

# SQL Data Compare 7

July 2008

Note: these pages apply to a version of this product that is not the current released version.

For the latest support documentation, please see <http://documentation.red-gate.com>

# Contents

---

Getting started .....	3
Worked example: synchronizing data in two databases .....	5
Worked example: restoring from a backup file .....	6
Working with projects.....	7
Project configuration .....	10
Setting data sources.....	12
Selecting tables and views .....	16
Mapping objects.....	18
Setting project options .....	20
Setting application options .....	24
Viewing the comparison results.....	25
Viewing the row differences.....	27
Viewing the data .....	29
Exporting the comparison results .....	31
Printing the comparison results.....	33
Setting up the synchronization.....	34
Using the synchronization wizard .....	36
Warnings .....	43
Starting SQL Data Compare with a specific project .....	45
Case-sensitive comparisons.....	46
Filtering the comparison with a WHERE clause .....	47
Synchronizing views .....	49
Comparing databases on different SQL Server versions.....	50
Creating a rollback script .....	51
Resolving mapping errors.....	52
Common error messages .....	56
Troubleshooting .....	58
Getting started with the SQL Data Compare command line.....	61
Basic command line features .....	62
Integrating the command line with applications .....	65
Frequently asked questions for the command line .....	92
Working with backups.....	94
Acknowledgements .....	95

## Getting started

---

SQL Data Compare enables you to compare and synchronize the data in two Microsoft® SQL Server™ databases. You can also compare a backup with a database, a scripts folder, or another backup. With SQL Data Compare, you can synchronize multiple tables whilst maintaining referential integrity.

This is useful, for example, to identify the differences between the contents of two databases following a failed replication; or to restore selected rows from a backup.

You can use SQL Data Compare to compare and synchronize data in SQL Server 2008, SQL Server 2005, SQL Server 2000, and SQL Azure databases.

### SQL Data Compare: step-by-step

1. Select a databases or database objects to compare (page 12)
2. View the comparison results (page 25)
3. Synchronize your databases or create a synchronization script

### Worked examples

Learn more about SQL Data Compare by following one of these detailed examples:

- Synchronizing data in two databases (page 5)
- Performing table-level and row-level restores from a backup (page 6)
- Deploying a database from source control

### SQL Server Management Studio integration

SQL Data Compare includes a free add-in for SQL Server Management Studio that enables you to set data sources to compare and synchronize from within SQL Server Management Studio.

For more information, see: [Using the add-in](#).

### Technical notes

SQL Data Compare's mapping features (page 18) allow you to compare any tables with compatible data types.

For more information, see [Which data types can be compared](#).

You can create a rollback script (page 51) to undo data synchronization.

You can filter the comparison using a WHERE clause (page 47).

SQL Data Compare can take time to compare and synchronize data sources. Speed depends primarily upon disk write speed, processor speed, memory, and the size of the

databases. However, there are a number of ways to improve the performance of SQL Data Compare (<http://documentation.red-gate.com/display/SDC104/Getting+better+performance+out+of+SQL+Data+Compare>).

## Worked example: synchronizing data in two databases

---

This worked example demonstrates a basic comparison and synchronization of two SQL Server databases.

In the example, the Magic Widget Company has a SQL Server database running on a live web server. This database contains a number of tables, views, stored procedures, and other database objects. The Magic Widget Company's development team has been working on an upgrade to their website. As part of this upgrade, they have made a number of changes to the database, which need to be transferred to the production database.

You can follow the example on your own system. You will need access to a SQL Server to do this.

This example has three steps:

1. Set up the comparison

Create the example databases, and specify the data sources, tables, and views you want to compare.

2. Select data to synchronize

Review the results and select the rows you want to synchronize.

3. Synchronize the databases

Create and run a synchronization script.

The worked example uses the following sample databases:

- WidgetDev is the development database
- WidgetLive is the production database

## Worked example: restoring from a backup file

---

This worked example demonstrates a table-level restore from a backup file. You can also restore specific rows.

In the example, the Magic Widget Company has a SQL Server database running on a test web server. The Magic Widget Company's test team has been testing the new version of the web site.

One of the software testers has updated the Contacts table, intending to update one email address. They did not specify a WHERE clause, and consequently have updated the entire table. During the day some other rows in the Contacts table have been modified. The database administrator has been asked to restore the data from a backup and apply some but not all of the changes that were made to the test server.

You can follow the example on your own system, if you are using the SQL Data Compare Professional edition. You will need access to a SQL Server to do this.

If you have not already followed the Comparing and synchronizing two databases (page 5) worked example, you are recommended to do so before starting this worked example.

This example has three steps:

1. Set up the comparison  
Create the example databases, and specify the data sources you want to compare.
2. Select rows to restore  
Review the results and select the rows you want to restore.
3. Synchronize the data sources  
Create and run a synchronization script.

The worked example uses the following data sources:

- WidgetTest is the test database
- WidgetLive is the database used to create the backup
- BeforeEmailUpdate.bak is the backup file

## Working with projects

---

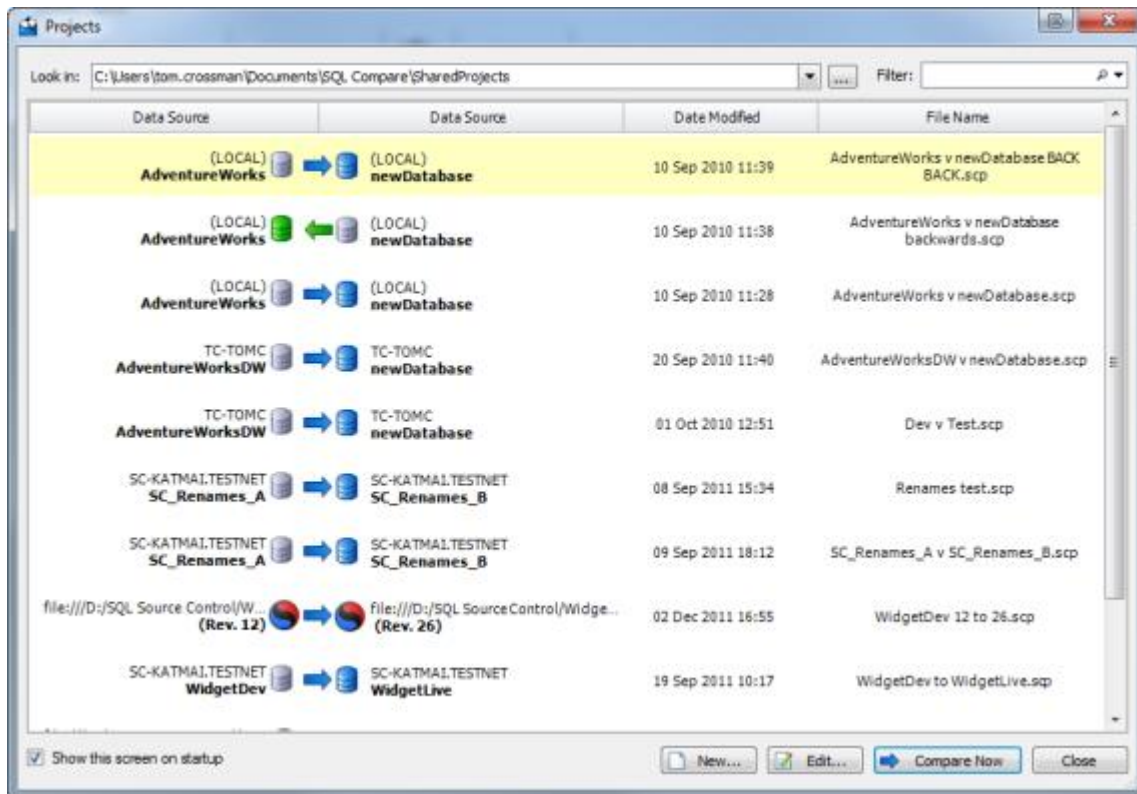
Whenever you compare databases, you set up a *project*. If you have any existing projects, the Project Configuration dialog box for your most recently used project is displayed when you start SQL Data Compare.

A project contains details of:

- which data sources you selected  
If you selected a backup as a data source, the project saves details of the backup set you selected.
- the connection details for your data sources
- which project configuration options you selected
- which objects you have selected for deployment
- your owner mappings
- your object mappings (page 18)
- which tables and views (page 16) you selected
- any WHERE clause (page 47) you have used to filter the results

### Finding a project

On the toolbar, click  (Open Project) to display the **Projects** dialog box:




The **Projects** dialog box shows details of your projects.

You can edit or compare these projects, or create a new project.

## Creating and editing a project

To create a new project, click  **New**

To edit the current project, click  **Edit**

To open and edit an existing project, double click the project on the **Projects** dialog box.

## Saving a project

SQL Data Compare does not automatically save projects.

To save a project, on the Project Configuration dialog box click **Save**. Alternatively, you can save a project when you are reviewing its comparison results. To do this, on the **File** menu click **Save Project**.

If there are unsaved changes in the current project, you will be prompted to save when you create a new project, open another project, or when you close the application.



## Copying a project

To make a copy of a project, on the **Projects** dialog box, select the project, right click, and select **Create Clone**.

Alternatively, open the project that you want to copy, and on the Project Configuration dialog box, click **Save As**.

To copy a project when you are reviewing the comparison results, on the **File** menu, click **Save Project As**.

## Project compatibility

You can open a SQL Data Compare 7, 6, or 5 project in version 8. Projects from earlier versions are not supported.

You cannot open a SQL Data Compare 8 project in earlier versions.

If you open and save a project in SQL Data Compare 8, it is converted to version 8 and cannot be opened in earlier versions.

You can open a SQL Compare 8.1 (or later) project in SQL Data Compare 8.0 or later.






You can open a SQL Data Compare 8.0 (or later) project in SQL Compare 8.1 or later.

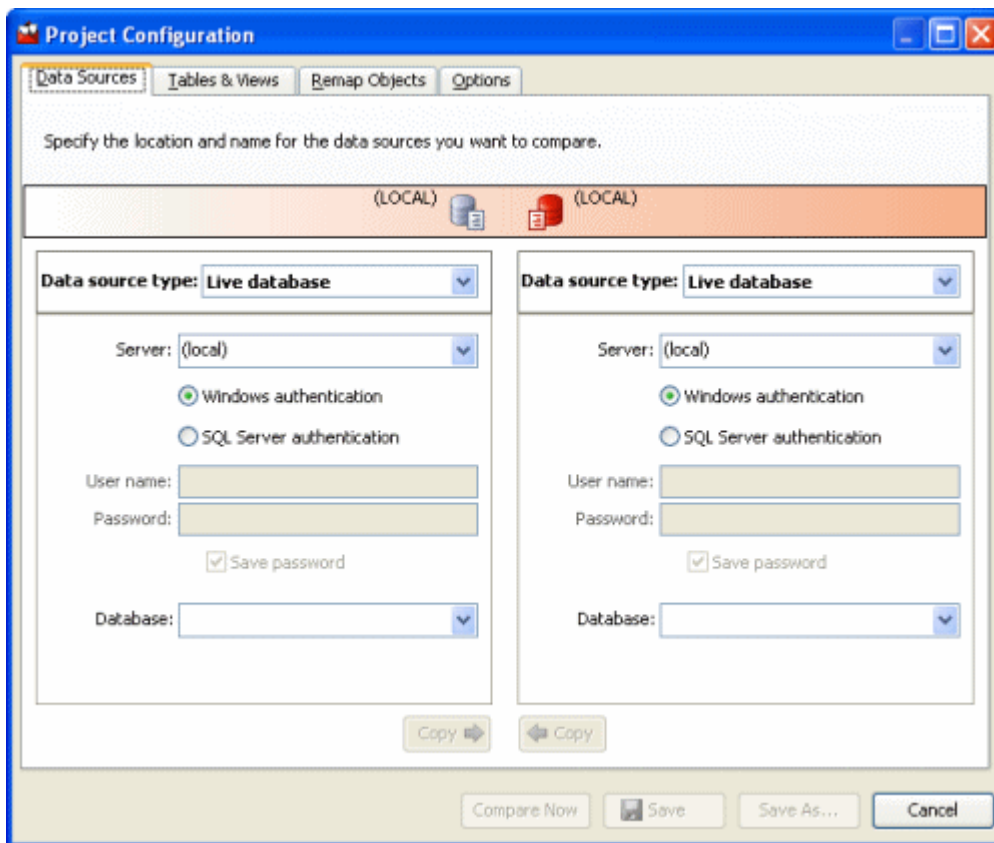
## Project configuration

---

Whenever you compare data sources, SQL Data Compare requires information about what to compare in the two data sources. You enter this information using the **Project Configuration** dialog box.

To display the **Project Configuration** dialog box:

- for the current project, click  **Edit Project**
- for a new project, click  **Comparison Projects**, and click  **New**
- for an existing project, click  **Comparison Projects**, select the project, and then click  **Edit**



The **Project Configuration** dialog box displays the following tabs:

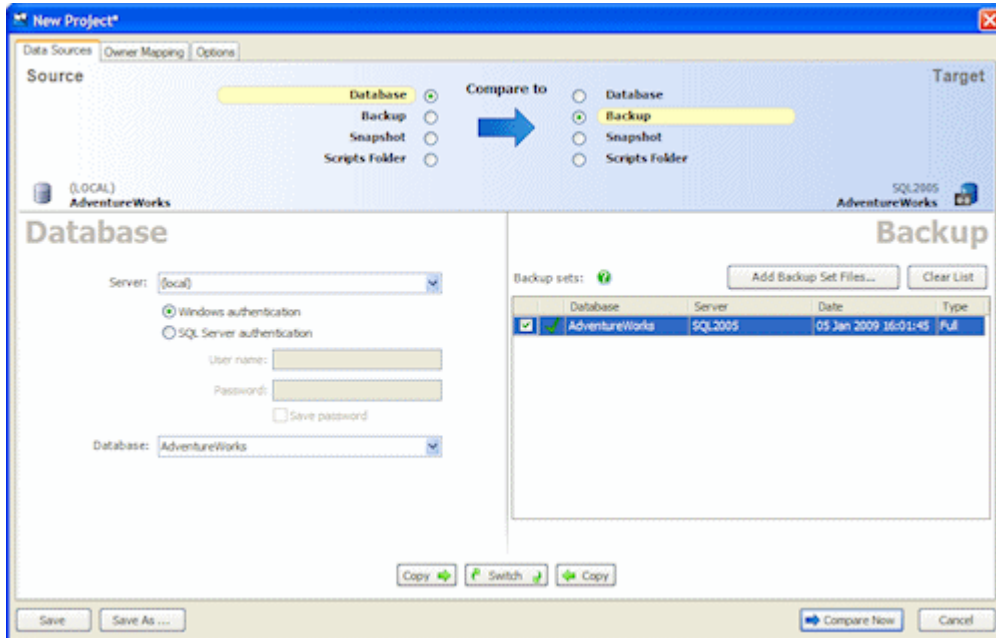
- **Data Sources** enables you to specify the data source details.  
See Setting data sources (page 12).
- **Tables & Views** enables you to select the tables, views, and columns for comparison.  
See Selecting tables and views (page 16).

- **Remap Objects** enables you to map the objects that SQL Data Compare was unable to map automatically.  
See Mapping data sources (page 18).
- **Options** enables you to edit the mapping, comparison, and synchronization options.  
See Setting project options (page 20).

## Setting data sources

---

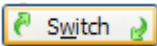
When you create a new comparison project, SQL Data Compare requires information about which two data sources you want to compare, and how to connect to them. You enter this information on the Project Configuration dialog box.



### Selecting data sources

Specify the two data sources you want to compare in the **Data Sources** tab. You specify a *source* and a *target*.

The *source* is the data source that will not change. The *target* is the data source that will change.

To switch the source and target, click 

You can compare the following types of data source:

- Databases  
Any database you can connect to on a SQL Server.
- Backups  
Native SQL Server backups or Red Gate SQL Backup backups. You can specify a backup as a data source only if you are using SQL Data Compare Professional edition.
- Scripts folders  
Scripts folders created using either SQL Data Compare or SQL Compare professional edition.

You can compare any combination of data source types in your project.

### Selecting a database

1. Under Source or Target, select **Database**.
2. Type or select the name of the SQL Server in the **Server** box.

If you experience problems selecting a SQL Server that is not running on the LAN, for example if you are accessing the SQL Server via an Internet connection, you may need to create an alias to the SQL Server using TCP/IP (refer to your SQL Server documentation for details). You can then type the alias name in the **Server** box to connect to the remote SQL Server.

To refresh the **Server** list, right-click the box and click **Refresh**, or scroll to the top of the list and click **Refresh**.

3. Select the authentication method, and for **SQL Server authentication** enter the **User name** and **Password**.

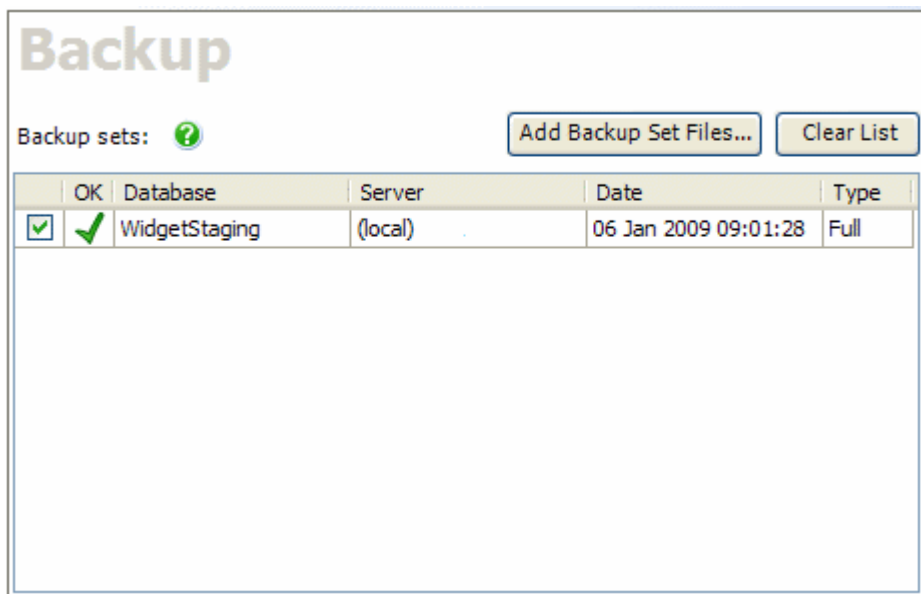
If you want SQL Data Compare to remember your password, select the **Save password** check box.

4. In the **Database** box, type or select the name of the database.

To refresh the **Database** list, right-click the box and click **Refresh**, or scroll to the top of the list and click **Refresh**.

### Selecting a backup

1. Under Source or Target, select **Backup**.



2. Click **Add Backup Set Files** to select all the files making up the backup set you want to compare.

To specify a network path, type the full path, including the server name, for example:





\\ServerName\MyFolder\MyFile

If any of the files you add are encrypted, the **Decrypt Backup Files** dialog box is displayed. Enter the password to decrypt these files.

To use a differential backup as a data source, you must also add the associated full backup.

Note that SQL Data Compare does not support using partial, filegroup, or transaction log backups as a data source.

When you have added a backup set, one of the following icons is displayed:

-  The backup set is valid and complete.  
Select the check box to use this backup as a data source.
-  The backup set you have selected cannot be used as a data source.  
This error is shown if the backup is corrupted, or if you have selected a partial, filegroup, or transaction log backup.
-  One or more files in the backup set is encrypted.  
Click the padlock icon to display the **Decrypt Backup Files** dialog box.
-  Either the backup set is incomplete, or a differential backup has been added without the corresponding full backup.

For more information, see Working with backups (page **Error! Bookmark not defined.**)

## Selecting a scripts folder

1. Under Source or Target, select **Scripts Folder**.

In the **Scripts Folder** box, select the folder, or click  to browse to the folder.

You can use a scripts folder as a data source only if it contains a database schema. You can create scripts folders using SQL Compare, and use SQL Data Compare to update them with static data. Alternatively, you can create a scripts folder using SQL Data Compare.

To create schema scripts in a folder using SQL Data Compare:

1. On the **Data Sources** tab of the Project Configuration dialog box, select a database, backup, or existing valid scripts folder as the *source*.
2. Under *Target*, select Scripts folder.
3. In the **Scripts folder** box, Browse to or select the folder you want to use.  
If the folder does not contain valid scripts folder metadata, the option to create schema scripts is enabled.
4. Click **Create Schema Scripts**.

The **Create New Scripts Folder** dialog box is displayed:

**Create New Scripts Folder**

**Create scripts folder**

Data source details

Data source type: Database

Server: (local)

Windows authentication  
 SQL Server authentication

User name:

Password:

Database: SprocketProduction

Scripts folder properties

New folder name: SprocketProduction [Script creation options...](#)

Create in: Z:\Work\Scripts Folders

Auto detect case sensitivity  
 Treat items as case sensitive  
 Decrypt encrypted objects on 2005 and 2008 databases

To update the target scripts folder with data from the source, perform the comparison, select the rows you want to include, and then synchronize.

For more information, see Working with scripts folders

## Selecting tables and views

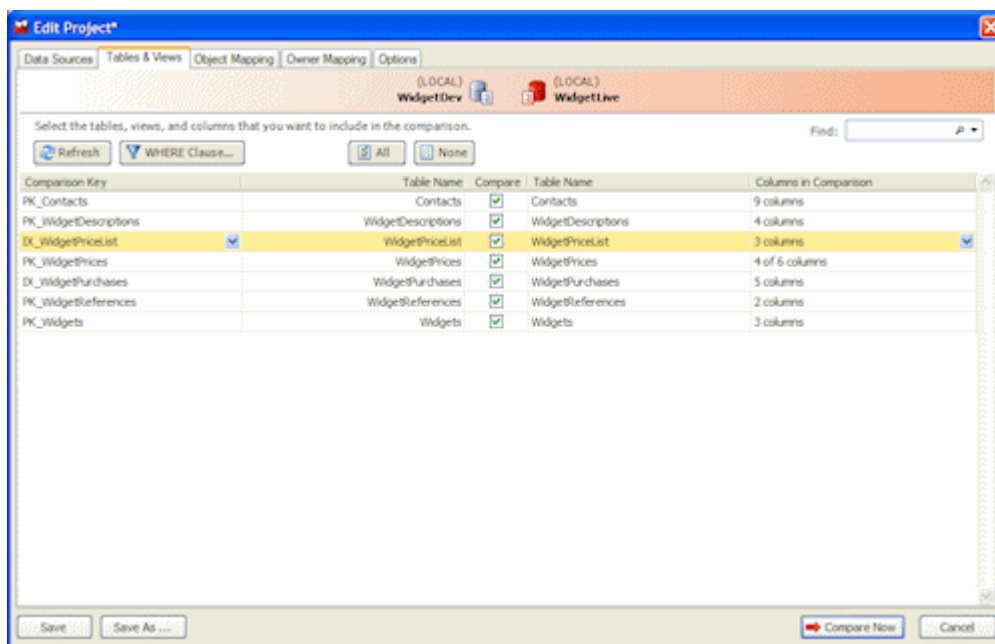
---

When you create a project and you have selected your data sources, you can specify which tables, views, and columns to compare. You enter this information using the **Tables & Views** tab on the Project Configuration (page 10) dialog box.

The **Tables & Views** tab enables you to:

- select the comparison key for each table or view
- select the tables and views that will be compared
- select the columns that will be compared

SQL Data Compare lists the tables and views in the source and target. Tables and views with identical or similar names are displayed side-by-side:



Note that:

- You can filter the specific rows that will be compared by entering a WHERE clause. Filtering can improve the performance of SQL Data Compare.


To filter rows, click **WHERE Clause**. The **WHERE Clause Editor** dialog box is displayed.

For more information, see Filtering the comparison results with a WHERE clause (page 47)

- Views are listed only if the data source is a database and the project option **Include views** is selected.
- You can change the order in which the tables and views are listed by clicking a column header.

To sort by multiple columns, click a column header, then hold down SHIFT and click another column header.



- Only the tables and views that are mapped are listed.  
If you are setting up a new project and SQL Data Compare is unable to map a table or view, you can map the tables and views manually.  
For more information, see [Mapping objects \(page 18\)](#)
- If you are editing an existing project and the structure of the database has changed since you last ran the project, the mappings may be incorrect. In this case, a warning symbol  is shown to indicate that those changes affect your project configuration.  
For more information, see [Mapping errors \(page 52\)](#)

## Mapping objects

---

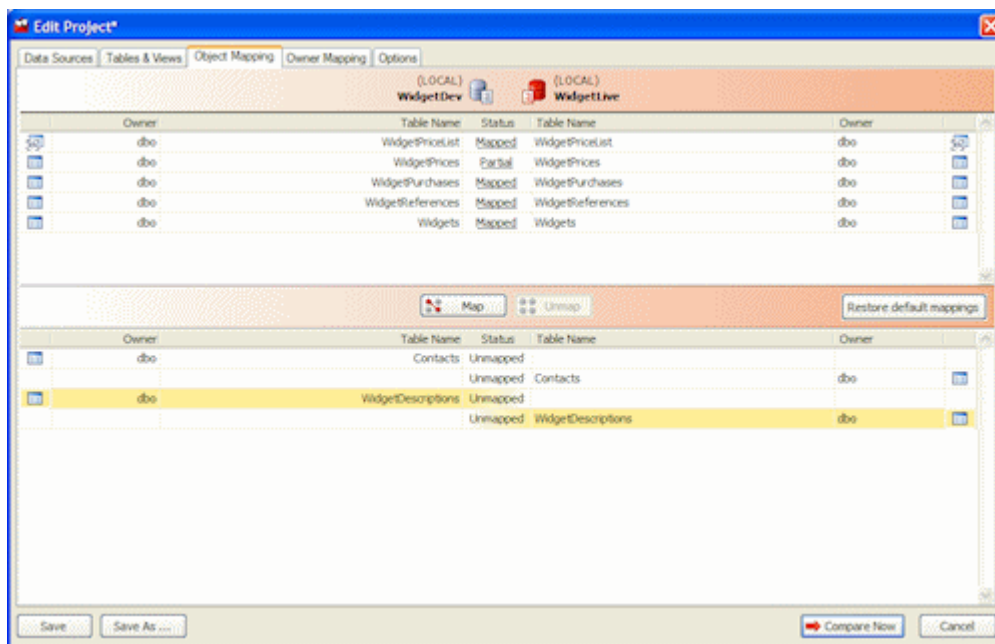
When you have selected your data sources, SQL Data Compare automatically maps tables and views with the same name and schema (owner).

However, if there are schema differences between the data sources, for example if two tables have different names, they may not be mapped automatically.

This topic provides information on:

- Mapping tables and views
- Mapping specific columns

To compare tables and views that are not automatically mapped, click the **Object Mapping** tab of the Project Configuration dialog box:




The upper pane displays tables and views that are fully **Mapped** or have **Partial** mapping. The lower pane displays **Unmapped** tables and views.

Note that:

Views are listed only if the data source is a database and the project option **Include views** is selected.

- If an object has a Partial mapping, some of its columns are not be mapped, and cannot be compared.

To set the column mappings for an object, click the **Status** box for the object you want to re-map.

- If the differences between objects are not significant, they are more likely to be automatically mapped if you select the options **Ignore case of object names**, **Ignore spaces in object names**, and **Ignore underscores in object names**
- If you are editing an existing project and the structure of the database has changed since you last ran the project, a warning symbol  is shown to indicate that those changes affect your project configuration.


## Setting project options

---

The project configuration options enable you to modify the behavior of SQL Data Compare. For example, you can set an option so that SQL Data Compare ignores certain objects even if they are different, or so that it does not script certain properties for synchronization (such as the collation order on columns).

When you create a new project, you should run the comparison with the default options, then review your comparison results. However, if your database is on a SQL Server with case-sensitive sort order, you must select the *Treat items as case sensitive* option. When you have reviewed your comparison results, you may then want to consider changing some of the options.

The options you set are saved for each project, and are modified on the Project Configuration dialog box.

To display the Project Configuration dialog box, click  (Edit Project), or select **Project Options** from the Tools menu.

The options you can set are described below. Note that some of the options apply only to the comparison; they do not affect the synchronization. Similarly, some options apply only to the synchronization. Options affecting mapping and comparison (for example, those relating to case sensitivity) are not applied instantly. You must re-compare the data sources to apply these options.

### Default options

To save the current selection of options as your defaults, click **Save As My Defaults**. To restore your defaults, click **My Defaults**. The saved defaults will be used for all new projects.

To reset all the options to their original settings, click **Red Gate Defaults**. The default options for a comparison project are as follows:

- Ignore spaces in object names
- Include identity columns
- Include timestamp columns
- Show identical values in results
- Disable foreign keys
- Reseed identity columns
- Include comment header in the synchronization script

### Use case sensitive object definitionsql

Considers the case of the object names (tables, views, users, roles, schemas, indexes, and columns) when mapping. For example, [dbo].[Widget] will be mapped to [dbo].[wIDgEt].

Note that

- if the databases that you are comparing are running on a SQL Server that uses case-sensitive sort order, you should ensure that this option is selected.
- if you compare a SQL Azure database with this option selected, SQL Data Compare may highlight false differences.

### **Ignore spaces in object names**

Ignores spaces in object names (tables, views, users, roles, schemas, indexes, and columns) when mapping. For example, [dbo].[Widget Prices] will be mapped to [dbo].[WidgetPrices].

### **Ignore underscores in object names**

Ignores underscores in the object names (tables, views, users, roles, schemas, indexes, and columns) when mapping. For example, [dbo].[Widget\_Prices] will be mapped to [dbo].[WidgetPrices].

### **Include views**

Includes views in the comparison. Generally, views can be synchronized only if the referenced rows are from a single table, and the referenced columns are simple (for example, they must not include identity columns or computed columns).

### **Include identity columns**

Includes identity columns in the comparison. You cannot synchronize a view if it includes an identity column.

### **Include timestamp columns**

Includes timestamp columns in the comparison. Timestamp columns cannot be synchronized.

### **Trim trailing spaces**

If the data in two columns differs only by the number of spaces at the end of the string, SQL Data Compare considers the data to be identical. This option does not apply to CLR or XML columns.

Trailing spaces are ignored during synchronization, if this option is selected.

### **Force binary collation (case-sensitive)**

For all string data types, forces binary collation irrespective of column collation, resulting in a case-sensitive comparison. When this option is selected and the comparison key is a string, this may result in slower performance because the indexes are not used.

## Show identical values in results

If this option is not selected, identical values will not be stored on disk nor appear in the comparison results.

## Use checksum comparison

Performs a checksum prior to comparison. The data is compared only if the checksums differ. Note that if the data differs only in text or image columns, the checksums will be identical and the data may be flagged incorrectly as identical.

On SQL Server 2000 db\_owner permissions are required.

You cannot select this option when comparing scripts folders.

You cannot select this option if the **Show identical values in results** project option is selected.

## Compress temporary files

Compresses the temporary files that SQL Data Compare generates while performing the comparison. This reduces the possibility of running out of temporary disk space when comparing very large databases.

When you select this option, you will not be able to sort the results of the comparison by clicking on a column header in the **Row Differences** pane.

## Disable foreign keys

Disables then re-enables foreign keys in the synchronization SQL script. Note that in some circumstances foreign keys will be dropped and recreated rather than disabled and re-enabled.

## Drop primary keys, indexes, and unique constraints

Drops then recreates primary keys, indexes, and unique constraints in the synchronization SQL script.

Note that:

- if the primary key, index, or unique constraint is the comparison key, it cannot be dropped.
- if you synchronize to a SQL Azure database, clustered index constraints are not dropped.

## Don't use transactions in SQL scripts

Does not insert BEGIN TRANSACTION at the beginning of the synchronization SQL script and COMMIT TRANSACTION at the end of the synchronization SQL script.

## **Transport CLR data types as binary**

Uses the binary representation of CLR types in the synchronization SQL script.

Note that:

- if this option is not selected SQL Data Compare uses the string representation.
- if you synchronize user-defined CLR types to a SQL Azure database, this option has no effect; SQL Azure does not support user-defined CLR types.

## **Disable DML triggers**

Disables then re-enables DML triggers on tables and views in the synchronization SQL script.

## **Disable DDL triggers**

Disables then re-enables DDL triggers in the synchronization SQL script.

## **Reseed identity columns**

Reseeds identity columns so that identity values in the database you are updating match values in the source database.

Note that if you synchronize to a SQL Azure database, this option has no effect; SQL Data Compare does not reseed identity values.

## **Include comment header in the synchronization script**

Includes a comment at the beginning of the synchronization script. The heading contains information about the data sources being synchronized, and the version of SQL Data Compare.

## **Don't include comments in the synchronization script**

If comments are included, it is easier to locate objects in the synchronization script. However, the script is smaller if comments are not included.

## **Force constraints to be re-enabled with CHECK**

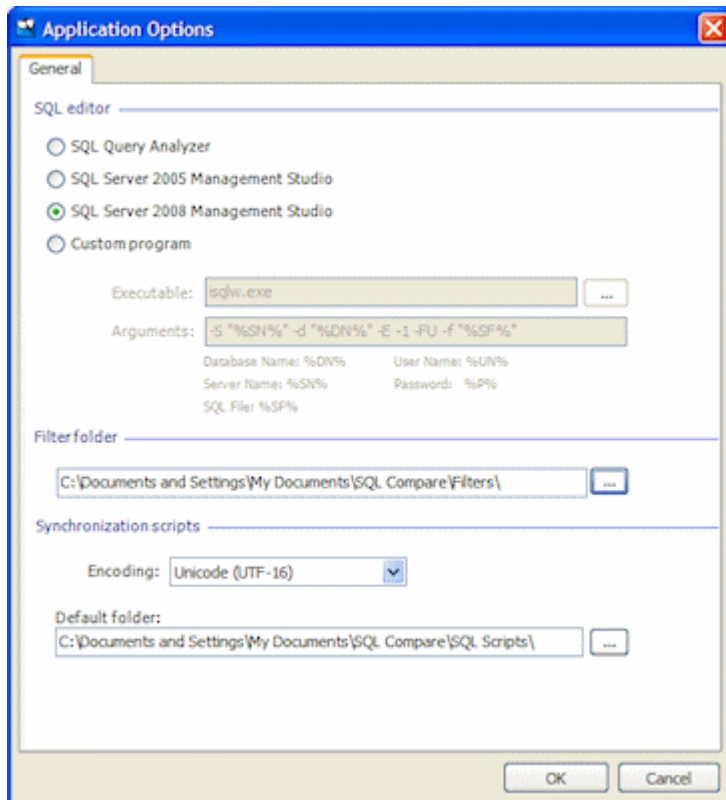
Forces any constraints (for example, those on foreign keys) disabled by SQL Data Compare to be re-enabled with CHECK.

## Setting application options

---

The **Application Options** dialog box enables you to specify options such as the default editor for opening the synchronization script, or the location where that script is saved.

To display the **Application Options** dialog box, on the **Tools** menu, click **Application Options**.



### Split transactions

By default, the synchronization script is enclosed in a single transaction. To use multiple transactions, select the application option **Split transactions**


You can specify the amount of data included in each transaction using the **Maximum transaction size** box.

If you do not want to use transactions in the synchronization script, select the project option **Don't use transactions in the synchronization script**



## Viewing the comparison results







When you have compared the data sources, SQL Data Compare displays the comparison results in the upper (Results) pane. The upper pane displays all the tables, views, and groups of row differences you can select for synchronization.

To compare the data sources again using the same project configuration, and update the comparison results, click  **Refresh**

Objects are displayed grouped by how they differ between the two data sources, by whether they match what is typed in the **Find** box, or ungrouped. When you first run the comparison, the objects are grouped by *Type of difference*.




### Object groups

To view the tables or views in a group, click  or click the grouping bar.

Type	All Different	Table Name				Table Name	All Identical
6 tables or views with differences in their rows							
	<input checked="" type="checkbox"/> 7	Contacts	0	<input checked="" type="checkbox"/> 7	0	Contacts	93
	<input checked="" type="checkbox"/> 3	WidgetDescriptions	0	<input checked="" type="checkbox"/> 2	<input checked="" type="checkbox"/> 1	WidgetDescriptions	0
	<input checked="" type="checkbox"/> 4	WidgetPriceList	<input checked="" type="checkbox"/> 3	0	<input checked="" type="checkbox"/> 1	WidgetPriceList	2
	<input checked="" type="checkbox"/> 4	WidgetPrices	<input checked="" type="checkbox"/> 3	<input checked="" type="checkbox"/> 1	0	WidgetPrices	2
	<input checked="" type="checkbox"/> 11010	WidgetPurchases	0	<input checked="" type="checkbox"/> 854	<input checked="" type="checkbox"/> 10156	WidgetPurchases	97425
	<input checked="" type="checkbox"/> 2	Widgets	<input checked="" type="checkbox"/> 1	0	<input checked="" type="checkbox"/> 1	Widgets	6



If you want to display the comparison results in a single list, in the **Group by** box, select *No groups*.

When objects are grouped by *Type of difference*, the **Type** column indicates the difference:





-  tables or views with differences in their rows
-  tables or views with identical rows only
-  tables or views that could not be compared

### Tables or views with differences in their rows

The comparison results are displayed as follows:

- Type** indicates the type of object; a table  or view 
- Different** displays the total number of differences for the object. Synchronize all of these rows to make the table or view identical.

Use the check box to include or exclude all of the rows for synchronization.

- **Table Name** displays the name of the table or view.
-  displays the number of rows for the table or view that exist in the source but not the target.  
Use the check box to include or exclude these rows for synchronization.
-  displays the number of rows for the table or view that exist in both databases but are different.  
Use the check box to include or exclude these rows for synchronization.
-  displays the number of rows for the table or view that exist in target but not the source.  
Use the check box to include or exclude these rows for synchronization.
- **Identical** displays the total number of rows in the table or view that are identical.
- **Actions**  displays a drop-down menu that enables you to select or clear the various check boxes.  
The **Actions** menu is not available if you are viewing the comparison results in a single list.  
For more information, see Setting up the synchronization (page 34)

### Tables or views with identical rows only




The comparison results are displayed as follows:

- **Type** indicates the type of object; a table or view.
- **Identical** displays the total number of identical rows within the table or view.

### Finding tables and views

To locate objects, type the search text in the **Find** box. To select a recent search, click the **Find** arrow button ▼

As you type, objects are grouped in the upper pane by whether they match or do not match what you type:

Type	Object Name	<input type="checkbox"/>	Object Name
3 objects match 'widget'		<input type="checkbox"/>	0 of 3
Table 	WidgetPrices	<input type="checkbox"/>	WidgetPrices
Table 	WidgetReferences	<input type="checkbox"/>	WidgetReferences
Table 	Widgets	<input type="checkbox"/>	Widgets
12 objects do not match 'widget'		<input type="checkbox"/>	0 of 2 (12)

SQL Data Compare searches object names and owners (schemas).

Note that the search is not case-sensitive.




To clear the **Find** box click the **x** button.

## Viewing the row differences


The lower (Row Differences) pane displays a side-by-side listing of values for the tables and views you have compared. To display the lower pane, click a table or view in the list of tables.



By default, all rows with differences are selected for synchronization. Use the **Include** check boxes to include or exclude the rows.


-  Rows that contain differences are displayed with a shaded background
-  Values that are different are displayed with a darker shaded background
-  Values that exist in one database but do not exist in the other are displayed with a shaded, patterned background.

In the example above, for the row where WidgetId is 2:



- **Description** is the same in both databases so the data is displayed in gray text
- The **Picture** values are different so they are displayed with the darker shaded background
-  WidgetID is the comparison key.

Shading on the column names also indicates differences. A column name with full shading indicates values that are different; a column name with partial shading indicates values that only exist in one of the data sources. In the illustration above, the **Picture** column contains values that are different so the column name is displayed with full shading.

Note that:

- The lower (Row Differences) pane cannot display values for **tables or views that could not be compared**.  
For more information, see Tables and views that cannot be compared
- If you have cleared the **Show identical values in results** project option, SQL Data Compare does not show rows that are identical and a warning symbol  is displayed.
- If you have selected the project option **Compress temporary files**, you will be unable to sort the comparison results by clicking on a column header.  
For more information, see Setting project options (page 20)

## Searching the data

To search the values, in the **Find** box, type your search text, then click  or  to find the next or previous match. If the **Find** box is not already displayed, right-click and click **Find**.

## Viewing the object synchronization script

To view the deployment script for the selected object, right-click in the SQL Differences pane, and then click **Show Object Deployment Script**.

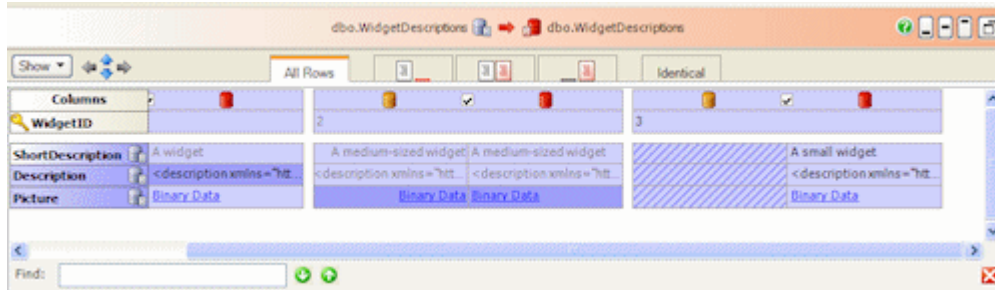
The **Single Object Deployment Script** dialog box is displayed.

You are recommended to deploy only using the full deployment script created by the Deployment wizard, because the single objects script cannot account for dependencies. If you use the object deployment script to deploy a single object, there may be unexpected results, or the deployment may fail.

To view the synchronization SQL that will make all the selected tables and views in the two data sources identical, use the Synchronization wizard.

## Pivoting the data

To rotate the data so that you can view the values for a record as a group, click **Show**, and select **Pivot View**; the values are displayed vertically. This is useful when there are a large number of columns in a table.



Note that you cannot change the sort order of the columns when you are viewing the data in the *pivot* view.

To scroll through the records, hold down SHIFT and rotate the mouse wheel.

To show the next or previous difference, use the arrow buttons or press the shortcut keys; the values that differ are shown with a green background.

## Viewing the data

The **Viewer** dialog box displays value details.

For more information, see Viewing the data (page 29).

## Viewing the data

---

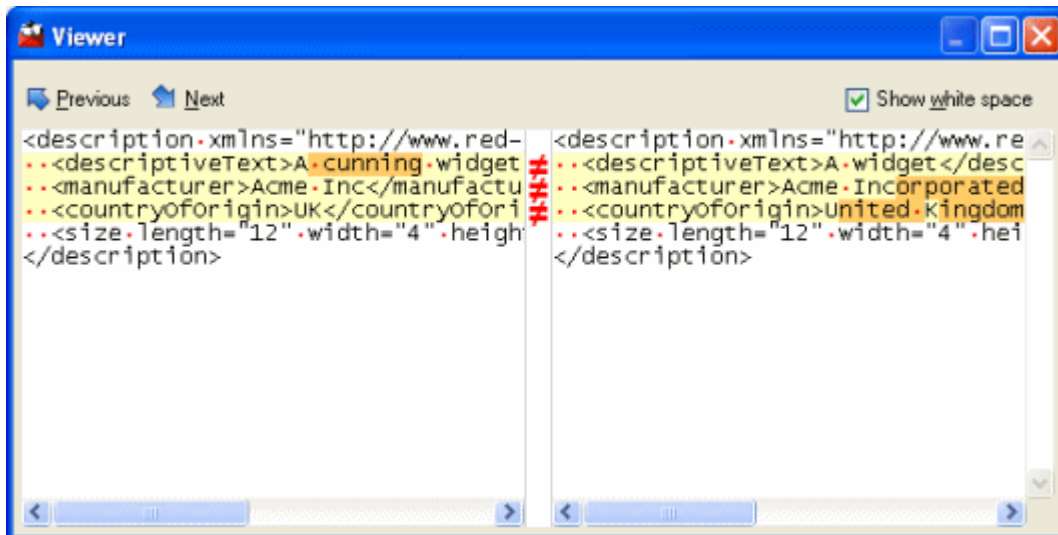
The **Viewer** dialog box displays value details. To display the **Viewer** dialog box, double-click a value in the lower (Row Differences) pane.

For example, if a value exists in only one of the data sources:



```
<description xmlns="http://www.red-gate.com/widgets/widgetDescriptions"
  <descriptiveText>This widget has been discontinued. It was last sold
  <manufacturer>Magic Widget Company</manufacturer>
  <countryoforigin>United Kingdom</countryoforigin>
  <size length="3" width="2" height="1" />
</description>
```

If the row exists in both databases but the values are different, the values are shown side-by-side, and the differences are highlighted.

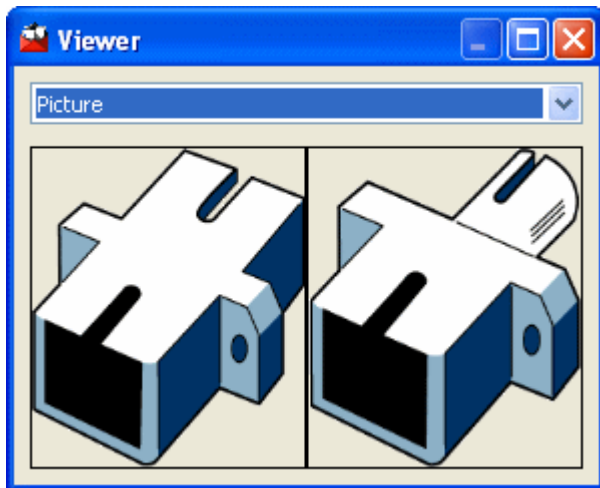


```
Previous Next Show white space
<description xmlns="http://www.red- <description xmlns="http://www.re
.. <descriptiveText>A.cunning.widget # .. <descriptiveText>A.widget</desc
.. <manufacturer>Acme.Inc</manufactu # .. <manufacturer>Acme.Incorporated
.. <countryoforigin>UK</countryofori # .. <countryoforigin>United.Kingdom
.. <size.length="12".width="4".heigh # .. <size.length="12".width="4".hei
</description> </description>
```

Use the **Previous** and **Next** buttons to go to the previous or next line with a difference.



To view the individual spaces, select the **Show white space** check box.

If the value is a binary value, such as an image, click the **Binary Data** link to view the details, for example:



You can select the required format in which to display the data.

### Searching the value details

For string values, you can search the value details by right-clicking, and clicking **Find**. In the **Find** box, type your search text and click  or  to find the next or previous match. If there are no further matches, the **Find** box changes color. If the value details are displayed side-by-side, to search the value details in the other data source, click **Find on this Side**.

To copy the value details for use in another application, select the data, right-click, and then click **Copy**. Alternatively, right-click, click **Select All**, then right-click, and click **Copy**.

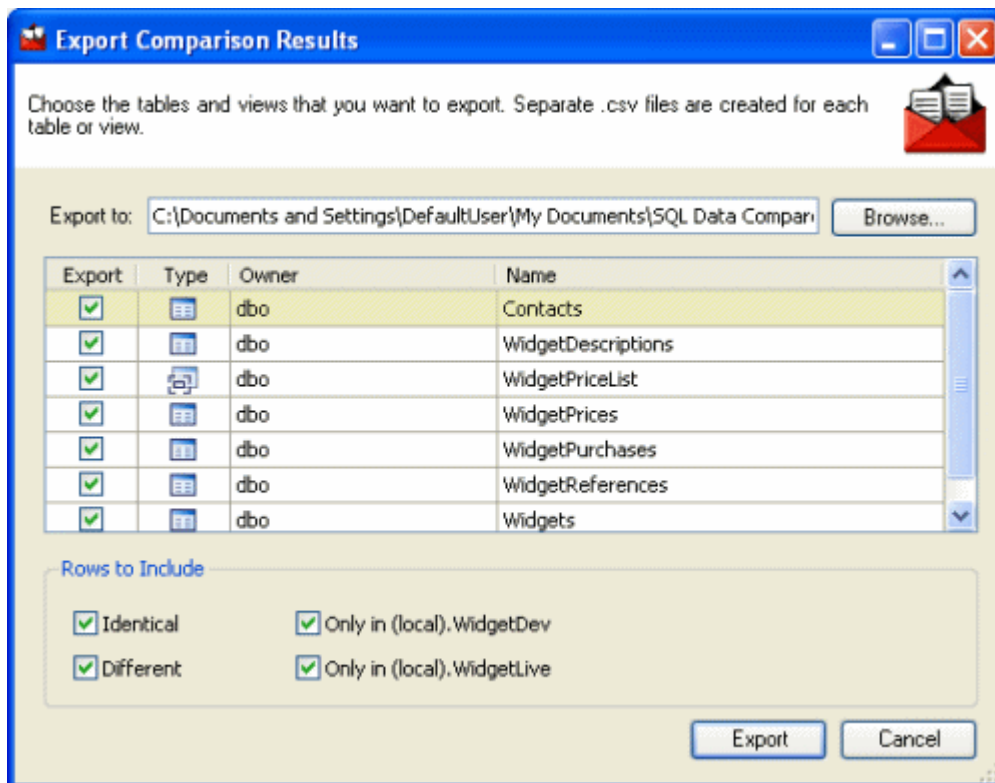
## Exporting the comparison results

---

When you have run a comparison, you can export the comparison, or a subset of the results, to a collection of comma-separated value (CSV) files. For example, you may want to produce a detailed report or to keep a record of the comparison results.

1. On the **Tools** menu, click **Export Comparison Results**.

The **Export Comparison Results** dialog box is displayed.



2. Type the path for the export files in the **Export to** box, or click **Browse** to choose the export folder.

You can change the default path in the Application Options (page 24) dialog box.

3. Select the check box in the **Export** column for the tables or views that you want to export.

You can right-click the list of tables and views and click **Include All** to select all the check boxes or **Exclude All** to clear all the check boxes.

4. Under **Rows to Include**, select the check boxes for the rows that you want to export:

- ◆ **Identical** includes rows that are the same in both databases.
- ◆ **Different** includes rows that exist in both databases but are different.
- ◆ **Only in database name** includes rows that exist only in that database.

5. Click **Export**.

For each selected table or view and for each of the row types that you selected under **Rows to Include**, SQL Data Compare creates a separate CSV file. SQL Data Compare also creates a *Results Summary* file.



## Printing the comparison results

---

When you have run the comparison on the project, you can print a summary of the comparison results.

SQL Data Compare Report							
(LOCAL).WidgetDev vs (LOCAL).WidgetLive							
5 May 2009 12:32 PM							
Type	Different	Table Name	Source Only	Different	Target Only	Table Name	Identical
<b>6 tables or views with differences in their rows</b>							
Table	7	Contacts	0	7	0	Contacts	93
Table	3	WidgetDescriptions	0	2	1	WidgetDescriptions	0
View	4	WidgetPrioList	3	0	1	WidgetPrioList	2
Table	4	WidgetPrices	3	1	0	WidgetPrices	2
Table	11010	WidgetPurchases	0	854	10156	WidgetPurchases	57425
Table	2	Widgets	1	0	1	Widgets	6
<b>1 table or view with identical rows only</b>							
Table	0	WidgetReferences	0	0	0	WidgetReferences	2

To see a preview of the summary, on the **File** menu, click **Print Preview**.

The **Preview** dialog box provides the following commands:



prints the summary



configures the page setup



zoom in or out



displays the first page, previous page, next page, or last page

## Setting up the synchronization

---

When you have reviewed the comparison results, select the rows you want to synchronize, and run the Synchronization wizard.




### Selecting the rows to synchronize

By default, all rows that differ are selected for synchronization.




Use the check boxes in the upper (Results) pane to select rows for synchronization:


Type	Different	Table Name	Source Only	Different	Target Only	Table Name
6 tables or views with differences in their rows						
11125 of 11125 (108560)						
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Contacts	<input checked="" type="checkbox"/> 2	<input checked="" type="checkbox"/> 98	<input checked="" type="checkbox"/> 2	Contacts
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	WidgetDescriptions	<input type="checkbox"/> 0	<input checked="" type="checkbox"/> 2	<input checked="" type="checkbox"/> 1	WidgetDescriptions
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	WidgetPriceList	<input checked="" type="checkbox"/> 3	<input type="checkbox"/> 0	<input checked="" type="checkbox"/> 1	WidgetPriceList
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	WidgetPrices	<input checked="" type="checkbox"/> 3	<input checked="" type="checkbox"/> 1	<input type="checkbox"/> 0	WidgetPrices
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	WidgetPurchases	<input type="checkbox"/> 0	<input checked="" type="checkbox"/> 854	<input checked="" type="checkbox"/> 10156	WidgetPurchases
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Widgets	<input checked="" type="checkbox"/> 1	<input type="checkbox"/> 0	<input checked="" type="checkbox"/> 1	Widgets
1 table or view with identical rows only						

Object selections are remembered when you save a project.

- **All Different** for all rows that are different for a table or view
-  for all rows in a table or view that exist in the source but do not exist in the target
-  for all rows in a table or view that exist in both databases but the rows are different
-  for all rows in a table or view that exist in the target but do not exist in the source

You can also use the **Actions**  drop-down menu to select rows:

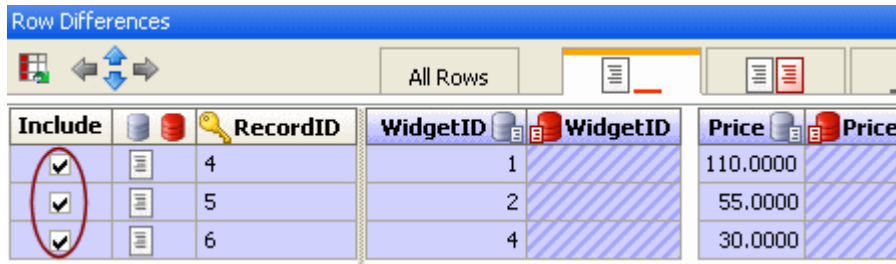
- **Include All** and **Exclude All** select and clear the check boxes for all rows.
- **Include All in Source** and **Exclude All in Source** select and clear the check boxes in the  (only in source) column.
- **Include All Different** and **Exclude All Different** select and clear the check boxes in the  (different) column.
- **Include All in Target** or **Exclude All in Target** select and clear the check boxes in the  (only in target) column.

Note that the **Actions**  drop-down menu is not available if you are viewing the comparison results in a single list.

Use the lower (Row Differences) (page 27) pane to select individual rows to synchronize:

1. Click a table or view to display its row differences.
2. Click the appropriate tab to view the type of row differences.

3. Select or clear the check boxes in the **Include** column as required.



Include	RecordID	WidgetID	Price
<input checked="" type="checkbox"/>	4	1	110.0000
<input checked="" type="checkbox"/>	5	2	55.0000
<input checked="" type="checkbox"/>	6	4	30.0000

### Selecting the rows to synchronize using keyboard shortcuts

To select a row for synchronization, highlight the row and press SPACEBAR to select or clear its check box.

To highlight multiple rows:


- For the previous or next row, press SHIFT+UP or SHIFT+DOWN
- For all rows from the current row to the first or last row, press CTRL+SHIFT+HOME or CTRL+SHIFT+END
- For all rows, press CTRL+A

You can also use SHIFT+Click and CTRL+Click to highlight multiple rows.

## Using the synchronization wizard

---

When you have selected the tables or rows (page 34) that you want to include in the synchronization, you use the Synchronization wizard to create the SQL script that will synchronize the databases.

To open the Synchronization wizard, click  **Synchronization Wizard**.

### Setting the synchronization direction

The **Direction** page enables you to change the direction in which the synchronization changes will be made.

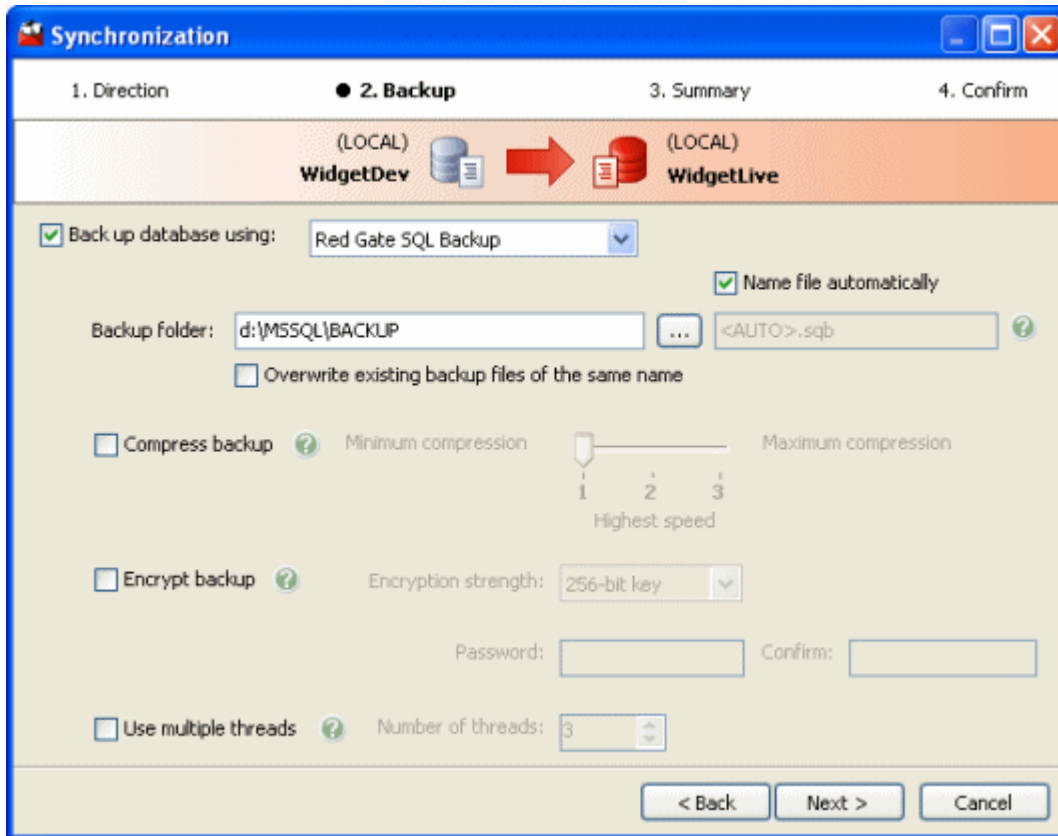


To change the synchronization direction, double-click the arrow.

If the target data source is a backup, the backup will not be modified, but you can generate a synchronization SQL script; note that the generated script does not take into account dependencies or foreign keys, therefore you are not recommended to run the script, if it references multiple tables.

## Backing up the target database

The **Backup** page enables you to create a full backup of the target database before it is updated. If the target data source is a backup, this page is omitted from the Synchronization wizard.




To back up the target database, select the **Back up database using** check box and choose:

- *SQL Server native* to create a backup using the native SQL Server BACKUP command
- *Red Gate SQL Backup* to create a backup using SQL Backup ([http://www.red-gate.com/products/SQL\\_Backup/index.htm](http://www.red-gate.com/products/SQL_Backup/index.htm)) version 4 or later


Note that you must have the SQL Backup server components installed on the SQL Server instance of the target database.

## Backup location

For **SQL Server native**:

1. Type the file path in the **Backup folder** box or click  to specify the path using the folder browser. By default, **Backup folder** is set to the default backup folder for the SQL Server instance.
2. Type the file name in the box to the right of the **Backup folder** box.

For **Red Gate SQL Backup**:

1. Type the file path in the **Backup folder** box or click  to specify the path using the folder browser. By default, **Backup folder** is set to the folder specified in the SQL Backup options for the SQL Server instance. If no backup file locations have been set up, SQL Backup uses the SQL Server instance's default backup folder.
2. Specify the file name in the box to the right of the **Backup folder** box. By default, the file name is set to `<AUTO>.sqb`; SQL Backup uses the SQL Backup options to generate the backup file path and file name. If no backup file locations have been set up, SQL Backup uses the SQL Server instance's default format for file names.

To change the file name, clear the **Name file automatically** check box, and type the required file name. You can use SQL Backup tags, if required. For information about tags, see *File Location Tags* in the SQL Backup online help.

To specify a network path in the **Backup folder** box, type the full path, including the server name, for example `\\ServerName\MyFolder`

Note that the file path is relative to the selected SQL Server. For example, if you have chosen to back up a database on a remote SQL Server instance called ServerA and you specify a local path such as `C:\Backups`, the backup files will be created on the C: drive on ServerA, not on the local computer.

Select the **Overwrite existing backup files of the same name** check box if you want to overwrite any files of the same name that exist for the file path you specified in the **Backup Folder** box. Note that if a file of the same name exists already and you have not chosen to overwrite it, the backup will fail if you are using SQL Backup.

### Compressing the backup

If you are using SQL Backup to back up the target database, you can choose from the three compression levels described below. Generally, the smaller the resulting backup file, the slower the backup process.

To compress the backup, select the **Compress backup** check box and select the compression level by moving the slider.

- **Compression level 3**

Compression level 3 uses the zlib compression algorithm. This compression level generates the smallest backup files in most cases, but it uses the most CPU cycles and takes the longest to complete.

- **Compression level 2**

This compression level uses the zlib compression algorithm, and is a variation of compression level 3.

On average, the backup process is 15% to 25% faster than when compression level 3 is used, and 12% to 14% fewer CPU cycles are used. Backup files are usually 4% to 6% larger.

- **Compression level 1**

This is the default compression level. It is the fastest compression, but results in larger backup files. On average, the backup process is 10% to 20% faster than when compression level 2 is used, and 20% to 33% fewer CPU cycles are used. Backup files are usually 5% to 9% larger than those produced by compression level 2. However, if a database contains frequently repeated values, compression level 1 can produce

backup files that are smaller than if you used compression level 2 or 3. For example, this may occur for a database that contains the results of Microsoft SQL Profiler trace sessions.

Note that if SQL Backup Lite is installed on the SQL Server, you can choose only compression level 1.

### Encrypting the backup

If you are using SQL Backup to back up the target database, you can encrypt the backup by selecting the **Encrypt backup** check box, then typing a password for the backup in **Password**, and again in **Confirm**.

If SQL Backup Pro is installed on the SQL Server, you can choose 128-bit or 256-bit encryption; if SQL Backup Standard is installed on the SQL Server, you can choose only 128-bit encryption; if SQL Backup Lite is installed on the SQL Server, you cannot encrypt the backup.

**You must remember your password**; if you do not, you will not be able to access the encrypted backup.

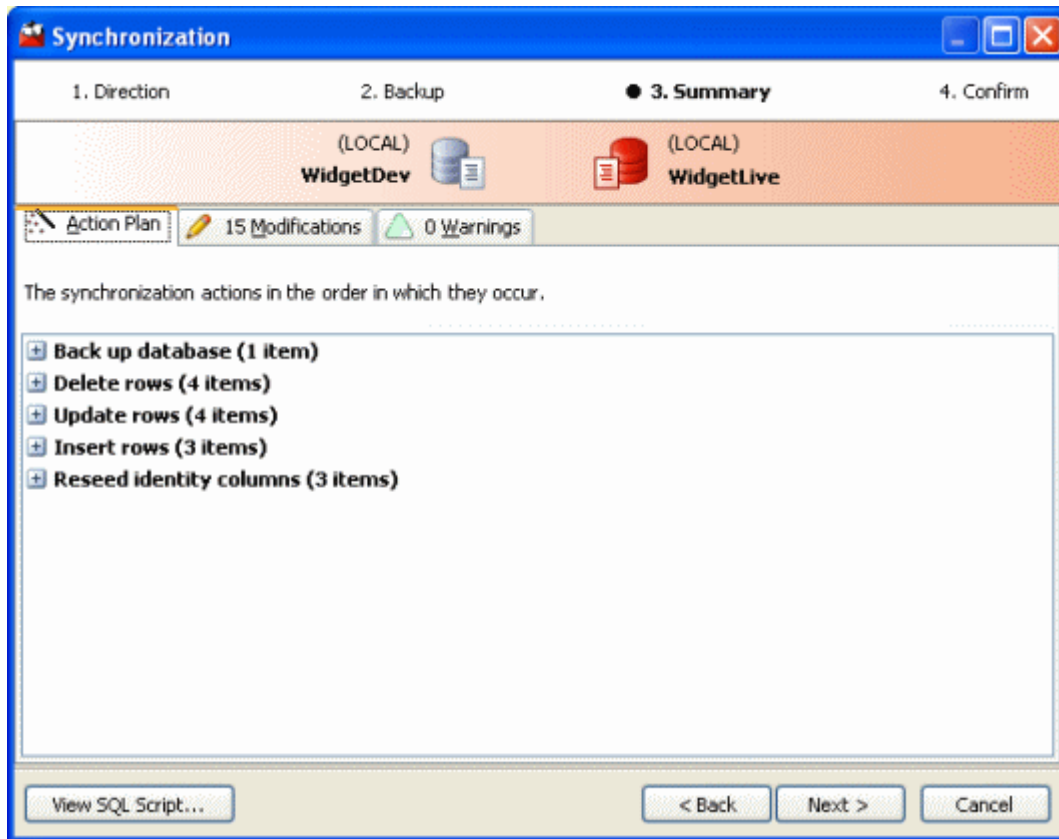
### Using multiple threads

If you are using SQL Backup to back up the target database and you are using a multi-processor system, using multiple threads can speed up the backup process. Select the **Use multiple threads** check box and type or select the number of threads up to a maximum of 32. You are recommended to start with one thread fewer than the number of processors. For example, if you are using four processors, start with three threads.

For details of how you can find out the most effective number of threads to use for your setup, see *Optimizing Backup Speed* in the online help for SQL Backup.


## Reviewing the synchronization summary

The **Summary** page lists the actions and modifications that will be carried out during the synchronization.

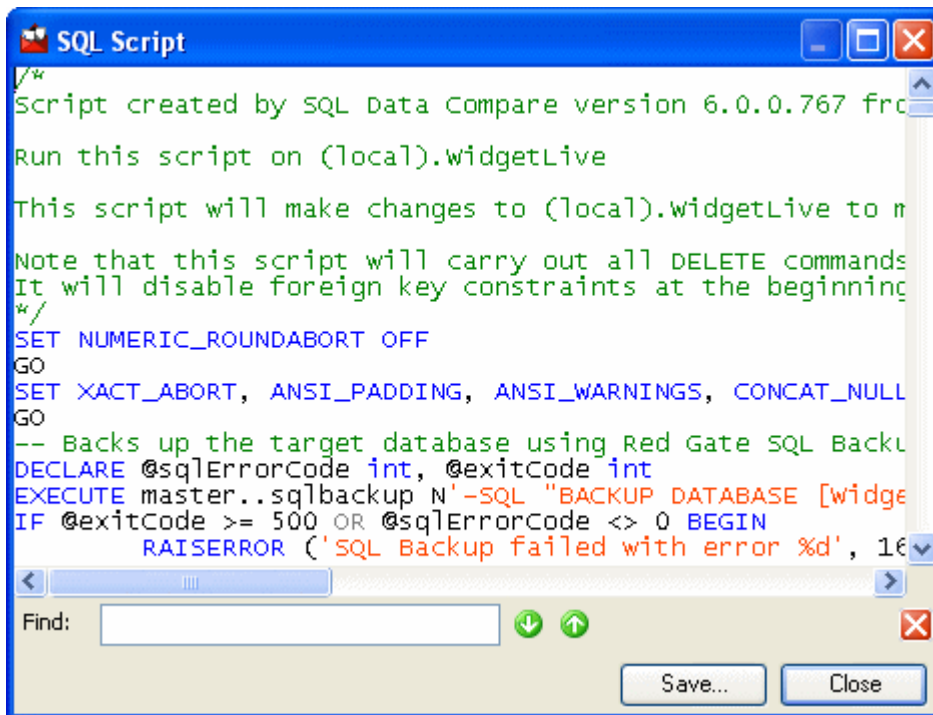


The **Summary** page displays the following tabs:



- **Action Plan** provides a synopsis of the script, grouped by command type, in the order in which the commands will run. To view the commands for each command type, click . To copy the action plan, right-click and click **Copy Action Plan to Clipboard**.
- **Modifications** provides a synopsis of the script, grouped by table or view. To copy the modifications, right-click and click **Copy Modifications to Clipboard**.
- **Warnings** displays any warnings about unexpected behavior that may occur when you synchronize the databases. To copy the warnings, right-click and click **Copy Warnings to Clipboard**.



To display the synchronization SQL script, click **View SQL Script**.



```

/*
Script created by SQL Data Compare version 6.0.0.767 from
Run this script on (local).widgetLive
This script will make changes to (local).widgetLive to m
Note that this script will carry out all DELETE commands
It will disable foreign key constraints at the beginning
*/
SET NUMERIC_ROUNDABORT OFF
GO
SET XACT_ABORT, ANSI_PADDING, ANSI_WARNINGS, CONCAT_NULL
GO
-- Backs up the target database using Red Gate SQL Backu
DECLARE @sqlErrorCode int, @exitCode int
EXECUTE master..sqlbackup N'-SQL "BACKUP DATABASE [widge
IF @exitCode >= 500 OR @sqlErrorCode <> 0 BEGIN
    RAISERROR ('SQL Backup failed with error %d', 16
  
```

You can:

- search the script  
In the **Find** box, type your search text and click  or  to find the next or previous match. If there are no further matches, the **Find** box changes color.
- copy the script details for use in another application  
Select the SQL statements, right-click, and then click **Copy**.  
Alternatively, right-click, click **Select All**, then right-click, and click **Copy**.
- save the script to open it in a different application, or to keep a record of it  
Click **Save**; a standard Windows® **Save As** dialog box is displayed.

## Running the synchronization

The **Confirm** page enables you to specify how you want to use the synchronization SQL script.



Do one of the following:

- To run the synchronization SQL script from within SQL Data Compare, select **Synchronize databases now**.  
Note that if the target data source is a backup, this option is not available.
  - ◆ If you want SQL Data Compare to re-compare the databases on completion of the synchronization, select **Compare databases following synchronization**.
  - ◆ If you want SQL Data Compare to save the synchronization SQL script before the databases are synchronized, select **Save a copy of the SQL script before synchronization**.
- To review the script, depending on your Application Options (page 24) settings, select **Launch the SQL script in Query Analyzer** or **Launch the SQL script in SQL Server Management Studio**.
- To save the synchronization SQL script without synchronizing the databases, select **Save the SQL script without synchronizing the databases**.

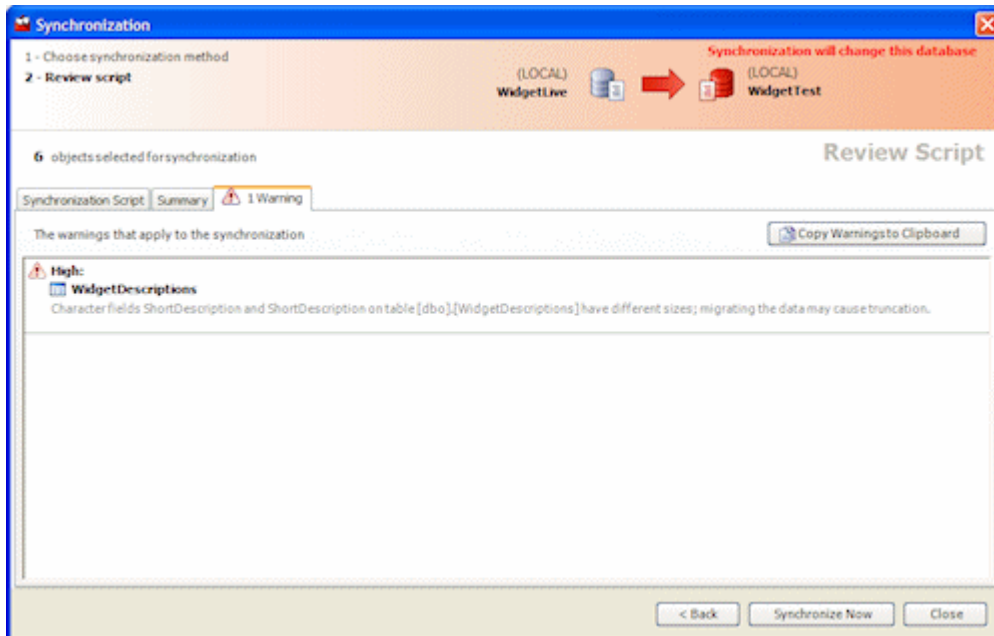
You can change the default query editor or the default location where the synchronization SQL script is saved by opening the **Tools** menu and clicking **Application Options**; you must close the Synchronization Wizard first.

If you selected **Synchronize databases now** and SQL Data Compare is unable to synchronize the data, an error dialog box is displayed, and where possible all changes are rolled back. Note that if you have cleared the project option (page 20) **Use transactions in SQL scripts**, the changes are not rolled back.

## Warnings

---

On the **Summary** page of the Synchronization wizard, you can click **Warnings** to view information about unexpected behavior that may occur when you synchronize the databases, or reasons the script may fail. The warnings are graded according to severity.



You can copy the warnings so that you can paste them into another application by clicking **Copy Warnings to Clipboard**.

Some of the warnings that SQL Data Compare may display are explained below.

### Possible truncation error

SQL Data Compare displays this warning when the length setting of a character or binary column has changed. Data may be truncated in the synchronization SQL script.

### Possible rounding error

This warning is displayed when the scale setting of a decimal or numeric column has changed. Data may be rounded in the synchronization SQL script.

### Possible overflow error

This warning is displayed when the precision setting of a decimal or numeric column has changed. There may be overflows when the data is synchronized.

## Conversion to XML

SQL Data Compare displays this warning when data is converted to an XML data type column. If the data is not valid XML, unexpected behavior may occur when you synchronize the databases.

## Collation mismatch

SQL Data Compare displays this warning when the collation setting for a column has changed.

You can set SQL Data Compare so that binary collation is used on all character string sorting by selecting the **Force binary collation (case-sensitive)** project option (page 20).

## Starting SQL Data Compare with a specific project

---

You can start SQL Data Compare with a specific project. You can load the project so that you can edit it, or you can load the project and let SQL Data Compare run the comparison for you.

### Using Windows® Explorer

Right-click the project file (.sdc), and then click **Edit** or click **Compare**.

### Using the command line

On the **Start** menu, click **Run**, and enter the commands shown below.

To edit the project:

```
"C:\Program Files\Red Gate\SQL Data Compare  
7.0\RedGate.SQLDataCompare.UI.exe" C:\Project.sdc
```

To load the project and run the comparison:

```
"C:\Program Files\Red Gate\SQL Data Compare  
7.0\RedGate.SQLDataCompare.UI.exe" C:\Project.sdc /compare
```

## Case-sensitive comparisons

---

By default, columns inherit the collation of the database. If you are working with case-insensitive databases, you can run a case-sensitive comparison by selecting the project configuration option **Force binary collation (case-sensitive)**. The setting applies to all columns that contain string data. To perform a case-sensitive comparison, SQL Data Compare uses the Latin1\_General\_BIN collation.

For example, if you compare *Widget* and *widget*, when the project option is not selected, SQL Data Compare considers the data to be identical. When the option is selected, SQL Data Compare identifies the case difference.

Note that if you want to compare objects whose names differ only by their case, such as table names or column names, ensure the **Ignore case of object names** option is selected.

## Filtering the comparison with a WHERE clause

---

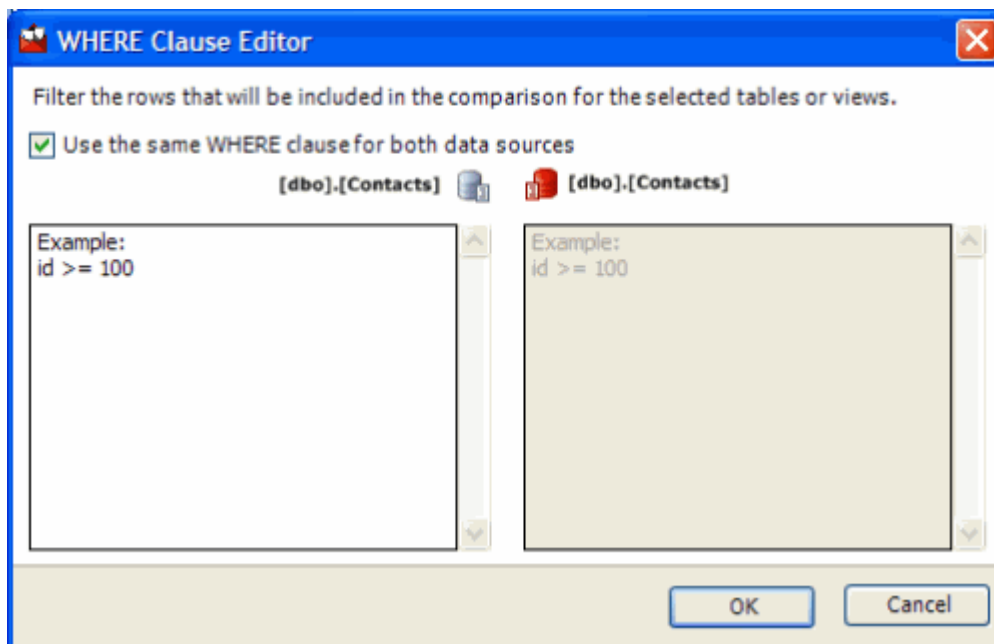
You can filter the rows that will be compared by applying a WHERE clause to the comparison. This is useful, for example, if you want to exclude a particular set of test data, or to speed up the comparison.

Note that:

- You can only filter rows if the data source is a database. WHERE clauses are not available for backups and scripts folders.
- The **WHERE Clause Editor** does not validate the WHERE clauses. The clauses are validated when the databases are compared.

To specify a WHERE clause for a table, on the **Tables & Views** tab, double click the table, or select it and click **WHERE Clause**

The **WHERE Clause Editor** is displayed:



To apply the same WHERE clause to multiple tables, use SHIFT+Click or CTRL+Click. Alternatively, you can right-click the tables or views and click **Open WHERE clause editor**.

Type a valid Transact-SQL WHERE clause. For example, if a table has columns *ID*, *FirstName*, and *LastName*, you may want to compare only rows where *LastName* is *Smith*. To do this, type the following in the box:

```
LastName='Smith'
```

Click **OK**. In the **Tables & Views** tab, **WHERE** indicates that the table or view will be filtered using a WHERE clause.

If you want to apply a different WHERE clause for each data source, clear the **Use the same WHERE clause for both data sources** check box and type the WHERE clause in the box for the database on the right.

For example, if you have two databases for two different time zones that are 5 hours apart and you want to repair damaged data in one of the tables, you can specify a different WHERE clause in each database for that table.

Since you know approximately when the problem occurred, you can limit the number of rows that are compared. Clear the **Use the same WHERE clause for both data sources** check box, and specify WHERE clauses for each data source.

Alternatively, you can construct a simple view on the table in both databases. You can then use SQL Data Compare to compare the views.



## Synchronizing views

---

SQL Data Compare can synchronize views directly only if they reference rows from a single table, and the referenced columns are simple (for example, they must not include identity columns or computed columns). If you want to synchronize views that reference more than one table, the following procedure is recommended:

1. Use SQL Data Compare to compare the views.  
Ensure that the **Include views** project option (page 20) is selected.
2. Note which rows are different.
3. Use SQL Data Compare to compare and synchronize the tables that contain the differing rows.

## Comparing databases on different SQL Server versions

---

If you are comparing databases that are on different versions of Microsoft® SQL Server™, you may encounter problems with some data types.

### CLR data types

You can update CLR data in a SQL Server 2008 or SQL Server 2005 database with values from a text or string data type in a SQL Server 2000 database. Ensure that the project option **Transport CLR data types as binary** is not selected.

SQL Data Compare considers the collation for string data. Therefore, if the collation is not the same, differences are reported.

Note that if you are comparing backup files, SQL Data Compare can compare CLR data types only as binary values.

### XML data types

You can update XML data in a SQL Server 2008 or SQL Server 2005 database with values from a text or string data type in a SQL Server 2000 database. SQL Data Compare will attempt to preserve white space. SQL Data Compare supports DTD (Document Type Definition), except for default attributes and entities.

Some data, such as XML encoding and DTD, cannot be stored in the SQL Server 2008 or SQL Server 2005 representation. Therefore, if you convert data from a string data type to an XML data type, and then you convert back to a string data type, this information will be lost.

SQL Data Compare considers the collation for string data. Therefore, if the collation is not the same, differences are reported.

Note that you cannot use an XML column as the comparison key.

## Creating a rollback script

---

If you want to be able to reverse a deployment, or to return a database to a specific state, you can create a rollback script.

Before you run the Deployment Wizard:

1. In the main window, right click in the Direction Bar, and click **Switch deployment direction**.
2. Use the Deployment wizard to create and save a deployment script.

For example, if you are migrating changes from *WidgetStaging* (the source) to *WidgetProduction* (the target), switching the deployment direction makes *WidgetProduction* the source, and the deployment script you create can be run on *WidgetProduction* in future to restore it to its current state.

You can also use SQL Data Compare to restore rows selectively from a backup.

## Resolving mapping errors

---

This topic provides information that may help you to troubleshoot mapping errors. It provides information about using the **Project Configuration** dialog box and what happens to a project when the database schema changes.

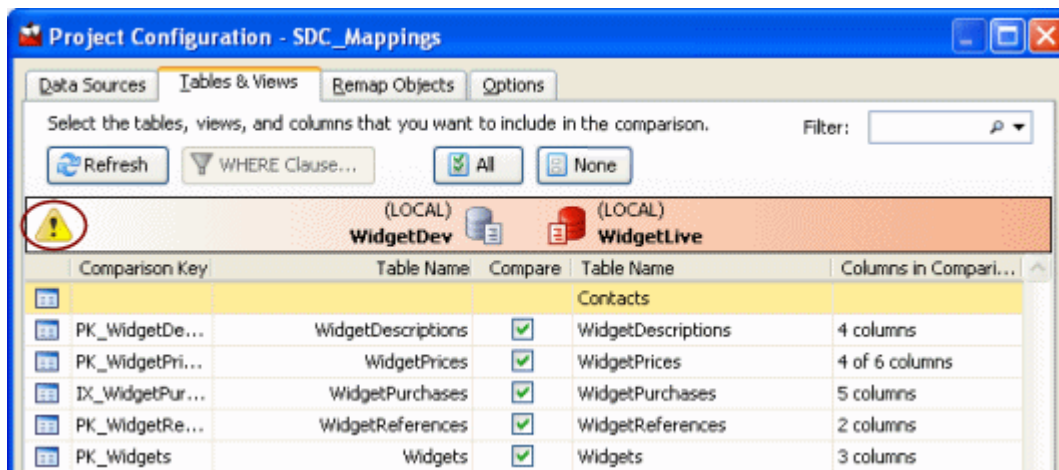
When you create a new project and you specify the data sources you want to compare, SQL Data Compare selects all tables for comparison if:

- the tables have similar structure
- the table names and owner names are similar
- the tables have suitable comparison keys

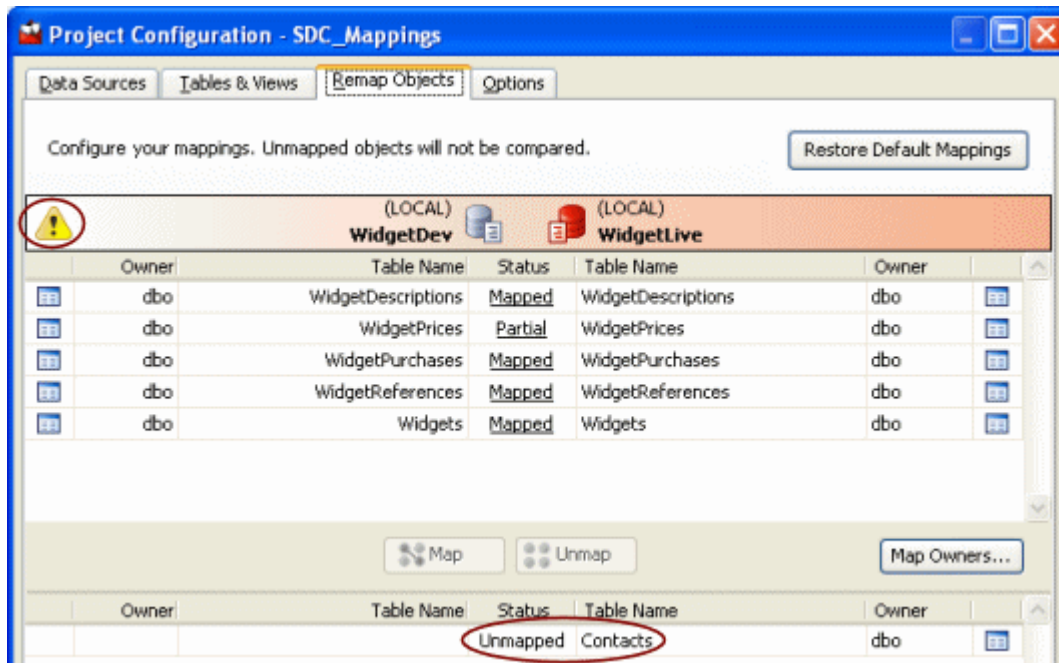
SQL Data Compare selects all columns for comparison if the column names are similar and the data types are compatible.

If the database schema has changed, if the project configuration options you have saved no longer match the current state of the data sources, or if an object is different between the data sources, SQL Data Compare may not be able to compare the affected objects.

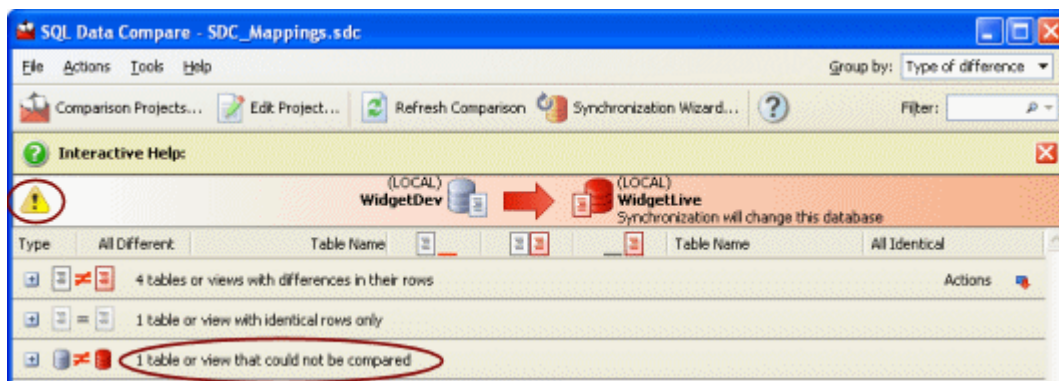
For example, on the **Tables & Views** tab, if you clear the check box for a particular table, and that table is later removed from the database schema, SQL Data Compare displays a warning symbol.



SQL Data Compare also shows a warning symbol on the **Object Mapping** tab, and in this example, the table's status is set to *Unmapped*.



If you ignore the warning and you run the comparison, SQL Data Compare compares only the tables for which there are no warnings; you will see a warning symbol in the direction bar in the **Results** pane, and in this example, the table is listed under **tables or views that could not be compared**.



### How do I get rid of the warning symbol?

If a warning is shown because an object exists in only one of the data sources, you must synchronize the database schema to remove the warning.

Red Gate Software offers SQL Compare, which will synchronize the structure of two databases. You can then use SQL Data Compare to synchronize the data.


For more information, see the SQL Compare product page ([http://www.red-gate.com/products/SQL\\_Compare/index.htm](http://www.red-gate.com/products/SQL_Compare/index.htm))

If the object exists in both data sources, but there are schema differences, you may be able to manually set the object or column mapping. Otherwise, you must synchronize the schema.

If you do not want to compare an *Unmapped* table, you can still run the comparison and synchronization but the table will be excluded.

If the column set as the comparison key is deleted, you can select a different column as the comparison key so that you can include the table in the comparison.

### What happens when a table is created or dropped?

If a new table has been added to the database schema since you last ran the comparison, SQL Data Compare will include the table in the comparison. However, if you excluded tables by clicking  **None** on the **Tables & Views** tab, the new table will not be included in the comparison.

If a table has been removed from the database schema since you last ran the comparison, SQL Data Compare does not display the table on the **Table & Views** tab.

However, if you made any configuration changes for that table, you will see a warning symbol on the **Table & Views** tab and the **Object Mapping** tab. Such changes include:

- selecting or clearing the table's check box
- setting a different comparison key or custom comparison key for that table
- setting WHERE clauses for that table
- selecting or clearing the check boxes for any columns in that table
- mapping any of the columns in that table
- mapping that table using the **Object Mapping** tab
- mapping schemas (owners)

### What happens when a column is added or dropped?

If a new column has been added to a table since you last ran the comparison, SQL Data Compare will include the column in the comparison.

If a column has been removed from a table and you made any configuration changes for that column, you will see a warning symbol on the **Table & Views** tab and the **Object Mapping** tab. The project configuration changes include:

- setting that column as a custom comparison key
- selecting or clearing the check boxes for that column
- mapping that column

### Can I use a single project to compare and synchronize more than one database?

Yes. If you are using SQL Data Compare to synchronize data for all your databases with similar schemas, for example synchronizing development, quality assurance, user acceptance testing, training, or production databases, you can select each database in turn on the **Data Sources** tab, and run the comparison and the synchronization. If there

are schema differences, your project configuration changes are retained but there may be some tables or columns that cannot be compared.

## Common error messages

---

Some of the more common error messages are explained below.

### Could not start a transaction for OLE DB provider (*name*)

SQL Data Compare displays this message when your source database contains an object that references a linked server, but the linked server is not defined on the target server.

### SQL Server doesn't exist or access is denied

SQL Data Compare cannot connect to the SQL Server. Try the following to rectify the problem:

1. Verify that the SQL Server is online and that the SQL Server name is listed in your LAN by *pinging* the address.

For example, open a command prompt and run the following command:

```
ping <ServerName>
```

where *ServerName* is the name of your SQL Server.

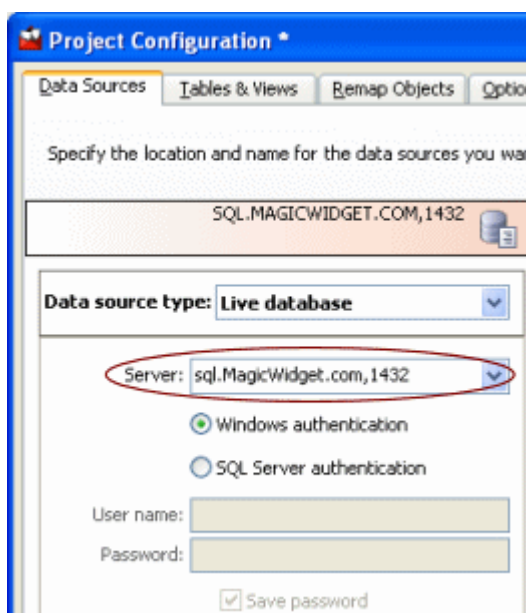
2. If the SQL Server is online, verify that you are connecting to the correct port.

If your SQL Server is not running on the default port (1433), type the following in **Server** on the **Project Configuration** dialog box:

```
<ServerName>,<Port>
```

where *ServerName* is the name of your SQL Server and *Port* is the number of the port on which your SQL Server is running.

For example:

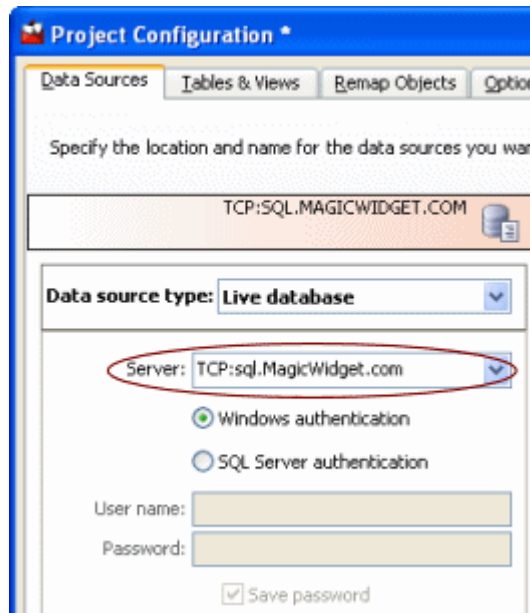




3. If you are sure that you are connecting to the correct port, force SQL Data Compare to use the TCP network protocol when it makes the connection, by typing the following in **Server** on the **Project Configuration** dialog box:

TCP:<ServerName>

For example:



## Troubleshooting

---

This topic provides information that may help you to understand your comparison and synchronization results.

### Missing tables or columns in Project Configuration

If the structure of the databases that you are comparing has changed while you are working on a comparison project, those changes are not automatically shown in the **Tables & Views** tab on the **Project Configuration** dialog box. For example, if you have added columns to both databases, these will not be shown and will therefore be omitted from the comparison.

To update the **Tables & Views** tab on the **Project Configuration** dialog box, click  **Refresh**. For more information, see:

- Selecting tables and views (page 16)
- Mapping errors (page 52)

### Identical CLR or XML data is flagged as different

SQL Data Compare considers the collation for CLR and XML data. Therefore, if the collation is not identical, differences are reported.

### Data has not been synchronized

This may occur if:

- there are triggers defined on the tables  
If you have a trigger defined on a table that inserts data into another table on INSERT, DELETE, or UPDATE, the data in the tables will change as the synchronization is run, which will cause unpredictable results. To avoid this, select the **Disable DML triggers** project configuration option before you generate the synchronization script.
- primary keys are defined on columns with differing collations  
If you compare tables that have primary keys defined on columns that have different collations, SQL Data Compare may produce unpredictable results.
- columns contain *timestamp* data  
SQL Data Compare cannot synchronize data in *timestamp* columns.
- tables do not have identical structure  
If the structure of the tables that you are synchronizing is not identical, SQL Data Compare may produce unpredictable results.

### CLR data has not been synchronized

Data for CLR types can be stored as string or binary values. When CLR data is compared, SQL Data Compare always compares the binary representations. However, by default,

when CLR data is synchronized, SQL Data Compare synchronizes the string representations, because binary formats are not always compatible.

If the binary representations are not compatible, they are always displayed as different even if the string representations are identical. String representations do not always contain the full information about the data, so when the databases are re-compared using the binary representations following synchronization, they may be displayed as different.

If you know that the binary formats are compatible, you can select the project configuration option **Transport CLR data types as binary**.

This forces SQL Data Compare to synchronize the binary representations.

### Primary keys, indexes, or unique constraints are not dropped for synchronization

If you select the project configuration option **Drop primary keys, indexes, and unique constraints**, note that primary keys, indexes or unique constraints that are selected as comparison keys are not dropped for the synchronization.

### Insufficient disk space

SQL Data Compare may be unable to compare databases if there is insufficient disk space. SQL Data Compare uses temporary files when it compares the databases. You can decrease the size of the temporary files by restricting the rows that are compared, for example, by using a WHERE clause or excluding objects that you know you do not want to update.

For more information, see [Selecting tables and views](#) (page 16)

You can decrease the size of the temporary files by selecting the project configuration option **Compress temporary files**.

Note that selecting this option means that you cannot sort the results following the comparison.

You can also decrease the size of the temporary files by clearing the **Show identical values in results** project option and selecting the **Use checksum comparison** project option.

For more information, see [Setting project options](#) (page 20)

To avoid running out of disk space, you can also change the location of the temporary files. The location of the temporary files is defined by the **RGTEMP** environment variable, or the **TMP** variable if **RGTEMP** does not exist (see your Windows® documentation for information about environment variables).

Note that changing the **TMP** variable will affect other programs that use the variable.

### Cannot sort columns in the Row Differences pane

If you select the project configuration option **Compress temporary files**, note that you will be unable to sort the comparison results in the **Row Differences** pane by clicking on a column header.

## Rollback on script failure or cancellation

If a script fails, or if it is cancelled, in most circumstances changes are rolled back. SQL Data Compare uses *transactions* to do this.

If you have cleared the project configuration option **Use transactions in SQL scripts** to remove transactions from the synchronization SQL script, no changes are rolled back when the script fails or is cancelled. This may be useful if you want to run a script up to the point of failure, for example for debugging.

SQL Data Compare always warns you if it is unable to roll back changes.

## Column mappings

SQL Data Compare cannot compare certain combinations of data types.

Note that:

- You can compare a *timestamp* column with another *timestamp* column but you cannot synchronize *timestamp* columns.
- You can compare an *xml* column with another *xml* column but you must ensure that your XML schemas are compatible.

## Getting started with the SQL Data Compare command line

---

The command line interface provides access to the functionality of SQL Data Compare. For example, using the command line interface you can:

- automate the comparison and deployment of database schema
- perform scheduled comparisons and deployments
- deploy multiple databases

You invoke the command line either from a script, such as a batch script or VBScript, or by using the facilities provided by compiled languages such as VB, C++ and C#.

### Prerequisites

To use the SQL Data Compare command line interface, you must have:

- A SQL Data Compare Professional Edition, SQL Developer Bundle, or SQL Toolbelt license
- If you do not have a license, you can use the command line for 14 days.
- .NET framework version 2.0 or later

This is required to run the command line interface, but it is not required when you develop applications and scripts that use the command line interface.

- MDAC 2.8 or later

## Basic command line features

---

This topic describes how to use the basic features of the command line.

### Getting help from the command line

To display help from the command line, enter:

```
sqldatacomapre /help
```

This displays a brief description of the tool, and basic help on all the command line switches.

For more detailed help enter:

```
sqldatacomapre /help /verbose
```

This displays a detailed description of each switch, any values it can accept, and all exit codes. To output the help in HTML format, enter:

```
sqldatacomapre /help /verbose /html
```

### Entering a command

When you enter a command line, the order of switches is unimportant. You are recommended to follow the Microsoft® convention of separating a switch from its values using a colon as shown below.

```
/out:output.txt
```

(You can separate a switch that accepts a single value from its value using a space, but this is not recommended.) Values that include spaces must be delimited by double quotation marks ( " ). For example:

```
/out:"c:\output file.txt"
```

Note that if you delimit a path with double quotation marks, you must not terminate the path with the backslash character ( \ ), because the backslash will be interpreted as an escape character. For example:

**Incorrect: /location:"C:\Packages\"**

**Correct: /location:"C:\Packages"**

For switches that accept multiple values, use commas to separate the values. For example:

```
/options:IgnoreSpaces,IgnoreUnderscores
```

For switches that accept a compound value, separate each part of the value using a colon. For example, the */include* and */exclude* switches are used to include and exclude database objects from the actions performed by the tool. For example:

```
/include:table:Product
```

includes all tables for which the table name contains the word *Product*.

## Aliases

Many of the switches have an alias. The alias provides a convenient short-hand way to specify the switch. For example, */?* is the alias for the */help* switch, and */v* is the alias for the */verbose* switch. Note that switches and aliases are not case-sensitive.

### ***/verbose* and */quiet* switches**

The standard output mode prints basic information about what the tool is doing while it is executing. You can specify verbose and quiet modes using the */verbose* and */quiet* switches, respectively: in verbose mode, detailed output is printed; in quiet mode, output is printed only if an error occurs.

### ***/options* switch**

You can use the */options* switch to change your options. For example, by default, comparisons do not consider trailing spaces; to specify that SQL Data Compare considers spaces at the end of the string:

```
/options:TrimTrailingSpaces
```

However, note that if you set any options explicitly, all of the default options are switched off.

For more information about the SQL Data Compare command line options, and which options are selected by default, see *Options used in the command line* (page 85).

### ***/argfile* switch**

You can use the */argfile* switch to specify command line arguments in an XML file.

For details, see *Using XML to specify command line arguments* (page 73).

## Redirecting command output

Output from all commands can be redirected to a file by one of several methods:

- Use the */out* switch to specify the file to which you want output directed:

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction  
/out:OutputLog.txt
```

where *OutputLog.txt* is the name of the file. If the file exists already, you must also use the */force* switch to force the tool to overwrite the file, otherwise an error will occur.

- Use the output redirection features that are provided by the shell in which you are executing the command.

From the standard command prompt provided by Windows, you can redirect output to a file as follows:

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction  
> OutputLog.txt
```

Note that the redirection operator ( *>* ) and file name must be the last items on the command line. If the specified file exists already, it will be overwritten. To append

output from the tool to an existing file, for example to append to a log without losing the data already present in the log, enter the following: `SQLDataCompare ... >> existinglog.txt` If you are scripting using a language such as VBScript, JScript, PHP, Perl, or Python, or if you want to access the tool from Web pages using ASP.NET, refer to the documentation for the language.



## Integrating the command line with applications

---

To integrate the SQL Data Compare command line tool with applications that you distribute to your customers, you must have a license for the SQL Data Compare Professional Edition, or SQL Toolbelt. When you have a license and you execute the tool, the distribution files that you need to distribute the applications are generated. These files are marked with an asterisk (\*) below.

The files that you should bundle into your application installer are listed below. The files should be installed in the same folder in which your application is installed.

- RedGate.SQLToolsCommandLine.dll
- RedGate.BackupReader.CryptoHelper.dll
- RedGate.BackupReader.dll
- RedGate.BackupReader.SqbReader.dll
- RedGate.Shared.Utils.dll
- RedGate.Shared.SQL.dll
- RedGate.SQLCompare.ASTParser.dll
- RedGate.SQLCompare.Engine.dll
- RedGate.SQLCompare.Rewriter.dll
- RedGate.SQLDataCompare.CommandLine.dll
- RedGate.SQLDataCompare.Distribution.dll\*
- RedGate.SQLDataCompare.Distribution.mod\*
- RedGate.SQLDataCompare.Engine.dll
- SQLDataCompare.exe
- SQLDataCompare.exe.config
- System.Data.SQLite.dll
- zlib1.dll

## Simple examples using the command line

---

This topic provides some simple examples of how to use the command line interface.

For detailed examples of how to include specific tables, see the following topics:

- Example: selecting single tables for comparison (page 69)
- Example: selecting tables with unrelated names (page 71)

### Comparing and synchronizing database data

To compare the data in *WidgetDev* and *WidgetLive* on the local instance:

```
sqldatacompare /Database1:WidgetDev /Database2:WidgetLive
```

To compare the data in *WidgetDev* and *WidgetLive*, and export the differences to .csv files:

```
sqldatacompare /Database1:WidgetDev /Database2:WidgetLive  
/Export:"D:\SQLDataCompareReports"
```

To compare the data in *WidgetDev* and *WidgetLive*, and synchronize the databases by updating *WidgetLive*:

```
sqldatacompare /Database1:WidgetDev /Database2:WidgetLive  
/Synchronize
```

To compare the data in *WidgetDev* and *WidgetLive*, specifying the comparison key for the *Widgets* table explicitly to be *PK\_Widgets*:

```
sqldatacompare /Database1:WidgetDev /Database2:WidgetLive  
/ComparisonKeys:Widgets:PK_Widgets
```

Note that you can specify a primary key or a unique index as the comparison key when you use the command line interface.

To use a project (page 7) that you have previously created using the graphical user interface:

```
sqldatacompare /Project:"C:\SQLDataCompare\Projects\Widgets.sdc"
```

When you use a project, all tables and views that were selected for comparison when the project was saved are automatically included; you do not need to explicitly include them using the */include* switch. You can override the inclusions by specifying the */exclude* switch as required, for example if you do not want to compare any views:

```
sqldatacompare /project:"C:\SQLDataCompare\Projects\Widgets.sdc"  
/exclude:View
```

Other project configuration details such as column settings, custom comparison keys, and WHERE clauses are also applied. Note that the project configuration does not include table and row selections for the comparison results. If the project includes backup settings that have been set up using the Synchronization Wizard, those settings are ignored; the command line cannot back up databases before synchronization.

To synchronize data in *WidgetDev* and *WidgetLive*, specifying that for table *WidgetPrices* only columns *RecordID*, *Price*, and *Active* are to be migrated:

```
sqldatacompare /database1:WidgetDev /database2:WidgetLive
               /columns:WidgetPrices:RecordID,Price,Active
               /synchronize
```

## Using a backup as a data source

To compare a backup of *WidgetDev* with *WidgetLive*:

```
sqldatacompare
/backup1:D:\MSSQL\BACKUP\WidgetDev_20080807_143235.sqb
/db2:WidgetLive
```

If you are using native SQL Server backups and the backup files contain multiple backup sets, use the */backupset1* and */backupset2* switches to specify the required backup set. If the backup set switches are not specified, SQL Data Compare uses the latest backup set.

```
sqldatacompare /backup1:D:\MSSQL\BACKUP\WidgetDev.bak
               /backupset1:"2008-09-23 Full Backup" /db2:WidgetLive
```

To specify more than one backup file, the file names are separated using semicolons.

```
sqldatacompare /backup1:D:\MSSQL\BACKUP\WidgetDev_Full.bak
               D:\MSSQL\BACKUP\WidgetDev_Diff.bak /db2:WidgetLive
```

For encrypted backups that have been created using SQL Backup, use the */password1* and */password2* switches to specify the passwords; when there is more than one password, the passwords are separated using semicolons.

```
sqldatacompare /backup1:D:\MSSQL\BACKUP\WidgetDev.sqb
               /password1:Pa$$w0rd /db2:WidgetLive
```

## Scheduling a comparison

You can use the Microsoft® Windows® Scheduled Task wizard to schedule a comparison by creating a script to run the comparison.

For example, to compare the data in two databases, you could create the following script:

```
C:
cd path_to_installation_folder
sqldatacompare /db1:FirstDatabaseName
/db2:SecondDatabaseName >> log_file
```

where:

- *path\_to\_installation\_folder* is the path to the folder in which you installed the tool. Alternatively, you can add the installation folder to your PATH environment variable and omit this line.
- *FirstDatabaseName* and *SecondDatabaseName* are the names of the databases that you want to compare.
- *log\_file* is the full path to the log file. For example, *C:\SQLCmdLine\Log.txt*

In this example MS-DOS batch scripting is used, a basic scripting language that is supported on all versions of Windows. If preferred, you could use VBScript, JScript, PHP, Perl, Python or any other scripting language of your choice.

Save the script as a .bat file. You can then specify the .bat file as the program to run from within the Scheduled Task wizard by browsing to it.



## Example: selecting single tables for comparison

---

This example illustrates how you select a single table for comparison.

In this example, the databases contain the following tables (amongst others):

- Product
- ProductCategory
- ProductCostHistory
- ProductDealerPriceHistory
- SpecialOfferProduct

You are interested only in the differences between the *Product* tables in two different versions of your database; you are not interested in any of the other tables or views in the databases.

### Using the command line

To specify the table to include, you use the */Include* switch:

```
sqldatacompare /db1:Products1 /db2:Products2  
/Include:table:\[Product\] /verbose
```

where:

- */db1:Products1*  
specifies that you want to compare the database *Products1*
- */db2:Products2*  
specifies that you want to compare the database *Products2*
- */Include:table:\[Product\]*  
specifies that you want to compare only the table that has a name that includes the string *[Product]*
- */verbose*  
specifies that you want to display detailed information about differences between objects

Note that you use .NET standard regular expressions to define the */Include* and */Exclude* arguments. Therefore, you must escape the square brackets ( [ ] ) with the backslash character ( \ ). Regular expression syntax is beyond the scope of this online help; refer to your Microsoft .NET framework documentation for more information.

You must include the brackets ( [ ] ) in the string; if you specify the argument without the brackets, */Include:table:Product*, the *ProductCategory* table is included because it contains the string *Product*. The full SQL Server table names are qualified by the owner name in SQL Server 2000, and the schema name in SQL Server 2005/2008, and include brackets. For example (in SQL Server 2000):

```
[dbo].[Product]  
[dbo].[ProductCategory]
```

and so on. Therefore, the brackets indicate that you are specifying the full table name. To include the owner (or schema) name in the regular expression, you would need also to escape the dot ( . ):

```
/Include:table:\[dbo\]\ .\[Product\]
```

The pipe character ( | ) in a regular expression is interpreted as a logical OR. The character must be escaped by the caret character ( ^ ), to prevent the operating system shell from interpreting it as the pipe operator. (Note that if you want to use the caret character itself as part of your regular expression, it must be escaped by a second caret.)

## Using XML

You can use XML as follows:

```
<?xml version="1.0"?> <commandline>  
<database1>Products1</database1>  
<database2>Products2</database2>  
<verbose/>  
<include>Table:\[Product\]</include>  
</commandline>
```

To execute the comparison using the XML file, enter the following command:

```
sqldatacomapre /Argfile:xmlfilename.xml
```

where *xmlfilename* is the name of the XML file.

## Example: selecting tables with unrelated names

---

This example illustrates how you select a number of individual tables for comparison when their names are not related in any way.

In this example, the databases contain the following tables:

- Product
- Supplier
- ProductCategory
- SpecialOffer
- Customer
- Order
- Invoice

You are interested only in the differences between the *Product*, *Customer*, *Order*, and *Invoice* tables in two different versions of your database, *Customers1* and *Customers2*.

### Using the command line

To specify the list of tables to include, you use the */Include* switch. You could use an *Include* switch for each table that you want to compare. However, this could get unwieldy if you have a long list of tables. Instead, you can use the pipe character ( | ) to separate the table names:

```
sqldatacomapre /db1:Customers1 /db2:Customers2
    /include:table:\[Product\]^|Customer^|Order^|Invoice
```

where:

- */db1:Customers1*  
specifies that you want to compare the database *Customers1*
- */db2:Customers2*  
specifies that you want to compare the database *Customers2*
- */Include:table:\[Product\]^|Customer^|Order^|Invoice*  
specifies that you want to compare only the tables that have a name that includes the strings *[Product]*, or *Customer*, or *Order*, or *Invoice*

Note that you use .NET standard regular expressions to define the */Include* and */Exclude* arguments. Therefore, you must escape the square brackets ( [ ] ) with the backslash character ( \ ). Regular expression syntax is beyond the scope of this online help; refer to your Microsoft .NET framework documentation for more information.

You must include the brackets ( [ ] ) in the string; if you specify the argument without the brackets, */Include:table:Product*, the *ProductCategory* table is included because it contains the string *Product*. The full SQL Server table names are qualified by the owner name in SQL Server 2000, and the schema name in SQL Server 2005/2008, and include brackets. For example (in SQL Server 2000):

```
[dbo].[Product]
[dbo].[ProductCategory]
```

and so on. Therefore, the brackets indicate that you are specifying the full table name. To include the owner (or schema) name in the regular expression, you would need also to escape the dot ( . ):

```
/Include:table:\[dbo\]\ .\[Product\]
```

The pipe character ( | ) in a regular expression is interpreted as a logical OR. The character must be escaped by the caret character ( ^ ), to prevent the operating system shell from interpreting it as the pipe operator. (Note that if you want to use the caret character itself as part of your regular expression, it must be escaped by a second caret.)

## Using XML

You can use XML as follows:

```
<?xml version="1.0"?> <commandline>
<database1>Customers1</database1>
<database2>Customers2</database2>
<sync/>
<include>Table:\[Product\]|Customer|Order|Invoice</include>
</commandline>
```

Note that the pipe character ( | ) (and other operating system operators) do not have to be escaped by the caret character ( ^ ) when they are specified in the XML file, but ( < ) and ( > ) must be escaped.

To execute the comparison using the XML file, enter the following command:

```
sqldatacomapre /Argfile:XMLFileName.xml
```

where *XMLfilename* is the name of the XML file.



## Using XML to specify command line arguments

---

You can use an XML file to specify the arguments for the command line interface. You may want to do this because:

- An XML file is easier to read than a long and complex command line, particularly where complex rules for including and excluding objects are specified.
- You can easily transform an XML file into other formats using XSLT.  
For example, you could transform your argument file to HTML for presentation on a Web page.
- Using an XML file overcomes some limitations that can be a problem when you want to specify regular expressions as command line arguments.  
For example, you may want to use the pipe character ( | ) as part of a regular expression, but it causes problems when it is used at the command prompt; if you use an XML file you can use the pipe character with no problems.
- Most programming languages support XML, through built-in or freely available third-party libraries.  
This makes it easy to generate and process the XML file.

Create the XML file in the following format:

```
<?xml version="1.0"?> <commandline> <switch_name1/>
<switch_name2>switch_value</switch_name2>
.... </commandline>
```

For example, for the */Include* and */Exclude* switches, use the following format:

```
<include>objecttype:RegularExpression</include>
```

To execute the command line tools using an XML argument file as input, at the command prompt enter:

```
sqlcompare /Argfile:XMLfilename.xml
```

When using an XML file note that:

- you cannot specify any other switches on the command line except */verbose* or */quiet*
- multiple options should be separated with commas:

```
<options>n,oc,t</options>
```

### Examples

Below are some examples of XML files that can be used with the SQLDataCompare tool. The command line versions of the examples (using aliases) are also provided for comparison. To migrate changes in the XML examples, use the *<synchronize/>* element.

#### To compare the data in all tables in two local databases (Windows authentication):

Using an XML file:

```
<?xml version="1.0"?> <commandline>
<database1>FirstDatabaseName</database1>
<database2>SecondDatabaseName</database2>
</commandline>
```

#### Using the command line:

```
sqldatacomapre /db1:FirstDatabaseName /db2:SecondDatabaseName
```

#### To compare the data in all tables in databases on different hosts:

##### Using an XML file:

```
<?xml version="1.0"?> <commandline>
<database1>FirstDatabaseName</database1>
<server1>Hostname1</server1>
<database2>SecondDatabaseName</database2>
<server2>Hostname2</server2>
</commandline>
```

##### Using the command line:

```
sqldatacomapre /db1:FirstDatabaseName /db2:SecondDatabaseName
                /s1:Hostname1 /s2:Hostname2
```

#### To compare the data in all tables in two databases using SQL Server authentication:

##### Using an XML file:

```
<?xml version="1.0"?> <commandline>
<database1>FirstDatabaseName</database1>
<username1>Username1</username1>
<password1>Password1</password1>
<database2>SecondDatabaseName</database2>
<username2>Username2</username2>
<password2>Password2</password2>
</commandline>
```

##### Using the command line:

```
sqldatacomapre /db1:FirstDatabaseName /u1:Username1
                /p1:Password1 /db2:SecondDatabaseName /u2:Username2
                /p2:Password2
```

#### To compare the data only in tables whose name contains the word *Product*:

##### Using an XML file:

```
<?xml version="1.0"?> <commandline>
<database1>FirstDatabaseName</database1>
<database2>SecondDatabaseName</database2>
<include>Table:Product</include>
</commandline>
```

Using the command line:

```
sqldatacomapre /db1:FirstDatabaseName /db2:SecondDatabaseName  
/include:table:\[Product\]
```

**To compare the data only in tables whose name contains the word *Product*, except for the *ProductHistory* table:**

Using an XML file:

```
<?xml version="1.0"?>  
<commandline> <database1>FirstDatabaseName</database1>  
<database2>SecondDatabaseName</database2>  
<include>Table:Product</include>  
<exclude>Table:ProductHistory</exclude>  
</commandline>
```

Using the command line:

```
sqldatacomapre /db1:FirstDatabaseName /db2:SecondDatabaseName  
/include:table:\[Product\] /exclude:table:\[ProductHistory\]
```

**To compare the data in tables using an index as a comparison key**

Using an XML file:

```
<?xml version="1.0"?> <commandline>  
<database1>FirstDatabaseName</database1>  
<database2>SecondDatabaseName</database2>  
<comparisonkeys>TableName:IndexName</comparisonkeys>  
</commandline>
```

Using the command line:

```
sqldatacomapre /db1:FirstDatabaseName /db2:SecondDatabaseName  
/comparisonkeys:TableName:IndexName
```

**To compare a backup file with a database:**

Using an XML file:

```
<?xml version="1.0"?> <commandline>  
<backup1>D:\MSSQL\BACKUP\BackupOfFirstDatabase.sqlb</backup1>  
<database2>SecondDatabaseName</database2>  
</commandline>
```

Using the command line:

```
sqldatacompare /backup1:D:\MSSQL\BACKUP\BackupOfFirstDatabase.sqlb  
/db2:SecondDatabaseName
```

**To retrieve verbose output of the data differences between two databases:**

Using an XML file:

```
<?xml version="1.0"?> <commandline>  
<database1>FirstDatabaseName</database1>  
<database2>SecondDatabaseName</database2>  
<verbose/>  
</commandline>
```

#### Using the command line:

```
sqldatacomapre /db1:FirstDatabaseName /db2:SecondDatabaseName  
/verbose
```

#### **To migrate data changes from the first database to the second database:**

##### Using an XML file:

```
<?xml version="1.0"?> <commandline>  
<database1>FirstDatabaseName</database1>  
<database2>SecondDatabaseName</database2>  
<synchronize/>  
</commandline>
```

##### Using the command line:

```
sqldatacomapre /db1:FirstDatabaseName /db2:SecondDatabaseName  
/synchronize
```

## Switches used in the command line (inexplicably broken)

---

You can use the following switches with the SQL Data Compare command line:

### **/allowidenticaldatabases**

Alias: */aid*

Lists all matching objects when compared databases are identical.

By default, when the compared databases are identical, an error message is returned and no objects are listed. Use of this switch allows the comparison of identical databases. This is useful if you suspect two databases may be identical and want to verify this in detail:

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction  
/allowidenticaldatabases
```

### **/argfile:<argfile>**

Runs a file containing an XML argument specification:

```
sqldatacompare /argfile:XMLFileName.xml
```

For more information see Using XML to specify command line arguments (page 73).

### **/backup1:<filename1>;<filename2>;...;<filenameN>**

Alias: */b1*

Allows one or more backup files to be specified in place of the first database:

```
sqldatacompare /backup1:D:\BACKUPS\WidgetStaging.bak  
/db2:WidgetStaging
```

To specify more than one backup file, the file names are separated using semicolons:

```
sqldatacompare /backup1:D:\BACKUPS\WidgetDev_Full.bak;  
D:\BACKUPS\WidgetDev_Diff.bak /db2:WidgetDev
```

### **/backup2:<filename1>;<filename2>;...;<filenameN>**

Alias: */b2*

Allows one or more backup files to be specified in place of the second database:

```
sqldatacompare /db1:WidgetStaging /backup2:D:\BACKUPS\WidgetStaging.bak
```

### **/backupset1:<backupset>**

Alias: */bs1*

Specifies which backup sets to use for the first backup.

*/backupset1* must be used with */backup1*

If you are using native SQL Server backups and the backup contains multiple backup sets, use the */backupset1* switch to specify the sets which make up the first backup, and use the */backupset2* switch to specify the sets which make up the second:

```
sqldatacompare /backup1:D:\BACKUP\WidgetDev.bak  
  /backupset1:"2008-09-23 Full Backup" /db2:WidgetDevelopment
```

If the backup set switches are not specified, SQL Data Compare uses the latest backup set.

### ***/backupset2:<backupset>***

Alias: */bs2*

Specifies which backup set to use for the second backup:

```
sqldatacompare /db1:WidgetProduction /backup2:D:\BACKUP\WidgetDev.bak  
  /backupset2:"2008-09-23 Full Backup"
```

### ***/casesensitive***

Alias: */cs*

Makes the comparison case-sensitive. For example: if you compare *Enormous\_Widget* and *enormous\_widget* when this switch is not specified, they are shown as identical.

```
sqldatacompare /database1:WidgetStaging  
  /database2:WidgetProduction /casesensitive
```

For more information, see Case-sensitive comparisons (page 46).

### ***/columns:<table or view name as regular expression>:<column name1>,<column name2>***

Alias: */cols*

Specifies which columns are included in the comparison. If you do not specify any columns, all columns are compared.

The name of the table or view is specified using a regular expression - you do not have to specify fully-qualified names. It is recommended that you use a regular expression which matches only one table or view.

Multiple column names are separated using commas:

```
sqldatacompare /database1:WidgetStaging /database2:WidgetProduction  
  /columns:\[WidgetPrices\]:Price,DateValidTo
```

### ***/comparisonkeys:<table or view name as regular expression>:<index name>***

Alias: */ck*

Specifies a unique index to be used to identify rows for comparison.

The name of the table or view is specified using a regular expression - you do not have to specify fully-qualified names. It is recommended that you use a regular expression which matches only one table or view:

```
sqldatacompare /database1:WidgetStaging /database2:WidgetProduction
               /columns:\[WidgetPrices\]:Price
               /comparisonkeys:\[WidgetPrices\]:PK_WidgetPrices
```

Note that:

- */comparisonkeys* must be used with the */columns* switch
- with */comparisonkeys* you can only specify an index as the comparison key, no other columns can be specified

To specify a comparison key that is not an index, use the GUI to set up and save a project with the settings you require. You can then use that project from the command line with the */project* switch.

For more information on using the GUI to set comparison keys, see *Selecting the comparison key*, under *Selecting tables and views* (page 16).

### ***/database1:<database1>***

Alias: */db1*

Specifies the first database to compare:

```
sqldatacompare /database1:WidgetStaging /database2:WidgetProduction
```

### ***/database2:<database2>***

Alias: */db2*

Specifies the second database to compare:

```
sqldatacompare /database1:WidgetStaging /database2:WidgetProduction
```

### ***/exclude:<object type>:<regular expression>***

Specifies a list of tables or views to exclude:

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
               /exclude:table:\[WidgetPrices\]
```

Here, */exclude:table:\[WidgetPrices\]* specifies that you do not want to compare the table *WidgetPrices* - all other tables and views are compared.

To specify more than one object or object type type for exclusion, use multiple */exclude* switches.

For a more detailed example of how to use the */include* and */exclude* switches, see *Example: selecting tables with unrelated names* (page 71).

### ***/export:<directory>***

Alias: */e*

Exports the comparison results to the specified directory as a .csv file.

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
               /export:C:\WidgetResults
```

### **/exportidenticaltables**

Alias: */eit*

By default, the .csv file output by */export* includes only tables and views with differences. Use this switch to include tables and views which are identical in the comparison results.

This switch can only be used with */export*.

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
               /export:C:\WidgetResults /exportidenticaltables
```

### **/force**

Alias: */f*

Forces the overwriting of any output files that already exist. If this switch is not used and a file of the same name already exists, the program exits with the exit code indicating an IO error.

### **/include:<object type>:<regular expression>**

Specifies a list of tables and views to be included in the comparison.

You can use an */include* switch for each table or view that you want to compare. However, this is unwieldy if you have a long list of tables. Instead, you can use the pipe character ( | ) to separate the names:

```
sqldatacompare /db1:Customers1 /db2:Customers2
               /include:table:\[Product\]^|Customer^|Order^|Invoice
```

*/include* and */exclude* switches are applied in the order they are specified.

For a more detailed example of how to use the */include* switch, see: Example: selecting tables with unrelated names (page 71).

### **/ignoreadditional**

Alias: */a*

Rows that are present only in the second database (*db2*) are not compared:

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
               /ignoreadditional
```

### **/ignoredifferent**

Alias: */d*

Rows that are different in both databases are not compared:



```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
  /ignoredifferent
```

### **/ignoreidentical**

Alias: */i*

Rows that are the same in both databases are not compared:

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
  /ignoreidentical
```

### **/ignoremissing**

Alias: */m*

Rows that are present only in the first database (*db1*) are not compared:

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
  /ignoremissing
```

### **/options:<option1>,<option2>,<option3>**

Alias: */o*

Applies the project configuration options used during comparison or synchronization:

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
  /options:Default,UseChecksumComparison
```

For a detailed list of these options, see Options used in the command line (page 85).

### **/out:<fileName>**

Redirects console output to the specified file:

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
  /out:C:\output file
```

### **/outputproject:<filename>**

Alias: */outpr*

Writes the settings used for the comparison to the specified SQL Data Compare project file:

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
  /exclude:table:\[WidgetPrices\]
  /outputproject:"C:\WidgetProject.sdc"
```

This generates a SQL Data Compare project file. These files end with a *.sdc* extension. If the file already exists, an error will occur unless you have also used the */force* switch.

Note that only the details of the data sources used, and any objects specified by */include* or */exclude* switches are saved in the project file. Project configuration options are not saved.

## **`/outputwidth:<columns>`**

Forces the width of console output.

This can be used to ensure that database object names are not truncated, and that SQL script lines are not wrapped or broken. This is particularly useful when redirecting output to a file as it allows you to overcome the limitations of the default console width of 80 characters.

## **`/password1:<password1>`**

Alias: `/p1`

The password for the first database.

You must also provide a user name. If you do not specify a user name and password combination, integrated security is used:

```
sqldatacompare /db1:WidgetStaging /username1:User1
               /password1:P@ssw0rd /db2:WidgetProduction /username2:User2
               /password2:Pa$$w0rd
```

## **`/password2:<password2>`**

Alias: `/p2`

The password for the second database.

You must also provide a user name. If you do not specify a user name and password combination, integrated security is used:

## **`/project`**

Alias: `/pr`

Uses a SQL Data Compare project (`.sdc`) file for the comparison.

Note that only the details of the data sources used, and any objects specified by `/include` or `/exclude` switches are saved in the project file. Project configuration options are not saved. If you wish to use anything other than the default options you must use the `/options` switch to specify the options you want to use:

```
sqldatacompare /project:WidgetProject.sdc /options:AllowIdenticalDatabases
```

## **`/scriptencoding`**

Alias: `/senc`

Specifies the character encoding used when writing the SQL script file. The arguments for the `/scriptencoding` switch are:

<code>UTF8</code>	UTF-8 encoding, without preamble
<code>UTF8WithPreamble</code>	UTF-8 encoding, with 3-byte preamble
<code>Unicode</code>	UTF-16 encoding

*ASCII*                      ASCII encoding

The default setting is UTF-8.

For example:

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
               /scriptfile:"C:\Scripts Folder\WidgetSyncScript.sql"
               /scriptencoding:ASCII
```

### **/scriptfile:<scriptfile>**

Alias: */sf*

Generates a SQL script which can be executed at a later time to carry out the synchronization. If the file already exists an error will occur, unless you use the */force* switch:

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
               /scriptfile:"C:\Scripts Folder\WidgetSyncScript.sql"
```

### **/server1:<server1>**

Alias: */s1*

Specifies the SQL Server on which the first database (*/db1*) or backup (*/b1*) is located. If an explicit path is not specified, it defaults to *Local*.

```
sqldatacompare /Server1:Widget_Server\SQL2008 /db1:WidgetStaging
               /db2:WidgetProduction
```

### **/server2:<server2>**

Alias: */s2*

Specifies the SQL Server on which the second database (*/db2*) or backup (*/b2*) is located. If an explicit path is not specified, it defaults to *Local*.

```
sqldatacompare /db1:WidgetStaging /Server2:Widget_Server\SQL2008
               /db2:WidgetProduction
```

### **/synchronize**

Alias: */sync*

Synchronizes the databases after comparison.

Changes are migrated from *<database1>* to *<database2>*:

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
               /synchronize
```

### **/username1:<username1>**

Alias: */u1*

The user name for the first database.

If no user name is specified, integrated security is used.

```
sqldatacompare /db1:WidgetStaging /username1:User1 /password1:P@ssw0rd  
/db2:WidgetProduction /userName2:User2 /password2:Pa$$w0rd
```

**/username2:<username2>**

Alias: /u2

The user name for the second database.

If no username is specified, integrated security is used:

```
sqldatacompare /db1:WidgetStaging /username1:User1  
/password1:P@ssw0rd /db2:WidgetProduction /userName2:User2  
/password2:Pa$$w0rd
```

## Options used in the command line

---

You can set project configuration options by using the */Options* switch.

For example, when mapping objects, SQL Data Compare considers underscores in object names to be differences by default. Therefore, if the objects [dbo].[Widget\_Prices] and [dbo].[WidgetPrices] were identical, they would not be mapped, and so could not be compared. To successfully compare these objects, use:

```
/Options:IgnoreUnderscores
```

SQL Data Compare now treats those objects as identical, and they can be compared.

To specify multiple options, separate the options using commas:

```
/Options:<option1>,<option2>,<option3>
```

If you do not explicitly set any options, the defaults are used. See *Defaults* below.

Note that there are changes to the command line syntax of SQL Data Compare in version 8. The names, aliases, and behaviour of some switches and options is different to that of earlier versions.

For more information, see *Changes to the command line in SQL Data Compare 8*

### Defaults

If you do not specify any options, the following default options apply:

- IgnoreSpaces
- IncludeTimestamps
- IncludeIdentities
- DisableKeys
- OutputComments
- ReseedIdentity
- (IgnoreCase) - This option is deprecated, and case insensitive comparison is now the default behaviour.

If you want to use these defaults with additional options, specify the *default* argument and the additional options. For example:

```
/Options:Default,TrimTrailingSpaces,CompressTemporaryFiles
```

If you do not specify the *default* argument, only the options you explicitly specify apply.

To specify no options, use the *none* argument.

Further options are detailed below.

### CaseSensitiveObjectDefinition

Alias: *cs*

Treats object definitions as case sensitive when mapping. For example, *Table\_A* and *table\_a* would not be mapped automatically.

### **CompressTemporaryFiles**

Alias: *ctf*

Compresses the temporary files that SQL Data Compare generates while performing the comparison. This makes it less likely that you will run out of disk space when comparing very large databases.

### **Default**

Alias: *d*

Applies the default options.

### **DisableAndReenableDDLTriggers**

Alias: *drd*

DDL triggers can cause problems when you run the synchronization. Select this option to disable any enabled DDL triggers before synchronizing the databases, and re-enable those triggers following synchronization.

### **DisableAndReenableDMLTriggers**

Alias: *t*

Disables DML triggers on tables and views before synchronizing the databases, and then re-enables those triggers following synchronization.

### **DisableKeys**

Alias: *k*

Disables foreign keys before synchronizing the databases, and then re-enables those foreign keys following synchronization. Note that in some circumstances foreign keys will be dropped and re-created rather than disabled and re-enabled.

### **DoNotOutputCommentHeader**

Alias: *nc*

When this option is selected, comment headers are not included in the output script.

### **DropConstraintsAndIndexes**

Alias: *c*

Drops primary keys, indexes, and unique constraints before synchronizing the databases, then re-creates them following the synchronization.

If the primary key, index, or unique constraint is the comparison key, it cannot be dropped.

### **ForceBinaryCollation**

Alias: *fbc*

Forces binary collation for all string data types, irrespective of column collation, resulting in a case-sensitive comparison. When this option is selected and the comparison key is a string, this may result in slower performance because the indexes are not used.

### **ForceCheck**

New option. Forces any constraints (for example, those on foreign keys) disabled by SQL Data Compare to be re-enabled with CHECK.

### **IgnoreCase**

This option is deprecated, and case insensitive comparison is now the default behaviour.

### **IgnoreSpaces**

Alias: *is*

When mapping objects for comparison, spaces in the names of objects are considered by default. This option ignores spaces in the names of objects, enabling them to be mapped. For example [dbo].[Widget Prices] is mapped to [dbo].[WidgetPrices].

### **IgnoreUnderscores**

Alias: *iun*

When mapping objects for comparison, underscores in the names of objects are considered by default. This option ignores underscores in the names of objects, enabling them to be mapped. For example, [dbo].[Widget\_Prices] is mapped to [dbo].[WidgetPrices].

### **IncludeIdentities**

Alias: *iid*

Includes identity columns in the comparison.

Note that you cannot synchronize a view if it includes an identity column.

### **IncludeIndexedViews**

Alias: *v*

Includes views in the comparison. Views can be synchronized only if the referenced rows are from a single table, and the referenced columns are simple. For example, they must not include identity columns or computed columns.

### **IncludeTimestamps**

Alias: *its*

Includes timestamp columns in the comparison.

Note that timestamp columns cannot be synchronized.

### **None**

Alias: *n*

To specify no options, use the *none* argument.

### **OutputComments**

Alias: *oc*

Includes comments in the synchronization SQL script.

### **ReSeedIdentity**

Alias: *rsi*

Re-seeds identity columns so that identity values in the database you are updating match values in the source database.

### **TransportCLRBinary**

Alias: *tclr*

When this option is selected, SQL Data Compare uses the binary representation of CLR types in the synchronization SQL script. If this option is not selected, CLR data types are represented as strings.

### **TrimTrailingSpaces**

Alias: *tts*

When this option is selected, and the data in two columns differs only by the number of spaces at the end of the string, SQL Data Compare treats those columns as identical.

Note that this option does not apply to CLR columns, or XML columns.

### **UseChecksumComparison**

Alias: *ucc*



When this option is selected, SQL Data Compare performs a checksum prior to comparison. Data is compared only if the checksums differ. You can use this option to improve the performance of SQL Data Compare.

Note that in SQL Server 2000 databases, db\_owner permissions are required to use this option.

## **UseTransactions**

Alias: *ut*

When this option is selected transactions are used in the synchronization SQL scripts, enabling changes to be rolled back if the synchronization fails. BEGIN TRANSACTION is inserted at the beginning of the synchronization SQL script, and COMMIT TRANSACTION at the end of the script.

## Exit codes used in the command line

---

If a task you are performing with the SQL Data Compare command line interface fails, and you do not see an error message explaining the reason for the failure, you may see one of the exit codes detailed below:

### **3 - Illegal argument duplication**

Some arguments cannot be used more than once in a command line.

### **8 - Unsatisfied argument dependency**

There is an unsatisfied argument dependency or violated exclusion when the command line is run. For example:

- */arg2* depends on */arg1* but you have specified */arg2* without specifying */arg1*
- */arg2* cannot be used with */arg1* but you have used both

### **32 - Value out of range**

The numeric value supplied for an argument is outside the range of valid values for that argument.

### **33 - Value overflow**

The value supplied for an argument is too large.

### **34 - Invalid value**

The value supplied for an argument is invalid.

### **35 - Invalid license**

Software license or trial period has expired.

### **62 - High level parser error**

SQL Data Compare encountered high level errors when parsing a scripts folder. Use */ignoreParserErrors* to force SQL Data Compare to continue without exiting.

### **63 - Databases identical**

The databases being compared are identical or no objects have been included.

## **64 - Command line usage error**

The command line was used incorrectly. For example, an incorrect flag, or incorrect syntax may have been used.

## **65 - Data error**

Data required by SQL Data Compare is invalid or corrupt.

This often indicates that a constraint violation occurred while running the synchronization script. Check your schema to ensure that when you are migrating data you do not violate any constraints in the schema. You should either modify the data you are migrating, or change the SQL Data Compare command line options to resolve the problem.

## **69 - Resource unavailable**

A resource or service required to run SQL Data Compare is unavailable.

## **73 - Failed to create report**

The report was not created.

## **74 - I/O error**

For example, this is returned if SQL Data Compare attempts to write to a file that already exists, and the */force* switch has not been set.

## **77 - Insufficient Permission**

The action cannot be completed because the user does not have the necessary permission.

## Frequently asked questions for the command line

---

### Licensing

#### How do I activate the command line tools?

To use the command line, you must have a SQL Data Compare Professional Edition or SQL Toolbelt license. If you do not have a license, you will be able to use the command line for 14 days. If you have any further questions send us an e-mail at [info@red-gate.com](mailto:info@red-gate.com) (mailto:info@red-gate.com).

#### Can I distribute the command line tools?

If you have a SQL Data Compare Professional Edition or SQL Toolbelt license, you can distribute the command line tools. For more information, see Integrating the command line with applications (page 65).

### Comparing databases

#### How can I include or exclude specific tables?

You can use the `/include` or `/exclude` switch with regular expressions to do this. For examples, see Example: selecting single tables for comparison (page 69) and Example: selecting tables with unrelated names (page 71).

Note that tables with zero rows are compared but are not listed in the command line output.

#### How do I schedule a database comparison?

You can use the Microsoft® Windows® Scheduled Task wizard to schedule a comparison by creating a script to run the comparison.

For example, to compare the data in two databases, you could create the following script:

```
C:  
cd path_to_installation_folder  
sqldatacompare /db1:FirstDatabaseName  
/db2:SecondDatabaseName >> log_file
```

where:

- *path\_to\_installation\_folder* is the path to the folder in which you installed the tool. Alternatively, you can add the installation folder to your PATH environment variable and omit this line.
- *FirstDatabaseName* and *SecondDatabaseName* are the names of the databases that you want to compare.
- *log\_file* is the full path to the log file. For example, *C:\SQLCmdLine\Log.txt*

In this example MS-DOS batch scripting is used, a basic scripting language that is supported on all versions of Windows. If preferred, you could use VBScript, JScript, PHP, Perl, Python or any other scripting language of your choice.

Save the script as a .bat file. You can then specify the .bat file as the program to run from within the Scheduled Task wizard by browsing to it.

### **What objects are included or excluded when I use a project?**

When you use a project, all objects that were selected for inclusion when the project was saved are automatically included; you do not need to explicitly include them using the /include switch. You can override the inclusion by specifying the /exclude switch as required. For examples, see Example: selecting single tables for comparison (page 69) and Example: selecting tables with unrelated names (page 71).

### **How do I integrate the command line tools with applications?**

For information about how to integrate the command line tools with applications that you distribute to your customers, see Integrating the command line with applications (page 65).

SQL Data Compare enables you to compare a backup with other data sources. This is useful, for example, when you want to retrieve the data from a backup and compare it with your database without running a restore operation or copying the backup from a remote network.

If you are comparing two backups, you do not need SQL Server to be installed on your computer.

Note that:

- Comparing backups is available only in SQL Data Compare Professional edition.
- SQL Data Compare can retrieve the data from full or differential backups. However, it does not support partial, filegroup, or transaction log backups.
- When you run a comparison using a backup, SQL Data Compare locks the backup files when it reads them, and you cannot overwrite, move, or delete them.
- SQL Data Compare does not read the log records of backup files, so if the database schema was modified while the backup was being created, it may not be shown as modified in the comparison results.
- You can specify a backup as the target; however, note that backups cannot be modified.

### Comparing and synchronizing backups

You can:

- compare a backup with another data source  
See Setting data sources (page 12)
- create a synchronization script from a backup

When you have selected a backup as the target, the deployment wizard creates a script to update the database from which the backup was created. Backups cannot be modified directly.

When a backup is the source, and a database is the target, the deployment script will synchronize the database with the backup.

### Compatibility with backups

You can compare backups from SQL Server 2008, SQL Server 2005 or SQL Server 2000 databases.

To use a differential backup as a data source, you must also add the associated full backup.

Note that SQL Data Compare does not support using partial, filegroup, or transaction log backups as a data source.

SQL Data Compare supports:

- native SQL Server backups
- SQL Backup backups  
You can use backups created with SQL Backup version 3 or later; you can use compressed or encrypted backups.

When you set up a comparison that uses backup files, SQL Data Compare does not support:

- tables that do not have a primary key, unique index, or unique constraint
- views

- string representations of CLR types  
Only the binary representation is supported

## Acknowledgements

---

### Trademarks and registered trademarks

Red Gate is a registered trademark of Red Gate Software Ltd registered in the U.S. Patent and Trademark Office.

.NET Reflector and SQL Compare are registered trademarks of Red Gate Software Ltd registered in the U.S. Patent and Trademark Office.

ANTS Performance Profiler, ANTS Memory Profiler, .NET Reflector Pro, Exception Hunter, Schema Compare for Oracle, SQL Backup, SQL Data Compare, SQL Comparison SDK, SQL Dependency Tracker, SQL Doc, SQL HyperBac, SQL Log Rescue, SQL Monitor, SQL Multi Script, SQL Packager, SQL Prompt, SQL Refactor, SQL Scripts Manager, SQL Storage Compress, SQL Toolbelt, SQL Virtual Restore, and Exchange Server Archiver are trademarks of Red Gate Software Ltd.

Microsoft, Windows, Windows 98, Windows NT, Windows 2000, Windows 2003, Windows XP, Windows Vista, Windows 7, Visual Studio, and other Microsoft products referenced herein are either registered trademarks or trademarks of Microsoft Corporation.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates.

InstallShield is a registered trademark and service mark of InstallShield Software Corporation.

PagerDuty is a registered trademark of PagerDuty, Inc.

WordPress is a registered trademark of the WordPress Foundation.

### Copyright information

All Red Gate applications are © Red Gate Software Ltd 1999 - 2013

SQL Backup, SQL Compare, SQL Data Compare, SQL Packager, and SQL Prompt contain software that is Copyright © 1995 - 2005 Jean-loup Gailly and Mark Adler.

SQL Doc includes software developed by Aspose (<http://www.Aspose.com>).

SQL Backup contains software that is Copyright © 2003 - 2008 Terence Parr. Refer to the ACKNOWLEDGEMENTS.txt file in your SQL Backup installation directory for the full license text.