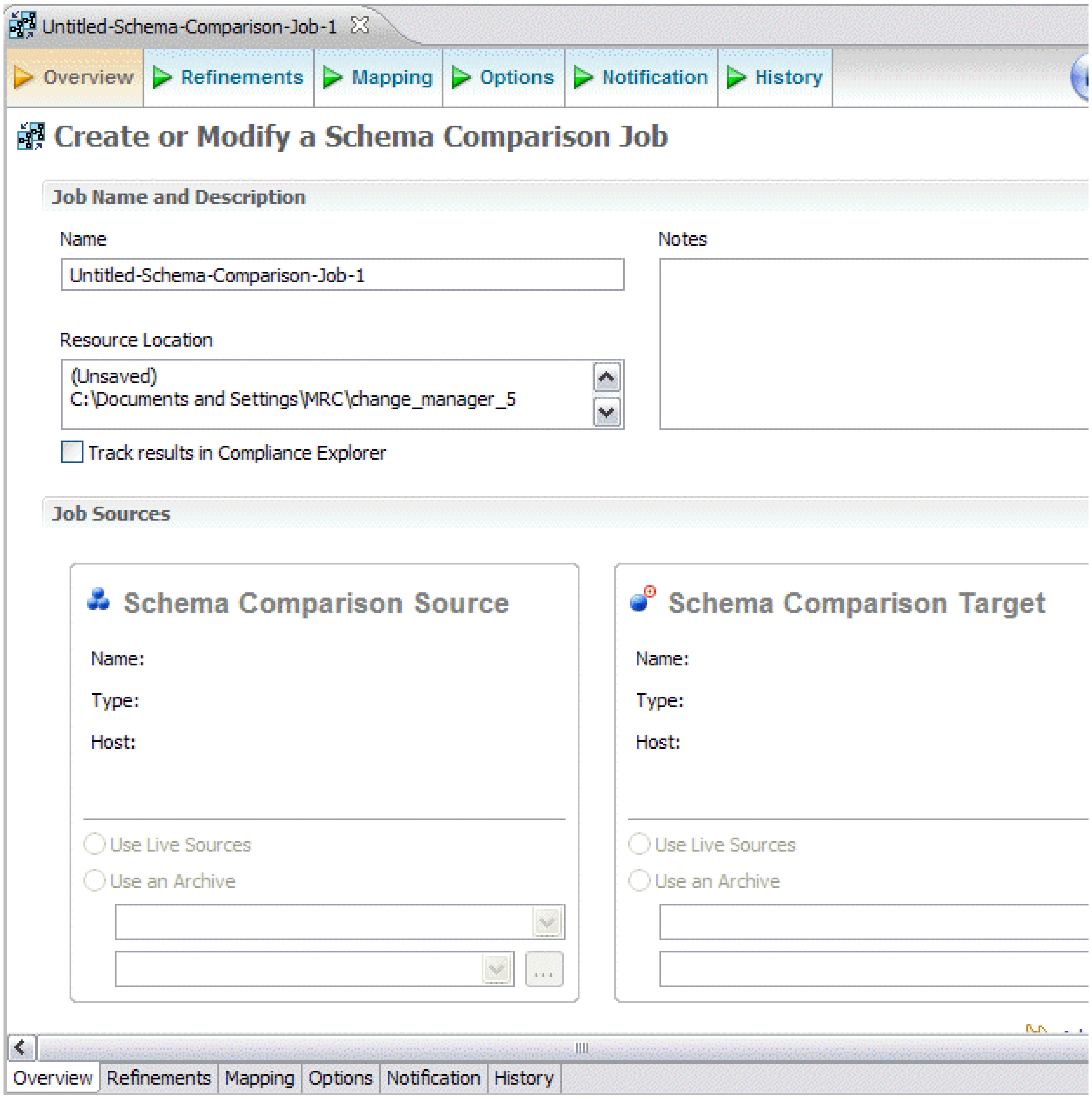
Name:

Type:

Host:

Schema Comparison Job Editor

The Schema Comparison Editor provides an interface for the comparison of two schemas on registered data sources, for the purposes of identifying changes and deviations between the two.



Create or Modify a Schema Comparison Job

Job Name and Description

Name: Untitled-Schema-Comparison-Job-1

Notes

Resource Location: (Unsaved) C:\Documents and Settings\MRC\change_manager_5

☐ Track results in Compliance Explorer

Job Sources

Schema Comparison Source

Name:

Type:

Host:

○ Use Live Sources

○ Use an Archive

Schema Comparison Target

Name:

Type:

Host:

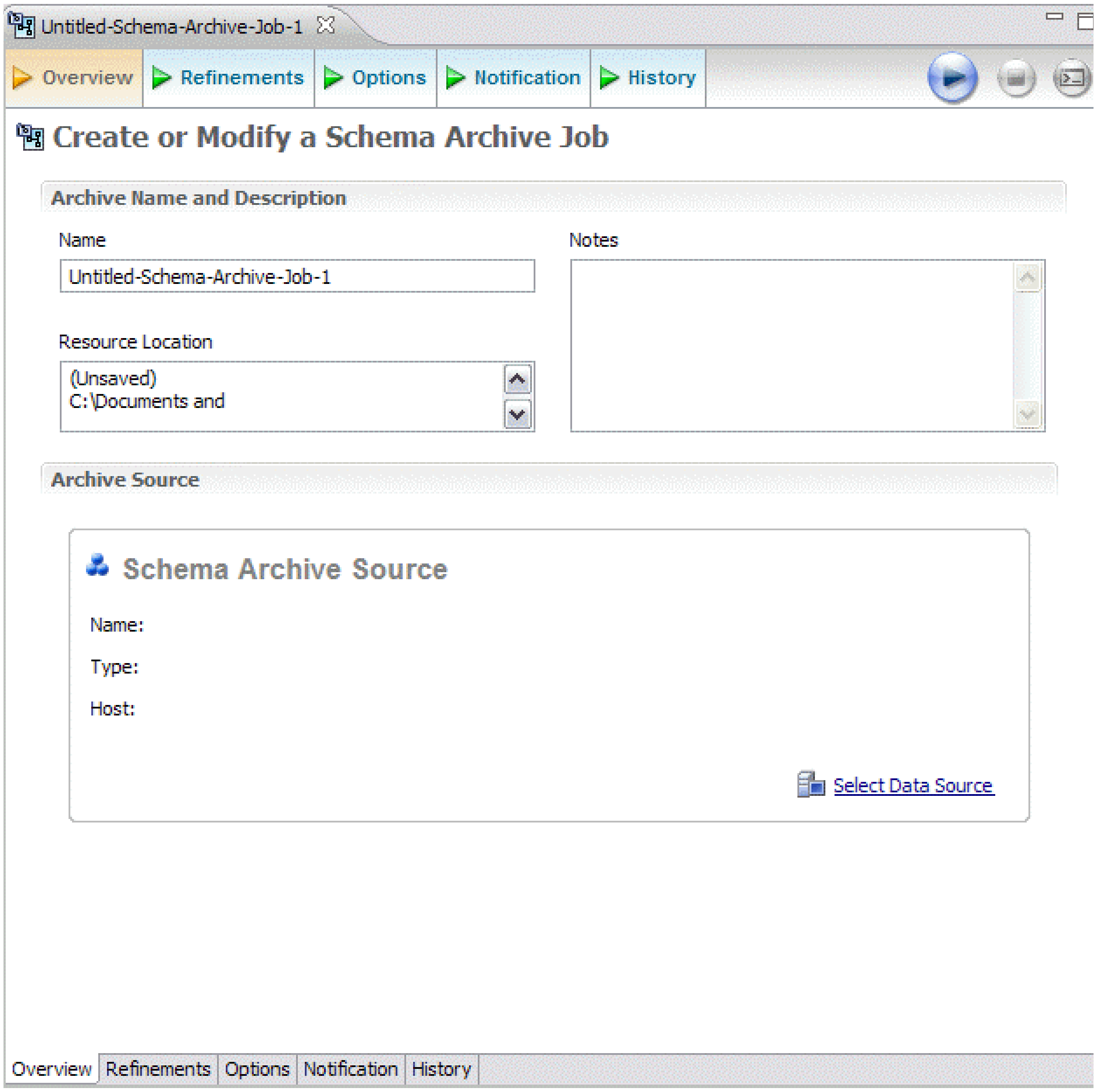
○ Use Live Sources

○ Use an Archive

You can compare the schemas of two live data sources, a live data source and an archive, or two archives. For example, in order to ensure that approved changes are rolled forward between testing and production environments, you could first archive a schema baseline and use it to compare with the live environment once the move is completed.

Schema Archive Editor

The Schema Archive Editor provides the ability to capture a ‘snapshot’ of an existing schema archive. The interface provides the ability to version the same archive multiple times, so users can store and utilize archived schemas from different points in time.



Create or Modify a Schema Archive Job

Archive Name and Description

Name: Untitled-Schema-Archive-Job-1

Notes:

Resource Location: (Unsaved) C:\Documents and

Archive Source

Schema Archive Source

Name:

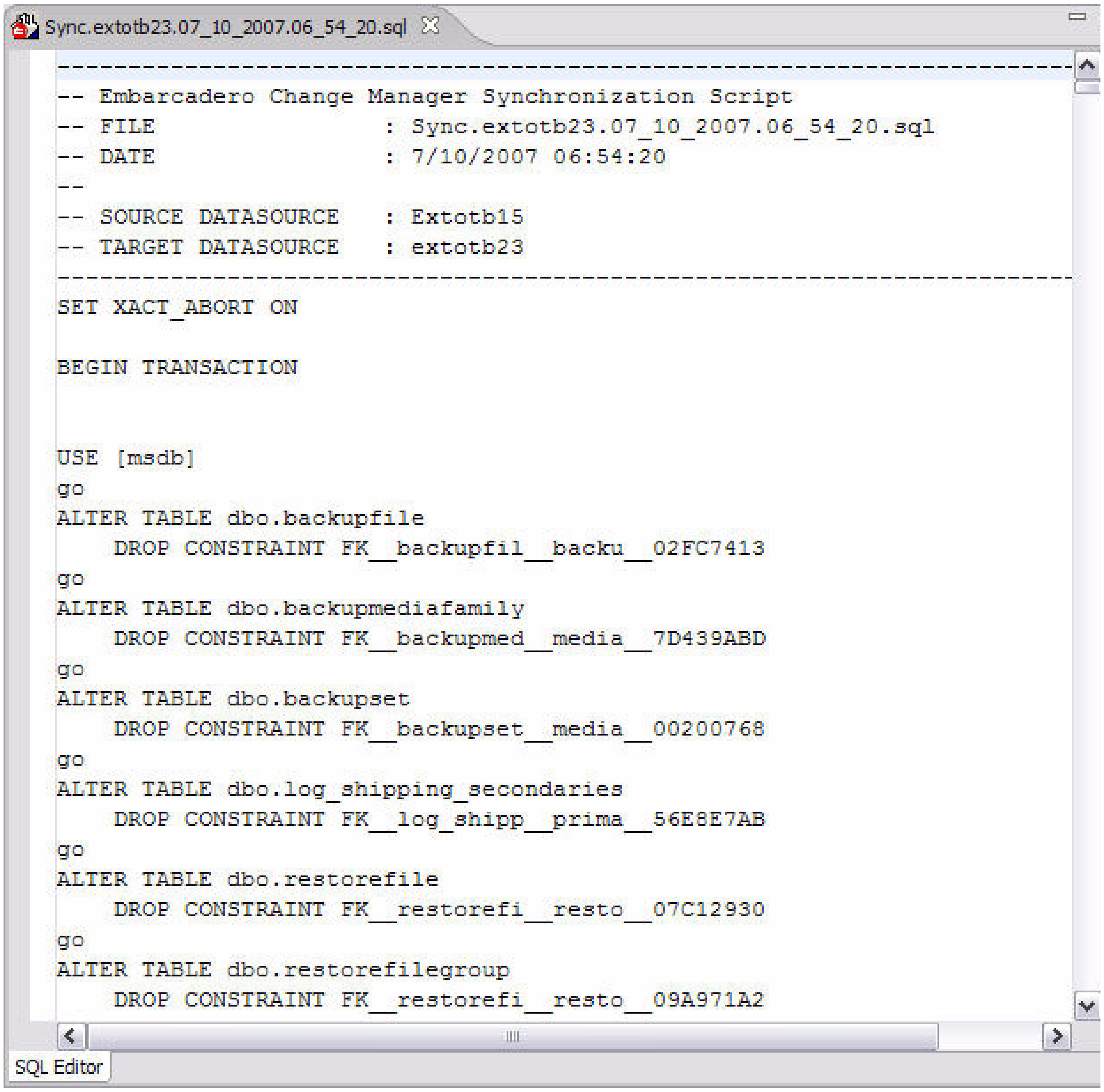
Type:

Host:

Schema archives are snapshots of existing schema values. Each data source in your environment can contain multiple archive versions. These versions can be used as baselines in schema comparison jobs against other data sources, archives, or standards.

SQL Editor

SQL Editor is an interface that provides the viewing and modification of jobs generated by the Data Comparison Editor.

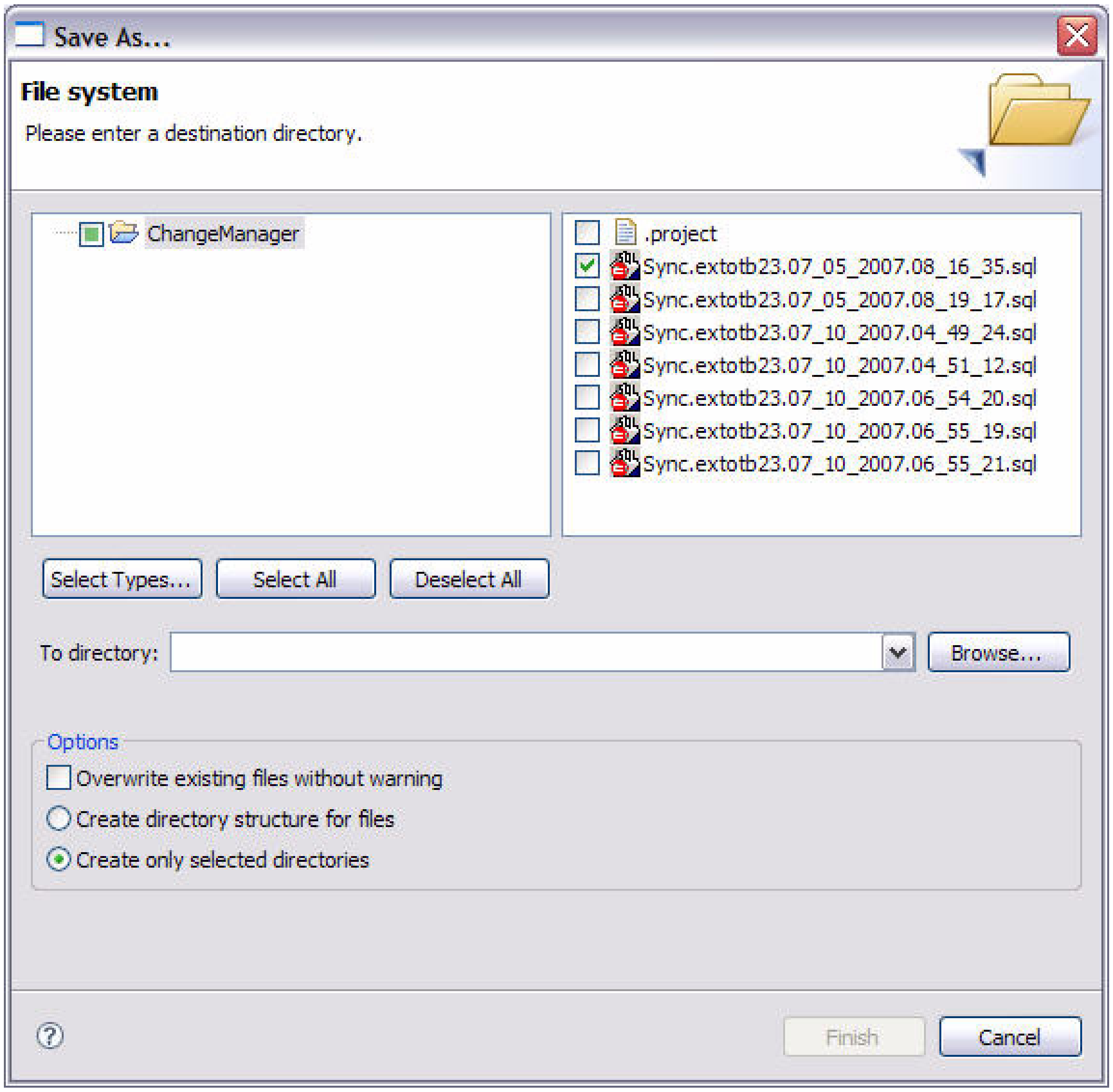


```
--------------------------------------------------------------------------
-- Embarcadero Change Manager Synchronization Script
-- FILE                  : Sync.extotb23.07_10_2007.06_54_20.sql
-- DATE                  : 7/10/2007 06:54:20
--
-- SOURCE DATASOURCE     : Extotb15
-- TARGET DATASOURCE     : extotb23
--------------------------------------------------------------------------
SET XACT_ABORT ON

BEGIN TRANSACTION


USE [msdb]
go
ALTER TABLE dbo.backupfile
    DROP CONSTRAINT FK__backupfil__backu__02FC7413
go
ALTER TABLE dbo.backupmediafamily
    DROP CONSTRAINT FK__backupmed__media__7D439ABD
go
ALTER TABLE dbo.backupset
    DROP CONSTRAINT FK__backupset__media__00200768
go
ALTER TABLE dbo.log_shipping_secondaries
    DROP CONSTRAINT FK__log_shipp__prima__56E8E7AB
go
ALTER TABLE dbo.restorefile
    DROP CONSTRAINT FK__restorefi__resto__07C12930
go
ALTER TABLE dbo.restorefilegroup
    DROP CONSTRAINT FK__restorefi__resto__09A971A2
```

All scripts, when they are generated initially, appear in SQL Editor. By default, new scripts are automatically saved and appear in the Navigator view in a default directory specified by Change Manager. However, scripts can be modified manually and saved via the **File > Save As** menu command, and stored in different directories, as required.


Save As...
File system
Please enter a destination directory.
ChangeManager
.project
Sync.extotb23.07_05_2007.08_16_35.sql
Sync.extotb23.07_05_2007.08_19_17.sql
Sync.extotb23.07_10_2007.04_49_24.sql
Sync.extotb23.07_10_2007.04_51_12.sql
Sync.extotb23.07_10_2007.06_54_20.sql
Sync.extotb23.07_10_2007.06_55_19.sql
Sync.extotb23.07_10_2007.06_55_21.sql
Select Types...
Select All
Deselect All
To directory:
Browse...
Options
Overwrite existing files without warning
Create directory structure for files
Create only selected directories
Finish
Cancel

You can rerun scripts that are saved in Change Manager by double-clicking the script name in the **Navigator** view.

Using Basic Commands

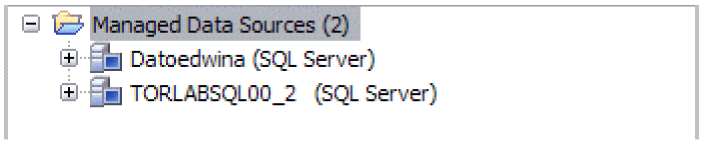
The following additional Change Manager functions may be useful when navigating, organizing, or creating new jobs, Archives, and Standards.

- Creating New Data Source Groups
- Searching for Jobs and Synchronization Scripts
- Viewing Change History
- Building Command Line Syntax and Creating Bulk Job Files
- Filtering Data Source Objects

Creating New Data Source Groups

All data sources displayed in **Data Source Explorer** can be organized in **Data Source Groups**.

Groups are represented as child nodes of the **Managed Data Sources** folder in **Data Source Explorer** and provide an additional level of organization when maintaining a large quantity of data sources.



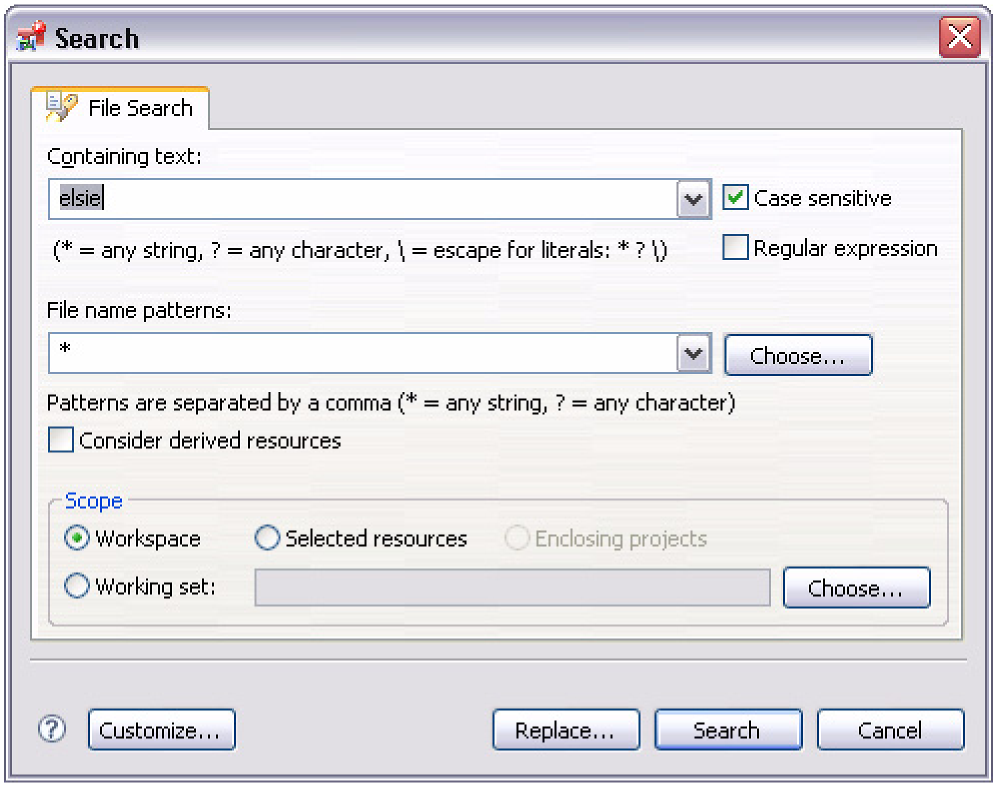
To create a Data Source Group

1. Right-click on the **Managed Data Source** icon in **Data Source Explorer**.
2. Enter a meaningful name for the new group in the **Name** field. The value entered in the field is how the group will be labeled in **Data Source Explorer**. If other **Data Source Groups** exist, select the parent folder of the new group folder in the **Choose a Parent Group** section.
3. Click **Finish**. The **Data Source Group** folder is added to **Data Source Explorer**. You can drag and drop data sources in the folder or add new data sources to the folder, as needed.

Searching for Jobs and Synchronization Scripts

The **Search** function provides a means of searching for the jobs and synchronization scripts generated by Change Manager.

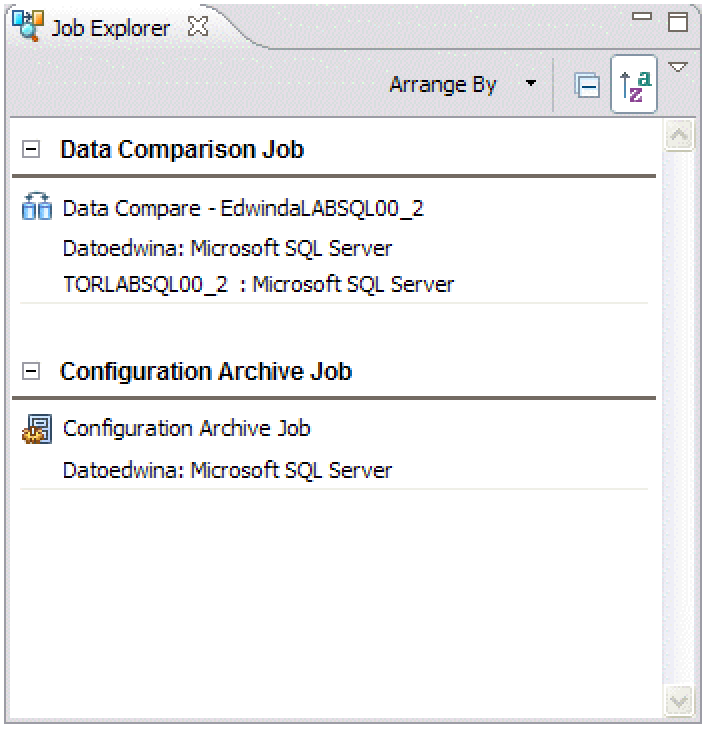
The **Search** function is accessed by clicking the **Search** icon in the Toolbar. Jobs and SQL scripts are searched by text string on the **File Search** tab of the resultant dialog.



- Type the value to search in the **Containing Text** field. Use the ***** character to indicate wildcard string values, the **?** character to indicate wildcard character values, and the \ character to indicate an escape character for literals. (*** ?** and **/**).
- Select **Case Sensitive** to take into account case when searching for string matches.
- Select **Regular Expression** to identify the string as a regular function in terms of search criteria.
- In the **File Name Pattern** field, specify the extension name of the files to search for explicitly. If the value in this field is a ***** character, the search function will search all files regardless of extension. Manually type the extensions to indicate the file type (separate multiple file types with commas), or click **Choose** and use the **Select Types** dialog to select the file extensions the process will use to search for the string. For example, selecting***.caj** will search for configuration archive job files, exclusively.
- Select **Consider Derived Resources** to include derived resources in the search.

Viewing Change History

Each time a job is run (data repository or configuration property comparison), the history of that job is recorded and maintained in the **Job Explorer View**.



Data Comparison Job

Data Compare - EdwindaLABSQL00_2
Datoedwina: Microsoft SQL Server
TORLABSQL00_2 : Microsoft SQL Server

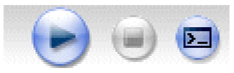
Configuration Archive Job

Configuration Archive Job
Datoedwina: Microsoft SQL Server

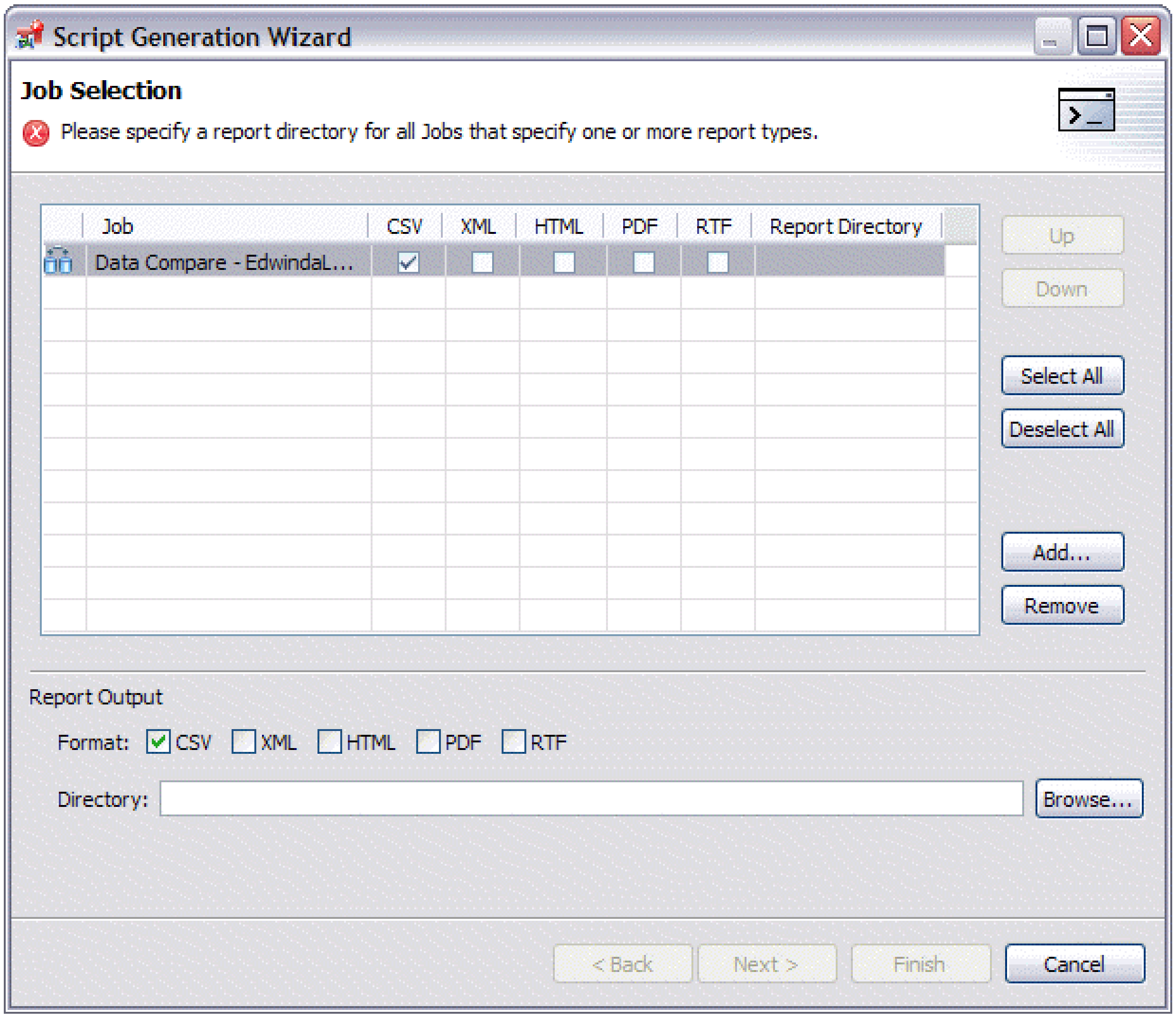
Use this view to examine all jobs that have been defined in Change Manager. You can click the **Arrange By** drop down menu to organize the list of jobs by **Date Created**, **DBMS**, **Last Modified**, **Last Run**, **Name**, **Data Source**, and **Job Type and Module**, **Job Type**, or **Module**, as required.

Building Command Line Syntax and Creating Bulk Job Files

On each tab in the job creation editors (Data Comparison, Configuration Comparison, and Archive and Standard Editors), the **Command Line Syntax** icon accesses the **Script Generation Wizard**, which enables users to create batch files for automating the process of running many jobs.



When the icon on the left-hand side of the bar is selected, the **Script Generation Wizard** appears and displays the command line syntax to execute the job.



Please specify a report directory for all Jobs that specify one or more report types.

Job	CSV	XML	HTML	PDF	RTF	Report Directory
Data Compare - EdwindaL...	✓					

Report Output

Format: [✓ CSV] XML HTML PDF RTF

Directory:

Each command line syntax is accompanied by a **Job Name**, which is unique to each job in Change Manager, and displayed in the specified editor, per job. The Script Generation Wizard enables you to generate a batch file composed of the specified syntax in a file location of your choosing. Additionally, you can specify output files in **CSV**, **XML**, **RTF**, **HTML**, or **PDF** format. These files can be saved to a specified path.

NOTE: This feature is only enabled for jobs that have been saved.

Additionally, jobs can be manually launched from the **changec** command line program, which is located in the Change Manager installation directory.

Job options must be specified by inserting the **-job** and **-output** commands when issuing a command from this location.

- The **-job** command specifies the jobs to run by job file name. This line must also include a directory location where the projects and jobs are stored.
- The **-output** command specifies reporting options for the job.

For example, the code below runs the MyCOnfigCOmpare database comparison job and generates a report in PDF format:

```
-job
"C:\Users\test1\change_workbench_5_0\workspace\DatabaseChange]MyConfigCompare.ccj"
-output PDF "c:/report"
```

In order to execute multiple jobs from the same command line, use the following syntax:

`changec -bulk` *`bulkfilename`*

Where *bulkfilename* is a file where each row has a -job attribute.

The -job command also accepts regular expressions:

- ****** indicates all sub folders in a directory location.
- * indicates any string.

For example, the following command runs all configuration archive jobs in all projects and folders in the workspace:

```
changec -job workspace:\**\*.caj
```

To create a batch file

1. In the editor of the job you want to create an execution script for, select the **Command Line Syntax** icon located in the upper-right hand corner of the interface window. The **Script Generation Wizard** appears, and the job appears in a table.
2. Add output report options, as needed, and specify the **Directory** where the files will be generated. Click **Next**.
3. Specify the **Location** where the batch file will be generated, and provide a file format for the script.
4. Click **Finish**. The batch file is generated in the directory you specified in Step **3**.

   NOTE: If you want to create a batch file to run multiple jobs,click **Add**, and then add the other jobs. The **Remove**, **Up**, and **Down** commands help you control the jobs in the file, as job order is sometimes important. For example, an archive needs to be captured before the version is used in a comparison.

To run a job from the command line

1. From the command line, enter an execution string with the following format:

   changec -job job_directory job_name -option report_format report_directory
2. The job you specified as the *job_name* runs. If you specified a report format and location, the report is generated as well.

   NOTE: The **Script Generation Wizard** builds command line options automatically and stores them in the file you specified. When the batch file is run, it executes the parameters as specified in the wizard.

Filtering Data Source Objects

Filters can be placed on data sources and corresponding data source objects to restrict their display in Data Source Explorer. This feature is useful if you have data sources that contain large numbers of database objects. You can apply filters to view only the schema objects of those you need for the development process.

There are two types of data source filters available:

- **Global filters** that affect all registered data sources in the Change Manager development environment.

- **Data source-specific filters** affect only the specified data source for which they are defined.

In both cases, data source object filters are defined via the **Object Filter Manager**, through the development of filter templates. Once defined, filter templates can be activated and deactivated as you need them.

Several filter templates can be combined at a global level or applied to a specific data source.

Global filters affect all registered data sources in the Change Manager development environment. When you create and apply a global filter to a platform vendor in Change Manager, all databases associated with that vendor are affected by the filter, as defined.

Individual global filter templates are separated, by supported data source platform, on tabs in the **SQL Filter** window. Select the appropriate tab to view existing filter templates or add new ones, as needed.

To define a global filter:

1. Select **Window > Preferences** from the Main Menu. The **Preferences** dialog appears.
2. Expand the **SQL Development** node and select the **SQL Filter** subnode. The **SQL Filter** pane appears.
3. Click **New**. The **Filter Template** dialog appears.
4. Specify the parameters of the filter template:
   - In the **Name** field, enter the name of the filter as you want it to appear in the selection window on the **SQL Filter** node.
   - The **Database Type** pane provides a list of data source objects. Deselect the data source objects that this template filters so that they do not appear in **Database Explorer** when displaying data source objects for the data source.
   - Click **New** to add filter parameters for data source objects properties. The **New SQL Filter Predicate** dialog appears.
   - Use the **Property** and **Operator** fields to supply the filter criteria. **Property** specifies if the value is a **Name** or **Schema**, and **Operator** specifies the matching type of the filter syntax. (**Equals**, **Not Equals**, **Like**, **Not Like**, **In**, **Not In**)
   - In the **Value** field, enter the full or partial syntax of the property or properties you want the template to filter in **data source Explorer**.
5. Click **OK**. The filter property specification is added to the Filter Template.
6. When you have finished defining the filter template, click **OK**. The template name is added to the **Properties** dialog. It can be enabled and disabled by selecting or de-selecting the check box beside its name, respectively.

Data source object filters are added and removed from the development environment by selecting and de-selecting the checkboxes associated with each filter template on both the global and data source-specific dialogs.

To define a specific filter

Data source-specific object filters affect only the specified data source.

To define data source specific filters:

1. In **Data Source Explorer**, right-click the data source and select **Properties**.

   The **Properties** dialog appears.
2. Select the **SQL Filter** node and deselect **Enable Data Source Specific Settings**. The other controls on the dialog become enabled.
3. Click **Add**. The **Filter Template** dialog appears.

4. Specify the parameters of the filter.
   - In the **Name** field, enter the name of the filter as you want it to appear in the selection window on the **SQL Filter** node.
   - The **Database Type** pane provides a list of data source objects. Deselect the data source objects that this template filters so that they do not appear in **Database Explorer** when displaying data source objects for the data source.
   - Click **New** to add filter parameters for data source object properties. The **New SQL Filter Predicate** dialog appears.
   - Use the **Property** and **Operator** fields to supply the filter criteria. **Property** specifies if the value is a **Name** or **Schema**, and **Operator** specifies the matching type of the filter syntax. (**Equals**, **Not Equals**, **Like**, **Not Like**, **In**, **Not In**)
   - In the **Value** field, enter the full or partial syntax of the property or properties you want to filter in **Data Source Explorer**.

   Click **OK**. The filter property specification is added to the Filter Template.
5. When you have finished defining the filter template, click **OK**. The template name is added to the **Properties** dialog. It can be enabled and disabled by selecting or deselecting the check box beside its name, respectively.

Comparing Data

Data comparison jobs enable you to compare the data repositories of two registered data sources, specify configuration adjustments and mapping refinement parameters, and synchronize any discrepancies discovered by the job, post-execution.

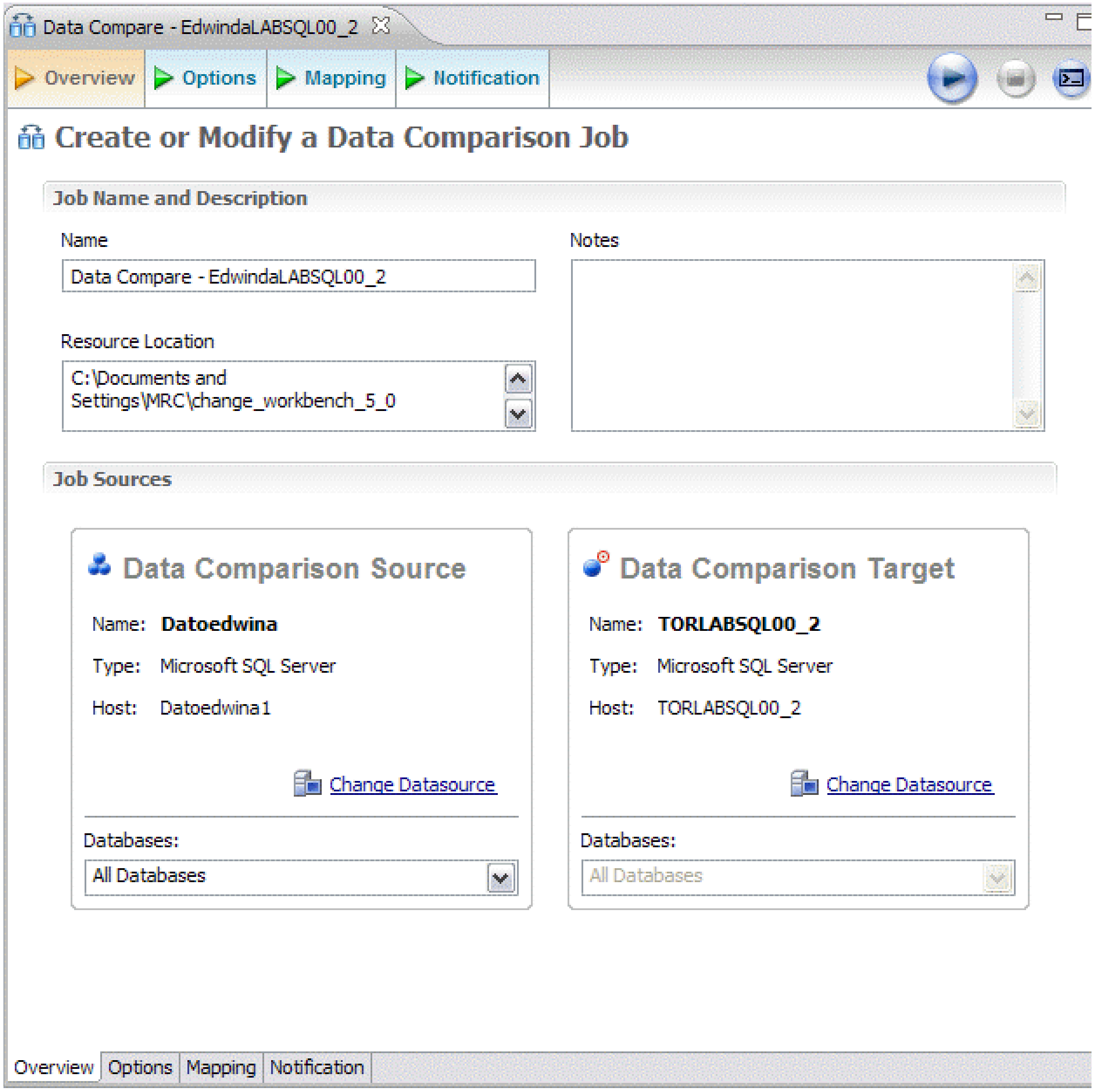
Comparison job results are viewed at row level per resident database, owner or schema, and mismatches can be updated automatically or missing rows added at the table level. The synchronization scripts used to update data sources are stored or executed immediately upon job completion.

The following tasks provide a high-level overview of the data comparison job process:

1. Create a New Data Comparison Job
2. Specify a Job Name
3. Specify the Source and Target Data Sources
4. Understand the Data Source Mapping Process
5. Customize the Data Source Mapping Process
6. Customize Database Mapping
7. Customize Table Mapping
8. Specify Execution, Comparison, and Resolution Options
9. Specify Notification Options
10. Execute the Data Comparison Job
11. Customize Data Resolution
12. Resolve Data Repositories by Generating a SQL Script
13. Executing a SQL Synchronization Script
14. Generate a Job Report

Create a New Data Comparison Job

Data comparison jobs are performed in the **Data Comparison Job Editor**.



Name: Data Compare - EdwindaLABSQL00_2

Resource Location: C:\Documents and Settings\MRC\change_workbench_5_0

Notes:

Data Comparison Source
Name: **Datoedwina**
Type: Microsoft SQL Server
Host: Datoedwina1
Databases: All Databases

Data Comparison Target
Name: **TORLABSQL00_2**
Type: Microsoft SQL Server
Host: TORLABSQL00_2
Databases: All Databases

To begin a new comparison job

- Select **File > New > Data Comparison Job**. The **Data Comparison Job Editor** opens.

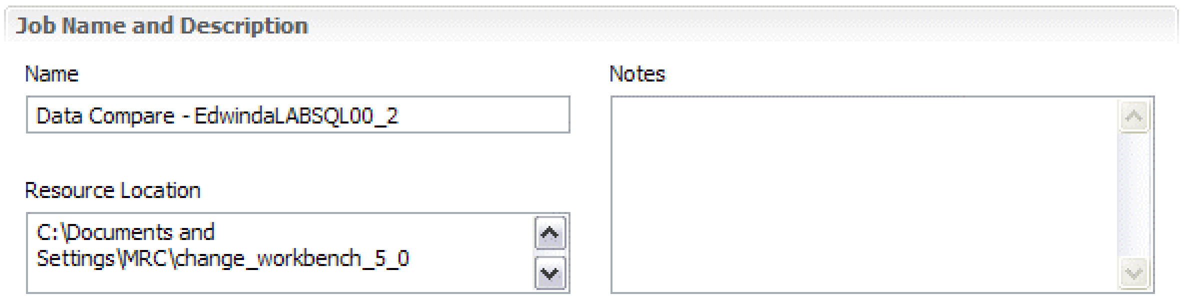
You can now proceed to set up the parameters of the new job.

NOTE: If any jobs already exist, they can be opened by selecting **File > Open Comparison** and choosing the name from the list of available jobs.

Specify a Job Name

A job name identifies the job in the application and should be specified with this in mind.

Specify a meaningful name that clearly identifies the job in the views and dialogs of the working environment.


Job Name and Description

Name: Data Compare - EdwindaLABSQL00_2

Resource Location: C:\Documents and Settings\MRC\change_workbench_5_0

Notes:

To name a job

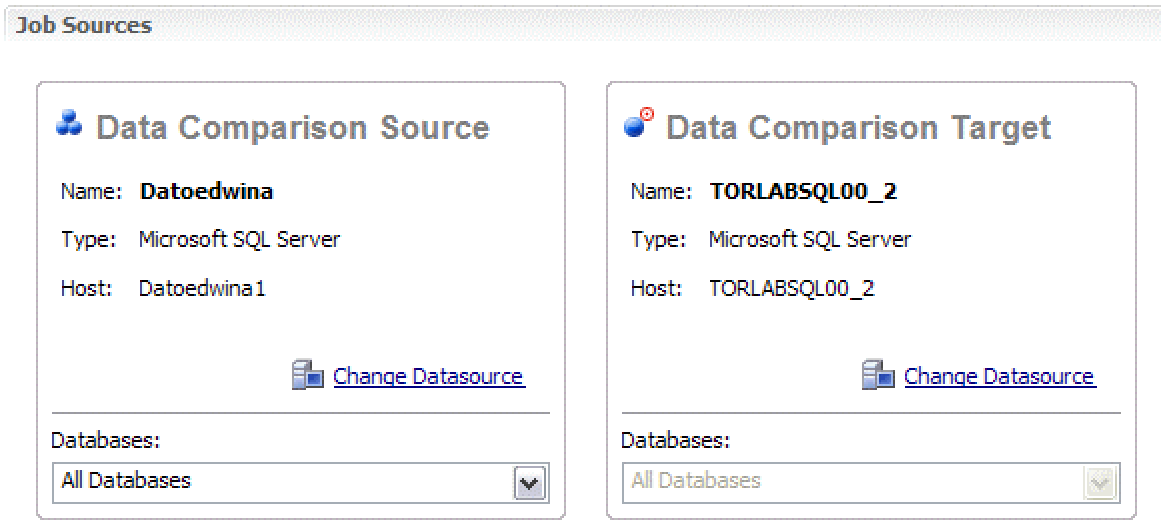
- Type the name of the job in the **Name** field of the **Data Comparison Job Editor**.

Ensure you specify a meaningful name that identifies the job in other views and dialogs. You can save the job by selecting **File > Save** or **File > Save All** from the Menu Bar, or highlight the Editor window and press **Ctrl+S**.

Specify the Source and Target Data Sources

The **Data Comparison Source** and **Data Comparison Target** boxes identify what data sources are compared when the job executes. Both boxes must contain the sources before the job can be executed. When a data source is added to a job, it is displayed by **Name**, **Type**, and **Host**.

Additionally, the drop down menus below the boxes enables you to map the data source at a high level by providing a list of the databases, schemas, or owners of the specified data source, depending on the DBMS.


Job Sources

Data Comparison Source
Name: Datoedwina
Type: Microsoft SQL Server
Host: Datoedwina1
Change Datasource
Databases: All Databases

Data Comparison Target
Name: TORLABSQL00_2
Type: Microsoft SQL Server
Host: TORLABSQL00_2
Change Datasource
Databases: All Databases

To add a data source to a job

- Click and drag a data source from **Data Source Explorer** to the **Comparison Source** or **Comparison Target** box, or click **Select Data Source** in the appropriate box and choose a data source from the dialog.

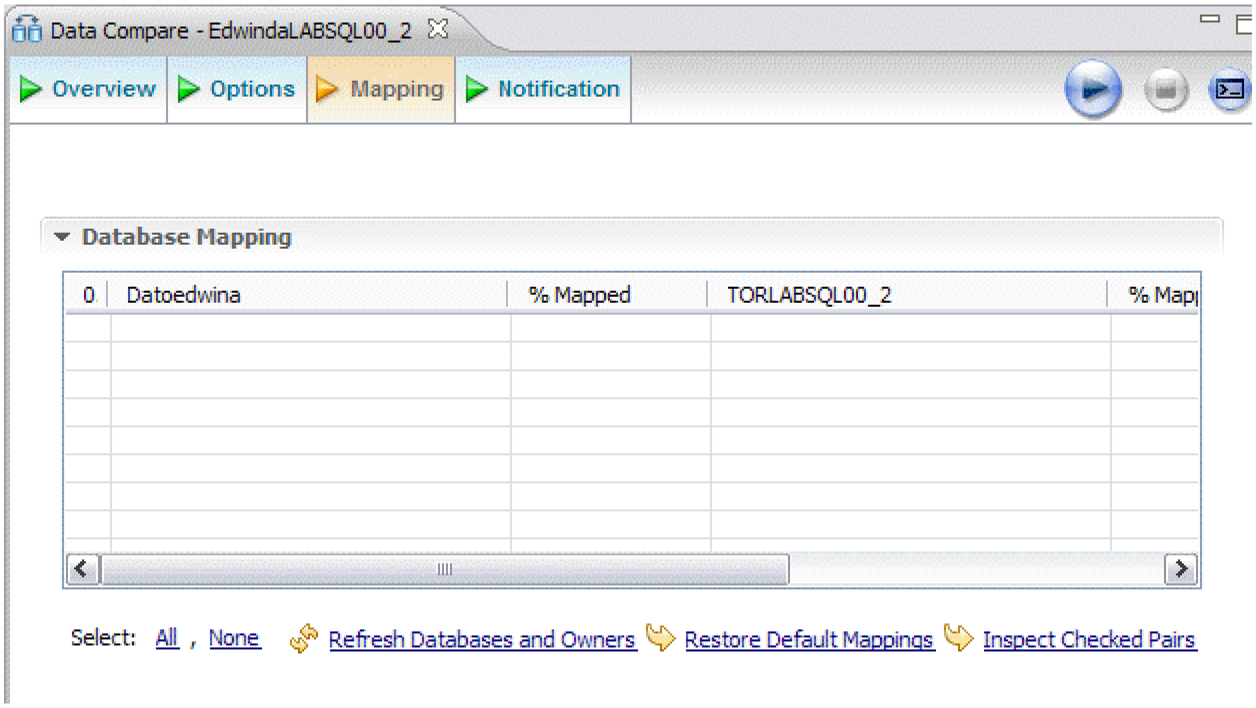
  NOTE: You can change a data source selection by clicking **Change Data Source** from the appropriate box, or by dragging a different data source to the box from **Data Source Explorer**.

Understand the Data Source Mapping Process

Mapping is the process of pairing data source elements between the source and target of a job. These elements (databases, tables, and row) are compared when the job executes to determine matches and mismatches between the two sources. Mapping details are viewed on the **Mapping** tab of the Editor.

The process automatically maps data source elements based on the interpretation of both schemas. Elements that do not match are excluded from the job when it executes, unless manually paired prior to running the job.

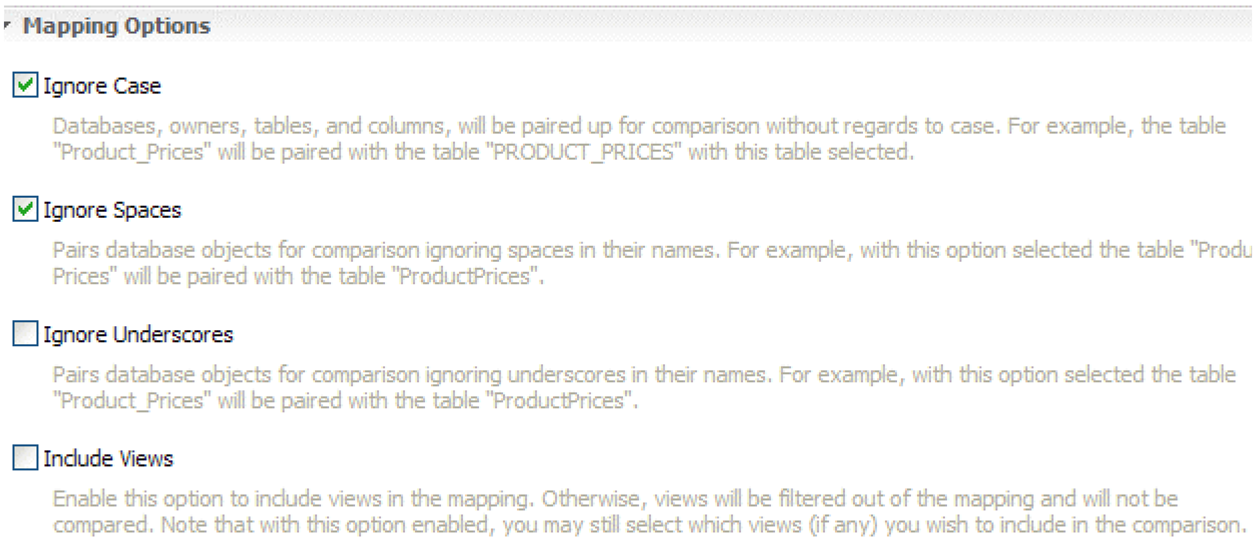
The **Database Mapping** table displays the paired databases or owners on the **Source** or **Target**. The **% Mapped** column indicates how much of the database or owner (a percentage value of total tables and rows) is mapped. Unpaired databases appear in the table as well, but are excluded from the job unless manually mapped to a corresponding database.



In order to view the table and view mappings of database pairs, you need to initiate the inspection process by clicking **Inspect Checked Pairs**. Once Change Manager has analyzed the selected pairs, you can view the table mappings of a specific pair, by clicking a row. The **Table Mapping** table populates with the paired tables and views of the selected pair. You can view the row mappings of a table or view pair by clicking **Edit** in the right column of the appropriate row. The row pairings and corresponding values appear in a new window. Individual owners and database pairs can be examined by selecting the appropriate icon in the **Inspect Now** column.

Customize the Data Source Mapping Process

You can manually configure the way Change Manager maps data source elements and then run the process again. The automatic mapping options are specified on the **Options** tab. You can re-run the mapping process by navigating back to the **Mapping** tab and clicking **Refresh Database and Owners** once this task is completed.



Mapping Options

[✓] Ignore Case

Databases, owners, tables, and columns, will be paired up for comparison without regards to case. For example, the table "Product_Prices" will be paired with the table "PRODUCT_PRICES" with this table selected.

[✓] Ignore Spaces

Pairs database objects for comparison ignoring spaces in their names. For example, with this option selected the table "Produ Prices" will be paired with the table "ProductPrices".

[ ] Ignore Underscores

Pairs database objects for comparison ignoring underscores in their names. For example, with this option selected the table "Product_Prices" will be paired with the table "ProductPrices".

[ ] Include Views

Enable this option to include views in the mapping. Otherwise, views will be filtered out of the mapping and will not be compared. Note that with this option enabled, you may still select which views (if any) you wish to include in the comparison.

To configure the automatic mapping process:

1 Navigate to the **Options** tab and select or deselect the check boxes in the **Mapping Options** section, as needed:

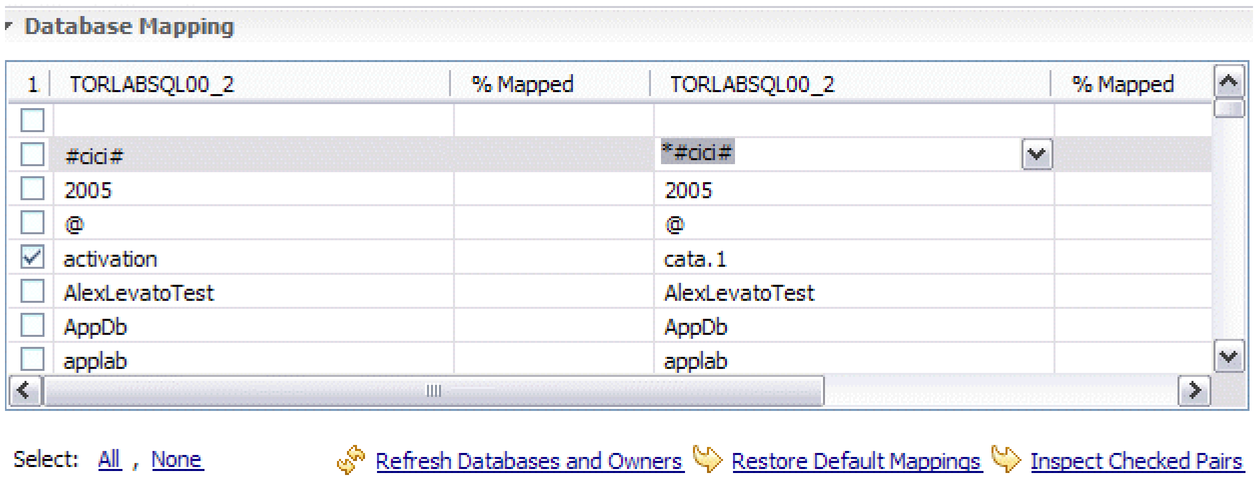
- **Ignore Case:** Pairs data source objects without regards to case. For example, the table “Product_Prices” is paired with the table “product_prices”.
- **Ignore Spaces:** Pairs data source objects and ignores extraneous spaces in object names. For example, the table “Product Prices” is paired with the table “ProductPrices”.
- **Ignore Underscores:** Pairs data source objects and ignores underscores in job names. For example, the table “Product_Prices” is paired with the table “Product Prices”.
- **Include Views:** Includes database views when performing the automatic mapping process. If this option is deselected (the default setting), views are filtered out and will not be compared. If this option is selected, you can still manually choose the views to include or exclude from the comparison.

2 Navigate back to the **Mapping** tab and click **Refresh Databases and Owners** to update the automatic comparison.

Customize Database Mapping

You can customize job database mapping parameters manually by choosing to exclude certain paired rows or redefining objects after the automatic mapping process is completed.

The check boxes beside each paired object displayed in the Database Mapping Source column indicate if a pair is included or excluded when the job runs.



1	TORLABSQL00_2	% Mapped	TORLABSQL00_2	% Mapped
☐				
☐	#cici#		*#cici#	
☐	2005		2005	
☐	@		@	
✓	activation		cata.1	
☐	AlexLevatoTest		AlexLevatoTest	
☐	AppDb		AppDb	
☐	applab		applab	

Database pairings can also be redefined via the drop down menu that appears when the corresponding item in the **Target** data source column is selected. The menu contains a list of all databases or owners that reside on the **Target** data source. Databases or owners that are already paired appear in the drop down menu with an asterisk to denote them. Additionally, using the same method, unpaired databases can be assigned a matching database on the **Source**.

On Microsoft SQL Server and Sybase platforms, you can pair up owners independently of databases by right-clicking on a row and selecting **Show Database Parts**, **Show Source Database Parts**, or **Show Target Database Parts** to decompose the database into its owner components. These components can individually be mapped and explored. To recompose the database, right-click and select **Hide Database Parts**.

To restore the Database Mapping table to its original settings, click **Restore Default Mappings**. All settings revert to the original object mappings as determined by the job, based on the selected Mapping options.

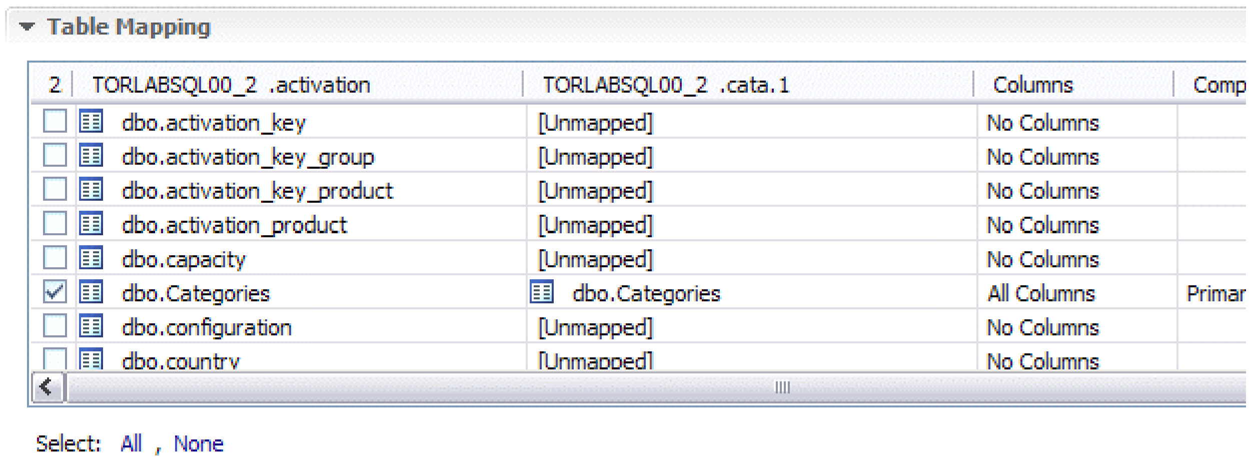
To include or exclude a database pair from the job

1. Select or deselect the check box beside each database pair, as needed. Use the **All** and **None** commands at the bottom of the table to select all or none of the rows in the table, respectively.
2. Click **Inspect Now** or **Inspect Checked Pairs** to refresh pairs if you want to examine them at the table level.

Customize Table Mapping

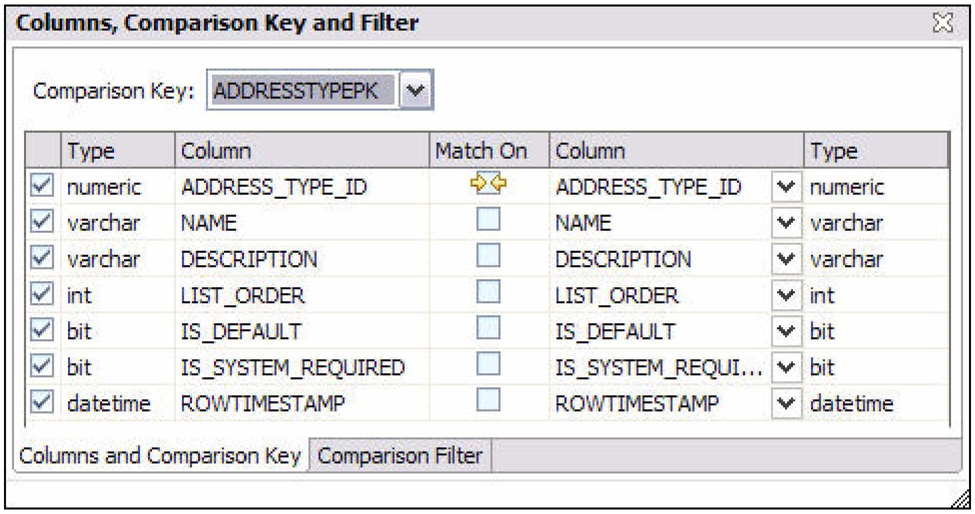
You can manually map databases, tables, and rows between the **Source** and **Target** data sources. Tables are displayed by name and contain mapping refinement column values (**Columns**, **Comparison Key**, and **Filter**.)

You can manually map unpaired tables between the databases by clicking the corresponding item in the **Target** data source column and define it via the drop down menu that appears, as with database mapping.



2	TORLABSQL00_2 .activation	TORLABSQL00_2 .cata.1	Columns	Comp
☐	dbo.activation_key	[Unmapped]	No Columns	
☐	dbo.activation_key_group	[Unmapped]	No Columns	
☐	dbo.activation_key_product	[Unmapped]	No Columns	
☐	dbo.activation_product	[Unmapped]	No Columns	
☐	dbo.capacity	[Unmapped]	No Columns	
✓	dbo.Categories	dbo.Categories	All Columns	Primar
☐	dbo.configuration	[Unmapped]	No Columns	
☐	dbo.country	[Unmapped]	No Columns	

Additionally, if you click **Edit** in the appropriate pairing, the **Columns**, **Comparison Key**, and **Filter** dialog enables you to further detail mapping refinements in terms of defining table roles in the execution of the job.

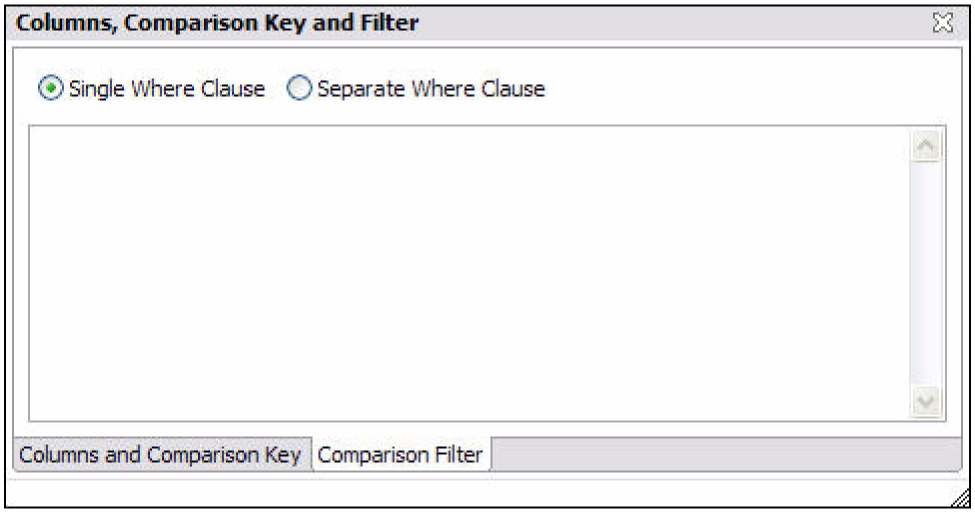

Columns, Comparison Key and Filter

Comparison Key: ADDRESSTYPEPK

	Type	Column	Match On	Column	Type
✓	numeric	ADDRESS_TYPE_ID	⇔	ADDRESS_TYPE_ID	numeric
✓	varchar	NAME	☐	NAME	varchar
✓	varchar	DESCRIPTION	☐	DESCRIPTION	varchar
✓	int	LIST_ORDER	☐	LIST_ORDER	int
✓	bit	IS_DEFAULT	☐	IS_DEFAULT	bit
✓	bit	IS_SYSTEM_REQUIRED	☐	IS_SYSTEM_REQUI...	bit
✓	datetime	ROWTIMESTAMP	☐	ROWTIMESTAMP	datetime

Columns and Comparison Key
Comparison Filter

- The **Columns and Comparison Key** enables the mapping of columns to compare between the paired tables. Select or deselect the columns, as needed, in the dialog provided, and use the drop down menu to change the comparison key if multiple keys are available.
- The **Comparison Filter** tab enables you to manually insert a WHERE clause for execution when the job runs. Click the ellipsis button and enter the WHERE clause into the text box provided. If you want to create a different WHERE clause per table in the pair, select the **Separate WHERE Clause** radio button. The dialog adds an additional text box for a second WHERE clause.

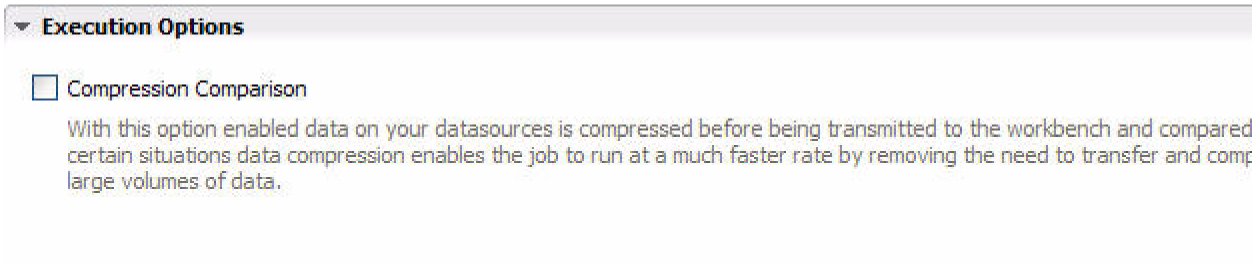

Columns, Comparison Key and Filter
Single Where Clause
Separate Where Clause
Columns and Comparison Key
Comparison Filter

NOTE: When entering selective WHERE clauses, **do not include the WHERE keyword** (e.g. use “AGE>12” instead of “WHERE AGE>12”.)

Specify Execution, Comparison, and Resolution Options

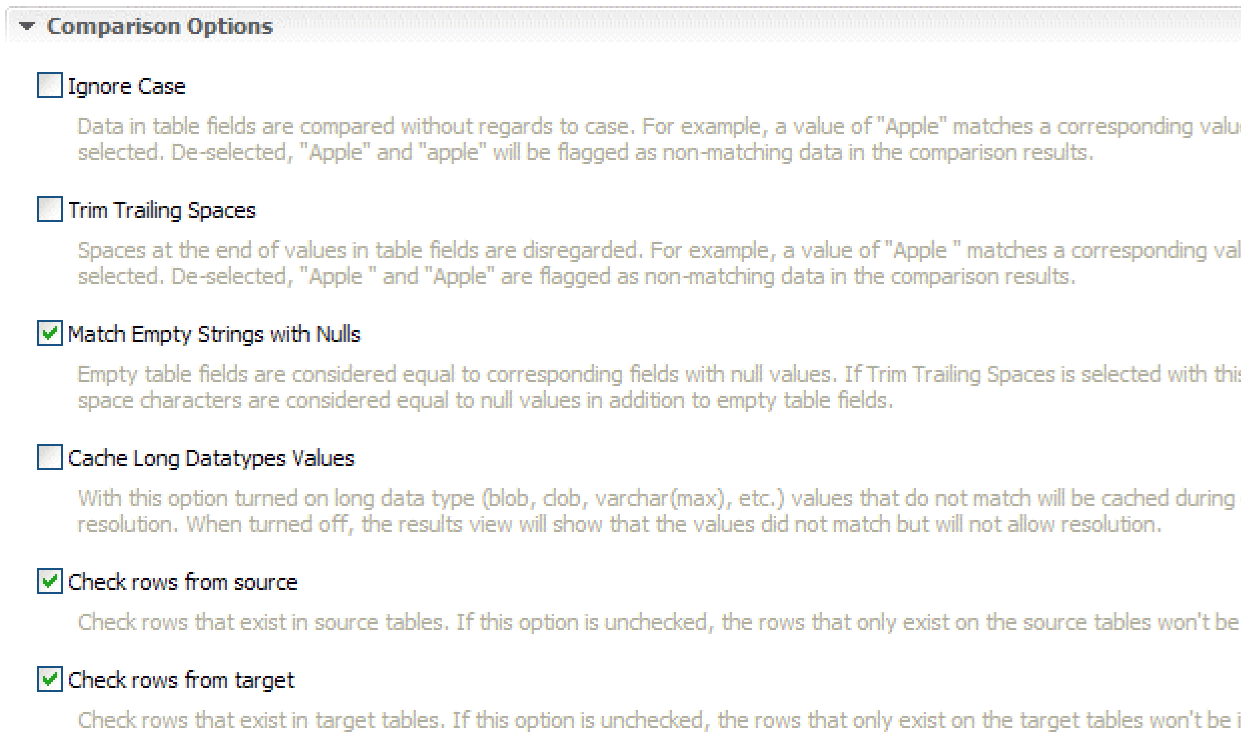
Prior to the execution of a job, parameters can be set to control aspects of the execution process itself. These options can be set prior to job execution on the **Options** tab.

The **Execution Options** section contains the **Compression Comparison** setting. This enables a faster version of the execution process where the job compresses row data for faster transmission and comparison. This features allows for a faster comparison process in general and does not affect job accuracy. This option is especially useful over slow networks with large volumes of data in each row, and with faster database servers.



With this option enabled data on your datasources is compressed before being transmitted to the workbench and compared certain situations data compression enables the job to run at a much faster rate by removing the need to transfer and comp large volumes of data.

The **Comparison Options** section specifies the parameters of how the job treats comparison in terms of naming conventions.



Comparison Options

☐ Ignore Case

Data in table fields are compared without regards to case. For example, a value of "Apple" matches a corresponding valu
selected. De-selected, "Apple" and "apple" will be flagged as non-matching data in the comparison results.

☐ Trim Trailing Spaces

Spaces at the end of values in table fields are disregarded. For example, a value of "Apple " matches a corresponding val
selected. De-selected, "Apple " and "Apple" are flagged as non-matching data in the comparison results.

☑ Match Empty Strings with Nulls

Empty table fields are considered equal to corresponding fields with null values. If Trim Trailing Spaces is selected with thi
space characters are considered equal to null values in addition to empty table fields.

☐ Cache Long Datatypes Values

With this option turned on long data type (blob, clob, varchar(max), etc.) values that do not match will be cached during
resolution. When turned off, the results view will show that the values did not match but will not allow resolution.

☑ Check rows from source

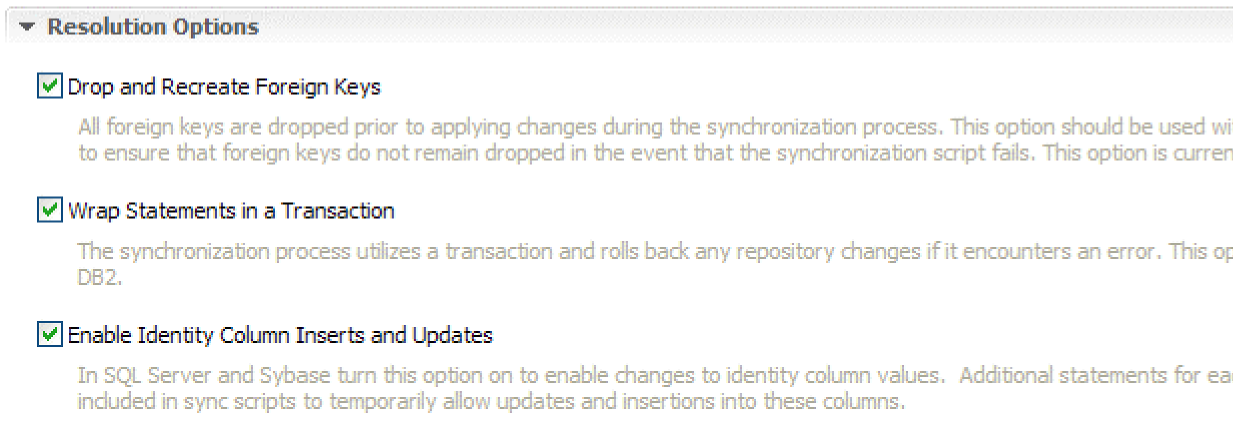
Check rows that exist in source tables. If this option is unchecked, the rows that only exist on the source tables won't be

☑ Check rows from target

Check rows that exist in target tables. If this option is unchecked, the rows that only exist on the target tables won't be i

- **Ignore Case** indicates the job ignores case when comparing data. For example, “Apple” matches a corresponding value of “apple”.
- **Trim Trailing Spaces** indicates the job comparison ignores extra spaces at the end of values when comparing data. For example, “Apple “ matches a corresponding value of “Apple”.
- **Match Empty Strings with Nulls** indicates that the job comparison considers empty values to be the same as null values for matching purposes.
- **Cache Long Datatype Values** indicates that long data types (blob, clob, varchar(max), etc.) values that do not match are cached during comparison and available for resolution during that stage of the process. When this value is disabled, the results of the job indicate that the values did not match but resolution is unavailable. Individual long datatype values can be retrieved when exploring results regardless of the selection of this property.
- **Check Rows from Source** checks the rows that exist in the source tables. If the option is not selected, the rows that exist on the source tables will not be included in the comparison.
- **Check Rows from Target** checks the rows that exist in the target tables. If the option is not selected, the rows that exist on the target tables will not be included in the comparison.

The **Resolution Options** section determines how the job resolves any discrepancies discovered during the comparison process.



Resolution Options

✓ Drop and Recreate Foreign Keys

All foreign keys are dropped prior to applying changes during the synchronization process. This option should be used wi
to ensure that foreign keys do not remain dropped in the event that the synchronization script fails. This option is curren

✓ Wrap Statements in a Transaction

The synchronization process utilizes a transaction and rolls back any repository changes if it encounters an error. This op
DB2.

✓ Enable Identity Column Inserts and Updates

In SQL Server and Sybase turn this option on to enable changes to identity column values. Additional statements for ea
included in sync scripts to temporarily allow updates and insertions into these columns.

- **Drop and Recreate Foreign Keys** indicates that all foreign keys are dropped prior to applying changes during the resolution process. This option should be used with **Wrap Statements in Transaction** to ensure that foreign keys do not remain dropped in the event that the synchronization script used to resolve it fails.

  NOTE: This option is only available for SQL Server scripts.

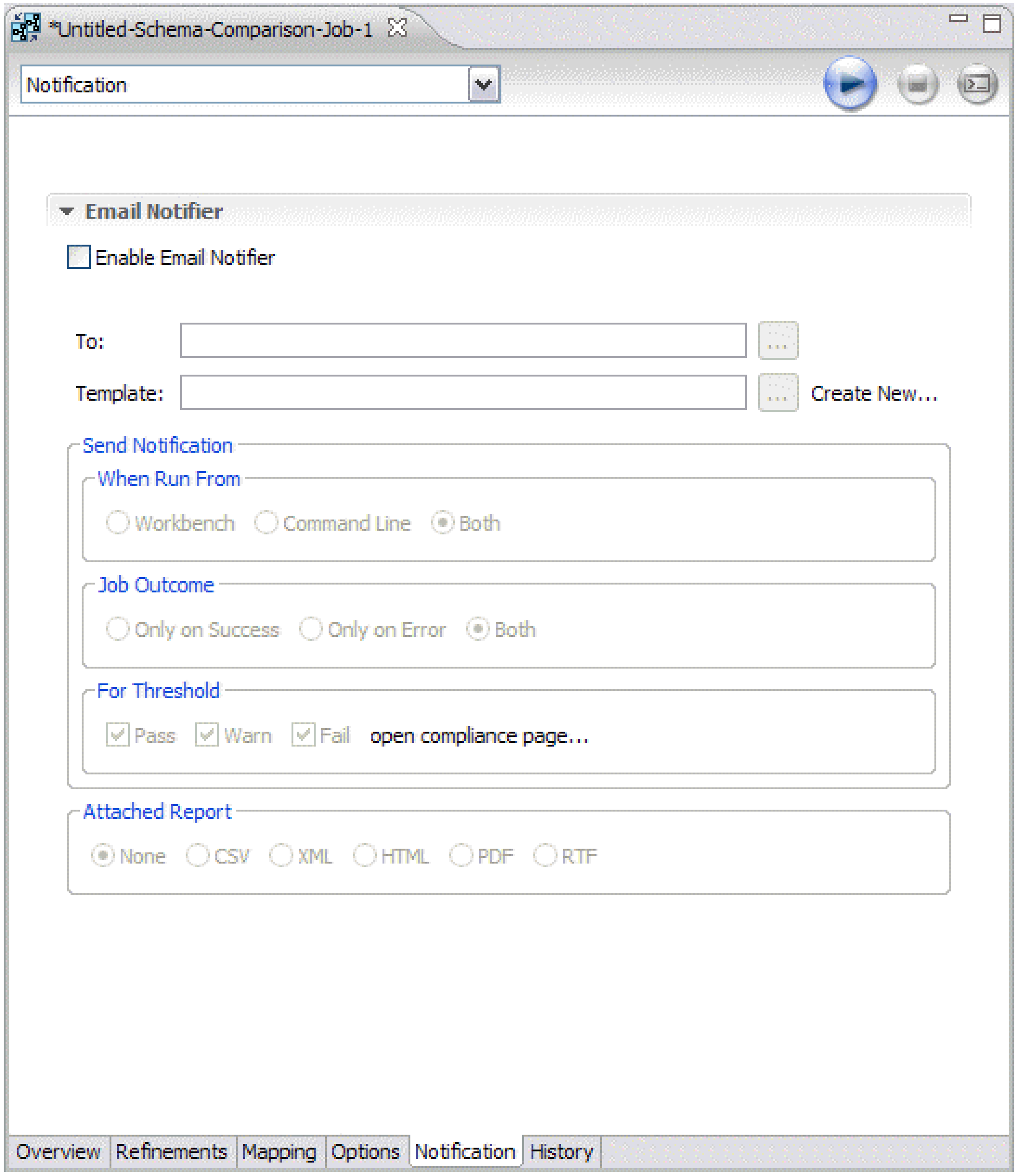
- **Wrap Statements in a Transaction** utilizes a transaction and rolls back any repository changes if it encounters an error.

  NOTE: This option is not supported for DB2 for LUW scripts.

- **Enable Identity Column Inserts and Updates** enables changes to identity column values on SQL Server and Sybase platforms. The update and insert statements are automatically included in the synchronization script when this option is selected.

Specify Notification Options

Notification options provide parameters for setting up email notification and report mailing for executed data comparison jobs. The notification options tab enables you to specify the email addresses to which you want to send notifications, the email message format, and the conditions under which a notification should be sent.



To enable notification, select the **Enable Email Notifier** check box and select the options, as available. Enter the email addresses to which the notification will be sent in the **To** field, and specify an email format template in the **Template Field**.

In the **Send Notification** options, specify when the email notification should be sent:

- Under the **When Run From** options, indicate that email notifications should be sent when the job is executed from the **Workbench**, **Command Line**, or **Both**.
- Under the **Job Outcome** options**,** indicate that email notification should be sent **Only on Success**, **Only on Error**, or **Both**.

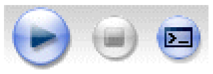
- Under the **For Threshold** options, specify the threshold level for which the email notification will be sent, based on the job results. Choose **Pass**, **Warn**, or **Fail**. You can click **Open Compliance Page** to specify the percentage values for each threshold warning, which are based on the percent match results of the job.

You can also specify the file format of an attached report that displays the job results with the notification email. Choose **None** (does not attach a report with the notification email), **CSV**, **XML**, **HTML**, **PDF**, or **RTF** to specify the file format of the report.

NOTE: Notification options can be modified via a plug-in extension that enables custom notification coding. See Creating and Implementing a Custom Configuration Plug-In for more information.

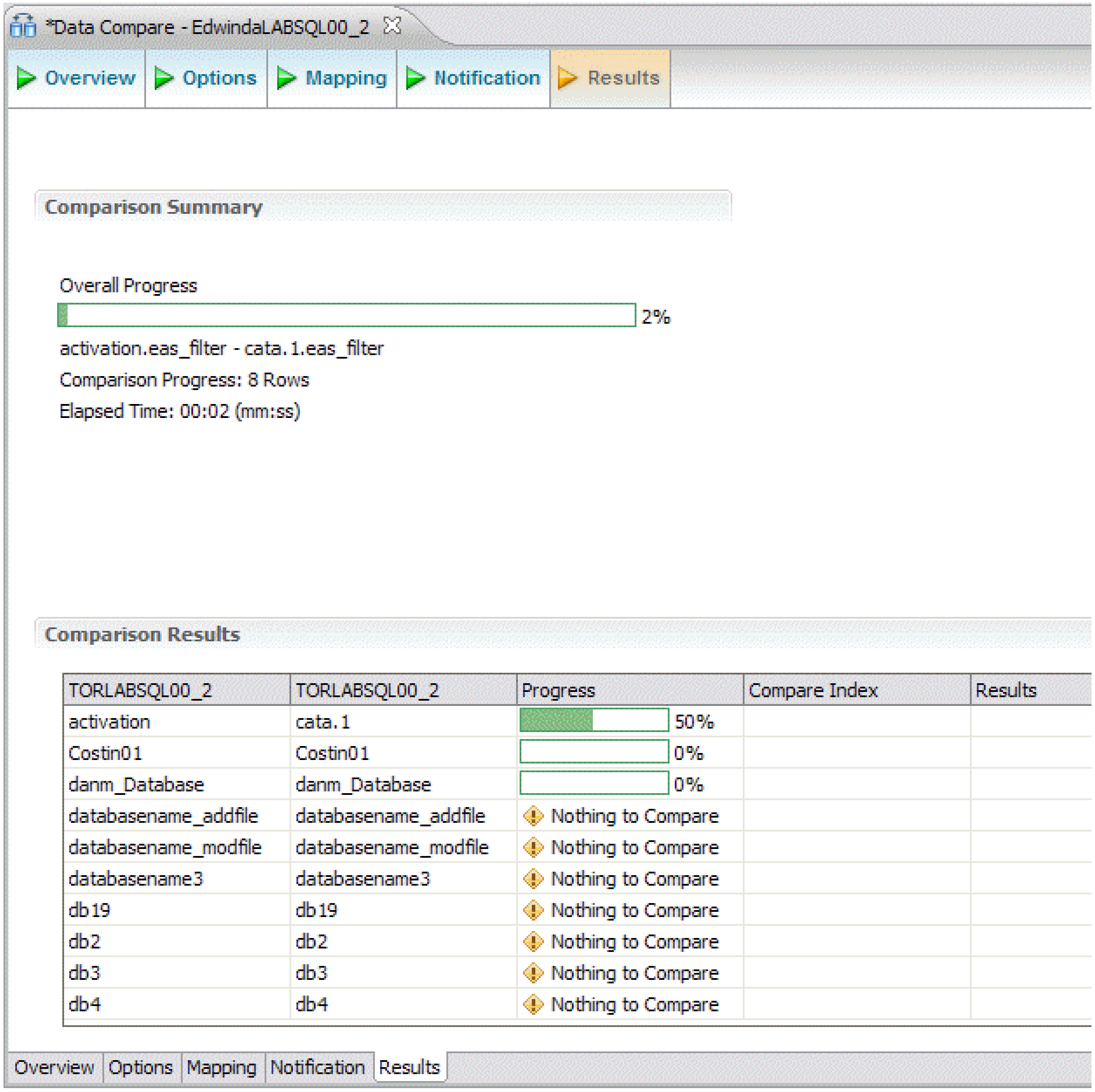
Execute the Data Comparison Job

The comparison job runs when you click the **Run** icon located in the upper right-hand corner of the Data Comparison Job Editor.



When the process runs, the **Results** tab appears and displays the progress of the job via the **Overall Progress** bar at the top of the tab.

In the **Comparison Results** table, the database pairs are displayed along with **Progress**, **Compare Index**, **Results**, and **Resolution** columns.



Comparison Summary

Overall Progress 2%

activation.eas_filter - cata.1.eas_filter

Comparison Progress: 8 Rows

Elapsed Time: 00:02 (mm:ss)

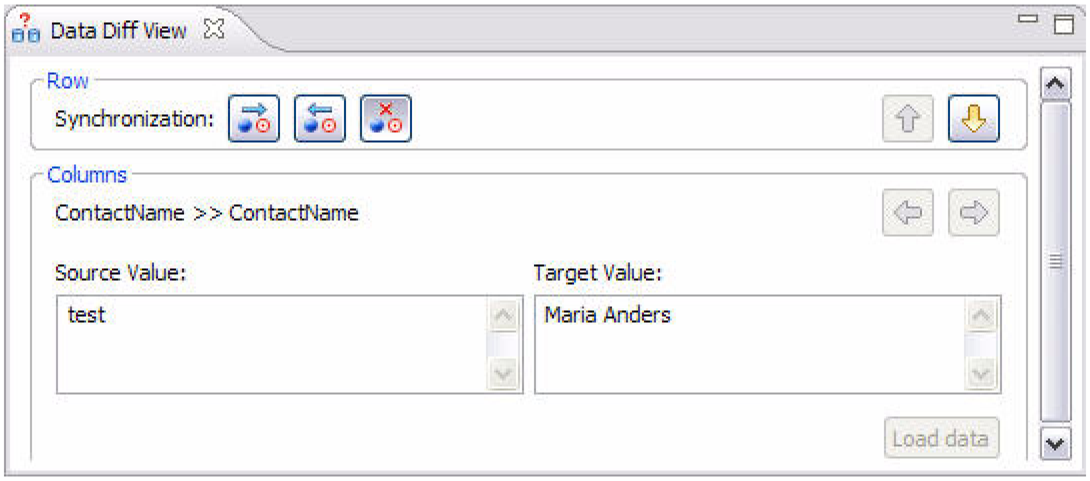
Comparison Results

TORLABSQL00_2	TORLABSQL00_2	Progress	Compare Index	Results
activation	cata.1	50%		
Costin01	Costin01	0%		
danm_Database	danm_Database	0%		
databasename_addfile	databasename_addfile	Nothing to Compare		
databasename_modfile	databasename_modfile	Nothing to Compare		
databasename3	databasename3	Nothing to Compare		
db19	db19	Nothing to Compare		
db2	db2	Nothing to Compare		
db3	db3	Nothing to Compare		
db4	db4	Nothing to Compare		

- The **Progress** column indicates, per database pair, the progress of the job. When the job is finished comparing the database row, a result of **Done** (should the comparison be successful), **Failed** (should the comparison be unsuccessful), or **Nothing to Compare** (if the pairs were not specified) will display in the column.
- The **Compare Index** column indicates the percentage value of data per database that the job matched. Failed rows and rows in which the progress is not completed do not display results.

Customize Data Resolution

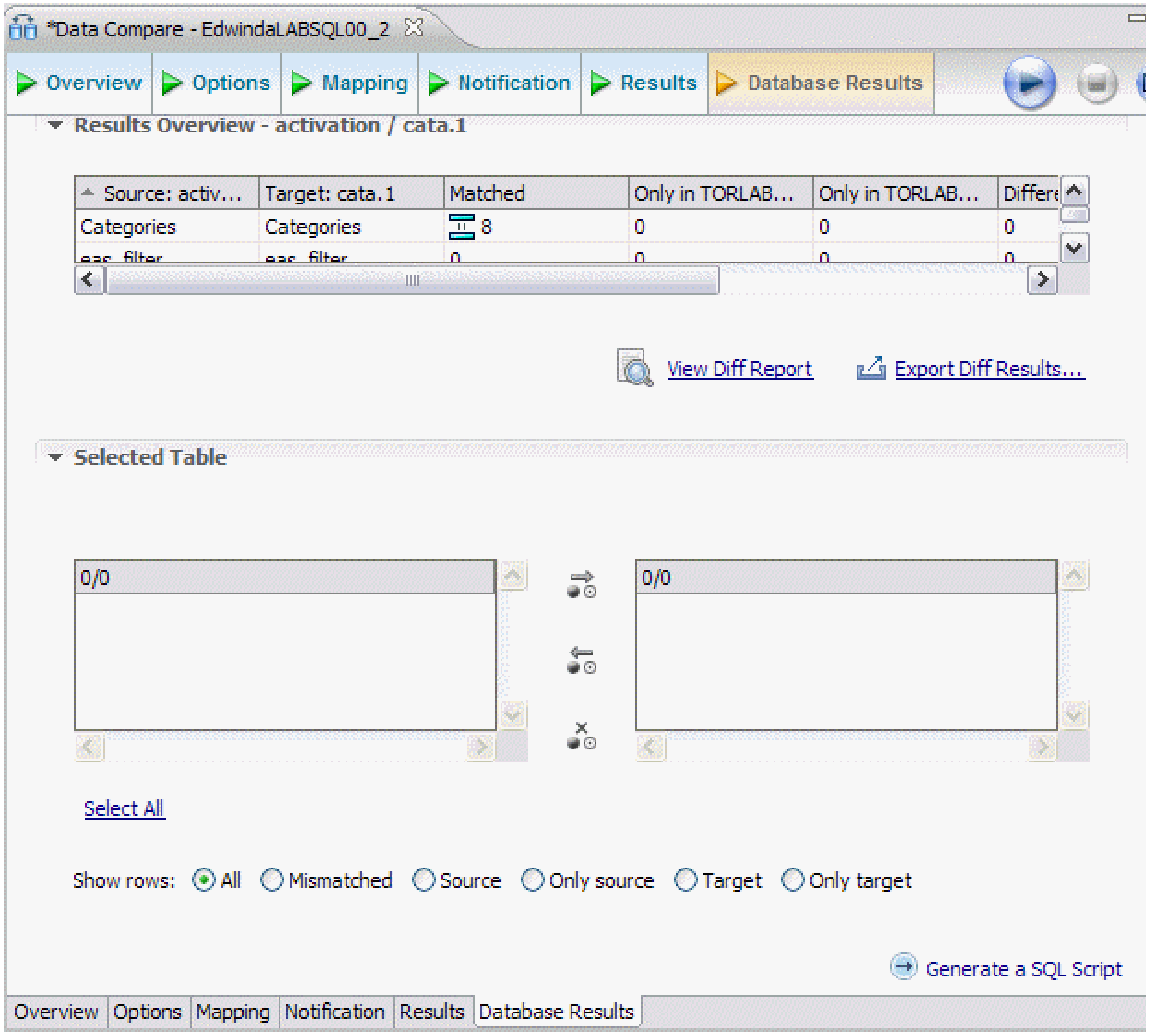
You can specify how data is resolved in the **Data Diff View** by selecting one of the appropriate icons beside the **Synchronization** prompt. By row, you can choose to **Make Target Match Source for Selected Row**, **Make Source Match Target for Selected Row**, or **Do Not Resolve Selected Row**, respectively.



Resolve Data Repositories by Generating a SQL Script

Once a data comparison job has completed, synchronization commands are used to even the data in the repositories so that they match. The synchronization process automates generation of UPDATE, INSERT, and DELETE statements on both databases as required at the global resolution level.

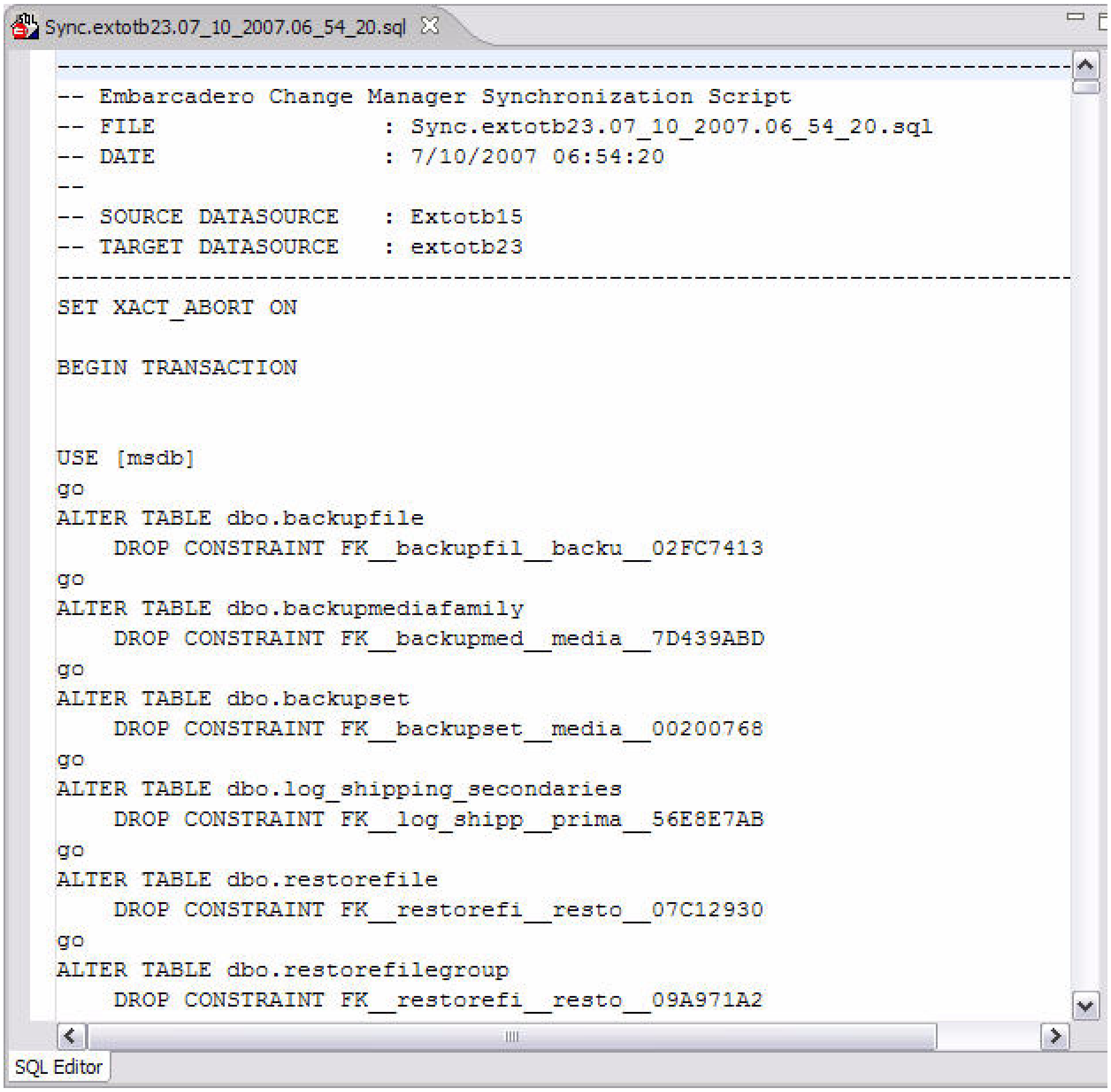
On the Results tab, click **View** in the row you want to resolve. The **Database Results** tab appears.


*Data Compare - EdwindaLABSQL00_2
Overview
Options
Mapping
Notification
Results
Database Results
Results Overview - activation / cata.1

Source: activ...	Target: cata.1	Matched	Only in TORLAB...	Only in TORLAB...	Differe
Categories	Categories	8	0	0	0
eas_filter	eas_filter	0	0	0	0

Show rows: [✓ All] Mismatched Source Only source Target Only target

Specify the type of **Resolution** by clicking the **All**, **Mismatched**, **Source**, **Only Source**, **Target**, or **Only Target** radio buttons, respectively, and then click **Generate a SQL Script**. The script appears in **SQL Editor**, and can be modified manually, as needed.



```
------------------------------------------------------------------------
-- Embarcadero Change Manager Synchronization Script
-- FILE                  : Sync.extotb23.07_10_2007.06_54_20.sql
-- DATE                  : 7/10/2007 06:54:20
--
-- SOURCE DATASOURCE    : Extotb15
-- TARGET DATASOURCE    : extotb23
------------------------------------------------------------------------
SET XACT_ABORT ON

BEGIN TRANSACTION


USE [msdb]
go
ALTER TABLE dbo.backupfile
    DROP CONSTRAINT FK__backupfil__backu__02FC7413
go
ALTER TABLE dbo.backupmediafamily
    DROP CONSTRAINT FK__backupmed__media__7D439ABD
go
ALTER TABLE dbo.backupset
    DROP CONSTRAINT FK__backupset__media__00200768
go
ALTER TABLE dbo.log_shipping_secondaries
    DROP CONSTRAINT FK__log_shipp__prima__56E8E7AB
go
ALTER TABLE dbo.restorefile
    DROP CONSTRAINT FK__restorefi__resto__07C12930
go
ALTER TABLE dbo.restorefilegroup
    DROP CONSTRAINT FK__restorefi__resto__09A971A2
```

The saved script can be executed immediately, or saved and run at any time to synchronize both databases based on the results of the current data comparison job.

Executing a SQL Synchronization Script

Files can be launched from within the Change Manager development environment for execution on a registered data source Files are executed via the commands in the **Run** menu.

When a SQL file is open in the Workspace, select it and choose a database and an associated catalog on which you want to execute file using the drop down menus in the Toolbar. You can click the execute icon to execute the code on the specified database and catalog, start a transaction or commit a transaction, or modify the SQL session options prior to execution.

NOTE: The synchronization script generation process does not support all object parameters. The following object parameters should be manually added after a synchronization script is generated, as required: **fillfactor**, **consumers**, **allow_dup_row**, and **statistics**.

To execute code

- Open the SQL file you want to run, ensure it is associated with the correct database and click the **Execute** icon. Change Manager executes the code on the data source you specified. Results are displayed in the same tab or in a new tab.

To execute a transaction

- Open the transaction file you want to run and ensure it is associated with the correct database, and then click the **Start Transaction** icon. Change Manager executes the transaction on the data source you specified.

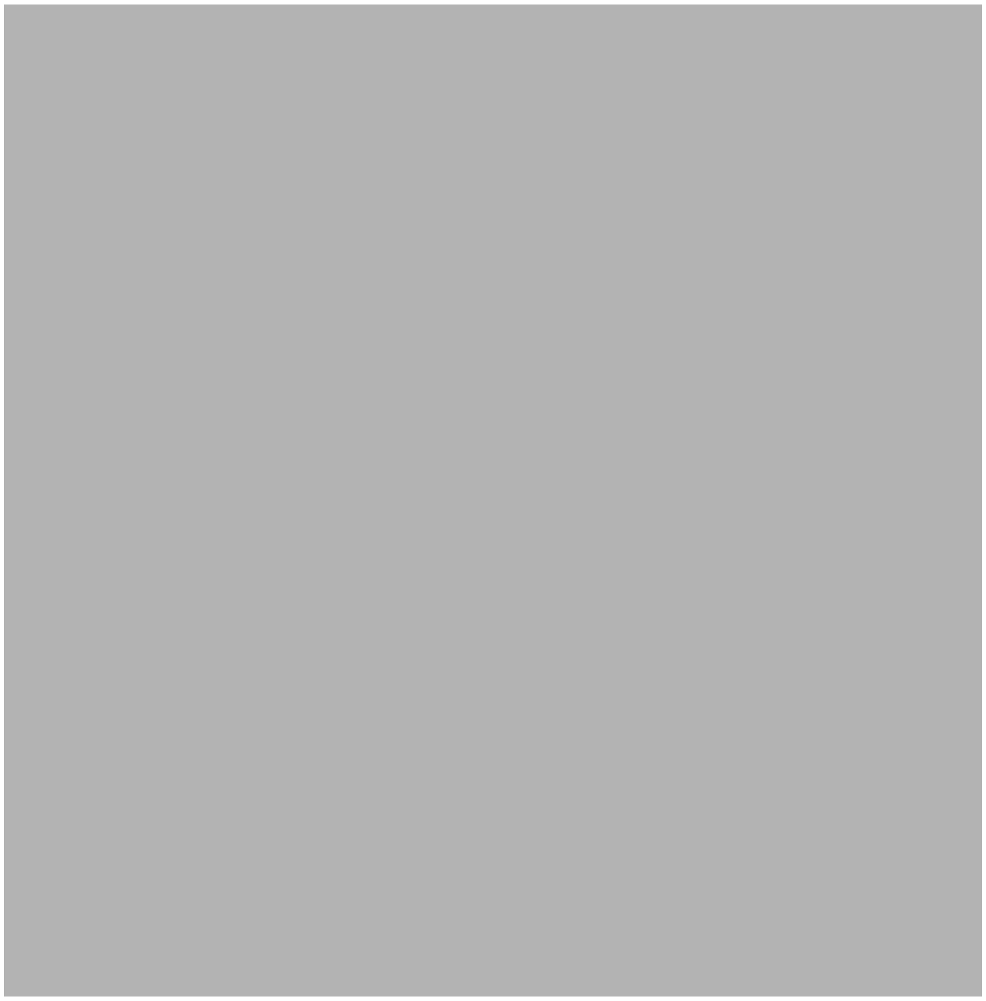
To commit a transaction

- Open the transaction file you want to commit, ensure it is associated with the correct database, and then click the **Commit Transaction** icon. Change Manager commits the transaction on the data source your specified.

You can set transactions to auto-commit prior to execution on the **SQL Execution** node of the **Preferences** panel in Change Manager.

Configure a SQL Session

The SQL Session Options dialog provides configuration parameters that indicate to Change Manager how to execute code in the development environment.



To modify SQL session options:

1 Click the SQL Session Options icon in the Toolbar.

The **SQL Session Options** dialog appears.

2 Click on individual parameters in the **Value** column to change the configuration of each property, as specified.

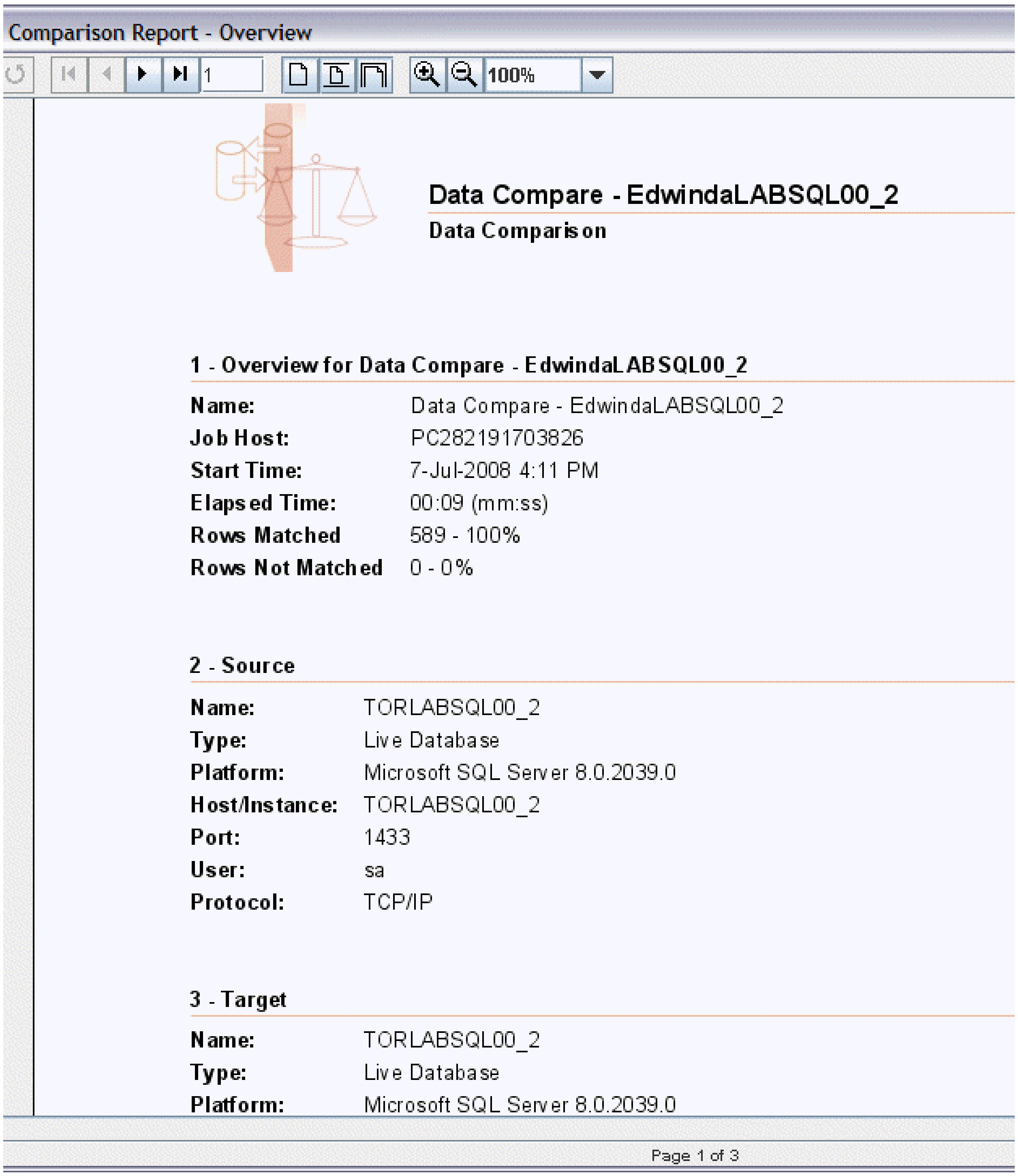
3 Click **Finish**.

The session options will be changed and Change Manager will execute the code as specified when you launch the script.

NOTE: Session options only apply to the corresponding editor and are not retained when executing multiple SQL files.

Generate a Job Report

Once you have executed a comparison job, you can generate a job report via the **View Report** command on the **Results** tab.



Data Compare - EdwindaLABSQL00_2

Data Comparison

1 - Overview for Data Compare - EdwindaLABSQL00_2

Name:	Data Compare - EdwindaLABSQL00_2
Job Host:	PC282191703826
Start Time:	7-Jul-2008 4:11 PM
Elapsed Time:	00:09 (mm:ss)
Rows Matched	589 - 100%
Rows Not Matched	0 - 0%

2 - Source

Name:	TORLABSQL00_2
Type:	Live Database
Platform:	Microsoft SQL Server 8.0.2039.0
Host/Instance:	TORLABSQL00_2
Port:	1433
User:	sa
Protocol:	TCP/IP

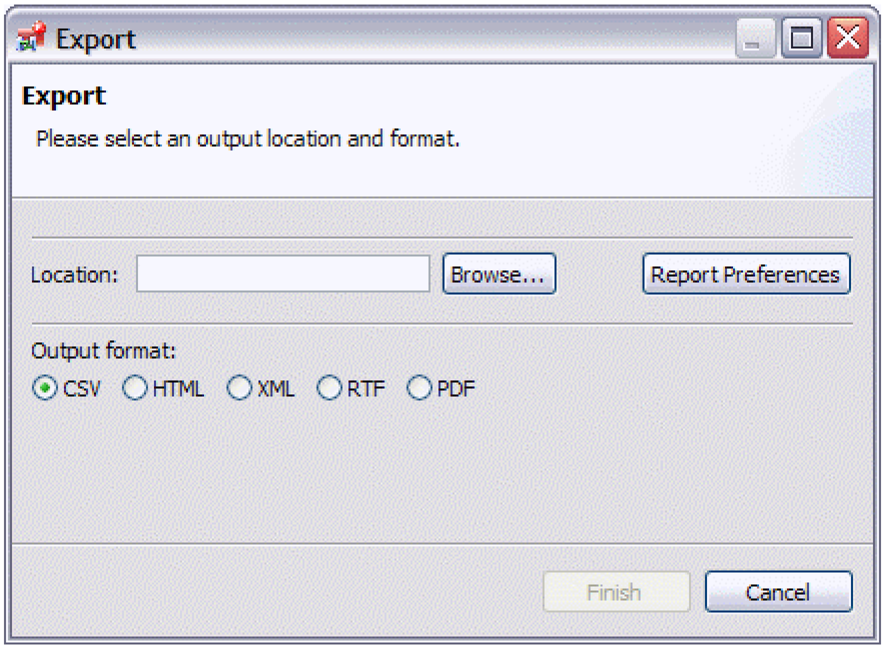
3 - Target

Name:	TORLABSQL00_2
Type:	Live Database
Platform:	Microsoft SQL Server 8.0.2039.0

Page 1 of 3

The report opens in a separate window and can be read on screen, sent to a printer, and saved to disk.

Additionally, the **Export Results** command will produce a report file in a specified format at a directory location of your choosing.



Click **Export Results** on the **Results** tab and fill in the fields, as required. When you click **Finish**, the report is created in the directory specified in the **Location** field, in the format specified in the **Output Format** section.

Comparing Configurations

Configuration comparison jobs enable you to compare the configuration settings between two registered data sources. They are used to ensure consistency between the settings of two data sources in order to meet guidelines regarding the storage and retention of data.

Comparison job results are viewed as an overall “percentage matched” between two data sources. Comparison jobs also provides a list of individual configuration properties and other, more granular details regarding both data source’s configuration settings. Additionally, the job determines the total properties matched, total properties that do not match, the properties available on only one data source, and the overall total properties compared in the job.

The following tasks provide a high-level overview of the configuration comparison job process:

- Create a New Configuration Comparison Job
- Specify a Comparison Job Name
- Specify Source and Target Data Sources
- Refine Property Comparisons
- Execute Comparison Job
- View Results
- Generating a Job Report

In addition to comparing two registered data sources, you can create a Standard -- a configuration comparison object that contains a customized list of configuration parameters, values, and comparison operators, or an Archive -- the older, stored configuration properties of an existing registered data source. Archives and Standards can then be used in comparison jobs in the same manner as registered data sources.

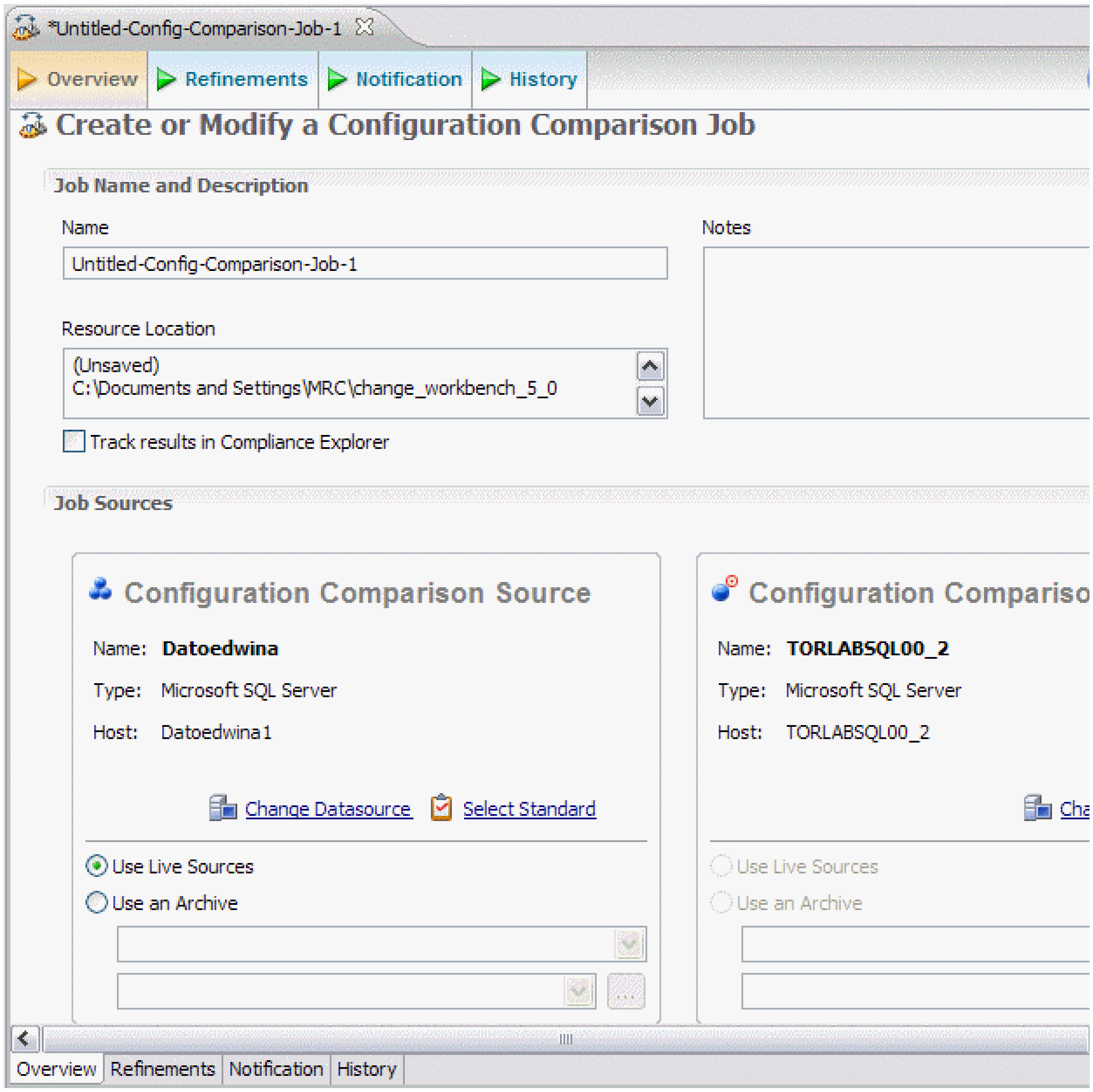
- Creating a Configuration Standard
- Creating a Configuration Archive

Finally, all comparison jobs, archives, and standards provide version information with regards to each data source platform. Each version parameter and description is available via the link below.

- Understanding Version Information

Create a New Configuration Comparison Job

Configuration comparison jobs are defined in the **Configuration Comparison Job Editor**.



Name: Untitled-Config-Comparison-Job-1

Notes

Resource Location: (Unsaved) C:\Documents and Settings\MRC\change_workbench_5_0

[ ] Track results in Compliance Explorer

Name: **Datoedwina**
Type: Microsoft SQL Server
Host: Datoedwina1

[✓ Use Live Sources]
Use an Archive

Name: **TORLABSQL00_2**
Type: Microsoft SQL Server
Host: TORLABSQL00_2

Use Live Sources
Use an Archive

To start a new comparison job

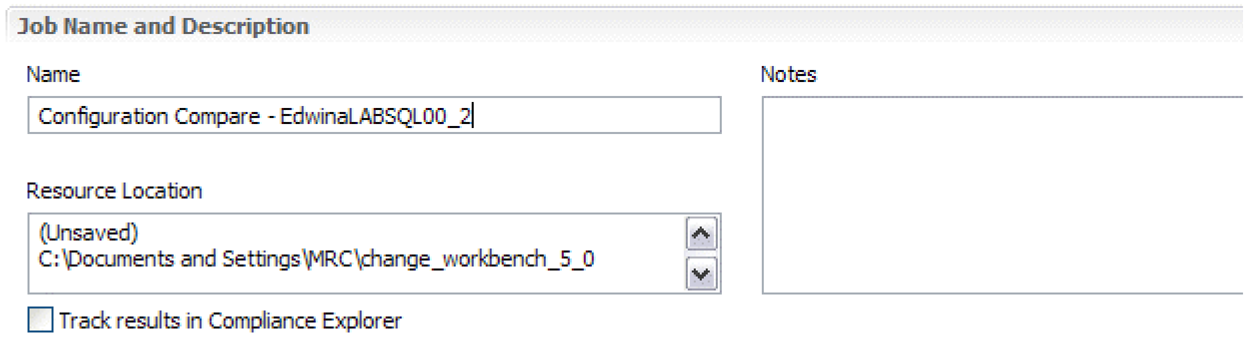
- Select **File > New > Configuration Comparison Job** in the Menu Bar.

  The **Configuration Comparison Job Editor** opens.

You can now proceed to define the parameters of the new job.

Specify a Comparison Job Name

A job name is used to distinguish between configuration comparison jobs composed of specific data sources. A name is defined when a new job is started and is used to save jobs for later use or repeated comparisons of the same pair of data sources.



Job Name and Description

Name: Configuration Compare - EdwinaLABSQL00_2

Resource Location: (Unsaved) C:\Documents and Settings\MRC\change_workbench_5_0

Notes:

☐ Track results in Compliance Explorer

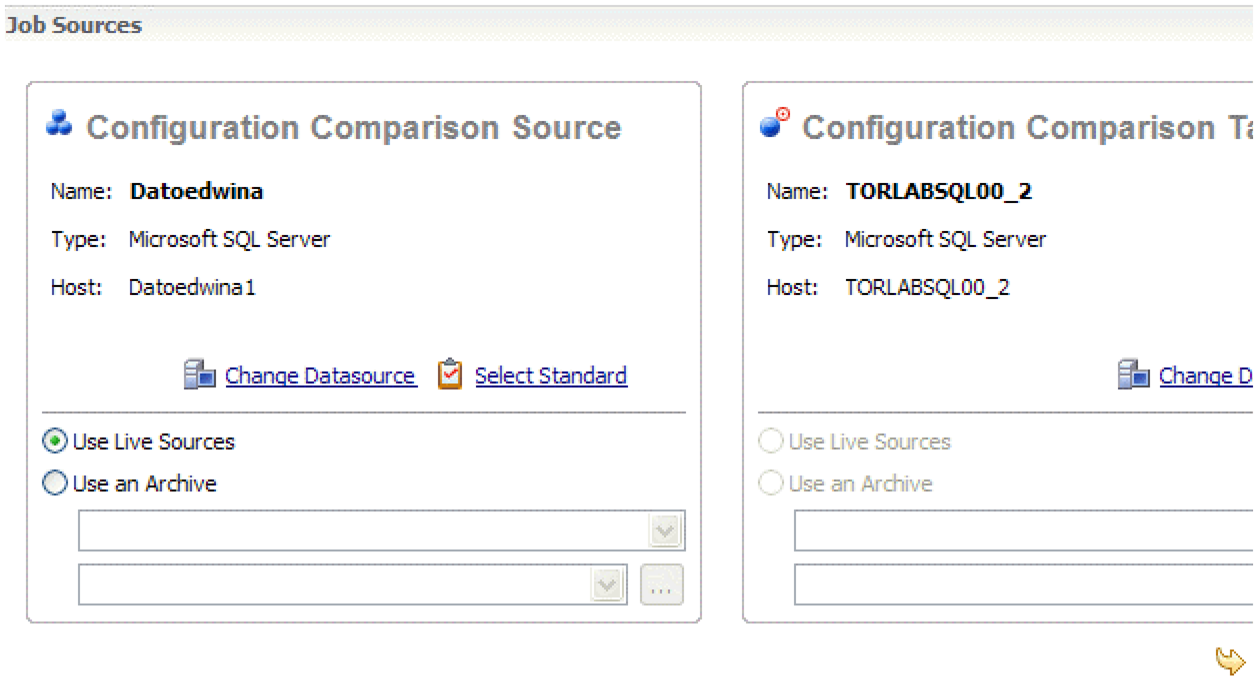
To name a configuration comparison job

- Type the name of the job in the **Name** field of the **Configuration Comparison Job Editor**.

NOTE: Click **Track Results in Compliance Editor** to enable this job in the **Compliance** view.

Specify Source and Target Data Sources

The **Job Sources** section of the Configuration Comparison Job Editor contains two boxes labeled **Comparison Source** and **Comparison Target**. These boxes identify the Data Sources, Archives, and Standards to be used in the job.



Job Sources

Configuration Comparison Source

Name: Datoedwina

Type: Microsoft SQL Server

Host: Datoedwina1

Change Datasource | Select Standard

◉ Use Live Sources

○ Use an Archive

Configuration Comparison Ta

Name: TORLABSQL00_2

Type: Microsoft SQL Server

Host: TORLABSQL00_2

Change D

○ Use Live Sources

○ Use an Archive

To add a data source to the job

- Click and drag a data source from **Data Source Explorer** to the Configuration Comparison Source or Configuration Comparison Target box, or click **Select Data Source** in the appropriate box and choose a data source from the dialog provided.

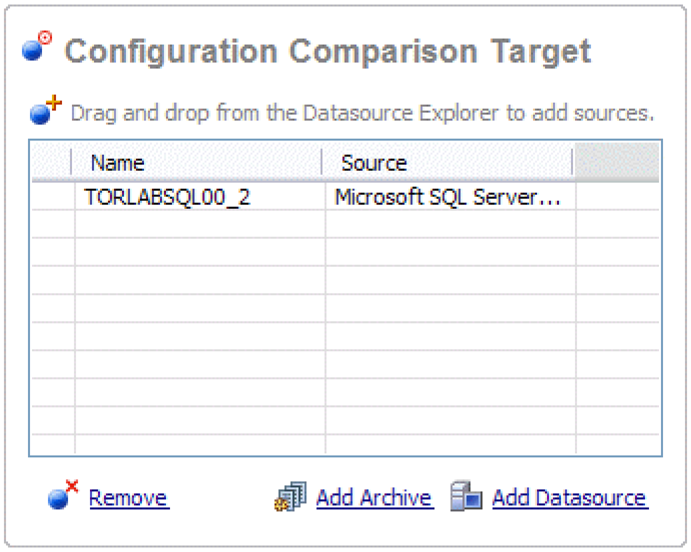
To add a Standard to the job

- Click and drag a Standard from **Data Source Explorer** to the **Comparison Source** box, or click **Select Standard** and choose a data source from the dialog provided.

When a data source is added to a job, it is displayed by **Name**, **Type**, and **Host**. You can change a data source by clicking **Change Data Source** in the appropriate box and selecting a new data source from the dialog, or by dragging a different data source to the box from data source Explorer.

The **Configuration Comparison Source** and **Configuration Comparison Target** boxes must contain data sources, Archives, or Standards before the job can be executed.

To add multiple Comparison Targets, click **Add More Targets** to enable a **One-to-Many** comparison.


Configuration Comparison Target
Drag and drop from the Datasource Explorer to add sources.
Remove
Add Archive
Add Datasource

Name	Source
TORLABSQL00_2	Microsoft SQL Server...

To add multiple targets to the comparison job

- Drag and drop additional data sources or Archives from **Data Source Explorer** to the **Configuration Comparison Target** box, or click the **Add Archive** or **Add Data Source** commands beneath the table and select the appropriate target from the dialog provided.

Additionally, you can specify whether you want to compare the current properties of the live data source, or an archived set of properties previously stored in Change Manager using the radio buttons beneath each box.

- Select **Use Live Sources** to compare the current properties of the data source when the job is executed.
- Select **Use an Archive** and select the Archive and version to use in the job.

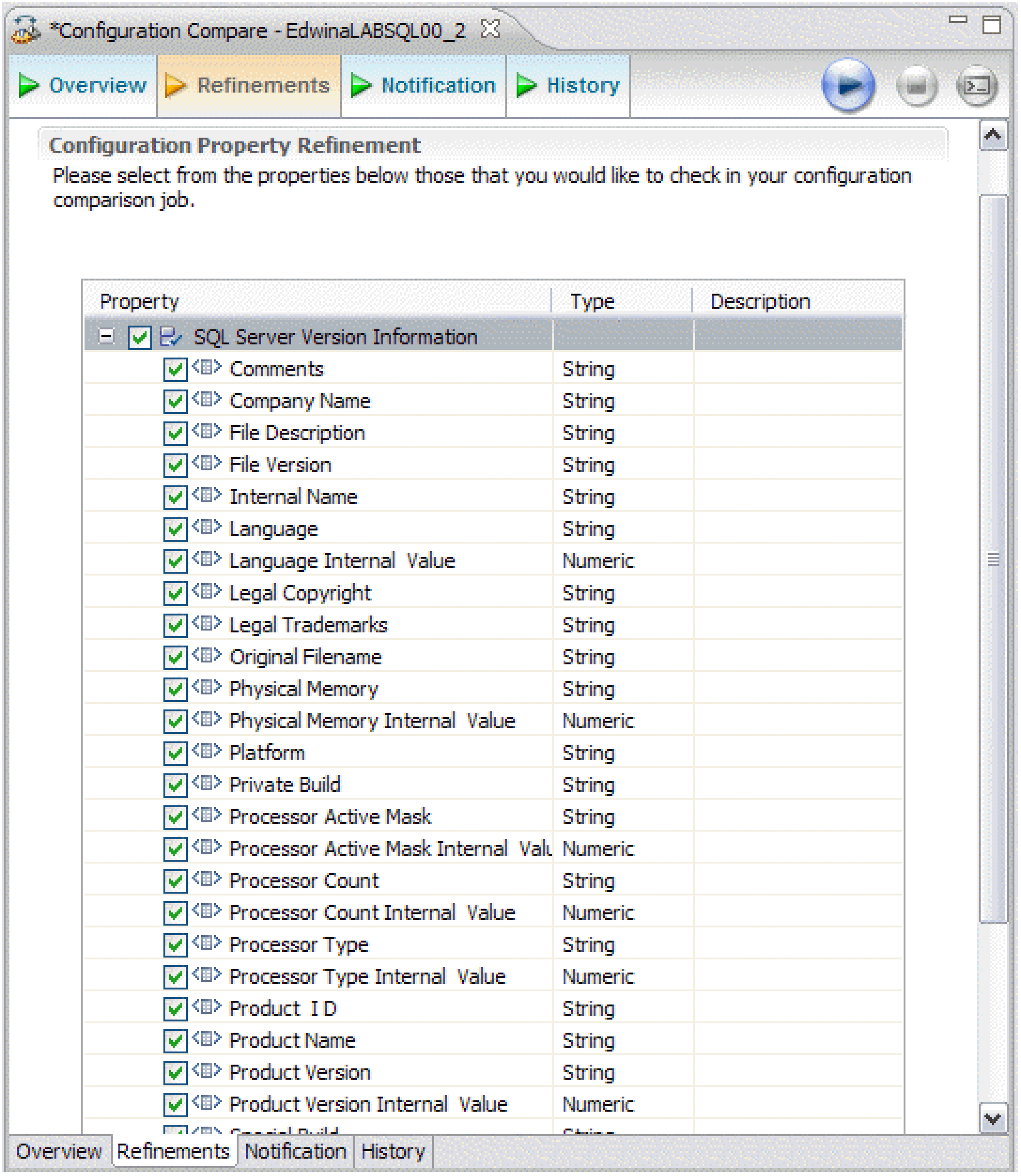
NOTE: You can drag and drop data sources, Archive Jobs, Archive Versions, and Standards into the Comparison Source and Comparison Target cells. A Standard can only be the Source of a Configuration Comparison Job.

Refine Property Comparisons

Configuration property refinement involves choosing specific configuration properties to compare and specific configuration properties to omit prior to executing a comparison job.

By default, when you specify a data source as a comparison job Source or Target, you can refine the properties to be compared prior to executing the job on the **Refinements** tab of the **Configuration Comparison Job Editor**.

Configuration properties are organized by tree and listed by **Property** name, **Description**, **Data Source Location**, and **Source**.



Configuration Property Refinement

Please select from the properties below those that you would like to check in your configuration comparison job.

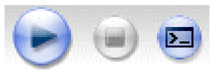
Property	Type	Description
✓ SQL Server Version Information		
✓ Comments	String	
✓ Company Name	String	
✓ File Description	String	
✓ File Version	String	
✓ Internal Name	String	
✓ Language	String	
✓ Language Internal Value	Numeric	
✓ Legal Copyright	String	
✓ Legal Trademarks	String	
✓ Original Filename	String	
✓ Physical Memory	String	
✓ Physical Memory Internal Value	Numeric	
✓ Platform	String	
✓ Private Build	String	
✓ Processor Active Mask	String	
✓ Processor Active Mask Internal Valu	Numeric	
✓ Processor Count	String	
✓ Processor Count Internal Value	Numeric	
✓ Processor Type	String	
✓ Processor Type Internal Value	Numeric	
✓ Product I D	String	
✓ Product Name	String	
✓ Product Version	String	
✓ Product Version Internal Value	Numeric	

To specify the properties used in the comparison job

Navigate to the **Refinements** tab and select or deselect the configuration properties in the table, as required. Selected properties are included in the comparison job, and de-selected properties are ignored during job execution.

Execute Comparison Job

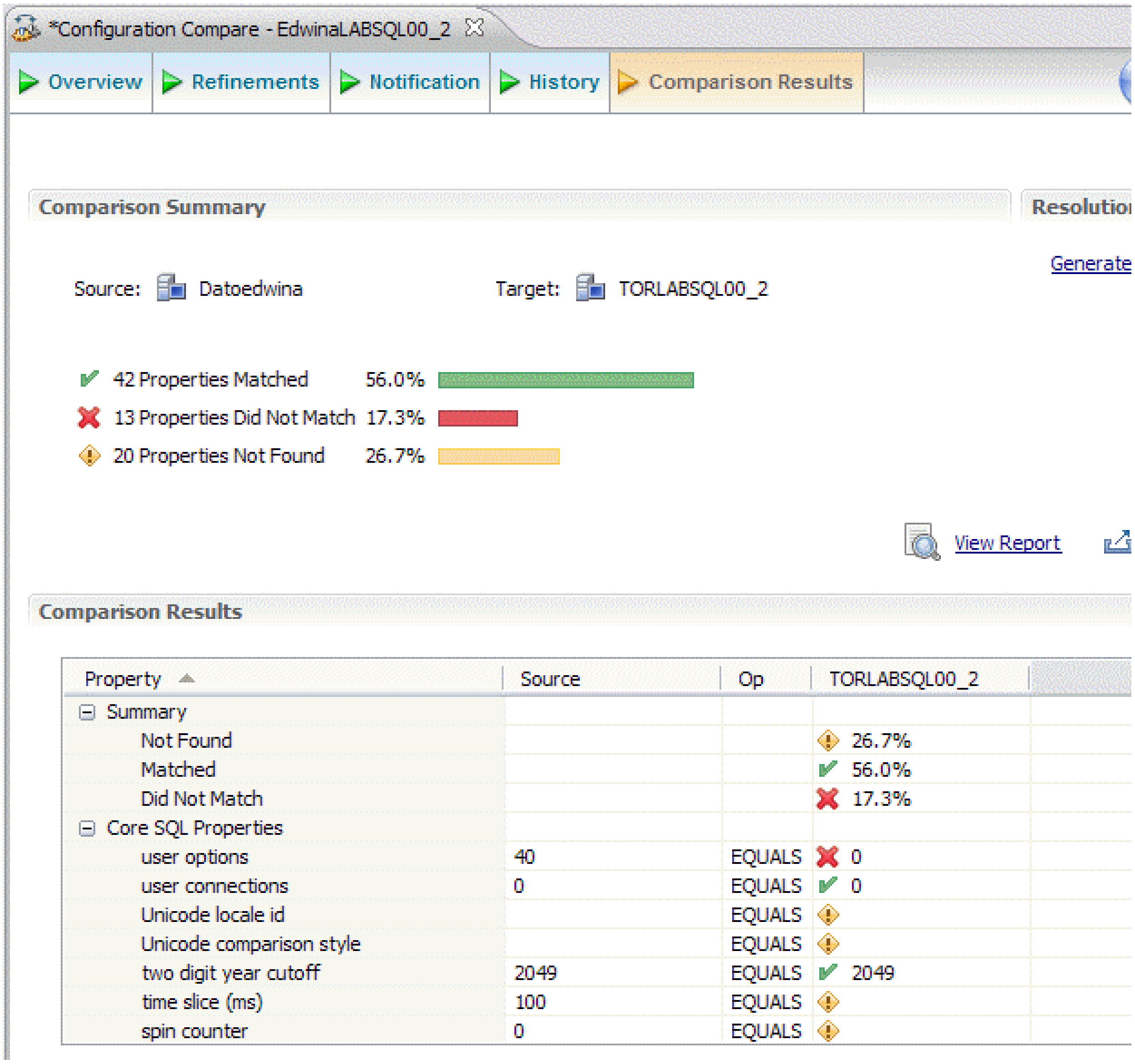
The configuration comparison job runs when you click **Run Job** on any tab in the Configuration Comparison Job Editor.



When the process executes, the **Comparison Results** tab appears and displays the progress of the job and a summary of the comparison job results.

View Results

When a job is complete, results are displayed on the **Comparison Results** tab.



Source: Datoedwina   Target: TORLABSQL00_2

✓ 42 Properties Matched	56.0%	🟩
✗ 13 Properties Did Not Match	17.3%	🟥
◆ 20 Properties Not Found	26.7%	🟨

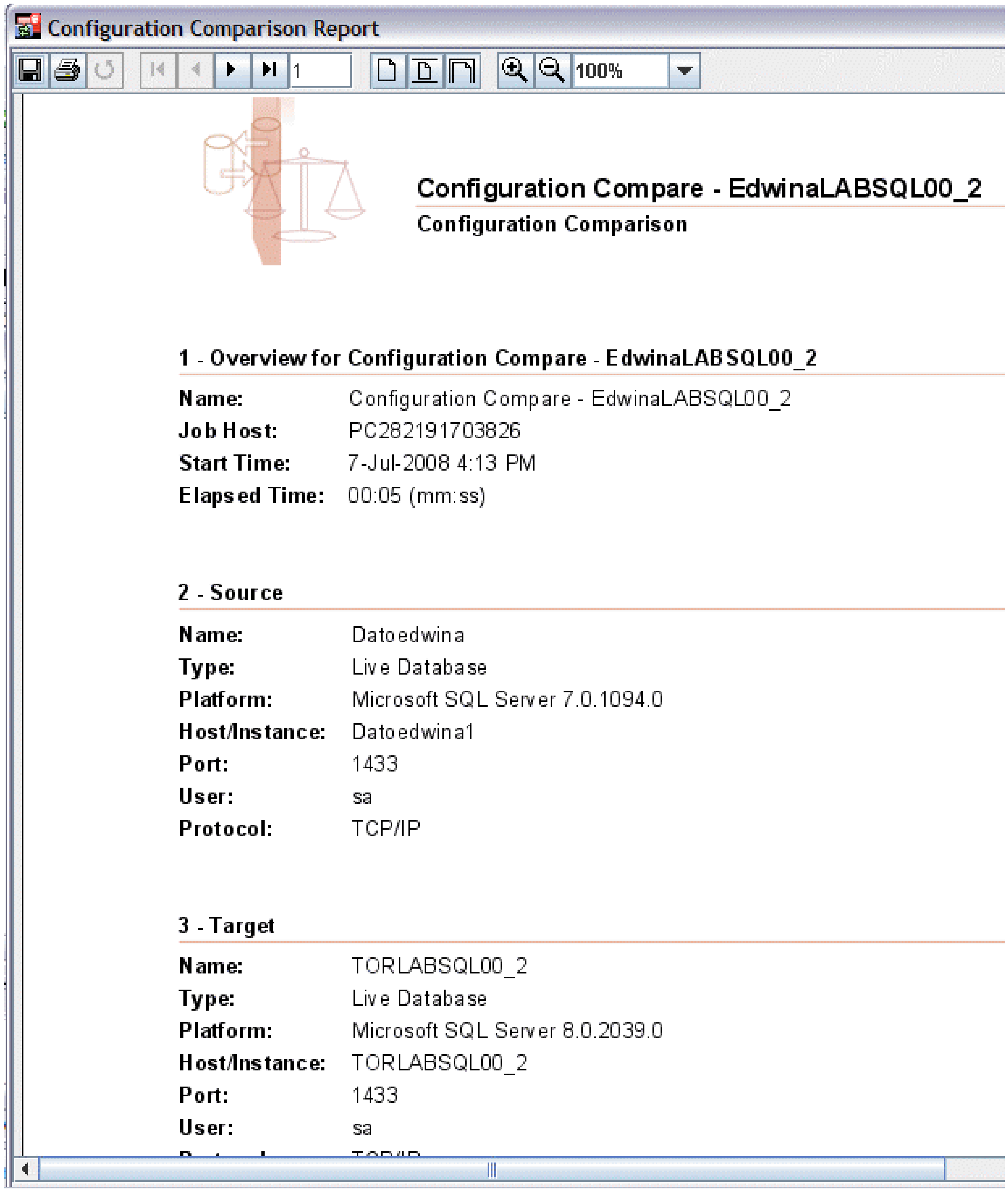
Property	Source	Op	TORLABSQL00_2
⊟ Summary			
Not Found			◆ 26.7%
Matched			✓ 56.0%
Did Not Match			✗ 17.3%
⊟ Core SQL Properties			
user options	40	EQUALS	✗ 0
user connections	0	EQUALS	✓ 0
Unicode locale id		EQUALS	◆
Unicode comparison style		EQUALS	◆
two digit year cutoff	2049	EQUALS	✓ 2049
time slice (ms)	100	EQUALS	◆
spin counter	0	EQUALS	◆

Job results are displayed by property with an overview of the matching and non-matching properties located at the top of the page. Individual details about each property are listed in the **Comparison Results** table.

- The **Result** column contains a check mark or X icon depending on whether the corresponding property value matched between the **Source** and **Target** data sources.
- The **Source Value** column contains the value of the property on the **Source**.
- The **Op** column provides the operator used in the job comparison that defines what constitutes a match. (This can only be specified when a **Standard** is used in the job. Otherwise, the operator is always **equals**.)
- The **Target Value** column contains the value of the property on the **Target**. The first three rows in the **Summary** section provide the overall number of matched and mismatched parameters for each source.

Generating a Job Report

Once you have executed a comparison job, you can generate a job report via the **View Report** command on the **Results** tab.



Configuration Compare - EdwinaLABSQL00_2
Configuration Comparison

1 - Overview for Configuration Compare - EdwinaLABSQL00_2

Name:	Configuration Compare - EdwinaLABSQL00_2
Job Host:	PC282191703826
Start Time:	7-Jul-2008 4:13 PM
Elapsed Time:	00:05 (mm:ss)

2 - Source

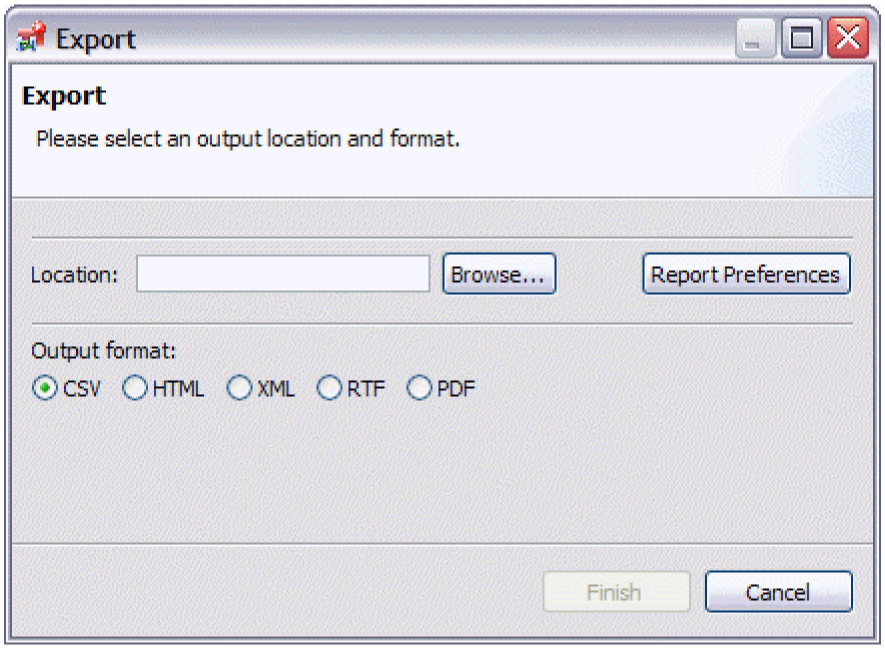
Name:	Datoedwina
Type:	Live Database
Platform:	Microsoft SQL Server 7.0.1094.0
Host/Instance:	Datoedwina1
Port:	1433
User:	sa
Protocol:	TCP/IP

3 - Target

Name:	TORLABSQL00_2
Type:	Live Database
Platform:	Microsoft SQL Server 8.0.2039.0
Host/Instance:	TORLABSQL00_2
Port:	1433
User:	sa

The report opens in a separate window and can be read on screen, sent to a printer, and saved to disk.

Additionally, the **Export Results** command will produce a report file in a specified format at a directory location of your choosing.



Export

Export

Please select an output location and format.

Location:

Output format: [✓ CSV] HTML XML RTF PDF

Click **Export Results** on the **Results** tab and fill in the fields, as required. When you click **Finish**, the report is created in the directory specified in the **Location** field, in the format specified in the **Output Format** section.

Creating a Configuration Standard

A Standard is an object specific to configuration comparison jobs that contains a custom set of stored data source configuration properties. Standards are used in configuration comparison jobs against registered data sources and Archives and provide a means of defining and retaining a specific set of configuration parameter values and thresholds to ‘check’ against your enterprise and ensure all network systems meet data complacency measures.

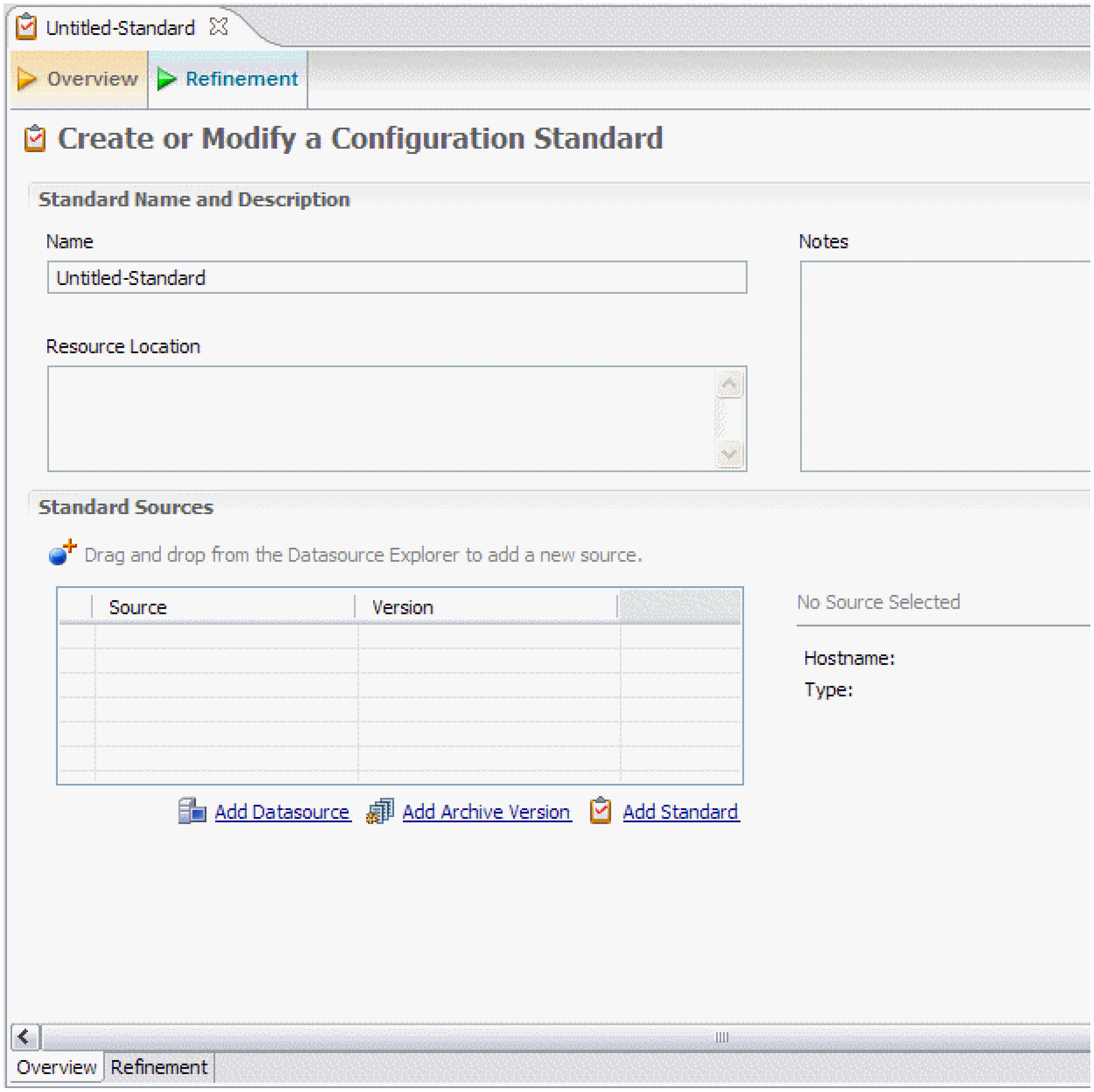
All Standards are composed of a name, description, and a list of properties that are assigned set values. Properties are derived from live data sources, Archives, or other Standards, Additionally, each property value is assigned an operator that asserts if the same value on an opposing data source/Archive in a comparison job falls within a specific range, is the member of a set, is greater than a value, etc.

The following tasks provide a high-level overview of the Standard definition process:

- Create a New Standard
- Specify a Standard Name
- Add Property Sources
- Refine Properties
- Save Standard
- Use Standard in Configuration Comparison Jobs

Create a New Standard

Standards are defined in **Standards Editor**.



To create a new Standard

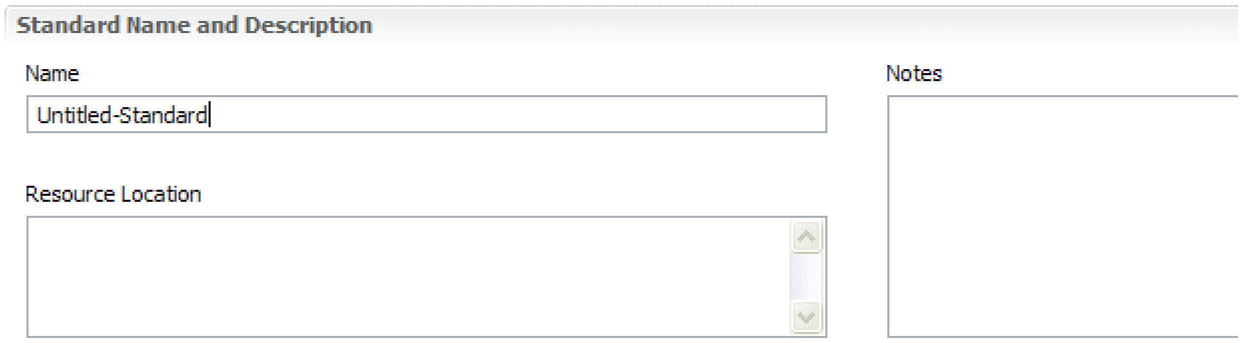
- Select **File > New > Configuration Standard** in the Menu Bar. The Standards Editor opens.

If you want to add the properties of an existing data source or Archive to a new Standard, click and drag the appropriate object from **Data Source Explorer** to the **Standard Sources** table in **Standard Editor**. All of the properties of the selected data source or Archive populate the new Standard.

You can now proceed to define the parameters of the new Standard.

Specify a Standard Name

A name must be defined when a new Standard is created. It is used to distinguish bet6ween Standards and other objects in Change Manager.



Ensure you specify a meaningful name that identifies the Standard in other views and dialogs later on. The **Resource Location** field automatically populates with the directory location of the Standard file once it has been saved. (See Save Standard for details on how to save the standard once it is fully defined.)

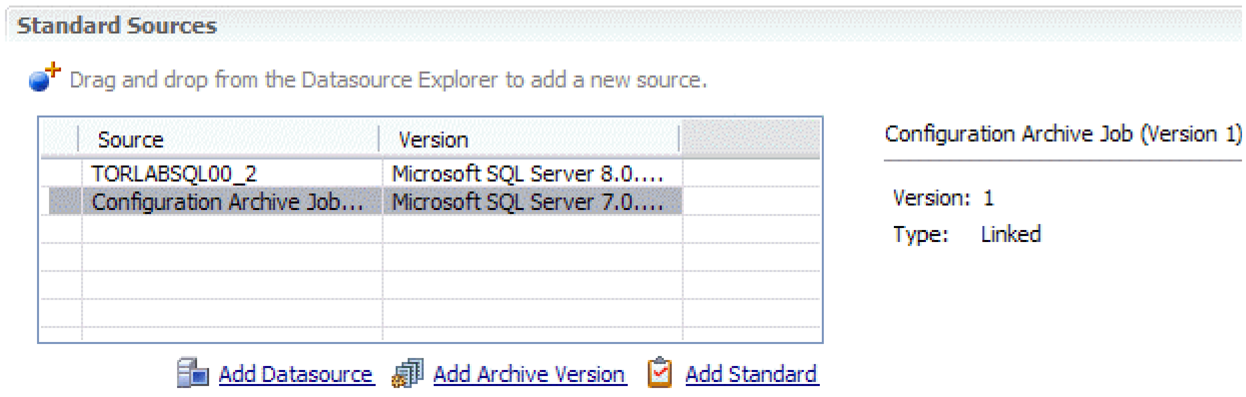
To name a Standard

- Enter the name of the Standard in the **Name** field of the **Standard Editor**.

You can now proceed to define the parameters of the new Standard.

Add Property Sources

The **Standard Sources** table specifies the amalgamation of registered data sources, Archives, and other Standards from which the new Standard properties are derived.


Standard Sources
Drag and drop from the Datasource Explorer to add a new source.
Add Datasource
Add Archive Version
Add Standard

Source	Version
TORLABSQL00_2	Microsoft SQL Server 8.0....
Configuration Archive Job...	Microsoft SQL Server 7.0....

Configuration Archive Job (Version 1)

Version: 1

Type: Linked

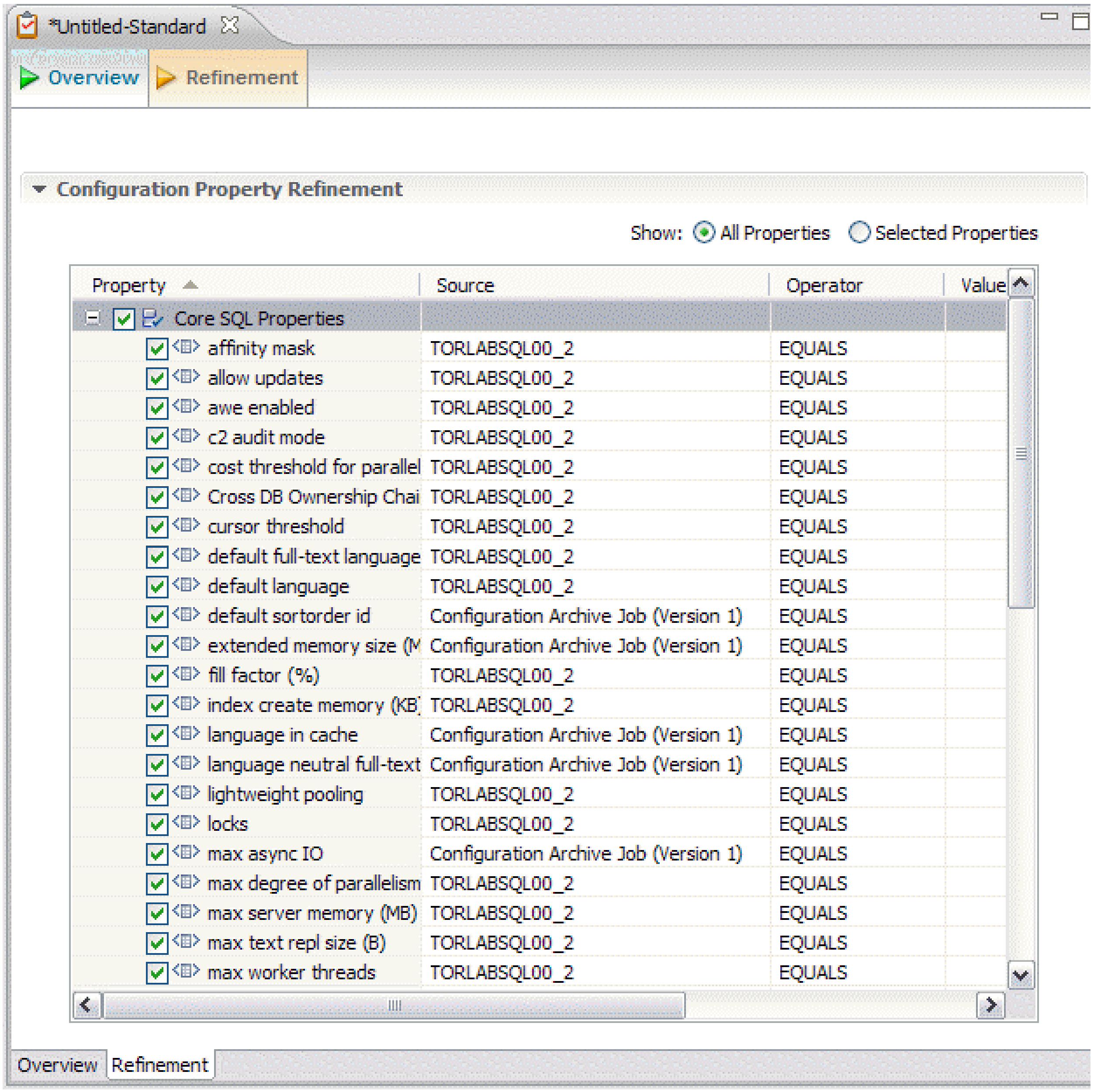
By default, Change Manager automatically adds all properties of a selected source to the Standard definition. You need to manually refine property selections on the Refinements tab once sources have been added. See Refine Properties for more information.

To add properties to a Standard

- Drag and drop a data source, Archive, or Standard from **Data Source Explorer** to the **Standard Sources** table. Alternatively, use the **Select Standard**, **Select Data Source**, and **Select Archive Version** commands beneath the table to add properties to the Standard definition.

Refine Properties

The **Standard Refinement** tab displays all sources and corresponding properties applied to the Standard. All properties are listed by **Property**, **Source**, **Type**, **Operator**, and **Value**.



Show: [✓ All Properties] Selected Properties

Property	Source	Operator	Value
Core SQL Properties			
affinity mask	TORLABSQL00_2	EQUALS	
allow updates	TORLABSQL00_2	EQUALS	
awe enabled	TORLABSQL00_2	EQUALS	
c2 audit mode	TORLABSQL00_2	EQUALS	
cost threshold for parallel	TORLABSQL00_2	EQUALS	
Cross DB Ownership Chai	TORLABSQL00_2	EQUALS	
cursor threshold	TORLABSQL00_2	EQUALS	
default full-text language	TORLABSQL00_2	EQUALS	
default language	TORLABSQL00_2	EQUALS	
default sortorder id	Configuration Archive Job (Version 1)	EQUALS	
extended memory size (M	Configuration Archive Job (Version 1)	EQUALS	
fill factor (%)	TORLABSQL00_2	EQUALS	
index create memory (KB	TORLABSQL00_2	EQUALS	
language in cache	Configuration Archive Job (Version 1)	EQUALS	
language neutral full-text	Configuration Archive Job (Version 1)	EQUALS	
lightweight pooling	TORLABSQL00_2	EQUALS	
locks	TORLABSQL00_2	EQUALS	
max async IO	Configuration Archive Job (Version 1)	EQUALS	
max degree of parallelism	TORLABSQL00_2	EQUALS	
max server memory (MB)	TORLABSQL00_2	EQUALS	
max text repl size (B)	TORLABSQL00_2	EQUALS	
max worker threads	TORLABSQL00_2	EQUALS	

You can choose to include or exclude properties in the Standard on the **Standard Refinement** tab, as well as set thresholds and otherwise specify what constitutes a ‘pass’ or a ‘fail’ when a comparison job is run per individual property value, via the **Operator** column.

The following Operators can be applied to properties in a Standard:

- Less Than
- Greater Than
- Greater Than Equal To
- Equals
- Less Than or Equal To
- Not Equals

The **Value** column indicates the value that the **Operator** definition must match.

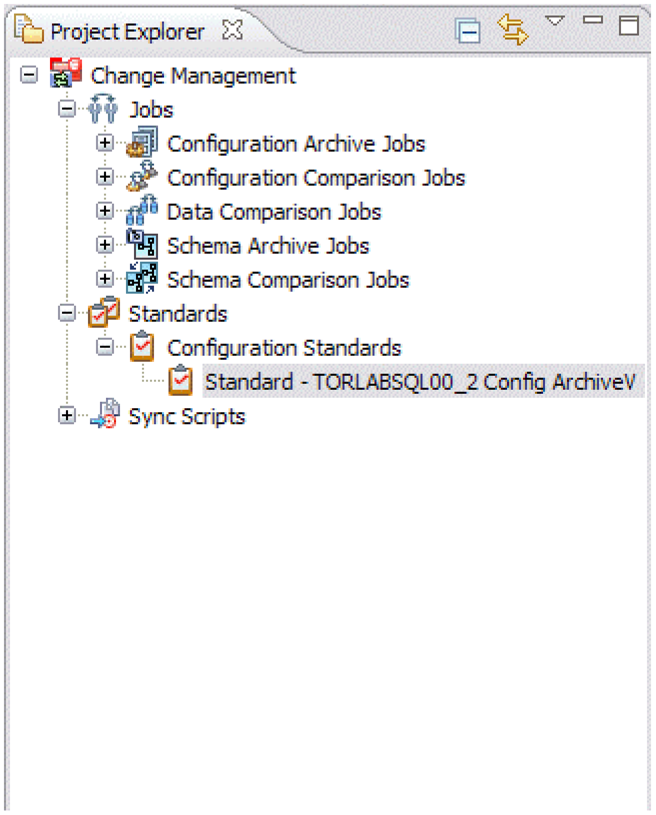
The **Type** column indicates if individual properties are linked or static. Linked property values refresh with the latest source values each time the Standard is run in a job, while Static property values remain fixed and do not updated based on source values. When you have multiple sources selected, select the source name in this column to specify which value the standard updates from.

To refine Standard properties

1. Define each property as **Linked** or **Static** by clicking the appropriate row value in the **Type** column and choosing a selection from the drop down menu.
2. Define the operator used to determine the conditions of a matching result when the job runs by clicking the appropriate row value in the **Operator** column and choosing a selection from the drop down menu. If you choose the **Pattern Match** operator, remember to include a regular expression in the corresponding **Value** column.

Save Standard

Once you have defined the Standard, it can be saved via the **File>Save**, **File>Save As**, or **File>Save All** commands in the Menu Bar.



Change Manager saves the Standard under the name you specified in the **Name** field in Project Explorer. You can use the Standard in configuration comparison jobs as you would a data source or Archive.

Use Standard in Configuration Comparison Jobs

Standards are used in configuration comparison jobs the same way as registered data sources or Archives, with the noted exception that they may only be designated as Sources due to the unique property operator values they contain.

To add a Standard to a configuration comparison job

- Drag and drop the Standard from **Project Explorer** to the **Comparison Source** box in the **Configuration Comparison Job Editor**. You can now proceed to run the job using the Standard's properties as you would a normal data source.

Creating a Configuration Archive

An Archive is an object related to configuration comparison jobs that contains a snapshot of the configuration parameters of a registered data source. A Configuration Archive is used in configuration comparisons against other data sources, standards, and archives to monitor property value slippage, provide backup settings, or otherwise enable the use of preserved data source configuration settings in configuration comparison jobs.

All Archives are composed of a name, description, and a list of configuration properties derived from a single live data source at a specific point in time. Additionally, Archives have Versioning functionality and can be periodically updated with the current values of an associated data source. Older Versions are retained within an Archive and can be recalled as needed.

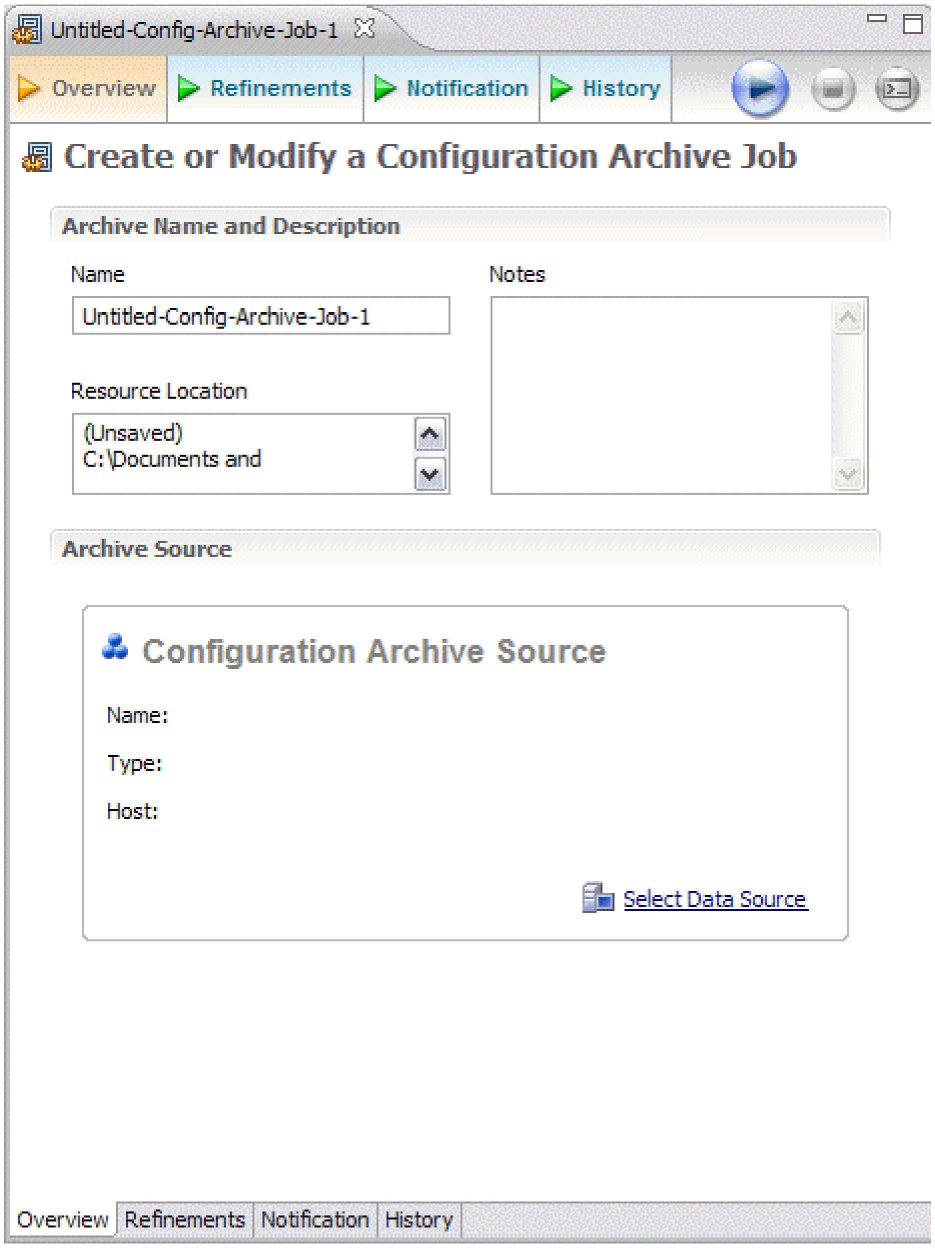
Archives are located in data source Explorer on the Configuration Archive Jobs node of the corresponding data source.

The following tasks provide a high-level overview of the Archive definition process:

- Create a New Configuration Archive
- Specify an Archive Name
- Specify an Archive Source
- Refine Properties
- Specify Notification Options
- Save the Archive/Manage Versioning
- Generating a Job Report
- Use Archive in Configuration Comparison Jobs

Create a New Configuration Archive

Archives are defined in the **Configuration Archive Job Editor**.



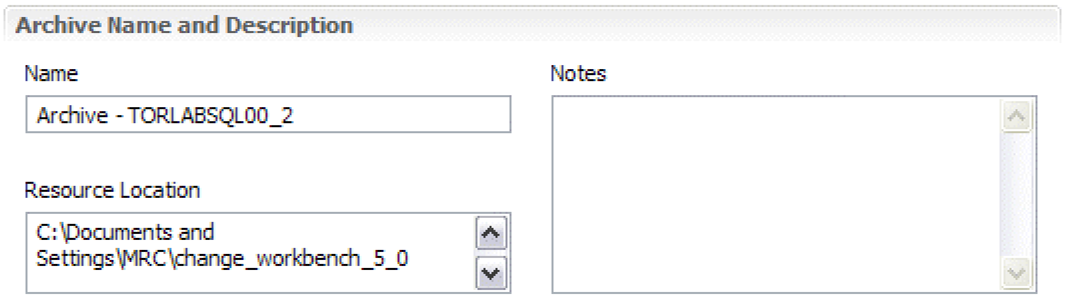
When an archive is initially created it is in an unsaved state and versionless. Once you define the archive and refine its properties, you need to issue the **Run Job** command to create a Version and save the archive in the Change Manager environment. Unsaved Archives cannot be used in comparison jobs and do not appear in data source Explorer. More information regarding the process of refining and saving a new Archive are located in the other steps of this section.

To create a Configuration Archive

- Select **File > New > Configuration Archive Job** from the menu. The Configuration Archive Editor opens.

Specify an Archive Name

A name must be defined when a new Archive is created. The name is used to distinguish between Archives and other objects in Change Manager, and is used in Versioning functionality later in the process.



Archive Name and Description

Name: Archive - TORLABSQL00_2

Resource Location: C:\Documents and Settings\MRC\change_workbench_5_0

Notes:

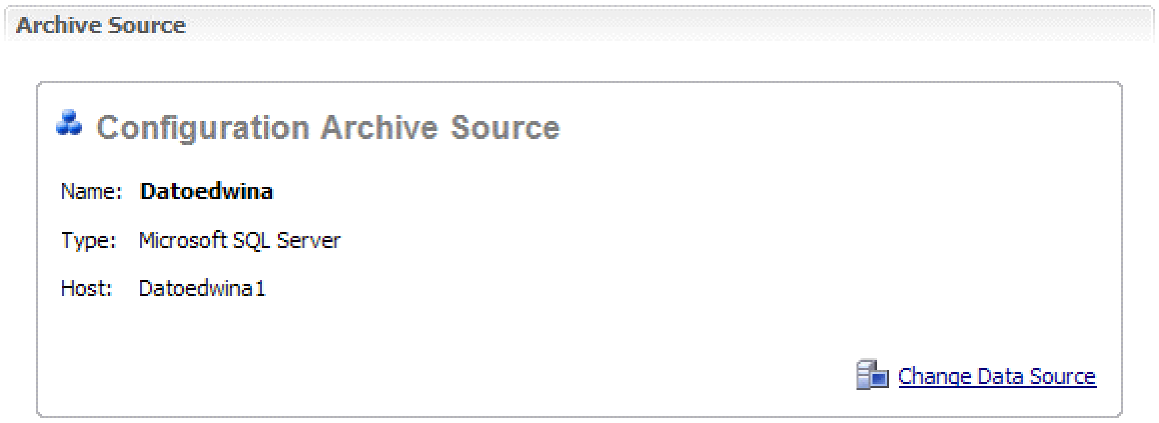
Ensure you specify a meaningful name that identifies the Archive in other views and dialogs later.

To name a Configuration Archive

- Enter the name of the Archive in the **Name** field of the Archive Job Editor.

Specify an Archive Source

The **Configuration Archive Source** table specifies the data source from which the archive is derived.



Archive Source

Configuration Archive Source

Name: **Datoedwina**

Type: Microsoft SQL Server

Host: Datoedwina1

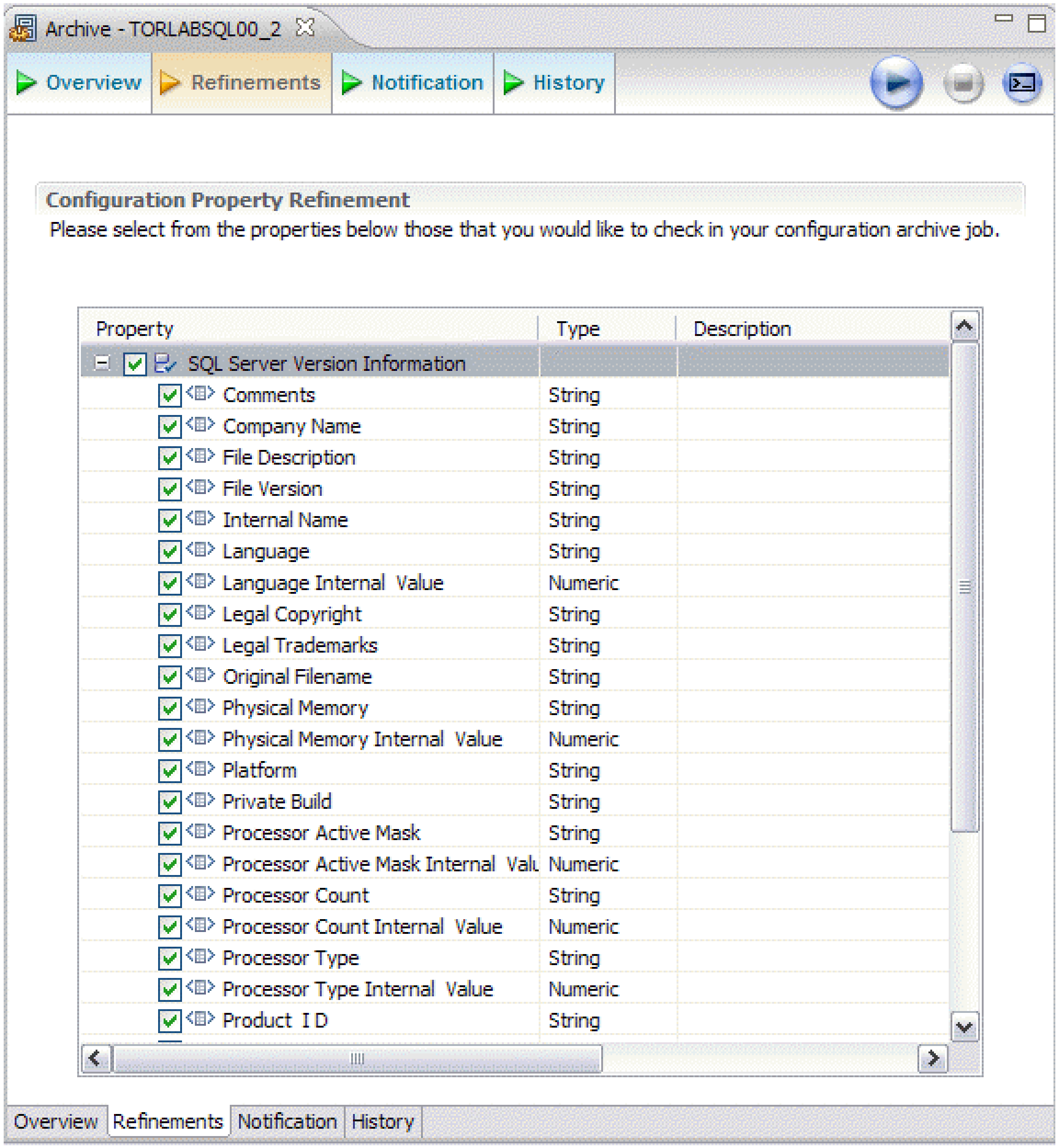
To specify the archive source

- Drag and drop the data source to be archived from **Data Source Explorer** to the **Configuration Archive Source** box. The data source is listed by name, DBMS type, and host.

Refine Properties

By default, when you create a new archive, all data source properties are included. However, you may only want or need to archive a specific group of properties per archive defined. For example, you may want to define an archive that retains only the security properties of the associated data source. Other archives can be created (and likewise refined) for the same data source, but maintaining a different group of selected properties, such as an archive that displays only performance settings.

Archive configuration properties are located on the **Refinements** tab of the **Archive Job Editor**, where they are listed by name.



Configuration Property Refinement

Please select from the properties below those that you would like to check in your configuration archive job.

Property	Type	Description
✓ SQL Server Version Information		
✓ Comments	String	
✓ Company Name	String	
✓ File Description	String	
✓ File Version	String	
✓ Internal Name	String	
✓ Language	String	
✓ Language Internal Value	Numeric	
✓ Legal Copyright	String	
✓ Legal Trademarks	String	
✓ Original Filename	String	
✓ Physical Memory	String	
✓ Physical Memory Internal Value	Numeric	
✓ Platform	String	
✓ Private Build	String	
✓ Processor Active Mask	String	
✓ Processor Active Mask Internal Valu	Numeric	
✓ Processor Count	String	
✓ Processor Count Internal Value	Numeric	
✓ Processor Type	String	
✓ Processor Type Internal Value	Numeric	
✓ Product I D	String	

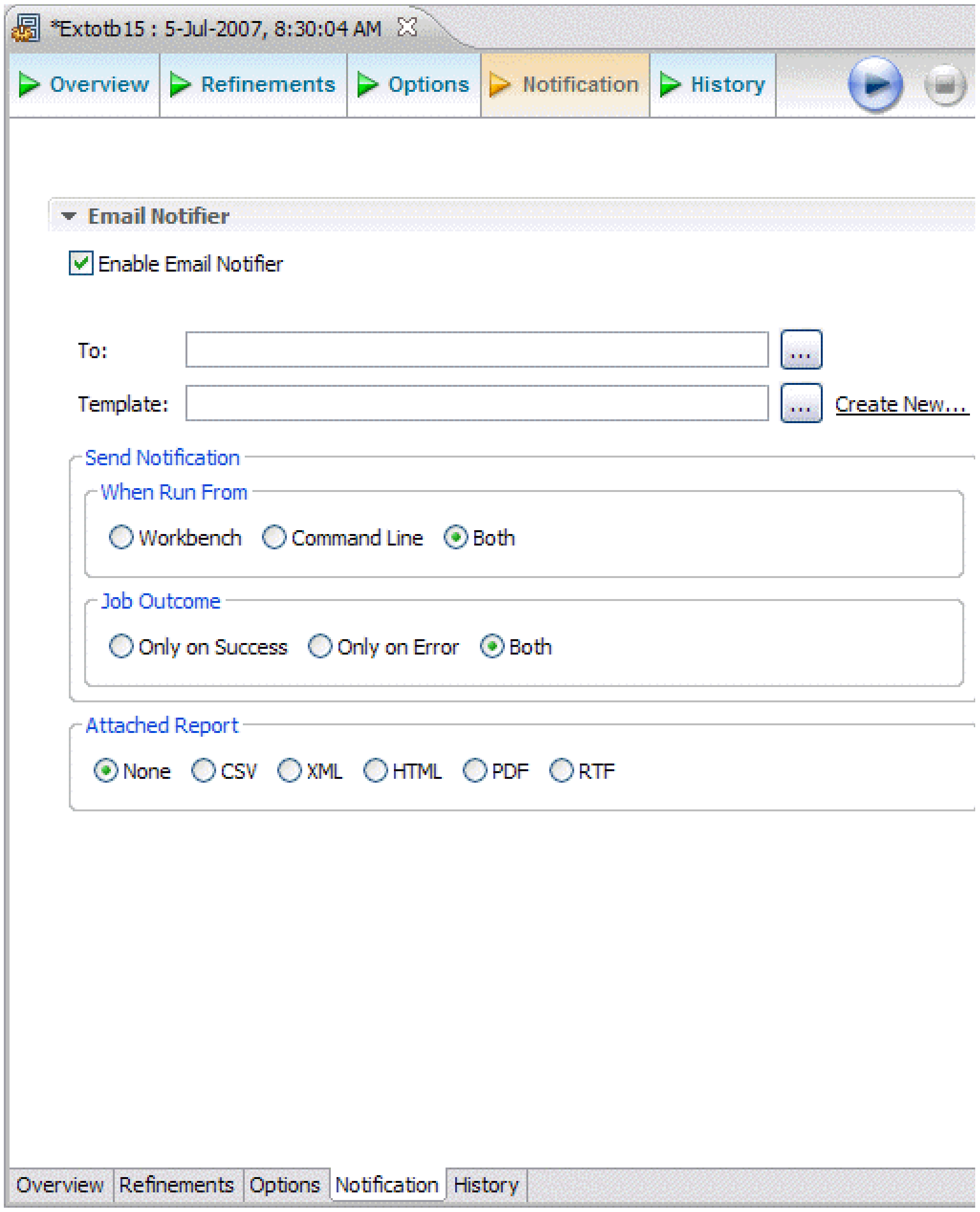
To refine Configuration Archive properties

- Use the check boxes to select or de-select properties as needed. Selected properties are included in the Archive when comparison jobs are run, while de-selected properties are ignored altogether.

When you have finished selecting the properties to include in the Archive, you will need to save the archive by creating a version (discussed in the following topic). Within the same Archive, property refinements are retained across Versions unless new refinements are specified. However, each time you create a new Archive, Change Manager includes all properties of the specified data source and you must again specify property refinements, if any.

Specify Notification Options

Notification options provide parameters for setting up email notification and report mailing for executed configuration archive jobs. The notification options tab enables you to specify the email addresses to which you want to send notifications, the email message format, and the conditions under which a notification should be sent.



[✓] Enable Email Notifier

To:

Template: Create New...

Send Notification

When Run From: Workbench, Command Line, [✓ Both]

Job Outcome: Only on Success, Only on Error, [✓ Both]

Attached Report: [✓ None], CSV, XML, HTML, PDF, RTF

To enable notification, select the **Enable Email Notifier** check box and select the options, as available. Enter the email addresses to which the notification will be sent in the **To** field, and specify an email format template in the **Template Field**.

In the **Send Notification** options, specify when the email notification should be sent:

- Under the **When Run From** options, indicate that email notifications should be sent when the job is executed from the **Workbench**, **Command Line**, or **Both**.
- Under the **Job Outcome** options**,** indicate that email notification should be sent **Only on Success**, **Only on Error**, or **Both**.
- Under the **For Threshold** options, specify the threshold level for which the email notification will be sent, based on the job results. Choose **Pass**, **Warn**, or **Fail**. You can click **Open Compliance Page** to specify the percentage values for each threshold warning, which are based on the percent match results of the job.

You can also specify the file format of an attached report that displays the job results with the notification email. Choose **None** (does not attach a report with the notification email), **CSV**, **XML**, **HTML**, **PDF**, or **RTF** to specify the file format of the report.

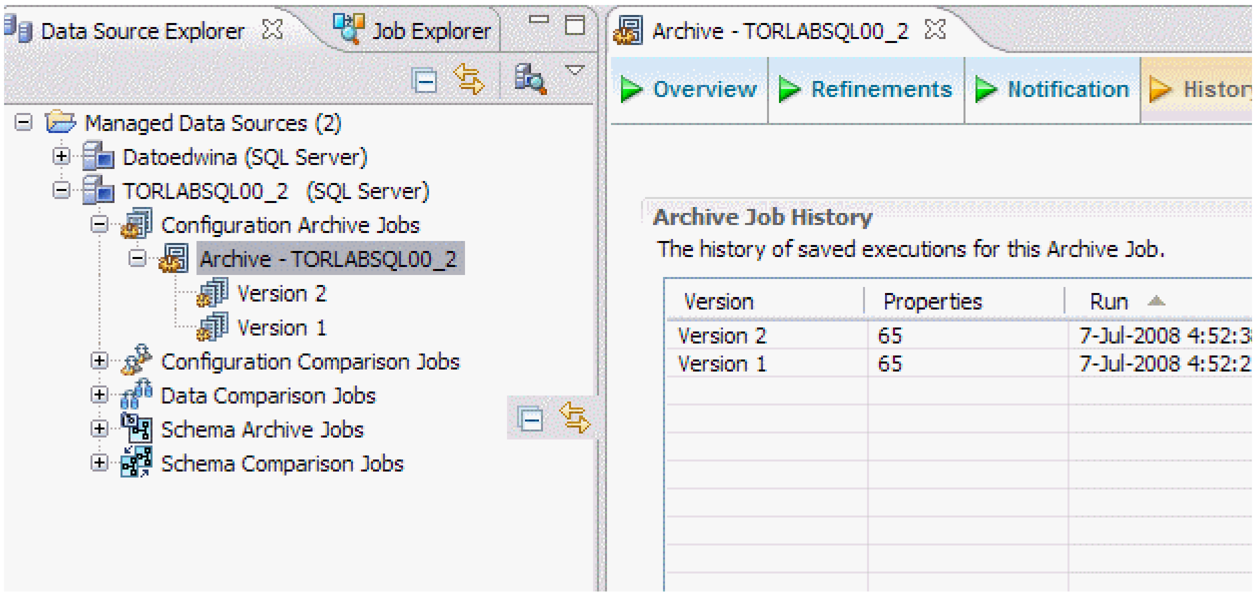
NOTE: Notification options can be modified via a plug-in extension that enables custom notification coding. See Creating and Implementing a Custom Configuration Plug-In for more information.

Save the Archive/Manage Versioning

A version is a specific set of configuration values associated with an **Archive** and is used to store older snapshots of the same archive for backup or tracking comparison purposes. Initially, you must create and save a version for the **Archive** to become valid within the system.

A version is created when you click the **Run** icon located in the lower right-hand corner of any tab in the **Archive Job Editor**.

When the process executes, the **New Unsaved Version** tab appears and lists the properties of the Archive by **Name**, **Operator**, and **Value** of the property as captured when the Archive was created. In order to create a version, highlight the tab and select **File > Save** or **File > Save All** from the Menu Bar. The system retains an older archive named **Version 1** that appears in the **Archive Job History** table on the **History** tab as well as in the **Data Source Explorer** tree under the **Configuration Archive Jobs** node.



Archive Job History
The history of saved executions for this Archive Job.

Version	Properties	Run
Version 2	65	7-Jul-2008 4:52:3
Version 1	65	7-Jul-2008 4:52:2

To create a version and save the archive job

- Click **Run Job**. A version appears in the **Archive Job History** table on the **History** tab containing the name of the **Archive**, the number of properties retained by the version, a time stamp, and where the archive job was run from. (Workbench for manual job execution, or command line, if automated.)

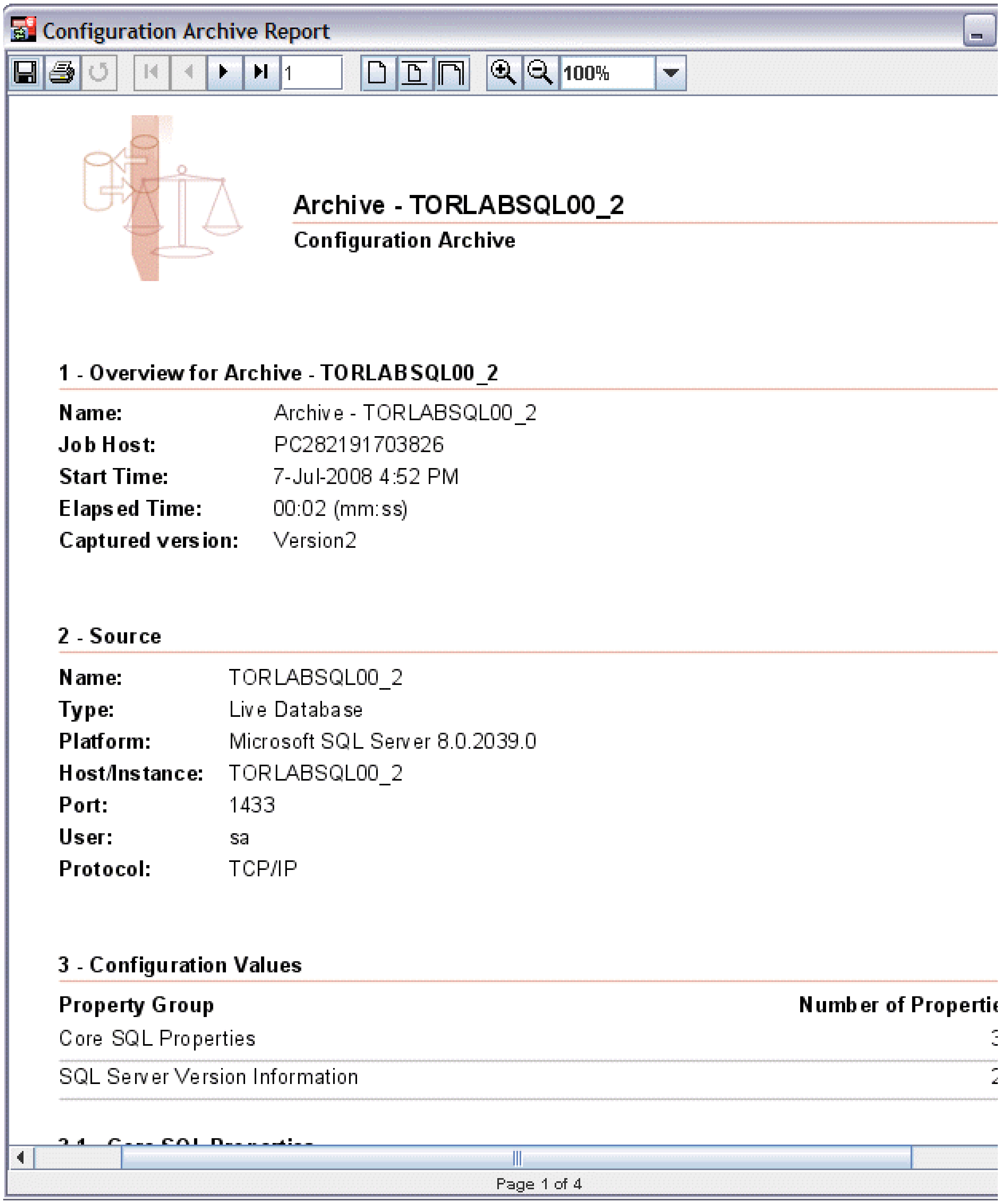
You create additional versions within the same archive job each time you click **Run Job**. The new version contains the latest values for the archive job properties as specified on the Refinements tab. Versions are listed on the Overview tab in a table that displays them in ascending order. A version can be accessed for viewing by double-clicking the appropriate Version in the table. The selected version opens in a new tab of the Archive Job Editor.

Additionally, Versions are listed as child nodes of the archive job to which they belong in Data Source Explorer. To view a Version, double-click the appropriate node.

Generating a Job Report

Once you have run an archive job, you can generate a job report that summarizes the archived configuration information. You can also generate job reports for any archive versions that exist in the system.

Click the **History** tab and then right-click on the version you want to create a report for. From the menu, select **View Report**, and a summary report of the configuration archive specific to the selected version appears in a separate window.



Archive - TORLABSQL00_2

Configuration Archive

1 - Overview for Archive - TORLABSQL00_2

Name:	Archive - TORLABSQL00_2
Job Host:	PC282191703826
Start Time:	7-Jul-2008 4:52 PM
Elapsed Time:	00:02 (mm:ss)
Captured version:	Version2

2 - Source

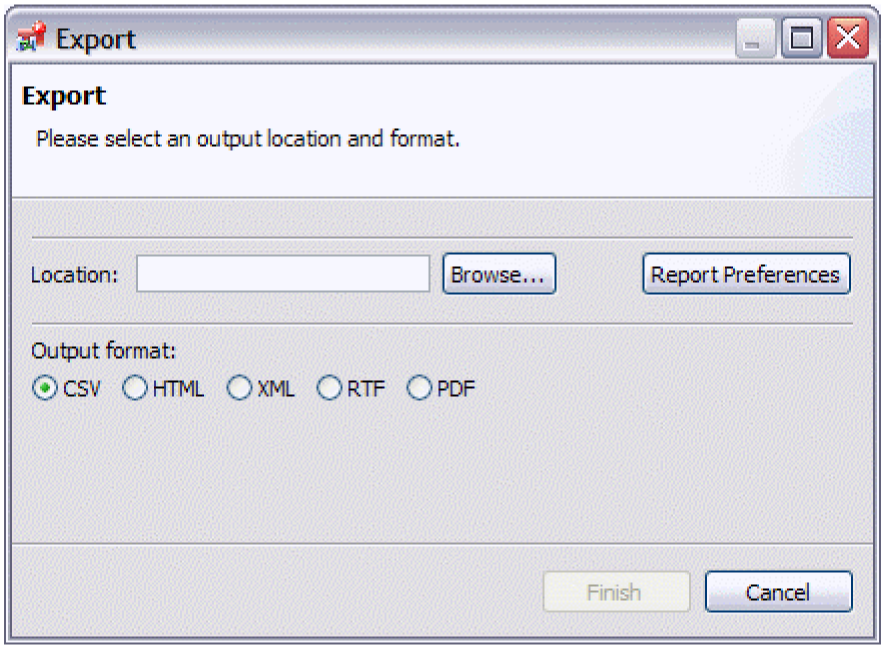
Name:	TORLABSQL00_2
Type:	Live Database
Platform:	Microsoft SQL Server 8.0.2039.0
Host/Instance:	TORLABSQL00_2
Port:	1433
User:	sa
Protocol:	TCP/IP

3 - Configuration Values

Property Group	Number of Propertie
Core SQL Properties	3
SQL Server Version Information	2

The report can be read on screen, sent to a printer, and saved to disk via the command icons located in the report window toolbar.

Additionally, the right-click menu contains a command named **Generate Report**. This will produce a report file in a specified format at a directory location of your choosing.



Fill in the fields, as required. When you click **Finish**, the report is created in the directory specified in the **Location** field, in the format specified in the **Output Format** section.

Use Archive in Configuration Comparison Jobs

Archives are used in configuration comparison jobs the same way as registered data sources or Standards.

To add an Archive to a configuration comparison job

- Drag and drop an Archive version from **Data Source Explorer** to the **Comparison Source** or **Comparison Target** boxes in the **Configuration Comparison Job Editor**. Alternatively, if a data source is selected as the **Source** or **Target**, select **Use an Archive** in the respective box and use the drop down menu to choose an associated archive from the list provided.

Comparing Schemas

Change Manager provides the ability to capture, compare, and synchronize schemas in databases and provide reports on schema comparison jobs that enable you to distribute information throughout your environment.

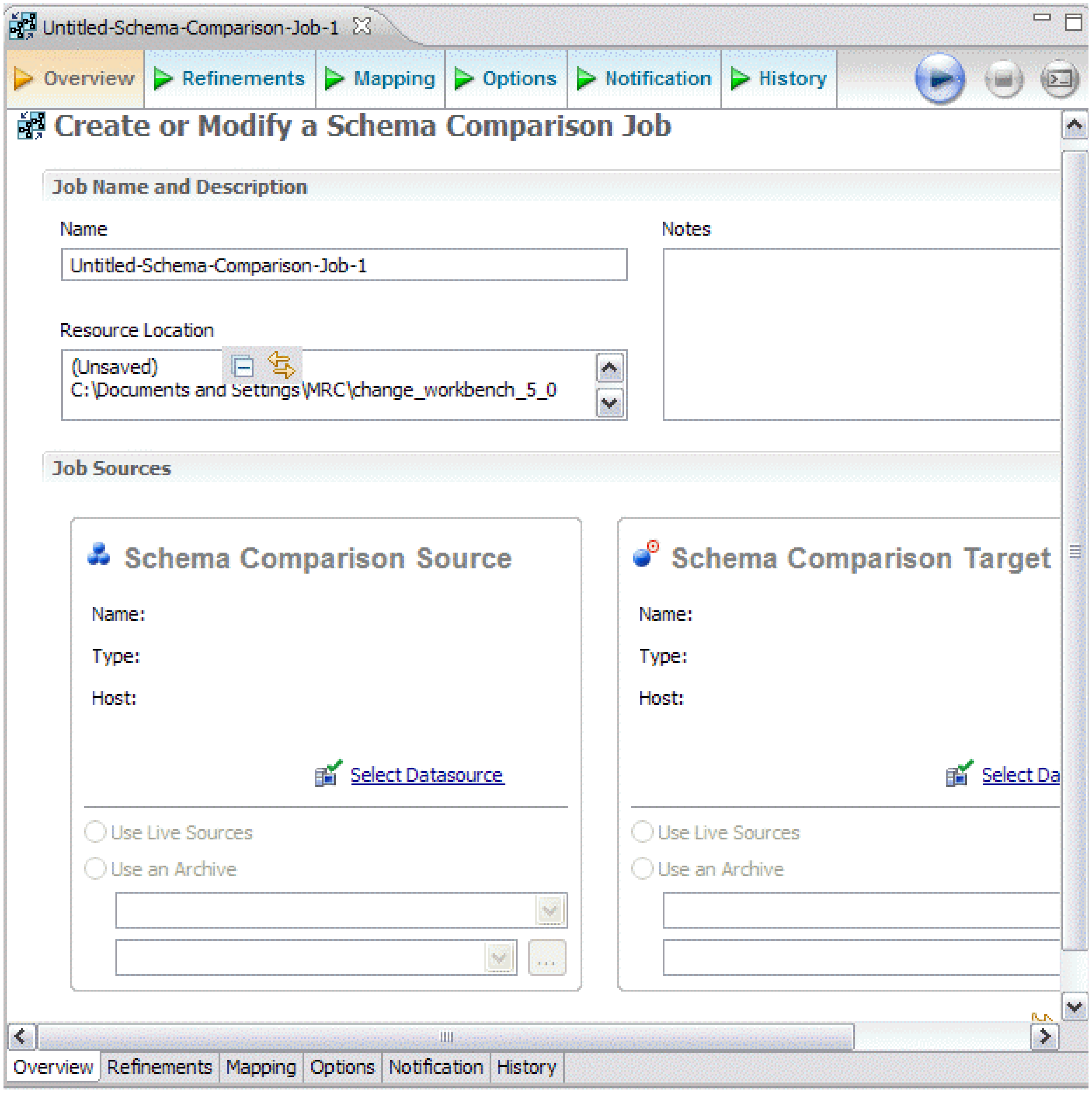
The following tasks provide a high-level overview of the schema comparison process:

- Create a New Schema Comparison Job
- Specify a Job Name
- Specify Source and Target Data Sources
- Refine Schema Comparisons
- Customize Schema Mapping
- Specify Job Options

- Specify Notification Options
- Execute the Schema Comparison Job
- Resolve Schema via a Generated SQL Script

Create a New Schema Comparison Job

Schema comparison jobs are defined in the **Schema Comparison Job Editor**.



Name: Untitled-Schema-Comparison-Job-1

Notes:

Resource Location: (Unsaved) C:\Documents and Settings\MRC\change_workbench_5_0

Schema Comparison Source — Name: Type: Host:

Schema Comparison Target — Name: Type: Host:

To start a new schema comparison job

- Select **File > New > Schema Comparison Job** in the Menu Bar.

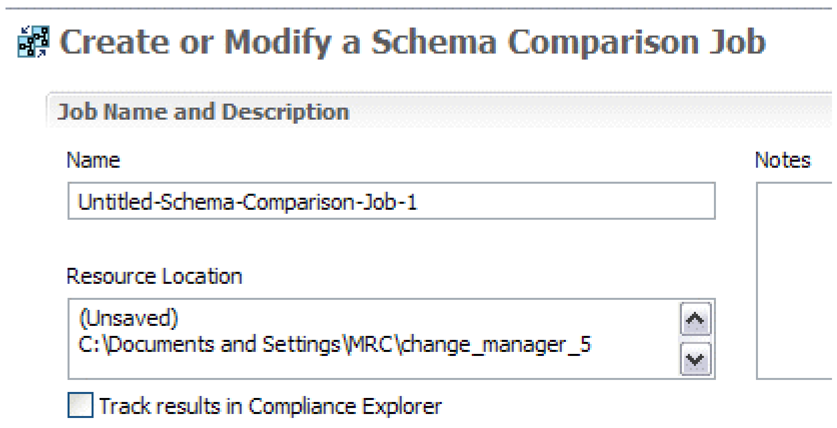
  The Schema Comparison Job Editor opens.

You can now proceed to define the parameters of the new job.

Specify a Job Name

A job name identifies the job in the application and should be specified with this in mind.

Specify a meaningful name that clearly identifies the job in the views and dialogs of the working environment.


Create or Modify a Schema Comparison Job
Job Name and Description
Name
Notes
Resource Location
Track results in Compliance Explorer

Untitled-Schema-Comparison-Job-1

(Unsaved)
C:\Documents and Settings\MRC\change_manager_5

To name a job

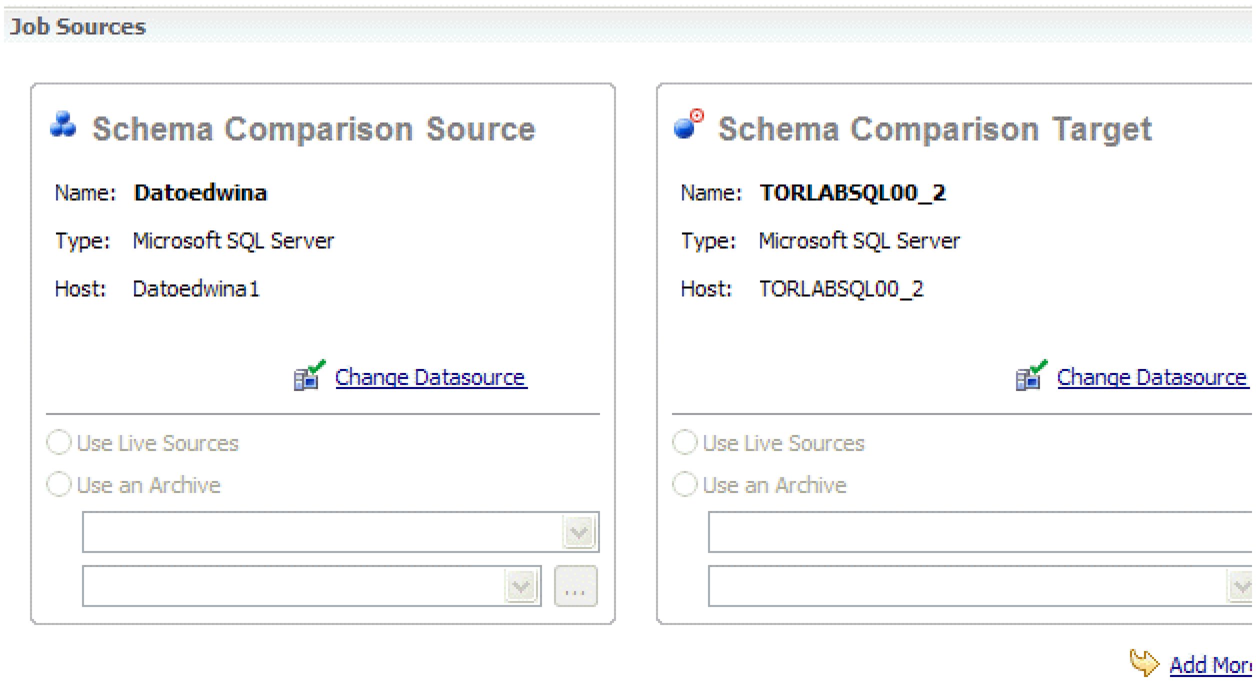
- Type the name of the job in the **Name** field of the Schema Comparison Editor.

  Ensure that you specify a meaningful name that identifies the job in other views and dialogs. You can save the job by selecting **File > Save** from the Menu Bar, or highlight the editor and press **Ctrl + S**.

  NOTE: Click **Track Results in Compliance Editor** to enable this job in the **Compliance** view.

Specify Source and Target Data Sources

The **Schema Comparison Source** and **Schema Comparison Target** boxes identify what data sources are compared when the job executes. Both boxes must contain the sources before the job can be executed. When a data source is added to a job, it is displayed by **Name**, **Type**, and **Host**.



Name: Datoedwina
Type: Microsoft SQL Server
Host: Datoedwina1

Name: TORLABSQL00_2
Type: Microsoft SQL Server
Host: TORLABSQL00_2

Additionally, Schema Archives can be used in schema comparison jobs in place of live data sources. See Creating a Schema Archive for more information on this process.

To add a data source to the job

- Click and drag a data source from **Data Source Explorer** to the Schema Comparison Source or Schema Comparison Target box, or click **Select Data Source** in the appropriate box and choose a data source from the dialog provided.

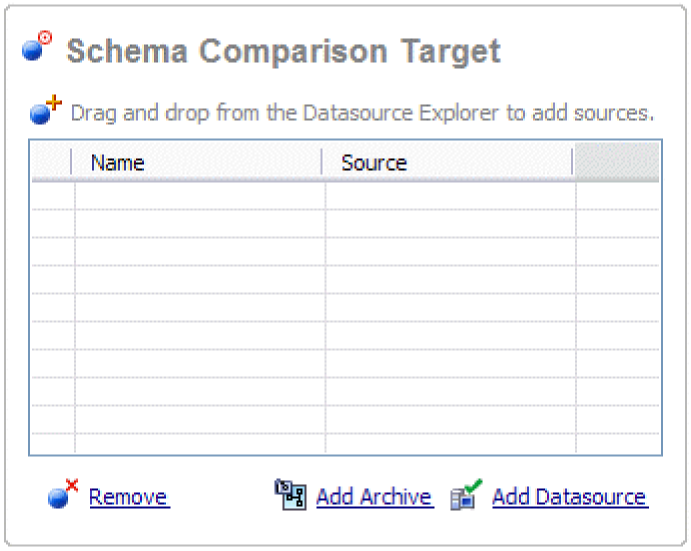
To add an archive to the job

- Click and drag a Schema Archive from **Data Source Explorer** to the **Schema Comparison Source** or **Schema Comparison Target** box.

When a data source is added to a job, it is displayed by **Name**, **Type**, and **Host**. You can change a data source by clicking **Select Data Source** in the appropriate box and selecting a new data source from the dialog, or by dragging a different data source to the box from data source Explorer.

The **Schema Comparison Source** and **Schema Comparison Target** boxes must contain data sources, Archives, or Standards before the job can be executed.

To add multiple Comparison Targets, click **Add More Targets** to enable a **One-to-Many** comparison.



To add multiple targets to the comparison job

- Drag and drop additional data sources or Archives from **Data Source Explorer** to the **Schema Comparison Target** box, or click the **Add Archive** or **Add Data Source** commands beneath the table and select the appropriate target from the dialog provided.

Additionally, you can specify whether you want to compare the current properties of the live data source, or an archived set of properties previously stored in Change Manager using the radio buttons beneath each box.

- Select **Use Live Sources** to compare the current properties of the data source when the job is executed.
- Select **Use an Archive** and select the Archive and version to use in the job.

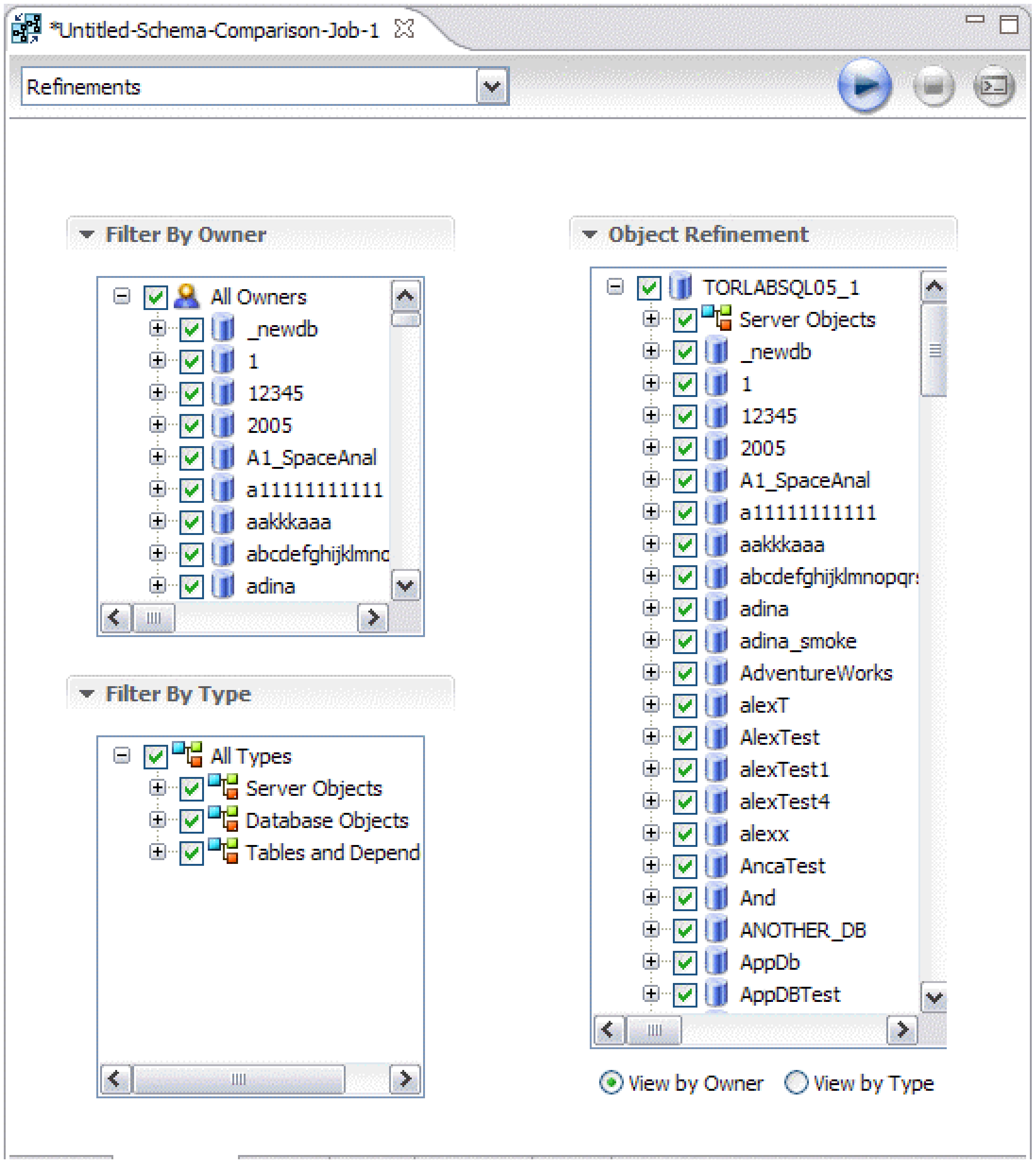
You can drag and drop data sources, Archive Jobs, Archive Versions, and Standards into the Comparison Source and Comparison Target cells.

Refine Schema Comparisons

Schema comparison refinement involves selecting the schema objects you want to include and exclude in the comparison process. All refinement is performed on the **Refinements** tab in the **Schema Comparison Editor**.

The **Object Refinement** window provides a view of the databases and corresponding objects of the comparison source. Use the check boxes beside each database or object to choose if they will be included or excluded in the comparison process. Objects are organized by tree and can be listed by owner or by object type.

The **Filter by Owner** and **Filter by Type** windows enable you to add or remove objects from the Object Refinement window by owner or by type, respectively.



To specify the properties used in the comparison job

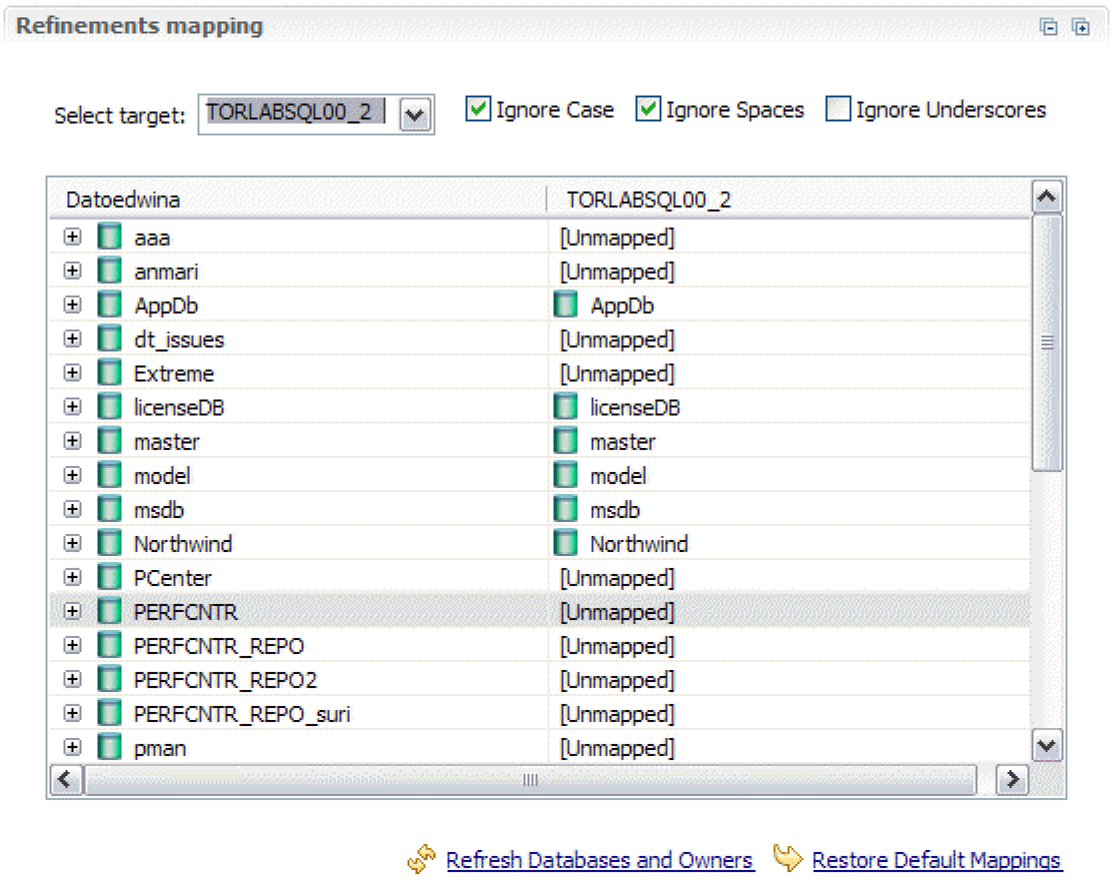
- Navigate to the Refinements tab and select or de-select the objects in the table, as required. Selected objects are included in the comparison job, and de-selected properties are ignored during job execution.

Customize Schema Mapping

Mapping is the process of pairing objects between the source and target of the job to pair high level containers so the objects inside can be compared. Mapping details are viewed and modified on the **Mapping** tab of the editor.

Change Manager automatically maps schema objects based on its interpretation of both schemas. Elements that do not match are excluded initially, unless manually paired prior to running the job.

The **Refinements Mapping** table provides a list of both schema elements for the source and targets of the job.



Select target: TORLABSQL00_2 — [✓ Ignore Case] [✓ Ignore Spaces] [ ] Ignore Underscores

Datoedwina	TORLABSQL00_2
aaa	[Unmapped]
anmari	[Unmapped]
AppDb	AppDb
dt_issues	[Unmapped]
Extreme	[Unmapped]
licenseDB	licenseDB
master	master
model	model
msdb	msdb
Northwind	Northwind
PCenter	[Unmapped]
PERFCNTR	[Unmapped]
PERFCNTR_REPO	[Unmapped]
PERFCNTR_REPO2	[Unmapped]
PERFCNTR_REPO_suri	[Unmapped]
pman	[Unmapped]

You can customize schema mapping parameters by choosing to redefine these pairings on an individual basis once the automatic mapping process is complete.

Object pairings are redefined via the drop down menu that appears when the corresponding object in the target column is selected. The menu contains a list of all schema objects that reside on the target data source.

To restore the schema mapping table to its original settings, click **Restore Default Mappings**. All settings revert to the original object mappings as determined by Change Manager.

To redefine object pairings

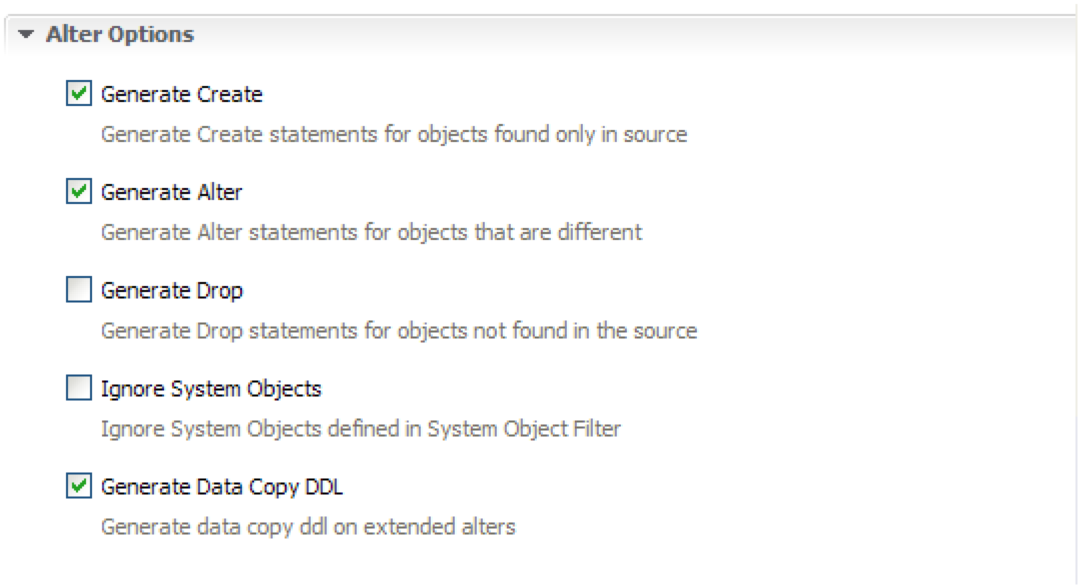
- Navigate to the **Mapping** tab and then click the target of the pair you want to change. You can use the drop down menu to redefine the schema object pairing as needed.

Specify Job Options

Prior to running a job, various options can be set to control aspects of the comparison process. These options are set prior to execution on the **Options** tab, and are grouped into categories: **Alter**, **Compare**, **Decorator**, and **Identifier**.

Options differ based on the DMBS of the source and target data sources. Additionally, they are affected by refined objects as well. For example, **Ignore Constraint Name Differences** won’t appear if constaints are excluded from the job during the filter process.

The **Alter Options** section contains parameters that affect how statements are created during the comparison process for the purpose of synchronizing the affected schema objects.


Alter Options

- [✓] Generate Create
  Generate Create statements for objects found only in source
- [✓] Generate Alter
  Generate Alter statements for objects that are different
- [ ] Generate Drop
  Generate Drop statements for objects not found in the source
- [ ] Ignore System Objects
  Ignore System Objects defined in System Object Filter
- [✓] Generate Data Copy DDL
  Generate data copy ddl on extended alters

SQL Server Alter Options

Option	Description
Generate Create	Indicates that the process will generate CREATE statements for objects found only in the source.
Generate Alter	Indicates that the process will generate ALTER statements for comparison objects that are different.
Generate Drop	Indicates that the process will generate DROP statements for objects that do not exist in the source.
Ignore System Objects	Indicates that the process ignores objects as defined by the System Object Filter.
Generate Data Copy DDL	Indicates that the process builds data copy DDL on extended ALTER statements.

Oracle Alter Options

Option	Description
Generate Create	Indicates that the process will generate CREATE statements for objects found only in the source.

Option	Description
Generate Alter	Indicates that the process will generate ALTER statements for comparison objects that are different.
Generate Drop	Indicates that the process will generate DROP statements for objects that do not exist in the source.
Generate Unused Statements	Indicates the process generates UNUSED statements to remove columns.
Ignore System Objects	Indicates that the process ignores objects as defined by the System Object Filter.
Generate Data Copy DDL	Indicates that the process builds data copy DDL on extended ALTER statements.

Sybase Alter Options

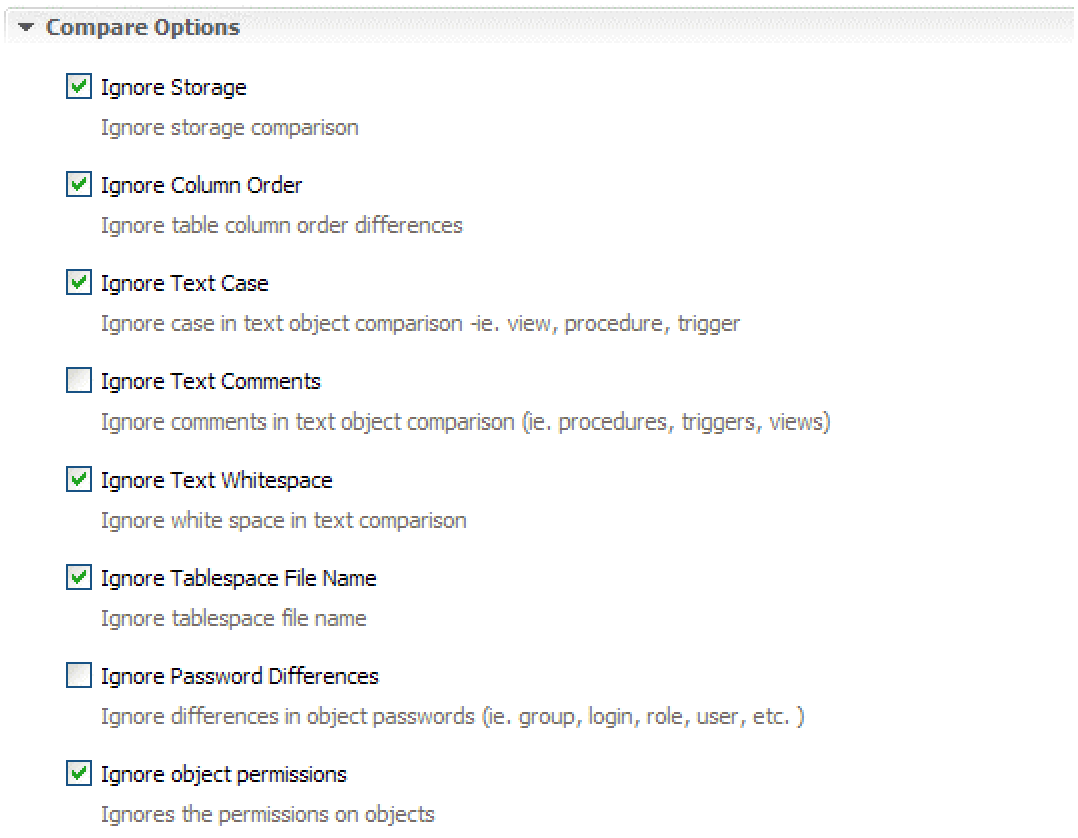
Option	Description
Generate Create	Indicates that the process will generate CREATE statements for objects found only in the source.
Generate Alter	Indicates that the process will generate ALTER statements for comparison objects that are different.
Generate Drop	Indicates that the process will generate DROP statements for objects that do not exist in the source.
Ignore System Objects	Indicates that the process ignores objects as defined by the System Object Filter.
Generate Data Copy DDL	Indicates that the process builds data copy DDL on extended ALTER statements.

IBM DB2 LUW Alter Options

Option	Description
Generate Create	Indicates that the process will generate CREATE statements for objects found only in the source.
Generate Alter	Indicates that the process will generate ALTER statements for comparison objects that are different.
Generate Drop	Indicates that the process will generate DROP statements for objects that do not exist in the source.

Option	Description
Ignore System Objects	Indicates that the process ignores objects as defined by the System Object Filter.
Generate Data Copy DDL	Indicates that the process builds data copy DDL on extended ALTER statements.

The **Compare Options** section provides settings that determine what elements of the source and target of a job will be omitted when running a schema comparison job.



- [✓] Ignore Storage
  Ignore storage comparison
- [✓] Ignore Column Order
  Ignore table column order differences
- [✓] Ignore Text Case
  Ignore case in text object comparison -ie. view, procedure, trigger
- [ ] Ignore Text Comments
  Ignore comments in text object comparison (ie. procedures, triggers, views)
- [✓] Ignore Text Whitespace
  Ignore white space in text comparison
- [✓] Ignore Tablespace File Name
  Ignore tablespace file name
- [ ] Ignore Password Differences
  Ignore differences in object passwords (ie. group, login, role, user, etc.)
- [✓] Ignore object permissions
  Ignores the permissions on objects

SQL Server Compare Options

Option	Description
Ignore Storage	Ignores storage comparison when processing the job.
Ignore Column Order	Ignores column order when processing the job.
Ignore Text Case	Ignores text case when processing the job.

Option	Description
Ignore Text Comments	Ignores text comments when processing the job.
Ignore Text Whitespace	Ignores text whitespace when processing the job.
Ignore Tablespace File Name	Ignores the tablespace file name when processing the job on DB2 and Oracle platforms. If this option is selected for SQL platforms, it ignores database names. If this option is selected for Sybase platforms, it ignores database device names.
Ignore Password Differences	Ignores password differences when processing the job.
Ignore Object Permissions	Ignores object permissions when processing the job.

Oracle Compare Options

Option	Description
Ignore Storage	Ignores storage comparison when processing the job.
Ignore Partition Differences	Ignores partition differences when processing the job.
Ignore Column Order	Ignores table column order differences when processing the job.
Ignore Check Constraints Quotes	Ignores quotes in check constraints when processing the job.
Ignore Table Comment	Ignores table comments when processing the job.
Ignore Text Case	Ignores syntax case in text object comparisons when processing the job. (views, procedures, trigger, etc.)
Ignore Text Comments	Ignores comments in text objects when processing the job.
Ignore Text Whitespace	Ignores white space when processing the job.
Ignore Tablespace File Name	Ignores tablespace file names when processing the job.
Ignore Password Differences	Ignores differences in object passwords when processing the job. (group, login, role, user, etc.)
Ignore Tablespace	Ignores tablespaces for table, index, cluster, etc.
Ignore Object Permissions	Ignores the permissions on objects when processing a job.

Sybase Compare Options

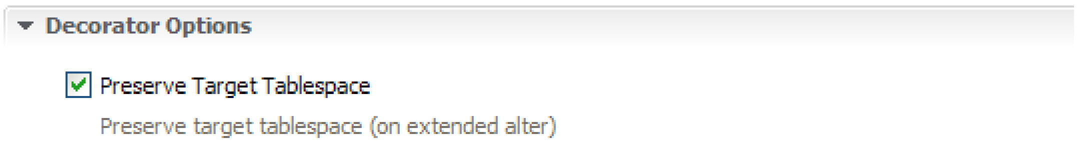
Option	Description
Ignore Storage	Ignores storage comparison when processing the job.
Ignore Column Order	Ignores table column order differences when processing the job.
Ignore Check Constraint Quotes	Ignores quotes in check constraints when processing the job.
Ignore Text Case	Ignores syntax case in text object comparisons when processing the job. (views, procedures, trigger, etc.)
Ignore Text Comments	Ignores comments in text objects when processing the job.
Ignore Text Whitespace	Ignores white space when processing the job.
Ignore Tablespace File Name	Ignores tablespace file names when processing the job.
Ignore Password Differences	Ignores differences in object passwords when processing the job. (group, login, role, user, etc.)
Ignore Object Permissions	Ignores the permissions on objects when processing a job.

IBM DB2 LUW Compare Options

Option	Description
Ignore Storage	Ignores storage comparison when processing the job.
Ignore Column Order	Ignores table column order differences when processing the job.
Ignore Check Constraint Quotes	Ignores quotes in check constraints when processing the job.
Ignore Table Comment	Ignores table comments when processing the job.
Ignore Data Capture	
Ignore Text Case	Ignores syntax case in text object comparisons when processing the job. (views, procedures, trigger, etc.)
Ignore Text Comments	Ignores comments in text objects when processing the job.
Ignore Text Whitespace	Ignores white space when processing the job.
Ignore Tablespace File Name	Ignores tablespace file names when processing the job.

Option	Description
Ignore Password Differences	Ignores differences in object passwords when processing the job. (group, login, role, user, etc.)
Ignore Tablespace	Ignores tablespaces for table, index, cluster, etc.
Ignore Object Permissions	Ignores the permissions on objects when processing a job.

The **Decorator Options** panel contains parameters that determine what elements of the source and target data source will be preserved during a schema comparison.



✓ Preserve Target Tablespace

Preserve target tablespace (on extended alter)

SQL Server Decorator Options

Option	Description
Preserve Target Tablespace	Maintains the target tablespace on an extended ALTER statement.

Oracle Decorator Options

Option	Description
Preserve Target Storage	Maintains object storage of the target on an extended ALTER statement.
Preserve Target Tablespace	Maintains the target tablespace on an extended ALTER statement.
Recompile	Recompiles dependent procedures after an extended ALTER statement.
Comment Out Extended Alters for Tablespaces	Comments out extended ALTER statements for tablespaces.

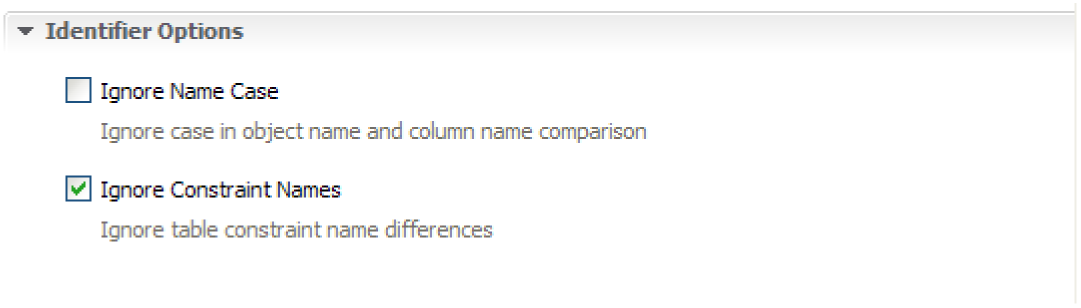
Sybase Decorator Options

Option	Description
Preserve Target Tablespace	Maintains the target tablespace on an extended ALTER statement.

IBM DB2 LUW Decorator Options

Option	Description
Preserve Target Tablespace	Maintains the target tablespace on an extended ALTER statement.
Comment Out Extended Alters for Tablespaces	Comments out extended ALTER statements for tablespaces.

The **Identifier Options** panel contains parameters that determine what elements of the source and target data source will be ignored during the schema comparison process.



[ ] Ignore Name Case
Ignore case in object name and column name comparison

[✓] Ignore Constraint Names
Ignore table constraint name differences

SQL Server Identifier Options

Option	Description
Ignore Name Case	Ignores the syntax case when performing object name and column name comparisons.
Ignore Constraint Names	Ignores table constraint name differences.

Oracle Identifier Options

Option	Description
Ignore Name Case	Ignores the syntax case when performing object name and column name comparisons.
Ignore Constraint Names	Ignores table constraint name differences.

Sybase Identifier Options

Option	Description
Ignore Name Case	Ignores the syntax case when performing object name and column name comparisons.

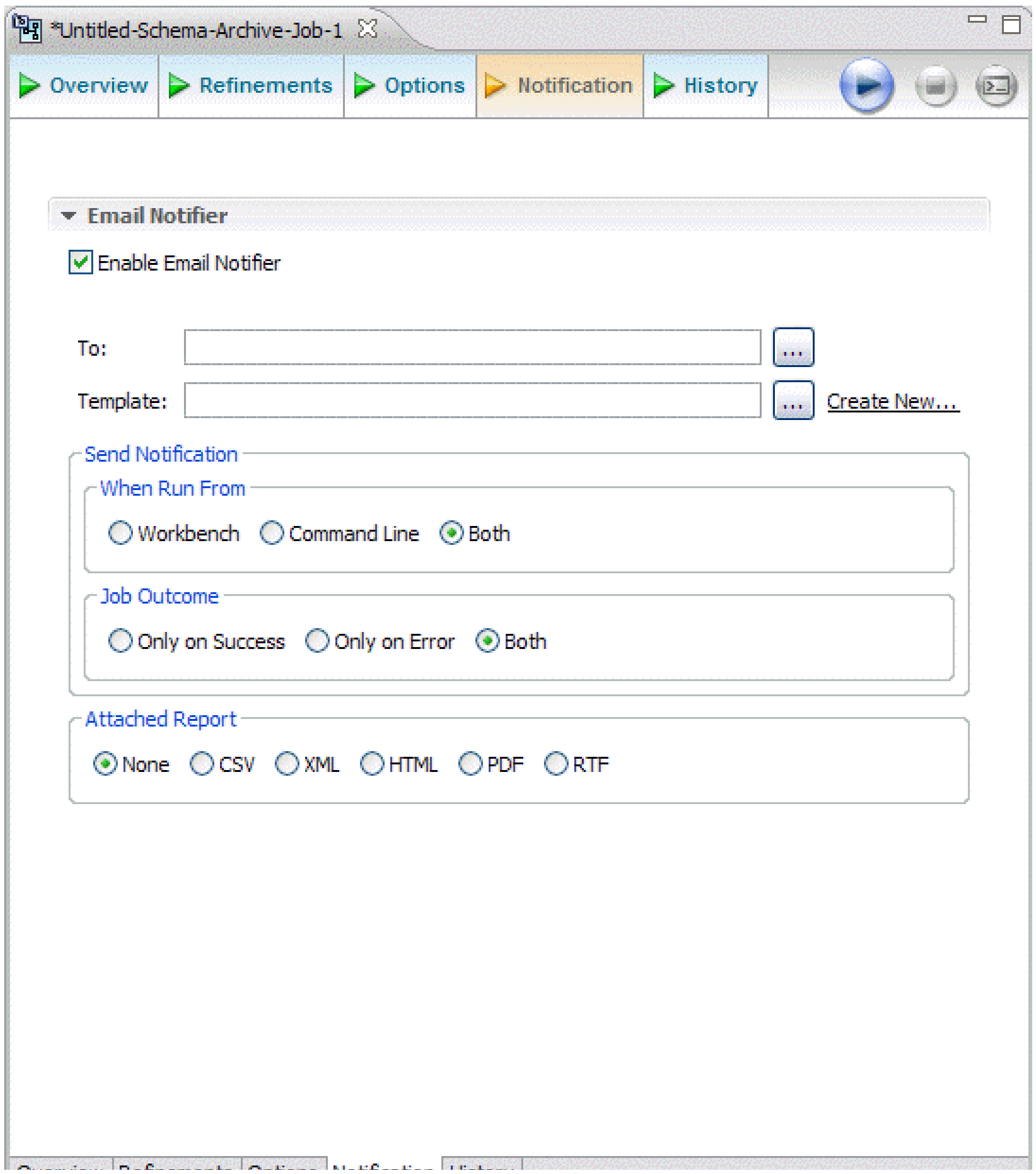
Option	Description
Ignore Constraint Names	Ignores table constraint name differences.

IBM DB2 LUW Identifier Options

Option	Description
Ignore Name Case	Ignores the syntax case when performing object name and column name comparisons.
Ignore Constraint Names	Ignores table constraint name differences.
Ignore Specific Name	Ignores SQL specific names in procedures, functions, and methods.

Specify Notification Options

Notification options provide parameters for enabling and setting up email notification and report mailing for executed schema comparison jobs. The notification options tab enables you to specify the email addresses to which you want to send notifications, the email message format, and the conditions under which a notification should be sent.



Email Notifier

[✓] Enable Email Notifier

To:

Template: Create New...

Send Notification

When Run From: Workbench, Command Line, [✓ Both]

Job Outcome: Only on Success, Only on Error, [✓ Both]

Attached Report: [✓ None], CSV, XML, HTML, PDF, RTF

To enable notification, select the **Enable Email Notifier** check box and select the options, as available. Enter the email addresses to which the notification will be sent in the **To** field, and specify an email format template in the **Template Field**.

In the **Send Notification** options, specify when the email notification should be sent:

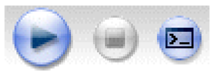
- Under the **When Run From** options, indicate that email notifications should be sent when the job is executed from the **Workbench**, **Command Line**, or **Both**.
- Under the **Job Outcome** options**,** indicate that email notification should be sent **Only on Success**, **Only on Error**, or **Both**.
- Under the **For Threshold** options, specify the threshold level for which the email notification will be sent, based on the job results. Choose **Pass**, **Warn**, or **Fail**. You can click **Open Compliance Page** to specify the percentage values for each threshold warning, which are based on the percent match results of the job.

You can also specify the file format of an attached report that displays the job results with the notification email. Choose **None** (does not attach a report with the notification email), **CSV**, **XML**, **HTML**, **PDF**, or **RTF** to specify the file format of the report.

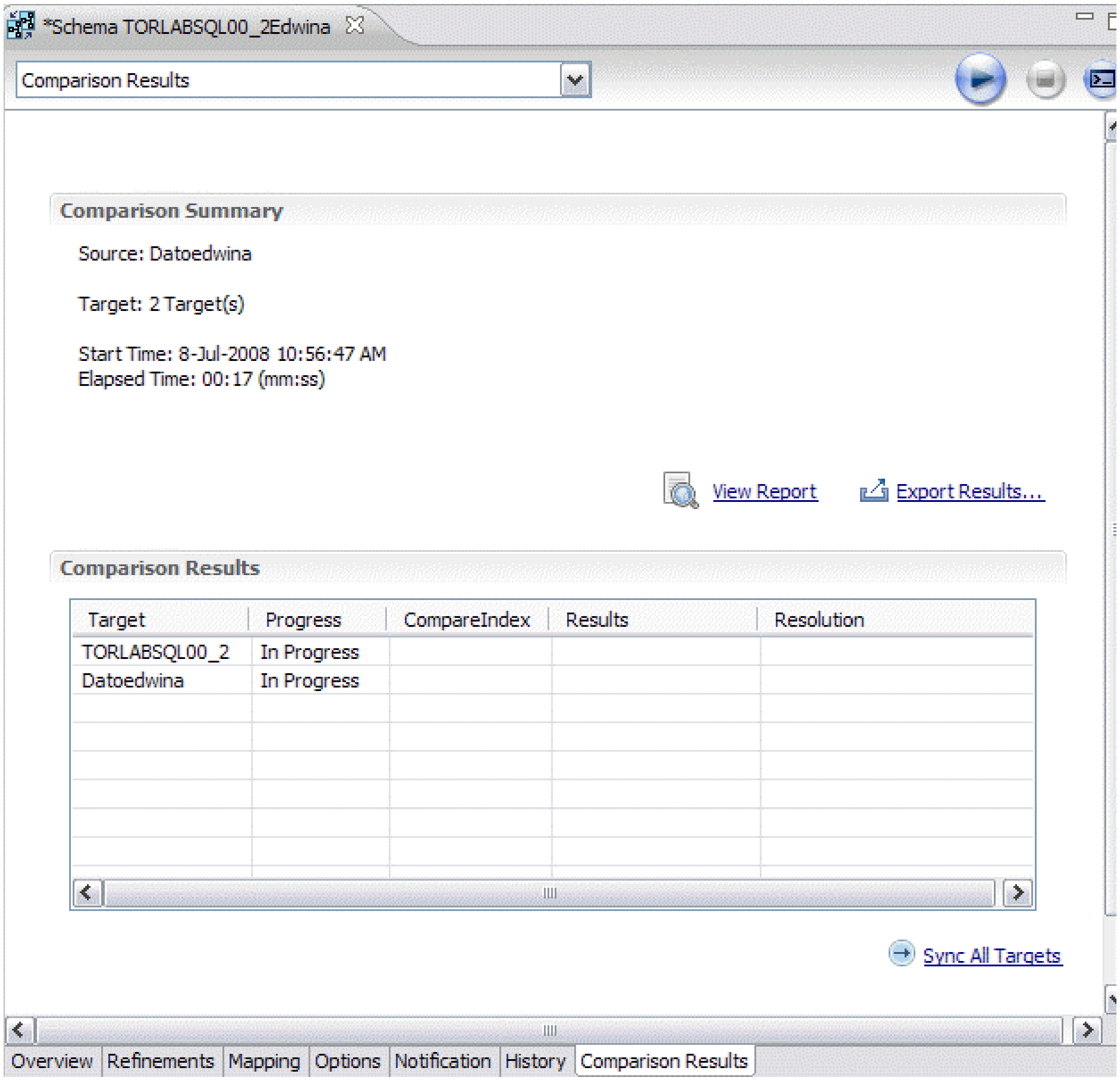
NOTE: Notification options can be modified via a plug-in extension that enables custom notification coding. See Creating and Implementing a Custom Configuration Plug-In for more information.

Execute the Schema Comparison Job

The comparison job runs when you click the **Run** icon located in the upper right-hand corner of the Schema Comparison Editor.



When the process runs, the **Comparison Results** tab appears and indicates the progress of the job.



Comparison Summary

Source: Datoedwina

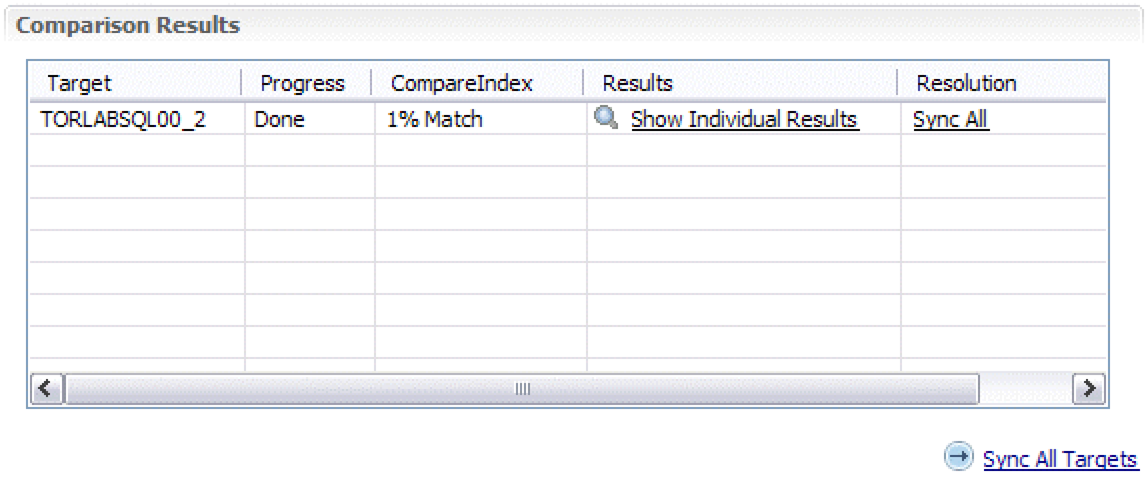
Target: 2 Target(s)

Start Time: 8-Jul-2008 10:56:47 AM
Elapsed Time: 00:17 (mm:ss)

Comparison Results

Target	Progress	CompareIndex	Results	Resolution
TORLABSQL00_2	In Progress			
Datoedwina	In Progress			

When the job is processed, the schema pairs are displayed along with the Compare Index, Results, and Resolution values of the job in the **Comparison Results** table.



Comparison Results

Target	Progress	CompareIndex	Results	Resolution
TORLABSQL00_2	Done	1% Match	Show Individual Results	Sync All

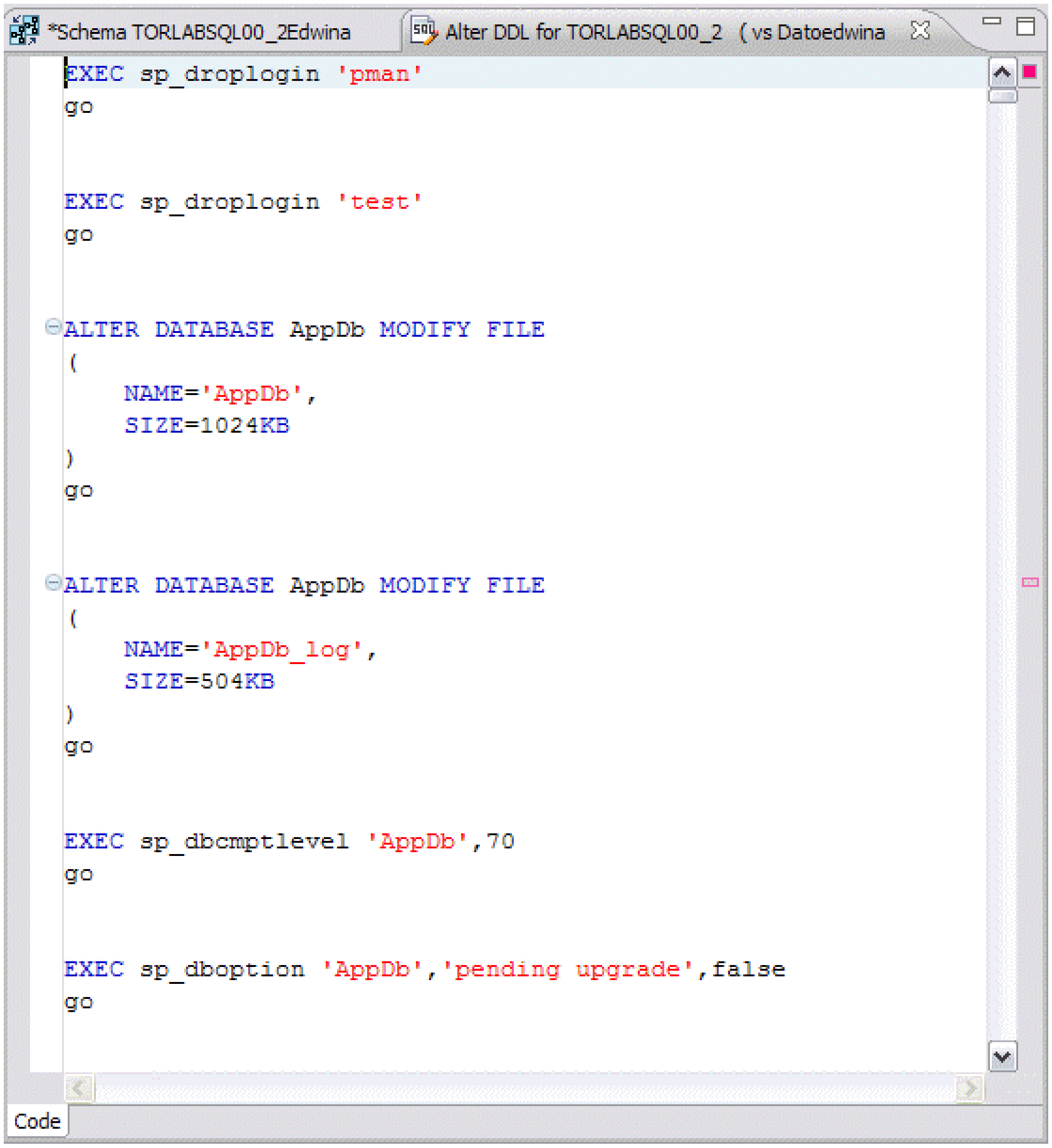
- The **Compare Index** column indicates the percentage of schema that matched between the source and targets of the job.
- The **Results** column provides a link that enables you to drill down and view the results between the pairs matched in the job. Click **Show Individual Results** to open the **Individual Comparison Results** tab, which provides a more detailed summarization of the schema comparison.
- The **Resolution** column provides a command that produces a synchronization script that, when executed, will automatically synchronizes the compared schema. For more information on generating a synchronization script, see Resolve Schema via a Generated SQL Script.

Resolve Schema via a Generated SQL Script

Once a schema comparison job has completed, synchronization commands are used to even the data so it matches. The synchronization process automates generation of CREATE, DROP, ALTER statements, and stored procedures on the source and target data sources, as required.

On the **Results** tab, click **Sync All** in the Resolution column of the Comparison Results table. A synchronizing script is produced in another window.

The script can be modified manually, executed as needed, and saved at any time.


*Schema TORLABSQL00_2Edwina
Alter DDL for TORLABSQL00_2 (vs Datoedwina
Code

```
EXEC sp_droplogin 'pman'
go


EXEC sp_droplogin 'test'
go


ALTER DATABASE AppDb MODIFY FILE
(
    NAME='AppDb',
    SIZE=1024KB
)
go


ALTER DATABASE AppDb MODIFY FILE
(
    NAME='AppDb_log',
    SIZE=504KB
)
go


EXEC sp_dbcmptlevel 'AppDb',70
go


EXEC sp_dboption 'AppDb','pending upgrade',false
go
```

Executing a SQL Synchronization Script

Files can be launched from within the Change Manager development environment for execution on a registered data source Files are executed via the commands in the **Run** menu.

When a SQL file is open in the Workspace, select it and choose a database and an associated catalog on which you want to execute file using the drop down menus in the Toolbar. You can click the execute icon to execute the code on the specified database and catalog, start a transaction or commit a transaction, or modify the SQL session options prior to execution.

NOTE: The synchronization script generation process does not support all object parameters. The following object parameters should be manually added after a synchronization script is generated, as required: **fillfactor**, **consumers**, **allow_dup_row**, and **statistics**.

To execute code

Open the SQL file you want to run, ensure it is associated with the correct database and click the **Execute** icon. Change Manager executes the code on the data source you specified. Results are displayed in the same tab or in a new tab.

To execute a transaction

Open the transaction file you want to run and ensure it is associated with the correct database, and then click the **Start Transaction** icon. Change Manager executes the transaction on the data source you specified.

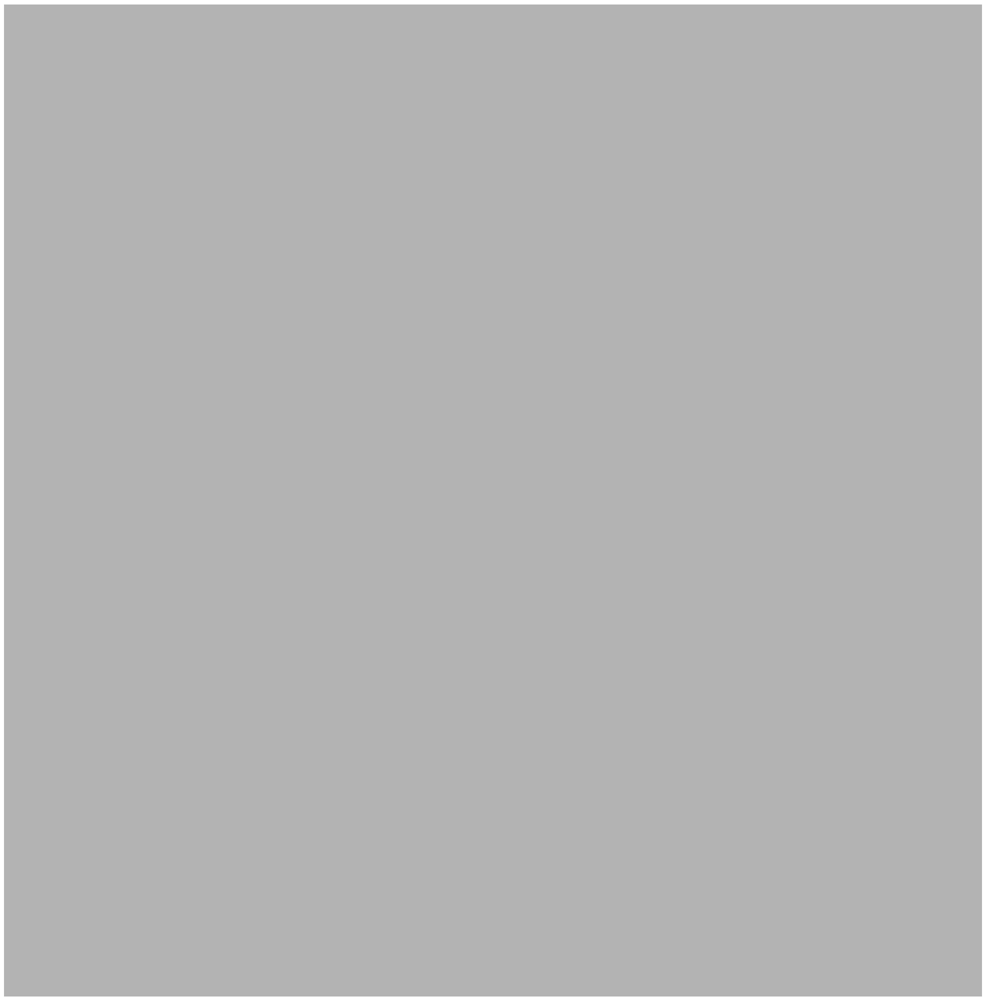
To commit a transaction

Open the transaction file you want to commit, ensure it is associated with the correct database, and then click the **Commit Transaction** icon. Change Manager commits the transaction on the data source your specified.

You can set transactions to auto-commit prior to execution on the **SQL Execution** node of the **Preferences** panel in Change Manager.

Configure a SQL Session

The SQL Session Options dialog provides configuration parameters that indicate to Change Manager how to execute code in the development environment.



To modify SQL session options:

1. Click the SQL Session Options icon in the Toolbar.

   The **SQL Session Options** dialog appears.

2. Click on individual parameters in the **Value** column to change the configuration of each property, as specified.

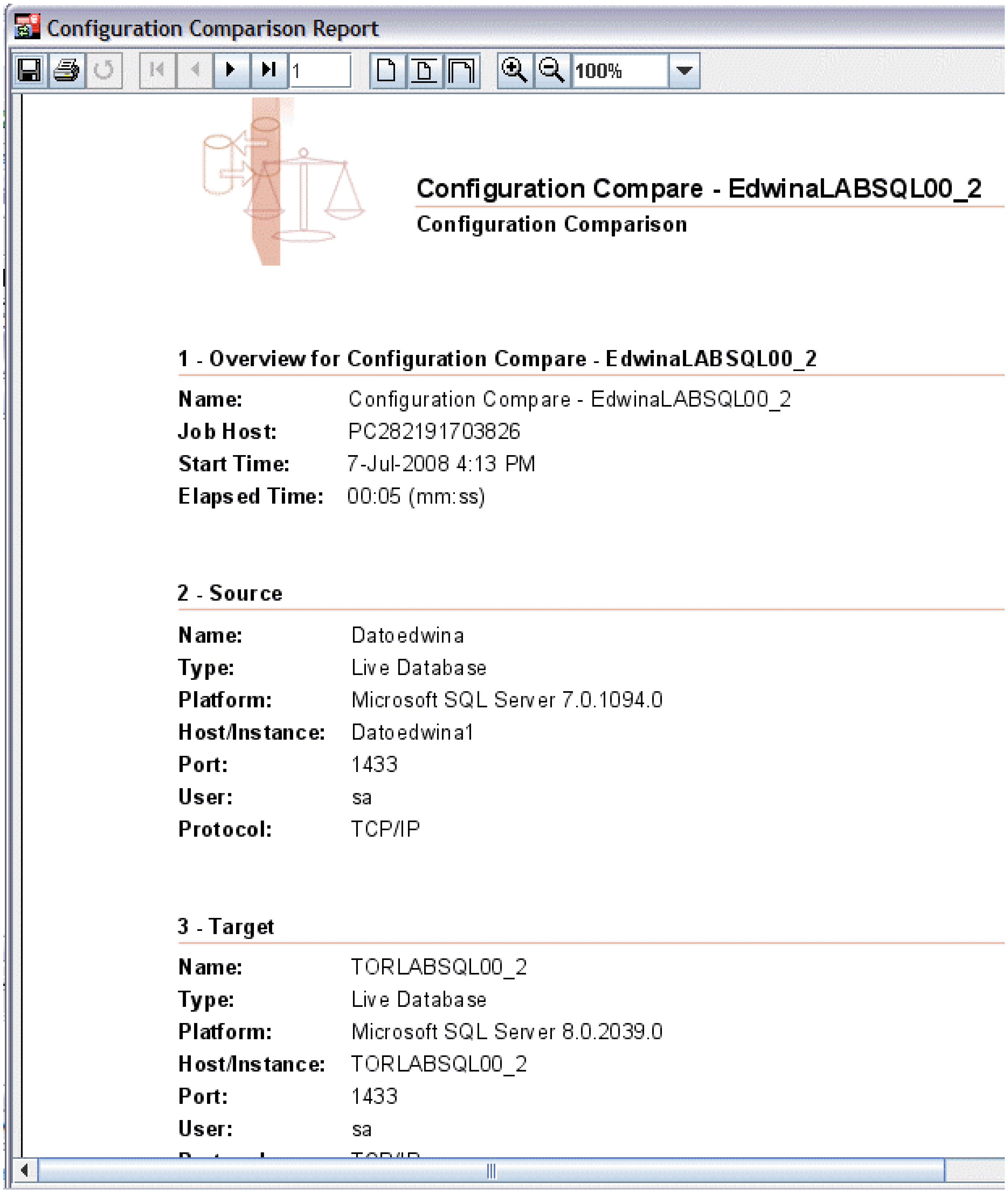
3. Click **Finish**.

   The session options will be changed and Change Manager will execute the code as specified when you launch the script.

   NOTE: Session options only apply to the corresponding editor and are not retained when executing multiple SQL files.

Generating a Job Report

Once you have executed a comparison job, you can generate a job report via the **View Report** command on the **Results** tab.



Configuration Compare - EdwinaLABSQL00_2

Configuration Comparison

1 - Overview for Configuration Compare - EdwinaLABSQL00_2

Name:	Configuration Compare - EdwinaLABSQL00_2
Job Host:	PC282191703826
Start Time:	7-Jul-2008 4:13 PM
Elapsed Time:	00:05 (mm:ss)

2 - Source

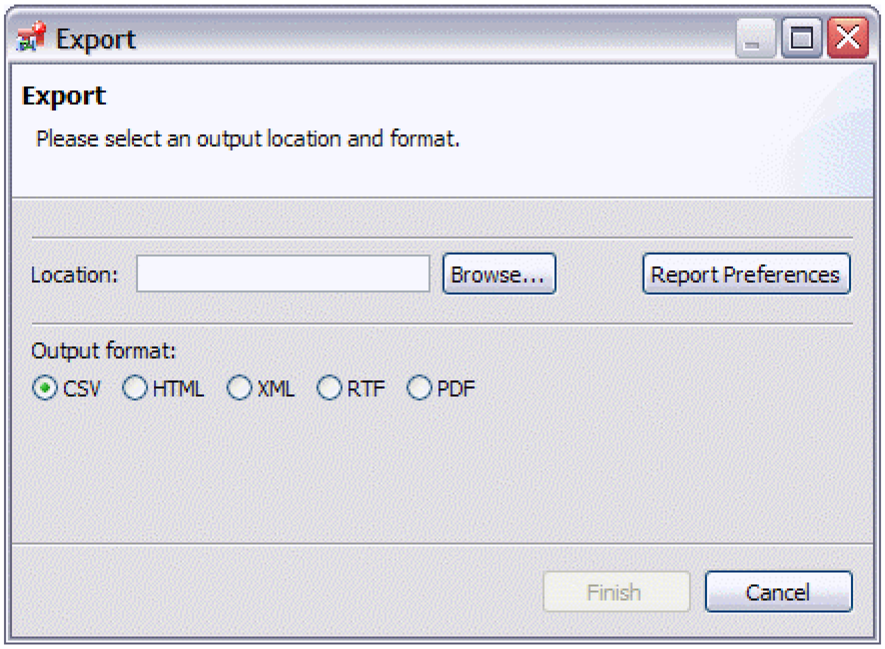
Name:	Datoedwina
Type:	Live Database
Platform:	Microsoft SQL Server 7.0.1094.0
Host/Instance:	Datoedwina1
Port:	1433
User:	sa
Protocol:	TCP/IP

3 - Target

Name:	TORLABSQL00_2
Type:	Live Database
Platform:	Microsoft SQL Server 8.0.2039.0
Host/Instance:	TORLABSQL00_2
Port:	1433
User:	sa

The report opens in a separate window and can be read on screen, sent to a printer, and saved to disk.

Additionally, the **Export Results** command will produce a report file in a specified format at a directory location of your choosing.


Export

Export

Please select an output location and format.

Location:

Output format: [✓ CSV] HTML XML RTF PDF

Click **Export Results** on the **Results** tab and fill in the fields, as required. When you click **Finish**, the report is created in the directory specified in the **Location** field, in the format specified in the **Output Format** section.

Creating a Schema Archive

An archive job is an object related to schema comparison jobs, in that it contains a snapshot of the schema of a registered data source. A Schema Archive is used in schema comparisons against other data sources and archive jobs as a baseline to monitor data loss and inconsistencies when moving between testing and production environments.

All archive jobs are composed of a name, description, and schema derived from a single live data source at a specific point in time. Additionally, archive jobs have versioning functionality and can be periodically updated with the current values of an associated data source. Older versions are retained within an Archive and can be recalled as needed.

Archive jobs are located in Data Source Explorer on the Schema Archive Jobs node of the corresponding data source.

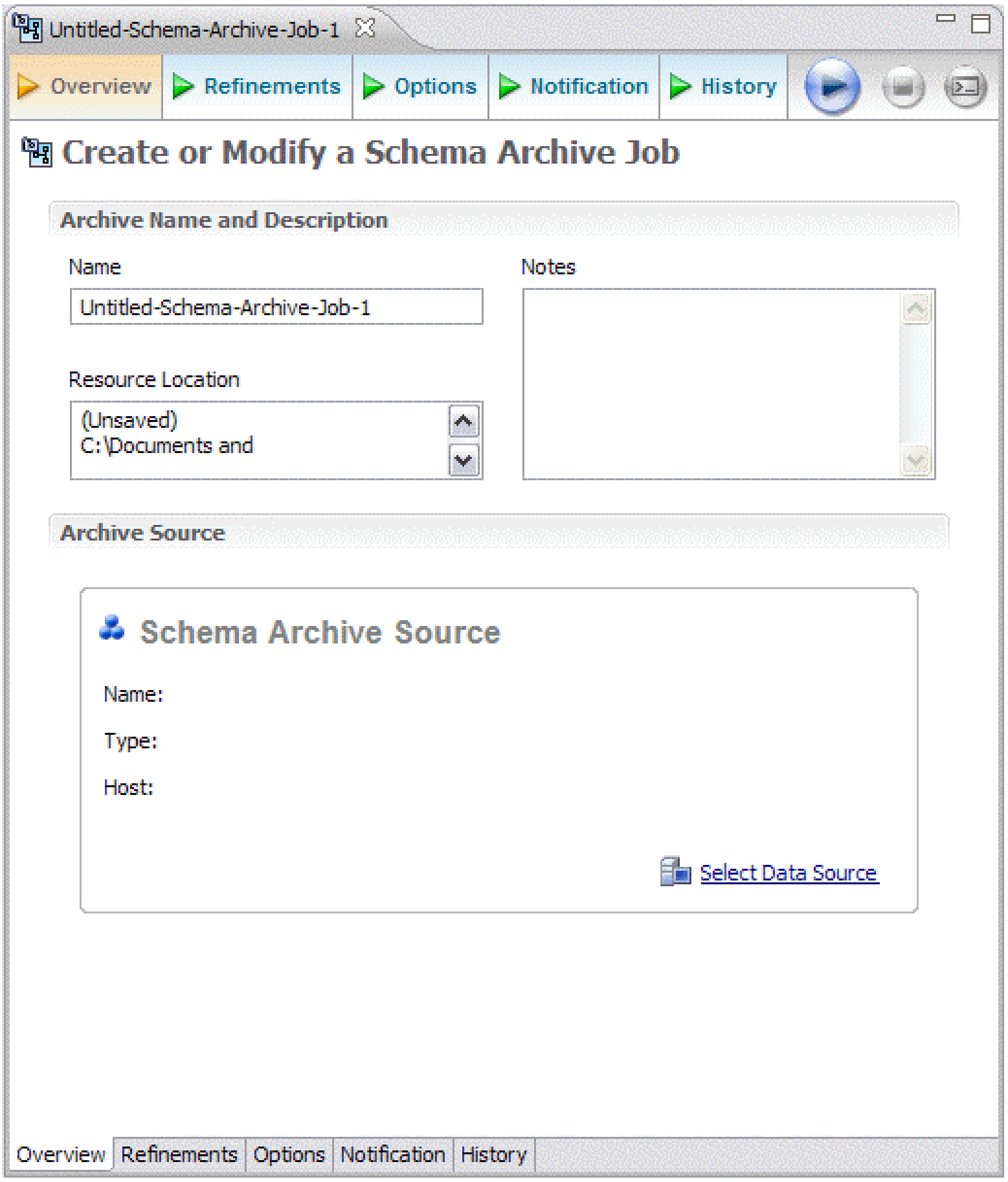
The following tasks provide a high-level overview of the Archive definition process:

- Create a New Configuration Archive
- Specify an Archive Name
- Specify an Archive Job Source
- Refine Properties
- Specify Extraction Options
- Specify Notification Options
- Save the Archive Job/Manage Versioning
- Compare Archive Contents and Extract DDL
- Generating a Job Report

- Use Archive in Schema Comparison Jobs

Create a New Configuration Archive

Archives are defined in the **Schema Archive Job Editor**.



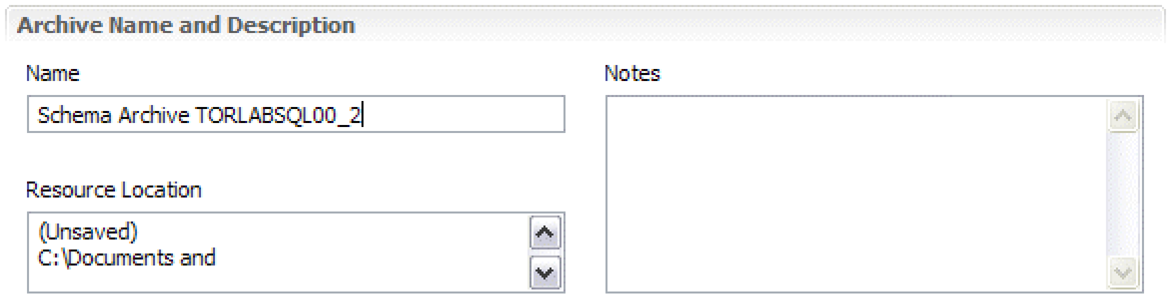
When an archive is initially created it is in an unsaved state and versionless. Once you define the archive job and refine its properties, you need to issue the **Run Job** command to create a version and save the archive job in the schema comparison environment. Unsaved Archives cannot be used in comparison jobs and do not appear in Data Source Explorer. More information regarding the process of refining and saving a new archive job are located in the other steps of this section.

To create a Schema Archive Job

- Select **File > New > Schema Archive Job** from the menu. The Schema Archive Job Editor opens.

Specify an Archive Name

A name must be defined when a new archive job is created. The name is used to distinguish between archive jobs and other objects in Change Manager, and is used in versioning functionality later in the process.



Name: Schema Archive TORLABSQL00_2

Resource Location: (Unsaved) C:\Documents and

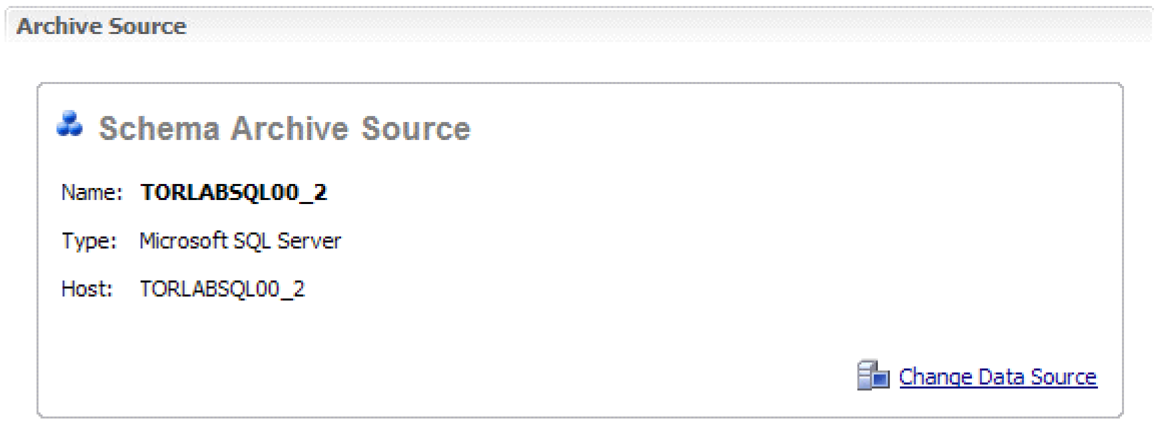
Ensure you specify a meaningful name that identifies the archive job in other views and dialogs later.

To name a Schema Archive Job

- Enter the name of the archive job in the **Name** field of the Archive Job Editor.

Specify an Archive Job Source

The **Schema Archive Source** table specifies the data source from which the archive job is derived.



Name: **TORLABSQL00_2**

Type: Microsoft SQL Server

Host: TORLABSQL00_2

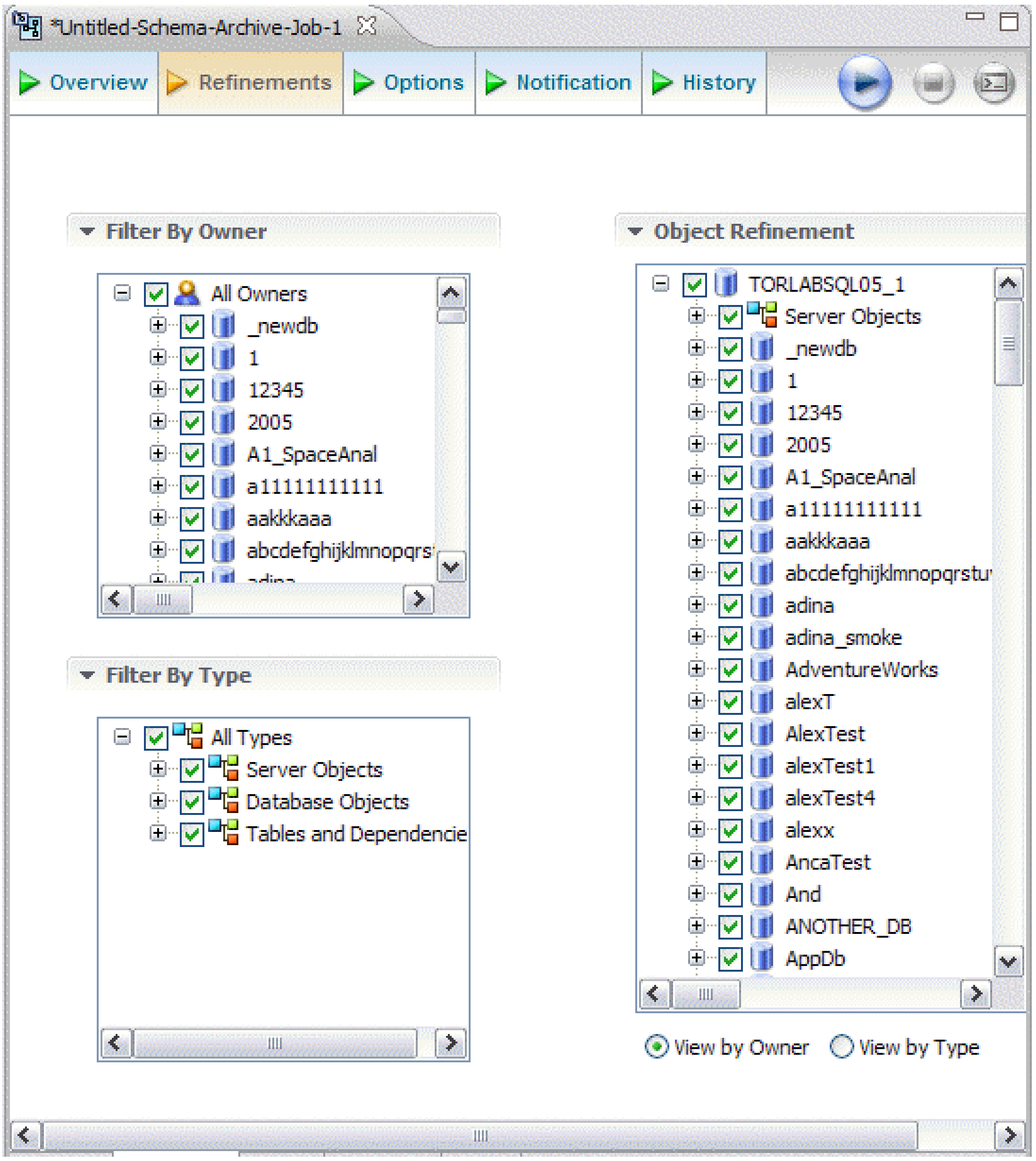
To specify the archive job source

- Drag and drop the data source to be archived from **Data Source Explorer** to the **Schema Archive Source** box. The data source is listed by name, DBMS type, and host.

Refine Properties

By default, when you create a new archive job, all schema is included. However, you may only want or need to archive a specific group of objects per archive defined. Other archive jobs can be created (and likewise refined) for the same data source, but can maintain a different group of selected schema objects.

Archive job properties are located on the **Refinements** tab of the **Schema Archive Job Editor**, where they are listed by object name, owner, and type.



To refine schema archive job properties

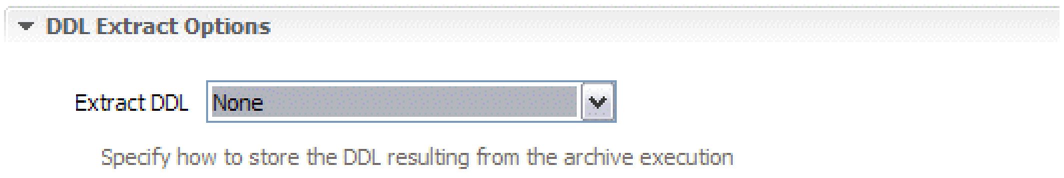
- Use the check boxes to select or de-select schema objects as needed. Selected objects are included in the archive job when comparison jobs are run, while de-selected properties are ignored altogether.

When you have finished selecting the properties to include in the archive job, you will need to save the Archive by creating a version (see Save the Archive Job/Manage Versioning). Within the same job, refinements are retained across versions unless new refinements are specified. However, each time you create a new archive job, Change Manager includes all schema of the specified data source and you must again specify refinements, if any.

Specify Extraction Options

The **Options** tab contains the **DDL Extraction Options** setting, which enables you to define how DDL is extracted and stored in Change Manager during the archive process. By default, this option is set to **None**, which indicates that the process does not retain extracted DDL when building the schema archive, although it can be extracted from the archive after the job has been executed.

If you choose to store extracted DDL, the directory location of the saved files can be viewed on the **History** tab of the editor when archive versions are created. Files can also be viewed from **Project Explorer** and the **Navigator** under the appropriate nodes. For more information on creating archive versions, see Save the Archive Job/Manage Versioning.



Extract DDL: None

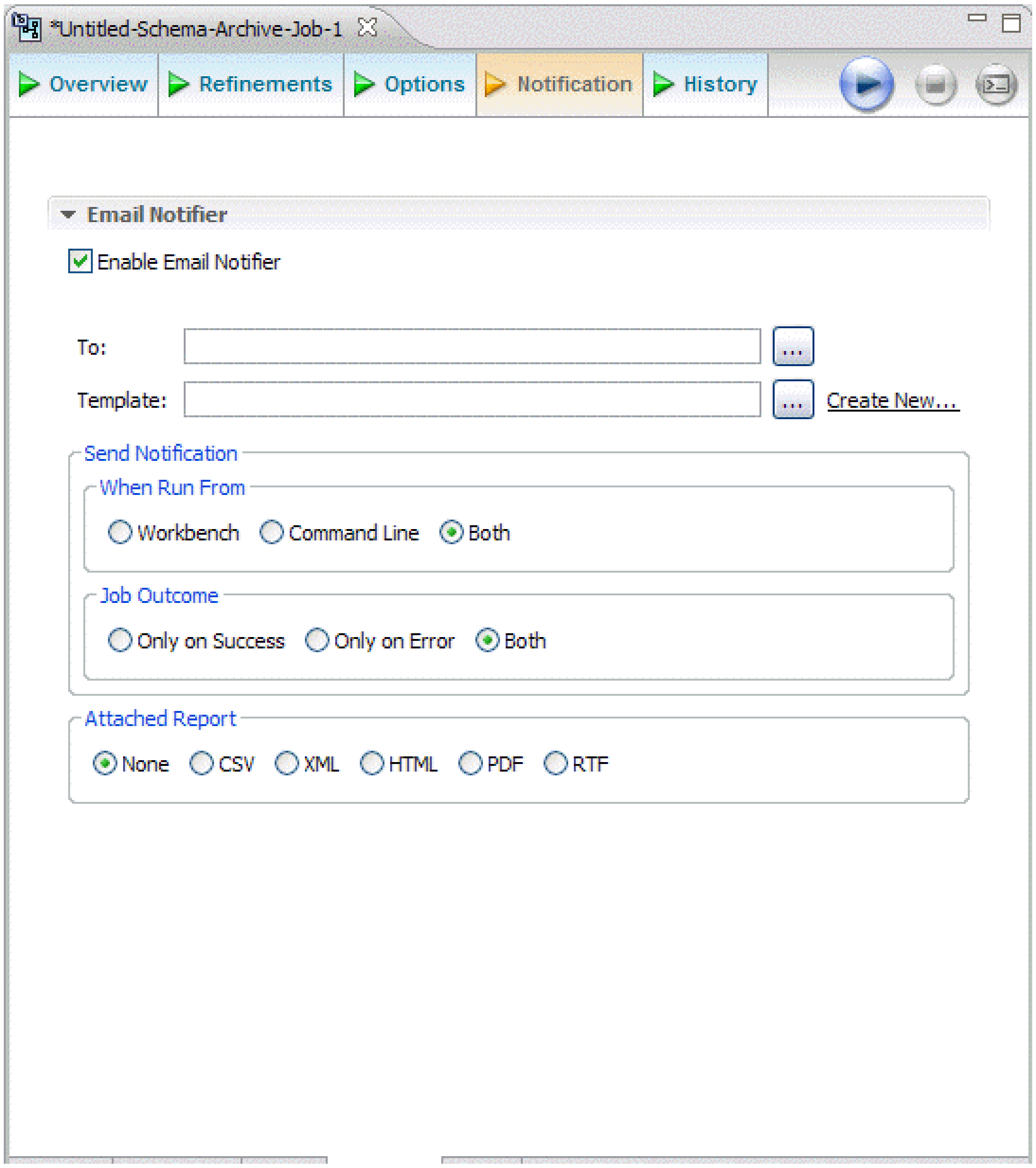
Specify how to store the DDL resulting from the archive execution

To define the DDL Extraction process

- Use the drop down menu to specify **None**, **One File**, or **Multiple Files** to indicate how DDL extraction occurs, if at all, during the archive execution process.

Specify Notification Options

Notification options provide parameters for setting up email notification and report mailing for executed schema archive jobs. The notification options tab enables you to specify the email addresses to which you want to send notifications, the email message format, and the conditions under which a notification should be sent.



Email Notifier

[✓] Enable Email Notifier

To:

Template: Create New...

Send Notification

When Run From: Workbench, Command Line, [✓ Both]

Job Outcome: Only on Success, Only on Error, [✓ Both]

Attached Report: [✓ None], CSV, XML, HTML, PDF, RTF

To enable notification, select the **Enable Email Notifier** check box and select the options, as available. Enter the email addresses to which the notification will be sent in the **To** field, and specify an email format template in the **Template Field**.

In the **Send Notification** options, specify when the email notification should be sent:

- Under the **When Run From** options, indicate that email notifications should be sent when the job is executed from the **Workbench**, **Command Line**, or **Both**.
- Under the **Job Outcome** options**,** indicate that email notification should be sent **Only on Success**, **Only on Error**, or **Both**.
- Under the **For Threshold** options, specify the threshold level for which the email notification will be sent, based on the job results. Choose **Pass**, **Warn**, or **Fail**. You can click **Open Compliance Page** to specify the percentage values for each threshold warning, which are based on the percent match results of the job.

You can also specify the file format of an attached report that displays the job results with the notification email. Choose **None** (does not attach a report with the notification email), **CSV**, **XML**, **HTML**, **PDF**, or **RTF** to specify the file format of the report.

NOTE: Notification options can be modified via a plug-in extension that enables custom notification coding. See Creating and Implementing a Custom Configuration Plug-In for more information.

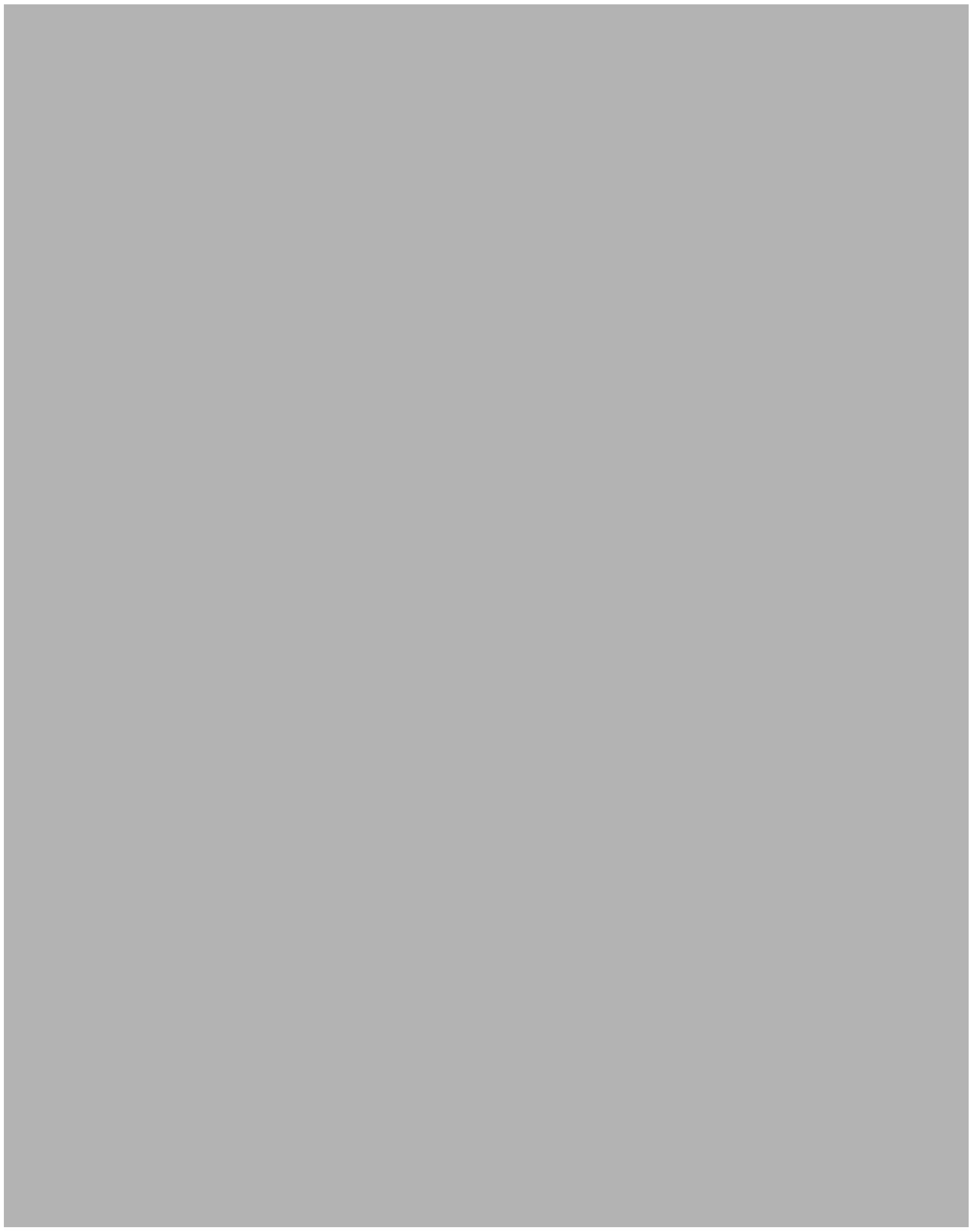
Save the Archive Job/Manage Versioning

A version is a set of schema objects captured by an archive job, and is used to store older snapshots of the same job for backup or tracking comparison purposes. Initially, you must create and save a version for the job to become valid within the system.

A version is created when you click the **Run** icon located in the upper right-hand corner of any tab in the **Schema Archive Job Editor**.

When the process executes, the **New Unsaved Version** tab appears and lists information related to the archive job, including the full archive contents and any extracted DDL.

In order to create a version, highlight the tab and select **File > Save** or **File > Save All** from the Menu Bar. The system retains an older archive job named **Version 1** that appears in the **Archive Job History** table on the **History** tab as well as in the **Data Source Explorer** tree under the **Configuration Archive Jobs** node.



To create a version and save the archive job

- Click the **Run Job** icon. When you have saved the executed job, a version appears in the **Archive Job History** table on the **History** tab containing the name of the archive job, the number of properties retained by the version, a time stamp, where the archive job was run from, and the number of errors. (Workbench for manual job execution, or command line, if automated.)

You create additional versions within the same archive job each time you click **Run Job**. The new version contains the latest values for the captured schema archive job, as specified on the **Refinements** tab. Versions are listed on the **Overview** tab in a table that displays them in ascending order. A version can be accessed for viewing by double-clicking the appropriate version in the table. The selected version opens in a new tab of the **Archive Job Editor**.

Additionally, versions are listed as child nodes of the archive job to which they belong in **Data Source Explorer**. To view a version, double-click the appropriate node.

Compare Archive Contents and Extract DDL

On an open version tab, the Schema Archive Editor enables you to compare that archive version with the live source by right-clicking on the **Archive Contents** table and selecting **Compare**.

Schema Comparison Job Editor opens immediately, and designates the archive version and the original source as its source and target data sources, respectively.

Additionally, you can select **Extract** from the **Archive Contents** menu to extract the DDL of the archived contents. The code opens in the **DDL Extract** window, and can be viewed, saved, or otherwise modified via the **Open Editor** command.

NOTE: Options to compare archive contents and extract DDL are also available via **Data Source Explorer**, under the appropriate archive node.

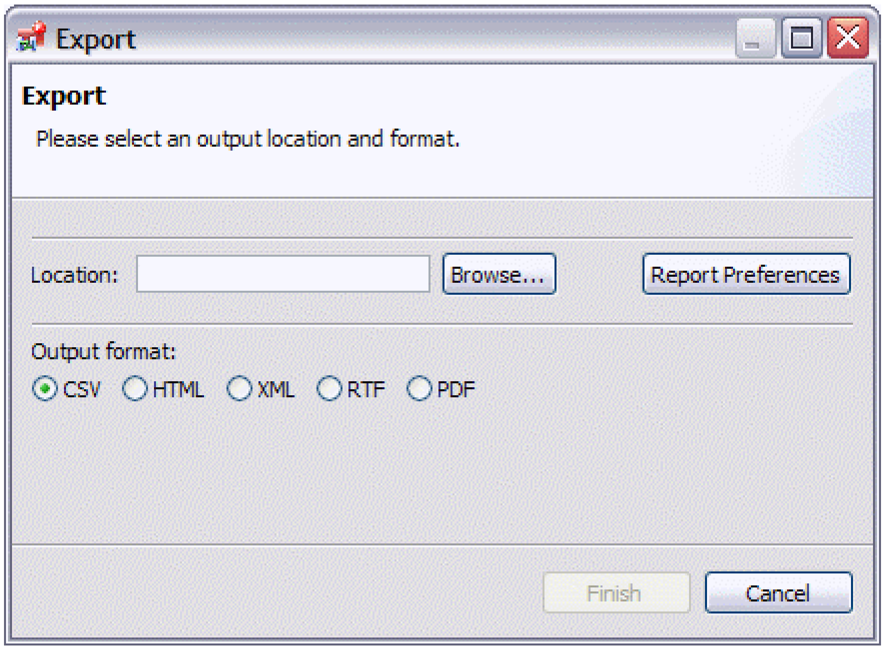
Generating a Job Report

Once you have run an archive job, you can generate a job report that summarizes the archived schema information. You can also generate job reports for any archive versions that exist in the system.

Click the **History** tab and then right-click on the version you want to create a report for. From the menu, select **View Report**, and a summary report of the configuration archive specific to the selected version appears in a separate window. Reports can also be created in **Data Source Explorer** and **Project Explorer** by right-clicking on the appropriate job and selecting **Generate Report**.

The report can be read on screen, sent to a printer, and saved to disk via the command icons located in the report window toolbar.

Additionally, the right-click menu contains a command named **Generate Report**. This will produce a report file in a specified format at a directory location of your choosing.



Fill in the fields, as required. When you click **Finish**, the report is created in the directory specified in the **Location** field, in the format specified in the **Output Format** section.

Use Archive in Schema Comparison Jobs

Archives are used in schema comparison jobs the same way as registered data sources.

To add an archive to a schema comparison job

- Drag and drop an Archive version from **Data Source Explorer** to the **Comparison Source** or **Comparison Target** boxes in the **Schema Comparison Job Editor**.

Configuring Change Manager

This section contains information regarding the general configuration and management of the Change Manager environment.

The following topics can be found in this section:

- Register Data Sources
- Extending Change Manager Functionality with Eclipse Plug-Ins
- Customizing Change Manager (Preferences)
- Using Source Control Systems

Register Data Sources

When you create any job or standard in Change Manager, **Data Source Explorer** provides a tree view selection of data sources to use.

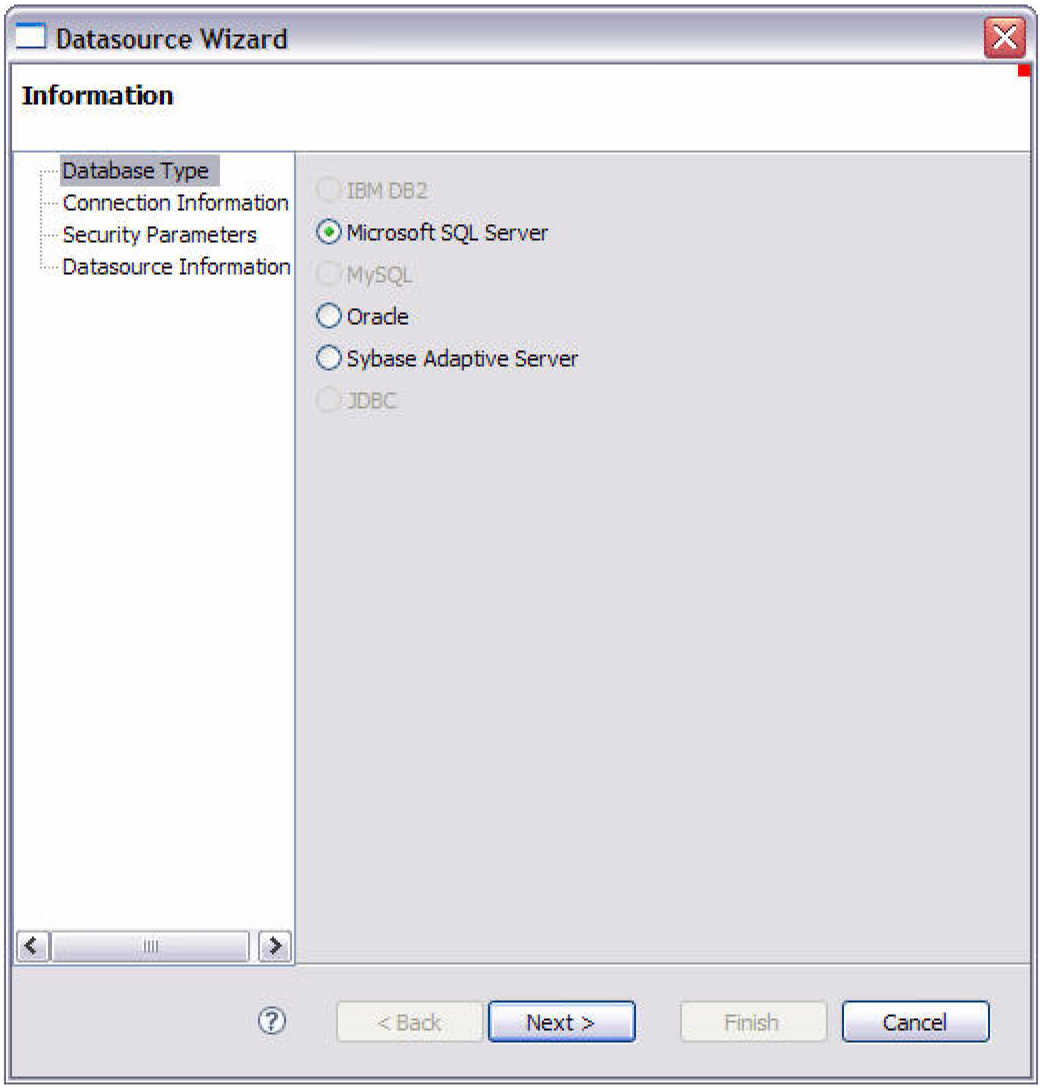
Some Embarcadero products contain data source catalogs that are shared with Change Manager. In other words, instead of manually adding data sources to the environment manually, you can import an existing data source catalog from other Embarcadero products (or older instances of Change Manager.)

To manually initiate a scan of local data sources, click the **Discover Data Sources** icon at the top of Data Source Explorer, or right-click **Managed Data Sources** in the tree and select **File > New > Data Source**. Once a data source is registered, it automatically appears in Data Source Explorer. Connection parameters are stored locally, and Change Manager can be set to connect automatically each time you select the data source from the tree.

The **Import/Export** command also enables you to import an existing data source catalog into Change Manager from various sources, or export your current catalog to an XML file, respectively. The following sources are valid data source catalog formats for the purpose of adding data sources to your environment:

- Embarcadero Windows registry (DBArtisan, Rapid SQL, and CM/Schema 4.0)
- Eclipse Data Tools Platform (DTP)
- Quest Software (TOAD)
- Exported Change Manager XML file

NOTE: When you export the Change Manager data source catalog, an XML file is created and stored in the file as designated under the system **Preferences**. (See Customizing Change Manager (Preferences) for more information.)



To register a new data source

1 Select a Data Source Group in Data Source Explorer and choose **File > New > Data Source** in the Menu Bar. The Data Source Wizard appears.

2 Follow the steps in the data source Wizard by selecting the data source platform and entering the appropriate connectivity parameters in the fields provided. Once you have completed this task, click **OK**.

The new data source appears under the Data Source Group and can be used immediately in comparison jobs.

To import a data source catalog

1 Select **File > Import**. The **Import Data Sources** dialog appears.

2 Choose **Embarcadero > Data Sources** from the import tree and click **Next**.

3 Choose a category from which you want to import the data sources. You can choose to import data sources from the DTP (Eclipse Data Tools Platform), an existing Embarcadero data source catalog stored in the Windows Registry, an existing Embarcadero data source catalog in XML format (created via the **Export** command), or from TOAD. Click **Next**.

4 Specify the location of the import source and click **Finish**. **Data Source Explorer** is automatically populated with the new data sources.

To export the existing data source catalog

1 Select **File > Export**. The **Export Data Sources** dialog appears.

2 Choose **Embarcadero > Data sources** from the export tree and click **Next**.

3 Use the check boxes beside each listed data source to indicate which data sources you want to export. Click **Next**.

4 Click **Finish**.

5 The data sources are automatically exported in the form of an XML file. You can import this file to other instances of Change Manager via the **Import** command.

Extending Change Manager Functionality with Eclipse Plug-Ins

Change Manager publishes two extensions that can be implemented via plug-ins to customize the application.

The two feature extensions are:

- **Notifications:** enable you to code a plug-in for Change Manager that implements any type of job execution notification required for your organization. Change Manager includes email notification by default, but other plug-ins can be coded in Java and added to Change Manager, as needed.
- **Configuraiton Parameters:** enable you to code a plug-in for Change Manager that adds additional configuraiton parameters and values for configuration standards, archive jobs, and comparisons. Change Manager includes plug-ins for standard configuration properties and version information, but other plug-ins can query data sources via Java JDBC for additional or environment-specific information.

In order to utilize the functionality of these features, you will need to build a plug-in and specify the parameters of the notification and custom configuration extensions. The following sections provide examples that demonstrate the possibilities regarding this feature.

- Creating and Implementing a Notification Plug-In
- Creating and Implementing a Custom Configuration Plug-In

Creating and Implementing a Notification Plug-In

The job notification plug-in is built on the Eclipse platform and some knowledge regarding java coding and Eclipse plug-in development is required.

Notifications can be set to alert you if a job is successful, unsuccessful, or either result. Additionally, a notification can also indicate if a job was launched from the workbench or from a command line, and can produce a report in a number of different file formats.

The following options and option parameters may be selected via the job notification plug-in:

Notification Option	Option Parameters
Notify When Run From	Workbench, Command Line, Both

Notification Option	Option Parameters
Attached Report	None, CSV, XML, HTML, PDF, RTF
Send Notification	Always, Only on Success, Only on Failure

Notification parameters are set independently on each job configured in Change Manager. Additionally, each set of notification options apply individually per job selected.

The notification plug-in must be defined in Eclipse and then activated in Change Manager.

The following tasks provide a high-level overview of defining and implementing the notification plug-in:

- Creating a Plug-In Project
- Configuring the Target Platform
- Defining Plug-In Dependencies
- Implementing a Notifier Class
- Implementing a Viewer Contribution
- Specifying Setup Constants
- Implementing the DirectoryNotifier Class
- Implementing the DirectoryNotifierViewerContribution Class
- Registering the Extensions
- Deploying the Notifier Plug-In

Creating a Plug-In Project

In order to start building a plug-in, you need to create a new plug-in platform in Eclipse.

To create a plug-in project

1 Select **File > New > Other**.

The **Select a Wizard** dialog appears.

2 Select **Plug-In Project** and click **Next**.

3 Type a name for the plug-in in the appropriate field, and leave the remainder of the parameters as they appear. Click **Next**.

4 Retain the default parameters on the next dialog and click **Finish**.

The new plug-in project is created in Eclipse and is ready for plug-in development.

Configuring the Target Platform

The target platform for the plug-in development process needs to be identified in Change Manager. This will indicate to the new plug-in for what product it extends, and will grant the plug-in access to the system.

1 Select **Window>Preferences**.

The Preferences dialog appears.

2 Choose the **Plug-In Development > Target Platform** node and click **Browse**.

3 Navigate to the install directory for Change Manager.

4 Click **Apply** and **OK**.

The Preferences dialog closes and the target platform of the plug-in is now indicated.

Defining Plug-In Dependencies

The new plug-in requires a pair of dependency definitions on the Change Manager Notification Plug-In:

- **com.embarcadero.change.notifications**
- **org.eclipse.ui.forms**

1 Navigate to the **META-INF** folder in your project and double-click **MANIFEST.MF**.

The **MANIFEST.MF** file opens in Eclipse.

2 Select the **Dependencies** tab and click **Add...**

3 Choose **com.embarcadero.change.notifications** and click **OK**.

The dependency is added.

4 Choose **org.eclipse.ui.forms** and click **OK**.

The dependency is added.

5 Press **Ctrl+S** to save the changes, or choose **File>Save** from the menu to retain the new dependencies and close the editor.

Implementing a Notifier Class

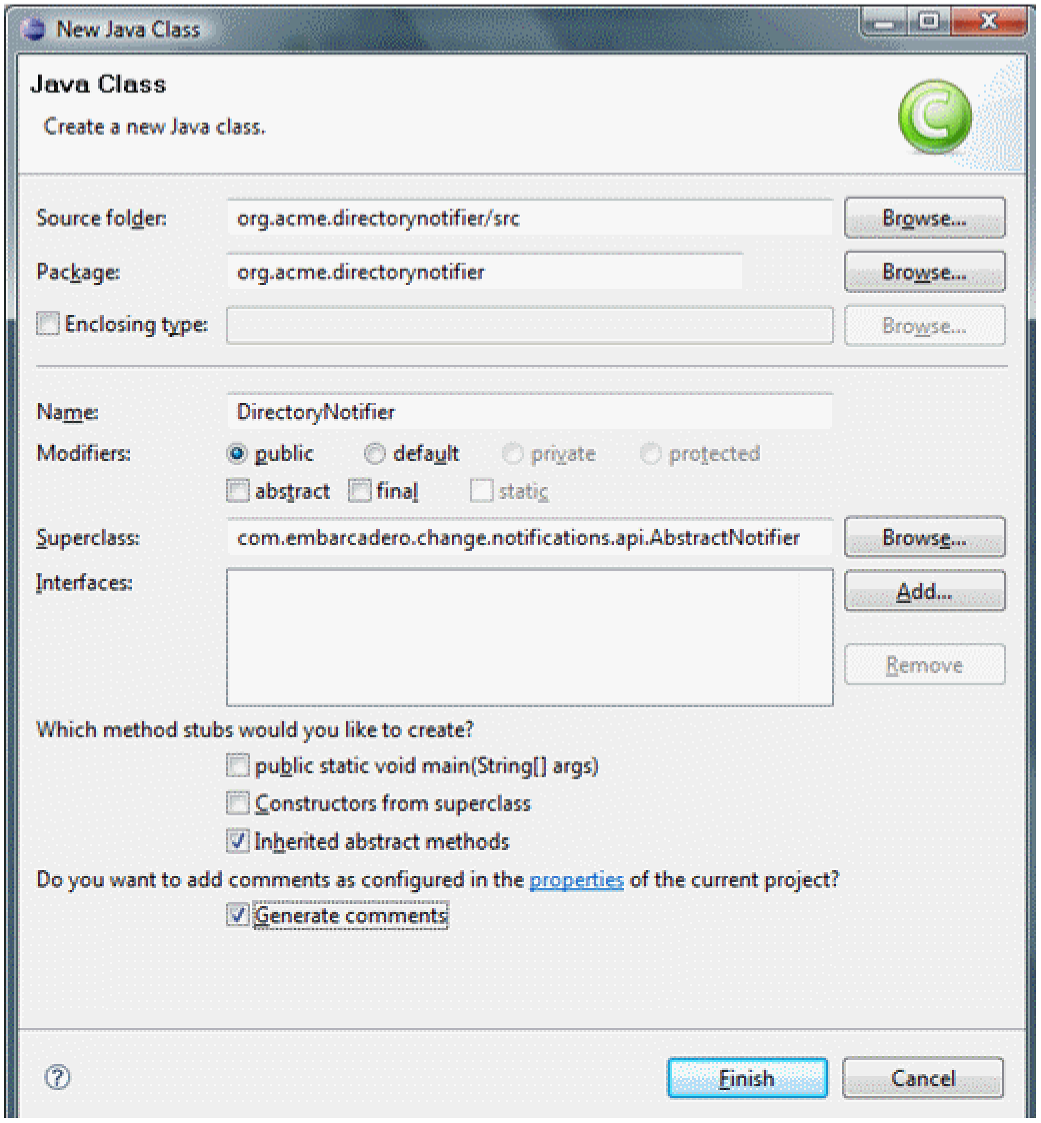
Each notification plug-in requires a Notifier class that must implement the interface **com.embarcadero.change.notifications.api.INotifier**.

To do this, you subclass **com.embarcadero.change.notifications.api.AbstractNotifier** and customize it.

NOTE: If AbstractNotifier or INotifier cannot be found in Eclipse, the target platform or plug-in dependencies have not been configured properly. Ensure that you have configured these prerequisites and then search again for the notifier classes.

1 Right-click on your project and select **New > Class**.

The **New Java Class** dialog appears.



Source folder: org.acme.directorynotifier/src

Package: org.acme.directorynotifier

Enclosing type: [ ]

Name: DirectoryNotifier

Modifiers: [✓ public] default private protected

[ ] abstract [ ] final [ ] static

Superclass: com.embarcadero.change.notifications.api.AbstractNotifier

Interfaces:

Which method stubs would you like to create?

[ ] public static void main(String[] args)

[ ] Constructors from superclass

[✓] Inherited abstract methods

Do you want to add comments as configured in the properties of the current project?

[✓] Generate comments

2 Enter the appropriate information in the fields provided to define the new Java class:

- In the **Name** field, type “DirectoryNotifier”.
- In the **Package** field, type “org.acme.directorynotifier”.
- In the **Superclass** field, type “com.embarcadero.change.notifications.api.AbstractNotifier”.

3 Click **Finish** to create the new class.

Once you have defined the DirectoryNotifier class, two methods require implementation:

- **isReportSupported** returns a Boolean value indicating if the notification can include a report. The directory notifier will return true, since reports can be replaced in a directory, but a notification may not support reports. This method is called when creating job editors to determine if the reporting section will be shown for this type of notification.
- **sendNotification** is called after a job runs and is reponsible for the notification. The three parameters are as follows:

Parameter	Type	Description
notifierData	INotifierData	This instance contains the configuration for the notification of the job just run.
jobMetaData	Map<String,String>	This map contains information about the job execution. The keys in the map correspond to entries in the NotificationPropertyEnum. For example, to get the data and time of the exection, you would code: “jobMetaData.get(NotificationPropertyEnum.DATE_TIME.getTag())” **Note:** The keys are the tag of the enum entries, so ensure that getTag() is called when accessing jobMetaData.
notificationInfo	IReportGenerator	Used to generate the report and access any DDL or sync script output.

Implementing a Viewer Contribution

Each notifier can be configured for a job in Change Manager.

The notifier will enable a user to select a target directory in which to put results. The email notifier that is supplied with the application enables users to select the email template that will be used to generate the message as well as the addresses of those who will received it.

In order to add a directory field to the job editor, a class needs to be written that implements the interface named **com.embarcadero.change.notifications.api.INotifierViewerContribution**.

- Create a new class named “DirectoryNotifierViewerContribution” and follow the same steps you used to create the DirectoryNotifier class. The exception to this process is that you need to leave the Superclass parameter set to “java.lang.Object”, as well as adding the interface “INotifierViewerContribution” to the list of implemented interfaces.

Specifying Setup Constants

The notifier only allows users to select a directory in which reports and outcomes are written whenever jobs are executed.

The **DirectoryNotifierConstants** class is a small constants class that stores the property name for storage and retrieval purposes.

1 Create a new class named **DirectoryNotifierConstants**.

2 Enter the following code for the class definition:

```
/**
    * Constants for this network share notifier
    */
    public final class DirectoryNotifierConstants
    {
        // Overview file name
        public static final String OVERVIEW_FILE_NAME = “outcome.txt”;

        // Persisted Job Settings Key
        public static final STring TARGET_DIRECTORY = “directorynotifier.targetdir”;

    /**
    * Constant utility class cannot be instantiated
    */
    private DirectoryNotifierConstants()
    {
    }
}
```

Implementing the DirectoryNotifier Class

The DirectoryNotifier class can be coded with two return methods:

- **isReportSupported**
- **SendNotification**

To implement the isReportSupported method

Enter the following code:

```
public boolean isReportSupported()
{
    return true;
}
```

To implement the SendNotification method

1 To add the notifierProperties and assign the targetDirectory to a variable to provide a directory that the user selected for the executed job, add the following code:

```
public void sendNotification ( INotifierData notifierData,
                                Map<String, String> jobMetaData,
                                notificationInfo

    Map<String, String> notifierProperties = notifierData.getNotificationProperties();

    String targetDirectory = notifierProperties.get (
                                DirectoryNotifierConstants.TARGET_DIRECTORY );
```

2 Enter the following code to define that the report and output file need to be placed in a directory whose name will be composed of the job execution date and time.

NOTE: If a directory cannot be created, the code below also includes a RuntimeException.

```
//Create a new directory named by the execution date and time
String dateTime = jobMetaData.get ( NotificationPropertyEnum.DATE_TIME.getTag() );
File dir = new File (targetDirectory +
                    File.seperatorChar +
                    dateTime.replace ( ‘:’, ‘-’ ) );
                    boolean createdDirectory = dir.mkdir();
                    if ( !createdDirectory )
                    {
                    throw new RuntimeException ( “Create Failed: ” +
dir.getAbsolutePath () );
                    }
```

3 Enter the following code to define the outcome file, which includes each of the job execution parameters:

```
try
{
    //Write the results to an output file
    File f = new File ( dir.getAbsolutePath() +
        File.seperatorChar +
        DirectoryNotifierConstants.OVERVIEW_FILE_NAME );
        f.createNewFIle();

        BufferedWriter writer = new BufferedWriter ( new FileWriter ( f ) );

        String newLine = System.getProperty ( “line.seperator” );

        writer.write ( “Outcome: “ +
                        jobMetaDAta.get (
                        NotificationPropertyEnum.JOB_OUTCOME.getTag() ) +
                        newLine );
        writer.write    ( newLine );
        writer.write    ( “Date & TimeL “ +
                            jobMetaData.get (
                            NotificationPropertyEnum.DATE_TIME.getTag() ) +
                            newLine );
        writer.write    ( “Elapsed Time: “ +
                            jobMetaData.get (
                            NotificationPropertyEnum.ELAPSED_TIME.getTag() ) +
                            newLine );
        writer.write        newLine );
        writer.write    ( “Name: “ +
                            jobMetaDAta.get (
                            NotificationPropertyEnum.JOB_NAME.getTag() ) +
                            newLine );
        writer.write (newLine );
        writer.write    ( “Host: “ +
                            jobMetaData.get (
                            NotificationPropertyEnum.JOB_HOST.getTag() ) +
                            newLine );
        writer.write    ( “Job Type: “ +
                            jobMetaData.get (
                            NotificationPropertyEnum.JOB_TYPE.getTag() ) _
                            newLine );
        writer.write    ( “Module: “ +
                            jobMetaData.get (
                            NotificationPropertyEnum.MODULE.getTag() ) +
                            newLine );
        writer.write    ( newLine );
        writer.write    ( “Sources: “ +
                            jobMetaData.get (
                            NotificationPropertyEnum.JOB_SOURCES.getTag() ) +
                            newLine );
        writer.write    ( “Targets: “ +
                            jobMetaData.get (
                            NotificationPropertyEnum.JOB_TARGETS.getTag() ) +
                            newLine );
        writer.write    ( newLine );
        writer.write    ( “Job Notes: “ +
                            jobMetaDAta.get (
                            NotificationPropertyEnum.JOB_NOTES.getTag() ) +
                            newLine );

        writer.flush();
        writer.close();

}

catch (Throwable t )
{
    throw new RuntimeException ( t );
    }
```

4 Enter the following code to generate the report. This code also provides a null report if the report type is set to none at execution time.

```
//Create the report if required
File reportFile = notificationInfo.getReport() ;

if ( reportFile !=null )
{
    File destinationReportFile = new File
        dir.getAbsolutePath() +
        File.seperatorChar +
        reportFile.getName() );
    reportFile.renameTo ( destinationReportFile );
}
```

Finally, output any DDL or synchronization SQL:

```
//Write the script
File scriptFile=notificationInfo.getScript();

if ( scriptFile !=null)

    File DestinationScriptFIle = new File (
        dir.getAbsolutePath() +
        File.separatorChar +
        scriptFile.getName () );
    scriptFile.renameTo (destinationScriptFile);
```

Implementing the DirectoryNotifierViewerContribution Class

Add the code below to the **DirectoryNotifierViewerContribution** class to provide functionality for the interface:

1 The DirectoryNotifierViewerContribution class requires a three class-level private attributes. These store notification data, which will be the composite on which any options will then be placed, as well as the **folderField**, which is used to enter a file directory address.

```
private INotifierData notificationData;
private Composite body;
private Text folderField;
```

2 The **createFormContent** attribute creates a new composite and provides a lay out template for options and entry fields. The code below adds a label and a text field to the interface:

```
public Composite createFormContent ( Composite parent, FormToolkit toolkit )
    body = toolkit.createComposite ( parent );
    body.setLayout ( new GridLayout () );

    toolkit.createLabel ( parent, “Target Directory:” );

    String folder = notificationData == null ? “ “ :
                        notificationData.getNotificationProperties().get (
                        DirectoryNotifierConstants.TARGET_DIRECTORY );

    folderField = toolkit.createText ( parent, folder, SWT.SINGLE | SWT.BORDER );
    folderField.setLayoutData ( new GridData ( SWT.FILL, SWT.CENTER, true, false ) );
    folderField.addModifyListener ( new ModifyListener () {
        public void modifyText ( ModifyEvent e )
        {
            notificationData.setNotificationPropert (
                DirectoryNotifierConstants.TARGET_DIRECTORY,
                folderField.getText().trim() );
        }
});

return body;
```

The above code:

- Creates a new composite body, and sets it to have a grid lay out.
- Creates a label for the target directory field.
- Retrieves the folder from the job, if the job is new or unsaved, or defaults to an empty string if the result is null.
- Initializes the folderField text element and sets it to justify.
- Adds a listener for changes on the folderField element that updates the notificationData, thus enabling save actions.
- Returns the body to be added to the Change Manager job editor to which this class applies.

3 Add the setData method by entering the following code, which is called to update the user interface with job information. This method accepts and stores the new instance of INotifierData containing job specific data for the notifier. If the folderField has already had something entered, the notification properties will be initialized and store the value that was previously entered.

```
public void setData ( INotifierData notificationData )

        this.notificationData = notificationData;

        if ( folderField !=null && !folderField.isDisposed() &&
                folderField.getText().trim().length() > 0 ) )

                    if ( notificationData !=null )

                        Map<String, String> propMap
                            notificationData.getNotificationProperties();

                        if ( propMap == null )
                        {
                            notificationData.setNotificationProperties(
                                new HashMap<String, String>() );

                            propMap = notificationData.getNotificationProperties();
                        }

                        notificationData.setNotificationProperty (
                            DirectoryNotifierConstants.TARGET_DIRECTORY,
                            folderField.getText() );
                        }
                }
        }
```

The following code should be used to implement the **setEnabled** method:

```
public void setEnabled ( boolean enabled )
{
    folderField.setEnabled( enabled );
}
```

The following code should be used to implement the **dispose** method:

```
public void dispose()
    {
        if ( body !=null && !body.isDisposed() )
        {
            body.dispose();
            body = null;
        }
    {
```

Registering the Extensions

Once the plug-in has been coded, and the notifier and user interface implemented, definitions must be included that explain how Change Manager should interact with the plug-in.

This is performed by registering extensions that define to Change Manager the Notifier and NotifierViewerContribution. When the plug-in is added to Change Manager, the extensions are registered and Change Manager understands the purpose of the plug-in.

1 Open the **MANIFEST.MF** file.

2 Navigate to the **plugin.xml** tab and display the source.

3 Add the following code:

```
<extension
    point=“com.embarcadero.change.notifications.notifier”>

    notifier
            id=“org.acme.directorynotifier”
            name=“Directory Notifier”
            notifier=“org.acme.directorynotifier.DirectoryNotifier”
            viewContribution=
            “org.acme.directorynotifier.DirectoryNotifierViewerContribution”>
    </notifier>
</extension>
```

4 Save the file by pressing **Ctrl+S** or selecting **File>Save** from the menu.

When the org.acme.directorynotifier is found, the notification extension point will be registered. The notifier and viewContribution attributes are the class names for two implementations and Eclipse will reflectively register these objects for use in Change Manager.

Deploying the Notifier Plug-In

Once the plug-in has been defined, you need to deploy the plug-in in Change Manager.

1 Stop Change Manager if it is currently running and select **File > Export**.

The **Export** dialog appears.

2 Choose **Deployable Plug-Ins and Fragments** from the **Plug-In Development** group.

3 Click **Next**.

4 Select **org.acme.directorynotifier** and ensure that the target directory is attribute to the Change Manager installation folder.

5 Click **Finish**.

6 Start Change Manager and create a new Data Comparison job. Define a Source and Target data source and then navigate to the **Notification** tab.

The **Directory Notifier** has been added as a section to the tab.

7 Choose **Enable Directory Notifier** to enable the feature.

8 Specify the **Target Directory**. For example, **C:\ChangeManagerNotifications**, and choose **PDF** in the **Attached Report** section.

9 Execute the job and open the target folder. A new folder is created containing two output files from the job.

Creating and Implementing a Custom Configuration Plug-In

Creating a Project Plug-In

In order to start building a plug-in, you need to create a new plug-in platform in Eclipse.

To create a plug-in project

1. Select **File > New > Other**.

   The **Select a Wizard** dialog appears.

2. Select **Plug-In Project** and click **Next**.
3. Type a name for the plug-in in the appropriate field, and leave the remainder of the parameters as they appear. Click **Next**.
4. Leave the parameters on the next screen as they appear and click **Finish**.

The new plug-in project is created in Eclipse and is ready for plug-in development.

Configuring the Target Platform

The target platform for the plug-in development process needs to be identified in Change Manager. This will indicate to the new plug-in for what product it extends, and will grant the plug-in access to the system.

1. Select **Window>Preferences**.

   The Preferences dialog appears.

2. Choose the **Plug-In Development>Target Platform** node and click **Browse**.
3. Navigate to the install directory for Change Manager.
4. Click **Apply** and **OK**.

   The Preferences dialog closes and the target platform of the plug-in is now indicated.

Defining Plug-In Dependencies

The plug-in requires a dependency for configuration comparison jobs in Change Manager. This dependency can be set by opening the **MANIFEST.MF** file located in the **META-INF** directory of your project.

1. Select the **Dependencies** tab and click **Add...**
2. Choose **com.embarcadero.change.config.propertysource** and click **OK**.

   The dependency is added to the plug-in and appears in the Dependencies list.
3. Press **CTRL-S**, or Select **File>Save** from the menu to save your changes.

Once the dependency definition has been saved, you can close the editor and the system will retain your selection.

Implementing a Property Source

Each configuration property source plug-in must contain a class that extends **com.embarcadero.change.config.propertysource.api.ConfigurationPropertySource**.

NOTE: If ConfigurationPropertySource cannot be found in Eclipse, it means you have misconfigured the target platform or target dependencies.

1. Right-click on the project and select **New>Class**.

   The **New Java Class** dialog appears.
2. Name the class **JDBCVersionInformation**, set the package to **org.acme.versioninformation**, and set the superclass to **com.embarcadero.change.config.propertysource.api.ConfigurationPropertySource**.
3. Click **Finish**.

   The new class is generated.

Determining Support

Each class that subclasses **ConfigurationPropertySource** contributes properties to configuration comparison job functionality.

The implementation is indicated via the **isSupported** method, which determines if there are attributes available for a specific data source.

For example, if a user added the Sybase @@Version tag as a configuration property, it would be pointless to run the code for Oracle, SQL Server, or DB2 for LUW data sources.

The method has the following signature:

```
boolean isSupported ( IDatasourceIdentifier dataSourceIdentifier )
```

This method is called by Change Manager, for any registered content plug-in when a data source is used in a Configuration Archive job, a Standard, or a comparison job. When it encounters a data source identifier, it returns true or false and indicates if the property source provides parameters for the data source, or not.

Implement the following code to define this method:

```
public boolean isSupported ( IDatasourceIdentifier dataSourceIdentifier )
{
    return dataSourceIdentifier.getDatabaseVersion().getDBMSType() !=DBMSType.DB2;
}
```

Defining Properties

The **getAvailableProperties** function returns a set of instances of **IConfigurationProperty** instances.

There are several utility methods to help instantiate these instances, named of the form, **createConfigurationProperty**.

Implement the following code to define the getAvailableProperties method:

```
public Set(IConfigurationProperty> getAvailableProperties (
                                   IDatasourceIdentifier identifier )
{
    Set<IConfigurationProperty> results = new HashSet<IConfigurationProperty>();

    for (EVersionProperties prop : EVersionProperties.values() )
    {
        results.add ( createCOnfigurationProperty ( prop.getDisplayName(),
                                                    prop.name(),
                                                    “”,
                                                    prop.getType(),
                                                    “” ) );
    }

return results;
}
```

Because the EVersionProperties code cannot be resolved, you can add a Java enum to define the properties provided by the plug-in. Create a new file named **EVersionProperties.java** and implement the following code:

```
package org.acme.versioninformation;

import java.sql.Connection;
import java.sql.SQLException;

import com.embarcadero.change.config.propertysource.api.ConfigurationPropertyType;

/**
* An enumeration of version properties
*/
public enum EVersionProperties
{
    PRODUCT_NAME ( ConfigurationPropertyType.STRING, “Product Name” )
    {
        /**
        * @returns the JDBC metadata’s information for product name
        * @throws SQLException if ther eis a JDBC error
        */
        public String getValue ( Connection connection ) throws SQLException
        {
            return connection.getMetaData().getDatabaseProductName();
        }
    },
    PRODUCT_VERSION ( ConfigurationPropertyType.STRING, “Product Version” )
    {
        /**
        * @returns the JDBC metadata’s information for product version
        * @throws SQLException if there is a JDBC error
        */
        public String getVAlue ( Connection connection ) throws SQLException
        {
            return connection.getMetaData().getDAtabaseProductVersion();
        }
    },
    MAJOR_VERSION ( ConfigurationPropertyType.NUMERIC, “Major Version” )
    {
        /**
        * @returns the JDBC metadata’s information for major version
        * @throws SQLException if there is a JDBC error
        */
        public String getValue ( Connection connection ) throws SQLException
        {
            return “ ” + connection.getMetaData().getDataMajorVersion();
        }
    };

//Attributes
private ConfigurationPropertyType type;
private String name;

/**
* Constructor to set entry attributes
* @param type the type
* @param name
*/
private EVersionProperties ( ConfigurationPropertyType type,
        String name)
}
    this.type = type;
    this.name = name;
{

/**
* Given a JDBC connection return the parameter’s value
* @param connection the database connection
* @throws SQLException if there is a JDBC error
*/
abstract String getValue ( Connection connection ) throws SQLException;
/**
* @return the display name of the parameter
*/
public String getDisplayName()
{
    return name;
}

/**
* @return the type of the parameter
*/
public ConfigurationPropertyType getType()
{
    return type;
}

/**
* Finds a property of a name and returns it
* @param name the name of the property
* @return the property
*/
public static EVersionProperties getProperty ( String name )
{
    for ( EVersionProperties p : values() )
    {
        if ( p.name().equals ( name ) )
        {
            return p;
        }
    }

    throw new RuntimeException ( EVersionProperties.class.getName() + " not found for
'" + name + "'" );
    }
}
```

In the above code, the four entries that capture from JDBC the product name, version, minor version, and major version have been added to the enum.

Gathering Results

Change Manager gathers results when it collects all of the properties and values for display purposes or to store in an archive. The extraction method is called when a data source is indicated in a Standard, or when an archive or comparison job is executed.

```
public Set<IConfigurationPropertyResult> extract( Set<IConfigurationProperty>
properties, IDatasourceIdentifier dataSourceIdentifier )
```

A set of IConfigurationPropertyResult instances that correspond to the properties must be returned, and values need to be added for the properties returned from the data source, as follows:

```
public Set<IConfigurationPropertyResult> extract ( Set<IConfigurationProperty>
properties, IDatasourceIdentifier dataSourceIdentifier)
{
    try
    {
        Set<IConfigurationPropertyResult> results = new
HashSet<IConfigurationPropertyResult> ();

        Connection connection = getConnection (dataSourceIdentifier );

        for (IConfigurationProperty configProp : properties )
        {
            EVersionProperties prop = EVersionProperties.getProperty (
                                    configProp.getPropertyIdentifier() );
            String value = prop.getValue ( connection );

            results.add( createConfigurationPropertyResult( configProp, true, value )
);
    }
return results;
}
catch ( SQLException sql )
{
    throw new RuntimeException ( sql );
    }
}
```

In the above implementation, the code creates an empty set to hold the results and then retrieves the JDBC connection to the data source. For each retrieval, the enum constant EVersionProperties on which the getValue is called to retrieve each property value from JDBC metadata. Finally, createConfigurationPropertyResult is called to create the result instance.

Synchronizing the Job

The synchronization method returns an empty array because users cannot synchronize version information.

```
public List<IPropertyResultsSynchronization> sync (
    IDatasourceIdentifier dataSourceIdentifier,
    List<IConfigurationPropertyResult> sourcePropertyValues,
    List<IConfigurationPropertyResult> targetPropertyValues )
{
    return new ArrayList<IPropertyResultsSynchronization>();
}
```

Registering Extensions

Once you have completed coding the plug-in, you will need to register it with Change Manager. This is achieved by implementing an Eclipse extension. When the plug-in is added to Change Manager, the extensions are registered and Change Manager understands that the plug-in adds a new type of notification to the module.

1 Open the **Manifest.MF** file and navigate to the **Extensions** tab.

2 Click **Add** to register an extension.

  The **New Extension** dialog appears.

3 Choose **com.embarcadero.change.config.propertysource.contribution**.

4 Click **Finish** and close the dialog.

  The extension is added to the **All Extensions** list.

5 Select the (source) entry on the right-hand side of the screen and enter the following details:

- **display-name:** JDBC Metadata Version Information
- **source-identifier:** org.acme.jdbc.versioninformation.source
- **class:** org.acme.jdbcversioninformation.JDBCVersionInformation

NOTE: The class name is used to enable Change Manager to instantiate the new class, while the display-name is the group under which the four JDBC metadata version properties will show.

6 Save the file by pressing **CTRL-S** or choose **File>Save** from the menu.

The class notification is complete, and Change Manager and Eclipse are informed about the property source. When the plug-in is added to Change Manager, it will add your properties to configuration comparison and Standard jobs.

Deploying the Custom Configuration Plug-In

To deploy the plug-in, perform the following steps:

1 Shut down Change Manager if it is running.

2 Choose **File>Export**.

The **Export** dialog appears.

3 Select **Deployable Plug-ins and Fragments** from the **Plug-In Development** group and click **Next**.

4 Select **org.acme.jdbcversioninformation** and ensure the target directory is the same as the Change Manager installation folder.

5 Click **Finish**.

The new plug-in is deployed.

Start up Change Manager and create a new configuration archive job. When you select an Oracle, Sybase, or SQL Server data source, the **Refinements** and the **Results** tab now contain the four version properties you defined in the new plug-in.

Customizing Change Manager (Preferences)

To customize aspects of Change Manager, select **Window > Preferences**. a dialog appears with a preferences tree that enables you to select the category of application preferences you want to change. Preferences specific to Change Manager appear in the **Change Manager** node.

The following sections outline preferences specific to Change Manager:

- Compliance
- Comparison Options
- Data Result Options
- Script Execution Options
- Script Generation Options
- Jobs
- Command Line

- Mapping
- Notification
- Reports
- Data Sources
- SQL Editor
- Code Assist
- Code Formatter
- Results Viewer
- Syntax Coloring
- SQL Filters

Compliance

This node is located on the **Preferences** panel, under the **Configuration Comparison** sub-node, under **Change Manager**.

The **Compliance** setting provides percentage amount parameters that specify when to **Pass**, **Warn**, or **Fail** a configuration comparison job in the Compliance view.

Adjust the **Pass**, **Warn**, and **Fail** values as needed. The symbols beside each value will appear beside a job in the view, depending on the percentage amount that results from the comparison job.

Comparison Options

This node is located on the **Preferences** panel, under the **Data Comparison** sub-node, under **Change Manager**.

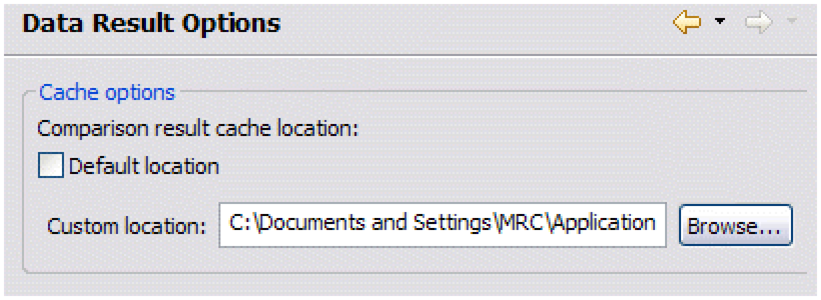
The **Comparison Options** setting enables you to specify when compression comparison is automatically initiated during a data comparison job, based on the total size of the rows in a given table.

Select **Optimize Comparison Based on Row Length** and then adjust the value (in bytes) in the box below. The value determines when compression is automatically enabled on a table during the data comparison process, based total size of all of the rows. If the size of the rows is greater than the this value, then compression is automatically enabled.

Data Result Options

This node is located on the **Preferences** panel, under the **Data Comparison** sub-node, under **Change Manager**.

The **Data Result Options** setting enables you to specify the directory location where the results of data comparison jobs are stored.


Data Result Options

Cache options

Comparison result cache location:

☐ Default location

Custom location: C:\Documents and Settings\MRC\Application

By default, results are cached locally in the system **Application** directory.

To change this location, de-select **Default Location** and enter the new directory address in the **Custom Location** field.

NOTE: This is a temporary cache, and during large data comparison jobs may use up much of the system resources.

Script Execution Options

This node is located on the **Preferences** panel, under the **Data Comparison** sub-node, under **Change Manager**.

The **Script Execution Options** setting enables you to indicate that auto commit is always applied when executing synchronization scripts for data comparison jobs.

Script Generation Options

This node is located on the **Preferences** panel, under the **Data Comparison** sub-node, under **Change Manager**.

The **Script Generation Options** setting specifies how Change Manager handles duplicate rows when generating a SQL synchronization script for data comparison jobs.

- Select **Warn About Duplicate Rows** to prompt a user when the data comparison synchronization script generation process encounters them.
- Select **Exclude Duplicate Rows** to indicate that the data comparison synchronization script generation process should ignore any duplicate rows that it encounters.

These options are useful if the match key for a table in a data comparison job does not reflect uniqueness. For example, if a match is performed on **last_name**, and there are several values of **Smith** in a table, the synchronization of data may have unexpected results.

Jobs

This node is located on the **Preferences** panel, under the **Schema Comparison** sub-node, under **Change Manager**.

The **Jobs** setting specifies how schema comparison jobs launch in the development environment.

- Select **Do Not Ask About Running the Created Jobs** if you want to ignore the system prompt when running schema comparison jobs.
- Select **Automatically Launch Jobs Created from Selected Objects** if you want the system to automatically execute a schema comparison job upon creation.

Command Line

This node is located on the **Preferences** panel, under the **Change Manager** node.

The **Command Line** node provides the **Default Script Output** setting, which enables you to define the operating system/file type by which command line scripts are stored as during the batch file creation process. (See Building Command Line Syntax and Creating Bulk Job Files for details of this process.)

- Select **Windows (*.bat)** (for scripts to be run on Windows), **Linux (*.sh)** (for scripts to be run on Linux), or **Ant (*.xml)** as appropriate.

Mapping

This node is located on the **Preferences** panel, under the **Change Manager** node.

The **Mapping** node is used to specify how the system handles syntax nuances between objects during comparison job mapping processes.

- Select **Ignore Case**, **Ignore Spaces**, and/or **Ignore Underscores**, as needed.

Notification

This node is located on the **Preferences** panel, under the **Change Manager** node.

The **Notification** node is composed of the **Email Notification** subnode and its associated branches. These settings are used to modify and set up how Change Manager handles job notifications for its various modules.

- The **Email Notification** node contains the **Host Settings** and **User Information** fields that the system will use to send email notifications.
- The **Advanced Settings** node contains JavaMail key-value pair definitions that the system uses to send email notifications. Refer to https://java.sun.com/products/javamail/downloads/index.html for more information. An FAQ is also available here: https://java.sun.com/products/javamail/FAQ.html, and SMTP properties are listed here: http://java.sun.com/products/javamail/javadocs/com/sun/mail/smtp/package-summary.html.
- The **Contacts** node enables you to create a list of address contacts to whom email notifications will be sent.
- The **Templates** node enables you to define and save email notification templates, which control the format in which notification emails are sent.

  NOTE: Pressing **Ctrl-Space** on the **Notification** screen opens a list of tags that Change Manager replaces with job-specific content.

Reports

This node is located on the **Preferences** panel, under the **Change Manager** node.

The **Reports** node provides settings and parameters that enable you to customize the look, feel, and format of reports issued when you generate a job report for comparison jobs.

- The **Report Style** tab enables you to change the logo and accent color of a report.
- The **Paper Setup** tab enables you to modify printer settings for paper reports.

Data Sources

This node is located on the **Preferences** panel, under the **Change Manager** node.

The **Data Sources** node contains a directory location field that specifies where registered data source definitions are stored.

SQL Editor

This node is located on the **Preferences** panel, under the **SQL Development** node.

1 Select **Window > Preferences > SQL Development > SQL Editor**.

2 Change the settings as appropriate in each section and then click **Apply**.

- **Enable SQL Parser** indicates that the parser that provides code development features in SQL Editor is enabled. If this option is disabled, functions such as code formatting, auto completion, semantic validation, hyperlinks, etc. will not be available. The options below the check box enable you to limit (by file size) when the parser is enabled when dealing with code on a file-by-file basis.
- **Parsing Delay** specifies the amount of time that the parser delays prior to activating features after a keystroke.
- **Severity Level for Semantic Validation Problems** determines how semantic code errors are flagged in the editor and the **Problems** view.

Clearing **Enable SQL Parser** will disable many of the “smart” SQL editor features, including code formatting, auto completion, semantic validation, and hyperlinks. For better performance, you may disable the parser for files above a specified size.

Code Assist

The **Code Assist** panel is used to specify configuration parameters that determine how code completion features in SQL Editor behave.

Select **Window > Preferences > SQL Development > Code Assist**.

- **Enable Auto Activation** enables or disables code assist functionality.
- **Insert Single Proposals Automatically** specifies if only a single code completion suggestion is returned, it is inserted automatically.
- **Fully Qualified Completions Automatically** specifies if code completion results are returned specific (fully qualified), rather than the minimum required to identify the object.

Code Assist Color Options specifies the color formatting of code completion proposals. Select background or foreground options from the menu and modify them as appropriate.

Code Formatter

The **Code Formatter** pane provides configuration options for code formatting functionality in SQL Editor.

Select **Window > Preferences > SQL Development > Code Formatter**.

The panel provides a drop down list of formatting profiles and a preview window that displays how each profile formats code.

Results Viewer

The **Results Viewer** pane provides configuration options that specify how the **Results** view displays results.

Select **Window > Preferences > SQL Development > Results Viewer**.

- **Grid Refresh Interval** indicates the speed in millisecords that the Results view refreshes.
- **Stripe the Rows of the Results Table** adds intermittent highlighted bars in the Results view.

Display Results in Separate Tab in SQL Editor opens the Results view in a separate window on the Workbench.

Syntax Coloring

The **Syntax Coloring** panel provides configuration options that change the look and feel of code syntax in SQL Editor.

Select **Window > Preferences > SQL Development > Syntax Coloring**.

Use the tree view provided in the **Element** window to select the comment type or code element you want to modify. Select the options to the right-hand side of the window to modify it. The **Preview** window shows a piece of sample code that updates according to the changes you made.

SQL Filters

See Filtering Data Source Objects for instructions.

Using Source Control Systems

Change Manager can be integrated with source control systems such as Concurrent Versions System (CVS) and Visual Source Safe (VSS).

Source control systems provide a repository for project files and are aimed specifically at team programming environments where developers can work on the same code simultaneously and merge it as needed. Most source control systems provide versioning as well, which provides backup instances of code when development regression is required.

Typically, a source control system repository and its associated files are updated by individual members of a development team as they add and modify files in the project. Work is shared in this manner as the project evolves, and the latest work is always available in a given project as individuals can update files from the specified Change Manager directory to the branch repository and vice-versa.

The Navigator provides an interface for accessing these files in the Change Manager development environment. Job files and archive versions are stored as XML files and synchronization scripts are stored as text, in a specified directory for the purpose of sharing them in some manner of third-party source control system.

For more information regarding the implementation of source control systems, as well as the client and tool application software and how it integrates with Change Manager (built on the Eclipse platform framework for source sharing and version control systems), consult the third-party documentation provided with these plug-ins and/or applications.

All source code control actions much be manually performed.

NOTE: Change Manager is built on the Eclipse 3.3 framework. Consequently, any source code control plug-in designed to work with Eclipse 3.3 can also be implemented in Change Manager.

Setting Up Concurrent Versions System

Concurrent Versions System (CVS) can be downloaded from the following Web site: http://ximbiot.com/cvs/cvshome. By default, Change Manager framework supports a plug-in for CVS.

Once you have installed CVS on your machine and have created a repository system (consult the CVS documentation for details regarding these tasks), you need to enable it in Change Manager.

To set up CVS in Change Manager:

1 Choose **Help > Software Updates > Find and Install** and choose **Search for New Features to Install**. Click **Next**.

2 Select **Eclipse 3.3 Update Site** or add it by selecting **New Remote Site...** and entering the following URL: http://update.eclipse.org/updates/3.3*

3 Choose **Automatically Select Mirrors** and click **Finish**. A tree appears.

4 Choose the CVS plug-ins from the tree and install them. You may need to include dependencies as well.

5 Click **Finish** and restart Change Manager.

6 Right-click in the **CVS Repositories** view and choose **New > Repository Location...**

7 Enter the server and directory in the fields provided. The connection type should be **pserver**.

8 Enter your Windows credentials and specify the default port.

9 Click **Apply** and then save the connection. It appears in the Repository view.

Using Concurrent Versions System

To connect a project to the repository

1 Highlight the **Navigator** view and right-click on the project you want to associate with CVS.

2 Select **Team > Share Project...** The **Share Project** dialog appears.

3 Select **CVS** and click **Next**.

4 Choose the existing repository and click **Next**.

5 Specify the default for **Enter Module Name** and click **Next**.

6 Complete the wizard as needed to check the files into the CVS repository. The project is added to CVS.

NOTE: If prompted, change any current job files and synchronization scripts from Binary to ASCII format.

To check in files

- Jobs and scripts that are not up to date in the CVS repository appear in the **Navigator** with a **>** prefix beside the file name. You can update files in the source control by right clicking on the file name and choosing **Team > Commit**. The **>** prefix will disappear. When you make new changes to the specified file, the prefix will appear again, indicating that it differs from the file stored in CVS.

To view version history

- To view the version history of a file, right-click on it in the **Navigator** and choose **Team > Show History**.

Setting Up Microsoft Visual Source Safe

In order to use VSS with Change Manager, you will need to download and install a plug-in that makes the two products compatible. You can download a VSS plug-in via the following link: http://sourceforge.net/projects/vssplugin.

Additionally, you will need to install Java Development Tools (JDT), which can be accessed via the **Find and Install** command in the Change Manager Main Menu.

To install JDT

1. Choose **Help > Software Updates > Find and Install...**. The **Find and Install** dialog appears.
2. Select **Search for New Features to Install** and click **Next**.
3. Choose **Eclipse 3.3 Update Site** or add it by clicking **New Remote Site** and entering the following URL: http://update.eclipse.org/updates/3.3*
4. Choose **Automatically Select Mirrors** and click **Finish**. Change Manager searches for the JDT and adds it to the system.
5. Choose **JDT (Java Development Tools)** and install it, then restart Change Manager.

To install the VSS plug-in

- Once you have downloaded the plug-in, close Change Manager and extract the plug-in zip file to Change Manager's **plugins** directory. When you restart Change Manager it will automatically be added to the application.

Using Microsoft Visual Source Safe

To connect a project to the repository

1 In the **Navigator**, right-click on the project you want to connect and choose **Team > Share Project...**

2 Select **VSS Configuration Wizard** and click **Next**.

1 Enter the VSS connection information as applicable and click **Finish**. The project is associated with VSS.

To check in files

1. In the **Navigator**, right-click on the file you want to add to VSS and select **Team > Commit Changes**. The **Commit Changes** dialog appears.
2. Click **OK**. The file is added to VSS.

To check out files

- In the **Navigator**, right-click on the file you want to check out of VSS and select **Team > Check Out**. The job is checked out of VSS and can be modified. Conversely, you can check in files via the **Team > Check In** command.

Viewing file differences against the repository version

- In the **Navigator**, right-click on the file you want to compare and choose **Compare With > Latest from VSS Repository...** The differences will be displayed, based on the description you specified the last time you checked out the file.

Reference

This section contains reference material you may find helpful when working in Change Manager. It is composed of the following topics:

- Change Manager Operations/DBMS Permissions Table
- Understanding Version Information

Change Manager Operations/DBMS Permissions Table

When Change Manager performs operations based on data source comparisons, there are minimal permissions required in some cases, dependent on data source platform and the action itself.

The following matrix describes common actions that the application might perform and lists the minimum requirements per DBMS, as necessary, for Change Manager to complete the given operation:

Operation Type	**Oracle**	**DB2**	**Sybase**	**SQL Server**
Schema Archive and Compare				
User's own and granted objects	No special permissions required	No special permissions required *	No special permissions required	No special permissions required **
Any User Object	SELECT_CATALOG_ROLE role	No special permissions required *	No special permissions required	No special permissions required **
Systems Objects	"DBA" role	No special permissions required*	No special permissions required	No special permissions required**
Settings	"DBA" role	No special permissions required	No special permissions required	No special permissions required
Alter Schema				
User's Own & Granted Objects			No special permissions required	No special permissions required
Any user object (regardless of owner)			dbo or alias to dbo	dbo
System objects	"DBA" role		sa_role and sp_configure "allow updates"	sysadmin and sp_configure "allow updates" and not allowed in 2005
Settings	"DBA" role		sa_role	sysadmin
Compare data	Select privilege on table or view	Select privilege on table or view or ownership *	Select privilege on table or view or ownership *	Select privilege on table or view or ownership **

Operation Type	Oracle	DB2	Sybase	SQL Server
Synchronize data	Insert/Update/Delete privilege on table or view	Insert/Update/Delete privilege on table or view	Insert/Update/Delete privilege on table or view or object ownership	Insert/Update/Delete privilege on table or view or object ownership
Compression compare				
Enabling compression compare	Create or Drop function, Create table, Java enabled on database	Create or Drop function, Create table, Java enabled on database	n/a ***	Create or Drop function, Create table, CLR enabled in database
Executing compression compare	As compare plus Execute function and needs to run as the same user who first executed the compression compare		n/a ***	

*** On Sybase, stored procedures cannot be hidden, and syscomment.text read must be allowed in order to compare or archive stored procedures.**

**** On Microsoft SQL Server, encrypted stored procedures cannot be viewed.**

***** The compression comparison function is unavailable for Sybase at this time.**

Understanding Version Information

The following section contains tables that outline the version information provided by Change Manager during the comparison, archive, or standard creation process.

- DB2
- Oracle
- SQL Server
- Sybase

The following version information is provided for DB2 data sources.

Parameter	Description
INST_NAME	Name of the current instance.
IS_INST_PARTIONABLE	Indicates if the current instance is a partitionable database server instance. Possible return values are 0 if it is not a partitionable database server instance, and 1 if it is a partitionable database server instance.
NUM_DBPARTITIONS	The number of database partitions. If it is not a partitioned database environment, this value is 1.

Parameter	Description
INST_PTR_SIZE	The bit size of the current instance (32 or 64).
RELEASE_NUM	The internal release number, as returned by the db2level command. For example, 03030106.
SERVICE_LEVEL	The service level, as returned by the db2level command. For example, DB2 v8.1.1.80.
BLD_LEVEL	The build level, as returned by the db2level command. For example, n041021.
PTF	The program temporary fix (PTF) identifier, as returned by the db2level command. For example U498350.
FIXPACK_NUM	The FixPak number, as returned by the db2level command. For example, 9.

On Oracle platforms, Change Manager provides a list of all products installed on the Oracle instance and their versions.

On SQL Server, the following parameters of note are included:

- Product Version (e.g. 9.00.3042.00)
- Platform (e.g. NT INTEL X86)
- Language
- Physical Memory
- Processor Count
- Processor Type
- Windows Version

NOTE: For entries that contain internal values, the numeric attribute is also available for standard comparisons. For example, the Product Version attribute has a character and an internal value of “8.00.2039” and 524288, respectively. Change Manager supplies the Product Version as “8.00.2039”, and the Product Version Internal Value as 524288.

On Sybase platforms, two groups of configuration parameters are supplied:

- The **Server Information** group displays the results of executing **sp_server_info**, where the attribute name is the configuration parameter and the value is its value.

- The **Version Information** group, which contains the following parameters:
  - Build Number
  - Build Type
  - Compilation Date & Time
  - Compilation Kernel
  - Compilation OS Bit Size
  - Compilation OS Platform
  - EBF Version
  - Post Build Type
  - Product
  - Product Code
  - Version

Glossary

A

Archive

A set of files, in an Embarcadero Change Manager-specific format, which represent a snapshot image of a database schema at a given point in time.

B

BCP Utility

Bulk Copy Utility. A command-prompt utility that copies data to or from an operating system file into a user-specified format.

C

Capture

The act of generating an archive file. A Capture job can be run on demand or scheduled to run at a later time.

Change Manager

An Eclipse-based application that provides comparison and synchronization functionality for schema, configuration settings, and data repository information between two or more data sources. The application also features archiving and standardization functionality.

Clone

A copy of a data source including all of the properties (thresholds, notifications, etc.) from the original data source.

Command Line Syntax Wizard

A feature that provides a command line interface for manual input of job commands to run Change Manager job functionality. The Command Line Syntax Wizard also provides a means to configure and run batch files for the purpose of running bulk comparison and archive jobs on an automated schedule.

Compare

The facility that identifies similarities and differences between servers, databases, database objects, and archives.

Compare Window

The window that opens after completing a Compare or Monitor job using the user interface (as opposed to a scheduled Compare or Monitor job). Also known as the **Compare Result Grid Window**. Database objects are displayed in grid format showing target and source data sources with the synchronization action required to synchronize the target to the source. From the Compare Window, you can filter the list of displayed objects, launch the Diff Analysis Window, open a Schema Compare Summary Report, generate synchronization scripts, re-compare selected objects, or review the comparison criteria.

Compliance View

A view that displays a history of all configuration comparison jobs. Jobs are listed by the last job that was executed and the last time the job failed.

Configuration Archive Editor

The interface through which configuration archives are captured in Change Manager. Once defined, archives are then used in configuration comparison jobs against live data sources and standards.

Configuration Comparison Job Editor

The interface through which configuration comparison job functionality is executed. The job editor enables the specification of multiple data source targets within a single configuration comparison job against one data source, archive, or standard.

Configuration Standard Editor

The interface through which configuration standards are defined in Change Manager. Once defined, standards are used in configuration comparison jobs against live data sources and archives.

D

Data Comparison Job Editor

The interface through which data comparison functionality is executed. The job editor enables the specification of data source targets in a data comparison job.

Data Diff View

Displays the values of mismatched row pairs as a result of a data comparison job. The view features row synchronization icons and the specific data values of the selected pair.

Data Source

A reusable connection to a specific database on a specific database server. A data source provides connection information, user name, and password to the desired database environment.

Data Source Explorer

Data Source Explorer provides an organizational tree of registered data sources and the comparison jobs and archives associated with them.

Diff Analysis Window

The Diff Analysis Window (also known as the Visual Differences Window or the Diff Window) is launched from the Compare Window after a Compare or Monitor job completes in the UI. The Diff Analysis window shows color-coded differences between each object on the source and target data sources along with the appropriate synchronization script.

E

Editor

A workbench interface component that enables Change Manager processes. That is, data, configuration, and schema comparison jobs.

Extended ALTER

A change which cannot be handled by a simple ANSI SQL ALTER statement, usually requiring that the database object be copied, dropped, and then re-created.

F

Filter System Objects List

A list that includes Embarcadero Change Manager's default system objects and any system objects you have added to that list. You have the option of ignoring the items on this list (filtering them out) when you run Capture, Compare or Monitor jobs.

G

Group

A folder used to organize registered data sources in Data Source Explorer. Groups provide an additional level of organization when maintaining a large quantity of data sources.

H

There are no entries for this letter.

I

Imported Archives

Archives created on a different machine (and possibly by a different user) that you have copied into your Embarcadero Change Manger environment.

J

Job

An application process that involves the creation of an object or element, or the comparison of an object or element. In some cases, such as data comparison jobs, a comparison is made (the data repositories of two or more data sources), and an object is created (the job definition is retained in Change Manager for reuse.)

Job Explorer

Provides a tree that lists all jobs defined and saved in Change Manager. Jobs are organized by date created, source and target platform type, last modified, last executed, job name, data source name, and job type.

Jobs Window

The Jobs Window presents an overview of all Compare, Capture, Monitor, and Migrate jobs in an expandable grid format.

K

There are no entries for this letter.

L

Login

A login is an form of identification that lets you connect to a data source. Permissions within specific databases are controlled by user accounts. The database administrator maps your login account to a user account in any database you are authorized to access.

Logins are one of the fundamental security mechanisms of Embarcadero Change Manager. Users who connect to a server must identify themselves using a specific login identifier (ID). Users can then only see the tables and views they are authorized to see.

M

Migrate

The facility that allows you to move schema archives and database object definitions to one or more databases.

Monitor

The facility that combines the capture and compare operations into a single job. Monitor jobs create an archive for a data source, and then compares the last archive against the next-to-last archive (latest version -1).

N

Navigator View

A view that provides a tree of all data comparison job synchronization scripts that are generated in the Data Comparison Job Editor. Scripts are launched directly from the view for execution and modification purposes.

Notification

Notifications are Job monitoring messages sent through email to the addresses you identify in the Options Editor.

O

There are no entries for this letter.

P

There are no entries for this letter.

Q

There are no entries for this letter.

R

There are no entries for this letter.

S

Schema

The definition or structure of data or database objects.

Shell Data Source

An empty data source with only the name and DBMS type defined. A shell data source is created during the import archive function when the data source specified in the imported archive does not exist. The connection information, user name, and password is intentionally missing from the shell data source for security and data integrity purposes.

SQL Editor

An interface that provides a means to view and modify the SQL synchronization scripts generated by the Data Comparison Editor after a data comparison job has been executed.

SQL Errors View

Displays any errors encountered while executing synchronization scripts in Change Manager.

Standard

Used to provided customized data source configuration properties and are used in configuration comparison jobs against data sources and archives to provide a specific set of values and threshold operators that check enterprise settings and ensure all system configuration values fall within acceptable ranges.

Synchronization Script

Synchronization scripts are automatically generated by the data comparison job editor when a data comparison job is executed and, when run, synchronize the data repositories of two or more data sources where disparities are discovered by the job process.

Synchronization Script Navigator

A view that provides a tree of all synchronization scripts generated during the execution process of comparison jobs. Scripts can be modified and executed from the Synchronization Script Navigator as required.

T

There are no entries for this letter.

U

There are no entries for this letter.

V

View

A workbench interface component that is used to navigate a hierarchy of information, open editors, or display properties.

W

Welcome Page

Provides a series of links and icons that are clicked to access tutorials, cheat sheets, and other information. The Welcome page is intended to provide access to information regarding Change Manager tasks and processes, and its design is aimed particularly towards orientating new users.

Wizard

A series of User Interface panels that lead you through a task.

Workbench

The Change Manager development environment. It provides the interface to create, manage, and navigate comparison jobs via an array of views, editors, and menus.

X

There are no entries for this letter.

Y

There are no entries for this letter.

Z

There are no entries for this letter.

Index