1. SQL Data Compare 10 documentation . . . 3
1.1 Requirements . . . 4
1.2 Installing . . . 5
1.3 Licensing . . . 6
1.3.1 Activating . . . 7
1.3.2 Deactivating . . . 13
1.3.3 Troubleshooting licensing and activation . . . 16
1.4 Upgrading . . . 19
1.4.1 Using Check for Updates . . . 20
1.4.2 Troubleshooting Check for Updates errors . . . 22
1.5 Setting up the comparison . . . 23
1.5.1 Working with projects . . . 24
1.5.2 Setting data sources . . . 26
1.5.3 Selecting tables and views . . . 29
1.5.4 Mapping objects . . . 32
1.5.5 Mapping owners . . . 34
1.5.6 Filtering the comparison with a WHERE clause . . . 35
1.5.7 Setting project options . . . 36
1.5.8 Setting application options . . . 40
1.5.9 What is a comparison key? . . . 41
1.5.10 Which data types can be compared? . . . 42
1.6 Reviewing the comparison results . . . 43
1.6.1 Viewing the comparison results . . . 44
1.6.2 Tables and views that cannot be compared . . . 46
1.6.3 Viewing the row differences . . . 47
1.6.4 Viewing the data . . . 49
1.6.5 Exporting the comparison results . . . 51
1.6.6 Printing the comparison results . . . 53
1.7 Deploying the data sources . . . 57
1.7.1 Setting up the deployment . . . 58
1.7.2 Using the Deployment Wizard . . . 60
1.7.3 Deploying views . . . 64
1.7.4 Backing up before deployment . . . 65
1.7.5 Warnings . . . 67
1.7.6 Data that wasn't deployed . . . 68
1.8 Working with other data sources . . . 69
1.8.1 Working with backups . . . 70
1.8.2 Working with scripts folders . . . 71
1.9 Using the command line . . . 74
1.9.1 Getting started with the SQL Data Compare command line . . . 75
1.9.2 Changes to the command line in SQL Data Compare 8 . . . 76
1.9.3 Command line basics . . . 79
1.9.4 Integrating the command line with applications . . . 82
1.9.5 Examples using the command line . . . 83
1.9.5.1 Simple examples using the command line . . . 84
1.9.5.2 Example - selecting single tables for comparison . . . 87
1.9.5.3 Example - selecting tables with unrelated names . . . 89
1.9.5.4 Using XML to specify command line arguments . . . 91
1.9.5.5 Deploying a database from source control . . . 95
1.9.6 Command line syntax . . . 97
1.9.6.1 Switches used in the command line . . . 98
1.9.6.2 Options used in the command line . . . 107
1.9.6.3 Exit codes used in the command line . . . 111
1.10 SQL Server Management Studio add-in . . . 113
1.10.1 Using the add-in . . . 114
1.11 Getting more from SQL Data Compare . . . 116
1.11.1 Deploying to SQL Azure databases . . . 117
1.11.2 Comparing databases on different SQL Server versions . . . 118
1.11.3 Creating a rollback script . . . 119
1.11.4 Getting better performance out of SQL Data Compare . . . 120
1.12 Worked examples . . . 124
1.12.1 Worked example - deploying data in two databases . . . 125
1.12.2 Worked example - restoring from a backup file . . . 137
1.13 Troubleshooting . . . 143
1.13.1 Common issues . . . 144
1.13.1.1 Case-sensitive comparisons . . . 145
1.13.1.2 Cleaning up a SQL script after SQL Compare or SQL Data Compare . . . 146
1.13.1.3 Comparing the data of two tables in the same database . . . 147
1.13.1.4 Improving the performance of SQL Data Compare . . . 148
1.13.1.5 Reseed applying "incorrect" identity values . . . 152
1.13.1.6 SQL Data Compare showing differences in two identical databases . . . 153
1.13.1.7 Tables or views that could not be compared . . . 154

1.13.1.8 Troubleshooting comparison and deployment performance problems . . . 156
1.13.1.9 Using a filter on a column on related (joined) tables . . . 157
1.13.2 Error messages . . . 159
1.13.2.1 NULL textptr passed to UPDATETEXT function when running deployment . . . 160
1.13.2.2 Running migration scripts using SqlCmd.exe . . . 161
1.13.2.3 This SQL Server has been optimized for x concurrent queries . . . 162
1.13.2.4 Troubleshooting System.OutOfMemoryException during comparison . . . 163
1.13.3 Unexpected behavior / technical questions . . . 164
1.13.3.1 How much free hard disk space is required? . . . 165
1.13.3.2 How to force SQL Compare and SQL Data Compare to use an encrypted connection . . . 166
1.13.3.3 Minimum database permissions for SQL Data Compare . . . 167
1.13.3.4 Using Windows authentication logons between domains . . . 168
1.13.4 Logging and log files . . . 169
1.14 Release notes and other versions . . . 170
1.14.1 SQL Data Compare 10.7 release notes . . . 171
1.14.2 SQL Data Compare 10.4 release notes . . . 172
1.14.3 SQL Data Compare 10.0 release notes . . . 173
1.14.4 SQL Data Compare 9.0 release notes . . . 174
1.14.5 SQL Data Compare 8.0 release notes . . . 175
1.14.6 SQL Data Compare 7 release notes . . . 177
2. Example GS . . . 178

SQL Data Compare 10 documentation

About SQL Data Compare

With SQL Data Compare, you can compare and deploy the data in two Microsoft SQL Server databases. You can also compare a backup with a database, a scripts folder, or another backup.

Quick links

SQL Data Compare product page

Lastest version: Release notes for SQL Data Compare 10.7

Selecting data sources to compare

Viewing the comparison results

Deploying your changes

Requirements

To use SQL Data Compare you need:

- One of the following Microsoft Windows operating systems:
  - Windows Server 2003
  - Windows 7
  - Windows Server 2008
  - Windows 8
- Microsoft .NET Framework 3.5 or later
- 256 MB RAM
- 200 MB hard disk space

The following versions of Microsoft SQL Server are supported:

- SQL Server 2005
- SQL Server 2008
- SQL Server 2008 R2
- SQL Server 2012
- Windows Azure SQL Database
- SQL Server on Amazon RDS

Installing

Most Redgate products are available as part of a bundle. You can select which individual products to install when you run the installer.

When you install a non-free product, you have 14 days to evaluate the product. For the DLM Automation Suite, DLM Automation Suite for Oracle, SQL Source Control, Schema Compare for Oracle, Data Compare for Oracle, and Source Control for Oracle, you have 28 days. For more information, see Licensing.

To install a Redgate product:

1. Download the product from the website.
2. Run the installer and follow the instructions.

The product is listed on the **Start** menu under **Red Gate**.

Licensing

When you install most Redgate products (apart from free ones), you have **14 days** to evaluate them without purchase.

For a few products, you have 28 days: DLM Automation Suite, DLM Automation Suite for Oracle, SQL Prompt, SQL Source Control, Source Control for Oracle.

If you need more time to evaluate a product, email licensing@red-gate.com.

Finding your serial number

When you buy a license for a product, we'll send you an invoice that contains your serial number to activate the product. Your invoice shows how many instances of a product the serial number can be used to activate. For information about how to activate, see Activating.

If you can't find your invoice, you can view your serial numbers at red-gate.com/myserialnumbers. You'll need to log in to your Redgate account with the email address and password you provided when you bought the product.

> If you need to reinstall products on the same computer (eg after installing a new operating system), you can reactivate them using the same serial number. This doesn't affect the number of distinct activations for the serial number. For information about moving a serial number to a different computer, see below.

Serial numbers for bundles and suites

If you've bought a bundle or suite of products, your serial number activates all the products in the bundle or suite. For bundles containing both server and client tools (such as the SQL DBA Bundle) you will have two serial numbers.

If you deactivate a bundle or suite serial number, all products using that serial number will be deactivated.

For information on which products are included in a bundle, see Bundle history.

Changing the serial number used to activate a product

To change the serial number used to activate a product, on the **Help** menu, select **Enter Serial Number**. For some products, you will need to deactivate the old serial number first.

Moving a serial number to a different computer

To move a serial number to a different computer, deactivate the serial number on the old computer, then use it to activate the product on the new computer.

To deactivate a serial number, on the **Help** menu, select **Deactivate Serial Number**. If the Deactivate Serial Number menu item isn't available, use the deactivation tool.

If you can't deactivate a serial number, use the Request Extra Activations page to request more activations for your serial number. You'll need to provide your serial number and the reason for the additional activations.

Activating

> This page applies to a number of Redgate products, so the screenshots below may not match your product.

When you activate a product with your serial number, the licensing and activation program sends an activation request to the Redgate activation server, using checksums of attributes from your computer. The checksums sent to the activation server do not contain any details that might pose a security risk. The activation server returns an activation response and an encrypted key to unlock the software. The licensing and activation program should activate your product within a few seconds.

If you experience problems with activating your products, you'll be directed to activate manually.

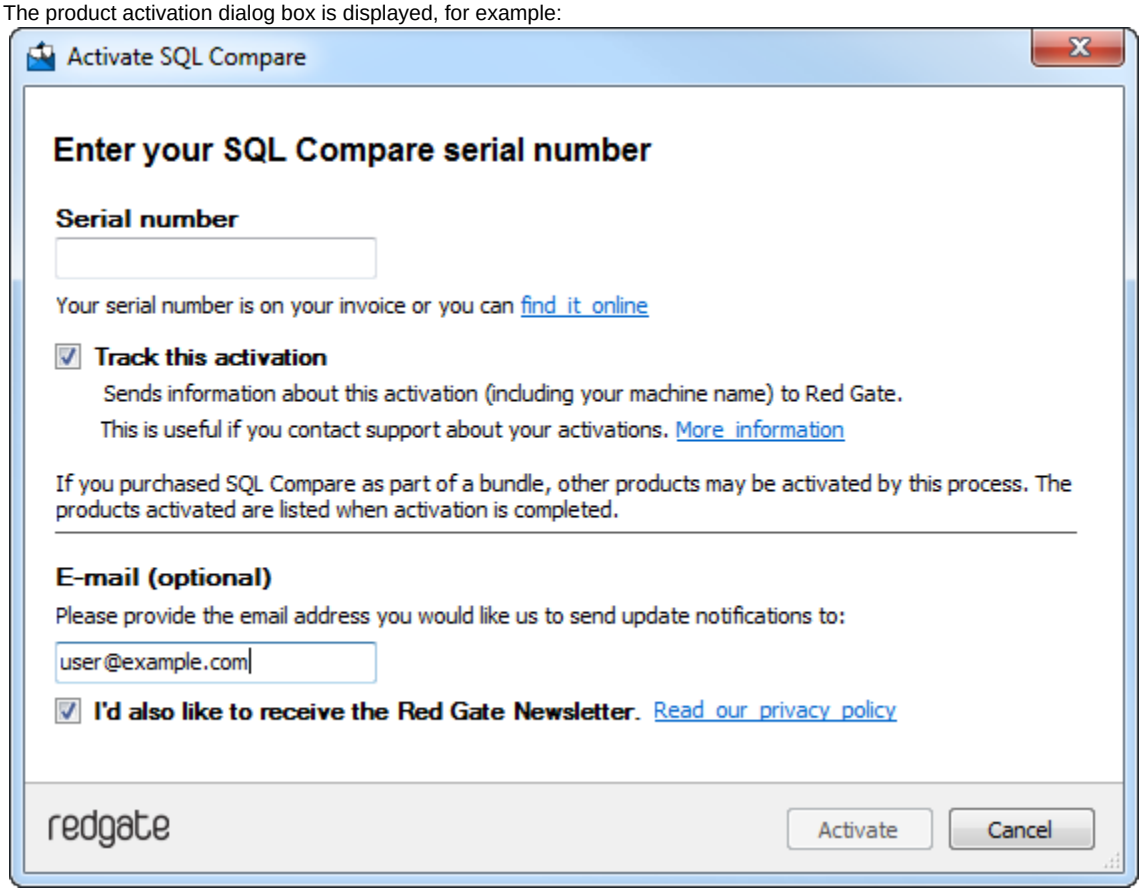
- Activating using the GUI
- Activating using the command line
- Manual activation

Activating using the GUI

> These instructions apply to a number of Redgate products, so the screenshots below may not match your product.

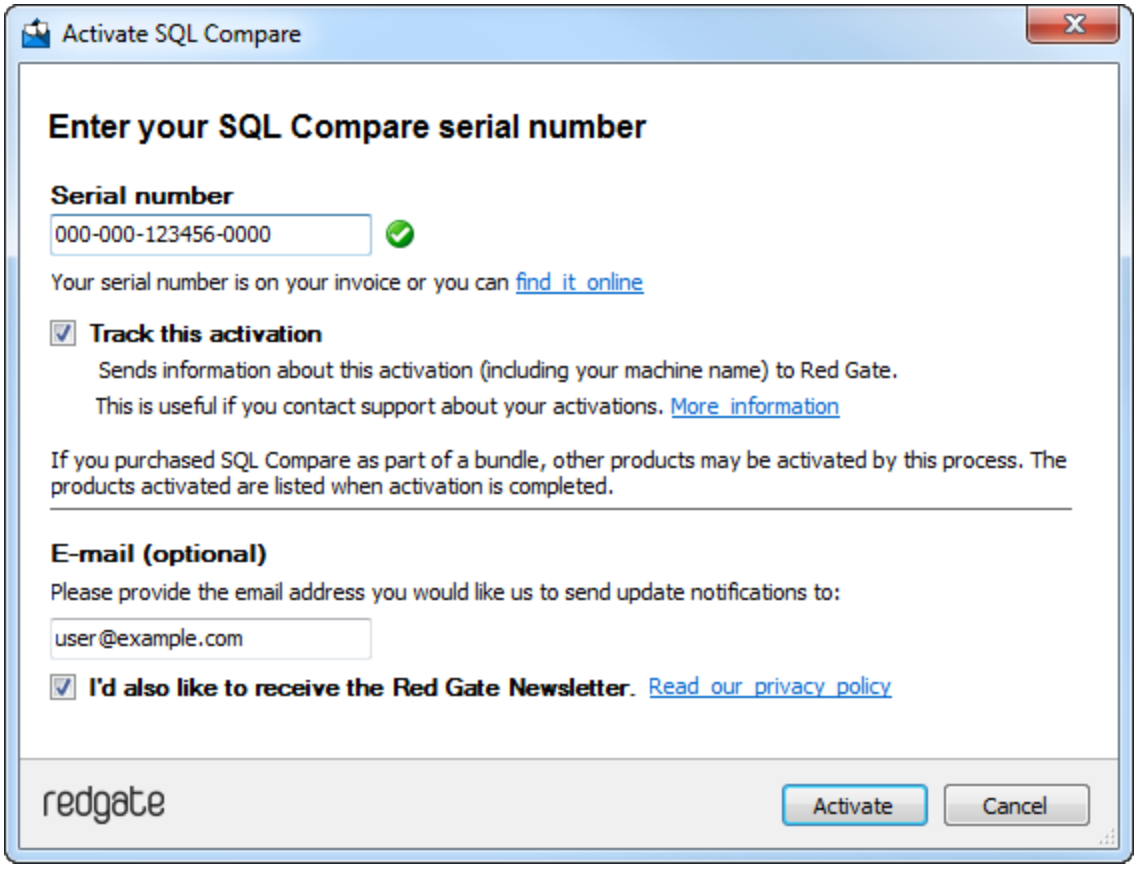
To activate your products:

1. On the **Help** menu, click **Enter Serial Number**.
   The product activation dialog box is displayed, for example:

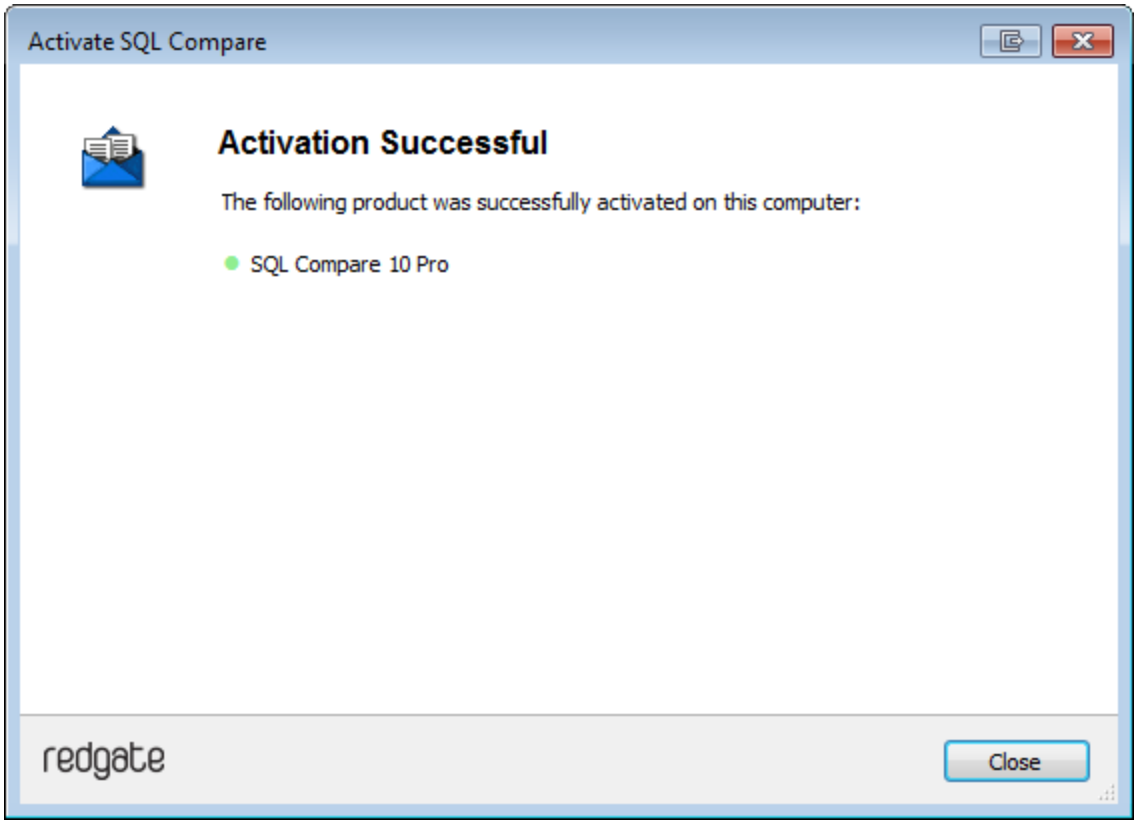


2. Enter your serial number.
   When you have entered a valid serial number,

   is displayed next to the serial number box:



3. If you want to receive email updates from Redgate, enter your email address.
   The list of identifiers and your email address may already be populated using information available to the licensing client from the Windows installation on your computer. No information is sent back to Redgate when the fields are populated.
   When you activate your product, the optional information you entered is recorded by Redgate with your serial number. Your email address is not linked to the data collected should you consent to participate in the Quality Improvement Program provided with some Red Gate products.
4. Click **Activate**.
   Your activation request is sent to the Red Gate activation server.
   When your activation has been confirmed, the **Activation successful** page is displayed, for example:



If there is a problem with your activation request, an error dialog box is displayed. For information about activation errors and what you can do to resolve them, see Troubleshooting licensing and activation errors. Depending on the error, you may want to try manual activation.

5. Click **Close**.
   You can now continue to use your product.

Activating using the command line

Open a command prompt, navigate to the folder where your product executable file is located and run a command with the following syntax:

```
<name of productEXE> /activateSerial:<serialNumber>
```

For example:

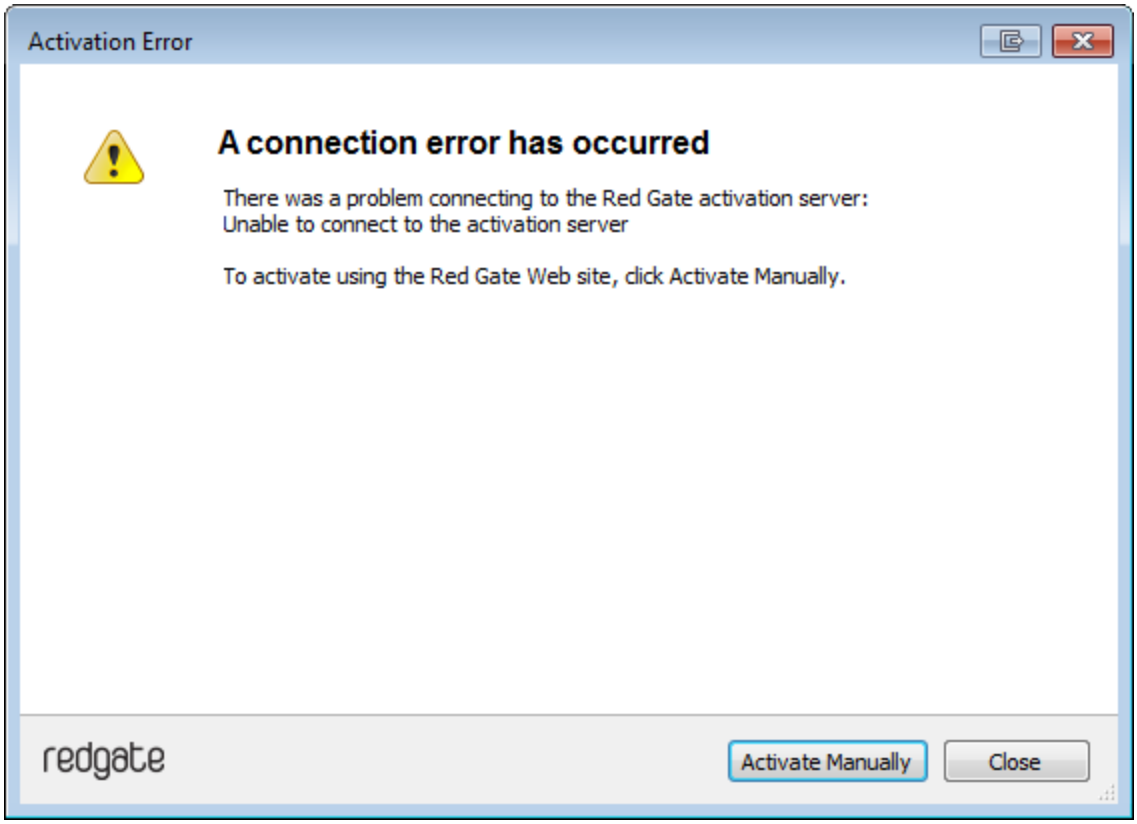
```
sqlcompare /activateSerial:123-456-789012-ABCD
```

The product activation dialog box is displayed. Follow the instructions below.

Manual activation

Manual activation enables you to activate products when your computer does not have an internet connection or your internet connection does not allow SOAP requests. You will need access to another computer that does have an internet connection.

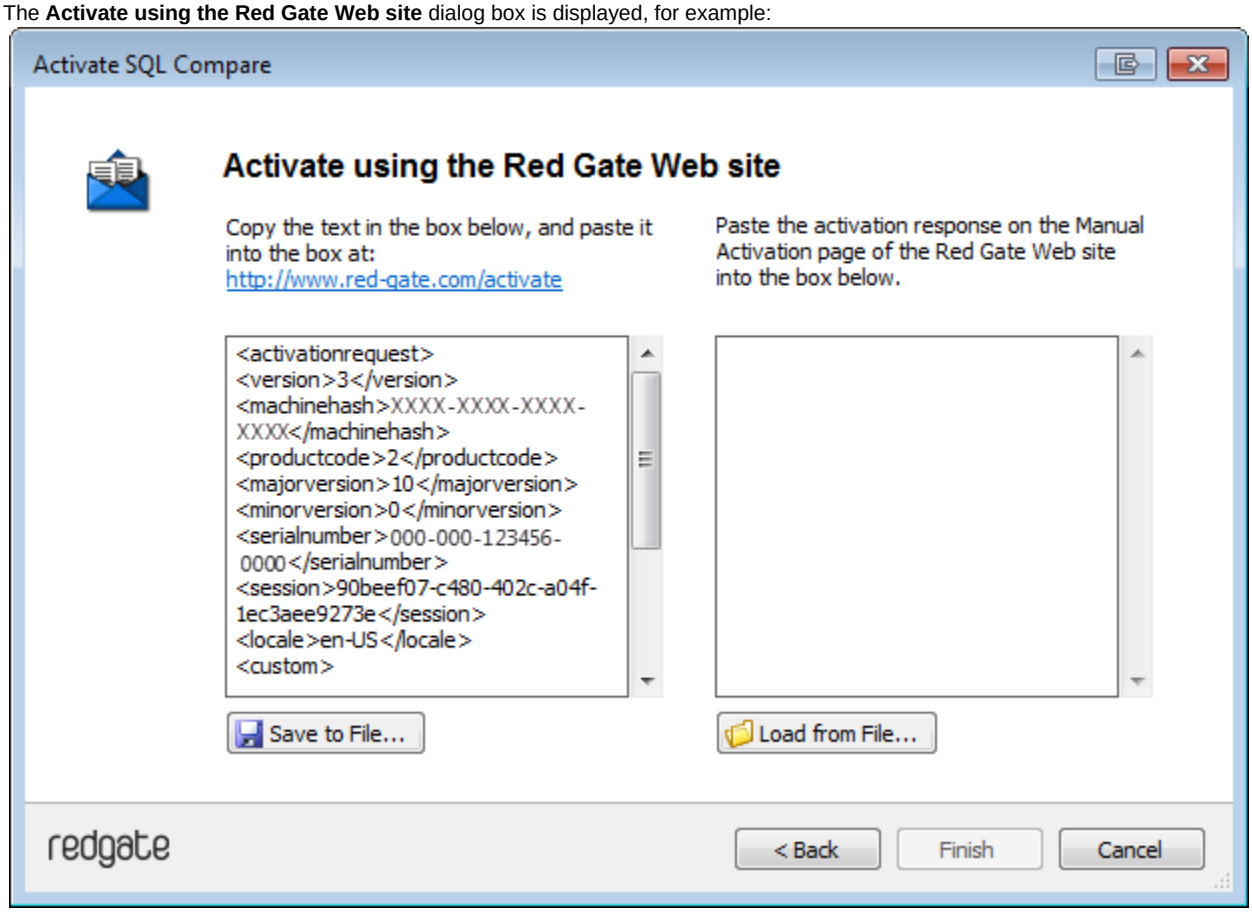
You can use manual activation whenever the **Activation Error** dialog box is displayed and the **Activate Manually** button is available, for example:



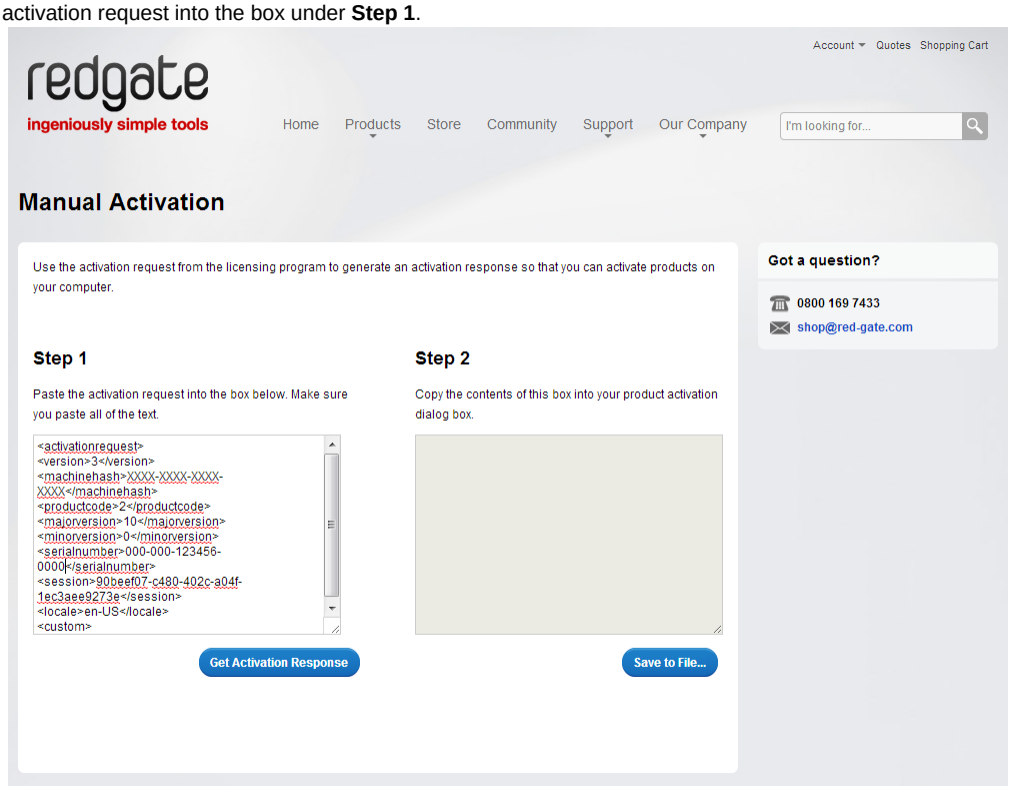
To activate manually:

1. On the error dialog box, click **Activate Manually**.

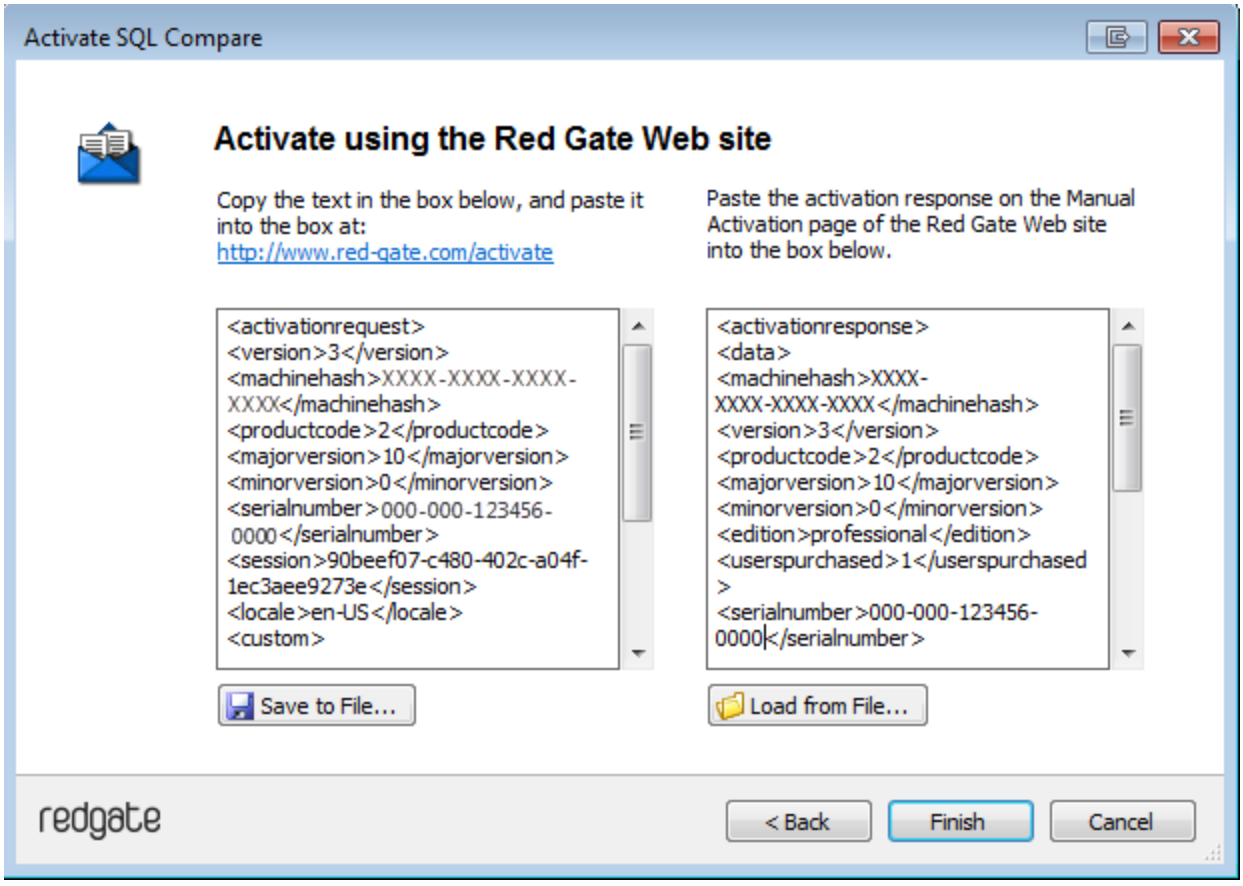
   The **Activate using the Red Gate Web site** dialog box is displayed, for example:



2. Copy all of the activation request, and **leave this dialog box open** (if you close the dialog box, you may have to start again). Alternatively you can save the activation request, for example to a location on your network or to a USB device.
3. On a computer that has an Internet connection, go to the **Manual Activation** page at http://www.red-gate.com/activate and paste the activation request into the box under **Step 1**.



4. Click **Get Activation Response**.
5. When the activation response is displayed under **Step 2**, copy all of it.
   Alternatively you can save the activation response to a .txt file.
6. On the computer where the licensing and activation program is running, paste the activation response or if you saved it, load it from the file.



7. Click **Finish**.
   The **Activation successful** page is displayed.
8. Click **Close**.
   You can now continue to use your product.

Deactivating

> This page applies to several Redgate products, so the screenshots below may not match your product.

Download deactivation tool

You can use the deactivation tool to deactivate a serial number so you can reuse it on another computer. You can also use it to deactivate serial numbers for products you've uninstalled.

> When you deactivate a serial number for a bundle of products, all the products in the bundle are deactivated. For information about what products are in your bundle, see Bundle history.

To deactivate a serial number, your computer must have an internet connection. If you can't deactivate a serial number, you can request additional activations for that serial number. You may need to do this if:

- your computer doesn't have an internet connection
- your network uses a proxy server that interrupts contact between the product and the Redgate activation server
- your serial numbers aren't displayed in the deactivation tool (eg if the product installation is corrupted)

Deactivating using the command line

Open a command prompt, navigate to the folder where your product executable file is located and run a command with the following syntax:

```
<productEXE> /deactivateSerial
```

For example:

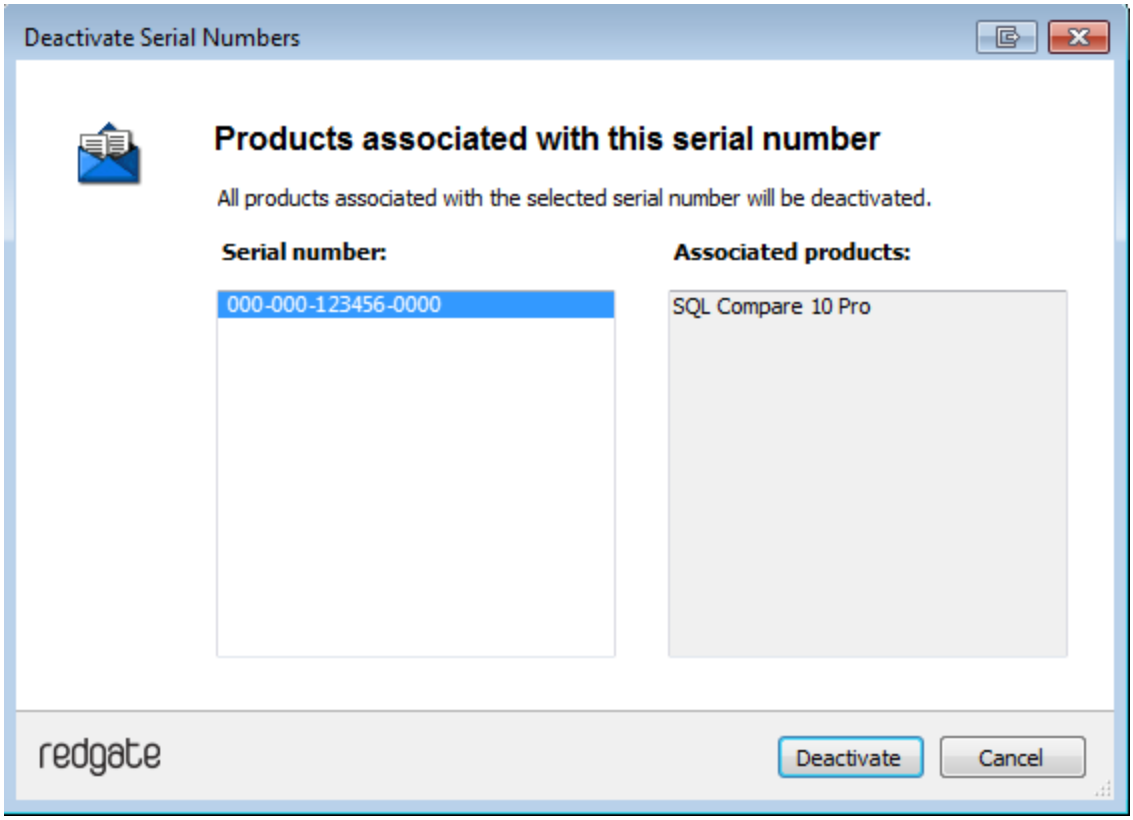
```
sqlcompare /deactivateSerial
```

The **Deactivate Serial Numbers** dialog box is displayed. Follow the instructions below.

Deactivating using the GUI

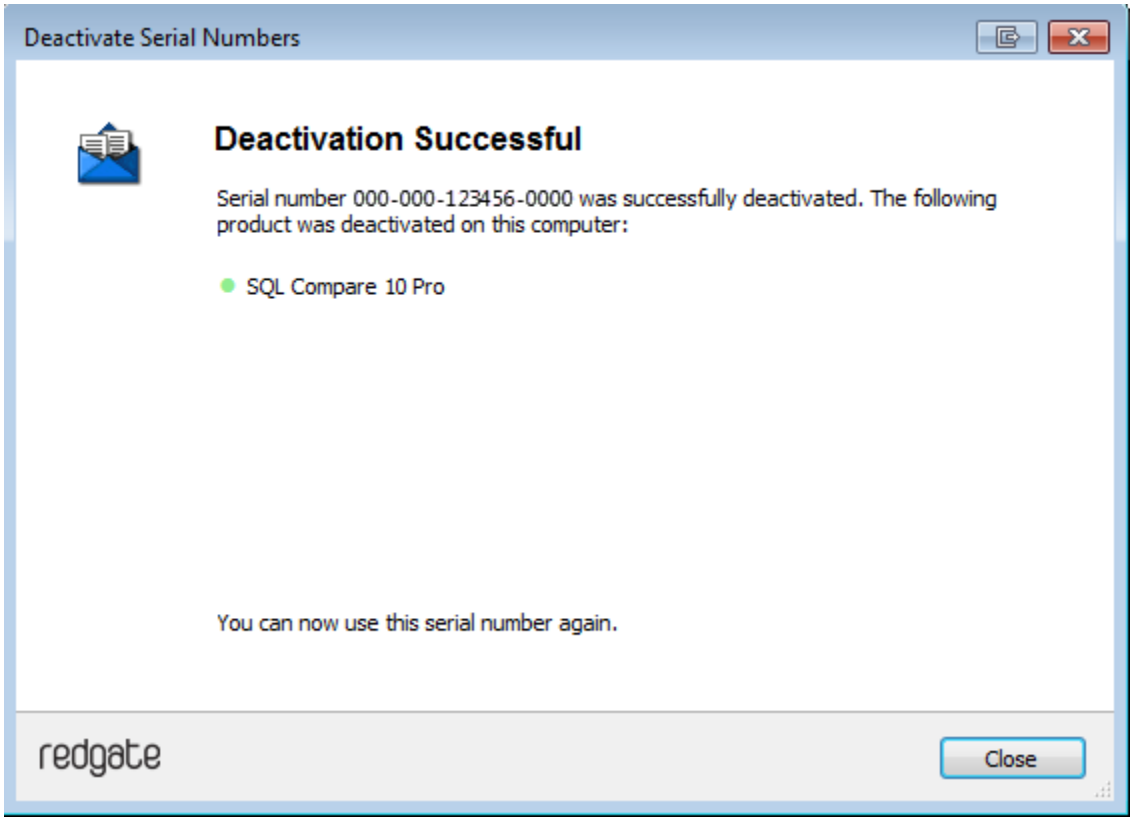
To deactivate your products:

1. Start the deactivation tool. To do this, either download the tool and run the executable file, or on the **Help** menu of the product, click **Deactivate Serial Number**.
   The **Deactivate Serial Numbers** dialog box is displayed. For example:



If you're running the executable file, the dialog box displays all the serial numbers for Red Gate products that have been activated on your computer.
If the serial number is for a bundle, all the products in the bundle are displayed under **Associated products**.

2. Select the serial number you want to deactivate and click **Deactivate**.
   Your deactivation request is sent to the Red Gate activation server.
3. When your deactivation has been confirmed, the **Deactivation successful** page is displayed. For example:



If there's a problem with your deactivation request, an error dialog box is displayed. For information about deactivation errors and how to resolve them, see Troubleshooting licensing and activation errors.

4. Click **Close**. You can now use this serial number on a different computer.

Troubleshooting licensing and activation

This page provides information about errors you may encounter when you activate Redgate products:

- The number of activations for this serial number has been exceeded
- This serial number has been disabled
- This serial number was for a trial extension
- This serial number is not registered with the activation server
- This serial number is not for <product name>
- This serial number is not for this version
- The activation request is in the wrong format
- The activation request contains an invalid machine hash
- The activation request contains an invalid session
- The activation request contains an invalid serial number
- The activation request contains an invalid product code or version number
- There's a problem deactivating your serial number
- This serial number is not activated on this computer
- Products not activated on this computer

The number of activations for this serial number has been exceeded

This error message is displayed when a serial number is activated on more computers than the number of licenses that were purchased for that serial number.

When you purchase products from Redgate, we send you an invoice that includes your serial numbers. The serial numbers enable you to activate the software a number of times, depending on how many licenses you purchased and the terms in the license agreement. When this limit is reached, you will see this error message.

To fix the problem, you can:

- deactivate the product on another computer to free up a license
- purchase more licenses
- request additional activations for your serial number

This serial number has been disabled

This error message is displayed when you try to activate a product using a serial number that Redgate has disabled.

When you upgrade a product, your existing serial numbers will be disabled and we will issue new ones with your invoice. If you cannot find your new serial numbers, you can review them at http://www.red-gate.com/myserialnumbers

Redgate will also disable serial numbers for non-payment of invoices or breach of the terms in the license agreement. If you think we have disabled your serial numbers in error, email licensing@red-gate.com

This serial number was for a trial extension

This error message is displayed when you have requested a trial extension and you try to reuse the serial number that was provided for the trial extension; trial extensions can be used one time only.

To continue using the product, you need to purchase it.

This serial number is not registered with the activation server

This error message is displayed when the serial number you entered does not exist on the Redgate activation server.

To find out your serial numbers, check your invoice or go to http://www.red-gate.com/myserialnumbers

This serial number is not for <product name>

This error message is displayed when the serial number you entered is not for the product you are trying to activate.

To find out your serial numbers, check your invoice or go to http://www.red-gate.com/myserialnumbers

This serial number is not for this version

This error message is displayed when the serial number you entered is for a different version of the product you are trying to activate.

If the serial number is for an older version of the product, and you don't have that version installed on your computer, you can download it from the Release notes and other versions page.

If you want to upgrade to the latest version of the product, go to the Upgrade center to get a quote or purchase an upgrade, or email sales@red-gate.com.

The activation request is in the wrong format

This error message is displayed:

- if your internet connection does not allow SOAP requests.
  Try using manual activation; on the error dialog box, click **Activate Manually**, and then follow the instructions that are displayed.
- if you are activating by email and there is a problem with the format of the activation request.
  Check that you copied and pasted all of the activation request.
  Alternatively, try using manual activation. Go to http://www.red-gate.com/activate and paste your activation request under **Step 1**.
- when you are using manual activation and there is a problem with the format of the activation request. If the format is incorrect, for example part of the request is missing, the Redgate activation server cannot process the request.
  Check that you copied and pasted all of the activation request.

For more information about activating manually, see Manual activation.

The activation request contains an invalid machine hash

This error message is displayed:

- if your internet connection does not allow SOAP requests.
  Try using manual activation; on the error dialog box, click **Activate Manually**, and then follow the instructions that are displayed. For more information, see Manual activation.
- when you are using manual activation and there is a problem with the format of the *machinehash* element in the activation request. The *machinehash* is a checksum of attributes from your computer. We use the *machinehash* to identify computers on which our products have been activated. If the format of the *machinehash* element is incorrect, the Redgate activation server cannot process the request.
  Check that you copied and pasted the activation request correctly.

The activation request contains an invalid session

This error message is displayed:

- if your internet connection does not allow SOAP requests.
  Try using manual activation; on the error dialog box, click **Activate Manually**, and then follow the instructions that are displayed. For more information, see Manual activation.
- when you are using manual activation and there is a problem with the format of the activation request. If the format of the *session* elemen t is incorrect, the Redgate activation server cannot process the request.
  Check that you copied and pasted the activation request correctly.

The activation request contains an invalid serial number

This error message is displayed:

- if your internet connection does not allow SOAP requests.
  Try using manual activation; on the error dialog box, click **Activate Manually**, and then follow the instructions that are displayed. For more information, see Manual activation.
- when you are using manual activation and there is a problem with the format of the activation request. If the format of the serial number is incorrect, the Redgate activation server cannot process the request.
  Check that you copied and pasted the activation request correctly.

The activation request contains an invalid product code or version number

This error message is displayed:

- if your internet connection does not allow SOAP requests.
  Try using manual activation; on the error dialog box, click **Activate Manually**, and then follow the instructions that are displayed. For more information, see Manual activation.
- when you are using manual activation and there is a problem with the format of the activation request. If the product code or version numbers are incorrect, the Redgate activation server cannot process the request.
  Check that you copied and pasted the activation request correctly.

There's a problem deactivating your serial number

This error message is displayed if your computer is not connected to the internet or your internet connection does not allow SOAP requests. You cannot deactivate a serial number if your computer does not have an internet connection.

Try deactivating again later. If the problem persists, contact your system administrator.

If you require more activations because you cannot deactivate your serial number, you can request them on the Request Extra Activations page.

This serial number is not activated on this computer

This error message is displayed when you try to deactivate a serial number that has not been activated on your computer.

If you think the product installation on your computer is corrupt, you can try re-activating the product, and then deactivating the product again.

If you require more activations because you cannot deactivate your serial number, you can request them on the Request Extra Activations page.

Products not activated on this computer

This error message is displayed when you try to deactivate a serial number for a bundle of Redgate products and those products were not activated on your computer.

If you think the product installation on your computer is corrupt, you can try re-activating the product, and then deactivating the product again.

If you require more activations because you cannot deactivate your serial number, you can request them on the Request Extra Activations page.

Upgrading

Minor releases are free for all users. For example, if you have a license for version 7.0 of a product, you can upgrade to version 7.1 at no cost. W hen you download and install a minor release, the product is licensed with your existing serial number automatically.

Major releases are free for users with a current Support and Upgrades contract. For example, if you have a license for version 7 of a product, you can upgrade to version 8 at no cost. When you download and install a major release, the product is licensed with your existing serial number automatically.

If you don't have a current Support and Upgrades contract, installing a major release will start a free 14-day trial. You'll need to buy a new license and activate the product with your new serial number.

To check whether you have a current Support and Upgrades contract or see the cost of upgrading to the latest major version of a product:

- visit the Upgrade Center
- email sales@red-gate.com
- call:
  - 1 866 733 4283 (toll free USA and Canada)
  - 0800 169 7433 (UK freephone)
  - +44 (0)870 160 0037 (rest of world)

To check the latest version of a product, see Current versions.

How to upgrade

You can download the latest version of a product using Check for Updates, the Upgrade Center, or the Redgate website.

- If you download the latest version from the Upgrade Center or our website, you need to run the installer to upgrade the product.

  Some Redgate products are available as part of bundle. You can select which products you want to upgrade when you run the installer.

- If you use Check for Updates, the installer runs automatically.

You can install the latest *major* version of any product (other than SQL Backup Pro) on the same machine as the previous version. For example, you can run version 9 and version 10 in parallel. However, installing a *minor* release will upgrade the existing installation.

To revert to an earlier version, uninstall the later version, then download and install the version you want from the Release notes and other versions page. You can use a serial number for a later version to activate an earlier version.

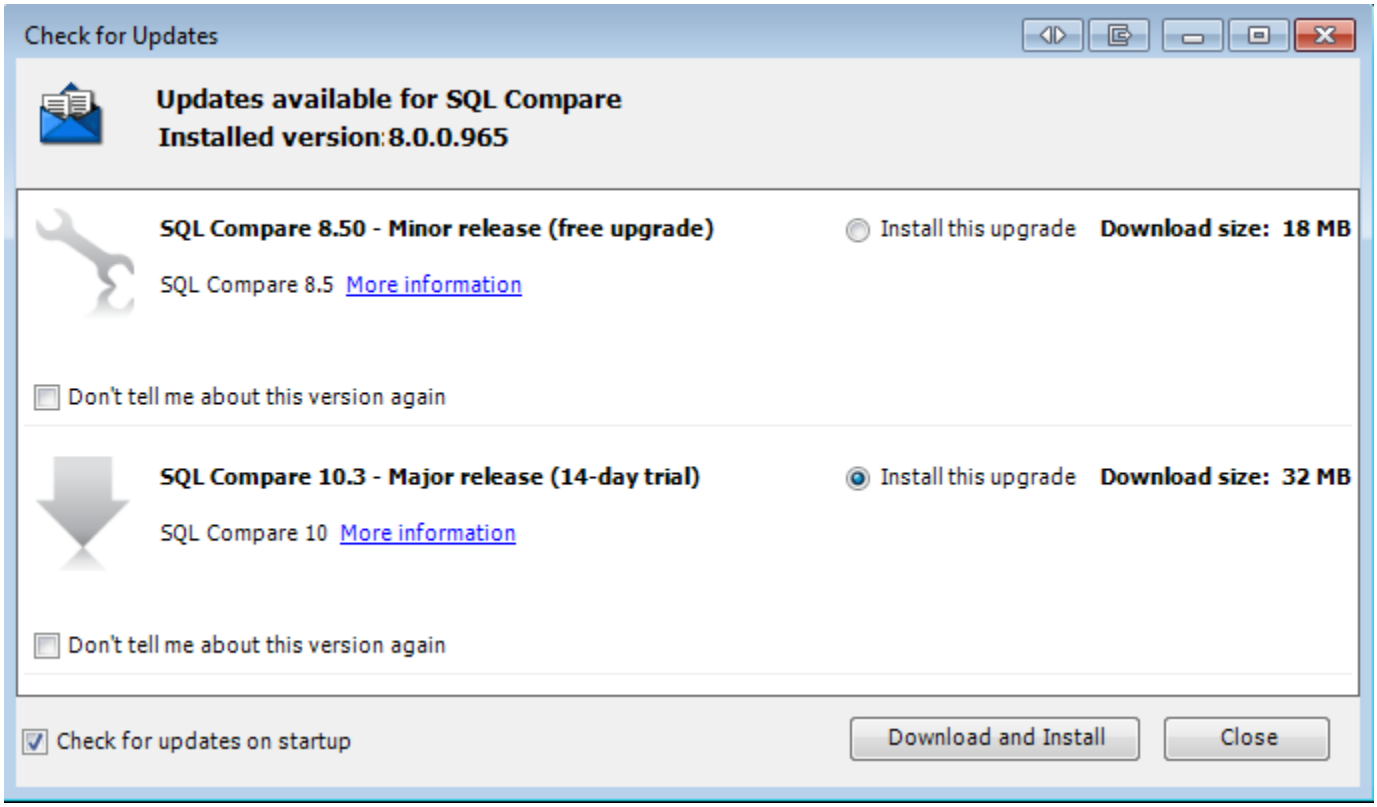
Using Check for Updates

> This page applies to several Redgate products, so the screenshots below may not match your product.

The Check for Updates service checks whether a more recent version of the product is available to download. To use the service, your computer must have a connection to the internet. If your internet connection uses a proxy server, make sure your web browser connection settings are configured correctly.

> The Check for Updates service doesn't work with automatic configuration scripts.

To check for updates for a Redgate product, on the **Help** menu, click **Check for Updates**. Any available updates are listed:



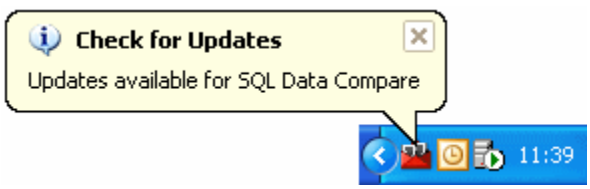
To view the full release details in your default web browser, click **More information**.

To get the update, click **Download and Install**. If you have a choice of updates, choose by selecting **Install this upgrade**, and then click **Download and Install**.

> The installer will ask you to close the program. If you're upgrading an add-in, you'll also be asked to close the host program (SQL Server Management Studio, Visual Studio or Query Analyzer).

About the Check for Updates service

When you start the application, the Check for Updates service informs you automatically when there are updates available:



If you don't want to receive these notifications for the product, clear the **Check for updates on startup** check box.

If you don't want the Check for Updates service to inform you about a particular update again, select the **Don't tell me about this version again** check box. The Check for Updates service will still inform you of new updates when they become available.

Troubleshooting Check for Updates errors

For details about how to use the Check for Updates service, see Using Check for Updates.

Error: There is a problem saving the download file to your computer

This error message is displayed if:

You don't have enough disk space

The Check for Updates service downloads the updates to the location defined by the *RGTEMP* environment variable, or the *TMP* variable if the *RGTEMP* variable doesn't exist.
If you don't have enough disk space, you can change the environment variable to a location with more space.

> Changing the *RGTEMP* or the *TMP* variables will affect other programs that use those variables. The *RGTEMP* variable affects only Redgate programs. For information about environment variables, see your Windows documentation.

There's a problem with permissions on your computer

The Check for Updates service downloads the updates to the location defined by the *RGTEMP* environment variable, or the *TMP* variable if the *RGTEMP* variable does not exist. If your user account doesn't have permissions to write to the location specified by these environment variables, contact your system administrator.

There's a problem with the download file on the Redgate web server

Contact Redgate support.

Error: There is a problem with the network connection

This error message is displayed if:

Your internet connection dropped while the Check for Updates service was downloading the updates

Try checking for updates again later.

Proxy authentication failed

Check your user name and password.

Your computer can't connect to the Check for Updates service.

Contact your system administrator. If you're using a proxy server, check it's configured correctly (see Control Panel > Internet Options > Connections).

> The Check for Updates service doesn't work with automatic configuration scripts.

There's a problem with the download file on the Redgate web server

Contact Redgate support.

Setting up the comparison

These pages explain how to set up a comparison in SQL Data Compare:

- Working with projects
- Setting data sources
- Selecting tables and views
- Mapping objects
- Mapping owners
- Filtering the comparison with a WHERE clause
- Setting project options
- Setting application options
- What is a comparison key?
- Which data types can be compared?

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.

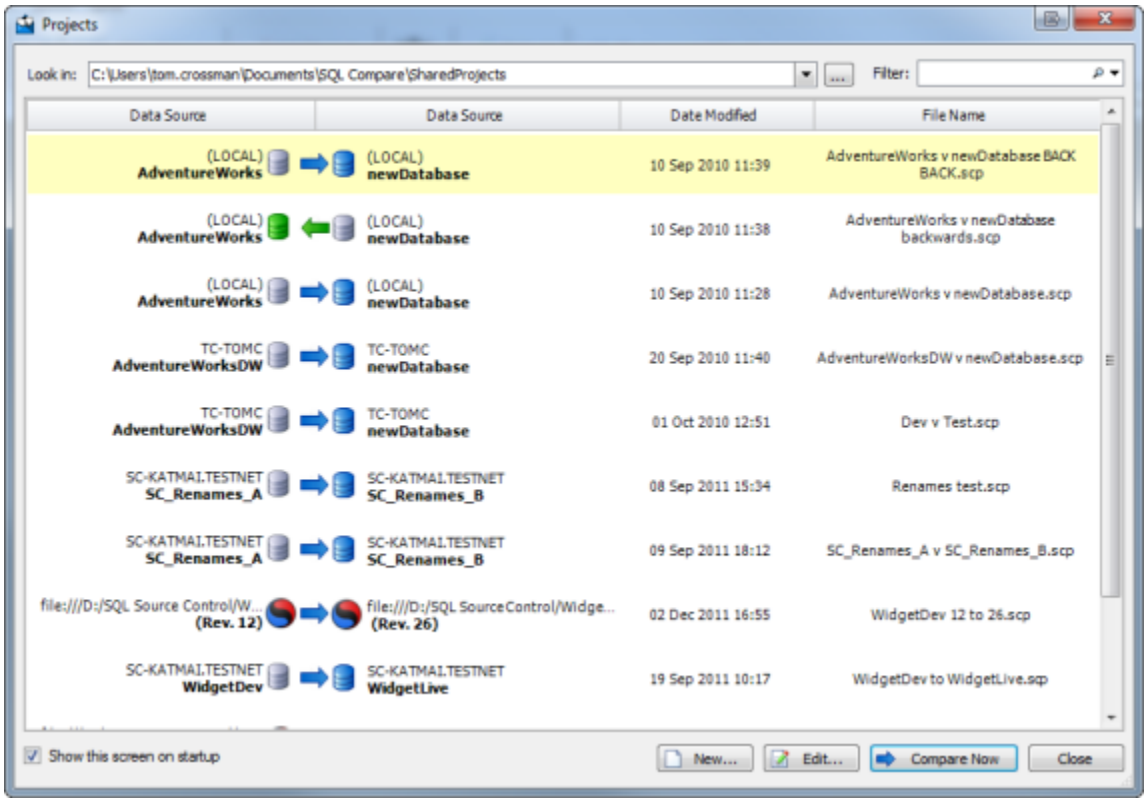
A project contains details of:

- which data sources you selectedIf 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
- which tables and views you selected
- any WHERE clause 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.

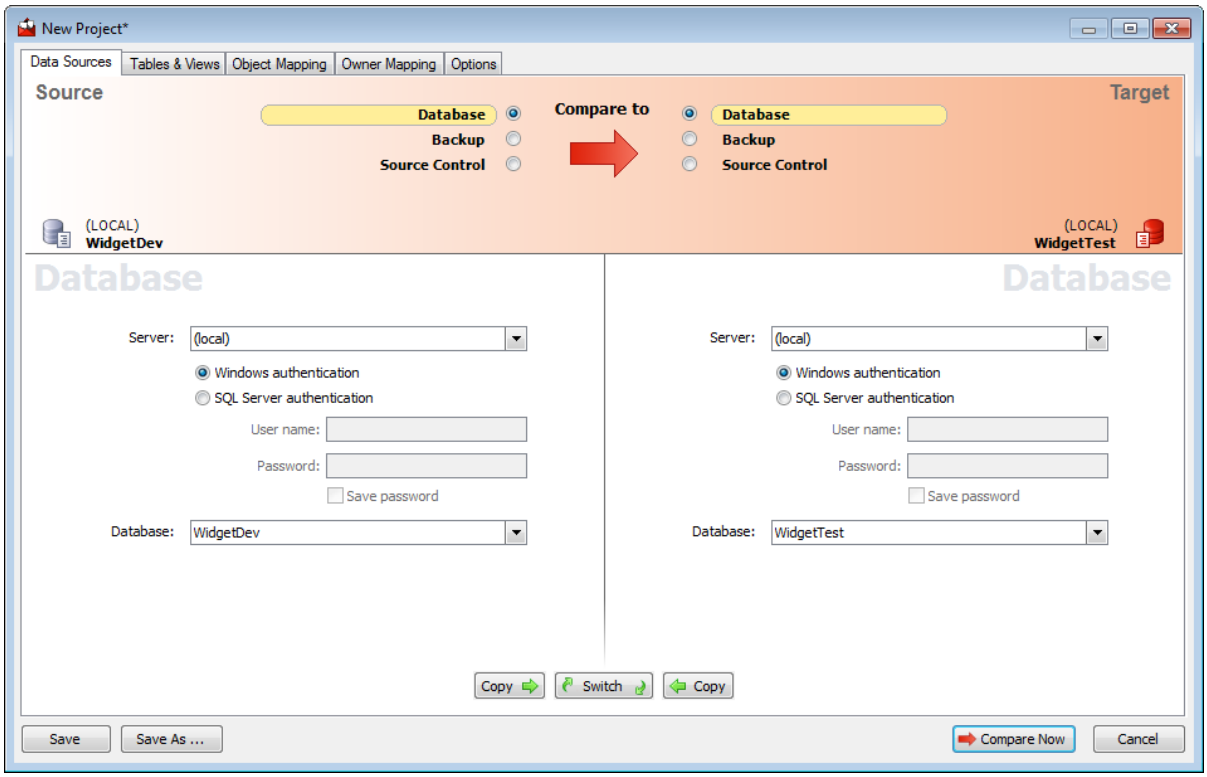
If you open and save a project in SQL Data Compare 8, it is converted to version 8 and cannot be opened in earlier versions.

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.

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 specify 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 **Switch**.

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.
- Source Control
  There are two options for the Source Control data source:
  - Direct from source control
    A revision of a database from a source control repository.
    For more information about getting your database into source control, see the SQL Source Control documentation.
  - Scripts folder
    A folder containing the SQL creation script files that collectively define the database structure. This is usually organized into a number of subfolders.
    For more information, see Working with scripts folders.

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

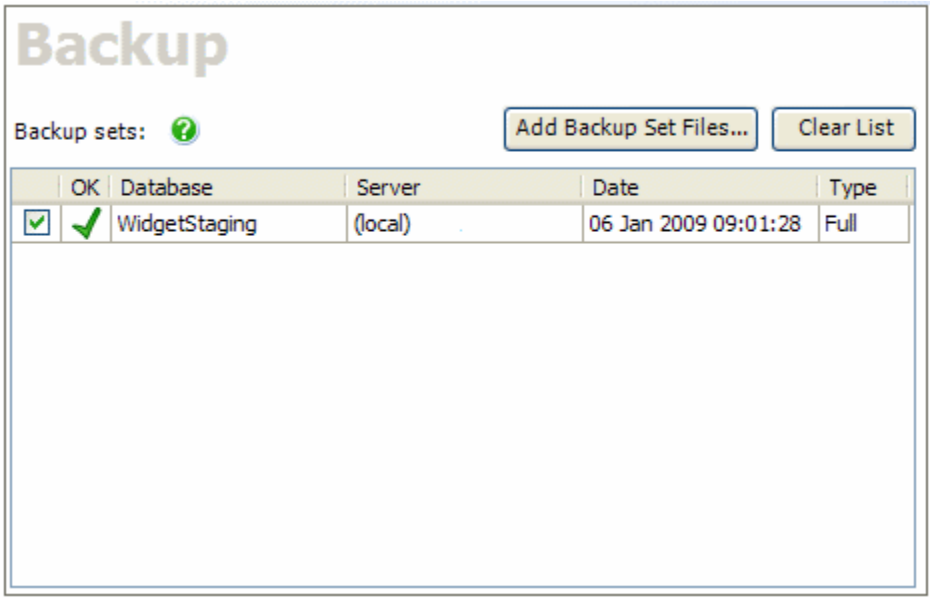
Selecting a database

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.

> 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

Selecting a scripts folder

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

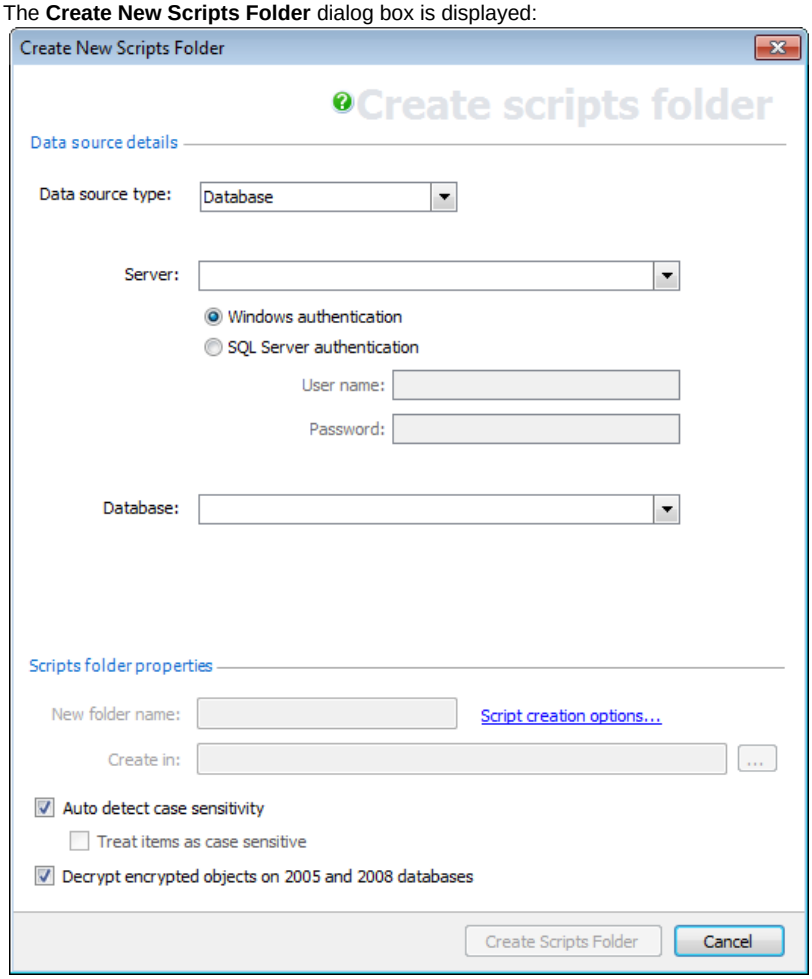
In the **Scripts Folder** box, select the folder, or click
...

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:



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

For more information, see Working with scripts folders

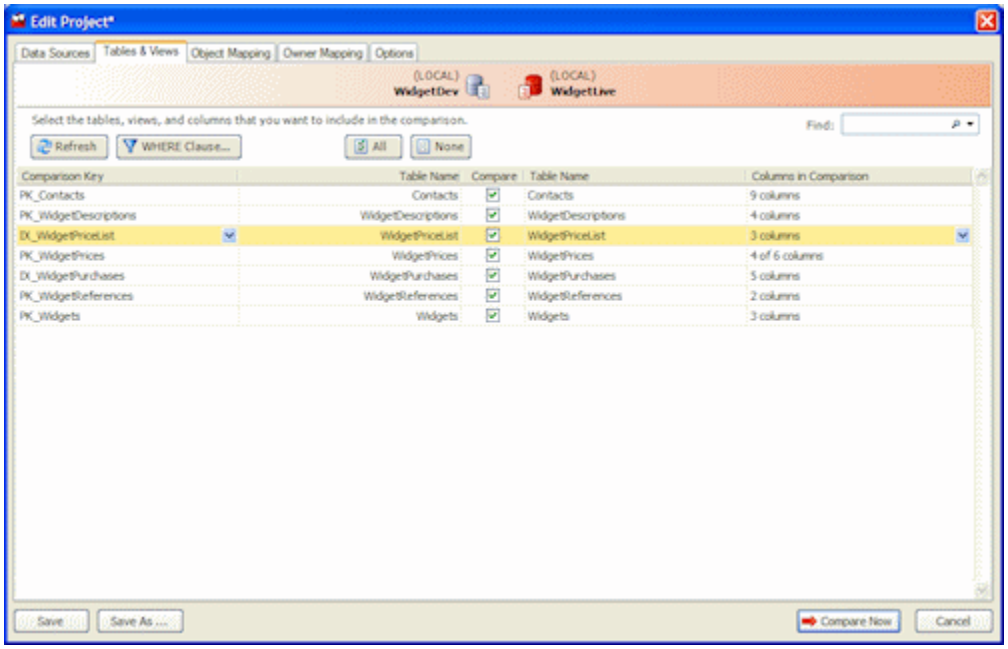
Selecting tables and views

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 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
- 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
- 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

Selecting the comparison key

To match rows in the two data sources, SQL Data Compare requires a comparison key for each table or view.

SQL Data Compare automatically selects a comparison key when:

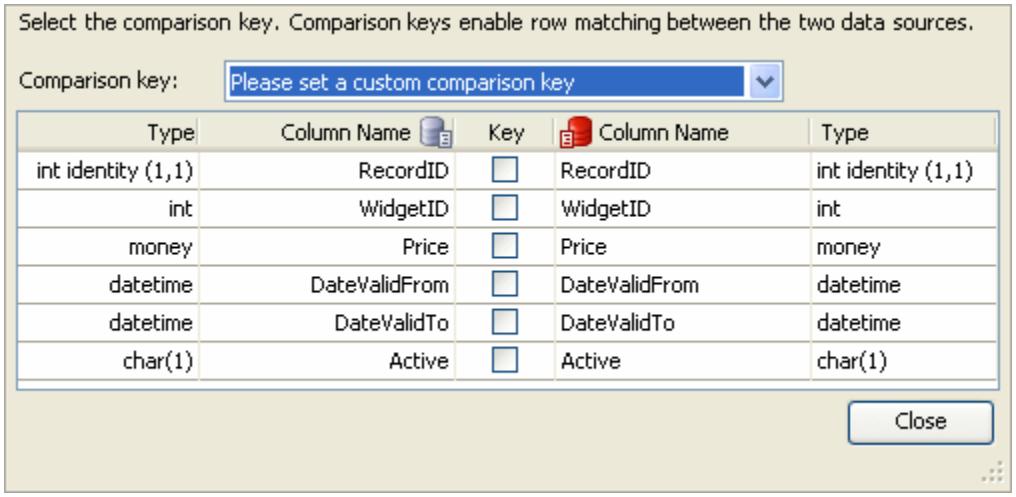
- tables contain a matching primary key, unique index, or unique constraint
- views contain a matching unique clustered index

For more information, see What is a comparison key?

If SQL Data Compare is unable to identify a suitable comparison key for a table or view, *Not Set* is shown in the **Comparison Key** box.

To set the comparison key for an object, click its **Comparison Key** box. A dialog box is displayed on which you can select the columns that will

comprise the key:



- A comparison key cannot include columns whose data type is image, ntext, nvarchar(max), sql_variant, text, varbinary(max), varchar(max), or xml.
- You cannot specify custom comparison keys if you are using a backup as a data source; however, you can select an alternative unique index or unique constraint.

For large databases, specifying a clustered index as the comparison key can result in improved performance.

Selecting the tables and views

Select the tables and views you want to compare by selecting or clearing the appropriate check boxes in the **Compare** column. To compare all tables and views, click

All; to clear all of the check boxes, click

None.

By default, the first time that you run a project all tables and views with identical or similar names are selected for comparison.

If the structure of the data sources you are comparing has changed while you are working on the project, click

Refresh to update the **Tables & Views** tab. For example, if a table has been added to the database, click

Refresh so that you can include the new table in the comparison.

If a table or view has been added to a database since you last ran the project, SQL Data Compare does not select the table or view by default.

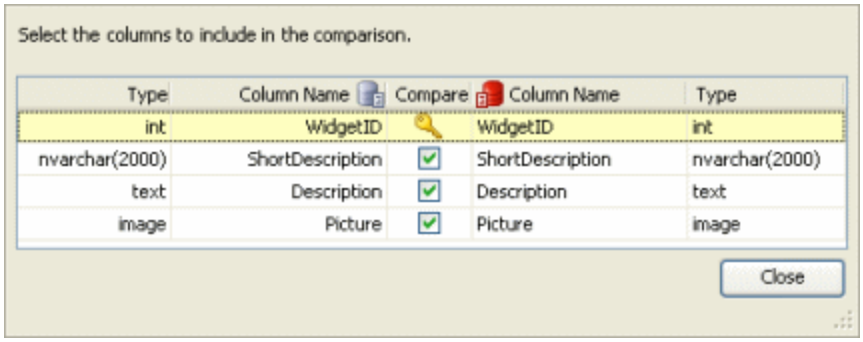
For more information, see Mapping objects

Selecting columns

SQL Data Compare displays the number of columns that will be compared for each table or view. By default, the first time that you run a project, all columns with identical or similar names are selected for comparison.

You can filter the comparison to consider only specific columns.

To select the columns to compare in a table, click on it in the **Columns in Comparison** column. A dialog box is displayed with check boxes to include or exclude each column:



You cannot exclude any columns that are used for the comparison key (indicated by

).

A warning symbol

is shown when columns you are comparing cannot be mapped. This occurs if those columns have:

- column names that do not match
- incompatible data types

> - you can compare a *timestamp* column with another *timestamp* column, but you can't deploy *timestamp* columns.
> - you can compare an *xml* column with another *xml* column, but you must make sure that your XML schemas are compatible.

For more information about the data types that SQL Data Compare can compare, see Which data types can be compared?

Mapping objects

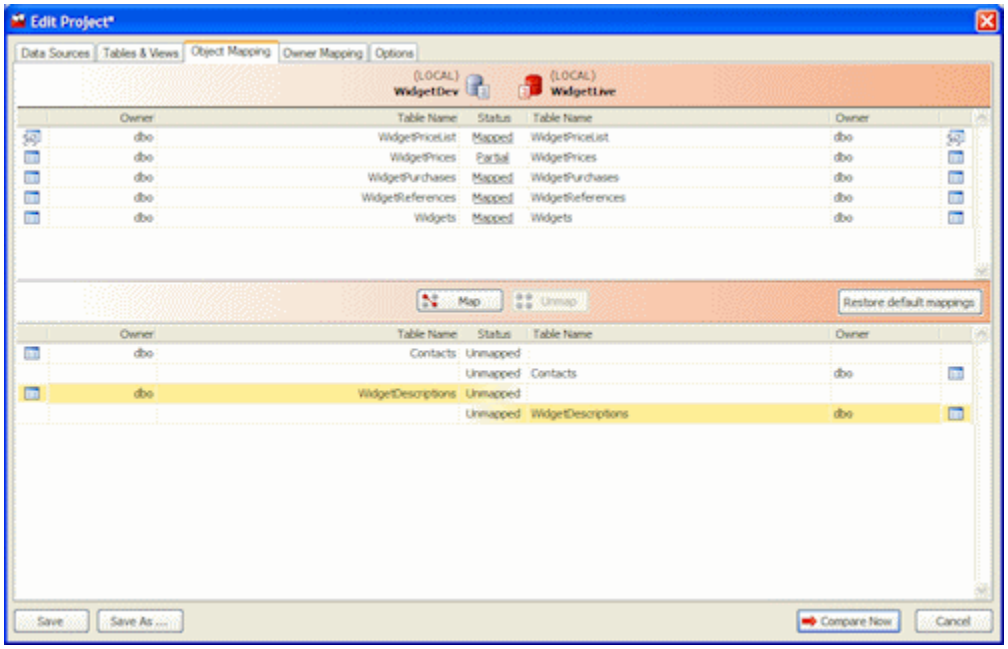
When you have selected your data sources, SQL Data Compare automatically maps tables and views with the same name and schema (owner).

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 page 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.

Mapping tables and views

To map tables and views:

1. On the **Object Mapping** tab, select an *Unmapped* table or view that you want to map from the *source* database.
2. Select the *Unmapped* table or view that you want to map from the *target* database.
3. Click

   Map.SQL Data Compare moves the tables or views to the upper pane.

Note that if the tables and views that you are mapping contain columns with incompatible data types, SQL Data Compare cannot compare those columns; the column mapping shows the incompatible data types as Uncomparable and a warning symbol

is shown in the **Tables & Views** tab.

For more information, see Which data types can be compared?

To unmap tables and views:

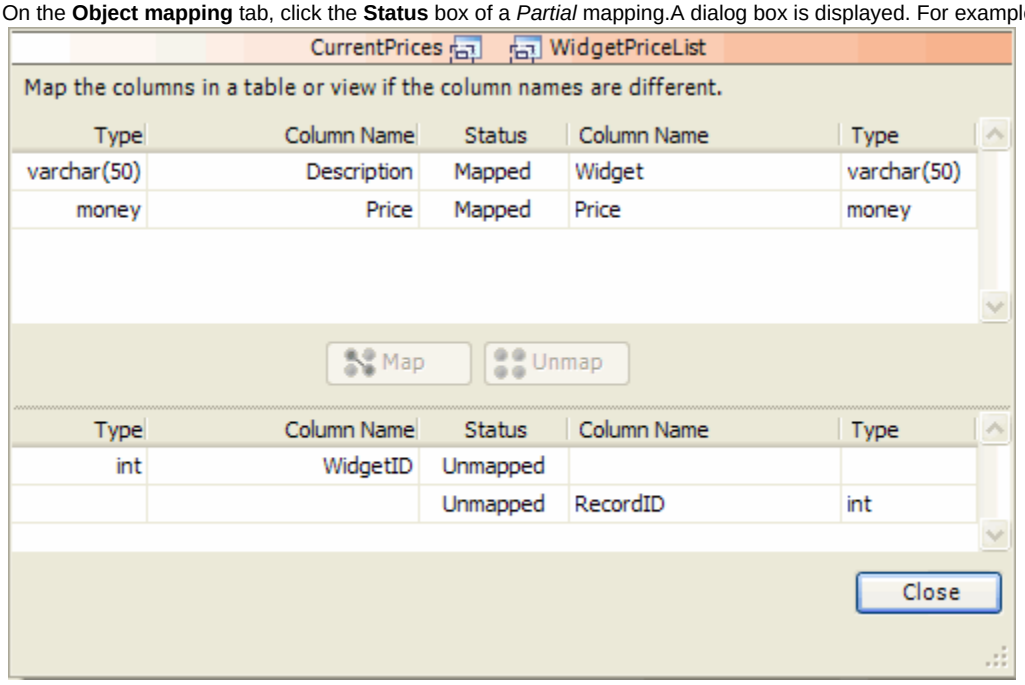
1. On the **Object mapping** tab, select a *Mapped* table or view, or a *Partial* mapping.
2. Click
   Unmap.

SQL Data Compare moves the tables or views to the lower pane.

Mapping specific columns

If you want to compare columns in a table or view and the column names are different, you can map the columns as required.

To map columns:

1. On the **Object mapping** tab, click the **Status** box of a *Partial* mapping.A dialog box is displayed. For example:
   

2. Select an *Unmapped* column that you want to map from the *source* database.
3. Select an *Unmapped* column that you want to map from the *target* database.
4. Click
   Map.
   SQL Data Compare moves the columns to the upper pane of the dialog box.
5. Click **Close**.

> If a column is shown as *Uncomparable*, SQL Data Compare cannot compare the data that is stored in the column.

For more information, see Which data types can be compared?

To unmap columns:

1. On the **Object mapping** tab, click the **Status** box of a *Mapped* table or view, or a *Partial* mapping.
   A dialog box is displayed for you to specify the column mappings, as shown above.
2. Select a *Mapped* column.
3. Click
   Unmap.
   SQL Data Compare moves the columns to the lower pane of the dialog box.
4. Click **Close**.

Mapping owners

When you set up a project and you select your data sources, SQL Data Compare automatically maps objects with the same name and the same owner (SQL Server 2000) or schema (SQL Server 2008 and SQL Server 2005) for you.

If you want to compare objects with the same name, but those objects have different owners or belong to different schemas, you can map the owners or schemas as required. For example, if you want to compare objects in a test database that are owned by *sales* with objects in a production database that are owned by *support*, you can map *sales* to *support*.

For example:

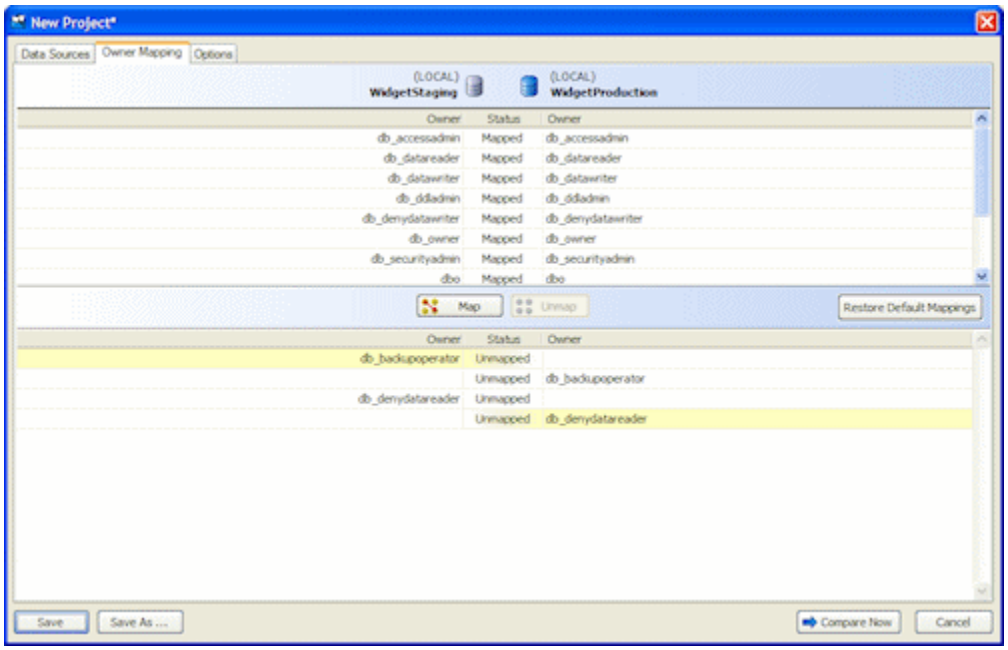
If you map the owners (or schemas) *sales* and *support*, this enables SQL Compare to compare the stored procedures [sales].[editCustomerDetails] and [support].[editCustomerDetails].

SQL Data Compare can then rename other objects that are referenced by the stored procedure. For example, if the [support].[editCustomerDetails] stored procedure references the table [support].[customerDetails], the stored procedure is updated to reference the table [sales].[customerDetails]. The same applies to functions, DML triggers, DDL triggers, views, defaults, and rules.

Tables in a schema that is not mapped are not compared. For example, if you unmap the schemas *Sales* and *Support*, and then map *Support* to *Sales*, the comparison will fail for all tables owned by *Sales* if you do not also map *Sales* to another schema.

Editing the owner mappings

To display the **Owner Mapping** tab, on the Project Configuration dialog box click **Owner Mapping**.



The Owner Mapping tab allows you to map different owners (schemas).

The upper pane displays a list of schemas, database roles, and database users that SQL Data Compare has automatically mapped for you; the lower pane displays a list of schemas, database roles, and database users that are not mapped.

- To undo a mapping, in the upper pane, select the mapping, and click
  Unmap
- To create a mapping, in the lower pane, select an owner from the *source*, then select a corresponding owner from the *target*, and click
  Map.
  For example, to map *sales* to *support*, you create a mapping for *sales* to *support* and then you create a mapping for *support* to *sales*.
  To reset the mappings to the defaults, click **Restore Default Mappings**.

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.

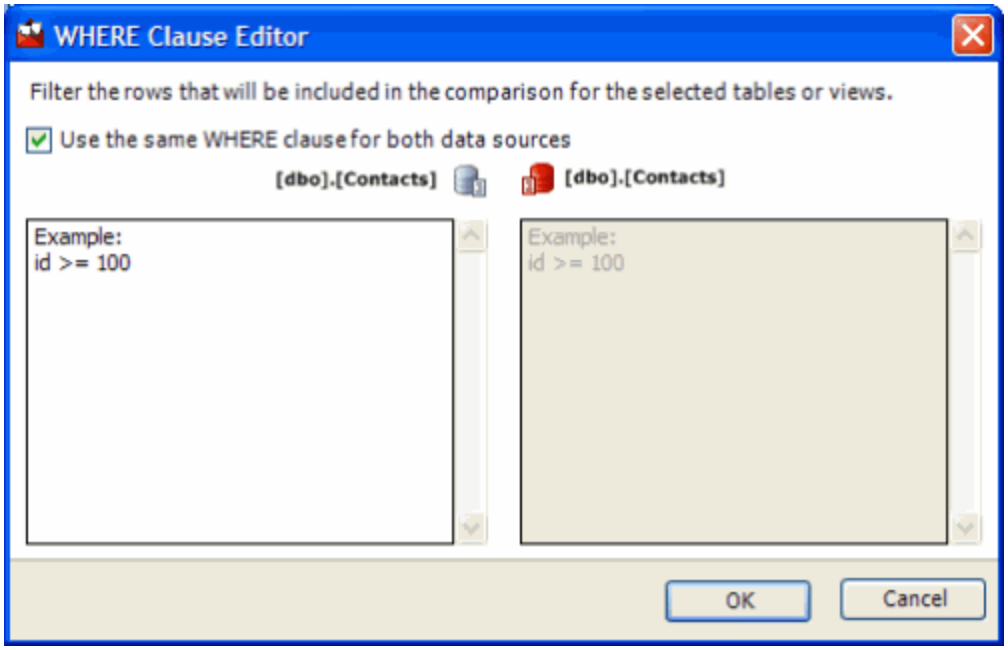
Note that:

- 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,

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.

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 deployment (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 deployment. Similarly, some options apply only to the deployment. 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 deployment script

Use case sensitive object definition

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 deployed 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 deploy a view if it includes an identity column.

Include timestamp columns

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

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 deployment, 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. On SQL Server 2000, db_owner permissions are required.

> 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.
> - 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** pan e.

Use maximum precision for floats

> This option is only available in SQL Data Compare 10.4.

Select this option is you want SQL Data Compare to compare floating point values to the maximum 17 digits of precision. By default, SQL Data Compare compares floats to 15 digits of precision.

> SQL Data Compare can't deploy a value if it is different only in the two additional digits of precision.

Disable foreign keys

Disables then re-enables foreign keys in the deployment 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 deployment SQL script.

> Note that:
>
> - if the primary key, index, or unique constraint is the comparison key, it cannot be dropped.
> - if you deploy 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 deployment SQL script and COMMIT TRANSACTION at the end of the deployment SQL script.

Transport CLR data types as binary

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

> Note that:
>
> - if this option is not selected SQL Data Compare uses the string representation.
> - if you deploy 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 deployment SQL script.

Disable DDL triggers

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

Reseed identity columns

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

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

Include comment header in the deployment script

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

Don't include comments in the deployment script

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

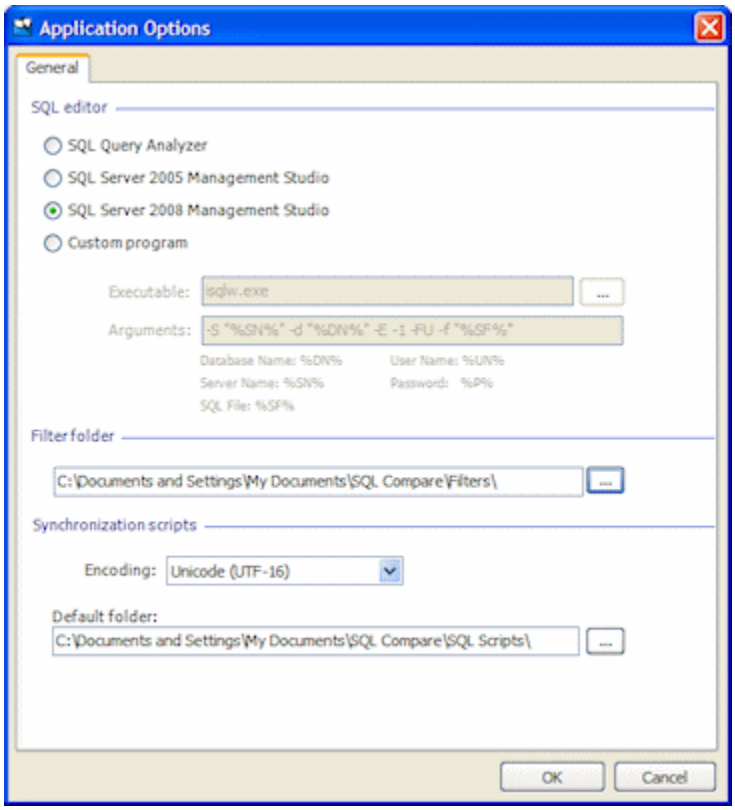
Force constraints to be re-enabled with CHECK

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

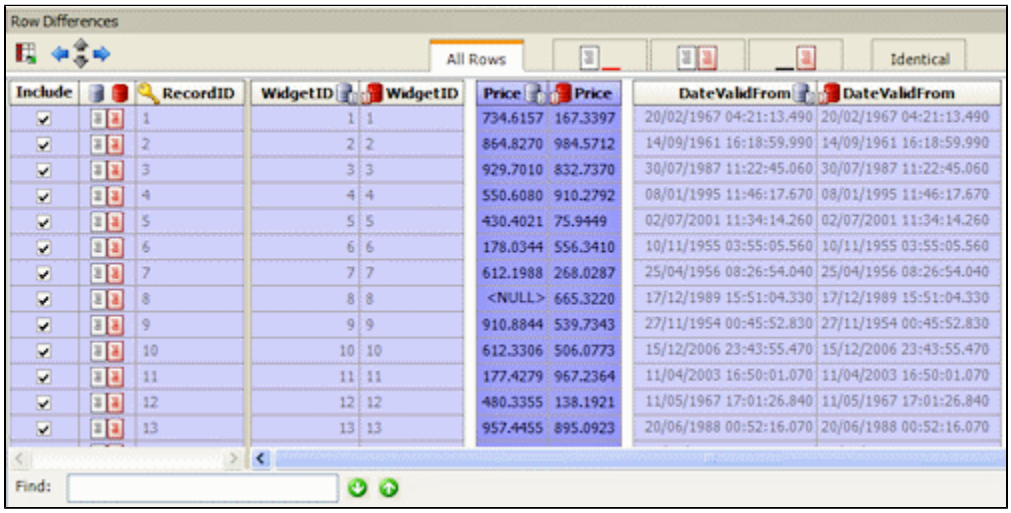
You can specify the amount of data included in each transaction using the **Maximum transaction size** box.

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

What is a comparison key?

When comparing data sources, SQL Data Compare looks for a matching primary key or other unique identifier in each data source to use as the *c omparison key*. This enables matching rows to be identified, and their differences to be compared.

For example, the databases *WidgetSales* and *WidgetDeploy* each contain the table [dbo].[WidgetPrices]:



Since rows can be inserted and deleted, we can't be certain that the third row in *WidgetSales* is the same as the third row in *WidgetDeploy*. Simply comparing rows in the order in which they appear in the table might result in a meaningless comparison. Similarly, rows can't be matched based on their *Price* values, as more than one widget can have the same price.

In [dbo].[WidgetPrices], *RecordID* is a primary key. No two widgets will have the same *RecordID*, and the rows are uniquely identified. Two matching *RecordID* values in *WidgetSales* and *WidgetDeploy* therefore represent the same piece of real world data.

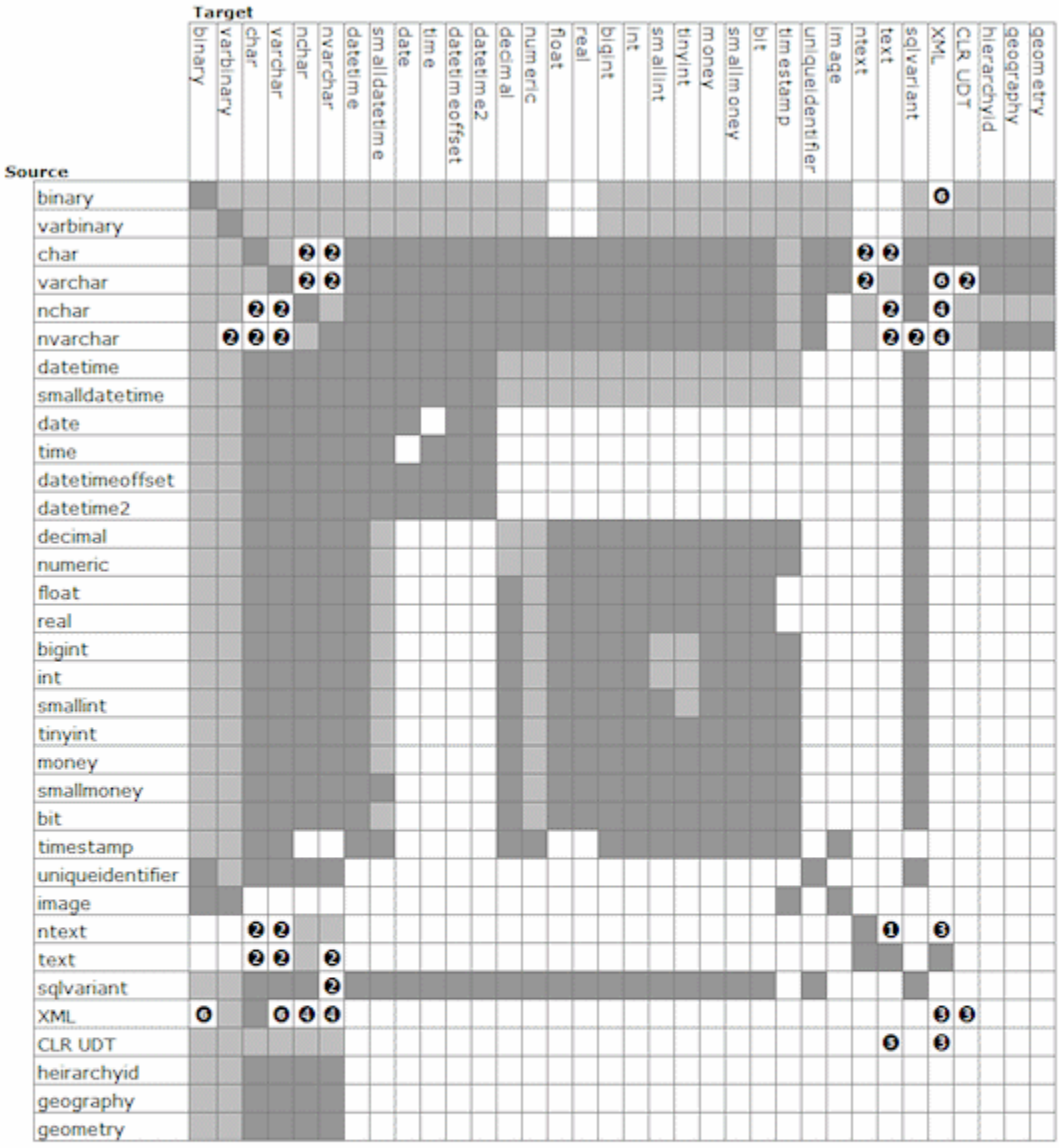
Where there's no matching primary key, or other unique identifier you must set an appropriate comparison key when selecting tables and views fo r comparison.

For example, if you know that two widgets with the same name are never added to the database on the same day, you can select the two columns *WidgetName* and *DateValidFrom* to form the comparison key. The rows in the tables can now be matched.

Which data types can be compared?

SQL Data Compare can compare similar data types, however, there is a risk of truncation in some cases.

The table below shows which data types can be compared:



	Cannot be compared
	Can be compared
	Possible truncation
❶	Unicode data loss
❷	Unicode data loss with possible truncation
❸	XML conversion
❹	XML conversion and possible truncation
❺	XML conversion and possible Unicode data loss
❻	XML conversion and possible Unicode data loss, and possible truncation

Reviewing the comparison results

These pages describe ways of dealing with the comparison results.

- Viewing the comparison results
- Tables and views that cannot be compared
- Viewing the row differences
- Viewing the data
- Exporting the comparison results
- Printing the comparison results

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

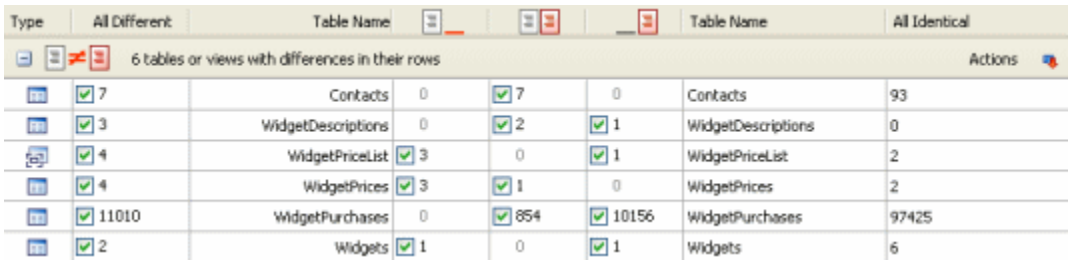
Refresh.

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.



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. deploy all of these rows to make the table or view identical.Use the check box to include or exclude all of the rows for deployment.
- **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 deployment.
- 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 deployment.
- 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 deployment.
- **Identical** displays the total number of rows in the table or view that are identical.
- **Actions**

displays a drop-down menu that lets you select or clear the various check boxes.The Actions menu isn't available if you are viewing the comparison results in a single list.For more information, see Setting up the deployment

Tables or views with identical rows only

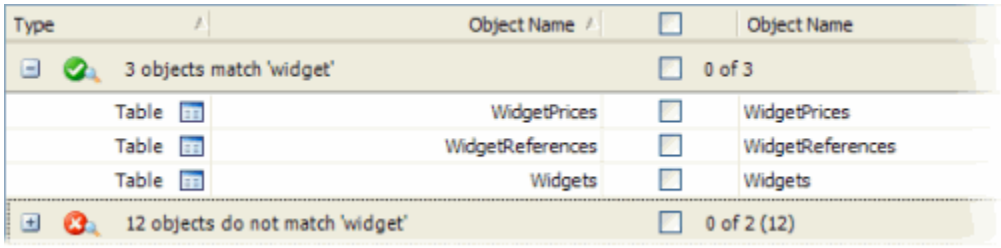
The comparison results are displayed as follows:

- **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:



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

Note that the search is not case-sensitive.

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

Tables and views that cannot be compared

Tables and views that cannot be compared are grouped in the upper (Results) pane. Objects cannot be compared if SQL Data Compare cannot automatically determine their object mappings, owner mappings, or a comparison key.

To compare an object that could not be compared initially, you can manually set the mappings and comparison keys.

Setting the mappings

If there are schema differences between the data sources, for example if a column has a different name in the source and target tables, you can map them on the Project Configuration dialog box.

For information on setting owner (schema) mappings, see Mapping owners

For information on mapping objects, or specific columns, see Mapping objects

Setting a comparison key

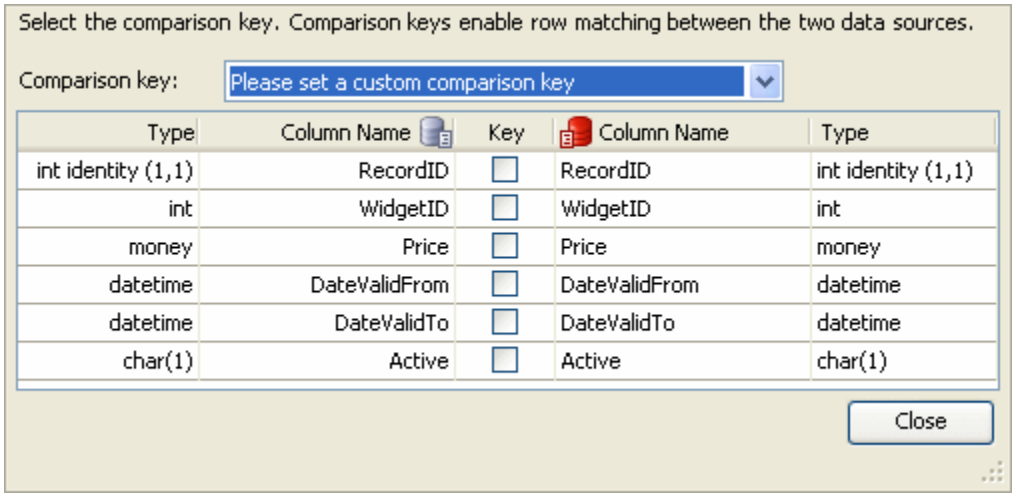
SQL Data Compare automatically selects a comparison key when:

- tables contain a matching primary key, unique index, or unique constraint
- views contain a matching unique clustered index

For more information, see What is a comparison key?

If SQL Data Compare is unable to identify a suitable comparison key for a table or view, *Not Set* is shown in the **Comparison Key** box.

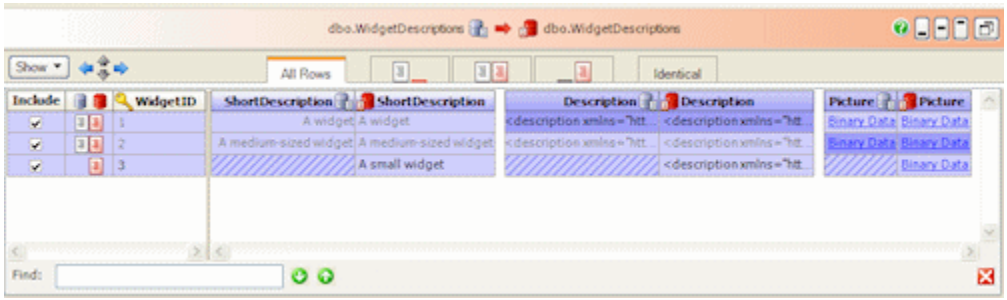
To set the comparison key for an object, click its **Comparison Key** box. A dialog box is displayed on which you can select the columns that will comprise the key:



- A comparison key cannot include columns whose data type is image, ntext, nvarchar(max), sql_variant, text, varbinary(max), varchar(max), or xml.
- You cannot specify custom comparison keys if you are using a backup as a data source; however, you can select an alternative unique index or unique constraint.
- For large databases, specifying a clustered index as the comparison key can result in improved performance.

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 deployment. 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.

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 deployment script

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

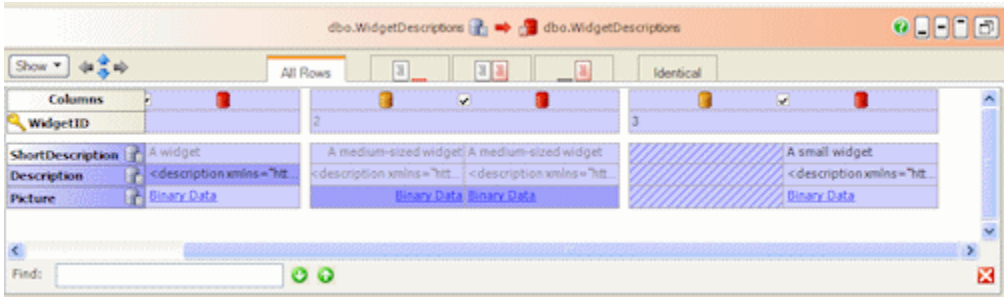
The **Single Object Deployment Script** dialog box is displayed.

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 deployment SQL that will make all the selected tables and views in the two data sources identical, use the deployment 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.

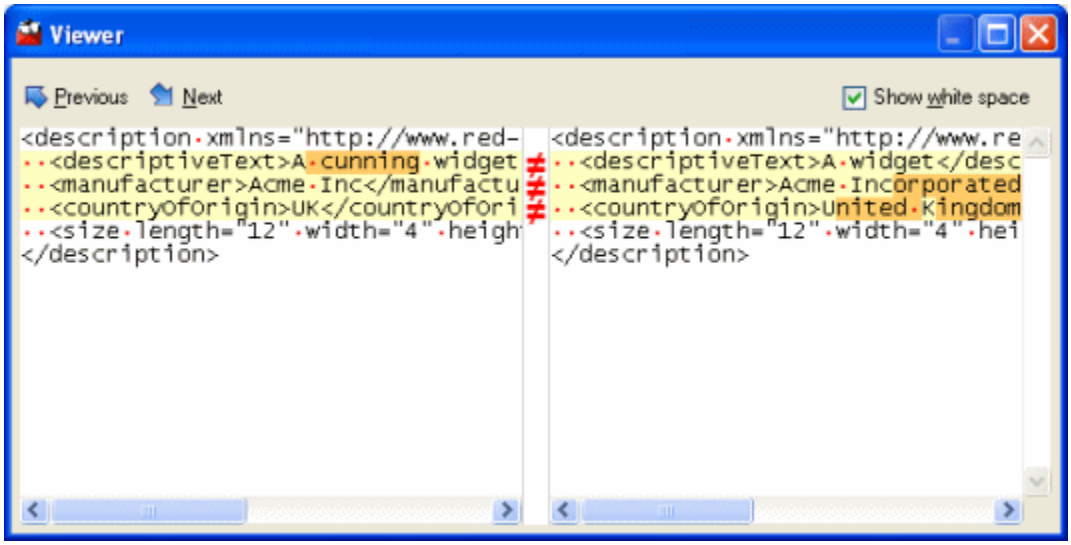
For more information, see Viewing the data.

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:

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



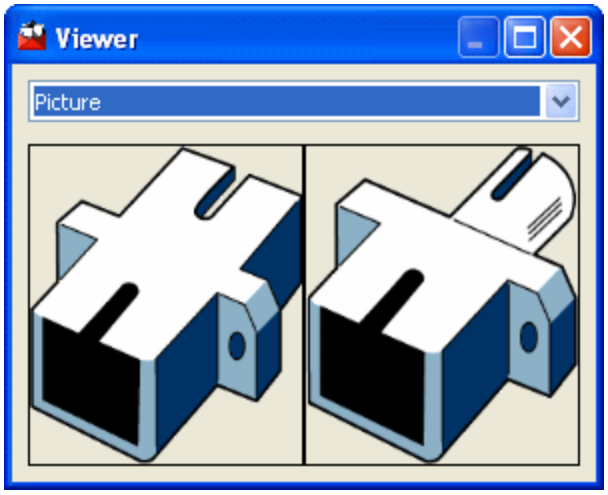
Use the

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**.

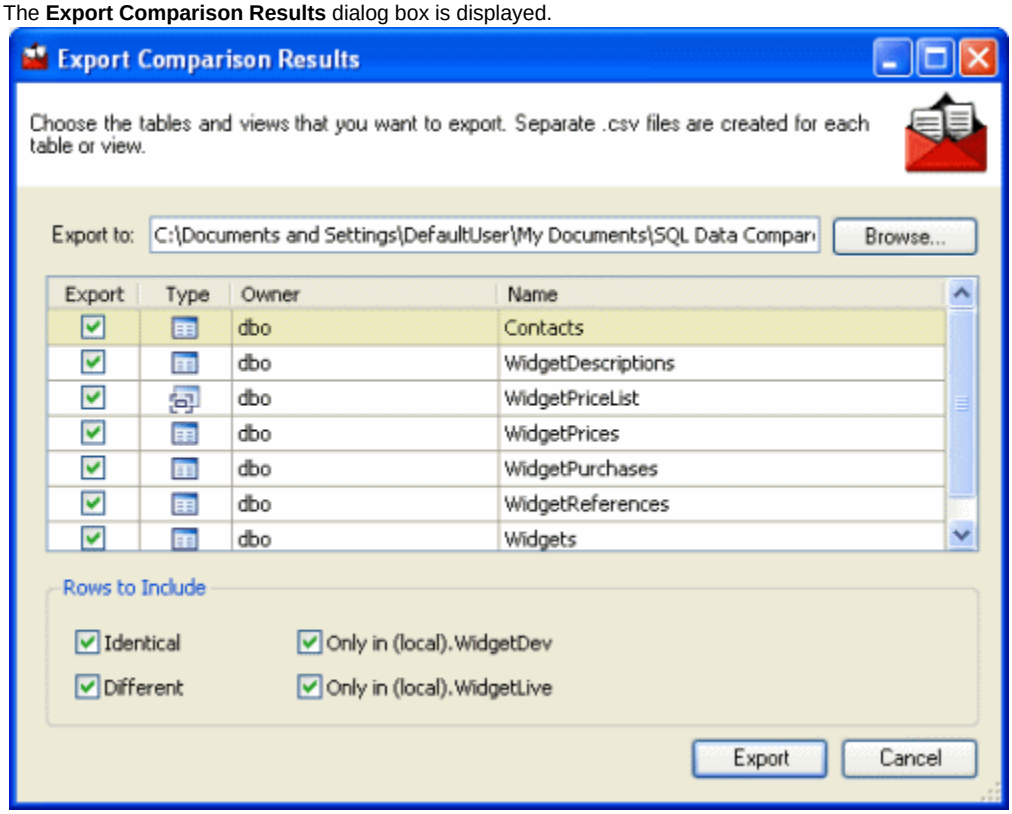
Exporting the comparison results

When you have run a comparison, you can export the comparison results. You may want to do this, for example, to produce a detailed report, or to keep a record of the comparison results.

Exporting to CSV files

To export the full comparison results, or a subset of the results, to a collection of comma-separated value (CSV) files:

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 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. As well as the data in the source and target tables, SQL Data Compare includes a **Type** column:

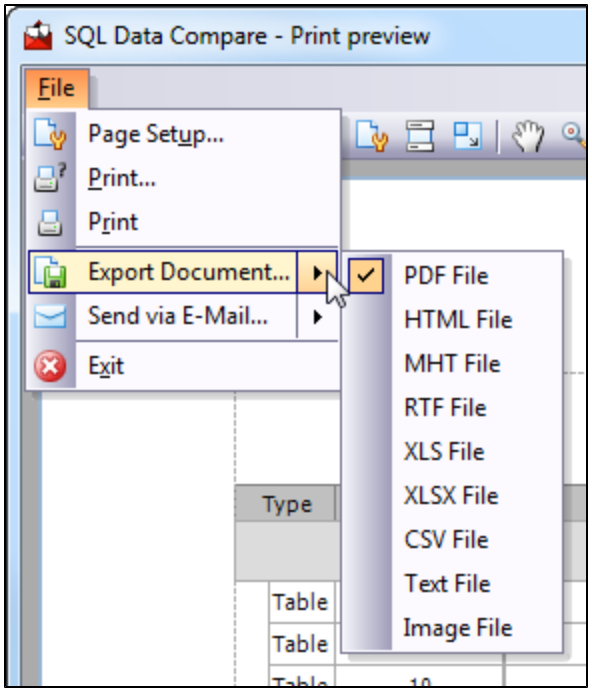
- **#a** means the row only exists in the source database
- **#b** means the row exists in both databases, but is different
- **#c** means the row only exists in the target database
- **#d** means the row exists in both databases and is identical

SQL Data Compare also creates a *Results Summary* file.

Exporting to other formats

To export a summary of the comparison results to other file formats, including PDF, text, and image files:

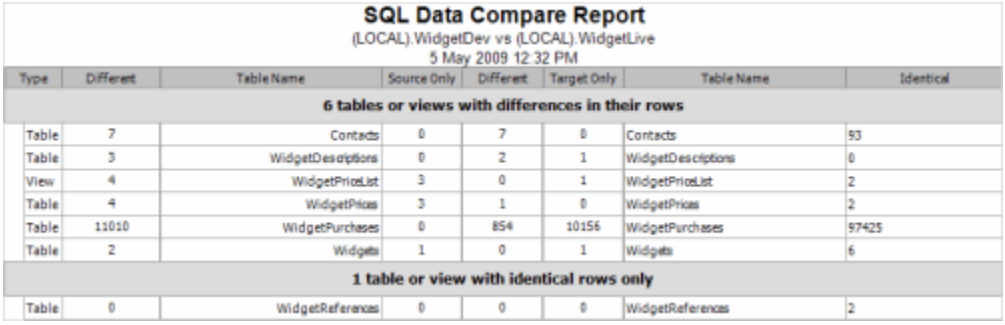
1. On the **File** menu, click **Print Preview**.
   The Print preview window opens.
2. In the Print preview window, on the **File** menu, hover over **Export Document** to see the list of export options:



3. Click on the file type you need.
   The Export Options dialog for that file type opens.
4. On the Export Options dialog, enter the details you need, and click **Ok**.
   The Save As dialog opens.
5. Choose the location to save the file to, and click **Save**.

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

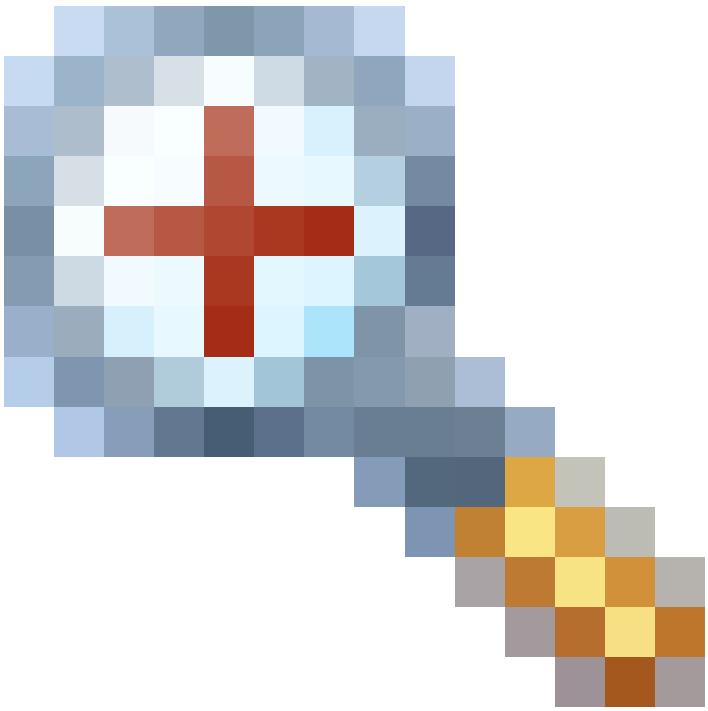
5 May 2009 12:32 PM

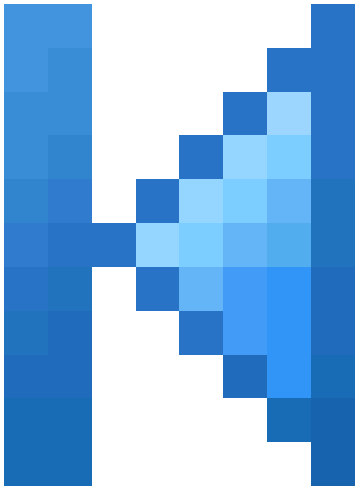
Type	Different	Table Name	Source Only	Different	Target Only	Table Name	Identical
6 tables or views with differences in their rows							
Table	7	Contacts	0	7	0	Contacts	93
Table	3	WidgetDescriptions	0	2	1	WidgetDescriptions	0
View	4	WidgetPriceList	3	0	1	WidgetPriceList	2
Table	4	WidgetPrices	3	1	0	WidgetPrices	2
Table	11010	WidgetPurchases	0	854	10156	WidgetPurchases	97425
Table	2	Widgets	1	0	1	Widgets	6
1 table or view with identical rows only							
Table	0	WidgetReferences	0	0	0	WidgetReferences	2

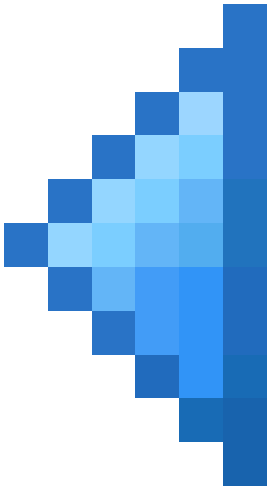
To see a preview of the summary, on the **File** menu, click **Print Preview**.

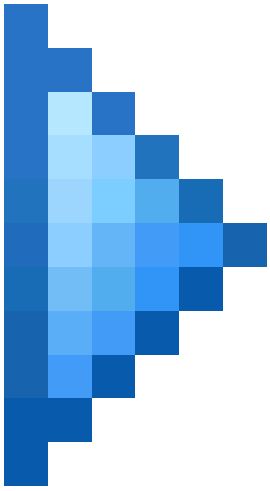
The **Preview** dialog box provides the following commands:

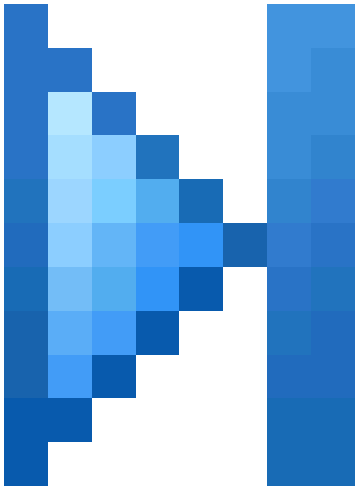
	prints the summary
	configures the page setup

	zoom in or out
	displays the first page, previous page, next page, or last page









Deploying the data sources

These pages explain how to deploy data sources using SQL Data Compare.

- Setting up the deployment
- Using the Deployment Wizard
- Deploying views
- Backing up before deployment
- Warnings
- Data that wasn't deployed

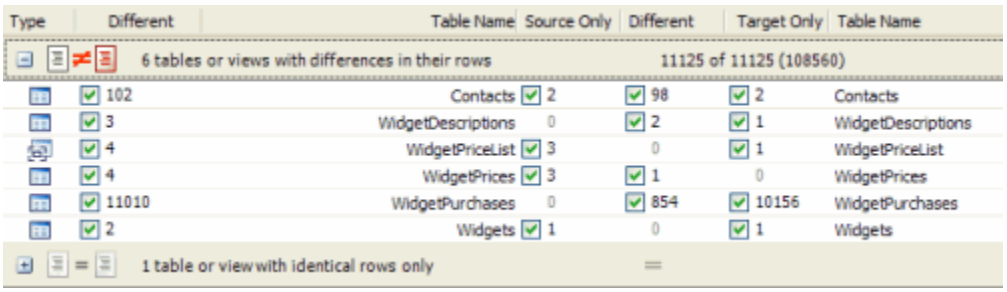
Setting up the deployment

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

Selecting the rows to deploy

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

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



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

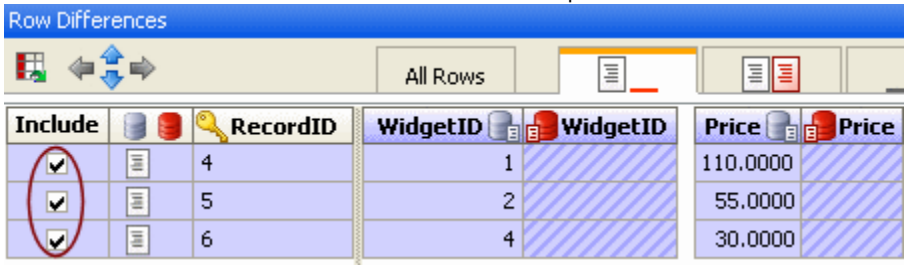
  (only in target) column.

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) pane to select individual rows to deploy:

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.



Selecting the rows to deploy using keyboard shortcuts

To select a row for deployment, 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 Deployment Wizard

When you have selected the objects that you want to include in the deployment, use the Deployment Wizard to create the SQL script that will deploy the databases.

To start the Deployment Wizard, click **Deployment Wizard**.

The wizard has three steps:

1. Choose deployment method
   Create and save a script, perform the deployment from SQL Data Compare, or update a scripts folder.
2. Configure Backup (optional step)
   If you do not choose to back up on the first page, this step is not part of the wizard.
3. Review
   View the deployment script, review a summary of the deployment actions, and see any warnings.

If SQL Data Compare is unable to deploy the data sources, an error message is displayed, and where possible all changes are rolled back. However, if you have disabled the project option **Use transactions in SQL scripts**, the changes are not rolled back.

Deploying backups

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 database can be deployed with the backup.

For more information, see Working with backups

Deploying scripts folders

When a scripts folder is the target, you can either:

- Create a deployment script to update the database from which the scripts folder was created, or
- Update files in the scripts folder directly

> When an object is dropped during deployment, its script file is not deleted.

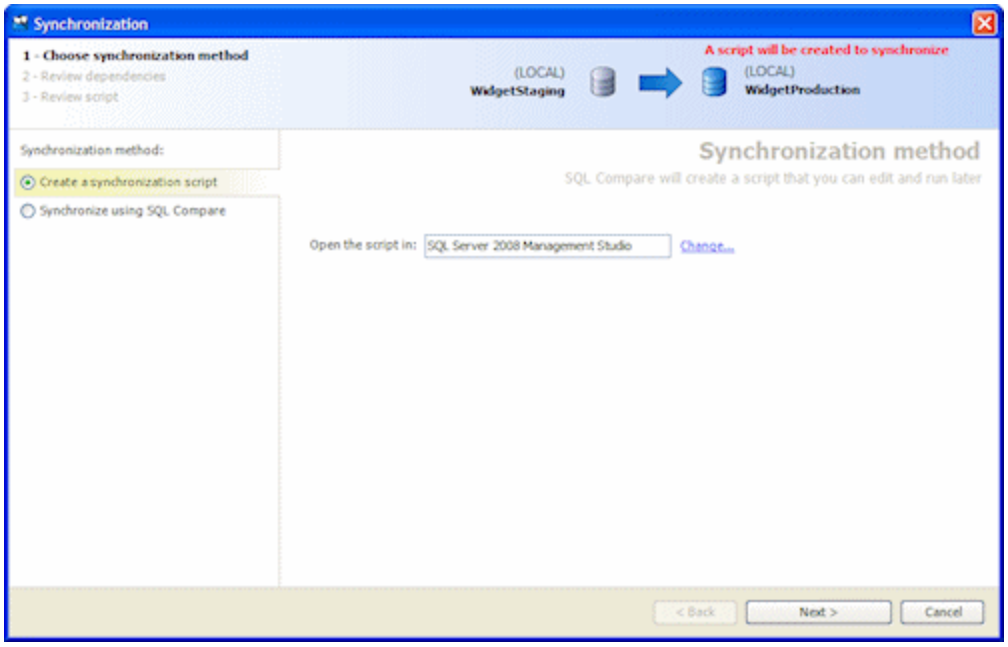
If a scripts folder is the target, and any of the script files that will be modified are designated as read-only, a warning is displayed. If you click **Yes** t o continue, these files will be made writable so that they can be modified. This may occur, for example, when you are working with a source control system that sets files to read-only status in some situations.

For more information, see Working with scripts folders

1. Choose deployment method

On the first page of the Deployment Wizard, you can choose to create and save a deployment script, perform the deployment using SQL Data Compare, or update a scripts folder:

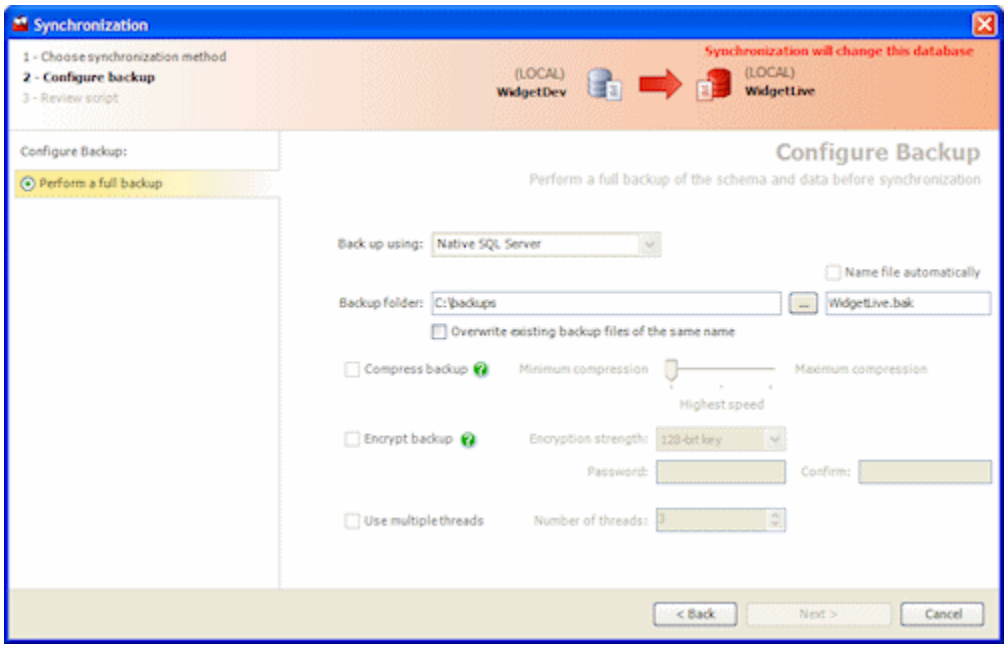
Create a deployment script



If you choose to create a deployment script, on the **Review** page you can save a copy of the script or open it in your SQL editor.

To change the application you use to open the script, click **Change**. The Application Options dialog box is displayed, and you can specify the default application used to edit SQL Scripts.

Deploy using SQL Data Compare



If you choose to deploy using SQL Data Compare, on the **Review** page you can save a copy of the script and exit the wizard, or click **Deploy Now** to perform the deployment.

If the target is a scripts folder, this option is instead replaced by **Update the scripts folder**, and the SQL script files in the target are modified when you deploy.

If you select **Back up target before deployment**, a step is added to the Deployment Wizard allowing you to specify details of the backup.

Note that if the target is a backup, the option to deploy using SQL Data Compare is not available. Instead, deploying creates a script to modify the data source form which the backup was created.

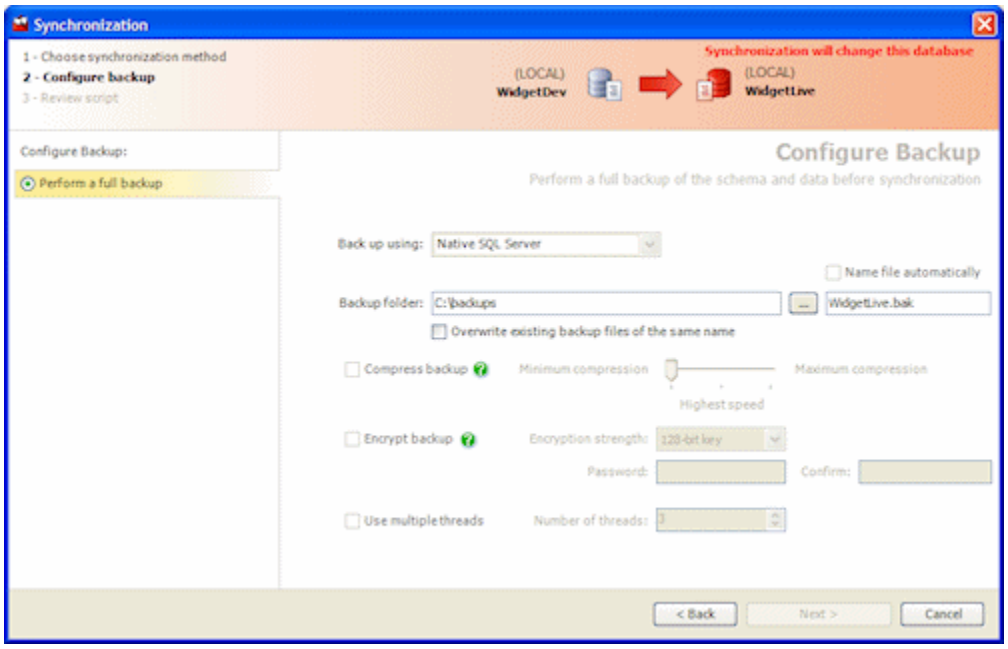
2. Configure backup (optional step)

If you selected the option to back up before deployment, the **Configure backup** page is added to the Deployment Wizard.

If the target is a database, this page allows you to perform a backup before deployment.

If the target is a scripts folder, this option is not available.

Perform a full backup

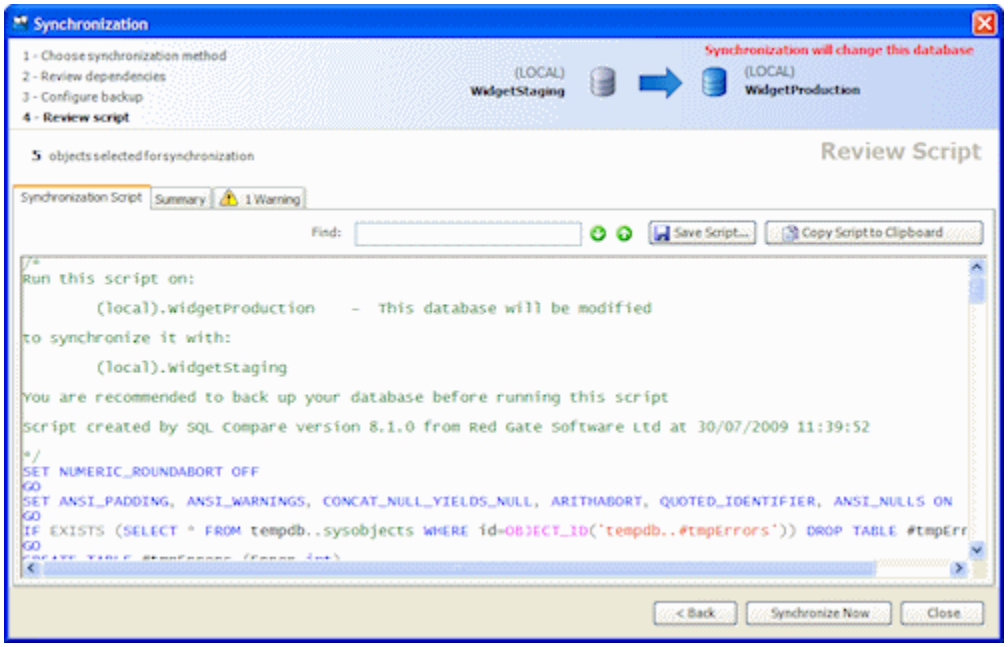


When the target is a database, you can perform a full backup. In the **Back up using** box, select:

- *Native SQL Server* to create a backup using the native SQL Server BACKUP command
- *Red Gate SQL Backup* to create a backup using SQL Backup version 4 or laterNote that to use Red Gate SQL Backup, you must have the SQL Backup server components installed on the SQL Server instance of the target database.

For more information, see Backing up before deployment.

3. Review



There are three tabs on the **Review** page:

- **Deployment script** shows the script to deploy the data sources. You can search the script, save it, or copy it to the clipboard.
- **Summary shows** a synopsis of the actions in the deployment script. You can view the summary grouped by the objects affected, by the type of modification, or by the order in which the script modifies the target.
- **Warnings shows** a list of any warnings about unexpected behavior that may occur when you deploy the databases.
  For more information, see: Deployment warnings.

If you are updating or creating a scripts folder, the **Files** tab lists the object creation and data script files modified or created during deployment.

When you have reviewed the script, click **Deploy Now** to perform the deployment.

Deploying views

SQL Data Compare can deploy 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 deploy views that reference more than one table, we recommend the following procedure:

1. Use SQL Data Compare to compare the views.
   Make sure the **Include views** project option is selected.
2. Note which rows are different.
3. Use SQL Data Compare to compare and deploy the tables that contain the differing rows.

Backing up before deployment

The **Configure Backup** page of the Deployment Wizard allows you to perform a full backup using either Red Gate SQL Backup, or SQL Server native backups.

Backup name and 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.

Backup compression (Red Gate SQL Backup only)

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.

> If SQL Backup Lite is installed on the SQL Server, you can choose only compression level 1.

Backup encryption (Red Gate SQL Backup only)

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.

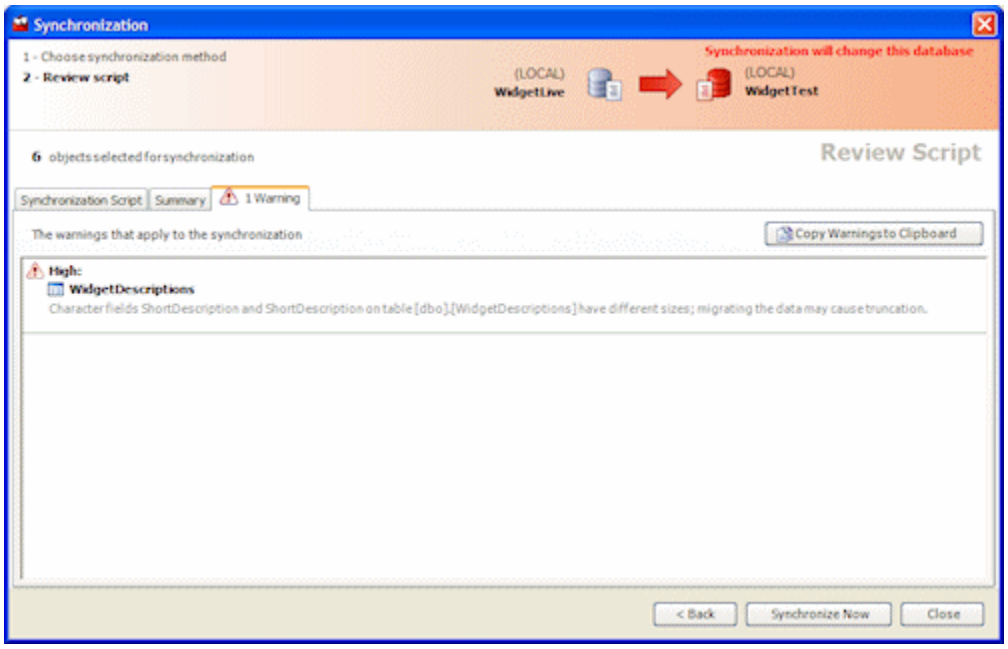
Using multiple threads (Red Gate SQL Backup only)

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.

Warnings

On the **Summary** page of the Deployment Wizard, you can click **Warnings** to view information about unexpected behavior that may occur when you deploy 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 deployment 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 deployment 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 deployed.

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 deploy 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.

Data that wasn't deployed

When you deploy the data sources, some rows may not be deployed 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 deployment is run, which will cause unpredictable results. To avoid this, select the **Disable DML triggers** project option before you generate the deployment 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 can't deploy data in *timestamp* columns.
- table structures are not identical

  If the structure of the tables you're deploying is not identical, SQL Data Compare may produce unpredictable results.

CLR data that wasn't deployed

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 deployed, SQL Data Compare deploys 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 deployment, 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 deploy the binary representations.

Primary keys, indexes, or unique constraints that weren't dropped for deployment

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 deployment.

Working with other data sources

These pages explain how to work with data sources other than SQL Server databases.

- Working with backups
- Working with scripts folders

Working with backups

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.

- 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 deploying backups

You can:

- compare a backup with another data source
  See Setting data sources.
- create a deployment 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 database can be deployed 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.

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
- computed columns
  However, you can compare persisted computed columns.
- string representations of CLR types
  Only the binary representation is supported.
- custom comparison keys
  However, you can select an alternative unique index, or unique constraint.
- row filtering using the WHERE Clause Editor

Working with scripts folders

A scripts folder is a set of object creation scripts representing a database's schema and (optionally) data.

In SQL Data Compare you can create a scripts folder from an existing data source when you set up the comparison. An object creation script file is created for each object, and different object types are stored within subfolders you can specify.

When you deploy to a scripts folder, static data files are created.

You can use scripts folders:

- for version control of databases
  For example, you may want to store the script files in a source control system, so that you can track the modified or new objects.
- to compare databases on unconnected SQL Servers

You can create and compare scripts folders only if you are using SQL Data Compare Professional Edition.

For an example using scripts folders, see Deploying a database from source control

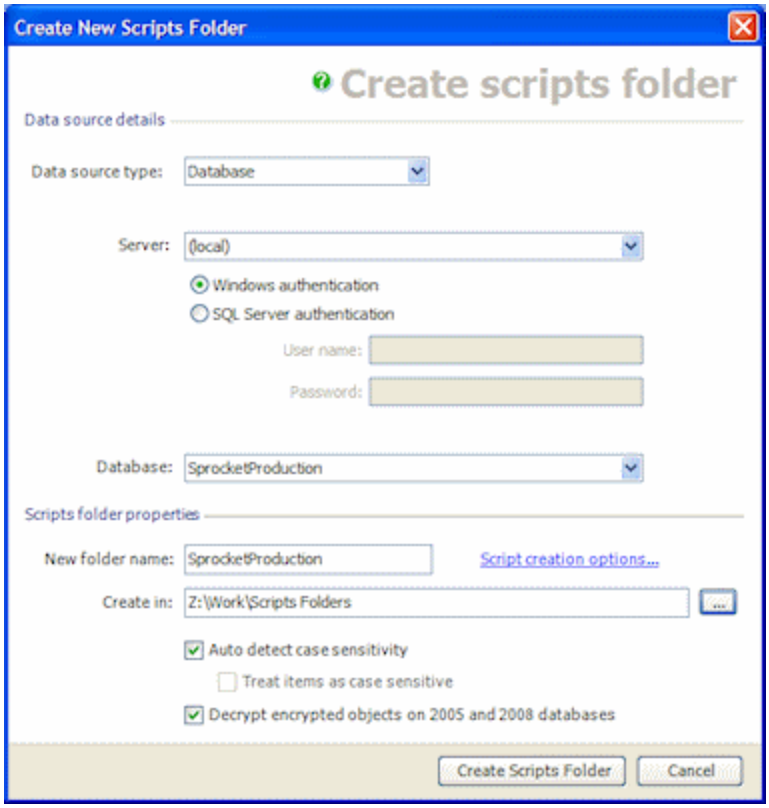
What's stored in a scripts folder?

- Object creation scripts
  A *.sql* file containing creation scripts is saved for each object in the schema when the scripts folder is created.
- Static data scripts (optional)
  A *.sql* file containing INSERT scripts is saved for each table when you deploy to the scripts folder.
- Database and schema metadata
  A *.sdcs* file is saved for each table when you deploy to the scripts folder. These files contain index information that enables SQL Data Compare to compare static data.
  The file *RedGateDatabaseInfo.xml* contains information about the objects in the schema, the SQL Server version, and the collation of the database from which the scripts folder was created.
  You are recommended not to delete or modify these files.

Creating a scripts folder

To create a scripts folder:

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:



5. Under **Data source details**, specify the details of the source for the scripts folder.
   You can create a scripts folder from a database, backup, snapshot, or another scripts folder.
6. Under **Scripts folder properties**, specify a name and location for the scripts folder.

SQL Data Compare automatically detects the case sensitivity of the data source. If you want to override this, clear the **Auto detect case sensitivity** check box, and select the **Treat items as case sensitive** check box if required.

By default, SQL Data Compare decrypts text objects in SQL Server 2008 and SQL Server 2005 databases that were created using the WITH ENCRYPTION option. When comparing large databases, selecting this option can result in slower performance.

You can customize the folder structure and character encoding of the scripts folder. To do this, click **Scripts creation options**. The **Edit Scripts Creation Options** dialog box is displayed, allowing you to specify the directories in which your database objects and static data are saved.

- When the source is a snapshot, the case sensitivity and decryption options are not available.
- When the source is a scripts folder, the decryption option is not available.

Comparing and deploying scripts folders

You can:

- compare a scripts folder with another data source
  See Setting data sources.
- deploy a scripts folder

When a scripts folder is the target, you can either:

- Create a deployment script to update the database from which the scripts folder was created, or
- Update files in the scripts folder directly
  Note that when an object is dropped during deployment, its script file is not deleted.

If a scripts folder is the target, and any of the script files that will be modified are designated as read-only, a warning is displayed. If you click **Yes** t o continue, these files will be made writable so that they can be modified. This may occur, for example, when you are working with a source control system that sets files to read-only status in some situations.

For more information, see Using the Deployment Wizard

More information about scripts folders

This section provides further information about using scripts folders as a data source in SQL Data Compare.

- White space
  SQL Server does not always process white space and comments correctly at the beginning and end of the object definition for some objects such as views, stored procedures, and rules. You are therefore recommended to select both the **Ignore spaces in object names** and **Trim trailing spaces** options when you use a scripts folder as a data source.
  For more information, see Setting project options
- Comments
  When you select a scripts folder as the target data source, SQL Data Compare preserves comments in object types such as views and stored procedures. However, this is not possible for non-textual object types such as tables. Comments that are part of a table definition will be lost when the table is modified and the object creation script updated.

Using the command line

These pages explain how to use SQL Data Compare through the command line.

- Getting started with the SQL Data Compare command line
- Changes to the command line in SQL Data Compare 8
- Command line basics
- Integrating the command line with applications
- Examples using the command line
- Command line syntax

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.

Changes to the command line in SQL Data Compare 8

These changes coincide with the release of SQL Compare 8, and are intended to make command line syntax more consistent between SQL Compare and SQL Data Compare.

In SQL Data Compare 8, there are changes to the names and functions of some command line switches and options, as well as their aliases.

- The command line syntax of previous versions of SQL Data Compare is considered deprecated, but continues to be supported. For example, in SQL Data Compare 7, the alias for */BackupSet1* was */bs1*. In SQL Data Compare 8, the alias is now */bks1*. You can continue to use */bs1* in SQL Data Compare 8, but a message is displayed informing you of the new alias.

Deprecated command line syntax will cease to be supported at a future release.

This page addresses the following:

- Which switches have changed?
- Which options have changed?
- Changes to case sensitivity behavior
- Compatibility with earlier versions

Which switches have changed?

/AllowIdenticalDatabases

This switch is deprecated. Instead use */Include:Identical*

/BackupSet1 and /BackupSet2

The aliases for these switches are now */bks1* and */bks2.*

The functionality of these switches has not changed.

/CaseSensitive

This switch is deprecated. Instead use */Options:CaseSensitiveObjectDefinition*

/Columns

This switch is deprecated. Instead use */IncludeColumns* and */ExcludeColumns*

/ExportIdenticalTables

This switch is deprecated. Instead use */Include:Identical*

/IgnoreParserErrors

If SQL Data Compare encounters any high level errors when parsing a scripts folder, it will exit with an error code of 62.

Use */ignoreParserErrors* to force SQL Data Compare to continue without exiting.

/Include and /Exclude

Specify which tables and views are included in the comparison.

/Include and */Exclude* have the following arguments:

- *Additional*: only those objects that are not present in the source (eg */db1*)
- *Missing:* only those objects that are not present in the target (eg */db2*)
- *Different:* only those objects that are present in both data sources, but are different.
- *Identical:* identical objects in the command line output and any generated reports.

- *User specified*: objects you specify with a regular expression (eg */Include:Table:WidgetPurchases*)

Note that if there is a conflict, */Exclude* takes precedence over */Include*

/IncludeColumns and /ExcludeColumns

Specifies which columns in a table are included in or excluded from the comparison.

Note that */ExcludeColumns* takes precedence over */IncludeColumns*

/IncludeAdditional

This switch is deprecated. Instead use */Include:Additional*

/ScriptFile

Alias: */sf*

Generates a SQL script to migrate the changes which can be executed at a later time. 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"
```

/Scriptfile can be used when the target (*/db2*, */scr2*, */sn2*) is a database, a snapshot, or a scripts folder.

If the target is a snapshot or a scripts folder, the generated script modifies a database with the schema represented by that snapshot or scripts folder.

Which options have changed?

DDLTriggerDisable

This option has been renamed: *DisableAndReenableDDLTriggers*, with the alias *drd*

DisableDMLTriggers

This option has been renamed: *DisableAndReenableDMLTriggers*, with the alias *t*

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. Case insensitive comparison is now the default behavior.

Changes to case sensitivity behavior

In earlier versions of SQL Data Compare the case sensitivity behavior of the command line interface was inconsistent.

/CaseSensitive affected the matching of object names for the */Include*, */Exclude*, */Columns*, and */ComparisonKeys* switches. The option *IgnoreCase* was on by default, and affected the case sensitivity of object and owner mapping.

These behaviors are now deprecated, and all switches and mappings share the same case sensitivity behavior.

In SQL Data Compare 8:

- the default setting is case insensitive
- case sensitivity is set using */Options:CaseSensitiveObjectDefinition*

Compatibility with earlier versions

Although the command line syntax of previous versions of SQL Data Compare is considered deprecated, it continues to be supported.

If you use deprecated syntax, a warning is displayed, but SQL Data Compare functions normally.

Note that for case sensitivity, the new syntax takes precedence. For example, if you set both */Options:CaseSensitiveObjectDefinition* and */Options :IgnoreCase*, the command line behavior is case sensitive.

You are recommended to use the new syntax. Deprecated command line syntax will cease to be supported at a future release.

Command line basics

This page provides information on the following aspects of the SQL Data Compare command line:

- Getting help
- Entering a command
- Aliases
- The /Options switch
- Verbose and quiet switches
- Redirecting command line output

For details and examples of all of the switches that are available for the SQL Data Compare command line, see Switches used in the command line

Getting help

To display a brief description of SQL Data Compare, and basic help on all the command line switches, enter:

```
sqldatacompare /Help
```

For more detailed help enter:

```
sqldatacompare /Help /Verbose
```

This displays a detailed description of each switch and the values it can accept (where applicable), and all exit codes.

To output the help to an HTML file, specify a file name and path:

```
sqldatacompare /Help /Verbose /Html > C:\SQLDataCompare8_Help.htm
```

Entering a command

When you enter a command, 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
```

Values that include spaces must be delimited by double quotation marks ("). For example:

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

> 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:IncludeDependencies,ForceColumnOrder
```

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 objects from the comparison and deployment:

```
/Include:Table:Product
```

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

> If you use the */include* switch to compare only tables (or any object type) matching a word or regular expression, all other objects not of that type will still be included in the comparison. In the above example, only tables that contain the word *Product* will be included, but all views, stored procedures, users and so on will still be included, unless further arguments to limit these object types are also specified.

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.

Switches and aliases are not case sensitive.

The /Options switch

You can use the */Options* switch to change your options. For example, comparisons are not case-sensitive by default; to specify case sensitive comparisons you would use the */options* switch:

```
/Options:CaseSensitiveObjectDefinition
```

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

For more information about options, and a list of default option settings, see Options used in the command line

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.

Redirecting command line output

The command line output can be redirect using either the */out* switch, or the output redirection features provided by the shell in which you are executing the command.

Using /out

You can use the */out* switch to specify the file to which you want output directed:

```
sqldatacompare ... /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.

Other redirection

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

```
sqldatacompare ... > 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, enter the following:

```
sqldatacompare... >> existinglog.txt
```

This adds the output to the existing file content, without data being lost.

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.

Specify command line arguments in an XML file that can be referenced using the */argfile* switch. For more on this topic see Using XML to specify command line arguments

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.

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.Engine.dll
- SQLDataCompare.exe
- SQLDataCompare.exe.config
- System.Data.SQLite.dll
- zlib1.dll

If you want to use the SQL Data Compare command line as part of a continuous integration process, you need the DLM Automation Suite (previo usly called the SQL Automation Pack).

Examples using the command line

These pages contain examples using the SQL Data Compare Command line.

- Simple examples using the command line
- Example - selecting single tables for comparison
- Example - selecting tables with unrelated names
- Using XML to specify command line arguments
- Deploying a database from source control

Simple examples using the command line

This page provides the following examples of how to use the command line interface:

- Comparing and deploying database data
  Gives examples of comparison, deployment, generating a report, specifying a comparison key, and using an existing project.
- Using a backup as a data source
  Gives examples of specifying backups, multiple backups, and backup sets as a data source.
- Scheduling a comparison
  Gives an example of using the Microsoft Windows Scheduled Task wizard to schedule a comparison.

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

- Example: selecting single tables for comparison
- Example: selecting tables with unrelated names

Comparing and deploying 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 deploy the databases by updating *WidgetLive*:

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

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 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 cannot use */project* with */include* or */exclude*
- all project configuration details - such as column settings, custom comparison keys, and WHERE clauses - are applied when you use */project*
- the project configuration does not include table and row selections for the comparison results
- the command line cannot back up databases before deploymentIf the project includes backup settings that have been set up using the deployment Wizard, those settings are ignored.

To deploy 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\\
/IncludeColumns:WidgetPrices:RecordID,Price,Active
     /deploy
```

Using a backup as a data source

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

```
sqldatacompare
     /Backup1(big grin):\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(big grin):\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(big grin):\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(big grin):\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* tab le 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:

```
sqldatacompare /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:

```
sqldatacompare /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:

```
sqldatacompare /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:

```
sqldatacompare /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:

```
sqldatacompare /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:

```
sqldatacompare /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:

```
sqldatacompare /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:

```
sqldatacompare /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:

```
sqldatacompare /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:

```
sqldatacompare /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.sqb</backup1>
<database2>SecondDatabaseName</database2>
</commandline>
```

Using the command line:

```
sqldatacompare /backup1(big grin):\MSSQL\BACKUP\BackupOfFirstDatabase.sqb
     /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:

```
sqldatacompare /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:

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

Deploying a database from source control

This page gives an example of using the SQL Compare and SQL Data Compare command line interfaces to deploy a database from source control.

In this example, changes have been made to the schema and data of the database *WidgetDev*. These changes must be deployed to the testing database *WidgetTest*.

The latest version of *WidgetDev* is maintained in source control as the scripts folder *WidgetScripts*.

The changes are deployed by using SQL Compare and then SQL Data Compare to deploy *WidgetTest* (the target) with *WidgetScripts* (the source).

For more information, see

- Working with scripts folders
- Continuous integration for databases using Red Gate SQL tools (pdf)

Before we can deploy, we must get the latest version of *WidgetScripts* from source control. In this example, the latest version is updated to the working folder *C:\Scripts\WidgetScripts*

We will deploy all changes to the database schema, and changes to only the static data. The transactional data in the table WidgetPurchases will not be modified.

> - Scripts folders and the command line interface are only available with the SQL Compare and SQL Data Compare Professional Editions.
> - Schema deployment must therefore be performed first, as deployment may fail if the schemas are not identical.

Deploying the schema

To deploy the schema and save a basic report of the process, at the command prompt type:

```
sqlcompare /scr1:"C:\Scripts\WidgetScripts" /db2:WidgetTest
     /o:Default
     /sync /v > "C:\SchemaDeploy.txt"
```

To create a more readable report of the schema differences, and save a copy of the deployment script used to deploy the changes, type:

```
sqlcompare /scr1:"C:\Scripts\WidgetScripts" /db2:WidgetTest
     /o:Default
     /Report:"C:\SchemaDiffReport.html"
     /ReportType:Interactive
     /ScriptFile:"C:\SchemaSyncScript.sql"
     /sync
```

Explanation

- */scr1:"C:\Scripts\WidgetScripts"* specifies *WidgetScripts* as the source
- */db2:WidgetTest* specifies *WidgetTest* as the target
- */o:Default* specifies that the default SQL Compare options will be used for comparison and deployment
- */sync* deploys the data, making *WidgetTest* the same as *WidgetScripts*
- */v > "C:\SchemaDeploy.txt"* directs detailed command line output to a file
- */Report* generates a report of the schema differences and writes it to the file specified
- */ReportType* specifies the format of the report
- */ScriptFile* saves a copy of the SQL script used to migrate the changes

Deploying the data

To deploy the data and create a basic report, at the command prompt type:

```
/sqldatacompare /scr1:"C:\Scripts\WidgetScripts" /db2:WidgetTest
     /o:Default
     /Exclude:table:WidgetPurchases
     /sync /v > C:\DataDeploy.txt
```

To save a copy of the deployment script used to deploy the changes, type:

```
/sqldatacompare /scr1:"C:\Scripts\WidgetScripts" /db2:WidgetTest
 /o:Default
     /Exclude:table:WidgetPurchases
     /ScriptFile:"C:\DataSyncScript.sql"
     /sync /v > C:\DataDeploy.txt
```

Explanation

- */scr1:"C:\Scripts\WidgetScripts"* specifies *WidgetScripts* as the source
- */db2:WidgetTest* specifies *WidgetTest* as the target
- */o:Default* specifies that the default SQL Data Compare options will be used for comparison and deployment
- */sync* deploys the data source, making *WidgetTest* the same as *WidgetScripts*
- */v > "C:\DataDeploy.txt"* directs detailed command line output to a file
- */exclude:table:WidgetPurchases* excludes WidgetPurchases. All other tables will be deployed.
- */ScriptFile* saves a copy of the SQL script used to migrate the changes

Automating the deployment

To automate this deployment, save the whole command line script as a *.bat* file:

```
cd "C:\Program Files\Red Gate\SQL Compare 8"
sqlcompare /scr1:"C:\Scripts\WidgetScripts" /db2:WidgetTest
     /o:Default
     /Report:"C:\SchemaDiffReport.html"
     /ReportType:Interactive
     /ScriptFile:"C:\SchemaSyncScript.sql"
     /sync
cd "C:\Program Files\Red Gate\SQL Data Compare 8"
sqldatacompare /scr1:"C:\Scripts\WidgetScripts" /db2:WidgetTest
     /o:Default
     /Exclude:table:WidgetPurchases
     /ScriptFile:"C:\DataSyncScript.sql"
     /sync /v > C:\DataDeploy.txt
```

You can then schedule deployment using the Microsoft Windows Scheduled Task wizard.

Additionally, you can automatically update the scripts in source control with the changes from the development database by using the development database as the source for the deployment (*/db1:WidgetDev*) and a scripts folder as the target (*/scr2:WidgetScripts*).

Command line syntax

These pages cover the command line switches, options and exit codes in detail:

- Switches used in the command line
- Options used in the command line
- Exit codes used in the command line

Switches used in the command line

This page provides a list of the switches you can use with the SQL Data Compare command line.

Note that:

- the first data source (/db1, /b1, and so on) is the *source*
- the second data source (/db2, /b2, and so on) is the *target*
- there are changes to the command line syntax of SQL Data Compare in version 8 and later
  The names, aliases, and behavior 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.

/activateSerial:<serial number>

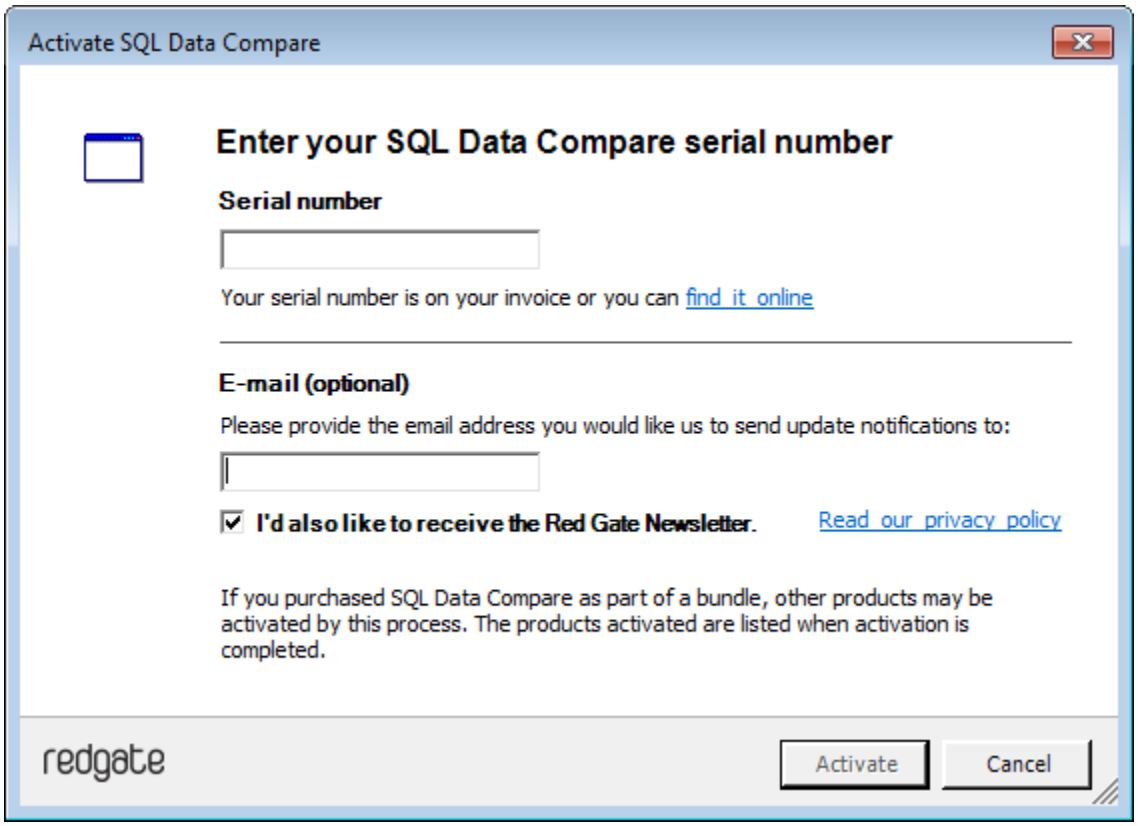
- This switch is case sensitive.
- An internet connection is required to activate SQL Data Compare from the command line.
  For information about how to activate manually without an internet connection, see Activating.

Attempts to activate SQL Data Compare.

You can specify a SQL Data Compare Professional serial number, or a serial number for bundle such as the SQL Developer Bundle:

```
sqldatacompare /activateSerial:123-456-789012-ABCD
```

If you run the switch without specifying a serial number, the **Activate SQL Data Compare** dialog box is displayed:



If you're using the SQL Data Compare command line on a server, you need a DLM Automation Suite license (previously called SQL Automation Pack).

For information about how to activate the DLM Automation Suite, see Activating.

/AllowIdenticalDatabases

This switch is deprecated. Instead use */Include:Identical*

/Include:Identical suppresses the exit code if the two data sources are identical. If */Include:Identical* is not set, SQL Data Compare returns the error code 63.

/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.

/Backup1:<filename1>;<filename2>;...;<filenameN>

Alias: */b1*

Specifies the backup to be used as the first data source (the *source*). You must add all of the files making up the backup set you want to compare:

```
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*

Specifies the backup to be used as the second data source (the *target*). You must add all of the files making up the backup set you want to compare:

```
sqldatacompare /db1:WidgetStaging /Backup2:D:\BACKUPS\WidgetStaging.bak
```

/BackupPasswords1:<Password1>,<Password2>,...,<Password1N>

Alias: */bpsw1*

Specifies the password for the first backup:

```
sqldatacompare /Backup1:D:\BACKUPS\WidgetStaging.bak
     /BackupPasswords1:P@ssw0rd /db2:WidgetProduction
```

/BackupPasswords2:<Password1>,<Password2>,...,<Password1N>

Alias: */bpsw2*

Specifies the password for the second backup:

```
sqldatacompare /db1:WidgetStaging
     /Backup2:D:\BACKUPS\WidgetProduction.bak /BackupPassword2:P@ssw0rd
```

/BackupSet1:<backupset>

Alias: */bks1*

If you are comparing a backup set that contains multiple files, use the */BackupSet1* switch to specify the files which make up the first backup set, and use the */BackupSet2* switches to specify the files which make up the second:

```
sqldatacompare /Backup1:"D:\MSSQL\BACKUP\WidgetDev.bak"
     /BackupSet1:"2008-09-23 Full Backup" /db2:WidgetLive
```

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

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:WidgetDevlopment
```

For encrypted backups that have been created using SQL Backup, use the */BackupPasswords1* and */BackupPasswords2* switches to specify the passwords; when there is more than one password, the passwords are separated using semicolons.

```
sqldatacompare /Backup1:D:\BACKUPS\WidgetDev.sqb /BackupPassword1:Pa$$w0rd
     /db2:WidgetLive
```

/BackupSet2:<backupset>

Alias: */bks2*

Specifies which backup set to use for the second backup:

```
sqldatacompare /db1:WidgetProduction /BackupSet2:"2008-09-23 Full Backup"
```

/CaseSensitive

This switch is deprecated. Instead use */Options:CaseSensitiveObjectDefinition*

/Columns

This switch is deprecated. Instead use */IncludeColumns* and */ExcludeColumns*

/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
     /IncludeColumns:[WidgetPrices]:Price
     /ComparisonKeys:[WidgetPrices]:PK_WidgetPrices
```

- */ComparisonKeys* must be used with the */IncludeColumns* 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 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.

/Database1:<database1>

Alias: */db1*

Specifies a database to use as the source:

```
sqldatacompare /Database1:WidgetStaging /Database2:WidgetProduction
```

/Database2:<database2>

Alias: */db2*

Specifies a database to use as the source:

```
sqldatacompare /Database1:WidgetStaging /Database2:WidgetProduction
```

/Exclude:<object type>:<regular expression>

Arguments:

• *Additional*	only those objects that are not present in the source (eg */db1*)
• *Missing*	only those objects that are not present in the target (eg */db2*)
• *Different*	only those objects that are present in both data sources, but are different.
• *Identical*	identical objects in the command line output and any generated reports.
• *User specified*	objects you specify with a regular expression (eg */Include:Table:WidgetPurchases*)

To specify the list of objects to exclude, use the */exclude* switch:

```
sqldatacompare /db1:Customers1 /db2:Customers2 /Exclude:table
```

`/Exclude:table` specifies that you do not want to compare tables; you only want to compare other objects such as views, stored procedures, and so on.

To specify more than one object or object type type for exclusion use multiple */Exclude* switches. For example, to exclude only tables and views:

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
     /Exclude:table:WidgetReferences /Exclude:view
```

Note that you cannot use */Exclude* with the */Include* and */Project* switches.

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

/ExcludeColumns:<table or view name>:<regular expression>

Alias: */ec*

Specifies which columns within a table or view are excluded from deployment.

Table or view names are matched using a regular expression so that you do not have to specify fully qualified names. However, it does not usually make sense to supply a regular expression that matches more than one table or view. Multiple column names should be specified as a comma-delimited list.

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
     /ExcludeColumns:WidgetReferences:WidgetName
```

Note that */ExcludeColumns* takes precedence over */IncludeColumns*

/Export:<directory>

Alias: */e*

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

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

Note that the file created is always called *Summary.csv*

You specify the directory in which the file is created.

/ExportIdenticalTables

This switch is deprecated. Instead use */Include:Identical*

/Force

Alias: */f*

This 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 will exit with the exit code indicating an IO error.

/IgnoreParserErrors

If SQL Data Compare encounters any high level errors when parsing a scripts folder, it will exit with an error code of 62.

Use */ignoreParserErrors* to force SQL Data Compare to continue without exiting.

/Include:<object type>:<regular expression>

Arguments:

• *Additional*	only those objects that are not present in the source (eg */db1*)
• *Missing*	only those objects that are not present in the target (eg */db2*)
• *Different*	only those objects that are present in both data sources, but are different.
• *Identical*	identical objects in the command line output and any generated reports.
• *User specified*	objects you specify with a regular expression (eg */Include:Table:WidgetPurchases*)

This switch is used to specify the list of objects to include. You can use an /*Include* switch for each object that you want to compare. However, this can be unwieldy if there is a long list. Instead, you can use the pipe character (|) to separate the table names:

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

/Include:Identical suppresses the exit code if the two data sources are identical. If */Include:Identical* is not set, SQL Data Compare returns the error code 63.

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

/IncludeAdditional

This switch is deprecated. Instead use */Include:Additional*

/IncludeColumns

Alias: */ic*

Specifies which columns within a table or view are included in the deployment.

Table or view names are matched using a regular expression so that you do not have to specify fully qualified names. However, it does not usually make sense to supply a regular expression that matches more than one table or view. Multiple column names should be specified as a comma-delimited list.

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
     /IncludeColumns:WidgetReferences:WidgetName
```

Note that */ExcludeColumns* takes precedence over */IncludeColumns*

/MakeScripts:<folder>

Alias: */mkscr*

Creates a scripts folder from the first (source) data source.

```
sqldatacompare /db1:WidgetStaging
     /MakeScripts:"C:\Scripts Folders\Widget staging scripts"
```

If the folder already exists an error will occur. To merge scripts into an existing scripts folder, compare them with that folder and use the */synchron ize* switch:

```
sqldatacompare /scr1:"C:\Scripts Folders\Widget dev scripts"
     /scr2:"C:\Scripts Folders\Widget staging scripts" /Synchronize
```

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

Alias: */o*

Applies the project configuration options used during comparison or deployment:

```
sqldatacompare /db1:WidgetStaging /db2:WidgetProduction
     /Options:Default,IgnoreWhiteSpace
```

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

/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
     /Options:Default,IgnoreWhiteSpace /OutputProject:"C:\WidgetProject.scp"
```

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

/OutputWidth:<columns>

Forces the width of console output.

This can be used to ensure that database object names etc 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 (source).

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

```
sqldatacompare /db1:WidgetStaging /UserName1:User1 /Password1:P@ssw0rd
     /db2:WidgetProduction /UserName2:User2 /Password2:Pa$$w0rd
```

Note that this switch is only used if the source is a database. If the source is a backup, use */BackupPasswords1*

/Password2:<password2>

Alias: */p2*

The password for the second database.

/Project

Alias: */pr*

Uses a SQL Data Compare project (*.scp*) file for the comparison.

To use a project you have saved as "*widgets.scp*" from the command line:

```
sqldatacompare /Project:"C:\SQLCompare\Projects\Widgets.scp"
```

Note that:

- When you use a project, all objects that were selected for comparison when you saved the project are automatically included.
- When you use the command line, your project option selections are ignored and the defaults are used. Use */Options* to specify any additional options you want to use with a command line project.For more information, see: Options used in the command line.
- If you want to include or exclude objects from an existing project, you must modify your selection using the graphical user interface. You cannot use the */Include* and */Exclude* switches with */Project*.

/SyncScriptEncoding

Alias: */senc*

Arguments:

• *UTF8*	UTF-8 encoding, without preamble
• *UTF8WithPreamble*	UTF-8 encoding, with 3-byte preamble
• *Unicode*	UTF-16 encoding
• *ASCII*	ASCII encoding

Specifies the character encoding used when writing the SQL script file. The default is UTF8.

For example:

```
sqldatacompare /db1:WidgetStaging /MakeScripts: D:\Scripts Folder
     /SyncScriptEncoding:ASCII
```

/ScriptFile:<scriptfile>

Alias: */sf*

Generates a SQL script to migrate the changes which can be executed at a later time. 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"
```

/Scriptfile can be used when the target (*/db2*, */scr2*, */sn2*) is a database, a snapshot, or a scripts folder.

If the target is a snapshot or a scripts folder, the generated script modifies a database with the schema represented by that snapshot or scripts folder.

/Scripts1:<folder>

Alias: */scr1*

Specifies the script folder to use as the first data source:

```
sqldatacompare /Scripts1:"C:\Scripts Folder\WidgetStagingScript"
     /db2:WidgetProduction
```

/Scripts2:<folder>

Alias: */scr2*

Specifies the script folder to use as the second data source.

/Server1:<server1>

Alias: */s1*

Specifies the server on which the first (*/db1:*) data source 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*

This specifies the server on which the second (*/db2:*) data source is located. If an explicit path is not specified, it defaults to *Local.*

/Synchronize

Alias: */sync*

Deploys the data after comparison.

The target (for example, */db2*) is modified; the source (for example, */db1*) is not modified:

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

/UserName1:<username1>

Alias: */u1*

The username for the first 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
```

/UserName2:<username2>

Alias: */u2*

The username for the second database.

If no username is specified, integrated security is used.

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 behavior 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 behavior.

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 deployment. Select this option to disable any enabled DDL triggers before deploying the databases, and re-enable those triggers following deployment.

DisableAndReenableDMLTriggers

Alias: *t*

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

DisableKeys

Alias: *k*

Disables foreign keys before deploying the databases, and then re-enables those foreign keys following deployment. 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 deploying the databases, then re-creates them following the deployment.

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 behavior.

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 deploy a view if it includes an identity column.

IncludeIndexedViews

Alias: *v*

Includes views in the comparison. Views can be deployed 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 deployed.

None

Alias: *n*

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

OutputComments

Alias: *oc*

Includes comments in the deployment 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 deployment 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 deployment SQL scripts, enabling changes to be rolled back if the deployment fails. BEGIN TRANSACTION is inserted at the beginning of the deployment 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 deployment 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.

70 - an unhandled exception occurred

See the log for more details.

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.

SQL Server Management Studio add-in

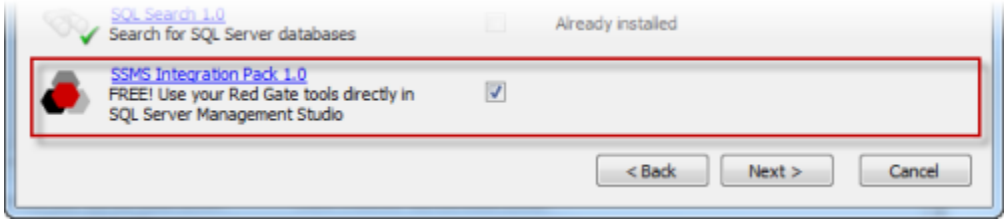
This section explains how to use the SQL Server Management Studio add-in

- Using the add-in

Using the add-in

SQL Data Compare includes a free add-in for SQL Server Management Studio that enables you to compare and deploy (deploy) data from within SQL Server Management Studio.

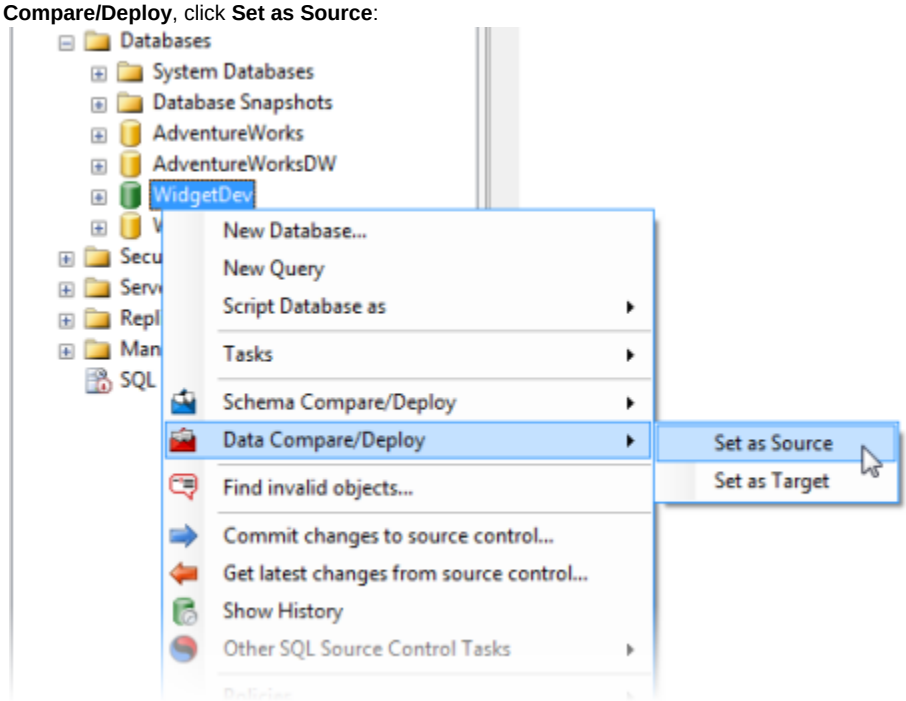
The add-in is bundled with the SQL Data Compare installer, which you can download here. When you run the installer, ensure that the **SSMS Integration Pack** check box is selected:



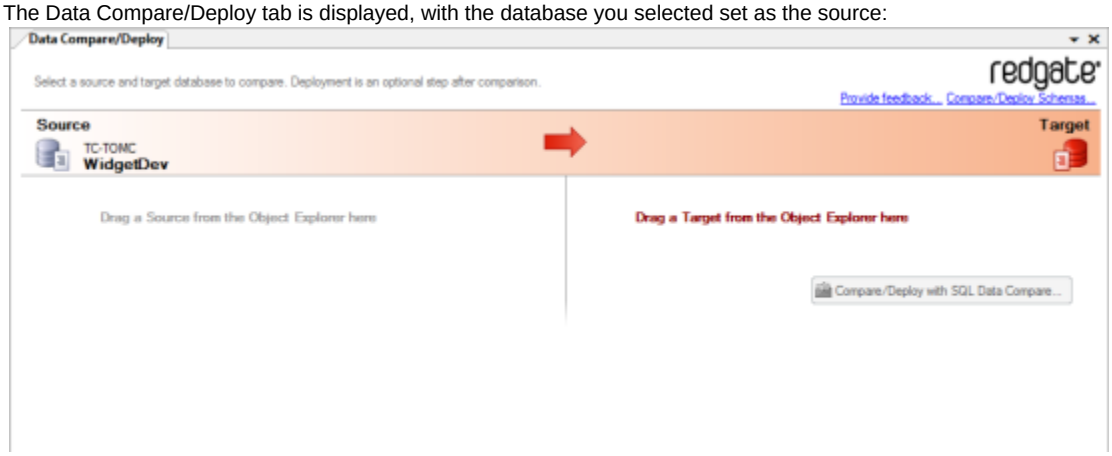
Comparing data

To compare two databases from SQL Server Management Studio:

1. In the SQL Server Management Studio Object Explorer, right-click the database you want to use as the source, and under **Data Compare/Deploy**, click **Set as Source**:



   The Data Compare/Deploy tab is displayed, with the database you selected set as the source:



2. Drag the database you want to use as the target from the Object Explorer to the Target pane.

3. Click **Compare/Deploy with SQL Data Compare.**
   SQL Data Compare launches a creates a new project using the databases you selected.
4. On the SQL Data Compare Project Configuration dialog box, click **Compare Now**.

Getting more from SQL Data Compare

- Deploying to SQL Azure databases
- Comparing databases on different SQL Server versions
- Creating a rollback script
- Getting better performance out of SQL Data Compare

Deploying to SQL Azure databases

With SQL Data Compare, you can compare and deploy SQL Azure databases.

Data types not supported

SQL Azure databases do not support the following data types:

- Common Language Runtime (CLR) User-Defined Types
- Large User-Defined Types (UDTs)

If you try to deploy an unsupported data type to a SQL Azure database, the deployment script will fail.

T-SQL and SQL Server feature support

SQL Azure has further limitations that can cause SQL Data Compare deployments to fail. For example, SQL Azure does not support Data Compression or SQL Server Replication.

For full details of these limitations, refer to your SQL Azure documentation:

- T-SQL Support (SQL Azure)
- SQL Server Feature Limitations (SQL Azure)

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.

Getting better performance out of SQL Data Compare

SQL Data Compare can take a long time to compare and deploy data sources. The speed of the comparison depends mostly on disk write speed, processor speed, memory, and the size of the databases. However, there are a number of things that you can do to get better performance out of SQL Data Compare.

This article explains how you can run faster comparisons and prevent your computer from running out of disk space. It also explains how to schedule the comparison and deployment during periods of low database activity.

Running out of disk space

When you run a comparison, SQL Data Compare retrieves the data from the two data sources and copies the data to a temporary location on your local machine. If the SQL Servers are remote, the data is copied across the network to your local machine.

Although there is no limit on the size of the data sources that you can compare, the amount of available disk space will restrict the amount of data that SQL Data Compare can compare. You'll need roughly twice the size of the database for the comparison and up to four times the size of the database to generate the deployment SQL.

If you do not have enough disk space, you can:

- change the location that SQL Data Compare uses for temporary files to a location with more disk space
- filter the data at table-level, column-level, or row-level so that the temporary files are smaller
- review the options for comparison behavior

Changing the location of the temporary files

By default, the location of the temporary files is defined by the TMP environment variable. You are not recommended to edit the TMP variable as this will affect all programs that use this variable. Instead, you create a new environment variable called RGTEMP, and specify the required location, for example a different hard disk with more disk space. The RGTEMP variable affects only Red Gate programs.

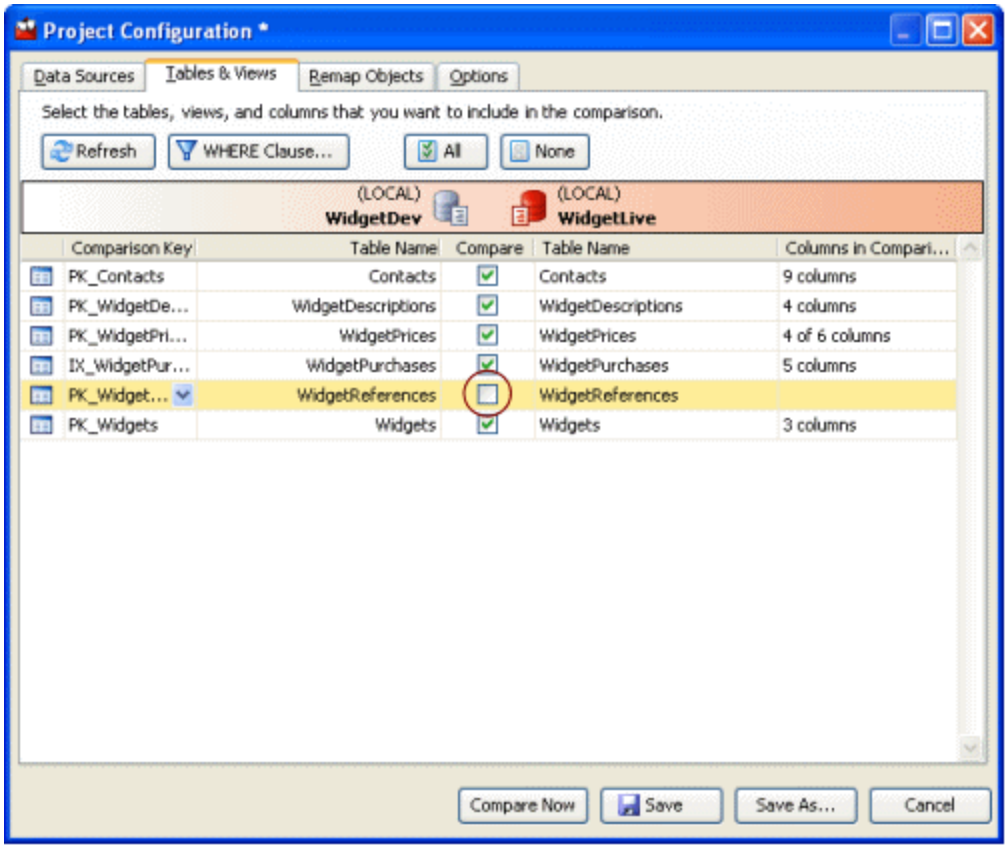
Note that you may need to log out of Windows for the change to take effect.

Filtering the data

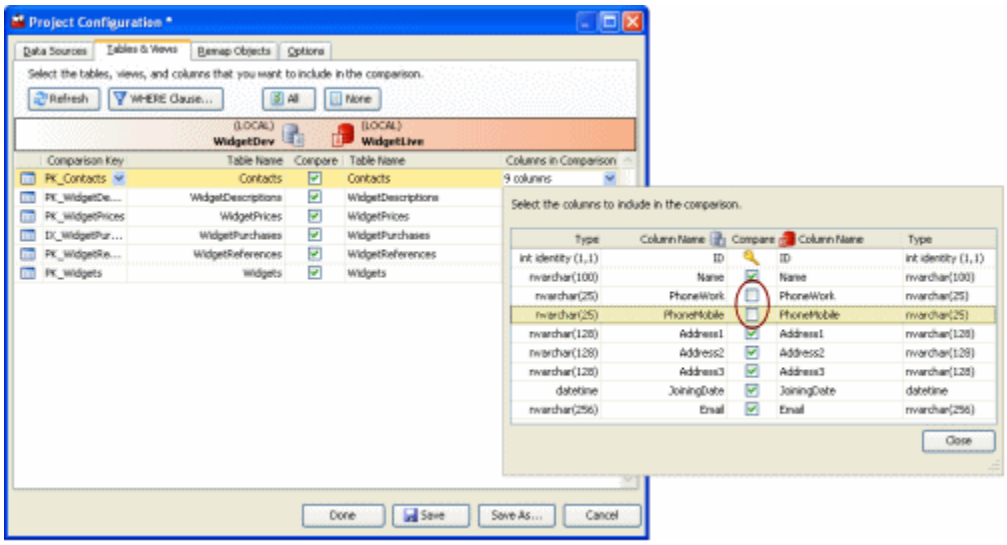
SQL Data Compare selects all tables, all columns, and all rows for comparison, so you'll be temporarily storing all the data that differs in the two data sources. By filtering the data, you compare only the data you're really interested in, and the size of the temporary files is reduced.

You filter data using the **Tables & Views** tab on the **Project Configuration** dialog box. The filtering techniques described here are useful when your tables contain a large amount of data that differs:

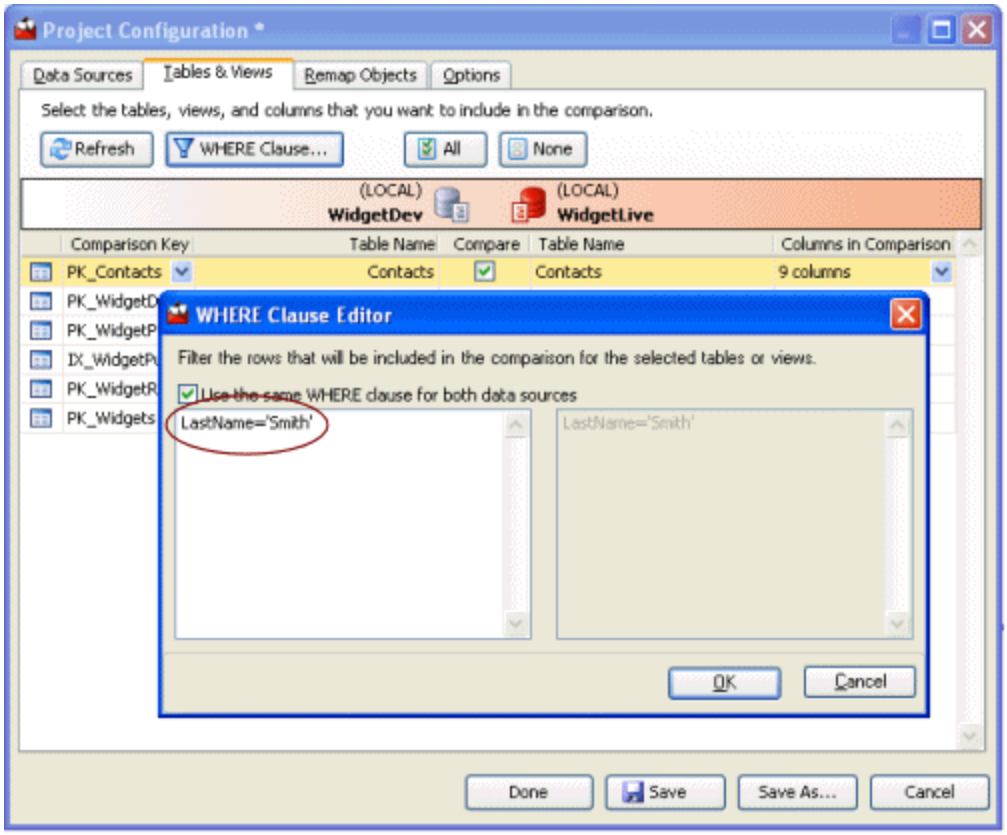
- Table filtering
  Exclude all the data for a table by clearing its check box:



- Column filtering (vertical filtering)
  Exclude all the data in particular columns by clicking in the **Columns in Comparison** box for the associated table, and then clearing the check boxes for the required columns:



- Row filtering (horizontal filtering)
  Exclude rows by clicking WHERE Clause, and then typing a T-SQL WHERE clause in the box:



You can apply the same WHERE clause to multiple tables, by highlighting the required tables before opening the **WHERE Clause Editor**.

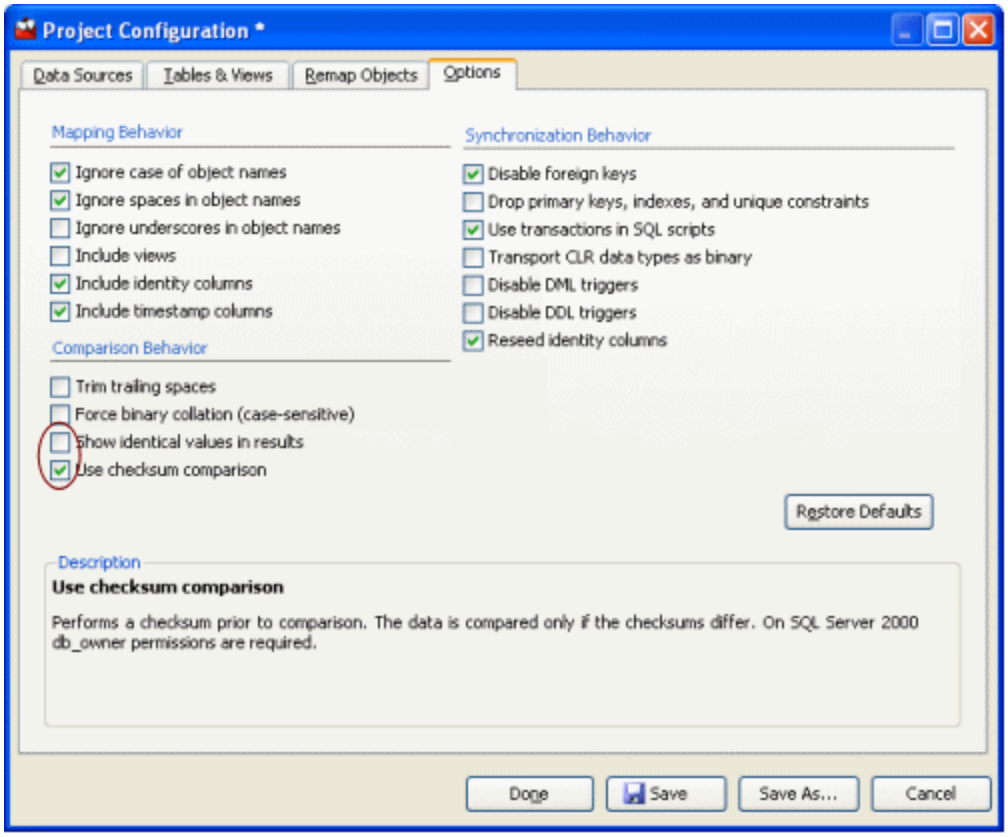
You can apply a different WHERE clause for each data source by clearing the **Use the same WHERE clause for both data sources** check box.

Note that row filtering is not available if you are using a backup as a data source.

Reviewing the options for comparison behavior

By default, SQL Data Compare includes identical values in the comparison results. If your data sources always contain similar data, clear the **Show identical values in results** check box on the **Options** tab of the **Project Configuration** dialog box. When you do this, SQL Data Compare does not store data that is identical, reducing the disk space that is required.

If you are comparing tables with large amounts of data that changes infrequently, select the **Use checksum comparison** check box. You'll need to clear the **Show identical values in results** check box before you do this. If the checksums are equal, SQL Data Compare won't compare that table, and the data is not stored for that table, reducing the disk space that is required.



Note that if your table contains large data types such as text or images, these columns are not included in the checksum, and you are not recommended to use the **Use checksum comparison** option.

Scheduling the comparison and deployment

If you still find the comparison or the deployment is taking too long to run, you may want to schedule these tasks to run at a time when your SQL Server has low levels of activity. To do this, you'll need the Pro edition of SQL Data Compare.

In SQL Data Compare version 6, when you set up your comparison project, you can save the project file wherever you want. You can then create a script with SQL Data Compare commands to open the comparison project and run the deployment.

For example, you could create a batch file (.bat) as follows:

```
C:
cd path_to_SQLDataCompare_installation_folder
SQLDataCompare

/project:"D:\SQLDataCompare\Projects\Widgets.sdc"
/deploy
```

You can then use the Microsoft Windows Scheduled Task wizard to run the batch file.

Summary

In this article, you've learned how to change the location of temporary files, filter the data before it is compared, and set the options that determine comparison behavior so that you don't run out of disk space. To find out more about setting up your comparison projects, see Selecting tables and views.

You've also learned how to schedule a comparison and deployment using the command line. To see more examples using the command line, see Examples using the command line.

Worked examples

These detailed examples demonstrate how to complete tasks using SQL Data Compare.

- Worked example - deploying data in two databases
- Worked example - restoring from a backup file

Worked example - deploying data in two databases

This worked example demonstrates a basic comparison and deployment 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've 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'll need access to a SQL Server to do this.

This example has four steps:

1. Set up the databases
   Create the example databases on your SQL Server.
2. Set up the comparison
   Specify the data sources, tables, and views you want to compare.
3. Select data to deploy
   Review the results and select the rows you want to deploy.
4. Deploy the databases
   Create and run a deployment script.

1. Set up the databases

The worked example uses the following sample databases:

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

To create the databases on your SQL Server:

1. Click here to download the SQL creation script for the databases.
2. Copy the script, paste it in your SQL editor application, and run it.

   > If databases named WidgetDev, WidgetTest or WidgetLive already exist on your SQL Server, the script will overwrite them.

   The databases are created and populated with data.

2. Set up the comparison

In this example, there are three steps to setting up the comparison:

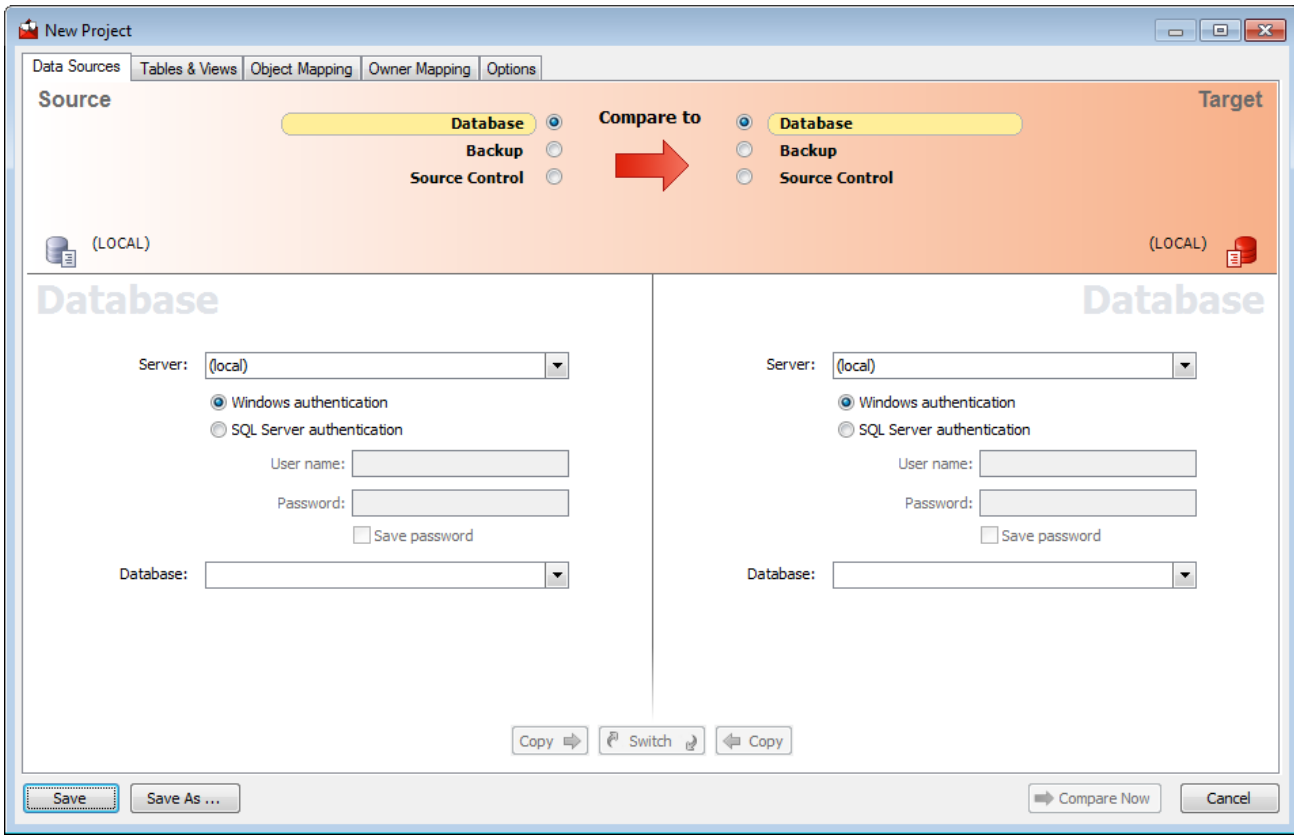
1. Specify data sources
2. Select tables and views
3. Set object mappings

> It's not always necessary to set object mappings. However, in this example there are some schema differences between the source and target. For more information, see Mapping objects

Specify data sources

1. Start SQL Data Compare if it is not already running.
   The Project Configuration dialog box is displayed showing your most recent project:
   You can edit the current project, or create a new project.
   If you want to create a new project, click **Cancel** to close the dialog box, and on the toolbar click

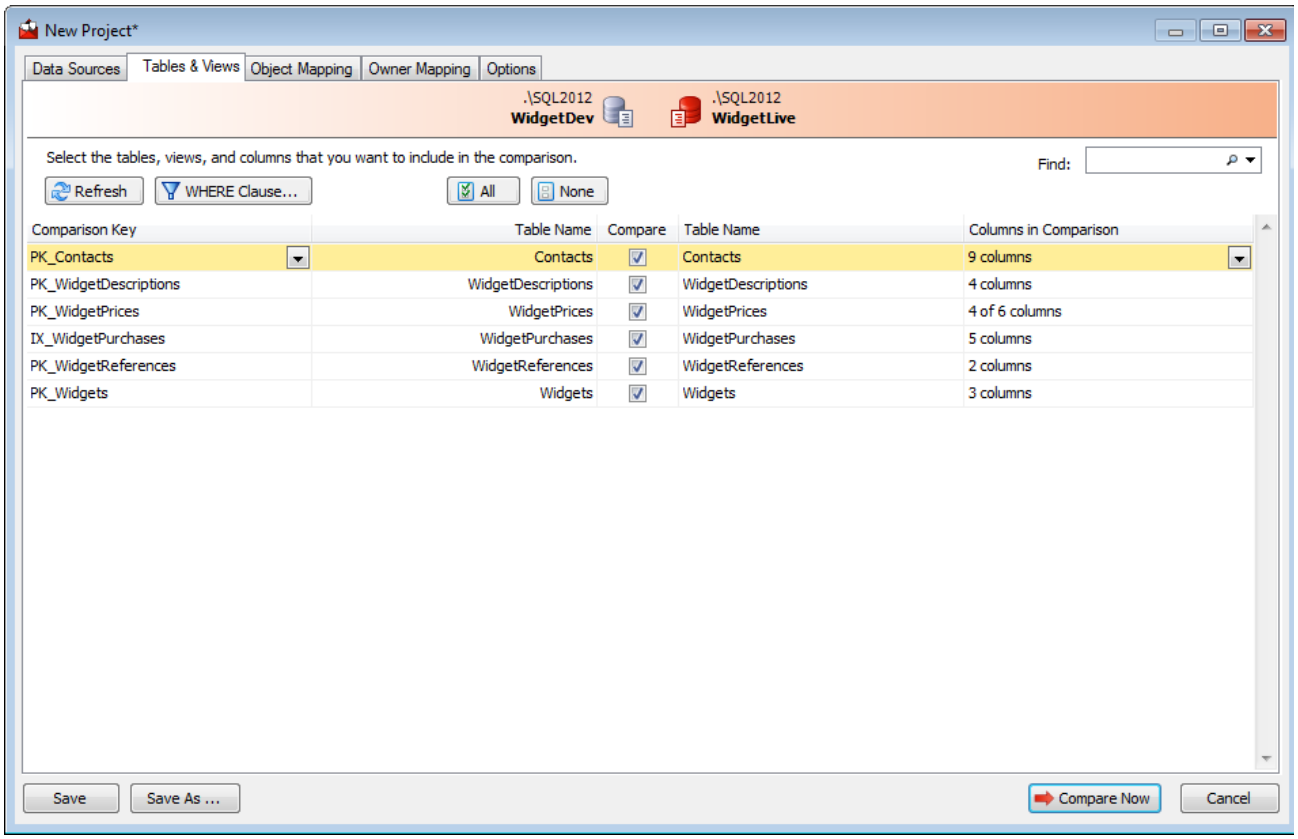
   (New Project).
   If you have any existing projects, and the Projects dialog box is displayed instead, click **New Project**.



2. In the shaded upper pane, ensure the Source and Target are set to **Database**.
   In this example, we will compare databases. You can also compare backups and scripts folders.
3. For each data source, in the **Server** box, type or select the name of the server on which you set up the databases.
4. For the source, in the **Database** box, type or select *WidgetDev***.**
   For the target, type or select ***WidgetLive***.
   If the databases are not displayed in the **Database** lists, right-click in each **Database** box and click **Refresh**, or scroll to the top of the list and click **Refresh**.

Select tables and views

1. To specify the tables and views you want to include, click the **Tables & Views** tab:

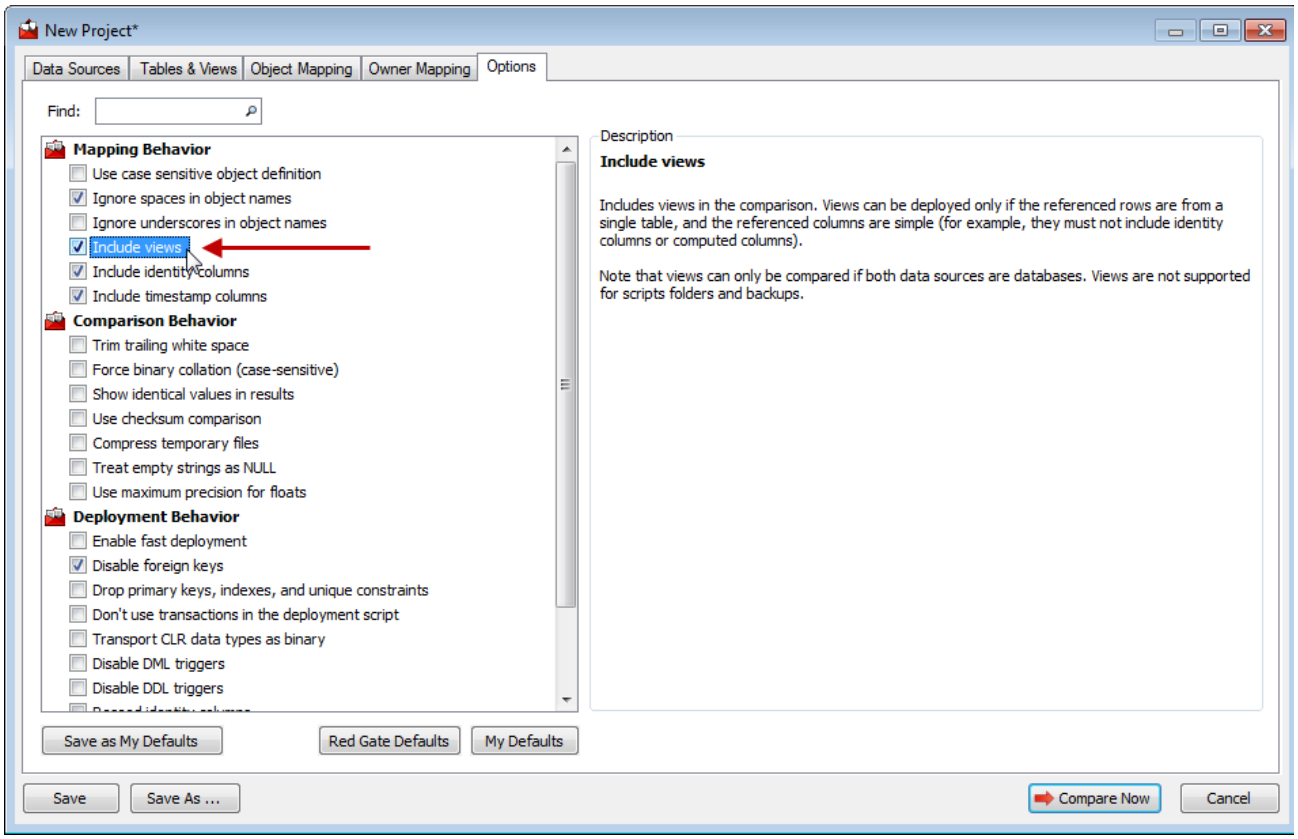


The **Tables & Views** tab enables you to specify:

- the tables and views you want to compare
- which specific columns you want to compare
- the comparison key SQL Data Compare uses to match rows in the two databases

> By default, SQL Data Compare does not compare views.

2. To include views, on the **Options** tab, select the check box for **Include views**:



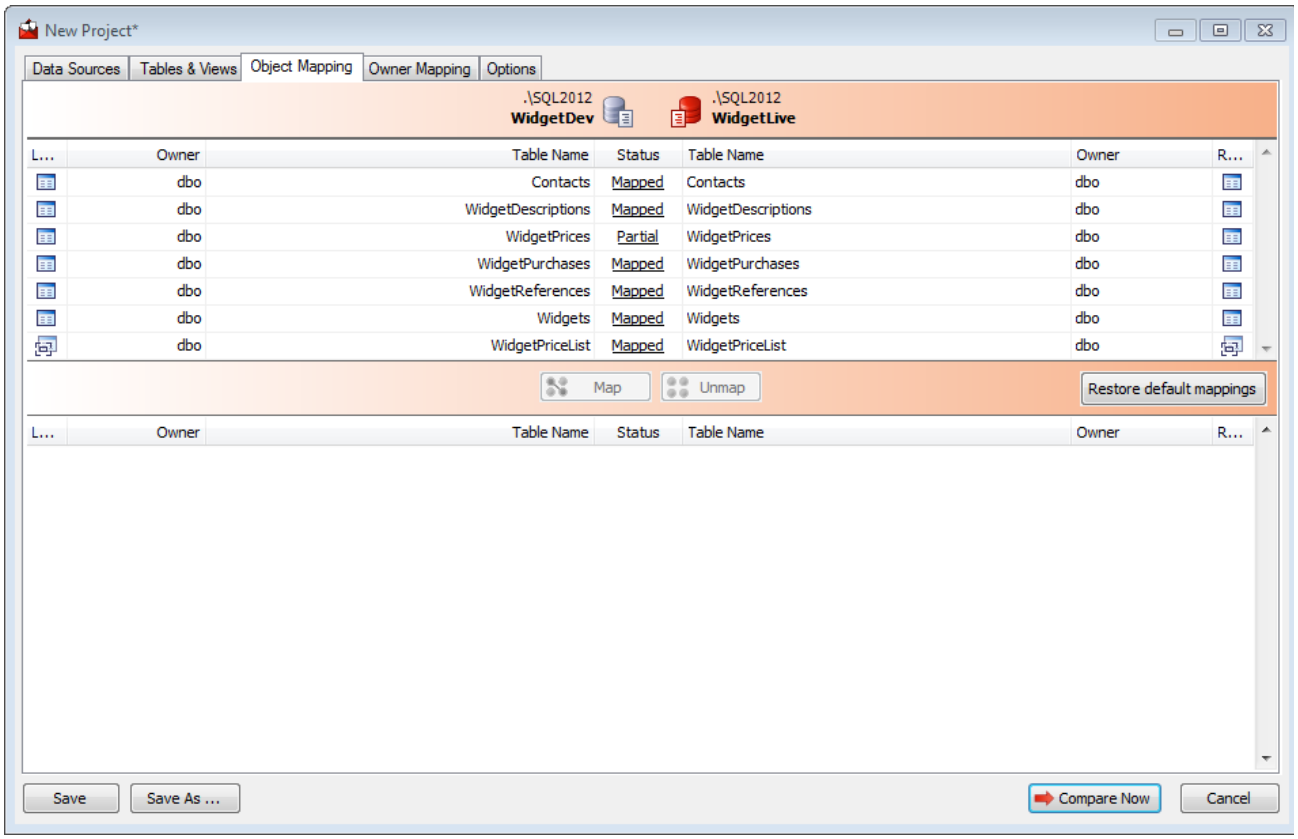
The **Options** tab allows you to modify the behavior of SQL Data Compare during comparison and deployment.

3. Click the **Tables & Views** tab again.
   The WidgetPriceList view is now included in the comparison.

> For the WidgetPrices table, only 4 out of the 6 columns will be compared, as two of the column names do not match. To include the remaining columns, you must map the objects

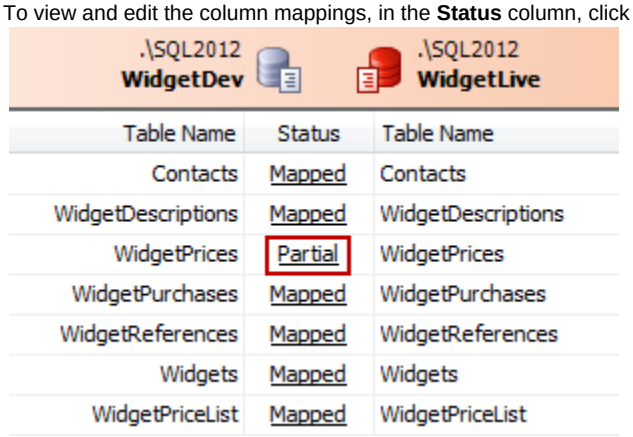
Set object mappings

1. To include the remaining columns, click the **Object Mapping** tab:

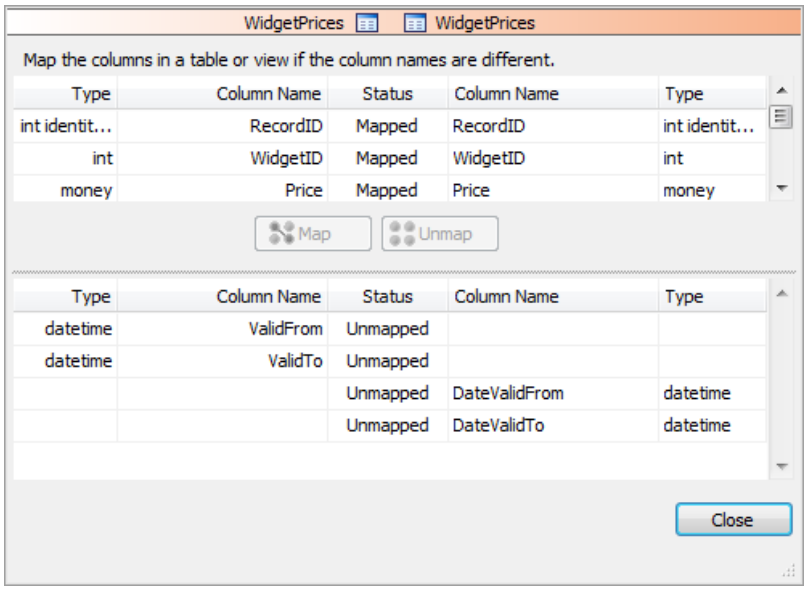


The **Object Mapping** tab displays a list of table and view mappings.
Mappings define which tables and views can be compared. For example, the table WidgetDescriptions in WidgetDev is mapped to WidgetDescriptions in WidgetLive. However, it can be mapped to any table with a sufficiently similar structure.
You can use the object mappings to compare tables with different or similar names.
The table WidgetPrices shows a **Partial** mapping, as not all of its columns could be automatically mapped.

2. To view and edit the column mappings, in the **Status** column, click **Partial** for the WidgetPrices table:



A dialog box is displayed for you to edit the column mappings:

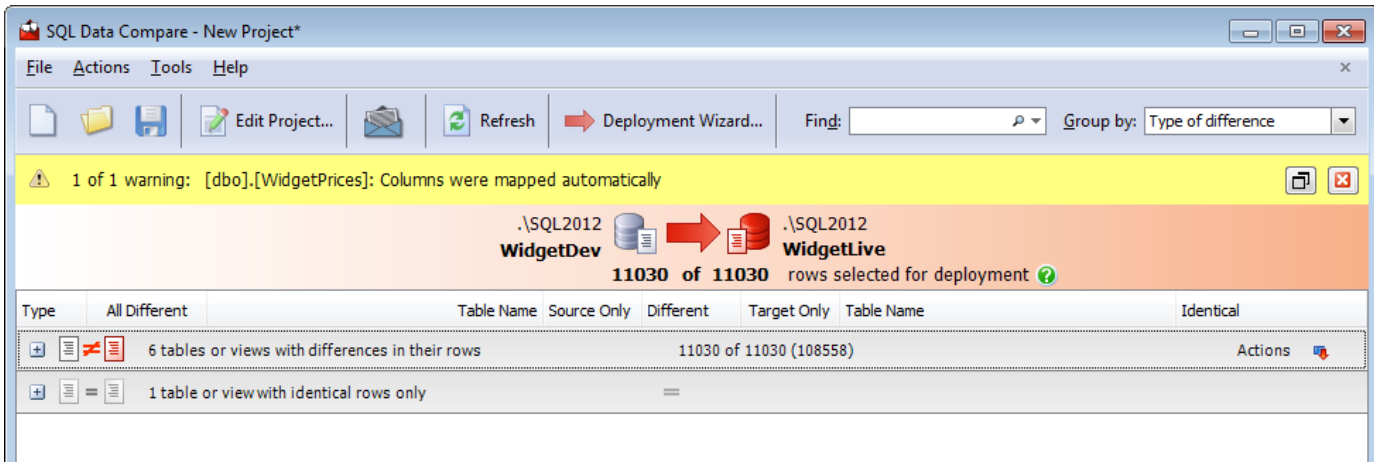


3. In the lower pane of the dialog box, click *DateValidFrom* and then *ValidFrom*, and then click
   Map to map the columns.The columns move to the upper pane of the dialog box.
4. Map the *DateValidTo* and *ValidTo* columns in the same way, and click **Close**.The status for the WidgetPrices table changes to *Mapped*.
5. Click **Compare Now**.

SQL Data Compare displays a message dialog box that shows the progress of the comparison. If you select the **Close dialog box on completion** check box, SQL Data Compare closes this message dialog box automatically the next time that you run a comparison on a project. For this example, leave the setting as it is, and click **OK** to close the message box.

3. Select data to deploy

The comparison results are displayed in the main window.



A yellow warning bar is displayed indicating that SQL Data Compare has automatically mapped columns in the WidgetPrices table:

In this example we mapped the columns manually, so we can ignore this warning.

In this example, the comparison results are grouped by:

tables or views with differences in their rows

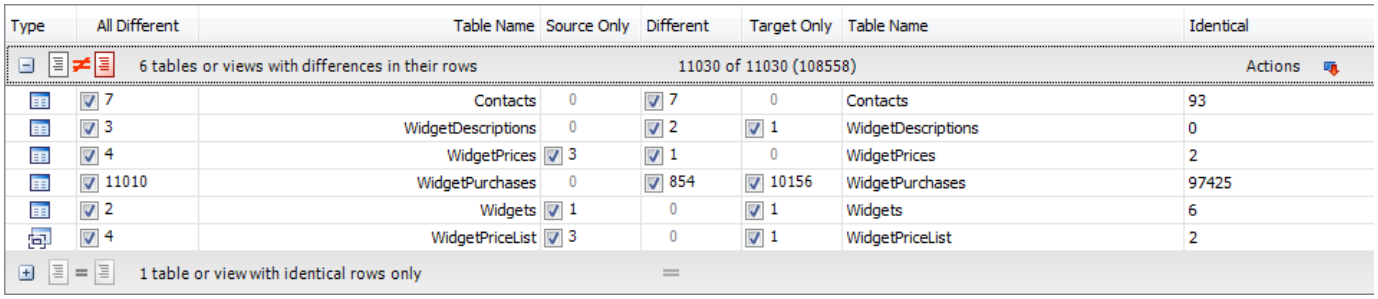
tables or views with identical rows only

To display the comparison results in a single list, in the **Group by** box, select *No groups.*

To display the comparison results in groups, select *Type of difference*.

To view the comparison results for each object group, click

or click the grouping bar:



The upper (Results) pane also shows how many rows of each type exist for each table or view. For example, the table WidgetPrices contains:

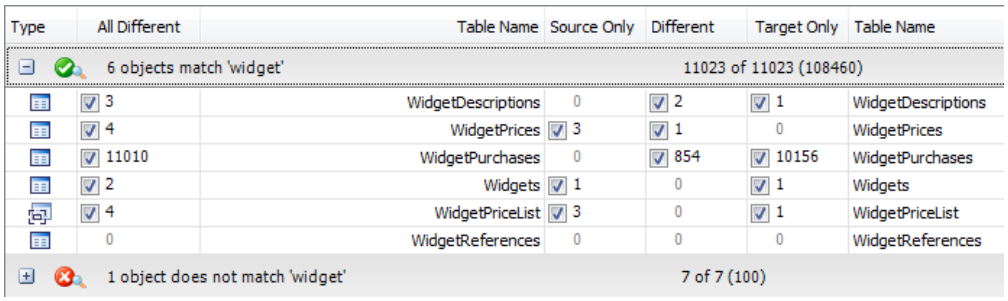
- three rows that exist in WidgetDev but not in WidgetLive
- one row that exists in both databases but has different values
- no rows that exist in WidgetLive but not in WidgetDev.

The Identical column shows that there are two rows that are identical.

Finding objects

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:



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

Note that the search is not case-sensitive.

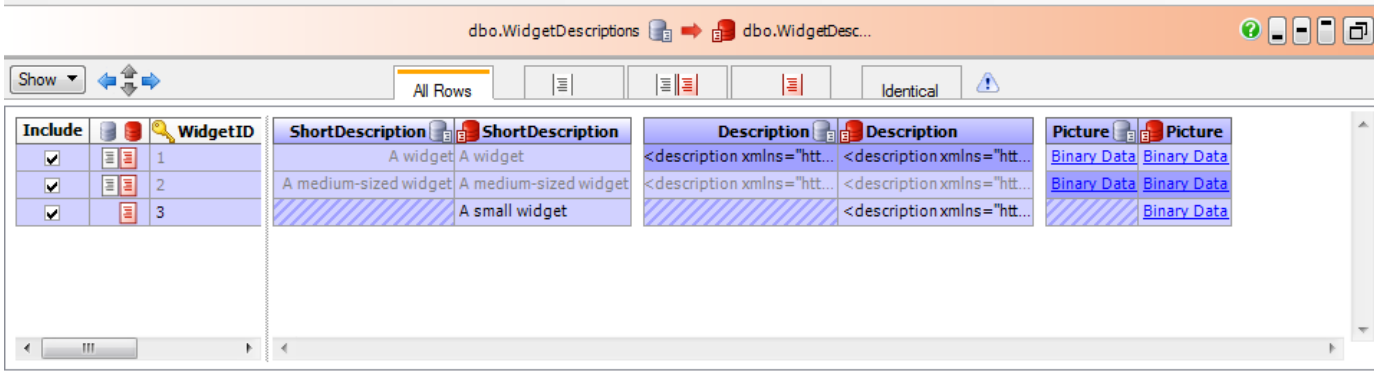
To clear the **Find** box click the

button.

Viewing row differences

To view detailed information about the data in a table or view, click on it in the upper pane. The lower (Row Differences) pane is displayed.

Click the table *WidgetDescriptions* to see the row differences:



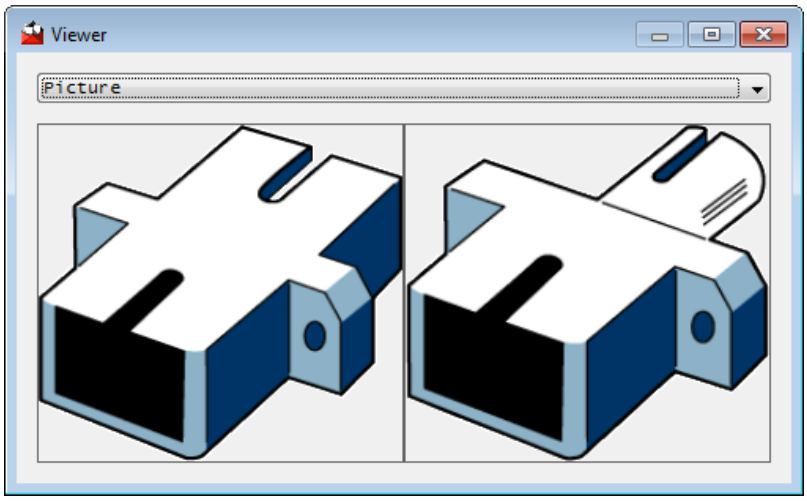
In this example **WidgetID** is the comparison key

, used by SQL Data Compare to match rows in the two databases.

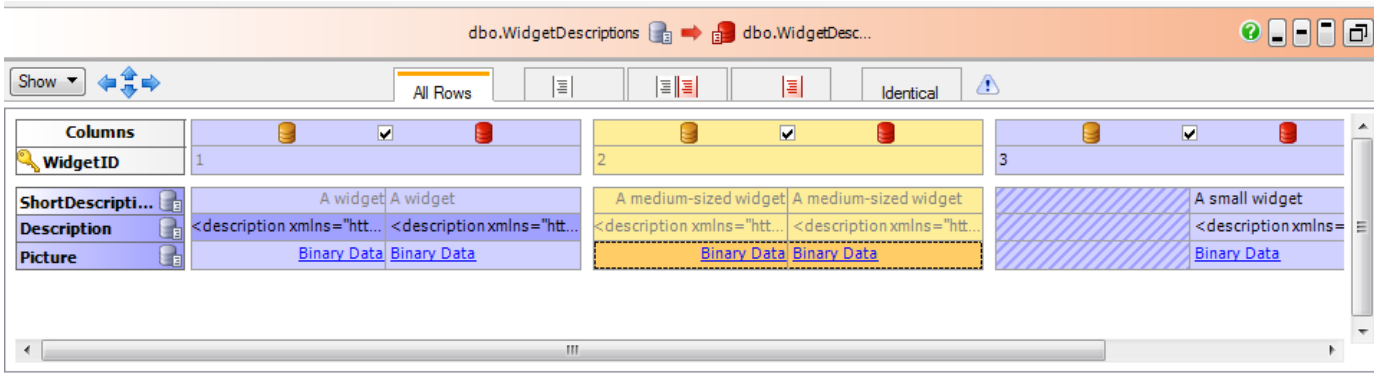
For the row where WidgetId is 2, the values for **Short Description** and **Description** are the same in both databases, so the data is displayed in gray text.

The **Picture** values are different so they are displayed with the darker shaded background.

To view a specific value in a row double-click it, or right-click and select **Show Viewer**. **Binary Data** values are displayed as links. Click the **Binary Data** link for the row where WidgetID is 2:



For tables or views that contain a large number of columns, you may find it useful to display the column values vertically so that you can see records more easily. To do this, click **Show**, then **Pivot View**:



To display the column values horizontally again, click **Show**, then **Pivot View**.

You can move through each difference in turn using the

and

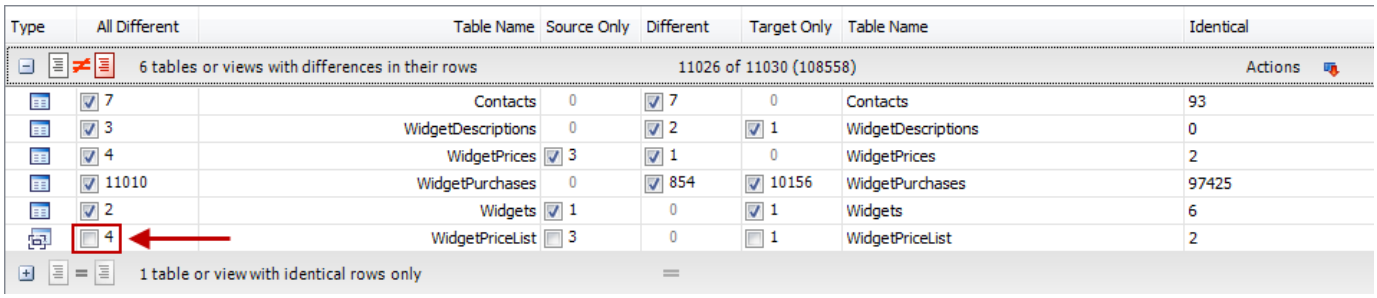
buttons. To move through the differences for the current column, click

or

Selecting rows to deploy

Use the check boxes in each pane to select data for deployment. You can select individual tables or views to deploy, and for each table or view you can select the rows that you want to deploy. By default, all rows that differ are selected when you run a comparison.

For this example, select all the tables and their rows. Make sure that the check boxes are cleared for the WidgetPriceList view. You do this by clearing the **All Different** check box for WidgetPriceList:



The number of rows selected for deployment is shown in the Direction Bar.

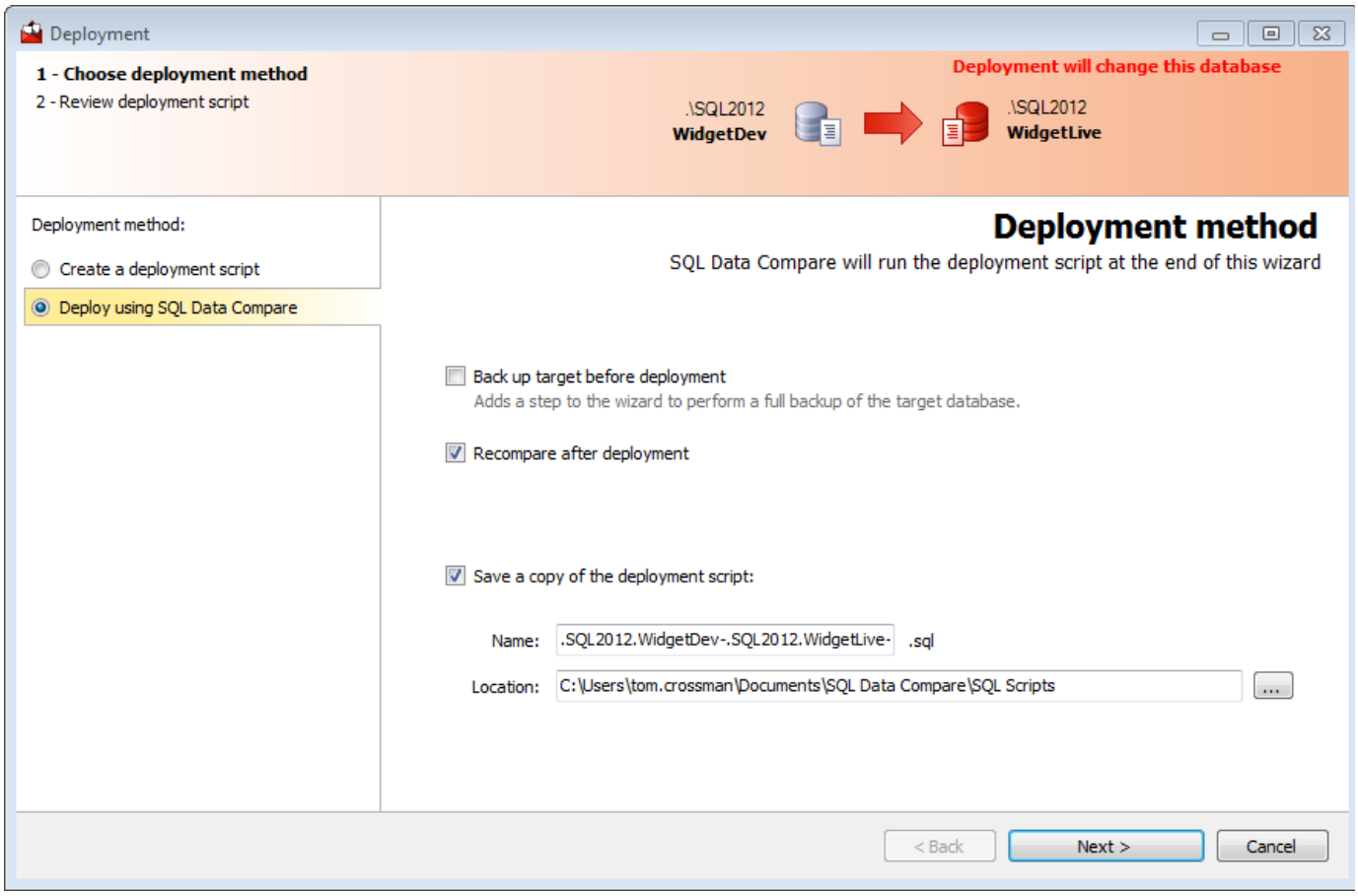
To deploy, click **Deployment Wizard.**

4. Deploy the databases

On the first page of the deployment Wizard you can choose to create and save a deployment script, or perform the deployment using SQL Data Compare.

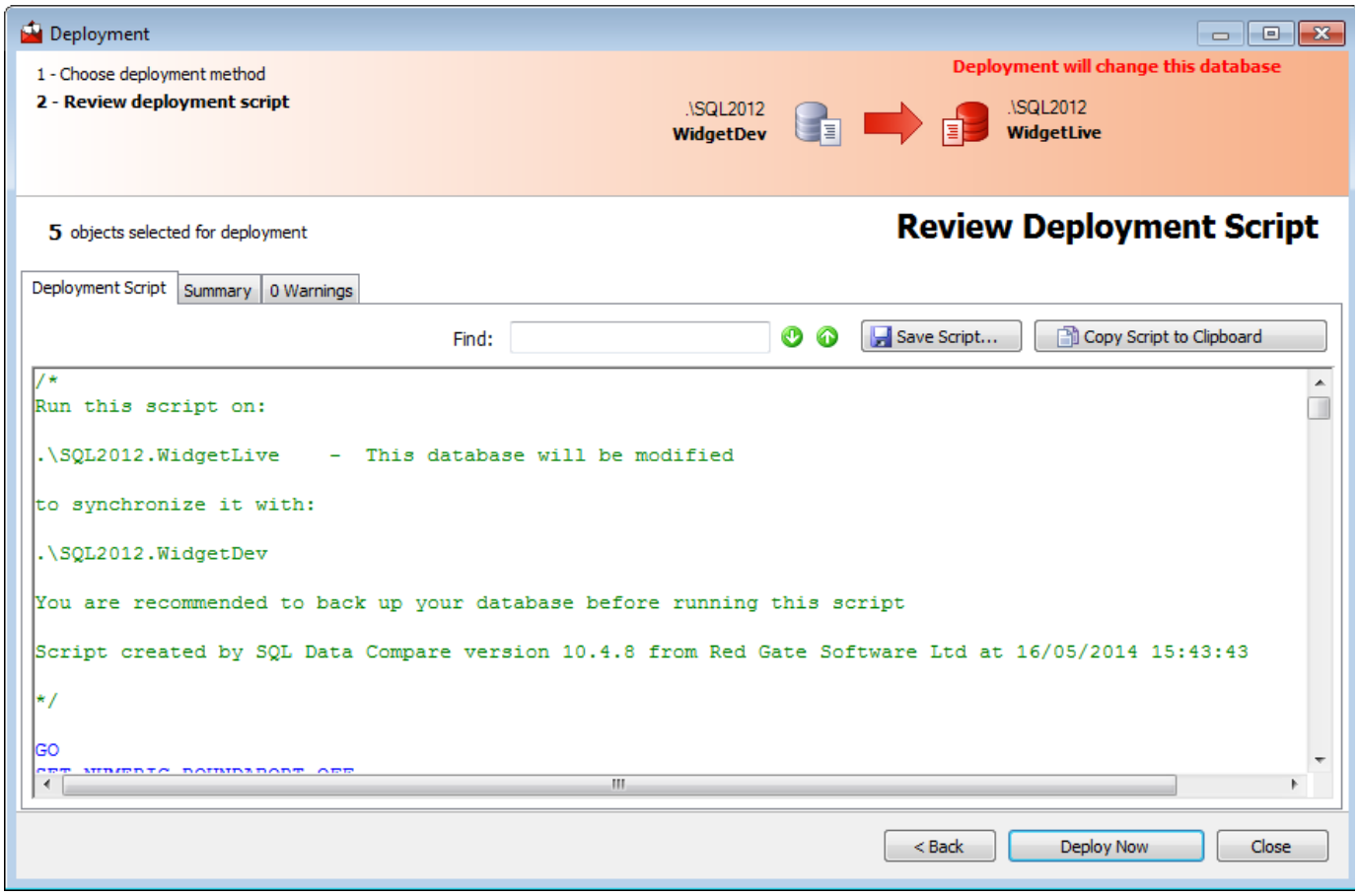
Choose deployment method

In this example, we will deploy using SQL Data Compare.



1. Make sure **Deploy using SQL Data Compare** is selected.
2. Clear the **Back up target before deployment** check box if it's selected.
   In this example, we won't back up before deployment.
   For more information, see Backing up before deployment
3. To generate the deployment script, click **Next**.
   The **Review** page is displayed.

Review the deployment script



There are three tabs on the **Review** page:

- **Deployment script** shows the script to deploy the data sources.
  You can search the script, save it, or copy it to the clipboard.
- **Summary** shows a synopsis of the actions in the deployment script.
  You can view the summary grouped by the objects affected, by the type of modification, or by the order in which the script modifies the target.
- **Warnings** shows a list of any warnings about unexpected behavior that may occur when you deploy the databases.

For more information, see Deployment warnings.

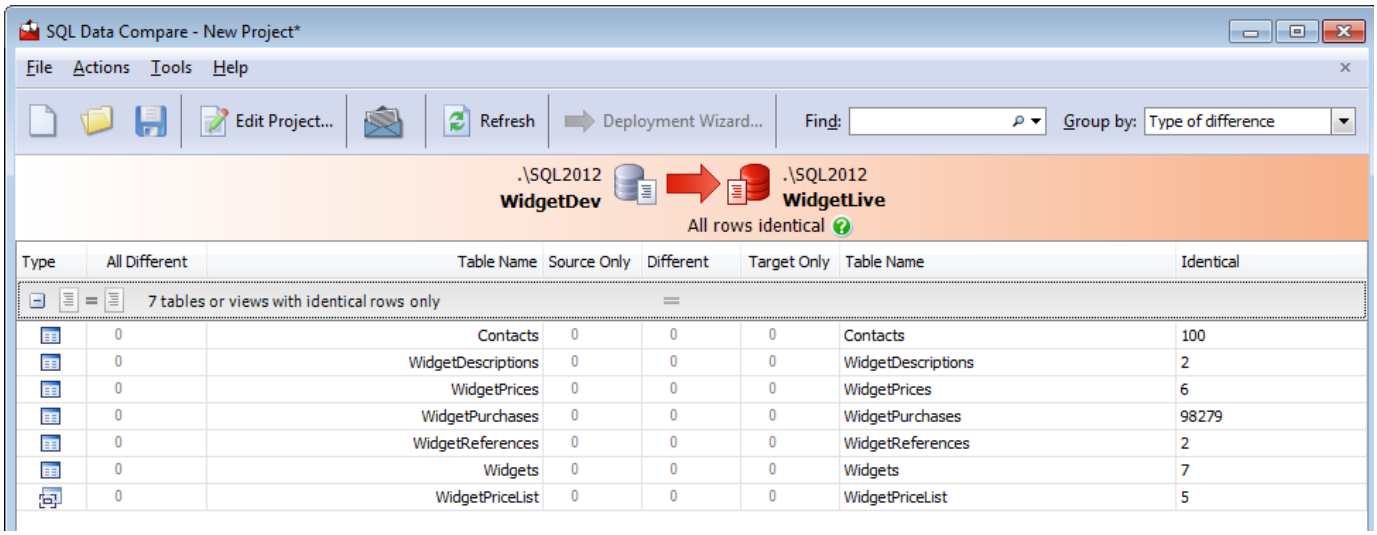
Perform the deployment

When you have reviewed the script, deploy the databases:

1. Click **Deploy Now** to perform the deployment.
2. A confirmation dialog box is displayed. Click **Deploy Now** to continue.
3. SQL Data Compare displays a message dialog box that shows the progress of the deployment.
   When the deployment is complete, click **OK** to close the message box.

SQL Data Compare then re-compares the databases. The results are shown in the main window. In this example, all tables and their rows are shown to be identical, confirming that the deployment has been a success.

Note that the WidgetPriceList views are also now identical, because the tables that they reference have been deployed:


SQL Data Compare - New Project*
File Actions Tools Help
Edit Project...
Refresh
Deployment Wizard...
Find:
Group by: Type of difference
.\SQL2012
WidgetDev
.\SQL2012
WidgetLive
All rows identical
Type All Different Table Name Source Only Different Target Only Table Name Identical
7 tables or views with identical rows only
0 Contacts 0 0 0 Contacts 100
0 WidgetDescriptions 0 0 0 WidgetDescriptions 2
0 WidgetPrices 0 0 0 WidgetPrices 6
0 WidgetPurchases 0 0 0 WidgetPurchases 98279
0 WidgetReferences 0 0 0 WidgetReferences 2
0 Widgets 0 0 0 Widgets 7
0 WidgetPriceList 0 0 0 WidgetPriceList 5

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.

> We recommended you follow the Deploying data in two databases worked example before starting this worked example.

This example has four steps:

1. Set up the databases
   Create the example databases on your SQL Server.
2. Set up the comparison
   Create the example databases, and specify the data sources you want to compare.
3. Select rows to restore
   Review the results and select the rows you want to restore.
4. Deploy the data sources
   Create and run a deployment script.

1. Set up the databases

The worked example uses the following data sources:

- WidgetDev is the development database
- WidgetTest is the testing database
- WidgetLive is the production database
- BeforeEmailUpdate.bak is the backup file

To create the databases on your SQL Server:

1. Click here to download the SQL creation script for the databases.
2. Copy the script, paste it in your SQL editor application, and run it.

   > If databases named WidgetDev, WidgetTest or WidgetLive already exist on your SQL Server, the script will overwrite them.

   The databases are created and populated with data.

To create the backup file, run the following command:

```
BACKUP DATABASE WidgetLive
TO DISK = 'c:\BeforeEmailUpdate.bak'
WITH INIT,
NAME='BeforeEmailUpdate'
```

The backup is created at the root of the C drive on the computer running the instance of SQL Server you are connected to.

2. Set up the comparison

To improve performance, and make the comparison results easier to interpret, in this example we will compare only a single column in a single table. To do this, we will specify the data sources, then the table, then the column. No other values are included in the comparison.

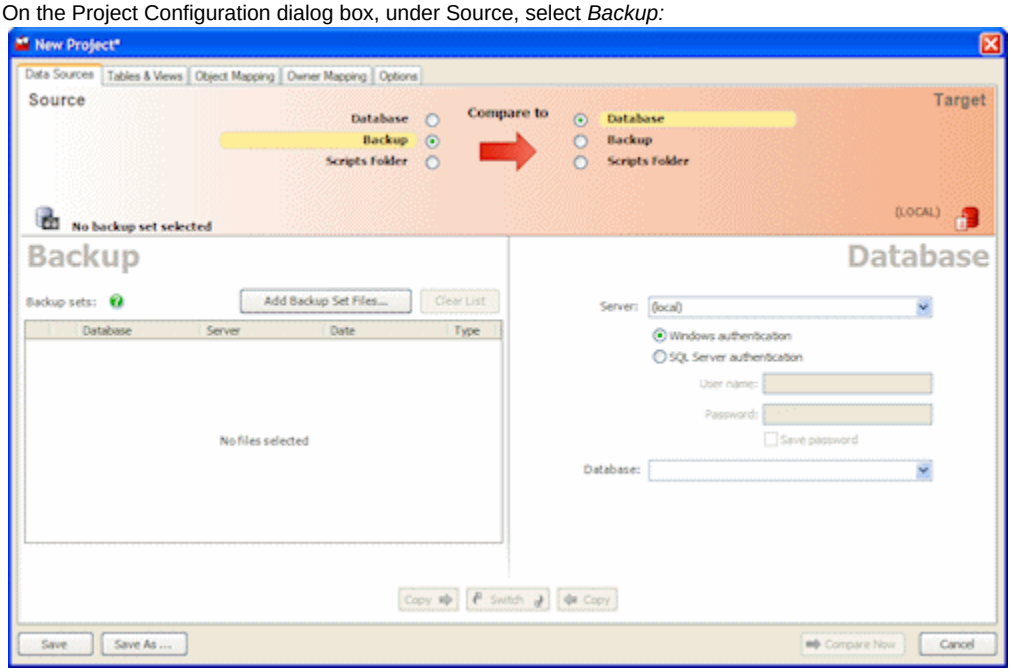
Specify data sources

1. Start SQL Data Compare if it is not already running.

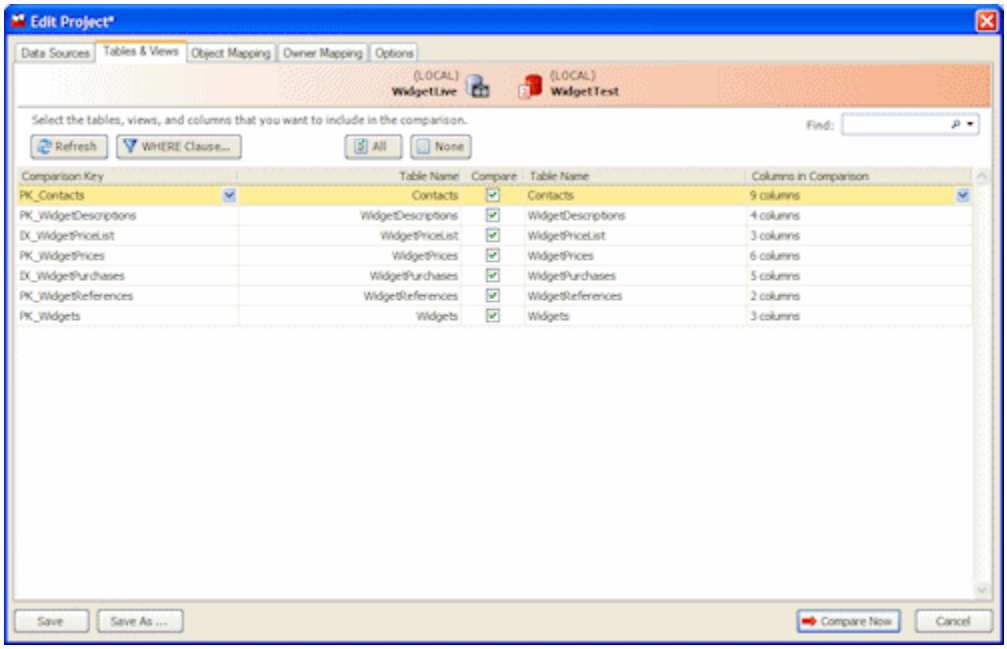
The Project Configuration dialog box is displayed showing your most recent project.
You can edit the current project, or create a new project.
If you want to create a new project, click **Cancel** to close the dialog box, and on the toolbar click

(New Project).
If you have any existing projects, and the Projects dialog box is displayed instead, click **New Project**.

2. On the Project Configuration dialog box, under Source, select *Backup:*



3. Click **Add Backup Set Files** to specify the backup.
   The **Add Backup set Files** dialog box is displayed.
4. Select the file *BeforeEmailUpdate.bak*, and click **Open**.
   The BeforeEmailUpdate backup set is displayed in the list of backup sets.
5. Under Target, select *Database*.
6. In the **Server** box, type or select the name of the SQL Server where you created the sample databases.
7. Type or select *WidgetTest* in the **Database** box.
   If the database is not displayed in the **Database** list, right-click the **Database** box and click **Refresh**, or scroll to the top of the list and click **Refresh**.
8. On the **Options** tab, click **Red Gate Defaults**, to ensure the default options are being used.
9. Click the **Tables & Views** tab:

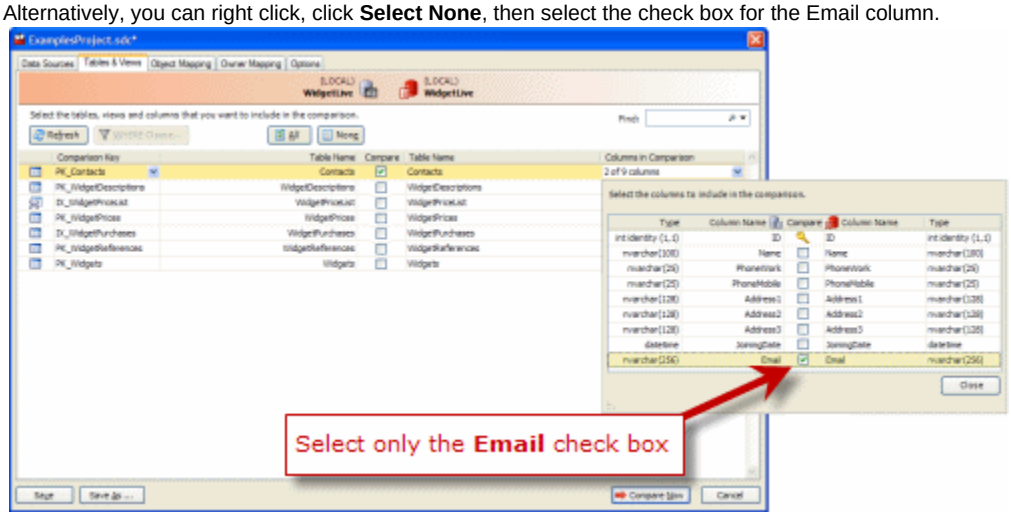


The **Tables & Views** tab enables you to select the tables you want to compare and which columns you want to compare.

By selecting only the tables that you want to compare, you will improve the performance of SQL Data Compare, as it does not need to read the entire backup file. This is useful for large backup files.Note that if you are using a backup as a data source, you cannot compare views.

10. Click

    None, then select the check box for the Contacts table.
    Only Contacts is included in the comparison.
11. Click in the **Columns in Comparison** box for the Contacts table, and on the dialog box, clear the check boxes for all columns except Email, then click **Close**.
    Alternatively, you can right click, click **Select None**, then select the check box for the Email column.



Only 2 out of the 9 columns in the Contacts table will be compared; the ID column (the comparison key) and the Email column.

12. Click **Compare Now**.

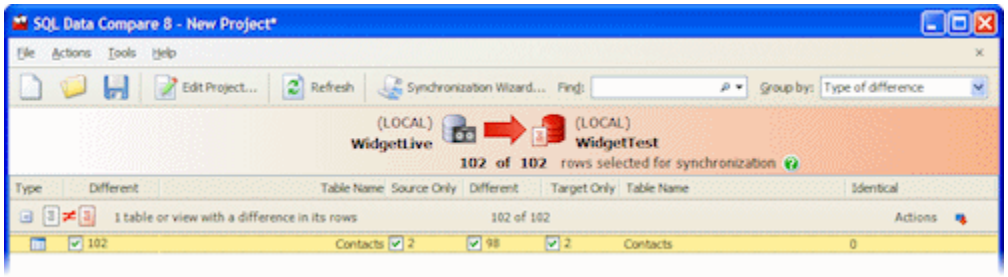
SQL Data Compare displays a message dialog box that shows the progress of the comparison. If you select the **Close dialog box on completion** check box, SQL Data Compare closes this message dialog box automatically the next time that you run a comparison on a project. For this example, leave the setting as it is, and click **OK** to close the message box.

3. Select rows to restore

The results of the comparison are displayed in the upper (Results) pane:

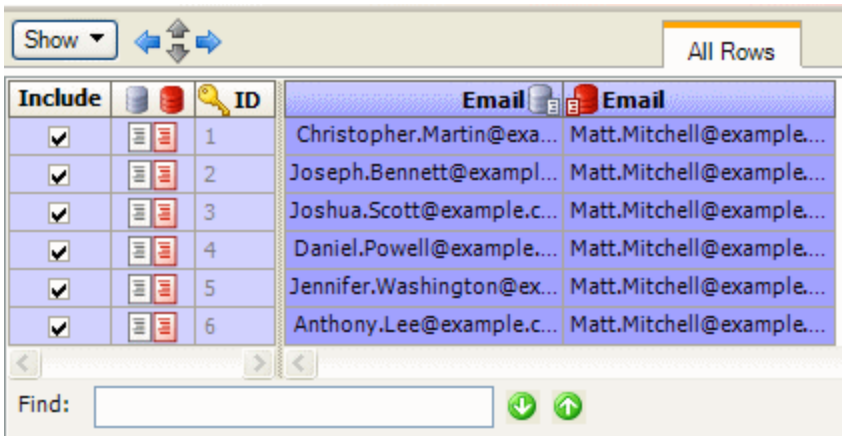
To view the comparison results, click

or click the object group bar:



The upper pane also shows how many rows of each type exist. There are 2 rows that exist in the backup but not in the WidgetTest database, 98 rows that exist in both the backup and the WidgetTest database but have different values, and 2 rows that exist in the WidgetTest database but not in the backup.

In the upper pane, click Contacts to display detailed information about the rows:



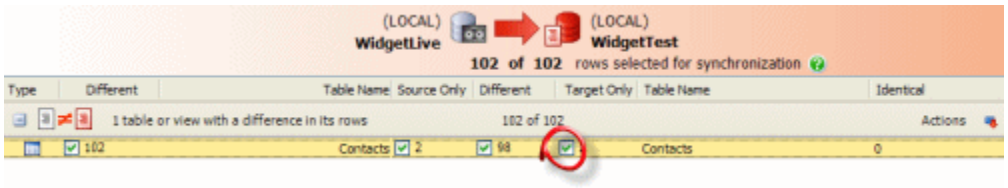
ID is the comparison key

. Email values that are different are displayed with a dark shaded background; Email values that do not exist in one data source but do exist in the other are displayed with a shaded, patterned background.

If you scroll through the rows, you can see that all the Email values have been set to *Matt.Mitchell@example.com* except for the two rows that do not exist in the WidgetTest database and the two rows that do not exist in the backup.

As restoring all rows would over-write those that exist only in the target, we must exclude these rows from the deployment.

In the upper pane, clear the **Target Only** check box:



100 of 102 rows are now selected for deployment.

To restore specific rows, you can use the **Include** check boxes in the lower (Row Differences) pane to select the rows you want to deploy. Alternatively, you can filter the comparison with a WHERE clause.

In this example, we will deploy the remaining 100 rows.

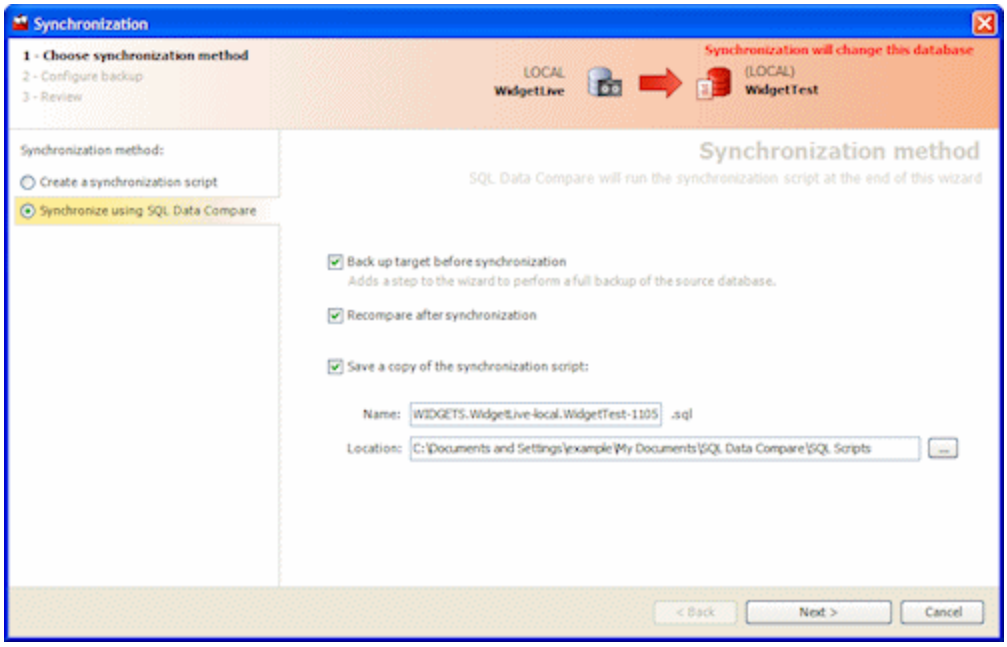
Click

Deployment Wizard.

4. Deploy the databases

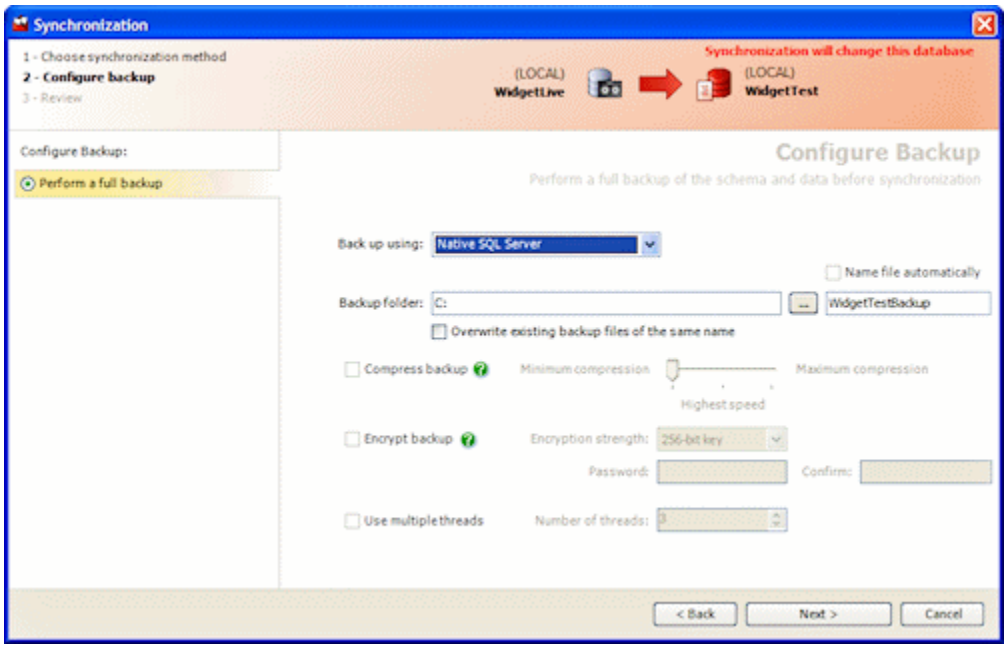
On the first page of the deployment Wizard you can choose to create and save a deployment script, or perform the deployment using SQL Data Compare. In this example, we will deploy using SQL Data Compare, and perform a backup before deployment.

Choose deployment method



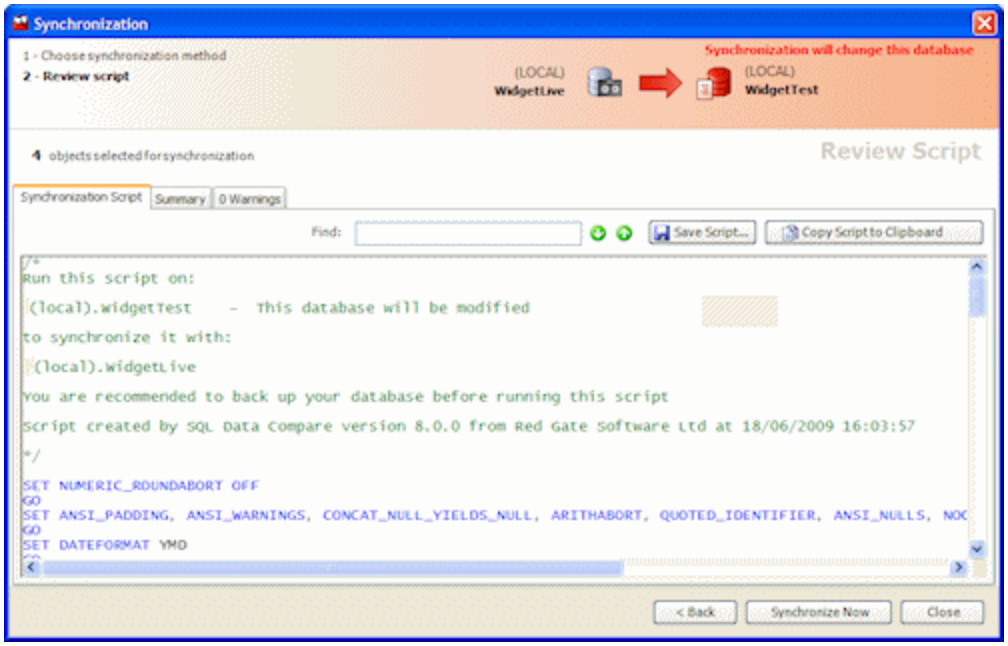
1. Ensure that **deploy using SQL Data Compare** is selected.
2. Ensure that **Back up target before deployment** is selected.
3. Ensure that **Save a copy of the deployment script** is selected.
4. Click **Next**

Configure backup



1. In the **Back up using** check box, select *Native SQL Server*
   If you have installed SQL Backup on the SQL Server, you can select *Red Gate SQL Backup* instead.
   For more information, see Backing up before deployment
2. In the **Backup folder** box, specify the backup location, and in the box on the right hand side, type a file name, for example *WidgetTestBackup.*
3. Click **Next**

Review



There are three tabs on the **Review** page:

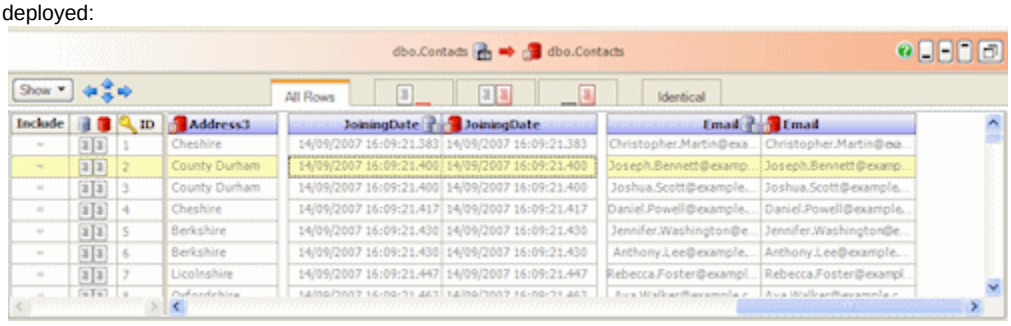
- **Deployment script** shows the script to deploy the data sources.
  You can search the script, save it, or copy it to the clipboard.
- **Summary shows** a synopsis of the actions in the deployment script.
  You can view the summary grouped by the objects affected, by the type of modification, or by the order in which the script modifies the target.
- **Warnings shows** a list of any warnings about unexpected behavior that may occur when you deploy the databases.

For more information, see: Deployment warnings.

You can choose to run the SQL script from within SQL Data Compare, launch your SQL editor so that you can modify the script, or save the SQL script without deploying the databases.

In this example, we will choose to run the script, then compare the databases again to check the results, and save a copy of the deployment SQL script (the default)

1. Click **Deploy Now** to run the deployment.
2. A confirmation dialog box is displayed. Click **Deploy Now** to continue.
   The backup is created, the deployment is performed, then the data sources are compared.
   For this example, leave the setting as it is, and click **OK** to close the message box.
   The comparison results show that the Email values for rows that existed in both the backup and the WidgetTest database have been deployed:



Troubleshooting

- Common issues
  - Case-sensitive comparisons
  - Cleaning up a SQL script after SQL Compare or SQL Data Compare
  - Comparing the data of two tables in the same database
  - Improving the performance of SQL Data Compare
  - Reseed applying "incorrect" identity values
  - SQL Data Compare showing differences in two identical databases
  - Tables or views that could not be compared
  - Troubleshooting comparison and deployment performance problems
  - Using a filter on a column on related (joined) tables
- Error messages
  - NULL textptr passed to UPDATETEXT function when running deployment
  - Running migration scripts using SqlCmd.exe
  - This SQL Server has been optimized for x concurrent queries
  - Troubleshooting System.OutOfMemoryException during comparison
- Unexpected behavior / technical questions
  - How much free hard disk space is required?
  - How to force SQL Compare and SQL Data Compare to use an encrypted connection
  - Minimum database permissions for SQL Data Compare
  - Using Windows authentication logons between domains
- Logging and log files

Common issues

- Case-sensitive comparisons
- Cleaning up a SQL script after SQL Compare or SQL Data Compare
- Comparing the data of two tables in the same database
- Improving the performance of SQL Data Compare
- Reseed applying "incorrect" identity values
- SQL Data Compare showing differences in two identical databases
- Tables or views that could not be compared
- Troubleshooting comparison and deployment performance problems
- Using a filter on a column on related (joined) tables

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.

> 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.

Cleaning up a SQL script after SQL Compare or SQL Data Compare

If you have produced a SQL script and you want to edit it by removing lines relating to certain databases or users for example, then you can use a simple DOS command to do all the tricky work for you in one line.

Create a DOS batch file the contents being as follows:

```
findstr /v <string to be removed> <inputfile> > temp.txt
move /y temp.txt <inputfile>
```

So to remove any lines containing 'sp_addrolemember' in a file called test1.sql the batch file will contain:

```
findstr /v sp_addrolemember test1.sql > temp.txt
move /y temp.txt test1.sql
```

Therefore if test1.sql starts off containing these lines:

```
EXEC sp_addrole N'roleMCEStore'
EXEC sp_addrolemember N'roleMCEStore',N'PROD\svcMCEStore'
GO
```

Run the batch file and test1.sql will now contain:

```
EXEC sp_addrole N'roleMCEStore'
GO
```

By typing `findstr /?` you will reveal the large number of options available for pre or post SQL processing.

Comparing the data of two tables in the same database

For testing and data sanitization reasons, you may want to compare data in two similar tables in the same database.

To do this, specify the same server and database as the source and target. Next, click the **Object Mapping** tab, and map together the tables you want to compare. For information about how to map tables, see Mapping objects.

If the columns of the tables are named differently, it may be necessary to map the individual columns together as well.

> Make sure the data types of the columns you map together are the same, or at least similar. For example, it's possible to map together CHAR and NVARCHAR columns, int and tinyint columns, and decimal and float columns.

Improving the performance of SQL Data Compare

SQL Data Compare can take a long time to compare and deploy data sources. The speed of the comparison depends mostly on disk write speed, processor speed, memory, and the size of the databases.

When you run a comparison, SQL Data Compare retrieves the data from the two data sources and copies it to a temporary location on your local machine. If the SQL Servers are remote, the data is copied across the network to your local machine.

Although there is no limit on the size of the data sources that you can compare, the amount of available disk space restricts the amount of data that can be compared. You need roughly twice the size of the database for the comparison and up to four times the size of the database to generate the deployment SQL.

To speed up the comparison, and avoid running out of disk space, you can:

- change the location of the temporary files
- filter the comparison at table-level, column-level, or row-level
- review the options for comparison behavior

Changing the location of the temporary files

SQL Data Compare uses temporary files when it compares the databases. To avoid running out of disk space, you can change the location of these 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).

You are not recommended to edit the **TMP** variable, as this will affect all programs that use the variable. Instead, you can create a new environment variable called **RGTEMP**, and specify the required location, such as a different hard disk with more free space. The **RGTEMP** variable affects only Red Gate programs.

> You may need to log out of Windows for the change to take effect.

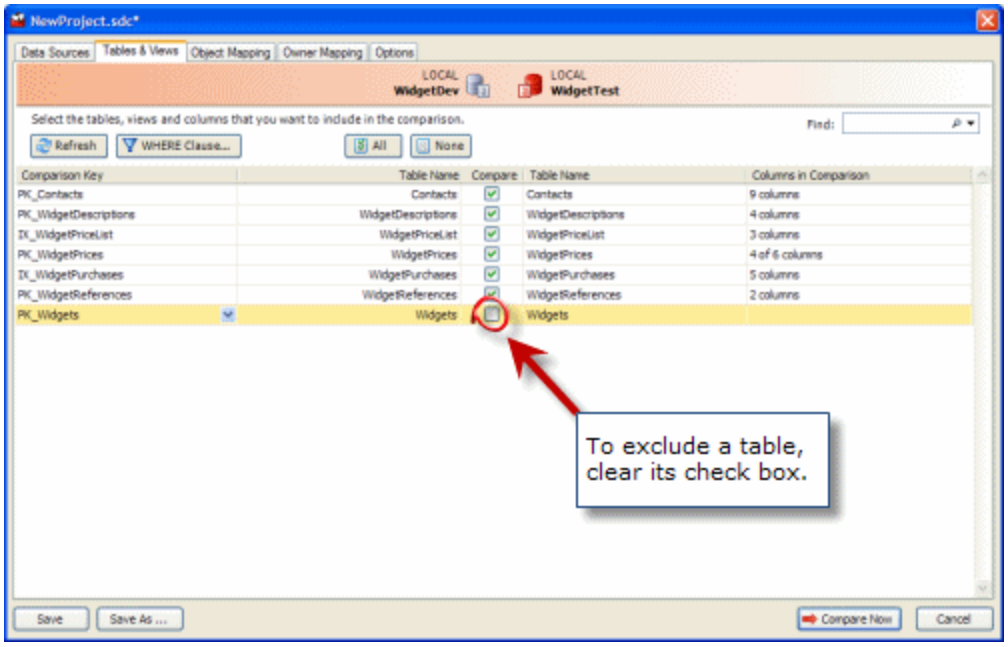
Filtering the comparison

By default, SQL Data Compare selects all tables, all columns, and all rows for comparison. This means all the data that differs in both data sources is stored temporarily. By filtering the data, you compare only the data you are really interested in, and the size of the temporary files is reduced.

You filter data using the **Tables & Views** tab on the **Project Configuration** dialog box. The filtering techniques described here are useful when your tables contain a large amount of data that differs:

Table filtering

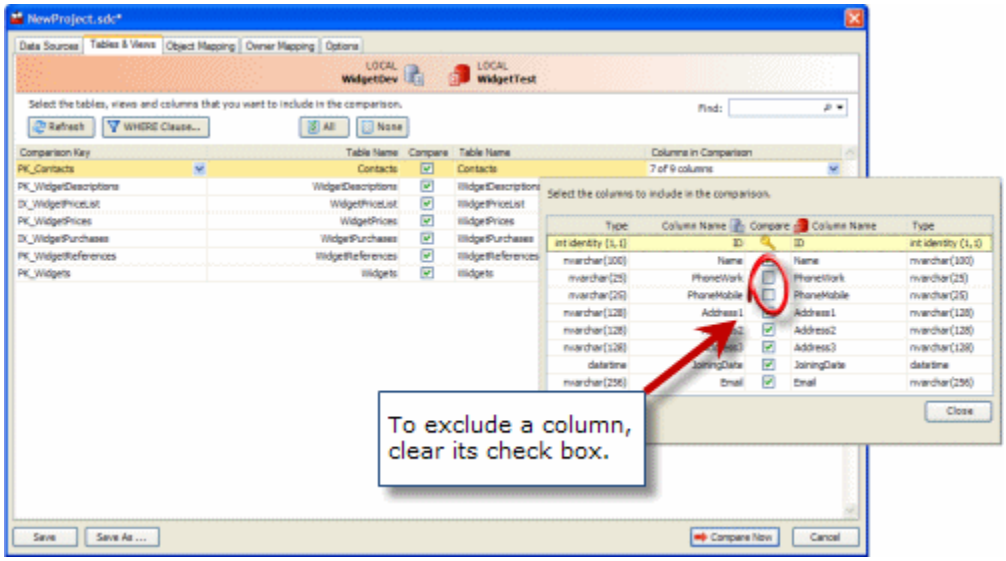
To exclude all the data in a table, on the **Tables & Views** tab of the **Project Configuration** dialog box, clear its check box:



The table will not be compared.

Column filtering (vertical filtering)

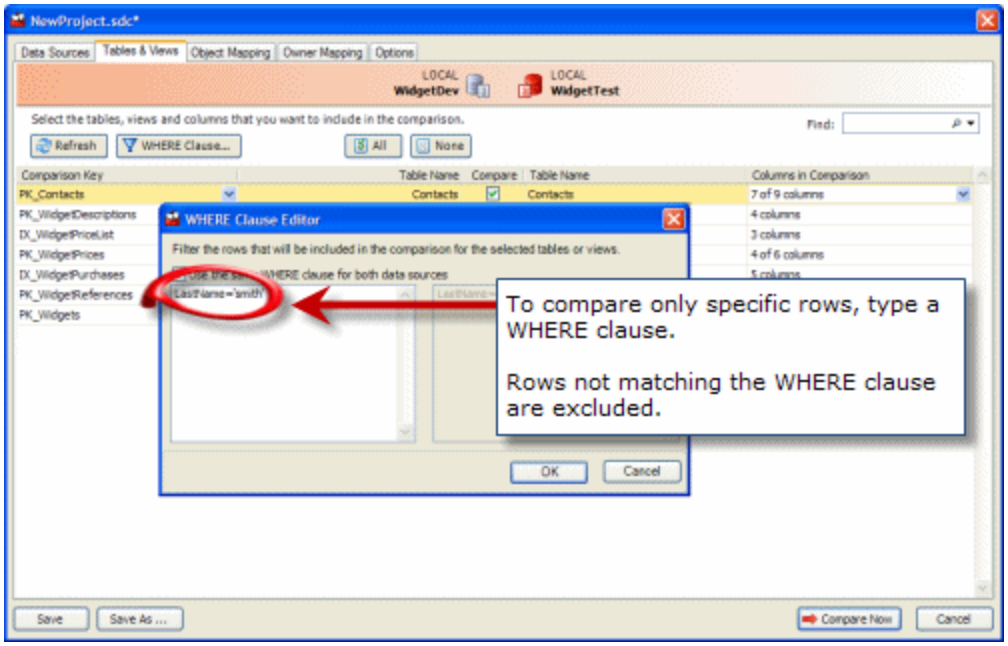
To exclude all the data in particular columns, click in the **Columns in Comparison** box for the associated table, and then clear the check boxes for the required columns:



Only the selected columns will be compared.

Row filtering (horizontal filtering)

To exclude specific rows, click **WHERE Clause**, and then type a T-SQL WHERE clause in the box:



You can apply the same WHERE clause to multiple tables, by highlighting the required tables before opening the **WHERE Clause Editor**.

You can apply a different WHERE clause for each data source by clearing the **Use the same WHERE clause for both data sources** check box.

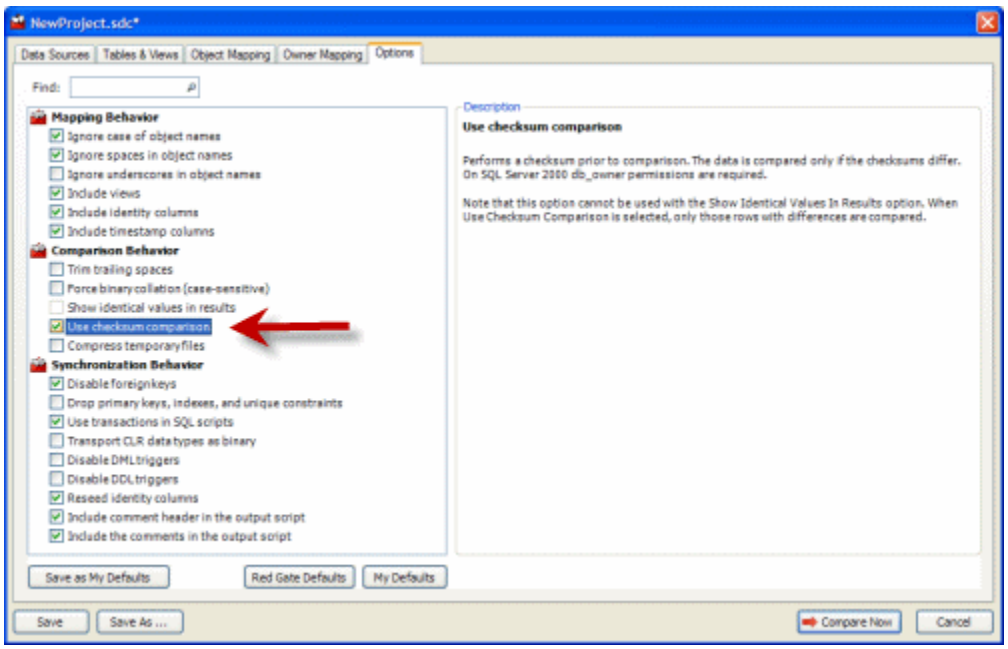
> You can only filter rows if the data source is a database. WHERE clauses are not available for backups and scripts folders.

Reviewing the options for comparison behavior

By default, SQL Data Compare includes identical values in the comparison results. If your data sources always contain similar data, on the **Options** tab of the **Project Configuration** dialog box, clear the **Show identical values in results** check box.

When you do this, SQL Data Compare does not store data that is identical, reducing the disk space that is required.

If you are comparing tables with large amounts of data that changes infrequently, select the **Use checksum comparison** check box. You must clear the **Show identical values in results** check box before you do this. If the checksums are equal, SQL Data Compare does not compare that table, and its data is not stored, reducing the disk space that is required.



If your table contains large data types such as text or images, these columns are not included in the checksum, and you are not recommended to use the **Use checksum comparison** option.

Reseed applying "incorrect" identity values

After deploying data using SQL Data Compare or SQL Packager, the next time data is inserted into one of the deployed tables, an error may result claiming that the automatically-generated value violates a primary key or unique constraint.

SQL Data Compare will reseed identity values automatically after deployment if the *Reseed Identity columns* project option is selected. It does this by gathering the current identity value from the source table and applying it to the target table. In this way, any identity columns with a primary key constraint will continue generating valid identity values.

There are a few circumstances, however, where the current identity value in the target database will become or remain invalid:

- A data update was taking place before the deployment or at the same time as the data comparison
- The direction of deployment was changed. In that case the databases need to be compared again because the identity value is gathered at registration-time
- There are no differences in the table, at that point SQL Data Compare does not reseed and the identity values remain different
- All records were deleted from the source table using a DELETE query. In this case the identity is not rolled backwards and SQL Data Compare will deploy the current identity, which may be higher than expected. Using TRUNCATE TABLE will roll the identity back to the base seed value.

SQL Data Compare showing differences in two identical databases

When comparing two identical databases, SQL Data Compare may show differences. This behavior occurs for one of the following reasons:

Trim trailing spaces option is not enabled

By default, SQL Data Compare includes spaces and other white space of text fields in the comparison. Likewise, comparing a fixed-length field in one database to a variable-length field will cause differences due to spaces. The **Trim trailing spaces** option will allow you to ignore these differences in the comparison of text data.

Force binary collation option is enabled

By default, SQL Data Compare uses the collation of a column to decide whether or not to compare data case-sensitively. Most SQL Server databases default to a case-insensitive collation, but this can be overridden using the **Force binary collation** option in SQL Data Compare.

Click **Edit Project** to open the Project Configuration dialog box. Select the **Options** tab. In the Comparison Behavior section ensure the **Trim trailing spaces** option is selected and the **Force binary collation** option is not selected.

Tables or views that could not be compared

Objects with rows that can not be matched automatically are shown in the **Tables or views that could not be compared** group. To match rows in the two data sources, SQL Data Compare requires:

- a *comparison key* for each table or view
- mappings between objects in the data sources

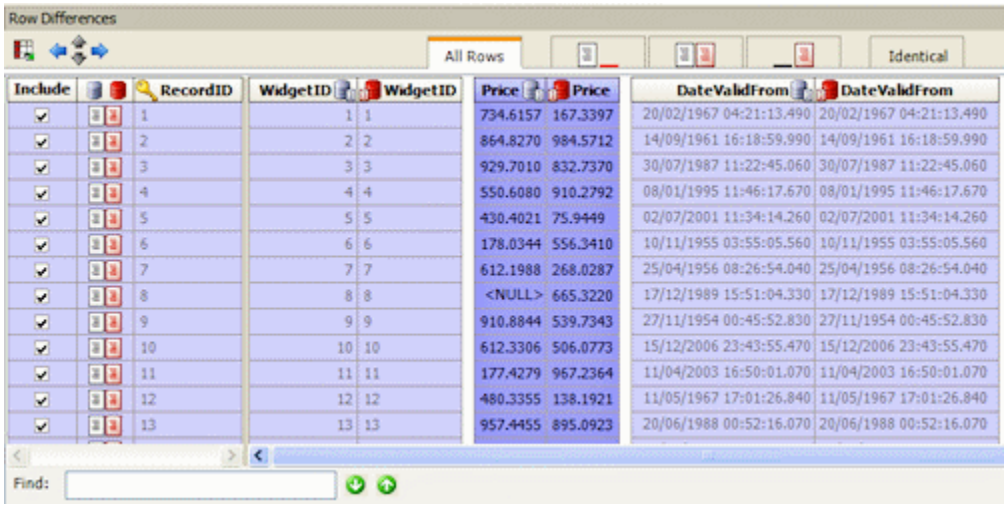
In some cases, SQL Data Compare is unable to map objects and select comparison keys automatically. For example, if there are substantial schema differences between the data sources. This article provides information on manually setting comparison keys.

> For more detailed information on setting mappings, see: Mapping data sources.

What is a comparison key?

When comparing data sources, SQL Data Compare looks for a matching primary key or other unique identifier in each data source to use as the *comparison key*. This enables matching rows to be identified, and their differences to be compared.

For example, the databases *WidgetSales* and *WidgetDeploy* both contain the table [dbo].[WidgetPrices]:



Since rows can be inserted and deleted, we can not be certain that the third row in *WidgetSales* is the same as the third row in*WidgetDeploy*. Simply comparing rows in the order in which they appear in the table can result in a meaningless comparison. Similarly, rows can not be matched based on their *Price* values - more than one widget can have the same price.

In [dbo].[WidgetPrices], *RecordID* is a primary key. No two widgets will have the same *RecordID*, and the rows are uniquely identified. Two matching *RecordID* values in *WidgetSales* and *WidgetDeploy* therefore represent the same piece of real world data.

Where there is no matching primary key, or other unique identifier you must set an appropriate comparison key.

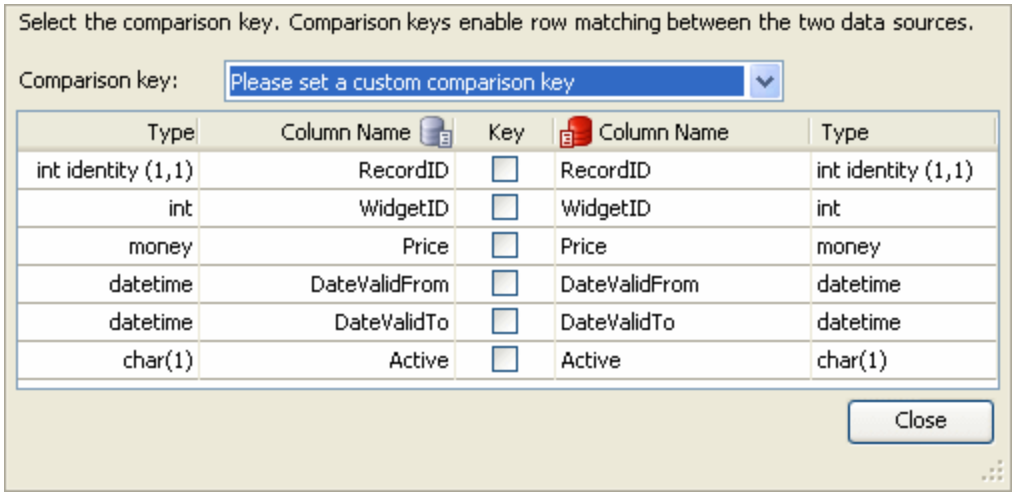
For example, if you know that two widgets with the same name are never added to the database on the same day, you can select the two columns *WidgetName* and *DateValidFrom* to form the comparison key. The rows in the tables can now be matched.

Setting comparison keys

SQL Data Compare automatically selects a comparison key, if your:

- tables contain a matching primary key, unique index, or unique constraint
- views contain a matching unique, clustered index

If SQL Data Compare is unable to identify a suitable comparison key for a table or view, *Not Set* is shown in the **Comparison Key** box, under **Tables & Views** on the Project Configuration dialog. You set the comparison key by clicking the appropriate **Comparison Key** box. A dialog box is displayed, for example:



Select the columns in the table or view that will comprise the comparison key by using the check boxes.

- A comparison key cannot include columns whose data type is image, ntext, nvarchar(max), sql_variant, text, varbinary(max), varchar(max), or xml.
- You cannot specify custom comparison keys if you are using a backup as a data source; however, you can select an alternative unique index or unique constraint.

For more information on setting up your comparison, see:

- Setting data sources
- Selecting tables and views

Troubleshooting comparison and deployment performance problems

SQL Data Compare can take a long time to compare databases depending on the environment. SQL Data Compare extracts the data from both databases and copies it to a temporary location on the hard disk of the local computer. Once it has copied the information from both databases to a temporary folder, it then compares the data and returns the results.

If the databases being compared are not on the same computer as SQL Data Compare, the data has to be copied over the network to the local machine. Depending on the speed, location of database, disk write speed, memory, size of database, and processor speed, this process can take a long time.

If you're experiencing unacceptable performance while deploying data using the SQL Data Compare GUI, you may want to save the deployment script to a file, copy it over to the target SQL Server and run it locally using a SQL query utility such as Management Studio. You can save the script by selecting 'save script' on the last stage of the Deployment Wizard instead of clicking Deploy Now.

You might also consider using the command line interface (Professional Edition only) to create a comparison and deployment command and schedule it to run at a time when the SQL server is experiencing least activity.

Using a filter on a column on related (joined) tables

It can be useful to filter records by a column's relationship with a related table - but the '"WHERE clause" option in SQL Data Compare Project Configuration dialog box does not offer direct access to columns in two related tables. In this article we examine how a view can be used instead.

Consider two tables, *Sales* and *LineItems*. Lets say a record in *LineItems* belongs to a district in the *Sales* table. I want to only compare *Lineitems* records that belong in a certain district *Suffolk*, and ignore all the other districts.

To run the example, create two databases, *TEST* and *TEST2*. Both have these databases have tables *LineItems*, *Sales* and a view *View_1*:

```
CREATE TABLE [dbo].[LineItems](
[ID] [smallint] NOT NULL,
[description] [nvarchar](50) NULL,
[SalesID] [smallint] NULL,
CONSTRAINT [PK_LineItems] PRIMARY KEY CLUSTERED
(
[ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]


CREATE TABLE [dbo].[Sales](
[SalesID] [smallint] NOT NULL,
[SalesDescription] [nvarchar](50) NULL,
CONSTRAINT [PK_Sales] PRIMARY KEY CLUSTERED
(
[SalesID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]

Create VIEW VIEW_1 AS
SELECT dbo.LineItems.ID, dbo.LineItems.description, dbo.LineItems.SalesID,
dbo.Sales.SalesDescription
FROM dbo.LineItems INNER JOIN
dbo.Sales ON dbo.LineItems.SalesID = dbo.Sales.SalesID
WHERE (dbo.Sales.SalesDescription = 'Suffolk')


--Populate TEST with the following:
INSERT INTO [dbo].[LineItems] ([ID], [description], [salesID]) VALUES (1, N'Fred ', 1)
INSERT INTO [dbo].[LineItems] ([ID], [description], [salesID]) VALUES (2, N'Chrs ', 1)
INSERT INTO [dbo].[LineItems] ([ID], [description], [salesID]) VALUES (3, N'Steve ',
1)
INSERT INTO [dbo].[LineItems] ([ID], [description], [salesID]) VALUES (4, N'Greg ', 2)
INSERT INTO [dbo].[LineItems] ([ID], [description], [salesID]) VALUES (5, N'Clare ',
2)
INSERT INTO [dbo].[LineItems] ([ID], [description], [salesID]) VALUES (6, N'David ',
2)
INSERT INTO [dbo].[LineItems] ([ID], [description], [salesID]) VALUES (7, N'Linda ',
3)

--And ...
INSERT INTO [dbo].[Sales] ([SalesID], [SalesDescription]) VALUES (1, N'Norfolk')
INSERT INTO [dbo].[Sales] ([SalesID], [SalesDescription]) VALUES (2, N'Suffolk')
INSERT INTO [dbo].[Sales] ([SalesID], [SalesDescription]) VALUES (3, N'Hampshire')
INSERT INTO [dbo].[Sales] ([SalesID], [SalesDescription]) VALUES (4, N'Essex')
```

Start SQL Data Compare and select the *Include Views* project option.

Sychronize TEST and TEST2. Now when clicking on VIEW_1 row the inserts created by the update script are:

```
INSERT INTO [dbo].[View_1] ([ID], [salesID], [SalesDescription], [description]) VALUES
(4, 2, N'Suffolk', N'Greg ')
INSERT INTO [dbo].[View_1] ([ID], [salesID], [SalesDescription], [description]) VALUES
(5, 2, N'Suffolk', N'Clare ')
INSERT INTO [dbo].[View_1] ([ID], [salesID], [SalesDescription], [description]) VALUES
(6, 2, N'Suffolk', N'David ')
```

We are limiting the update SQL script to only the *Suffolk*. In this way we can see how a simple view can control the comparison output by linking related tables.

Error messages

- NULL textptr passed to UPDATETEXT function when running deployment
- Running migration scripts using SqlCmd.exe
- This SQL Server has been optimized for x concurrent queries
- Troubleshooting System.OutOfMemoryException during comparison

NULL textptr passed to UPDATETEXT function when running deployment

When running a deployment script produced by SQL Data Compare against an empty database (identical schema but no data), this error may result when attempting to update or insert a row which contains a TEXT or NTEXT column:

```
NULL textptr (text, ntext, or image pointer) passed to UPDATETEXT function
```

Check to see if any WHERE clause is included in the project settings for the table being updated. SQL Data Compare updates text using the UPDATETEXT function, and in order to locate the correct row to update, it selects a TEXTPTR for the row. The problem occurs because the WHERE clause specified to Data Compare is used as the selection criteria for TEXTPTR.

For example, create a project comparing widgetdev to an identical copy of widgetdev's schema, only containing no data. Now apply a where clause to the widgetdescriptions table:

```
WidgetID IN (SELECT RecordID FROM Widgets)
```

After running the deployment wizard, part of the deployment script reads:

```
DECLARE @pv binary(16)
SELECT @pv=TEXTPTR([Picture]) FROM [dbo].[WidgetDescriptions] WHERE [WidgetID]=2 AND
(WidgetID IN (SELECT RecordID FROM Widgets))
```

If there is currently no data in the widgets table in the target database, the query will return NULL, resulting in the error.

The solution is to not use the same WHERE clause on both databases. The WHERE clause should only apply to the database on the left in order to work against a blank database.

Running migration scripts using SqlCmd.exe

Microsoft provides a command-line utility called SqlCmd.exe with SQL Server 2005 and later, but this utility will not run a SQL script over 500MB in size, and will return the following when this condition is encountered:

```
Sqlcmd: Error: Scripting error
```

The way around this is to switch off transactions while doing SQL Data Compare and obtain the script in the usual way. You then have to parse the script in an application, written in, for example, Visual Basic or Perl that will divide the queries into smaller batches. As you will not have any transactional integrity you will have to take a backup before you attempt to run the migration SQL script.

The following example contains VB script that injects a "GO" every 100 lines:

http://www.red-gate.com/MessageBoard/viewtopic.php?t=8109

Other workarounds might be to use a WHERE clause to restrict the amount of data being deployed, or to select fewer objects for the deployment and then run several deployments.

This SQL Server has been optimized for x concurrent queries

When comparing and/or deploying, the following error may be returned by SQL Server:

```
This SQL Server has been optimized for x concurrent queries. This limit has been exceeded by x queries and performance may be adversely affected.
```

This may make it seem as if SQL Compare or Data Compare is making many connections to the database, however, SQL Data Compare and SQL Compare will only make a single connection to each database it is comparing.

If you want to check to see exactly what connections are being made to SQL server, by Red Gate tools or other processes, you can check the current connections by running sp_who (see books online) in a query editor in the context of the master database. Using that, you should be able to see which connections are currently open and what is using them.

Troubleshooting System.OutOfMemoryException during comparison

This error means there isn't enough storage available for the data comparison.

SQL Data Compare has been tested on extremely large databases (up to terabytes in size), and is very scalable. There are two situations where an out of memory situation can occur:

There isn't enough temporary storage

Red Gate tools store temporary data in the %TMP% folder. By default, this folder is in your Windows profile folder (eg *C:\Users\<username>\Temp*). If you don't have much space on the hard drive, this folder may fill up, causing memory errors.

To fix this, you can have Red Gate tools use a different location for temporary files on a different hard drive with more space. For example, to use the folder *RGtemp* on drive D:

1. On the Start menu, right-click **Computer** and select **Properties**.
2. In the **Advanced** tab, click **Environment Variables**.
3. Click **New**.
4. In the **Variable name** field, type: *RGTEMP*
5. In the **Variable value** field, type: *D:\RGtemp*
6. Click **OK**.
7. Close Management Studio and standalone Red Gate tools.
8. Log out of Windows and log in again.

All Red Gate tools will use the folder you specified to save temporary data. You can test this by running a comparison in Data Compare (or running an operation in another Red Gate tool) and checking that files are added to the new location.

To undo this and have Red Gate tools use the default folder, delete the RGTEMP variable from the Environment Variables list, and follow steps 7 and 8 again.

Very large BLOBs are being compared

If your database makes heavy use of Binary Large Objects such as TEXT and IMAGE fields, each individual field from both databases needs to be loaded into memory at once. If these fields are large, you may exhaust your physical memory when trying to compare them. For instance, if you have got an individual BLOB column that is 1GB in size in one database and 1GB in the other database, 2GB of memory is needed to compare them. Unfortunately your only courses of action are to use a machine with more memory (a 64-bit machine may be needed to address more than 2GB of memory in a single process) or ignore these columns and deploy them manually.

To detect all BLOB columns in a database, these queries are useful:

```
SELECT o.[name] AS [Table Name],c.[name] AS [Column Name],
t.[name] AS [Column Type]
FROM sys.all_columns c
INNER JOIN sys.all_objects o
ON c.object_id = o.object_id
INNER JOIN sys.types t
ON c.system_type_id = t.system_type_id
WHERE c.system_type_id IN (35, 165, 99, 34, 173)
AND o.[name] NOT LIKE 'sys%'
AND o.[name] <> 'dtproperties'
AND o.[type] = 'U'
GO
```

Unexpected behavior / technical questions

- How much free hard disk space is required?
- How to force SQL Compare and SQL Data Compare to use an encrypted connection
- Minimum database permissions for SQL Data Compare
- Using Windows authentication logons between domains

How much free hard disk space is required?

You may encounter messages stating that you are out of disk space when using SQL Data Compare on larger databases.

As a rule of thumb you need 3-4 times the size of the database free disk space (this is for a local copy of the data from each database and space for the SQL Script creation). If your databases contain very similar data in both instances then we would suggest looking at the option **Show identical values in results** under **Comparison Options** on the Options tab as this will stop data compare storing any data which is identical in the two databases allowing the comparison to happen with a much smaller disk space footprint.

If your databases contain a large amount of different data we'd suggest splitting the table horizontally or vertically and performing the comparisons in parts. You can split the table horizontally by using the "WHERE filter" feature. On the **Tables and Views** tab right-click on the tables with a large amount of data in them and choose **Open WHERE clause editor**. This will open up a window where you can specify a WHERE clause and split the table into parts.

You can also do the equivalent vertically by clicking on the **Columns in Comparison** column next to a table on the **Tables and Views** tab and removing some of the columns.

If you're running out of space you can change the location that SQL Data Compare uses for its temporary directory:

1. Right click on **My Computer** and choose **Properties**.
2. On the **Advanced tab** click the **Environmental Variables** button.
3. Then under User or System variables click **New** and give the variable name as RGTEMP and the variable value as the path you want Red Gate SQL Compare and SQL Data Compare to use as their temp directory.

> SQL Data Compare 7 and later can compress temporary storage files, so won't require as much space. The space saved depends on the type of data being compared.

How to force SQL Compare and SQL Data Compare to use an encrypted connection

By default SQL Compare and SQL Data Compare do not have an option that forces the use of an encrypted connection when connecting to databases.

The work around is to specify the force encryption properties into the SQL Compare or SQL Data Compare connection string.

1. Start up SQL Compare or SQL Data Compare.
2. Create a new project or edit an existing project.
3. Go to Project Configuration -> Data Sources.
4. Enter the server name, and enter the following after the server to force an encrypted connection:
   For a Default Instance: <Server Name>;ENCRYPT=TRUE;TRUSTSERVERCERTIFICATE=TRUE
   For a Named Instance: <Server Name>\<Instance Name>;ENCRYPT=TRUE;TRUSTSERVERCERTIFICATE=TRUE
   If server is not running on the default SQL Port 1433:<Server Name>,<port number>;ENCRYPT=TRUE;TRUSTSERVERCERTIFICATE=TRUE or <Server Name>\<Instance Name>,<port number>;ENCRYPT=TRUE;TRUSTSERVERCERTIFICATE=TRUE

Minimum database permissions for SQL Data Compare

SQL Data Compare can compare and deploy data in tables and views of a SQL Server database. In order to do this effectively, the user connecting to the database needs access to the schema (data definitions) and the data objects of a database, and may require additional privileges if the data is being deployed with or without the use of some of the options available for controlling the deployment behavior.

In SQL Server 2000, it is only necessary to be a member of the PUBLIC role in order to gather information about the database schema. To compare data, SELECT permissions are required on all objects whose data is being compared. The easiest way to grant SELECT permissions on all objects is to add the SQL Data Compare user to the database's db_datareader built-in role. To update the database being compared, it is necessary to have INSERT and UPDATE rights granted in addition to SELECT. The easiest way to grant these permissions for all objects in the database is to add the user to the db_datawriter user role.

There is one special circumstance that requires elevated privileges, and that is when the 'disable foreign keys', 'Drop primary keys, indexes, and unique constraints', 'Disable DML triggers', 'Disable DDL triggers', or 'Reseed identity columns' options are used. These actions require access Data Definition Language modification, so it may be necessary to add the user to the db_ddladmin role or DBO role to allow the deployment to succeed if any of the specified options are configured.

Under SQL Server 2005 and 2008, the ability to view schema may be restricted for any users with PUBLIC role membership if the VIEW DEFINITION right has been revoked. One must have VIEW DEFINITION permission granted in order to see the definition of objects in the database.

Using Windows authentication logons between domains

SQL Compare and Data Compare make use of the built-in authentication mechanisms provided by Windows and are therefore limited to the Active Directory domain security model. If you wanted to compare schema or data across domains, for example, you could use SQL authentication accounts like SA. When choosing Windows Integrated authentication, the current desktop user's credentials are used to authenticate to the SQL Server so you never get the opportunity to enter a Windows username and password. If the SQL Server's Windows domain is different than the user logged into the desktop, then the SQL Server's Active Directory server would normally be unable to verify the identity of the logged-in user on behalf of the SQL Server.

If it's absolutely necessary to use Windows accounts to log into the SQL Server, for instance if the SQL Server is not configured to allow SQL logins and passwords, there are two possible workarounds to allow login credentials from one domain to be used in another. The first option could be to create a trust between the two domains. It's unlikely a network administrator would do this for you unless you're really, really nice to them, then maybe not even then, because of the security considerations and management overhead of a domain trust. The second option is called matching accounts - if you create a local Windows user on the SQL Server computer and on the local workstation, making sure to grant this account access to the SQL Server, and these two accounts share the same username and password, logging into the local workstation and connecting SQL Compare or Data Compare using Windows authentication will work.

As long as the SQL Server allows Windows accounts to authenticate using NTLM authentication, all that is necessary to allow access to the SQL Server is a token constructed using the Windows user name and password. If the usernames and passwords of two local Windows accounts share these attributes, then the hash will be accepted by both computers and successful authentication will occur.

Logging and log files

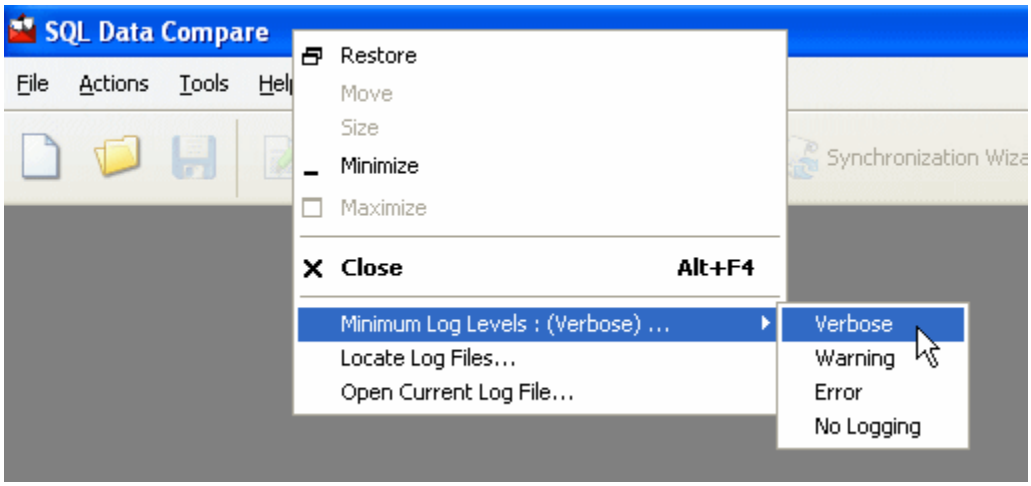
Log files collect information about the application while you are using it. These files are useful to us if you have encountered a problem.

By default, logging is disabled and no log files are stored.

Enabling logging

To enable logging, select the log level you require:

- Right-click the application title bar, click **Minimum Log Levels**, and then click the required log level.



Select a minimum log level based on how much information you need to be reported:

Verbose	Reports all messages in the log file.
Warning	Reports warning and error messages. For example, a warning message might report a handled exception, or a problem which does not prevent you from using the application.
Error	Reports serious and fatal errors. For example, an error message might report a failed operation.
No Logging	Disables logging.

Note that the selected log level may affect performance. Verbose logging reports all messages, and so writes the most information to disk and produces the largest log files.

If a problem has been resolved and you no longer require logging, you are recommended to select **No Logging**.

Locating the log files

To open the folder where the log files are stored, click **Locate Log Files**. By default the log files are located in:

%ALLUSERSPROFILE%\Application Data\Red Gate\Logs

To view the current log file in your default text editor, click **Open Current Log File**.

If the minimum log level is set to **No Logging**, you cannot locate or open log files from within the application.

Release notes and other versions

Version 11.4 (current)	December 21st, 2015	Release notes	Documentation
Version 11.3	September 1st, 2015	Release notes	
Version 11.2	May 7th, 2015	Release notes	
Version 11.1	January 29th, 2015	Release notes	
Version 11.0	October 6th, 2014	Release notes	
Version 10.7	April 23rd, 2014	Release notes	Documentation
Version 10.4	June 3rd, 2013	Release notes	
Version 10.0	March 2nd, 2012	Release notes	
Version 9.0	March 16th, 2011	Release notes	Documentation
Version 8.0	July 16th, 2009	Release notes	Documentation
Version 7	July - Sep 2008	Release notes	Documentation (PDF)

If you need to install an old version of SQL Data Compare, go to Download old versions of products.

SQL Data Compare 10.7 release notes

April 23rd, 2014

New features

- Supports connecting to SQL Server 2014 databases

SQL Data Compare 10.4 release notes

SQL Data Compare 10.4 release notes

January 3rd, 2013

New features

- This release of SQL Data Compare contains significant performance improvements - up to 50% faster on script folder registration, 60% faster on the registration and a 50% reduction on the memory usage for some LOBs.

Fixes

- Changed default script encoding to UTF-8 with preamble - SQLCMD didn't recognise some UTF-8 scripts otherwise
- Various small UI fixes (un-greying some text, fixing some cut-off / overprinted text, improved some option descriptions)
- Improved UI stability (fixed several race conditions that occasionally produced null reference exceptions or invalid operation exceptions in the UI)
- SQL Data Compare can now load projects with source-controlled databases saved by SQL Compare correctly
- Connection details now preserved when launching SQL Data Compare from SQL Compare
- Fixed some KeyNotFoundExceptions caused by UI state getting out of sync

SQL Data Compare 10.0 release notes

March 2nd, 2012

New features

- Select a version of your database from source control.
- Recently used server list and recently accessed repository list.
- Fast Deploy - two-click deploy from the project list for common deployments.

Major feature requests implemented / bugs fixed

- SDC-113 - New option to compare null values as equal to empty strings.
- SDC-206 - New option to compare floats that only have unsynchronizable differences as equal.
- SDC-1137 - Viewer now correctly displays full length of float values.
- SDC-1293 - int datatypes can now successfully map to varchar and some other string types.

Minor bug fixes

- SDC-1304 - Command line help can now be written to a file with /out.
- SDC-1290 - Handling of unusual unicode values in varchar(max) blobs improved.
- SDC-1301 - UTF-8 script encoding option now no longer writes preamble by default.

SQL Data Compare 9.0 release notes

March 16th, 2011

New features

- SQL Azure support
  For more information, see Deploying to SQL Azure.

Fixes

- SDC-1184 Comments in script for source/target incorrect if source is a scripts folder.
- SDC-1180 Error parsing .sdc file from command line.
- SDC-1152 Case insensitive compare DB<>Scripts shows case diff, but doesn't script.
- SDC-1145 Script folder comparison sensitive to case within schemas even when this should be ignored.
- SDC-1130 Error during sync: An explicit DROP INDEX is not allowed on index <index>. It is being used for FOREIGN KEY constraint enforcement.
- SDC-1043 Change reseed behavior.
- SDC-909 Partial updates are not supported by filestream.
- SDC-1118 Deselected rows are included in sync.
- SDC-1110 Split Script with GO's when not using transactions.
- SDC-1034 Request for the split transaction feature to be available in the command line (see SqlDataCompare /? /v for updated options and switches).
- SDC-1024 Top row of results unchecks after sync.
- CSD-127 CompareDatabases method changes the thread's CurrentCulture.
- SDC-837 Random column order when comparing a view.
- SC-4684 Launch SQL Data Compare is disabled (same fix for launching SQL Compare from SQL Data Compare).
- SDC-1132 Command line + script folder fails.
- SDC-1136 Sync fails on objects with unicode characters.
- SDC-1142 Command line distribution behavior differs from 8.0 to 8.1.

SQL Data Compare 8.0 release notes

July 16th, 2009

SQL Data Compare Version 8.0 is a major release introducing support for static data in scripts folders, user interface and command line enhancements, and bug fixes.

New features

SQL Data Compare 8.0 introduces the following features:

- **The ability to store static data in scripts folders**

  SQL Data Compare 8 allows you to store data scripts in a scripts folder, making it easier to keep databases under source control.
- **The ability to share projects with SQL Compare**

  To open a SQL Data Compare project in SQL Compare, on the **Projects** dialog box, select a project, right click, and click **Launch in SQL Compare**.

  Alternatively, to open the current project in SQL Compare, click the SQL Compare icon on the toolbar of the main window.

  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.
- **Multiple transactions in the synchronization script**

  In previous versions, the synchronization can be enclosed in a single transaction. This allows rollback, but can result in poor performance for large databases. In SQL Data Compare 8.0 you have the option to use multiple transactions. You can specify the amount of data included in each transaction using the Application Options.
- **User interface refinements**

  SQL Data Compare 8.0 includes many of the user interface improvements made for SQL Compare 8, bringing the workflow of the two products closer together.
- **Command line changes**

  The command line interface now supports column-level inclusion and exclusion. There are changes to the behavior of some existing switches and options.
- **Stability and performance improvements**

  SQL Data Compare 8.0 addresses some known stability issues. There are now fewer *Out of memory* errors when comparing large databases.

Working with scripts folders

A scripts folder is a set of SQL scripts representing a database's schema and data.

For the schema, each object's creation script is saved as a file.

For the data, an insert script is created for each table. A metadata file that allows SQL Data Compare to compare the scripts is also created. The metadata file has the extension *.sdcs*

When using scripts folders, note that:

- Scripts folders created using SQL Compare version 8.0 and below are not supported.
  To accommodate static data, the structure of scripts folders has changed. SQL Data Compare can only be used with script folders it creates, or that are created by SQL Compare 8.1.

  If an invalid folder is used, the message *Scripts folder metadata file incomplete* is displayed, and you are recommended to re-create the schema scripts using SQL Data Compare.
- Views are not supported.
  Scripts folders can only represent data in tables.
- filtering with a WHERE clause is partially supported.
  For scripts folders, on the **Tables and Views** tab of the **Project Configuration** dialog box, you can use a WHERE clause to restrict which rows are compared. However, only one of the data sources can be a scripts folder. If you are comparing two scripts folders, a WHERE clause cannot be used.
- Scripts folders must contain schema scripts.
  SQL Data Compare cannot create, compare, or synchronize data scripts if the associated schema scripts are not present in the folder. The schema scripts must be created by SQL Data Compare or SQL Compare 8.1.

- SQL Data Compare does not support user-created sql scripts.
  SQL Data Compare can only compare and synchronize a scripts folder if the files in that folder were created by SQL Data Compare, or SQL Compare 8.1
- You are recommended not to edit a SQL Data Compare script file.
  If you edit the contents of a scripts folder, the comparison and synchronization may fail or produce unexpected results.
- You are recommended to use SQL Data Compare to synchronize databases with scripts folders.
  If you run the scripts manually, they may fail or produce unexpected results.
- You are recommended not to delete or modify the *.sdcs* metadata files created by SQL Data Compare.

  These files contain index information that enables SQL Data Compare to compare and synchronize scripts folders. You will encounter poor performance if valid metadata files are not present in the scripts folder.

Other known issues

- 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.
- Performance
  If you are using a scripts folder as a data source, and the folder contains a lot of data, the comparison may be slow.
- Installing in a 64 bit environment
  There is a known issue that, in some cases, when installing in a 64 bit environment, the installer displays incorrect progress messages. However, the installation proceeds normally.

  If an earlier version of SQL Data Compare is installed, and you install version 8.0, the two versions are installed side by side. This is confirmed by progress messages in the installer. However, in a 64 bit environment, a successful side by side installation mistakenly shows progress messages for a clean install. In these cases, SQL Data Compare 8 is installed, and previous versions are unaffected.
- Option to launch SQL Compare disabled
  The toolbar of the **Comparison Results** screen and the right click menu of the **Projects** dialog box show the option to launch a project using SQL Compare. This functionality is only compatible with SQL Compare version 8.1, and later.
- SQL Compare 8.0 shows parser errors for data scripts
  If you are using a SQL Data Compare 8 scripts folder with SQL Compare 8.0, the **Error Parsing Scripts** dialog box is displayed each time you compare the data sources.

  This is because SQL Compare 8.0 does not parse the INSERT statements used to store static data. However, SQL Data Compare scripts folders are supported by SQL Compare 8.1

  This potentially leads to a large number of errors. In most cases, these errors can safely be ignored.

If you encounter any issues not documented here, or if these issues have unexpected consequences, you are recommended to contact support, or post a detailed description of the error you encountered in the SQL Data Compare 8 support forum

Command line syntax changes

In SQL Data Compare 8, there are changes to the names and functions of some command line switches and options, as well as their aliases.

The command line syntax of previous versions of SQL Data Compare is considered deprecated, but continues to be supported.

For example, in SQL Data Compare 7, the alias for */BackupSet1* was */bs1*. In SQL Data Compare 8, the alias is now */bks1*. You can continue to use */bs1* in SQL Data Compare 8, but a message is displayed informing you of the new alias.

Deprecated command line syntax will cease to be supported at a future release.

SQL Data Compare 7 release notes

SQL Data Compare 7.1 - Sep 2008

Version 7.1 is a minor release of SQL Data Compare. Along with a number of bug fixes

It is now possible to trial the SQL Data Compare command line in continuous integration environments.

SQL Data Compare 7.0 - July 2008

Support for SQL Server 2008:

- SQL Data Compare 7.0 supports SQL Server 2008 based on the currently available release candidate. SQL Server 2008 introduces new data types (spatial data types, temporal data types, and hierarchy data type), sparse columns, and new collations, all of which are supported by SQL Data Compare 7.0.
- SQL Data Compare 7.0 Professional also supports the new native and compressed backup formats in SQL Server 2008, in addition to the backup files of SQL Server 2000, SQL Server 2005, and Red Gate's SQL Backup.

Example GS

```
-- Drop if exist
if @@trancount > 0 begin rollback end
use tempdb
go
IF EXISTS (SELECT name FROM master.dbo.sysdatabases WHERE name = N'WidgetDev')
    DROP DATABASE WidgetDev
IF EXISTS (SELECT name FROM master.dbo.sysdatabases WHERE name = N'WidgetLive')
    DROP DATABASE WidgetLive
IF EXISTS (SELECT name FROM master.dbo.sysdatabases WHERE name = N'WidgetTest')
 DROP DATABASE WidgetTest

--Create the databases and structure
CREATE DATABASE WidgetDev
GO

USE WidgetDev
GO

CREATE TABLE [dbo].[WidgetDescriptions] (
 [WidgetID] [int] NOT NULL ,
 [ShortDescription] nvarchar(2000) NULL ,
 [Description] [text] ,
 [Picture] [image] NULL
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO

CREATE TABLE [dbo].[WidgetPrices] (
 [RecordID] [int] IDENTITY (1, 1) NOT NULL ,
 [WidgetID] [int] NULL ,
 [Price] [money] NULL ,
 [ValidFrom] [datetime] NULL ,
 [ValidTo] [datetime] NULL ,
 [Active] [char] (1) NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[WidgetReferences] (
 [WidgetID] [int] NOT NULL ,
 [Reference] [varchar] (50) NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[Widgets] (
 [RecordID] [int] IDENTITY (1, 1) NOT NULL ,
 [Description] [varchar] (50) NULL ,
 [SKU] [varchar] (20) NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[WidgetPurchases] (
 [PurchaseID] [int] IDENTITY (1, 1) NOT NULL ,
 [WidgetPriceID] [int] NOT NULL ,
 [Quantity] [int] NOT NULL DEFAULT (1) ,
 [InvoiceNumber] [nvarchar](20) NULL ,
 [Date] [datetime] NOT NULL DEFAULT (getdate())
```

```
)
GO

CREATE TABLE [dbo].[Contacts] (
 [ID] [int] IDENTITY (1, 1) NOT NULL ,
 [Name] [nvarchar](100) NULL ,
 [PhoneWork] [nvarchar](25) NULL ,
 [PhoneMobile] [nvarchar](25) NULL ,
 [Address1] [nvarchar](128) NULL ,
 [Address2] [nvarchar](128) NULL ,
 [Address3] [nvarchar](128) NULL ,
 [JoiningDate] [datetime] NULL DEFAULT (getdate()),
 [Email] [nvarchar](256) NULL
)
GO

ALTER TABLE [dbo].[WidgetDescriptions] WITH NOCHECK ADD
 CONSTRAINT [PK_WidgetDescriptions] PRIMARY KEY  CLUSTERED
 (
  [WidgetID]
 )  ON [PRIMARY]
GO

ALTER TABLE [dbo].[WidgetPrices] WITH NOCHECK ADD
 CONSTRAINT [DF_WidgetPrices_ValidFrom] DEFAULT (getdate()) FOR [ValidFrom],
 CONSTRAINT [DF_WidgetPrices_Active] DEFAULT ('N') FOR [Active],
 CONSTRAINT [PK_WidgetPrices] PRIMARY KEY  NONCLUSTERED
 (
  [RecordID]
 )  ON [PRIMARY]
GO

ALTER TABLE [dbo].[WidgetReferences] WITH NOCHECK ADD
 CONSTRAINT [PK_WidgetReferences] PRIMARY KEY  NONCLUSTERED
 (
  [WidgetID]
 )  ON [PRIMARY]
GO

ALTER TABLE [dbo].[Widgets] WITH NOCHECK ADD
 CONSTRAINT [PK_Widgets] PRIMARY KEY  NONCLUSTERED
 (
  [RecordID]
 )  ON [PRIMARY]
GO

ALTER TABLE [dbo].[Contacts] WITH NOCHECK ADD
 CONSTRAINT [PK_Contacts] PRIMARY KEY CLUSTERED
 (
  [ID]
 )
GO

 CREATE  INDEX [IX_WidgetPrices] ON [dbo].[WidgetPrices]([WidgetID]) ON [PRIMARY]
GO

 CREATE  INDEX [IX_WidgetPrices_1] ON [dbo].[WidgetPrices]([ValidFrom]) ON [PRIMARY]
GO
```

```
 CREATE  INDEX [IX_WidgetPrices_2] ON [dbo].[WidgetPrices]([ValidTo]) ON [PRIMARY]
GO

 CREATE UNIQUE CLUSTERED INDEX [IX_WidgetPurchases] ON
[dbo].[WidgetPurchases]([PurchaseID]) ON [PRIMARY]
GO

-- Create indexed view

CREATE VIEW dbo.WidgetPriceList
WITH SCHEMABINDING
AS
SELECT     dbo.Widgets.RecordID, dbo.Widgets.Description AS Widget,
dbo.WidgetPrices.Price
FROM       dbo.Widgets INNER JOIN
           dbo.WidgetPrices ON dbo.Widgets.RecordID = dbo.WidgetPrices.RecordID
GO

CREATE UNIQUE CLUSTERED INDEX [IX_WidgetPriceList] ON [dbo].[WidgetPriceList]
([RecordID])
GO



-- ********************************** WidgetLive begins here
*********************************
CREATE DATABASE WidgetLive
GO

USE WidgetLive
GO

CREATE TABLE [dbo].[WidgetDescriptions] (
 [WidgetID] [int] NOT NULL ,
 [ShortDescription] nvarchar(2000) NULL ,
 [Description] [text] NULL ,
 [Picture] [image] NULL
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO

CREATE TABLE [dbo].[WidgetPrices] (
 [RecordID] [int] IDENTITY (1, 1) NOT NULL ,
 [WidgetID] [int] NULL ,
 [Price] [money] NULL ,
 [DateValidFrom] [datetime] NULL ,
 [DateValidTo] [datetime] NULL ,
 [Active] [char] (1) NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[WidgetReferences] (
 [WidgetID] [int] NOT NULL ,
 [Reference] [varchar] (50) NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[Widgets] (
 [RecordID] [int] IDENTITY (1, 1) NOT NULL ,
 [Description] [varchar] (50) NULL ,
```

```
 [SKU] [varchar] (20) NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[WidgetPurchases] (
 [PurchaseID] [int] IDENTITY (1, 1) NOT NULL ,
 [WidgetPriceID] [int] NOT NULL ,
 [Quantity] [int] NOT NULL DEFAULT (1) ,
 [InvoiceNumber] [nvarchar] (20) NULL ,
 [Date] [datetime] NOT NULL DEFAULT (getdate())
)
GO

CREATE TABLE [dbo].[Contacts] (
 [ID] [int] IDENTITY (1, 1) NOT NULL ,
 [Name] [nvarchar](100) NULL ,
 [PhoneWork] [nvarchar](25) NULL ,
 [PhoneMobile] [nvarchar](25) NULL ,
 [Address1] [nvarchar](128) NULL ,
 [Address2] [nvarchar](128) NULL ,
 [Address3] [nvarchar](128) NULL ,
 [JoiningDate] [datetime] NULL DEFAULT (getdate()),
 [Email] [nvarchar](256) NULL
)
GO

ALTER TABLE [dbo].[WidgetDescriptions] WITH NOCHECK ADD
 CONSTRAINT [PK_WidgetDescriptions] PRIMARY KEY  CLUSTERED
 (
  [WidgetID]
 )  ON [PRIMARY]
GO

ALTER TABLE [dbo].[WidgetPrices] WITH NOCHECK ADD
 CONSTRAINT [DF_WidgetPrices_DateValidFrom] DEFAULT (getdate()) FOR [DateValidFrom],
 CONSTRAINT [DF_WidgetPrices_Active] DEFAULT ('N') FOR [Active],
 CONSTRAINT [PK_WidgetPrices] PRIMARY KEY  NONCLUSTERED
 (
  [RecordID]
 )  ON [PRIMARY]
GO

ALTER TABLE [dbo].[WidgetReferences] WITH NOCHECK ADD
 CONSTRAINT [PK_WidgetReferences] PRIMARY KEY  NONCLUSTERED
 (
  [WidgetID]
 )  ON [PRIMARY]
GO

ALTER TABLE [dbo].[Widgets] WITH NOCHECK ADD
 CONSTRAINT [PK_Widgets] PRIMARY KEY  NONCLUSTERED
 (
  [RecordID]
 )  ON [PRIMARY]
GO

ALTER TABLE [dbo].[Contacts] WITH NOCHECK ADD
 CONSTRAINT [PK_Contacts] PRIMARY KEY CLUSTERED
 (
```

```
  [ID]
 )
GO

 CREATE  INDEX [IX_WidgetPrices] ON [dbo].[WidgetPrices]([WidgetID]) ON [PRIMARY]
GO

 CREATE  INDEX [IX_WidgetPrices_1] ON [dbo].[WidgetPrices]([DateValidFrom]) ON
[PRIMARY]
GO

 CREATE  INDEX [IX_WidgetPrices_2] ON [dbo].[WidgetPrices]([DateValidTo]) ON [PRIMARY]
GO

 CREATE UNIQUE CLUSTERED INDEX [IX_WidgetPurchases] ON
[dbo].[WidgetPurchases]([PurchaseID]) ON [PRIMARY]
GO

-- Create indexed view

CREATE VIEW dbo.WidgetPriceList
WITH SCHEMABINDING
AS
SELECT     dbo.Widgets.RecordID, dbo.Widgets.Description AS Widget,
dbo.WidgetPrices.Price
FROM       dbo.Widgets INNER JOIN
           dbo.WidgetPrices ON dbo.Widgets.RecordID = dbo.WidgetPrices.RecordID
GO

CREATE UNIQUE CLUSTERED INDEX [IX_WidgetPriceList] ON [dbo].[WidgetPriceList]
([RecordID])
GO

-- ********************************** WidgetTest begins here
*********************************
CREATE DATABASE WidgetTest
GO

USE WidgetTest
GO

CREATE TABLE [dbo].[WidgetDescriptions] (
 [WidgetID] [int] NOT NULL ,
 [ShortDescription] nvarchar(2000) NULL ,
 [Description] [text] NULL ,
 [Picture] [image] NULL
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO

CREATE TABLE [dbo].[WidgetPrices] (
 [RecordID] [int] IDENTITY (1, 1) NOT NULL ,
 [WidgetID] [int] NULL ,
 [Price] [money] NULL ,
 [DateValidFrom] [datetime] NULL ,
 [DateValidTo] [datetime] NULL ,
 [Active] [char] (1) NULL
) ON [PRIMARY]
GO
```

```
CREATE TABLE [dbo].[WidgetReferences] (
 [WidgetID] [int] NOT NULL ,
 [Reference] [varchar] (50) NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[Widgets] (
 [RecordID] [int] IDENTITY (1, 1) NOT NULL ,
 [Description] [varchar] (50) NULL ,
 [SKU] [varchar] (20) NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[WidgetPurchases] (
 [PurchaseID] [int] IDENTITY (1, 1) NOT NULL ,
 [WidgetPriceID] [int] NOT NULL ,
 [Quantity] [int] NOT NULL DEFAULT (1) ,
 [InvoiceNumber] [nvarchar] (20) NULL ,
 [Date] [datetime] NOT NULL DEFAULT (getdate())
)
GO

CREATE TABLE [dbo].[Contacts] (
 [ID] [int] IDENTITY (1, 1) NOT NULL ,
 [Name] [nvarchar](100) NULL ,
 [PhoneWork] [nvarchar](25) NULL ,
 [PhoneMobile] [nvarchar](25) NULL ,
 [Address1] [nvarchar](128) NULL ,
 [Address2] [nvarchar](128) NULL ,
 [Address3] [nvarchar](128) NULL ,
 [JoiningDate] [datetime] NULL DEFAULT (getdate()),
 [Email] [nvarchar](256) NULL
)
GO

ALTER TABLE [dbo].[WidgetDescriptions] WITH NOCHECK ADD
 CONSTRAINT [PK_WidgetDescriptions] PRIMARY KEY  CLUSTERED
 (
  [WidgetID]
 )  ON [PRIMARY]
GO

ALTER TABLE [dbo].[WidgetPrices] WITH NOCHECK ADD
 CONSTRAINT [DF_WidgetPrices_DateValidFrom] DEFAULT (getdate()) FOR [DateValidFrom],
 CONSTRAINT [DF_WidgetPrices_Active] DEFAULT ('N') FOR [Active],
 CONSTRAINT [PK_WidgetPrices] PRIMARY KEY  NONCLUSTERED
 (
  [RecordID]
 )  ON [PRIMARY]
GO

ALTER TABLE [dbo].[WidgetReferences] WITH NOCHECK ADD
 CONSTRAINT [PK_WidgetReferences] PRIMARY KEY  NONCLUSTERED
 (
  [WidgetID]
 )  ON [PRIMARY]
GO

ALTER TABLE [dbo].[Widgets] WITH NOCHECK ADD
```

```
 CONSTRAINT [PK_Widgets] PRIMARY KEY  NONCLUSTERED
 (
  [RecordID]
 )  ON [PRIMARY]
GO

ALTER TABLE [dbo].[Contacts] WITH NOCHECK ADD
 CONSTRAINT [PK_Contacts] PRIMARY KEY CLUSTERED
 (
  [ID]
 )
GO

 CREATE  INDEX [IX_WidgetPrices] ON [dbo].[WidgetPrices]([WidgetID]) ON [PRIMARY]
GO

 CREATE  INDEX [IX_WidgetPrices_1] ON [dbo].[WidgetPrices]([DateValidFrom]) ON
[PRIMARY]
GO

 CREATE  INDEX [IX_WidgetPrices_2] ON [dbo].[WidgetPrices]([DateValidTo]) ON [PRIMARY]
GO

 CREATE UNIQUE CLUSTERED INDEX [IX_WidgetPurchases] ON
[dbo].[WidgetPurchases]([PurchaseID]) ON [PRIMARY]
GO

-- Create indexed view

CREATE VIEW dbo.WidgetPriceList
WITH SCHEMABINDING
AS
SELECT     dbo.Widgets.RecordID, dbo.Widgets.Description AS Widget,
dbo.WidgetPrices.Price
FROM       dbo.Widgets INNER JOIN
           dbo.WidgetPrices ON dbo.Widgets.RecordID = dbo.WidgetPrices.RecordID
GO

CREATE UNIQUE CLUSTERED INDEX [IX_WidgetPriceList] ON [dbo].[WidgetPriceList]
([RecordID])
GO




-- ************************************* Insert data
*************************************

USE WidgetDev
GO

BEGIN TRANSACTION
SET IDENTITY_INSERT [dbo].[Widgets] ON
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (1, 'Red
widget', 'RW')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (2, 'Extended
widget2', 'EW')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (4,
'Grow-your-own widget', 'GW')
```

```
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (5, 'Widget
access kit', 'AW')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (6, 'Test kit',
'TK')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (7, 'Exploded
widget diagram', 'ED')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (8, 'New
widget', 'NW')
SET IDENTITY_INSERT [dbo].[Widgets] OFF
INSERT INTO [dbo].[WidgetReferences] ([WidgetID], [Reference]) VALUES (1, 'Widget 1')
INSERT INTO [dbo].[WidgetReferences] ([WidgetID], [Reference]) VALUES (2, 'Widget 2')
SET IDENTITY_INSERT [dbo].[WidgetPrices] ON
INSERT INTO [dbo].[WidgetPrices] ([RecordID], [WidgetID], [Price], [ValidFrom],
[ValidTo], [Active]) VALUES (1, 1, 100, '2000-01-01 00:00:00.000', '2002-01-01
00:00:00.000', 'Y')
INSERT INTO [dbo].[WidgetPrices] ([RecordID], [WidgetID], [Price], [ValidFrom],
[ValidTo], [Active]) VALUES (2, 2, 50, '2000-01-01 00:00:00.000', '2002-01-01
00:00:00.000', 'Y')
INSERT INTO [dbo].[WidgetPrices] ([RecordID], [WidgetID], [Price], [ValidFrom],
[ValidTo], [Active]) VALUES (3, 4, 25, '2000-01-01 00:00:00.000', '2002-01-01
00:00:00.000', 'Y')
INSERT INTO [dbo].[WidgetPrices] ([RecordID], [WidgetID], [Price], [ValidFrom],
[ValidTo], [Active]) VALUES (4, 1, 110, '2002-01-01 00:00:00.000', '2003-01-01
00:00:00.000', 'N')
INSERT INTO [dbo].[WidgetPrices] ([RecordID], [WidgetID], [Price], [ValidFrom],
[ValidTo], [Active]) VALUES (5, 2, 55, '2002-01-01 00:00:00.000', '2003-01-01
00:00:00.000', 'N')
INSERT INTO [dbo].[WidgetPrices] ([RecordID], [WidgetID], [Price], [ValidFrom],
[ValidTo], [Active]) VALUES (6, 4, 30, '2002-01-01 00:00:00.000', '2003-01-01
00:00:00.000', 'N')
SET IDENTITY_INSERT [dbo].[WidgetPrices] OFF
INSERT INTO [dbo].[WidgetDescriptions] ([WidgetID], [ShortDescription], [Description],
[Picture]) VALUES (1, N'A widget', N'<description
xmlns="http://www.red-gate.com/widgets/WidgetDescriptionSchema">
  <descriptiveText>A cunning widget</descriptiveText>
  <manufacturer>Acme Inc</manufacturer>
  <countryOfOrigin>UK</countryOfOrigin>
  <size length="12" width="4" height="7" />
</description>',
0x474946383961db00d800e643000000000698495465863cccccc010f1b003366000b143333336666669999
99ffffff032d51808080021e3600333334424aafafaf01162884a5bae0e0e07b9aae586e7c111618669999
1a21252020201f1f1f4f4f4fb0b0b0dfdfdf4f636f7f7f7f6079883d4d57505050021a2f00003303294ac6
d8e399cccc84a9c1b7cedc739bb61e517de9f0f494b5cad4e1ea336699e2ebf15a85a5517d9f7b878d3651
65144876b0c9d8333366afb3b613395a0916204d5d678fa3af2d4252bed3df7f8386d0dae3cddce68db1c7
ffffff000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000021f90401000043002c00000000db00d8000007ff804382838485868788898a8b
8c8d8e8f43090708039096979899 9a9b9c9d9e890900a20019081c9fa8a9aaabacad8708a21a1aa3a22209
0aaeb9babbbcabb000070a0a031f07b4c00913bdcbcccdce0a22a21bc2d40a1d091bc7a595ceddedf9a0a
c60008d5e6c209bfb4a6b8e0eeeff0e2a2e5e7f5c308b3b4b6caf0fdfebb0346ddb2477098a463071870fb
c7b0e1a680a206169c782d1bad52a71c6adca888c3280813439a83a06e14825b1c536e0c256a80c897e606
e043984ca54d782c01b884c9b3dab571a3262dbc49b4d7af603d939ec3a6cd54d1a7ae8e2a9d5a8f643e51
07a06afd248f1cd5aff5208ada4a36dc387a60d30a030aa0ac5b4813ffceaa552b76c482b66ff32a122b71
2ed59c0e0adcd54bd8105fbf60d515583c3891020efc0a13cdb913f1548b061633c67b48ec819a925352b6
4c751c09cd9b1189a5f53972e87f0cb05626cdb32eead4863cb62449d3f56b7052692785302af0ed020e38
0f1a4d8d376bd0bfbb0517ce33e782e3c8c915627ece7950e8d17775454bfda562ecc913149a5ed03b56f0
e1578d2fcff35766ec1100a827c45ea47b64bec5d7c9001988f2017d308d13017605e4b79f200afc32cd54
```

ffb526e08002212852070a32e82021f3a555217c173a729886138965dc711f0e12a25f2306582222dca168
8f8a0c92a01321038cd3976531ce08ca28b3d9b8d4280c1660c0ff8e839c485f90420e7254074612945392
4b0ee5248a5096d85f95d55cc9609684d4586585198517215660dac3d27dd81970c05066b6a9807bf1bd68
27356f26094c3b82b0a441917bf23956747a16da27837ff2c766a1f6e070682f1c5808508690a2230a9cc7
352ae5a3999a03440e93eab2e677327eb2a5a29bfa198ca37f865acd003fd4709772aef4184b6faae80600
48a142c4e96d5eb92897acd40494d9adbbe4444f979bd469670748a2a79db1f3209b2c00f731eb0a341f75
5712809a486be7380c26870088c76aabac66deb24220a8f640dbc897ac02b0226aeaf258a07eda0af32ebc
a5a6129b8130d98b08be851a4380b5eb3689a9bbdcde162fff2a71c542a87fe35a6a48a2a18a52c271e9f1
3831b203a376b127be4ef815b420870cc075a8b52888581b439ab2ca0573a2ce8f60c598b157010b234a03
35eb57269145efcc33ae9ae80a4cce228e7b72c0bf186773244c07ecf4d39e38aba17b40070c948e0f72a0
0ed57b7e0df626e08a02ac8d107cc036b2161dc3dadd6dbafd7626f3024365d1eee6fd1ce17eff7d494e07
12eef80090433e78d315274930d48e98ebf8e65e576ef9669837121b527c736e7adf9e7f2e58cf8e8cde02
0ff3cc7dfaec76268eddca8f8c2e841012547056e9b4073f97edb7b32e3a30bb274f81071800a00125c247
4f1bf1c587ce88eec9674f410816389fc0e4d2873f15fff5d567827df6e85fc03d32e08bef7e8aa9abae38
24e7a36f3f080f4853f6fbfc0f13bffc97cb044bec47c0ece14f1ab2eb9ffbc89724dc994814058ce0ee7a
7796042a30780c6ca0f1180111097a907715681e398077c160fd0f80f37b44073ff8c1e535ef7925345d06
2de7c046ac90852cbc40f3f617c34ccd90861b5cc40d71e8c10a10ad879d1b160acb87892112b180f9e321
12db76c2255a2c887b81e0132318c52926d18aaaab2107b5b845fb75d18b28ab221853f8400094117d1268
1e09d188a01f02508c4224e31be3a8133acaca8e77c462229cf8440a6060507e34a112d7c8203c66d18d6f
bc40ac12a9333532f28ad6cb2324b72849a450b2ff50805ca22307a9471c0660929fac9d252f8949c095f2
8346244f2ac1144a2b8e5235af94602c6749c545b25283997c24113d0080c6f19296abfc652b9b984b029e
f19855aae51a6fd99966a2ef99d0b49134a729485c02e002129480312c984dfa6c939bc15484280210413e
ceb19c6939273a35b1ce02ba139eda4ca6321bd94d44d4d37e3a9c133e51244f4652f310ffcc5e27076aa3
0f706b5ffb44e1410d91d0dd2d94a174fb0509201a5120a633111505c111316a24e79060641df5283d01c0
cedded92a47b620a002280d2942e1313ff7c294c217510008cc0a66c74443d4300b09dcaea27a2f8a94d27
5a887562d3a87f9c8901387a49a612421433ff282a5411679f9a56b59f8669c956677710655a7508be6a1f
a43a50b70f24e09dbc7401ef00803456e6e7a3dbe95ab00c370a63c254aebb130000ec3a0f4b080aaed4c9
89dee406d314644f0274656472aef50886d949b18b652c431d8b3ec1ae71b2afaa2cbd3285d9cc8a82a127
202064a9fa39363d22663135ed62a548c9d44271b04b24d26bdb152a87ca566f7ee5250b5a20410ae80b85
4b62522386264b4859edb75ae55c2836405b9ec080b81ecc1f0093fba03146445bcffdad2739e7d008fc0e
2c806de1713f77d7ee6a32ba990aef6f67078bc0484002cc73de07d42a92203c51bb96bb6bc41aa1b936f1
15ba62355d7d0b400001244f7dddfb0cff4f384b44e3b2363b94654481c1c41604279873c6d04c83ef973f
7290d31c3678238049e65a026b4c5b1df63062ab1462113b58b5be731ef4ea61db325a183b2d668465cfe5
e163cc589b0718d6888b2b8008f785053d56316e51a3db7b8db69245a6c59109fabf257ff00225a6ee70df
a8bdf52e26b943710c6f7d98652dcbf0845ec6618e01400132676fc5dc6d046c55d9e60f3b2e140bc24e9c
8928010b3cc0ceca9b59832262c3710497a77df633e1009da44113d1887546b4100e4000013bc247da2aad
8c151cd94adff88d86d6b4102ed0bd0c2fc218cdcd579fb7aca1055bced24414293811ad021d78aa11c6d8
010c6425dfdfd21a41b6bef5a9dfff788003f09ac1006000248ce16013c437d27d04f1943f876b229eb2a5
658cc1620810ed6903e0c62770c19e8a2ddb63d3a7c6f2eb36119bfd461988b8dc8fa076f64e606d1a63db
ddd41980067cc920799b92a55b7c016ac82ded7c9fdb7ea7dcc0892d13e3516fee8706c721bd718802852f
1cdf9f7e38fa4e190b04f0772e154730c0851390875931e32cfc360b51b082e330dcdccbde1dc90f8798d5
b479e5b4a1341861cec28d4b500535b739c8812df2eced5c6f3b068bcfb30c74d2087de83927b3244170f4
4a2f9d11fa1eb978ab5b90a917b9ea964936d655bdbb0758208217b8f5d75fddf4e43d5db60898b895b09d
6dc7a97ded6c376e0508286effb9373ce459bffb6f35b0df9788dae28ecbc6850148741cbadd7e1e373cce
211e690917c4b791d67ba6e0cdcacab350f0d9cbbce61d9e78be8fb01eec962dd9f744fad2671dd1971702
cd293f774584dde9ae7f4f35626fdad9a38ee080f72005022001f55660f7bc3f3cd35b1f7c934020e50836
3e32919ffc0282a0d5c0c874040fd08397f73e11bfb77bf5d70f5f6d11e734293578fe025d8225edba8007
a82b0a6fce7aceb3dff5da572557d751f2664410a523cd877ffa177d9b2776ff876d016824034880b73757
4a452c87a680e6277d6057773af78091168136f277145840c4942e74a6814bc47f88e77f20586422882224
588204840117d8291ea0ff82fb777e88907e1ff8821ee63245630c5e0554b8065934831fce4640f9b7810d
087c40586451272bb507540c767ba76439c9a1830cd87f0e188545a601df93291cb26d56386eb7674440c4
85f1c68387e08342a078608860d4d54b676873b727002ec74f6ca83a2c387d2e38877d36851a5250f19686
6678457dc86d6e6808702887821886638820137887684840599824 25b049e8d3842bd88885f08891587d75
985833658982767bc6653923b084f6e3893bc88174477da3587d84a816a17083a868630564010b482c21b0
88abd7825f588bd5278627971433b88bdd4654fc747f9df88b7e088a84208ac6f880a54815cbc88c15f863
36e78aaf288d8c28ff8bbee78171788d40788b22610c49b88b7818419e751bf9018dd1e8845e0885e8f882
c8982089e88e0b57816d67663a3278e3278ec30888c5988f2f286613010bfea86c1e2458a360015c2741b0
d885c4888f0a1985ec740227e00326109229e09142a0870f696a1f2401015001e0669106e975e4887ee608
891bc97e2d4940267992a9c876d973916d18933d389335098637693f39a993b7617a1af79205478d83608d

```
43f98245d9597b8894bcc893bbe393d308946f2894512995115995567985589995 4c29684e290850f995ff
478f46299656a9944577964ac7958ee8956cf97f1f749463499665a995e3f884ea97977a19967d79955809
984932597619ff8a784998d5b79770899472f9418a692dbf86901a0999ae074bfda89395e941978949a175
8f83c999c1f741997898a1e99200340aa9528eb4889a7ca79a9f7992ad5990f2536596b096b41969b67998
97f89774c99bbdf998bfd967c1299cb9298c9a81669ae09bc99965cb79980b207e3c399a79169dc8399d45
569d8759966689258cb609d2e99d08069e56991c0f5091d969903aa2349c709ee8f95beaa99393350aedc9
7680e9699d409ff599596fe741ab899477d56cc7f0006ef9445ae99fffd99d016a5ae05840057a92078a0b
13c000ff220a1610020b6a99bf380285f509001aa108719fee78a18520131b0a001dfaa15c085aa950a226
1a14288aff8a2aaa1a08d0a22f3a979a015a80e209345aa3c843a0b779860e2a443b4a0b18e001d8198e8b
01a4aa30a4353aa19878a44095a48ea036c7d0a44f3a9e05200a1910a4240aa144ca1a376a855a6a095cca
a44eda9348639ca840a5266aa59ce78fe4e66a9ad0a6a3800115d07c34909f6936a7667aa65891a629955c
03860a1c100dac7103723aa3e6284e869a59762a76bba8a8df92008e4a0bee35a5e6b83bdb53a9276aa4a8
a8a9e2c1a9e5e90a70883e2144aa452a41156a53dbb90c0a109b840a900a5562677aa9c07787b52a19e320
005faa5a1e007e11aaab1f7886c12a198af500538989bc5a9fca7a8e56d8aca131012561017f7a7a4d46ad
88caff480251220a90002d1a02c54a4020807d7959adb3fa4be31a2503d0a9c010adf633aabfe9ae586a45
f11a2582a0adb460011e9080b0c4ae1ba9af1dd5affeea223d250a0f90ae0035ad5f89b0fb14a90bdb2425
7100eee941bd83ac3549b1f0da1217bb08191ab0036b4a129b8f207b49163bb287d0b000a0a0382401dfaa
902b0b469b28b22eab42194b902cb4ae2a1bae5ad8b23bab080aa0a113290004ab7c12598b378b42825ab4
e5c216d07a6906cb7e3e1b41efaa3a512bb50f51127e5a61290b82f6aa7e6034591930a85e8b09e5daa2c4
3ab3af0a8465bbac4b24a56bcba8543bb74e37b6c1a7b75b8b829979b79f00b013d9ad2c845f1eeb7a7ebb
ffaf3f1a6482ab0a6d4b0be8ea6d578b608b0b40221ab88fbb0af3ca1a7aab3c358b6d97ab3a39bab9ba40
b81c7ab27256b98b35ba01e6b8a66b2ae64a0b0f5b6144d566aeeb21b01bbbbc2013acb1b11c1bb7968ba8
77250264cabbcb50b213a9ba39c4b7c790bbf238a2c80b0e302bb3337bacb205bd9ab1a6d3eb0dbe1b1459
ab9a7cabbd8ba6a7dd0b0e47bba116a0b41516bad6f4ab4929bde7fb0fd50ba3ea0a14316786a83abf0ef1
bda210b6b6cb895abb6dfbcbbf1a11b9a3f0b6e2891a056cc01c5129b4fbb944f49cf2ebc02a81ba00e0a7
4b6b6767b6aa164c14082c0a938b684ae2c11ffc148deab9642601d87ac24f81c1dcbac11c2b429fea39c2
5a11c2310bb10a45c3365c189d1b14c0fb8535dcc36ea1bca94bb02a995f264cc48501b396aab64cecc3cf
8511514cae910339c7ebc28100003b)
INSERT INTO [dbo].[WidgetDescriptions] ([WidgetID], [ShortDescription], [Description],
[Picture]) VALUES (2, N'A medium-sized widget', N'<description
xmlns="http://www.red-gate.com/widgets/WidgetDescriptionSchema">
  <descriptiveText>This is a medium-sized widget, ideally suited for the home and
garden environment</descriptiveText>
  <manufacturer>Magic Widget Company</manufacturer>
  <countryOfOrigin>USA</countryOfOrigin>
  <size length="15" width="12" height="2" />
</description>',
0x474946383961db00d800e643000000006984954658630ccccc010f1b003366000b143333336666669999
99ffffff032d51808080021e3600333334424aafafaf01162884a5bae0e0e07b9aae586e7c111618669999
1a21252020201f1f1f4f4f4fb0b0b0dfdfdf4f636f7f7f7f6079883d4d57505050021a2f00003303294ac6
d8e399cccc84a9c1b7cedc739bb61e517de9f0f494b5cad4e1ea336699e2ebf15a85a5517d9f7b878d3651
65144876b0c9d8333366afb3b613395a0916204d5d678fa3af2d4252bed3df7f8386d0dae3cddce68db1c7
ffffff000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000021f90401000043002c00000000db00d8000007ff80438283848586878898a8b
8c8d8e8f43090708039096979899a9b9c9d9e890900a20019081c9fa8a9aaabacad8708a21a1aa3a22209
0aaeb9babbbcabb000070a0a031f07b4c00913bdcbcccdce0a22a21bc2d40a1d091bc7a595ceddedf9a0a
c60008d5e6c209bfb4a6b8e0eeeff0e2a2e5e7f5c308b3b4b6caf0fdfebb0346ddb2477098a463071870fb
c7b0e1a680a206169c782d1bad52a71c6adca888c3280813439a83a06e14825b1c536e0c256a80c897e606
e043984ca54d782c01b884c9b3dab571a3262dbc49b4d7af603d939ec3a6cd54d1a7ae8e2a9d5a8f643e51
07a06afd248f1cd5aff5208ada4a36dc387a60d30a030aa0ac5b4813ffceaa552b76c482b66ff32a122b71
2ed59c0e0adcd54bd8105fbf60d515583c3891020efc0a13cdb913f1548b061633c67b48ec819a925352b6
4c751c09cd9b1189a5f53972e87f0cb05626cdb32eead4863cb62449d3f56b7052692785302af0ed020e38
0f1a4d8d376bd0bfbb0517ce33e782e3c8c915627ece7950e8d17775454bfda562ecc913149a5ed03b56f0
e1578d2fcff35766ec1100a827c45ea47b64bec5d7c9001988f2017d308d13017605e4b79f200afc32cd54
ffb526e08002212852070a32e82021f3a555217c173a729886138965dc711f0e12a25f2306582222dca168
8f8a0c92a01321038cd3976531ce08ca28b3d9b8d4280c1660c0ff8e839c485f90420e7254074612945392
```

4b0ee5248a5096d85f95d55cc9609684d4586585198517215660dac3d27dd81970c05066b6a9807bf1bd68
27356f26094c3b82b0a441917bf23956747a16da27837ff2c766a1f6e070682f1c5808508690a2230a9cc7
352ae5a3999a03440e93eab2e677327eb2a5a29bfa198ca37f865acd003fd4709772aef4184b6faae80600
48a142c4e96d5eb92897acd40494d9adbbe4444f979bd469670748a2a79db1f3209b2c00f731eb0a341f75
5712809a486be7380c26870088c76aabac66deb24220a8f640dbc897ac02b0226aeaf258a07eda0af32ebc
a5a6129b8130d98b08be851a4380b5eb3689a9bbdcde162fff2a71c542a87fe35a6a48a2a18a52c271e9f1
3831b203a376b127be4ef815b420870cc075a8b52888581b439ab2ca0573a2ce8f60c598b157010b234a03
35eb57269145efcc33ae9ae80a4cce228e7b72c0bf186773244c07ecf4d39e38aba17b40070c948e0f72a0
0ed57b7e0df626e08a02ac8d107cc036b2161dc3dadd6dbafd7626f3024365d1eee6fd1ce17eff7d494e07
12eef80090433e78d315274930d48e98ebf8e65e576ef9669837121b527c736e7adf9e7f2e58cf8e8cde02
0ff3cc7dfaec76268eddca8f8c2e841012547056e9b4073f97edb7b32e3a30bb274f81071800a00125c247
4f1bf1c587ce88eec9674f410816389fc0e4d2873f15fff5d567827df6e85fc03d32e08bef7e8aa9abae38
24e7a36f3f080f4853f6fbfc0f13bffc97cb044bec47c0ece14f1ab2eb9ffbc89724dc994814058ce0ee7a
7796042a30780c6ca0f1180111097a907715681e398077c160fd0f80f37b44073ff8c1e535ef7925345d06
2de7c046ac90852cbc40f3f617c34ccd90861b5cc40d71e8c10a10ad879d1b160acb87892112b180f9e321
12db76c2255a2c887b81e0132318c52926d18aaaab2107b5b845fb75d18b28ab221853f8400094117d1268
1e09d188a01f02508c4224e31be3a8133acaca8e77c462229cf8440a6060507e34a112d7c8203c66d18d6f
bc40ac12a9333532f28ad6cb2324b72849a450b2ff50805ca22307a9471c0660929fac9d252f8949c095f2
8346244f2ac1144a2b8e5235af94602c6749c545b25283997c24113d0080c6f19296abfc652b9b984b029e
f19855aae51a6fd99966a2ef99d0b49134a729485c02e002129480312c984dfa6c939bc154842802104l3e
ceb19c6939273a35b1ce02ba139eda4ca6321bd94d44d4d37e3a9c133e51244f4652f310ffcc5e27076aa3
0f706b5ffb44e1410d91d0dd2d94a174fb0509201a5120a633111505c111316a24e79060641df5283d01c0
cedded92a47b620a002280d2942e1313ff7c294c217510008cc0a66c74443d4300b09dcaea27a2f8a94d27
5a887562d3a87f9c8901387a49a612421433ff282a5411679f9a56b59f8669c956677710655a7508be6a1f
a43a50b70f24e09dbc7401ef00803456e6e7a3dbe95ab00c370a63c254aebb130000ec3a0f4b080aaed4c9
89dee406d314644f0274656472aef50886d949b18b652c431d8b3ec1ae71b2afaa2cbd3285d9cc8a82a127
202064a9fa39363d22663135ed62a548c9d44271b04b24d26bdb152a87ca566f7ee5250b5a20410ae80b85
4b62522386264b4859edb75ae55c2836405b9ec080b81ecc1f0093fba03146445bcffdad2739e7d008fc0e
2c806de1713f77d7ee6a32ba990aef6f67078bc0484002cc73de07d42a92203c51bb96bb6bc41aa1b936f1
15ba62355d7d0b400001244f7dddfb0cff4f384b44e3b2363b94654481c1c41604279873c6d04c83ef973f
7290d31c3678238049e65a026b4c5b1df63062ab1462113b58b5be731ef4ea61db325a183b2d668465cfe5
e163cc589b0718d6888b2b8008f785053d56316e51a3db7b8db69245a6c59109fabf257ff00225a6ee70df
a8bdf52e26b943710c6f7d98652dcbf0845ec6618e01400132676fc5dc6d046c55d9e60f3b2e140bc24e9c
8928010b3cc0ceca9b59832262c3710497a77df633e1009da44113d1887546b4100e4000013bc247da2aad
8c151cd94adff88d86d6b4102ed0bd0c2fc218cdcd579fb7aca1055bced24414293811ad021d78aa11c6d8
010c6425dfdfd21a41b6bef5a9dfff788003f09ac1006000248ce16013c437d27d04f1943f876b229eb2a5
658cc1620810ed6903e0c62770c19e8a2ddb63d3a7c6f2eb36119bfd461988b8dc8fa076f64e606d1a63db
ddd41980067cc920799b92a55b7c016ac82ded7c9fdb7ea7dcc0892d13e3516fee8706c721bd718802852f
1cdf9f7e38fa4e190b04f0772e154730c0851390875931e32cfc360b51b082e330dcdccbde1dc90f8798d5
b479e5b4a1341861cec28d4b500535b739c8812df2eced5c6f3b068bcfb30c74d2087de83927b3244170f4
4a2f9d11fa1eb978ab5b90a917b9ea964936d655bdbb0758208217b8f5d75fddf4e43d5db60898b895b09d
6dc7a97ded6c376e0508286effb9373ce459bffb6f35b0df9788dae28ecbc6850148741cbadd7e1e373cce
211e690917c4b791d67ba6e0cdcacab350f0d9cbbce61d9e78be8fb01eec962dd9f744fad2671dd1971702
cd293f774584dde9ae7f4f35626fdad9a38ee080f72005022001f55660f7bc3f3cd35b1f7c934020e50836
3e32919ffc0282a0d5c0c874040fd08397f73e11bfb77bf5d70f5f6d11e734293578fe025d8225edba8007
a82b0a6fce7aceb3dff5da572557d751f2664410a523cd877ffa177d9b2776ff876d016824034880b73757
4a452c87a680e6277d6057773af78091168136f277145840c4942e74a6814bc47f88e77f20586422882224
588204840117d8291ea0ff82fb777e88907e1ff8821ee63245630c5e0554b8065934831fce4640f9b7810d
087c40586451272bb507540c767ba76439c9a1830cd87f0e188545a601df93291cb26d56386eb7674440c4
85f1c68387e08342a078608860d4d54b676873b727002ec74f6ca83a2c387d2e38877d36851a5250f19686
6678457dc86d6e6808702887821886638820137887684840599824255b049e8d3842bd88885f08891587d75
985833658982767bc6653923b084f6e3893bc88174477da3587d84a816a17083a868630564010b482c21b0
88abd7825f588bd5278627971433b88bdd4654fc747f9df88b7e088a84208ac6f880a54815cbc88c15f863
36e78aaf288d8c28ff8bbee78171788d40788b22610c49b88b7818419e751bf9018dd1e8845e0885e8f882
c8982089e88e0b57816d67663a3278e3278ec30888c5988f2f286613010bfea86c1e2458a360015c2741b0
d885c4888f0a1985ec740227e00326109229e09142a0870f696a1f2401015001e0669106e975e4887ee608
891bc97e2d4940267992a9c876d973916d18933d389335098637693f39a993b7617a1af79205478d83608d
43f98245d9597b8894bcc893bbe393d308946f2894512995115995567985589995 4c29684e290850f995ff
478f46299656a9944577964ac7958ee8956cf97f1f749463499665a995e3f884ea97977a19967d79955809

```
984932597619ff8a784998d5b79770899472f9418a692dbf86901a0999ae074bfda89395e941978949a175
8f83c999c1f741997898a1e99200340aa9528eb4889a7ca79a9f7992ad5990f2536596b096b41969b67998
97f89774c99bbdf998bfd967c1299cb9298c9a81669ae09bc99965cb79980b207e3c399a79169dc8399d45
569d8759966689258cb609d2e99d08069e56991c0f5091d969903aa2349c709ee8f95beaa99393350aedc9
7680e9699d409ff599596fe741ab899477d56cc7f0006ef9445ae99fffd99d016a5ae05840057a92078a0b
13c000ff220a1610020b6a99bf380285f509001aa108719fee78a18520131b0a001dfaa15c085aa950a226
1a14288aff8a2aaa1a08d0a22f3a979a015a80e209345aa3c843a0b779860e2a443b4a0b18e001d8198e8b
01a4aa30a4353aa19878a44095a48ea036c7d0a44f3a9e05200a1910a4240aa144ca1a376a855a6a095cca
a44eda9348639ca840a5266aa59ce78fe4e66a9ad0a6a3800115d07c34909f6936a7667aa65891a629955c
03860a1c100dac7103723aa3e6284e869a59762a76bba8a8df92008e4a0bee35a5e6b83bdb53a9276aa4a8
a8a9e2c1a9e5e90a70883e2144aa452a41156a53dbb90c0a109b840a900a5562677aa9c07787b52a19e320
005faa5a1e007e11aaab1f7886c12a198af500538989bc5a9fca7a8e56d8aca131012561017f7a7a4d46ad
88caff480251220a90002d1a02c54a4020807d7959adb3fa4be31a2503d0a9c010adf633aabfe9ae586a45
f11a2582a0adb460011e9080b0c4ae1ba9af1dd5affeea223d250a0f90ae0035ad5f89b0fb14a90bdb2425
7100eee941bd83ac3549b1f0da1217bb08191ab0036b4a129b8f207b49163bb287d0b000a0a0382401dfaa
902b0b469b28b22eab42194b902cb4ae2a1bae5ad8b23bab080aa0a113290004ab7c12598b378b42825ab4
e5c216d07a6906cb7e3e1b41efaa3a512bb50f51127e5a61290b82f6aa7e6034591930a85e8b09e5daa2c4
3ab3af0a8465bbac4b24a56bcba8543bb74e37b6c1a7b75b8b829979b79f00b013d9ad2c845f1eeb7a7ebb
ffaf3f1a6482ab0a6d4b0be8ea6d578b608b0b40221ab88fbb0af3ca1a7aab3c358b6d97ab3a39bab9ba40
b81c7ab27256b98b35ba01e6b8a66b2ae64a0b0f5b6144d566aeeb21b01bbbbc2013acb1b11c1bb7968ba8
77250264cabbcb50b213a9ba39c4b7c790bbf238a2c80b0e302bb3337bacb205bd9ab1a6d3eb0dbe1b1459
ab9a7cabbd8ba6a7dd0b0e47bba116a0b41516bad6f4ab4929bde7fb0fd50ba3ea0a14316786a83abf0ef1
bda210b6b6cb895abb6dfbcbbf1a11b9a3f0b6e2891a056cc01c5129b4fbb944f49cf2ebc02a81ba00e0a7
4b6b6767b6aa164c14082c0a938b684ae2c11ffc148deab9642601d87ac24f81c1dcbac11c2b429fea39c2
5a11c2310bb10a45c3365c189d1b14c0fb8535dcc36ea1bca94bb02a995f264cc48501b396aab64cecc3cf
8511514cae910339c7ebc28100003b)

-- Populate Contacts
SET IDENTITY_INSERT [dbo].[Contacts] ON
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (1, N'Christopher Martin',
N'01446 175923', N'07634 717173', N'17 High Street', N'Darlington', N'Cheshire',
'20070914 16:09:21.383', N'Christopher.Martin@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (2, N'Joseph Bennett', N'01147
771824', N'07956 758471', N'37 Green Lane', N'Portsmouth', N'County Durham', '20070914
16:09:21.400', N'Joseph.Bennett@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (3, N'Joshua Scott', N'01940
919059', N'07331 757163', N'33 George Street', N'Llandudno', N'County Durham',
'20070914 16:09:21.400', N'Joshua.Scott@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (4, N'Daniel Powell', N'01659
216066', N'07272 134093', N'93 North Street', N'Newport', N'Cheshire', '20070914
16:09:21.417', N'Daniel.Powell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (5, N'Jennifer Washington',
N'01502 976179', N'07225 483640', N'43 The Grove', N'Hull', N'Berkshire', '20070914
16:09:21.430', N'Jennifer.Washington@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (6, N'Anthony Lee', N'01200
660059', N'07696 849723', N'97 Stanley Road', N'Newport', N'Berkshire', '20070914
16:09:21.430', N'Anthony.Lee@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (7, N'Rebecca Foster', N'01340
503829', N'07960 669272', N'79 Springfield Road', N'Salisbury', N'Licolnshire',
'20070914 16:09:21.447', N'Rebecca.Foster@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (8, N'Ava Walker', N'01978
```

```
657197', N'07610 445571', N'29 North Street', N'Southall', N'Oxfordshire', '20070914
16:09:21.463', N'Ava.Walker@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (9, N'Matthew Ward', N'01324
940147', N'07903 833764', N'71 Kingsway', N'London EC', N'Staffordshire', '20070914
16:09:21.477', N'Matthew.Ward@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (10, N'Matthew Mitchell',
N'01661 434756', N'07822 453924', N'70 King Street', N'Worcester', N'Rutland',
'20070914 16:09:21.477', N'Matthew.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (11, N'Megan Stewart', N'01653
610021', N'07363 920914', N'59 Broadway', N'Chelmsford', N'Berkshire', '20070914
16:09:21.493', N'Megan.Stewart@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (12, N'Mia Robinson', N'01594
360107', N'07295 889232', N'22 The Avenue', N'Wigan', N'Norfolk', '20070914
16:09:21.510', N'Mia.Robinson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (13, N'Cameron Davis', N'01594
360107', N'07295 889232', N'35 Windsor Road', N'Watford', N'Rutland', '20070914
16:09:21.523', N'Cameron.Davis@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (14, N'Emma Howard', N'01788
915553', N'07736 274237', N'43 Park Avenue', N'Wigan', N'Derbyshire', '20070914
16:09:21.540', N'Emma.Howard@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (15, N'Liam Taylor', N'01286
103534', N'07939 391131', N'47 York Road', N'Harrow', N'Shropshire', '20070914
16:09:21.540', N'Liam.Taylor@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (16, N'Ross Ramirez', N'01945
464335', N'07126 593543', N'33 Kingsway', N'London WC', N'Herefordshire', '20070914
16:09:21.557', N'Ross.Ramirez@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (17, N'Sarah Lewis', N'01287
924924', N'07149 593067', N'91 George Street', N'Bristol', N'Cumberland', '20070914
16:09:21.570', N'Sarah.Lewis@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (18, N'Emma Taylor', N'01287
924924', N'07149 593067', N'92 Queen Street', N'Dundee', N'Kent', '20070914
16:09:21.587', N'Emma.Taylor@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (19, N'Anthony Rivera', N'01788
915553', N'07736 274237', N'97 Church Road', N'Southampton', N'Huntingdonshire',
'20070914 16:09:21.587', N'Anthony.Rivera@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (20, N'Ellie King', N'01945
464335', N'07126 593543', N'29 George Street', N'Lincoln', N'Lancashire', '20070914
16:09:21.603', N'Ellie.King@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (21, N'Megan Barnes', N'01469
486350', N'07759 520189', N'92 West Street', N'York', N'Surrey', '20070914
16:09:21.620', N'Megan.Barnes@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (22, N'Scott Watson', N'01630
240971', N'07886 967048', N'91 Mill Road', N'Chester', N'Norfolk', '20070914
16:09:21.633', N'Scott.Watson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
```

```
[Address2], [Address3], [JoiningDate], [Email]) VALUES (23, N'Emily Anderson', N'01487
403649', N'07449 889487', N'100 George Street', N'Dorchester', N'Hertfordshire',
'20070914 16:09:21.633', N'Emily.Anderson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (24, N'Nicole Martin', N'01721
599988', N'07525 825680', N'62 Kingsway', N'Colchester', N'Hertfordshire', '20070914
16:09:21.650', N'Nicole.Martin@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (25, N'David Perry', N'01212
359994', N'07270 368607', N'59 Queen Street', N'London N', N'Cheshire', '20070914
16:09:21.667', N'David.Perry@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (26, N'Abbie Rogers', N'01120
243659', N'07237 275885', N'62 Manchester Road', N'Kirkcaldy', N'Cornwall', '20070914
16:09:21.680', N'Abbie.Rogers@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (27, N'Liam Kelly', N'01140
882420', N'07354 721715', N'88 St. John''s Road', N'Manchester', N'Northamptonshire',
'20070914 16:09:21.680', N'Liam.Kelly@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (28, N'Christopher Walker',
N'01889 655242', N'07332 289915', N'73 North Street', N'Paisley', N'Yorkshire',
'20070914 16:09:21.697', N'Christopher.Walker@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (29, N'Matthew Collins',
N'01305 665775', N'07352 869339', N'60 York Road', N'Outer Hebrides',
N'Staffordshire', '20070914 16:09:21.713', N'Matthew.Collins@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (30, N'Elidh Peterson', N'01183
454781', N'07295 215265', N'59 Albert Road', N'Norwich', N'Cheshire', '20070914
16:09:21.727', N'Elidh.Peterson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (31, N'Jamie Price', N'01624
725814', N'07773 220491', N'40 Manor Road', N'Kingston upon Thames', N'Rutland',
'20070914 16:09:21.743', N'Jamie.Price@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (32, N'Katie Sanchez', N'01192
635557', N'07695 694205', N'50 New Street', N'Hemel Hempstead', N'Cornwall', '20070914
16:09:21.743', N'Katie.Sanchez@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (33, N'Erin Rodriguez', N'01701
154719', N'07699 485122', N'8 York Road', N'Swindon', N'Derbyshire', '20070914
16:09:21.760', N'Erin.Rodriguez@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (34, N'Abbie Brooks', N'01287
924924', N'07149 593067', N'73 Springfield Road', N'Ipswich', N'County Durham',
'20070914 16:09:21.773', N'Abbie.Brooks@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (35, N'Lauren Simmons', N'01442
337633', N'07986 644216', N'98 London Road', N'London E', N'Cheshire', '20070914
16:09:21.790', N'Lauren.Simmons@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (36, N'Emily Bennett', N'01907
200803', N'07636 884168', N'90 West Street', N'Salisbury', N'Herefordshire', '20070914
16:09:21.790', N'Emily.Bennett@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (37, N'Christopher Flores',
N'01198 529461', N'07207 549127', N'74 Queens Road', N'Crewe', N'Cheshire', '20070914
16:09:21.807', N'Christopher.Flores@example.com')
```

```
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (38, N'Alexander Rivera',
N'01120 243659', N'07237 275885', N'25 Grange Road', N'Dudley', N'Huntingdonshire',
'20070914 16:09:21.820', N'Alexander.Rivera@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (39, N'Rachel Williams',
N'01512 965727', N'07889 215998', N'62 Victoria Road', N'Dundee', N'Hampshire',
'20070914 16:09:21.837', N'Rachel.Williams@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (40, N'Olvia Howard', N'01547
693549', N'07341 808038', N'17 Kings Road', N'Guildford', N'Warwickshire', '20070914
16:09:21.837', N'Olvia.Howard@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (41, N'Elizabeth Flores',
N'01231 471798', N'07493 384582', N'64 The Green', N'Bournemouth', N'Staffordshire',
'20070914 16:09:21.853', N'Elizabeth.Flores@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (42, N'Kyle Griffin', N'01558
179557', N'07522 904690', N'21 Manchester Road', N'Kirkcaldy', N'Hampshire', '20070914
16:09:21.870', N'Kyle.Griffin@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (43, N'Laura Evan', N'01495
153619', N'07390 958114', N'68 Station Road', N'St Albans', N'Berkshire', '20070914
16:09:21.883', N'Laura.Evan@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (44, N'Katie Martinez', N'01285
948244', N'07424 397578', N'99 Windsor Road', N'Lerwick', N'Oxfordshire', '20070914
16:09:21.883', N'Katie.Martinez@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (45, N'Sophie Griffin', N'01965
513379', N'07605 294144', N'10 The Green', N'Sheffield', N'Surrey', '20070914
16:09:21.900', N'Sophie.Griffin@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (46, N'Elidh Wood', N'01198
529461', N'07207 549127', N'88 North Road', N'Northampton', N'Essex', '20070914
16:09:21.917', N'Elidh.Wood@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (47, N'Abigail Hill', N'01555
443077', N'07439 738948', N'86 The Drive', N'Sunderland', N'Hampshire', '20070914
16:09:21.930', N'Abigail.Hill@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (48, N'Scott Bell', N'01469
486350', N'07759 520189', N'25 Park Avenue', N'Harrogate', N'Surrey', '20070914
16:09:21.930', N'Scott.Bell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (49, N'Chloe Martin', N'01329
343662', N'07783 161081', N'33 New Street', N'Telford', N'Dorset', '20070914
16:09:21.947', N'Chloe.Martin@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (50, N'Lauren Bryant', N'01752
908316', N'07676 584363', N'70 School Lane', N'Swindon', N'Middlesex', '20070914
16:09:21.963', N'Lauren.Bryant@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (51, N'Lucy Rivera', N'01287
924924', N'07149 593067', N'31 The Grove', N'Rochester', N'Cumberland', '20070914
16:09:21.977', N'Lucy.Rivera@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (52, N'Chloe Davis', N'01346
526218', N'07291 685237', N'99 The Crescent', N'Coventry', N'Shropshire', '20070914
```

```
16:09:21.977', N'Chloe.Davis@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (53, N'Joseph Thompson',
N'01237 661433', N'07302 871821', N'75 Grove road', N'Truro', N'Northumberland',
'20070914 16:09:21.993', N'Joseph.Thompson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (54, N'William Wright', N'01417
502667', N'07177 363555', N'57 Station Road', N'Romford', N'Middlesex', '20070914
16:09:22.010', N'William.Wright@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (55, N'William Hughes', N'01407
629329', N'07309 884547', N'34 High Street', N'Torquay', N'Dorset', '20070914
16:09:22.023', N'William.Hughes@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (56, N'Samantha Miller',
N'01333 824728', N'07634 606994', N'60 Park Lane', N'Cardiff', N'Surrey', '20070914
16:09:22.040', N'Samantha.Miller@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (57, N'William Ramirez',
N'01626 530682', N'07768 951632', N'95 Grange Road', N'Cardiff', N'Berkshire',
'20070914 16:09:22.040', N'William.Ramirez@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (58, N'Lauren Green', N'01811
269842', N'07316 264705', N'28 Main Street', N'Huddersfield', N'Somerset', '20070914
16:09:22.057', N'Lauren.Green@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (59, N'Jordan Henderson',
N'01624 725814', N'07773 220491', N'67 The Drive', N'Tonbridge', N'Surrey', '20070914
16:09:22.070', N'Jordan.Henderson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (60, N'Michael Richardson',
N'01346 526218', N'07291 685237', N'76 Highfield Road', N'Kilmarnock', N'County
Durham', '20070914 16:09:22.087', N'Michael.Richardson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (61, N'Caitlin Washington',
N'01291 914147', N'07913 286110', N'55 School Lane', N'Peterborough', N'Licolnshire',
'20070914 16:09:22.103', N'Caitlin.Washington@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (62, N'Anna Long', N'01131
669331', N'07113 804523', N'41 The Avenue', N'Stevenage', N'Cornwall', '20070914
16:09:22.103', N'Anna.Long@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (63, N'Ethan Parker', N'01940
919059', N'07331 757163', N'76 York Road', N'Blackpool', N'Middlesex', '20070914
16:09:22.120', N'Ethan.Parker@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (64, N'Lauren Ramirez', N'01417
502667', N'07177 363555', N'74 Albert Road', N'Crewe', N'Kent', '20070914
16:09:22.133', N'Lauren.Ramirez@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (65, N'Joshua Price', N'01404
421474', N'07720 314778', N'24 The Green', N'Southend on Sea', N'Yorkshire', '20070914
16:09:22.150', N'Joshua.Price@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (66, N'Kyle Hall', N'01291
914147', N'07913 286110', N'25 Kingsway', N'Aberdeen', N'Lancashire', '20070914
16:09:22.150', N'Kyle.Hall@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (67, N'Bethany Hughes', N'01210
```

```
611735', N'07709 761071', N'6 New Street', N'Stevenage', N'Buckinghamshire', '20070914
16:09:22.167', N'Bethany.Hughes@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (68, N'Kyle Smith', N'01555
443077', N'07439 738948', N'90 Main Street', N'Cardiff', N'Buckinghamshire', '20070914
16:09:22.180', N'Kyle.Smith@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (69, N'Caitlin Griffin',
N'01469 486350', N'07759 520189', N'100 Mill Road', N'Oxford', N'Sussex', '20070914
16:09:22.197', N'Caitlin.Griffin@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (70, N'Jamie Perez', N'01572
459202', N'07858 711874', N'86 Queens Road', N'Nottingham', N'Staffordshire',
'20070914 16:09:22.213', N'Jamie.Perez@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (71, N'Jennifer Turner',
N'01192 635557', N'07695 694205', N'46 London Road', N'Brighton', N'Derbyshire',
'20070914 16:09:22.213', N'Jennifer.Turner@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (72, N'Alexander Barnes',
N'01281 554917', N'07800 642581', N'78 North Street', N'London W', N'Lancashire',
'20070914 16:09:22.227', N'Alexander.Barnes@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (73, N'Megan Campbell', N'01205
396727', N'07198 312473', N'20 Park Avenue', N'Blackburn', N'Suffolk', '20070914
16:09:22.243', N'Megan.Campbell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (74, N'Lewis Collins', N'01237
661433', N'07302 871821', N'10 Victoria Road', N'Southall', N'Norfolk', '20070914
16:09:22.260', N'Lewis.Collins@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (75, N'Bethany Peterson',
N'01784 976829', N'07885 861269', N'43 The Crescent', N'Worcester', N'Surrey',
'20070914 16:09:22.260', N'Bethany.Peterson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (76, N'Erin Martin', N'01428
428418', N'07377 251946', N'39 The Avenue', N'Milton Keynes', N'Suffolk', '20070914
16:09:22.273', N'Erin.Martin@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (77, N'Joseph Coleman', N'01788
915553', N'07736 274237', N'95 Park Avenue', N'Wolverhampton', N'Leicestershire',
'20070914 16:09:22.290', N'Joseph.Coleman@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (78, N'Jordan Scott', N'01500
424473', N'07824 778422', N'12 Stanley Road', N'Galashiels', N'Cambridgeshire',
'20070914 16:09:22.307', N'Jordan.Scott@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (79, N'Kieran Kelly', N'01633
392324', N'07960 870808', N'91 London Road', N'Twickenham', N'Northumberland',
'20070914 16:09:22.320', N'Kieran.Kelly@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (80, N'Anthony Brown', N'01231
471798', N'07493 384582', N'41 Queensway', N'Falkirk', N'Berkshire', '20070914
16:09:22.337', N'Anthony.Brown@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (81, N'Joshua Sanders', N'01902
365096', N'07414 740869', N'12 Albert Road', N'Dundee', N'Hampshire', '20070914
16:09:22.353', N'Joshua.Sanders@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
```

```
[Address2], [Address3], [JoiningDate], [Email]) VALUES (82, N'Bethany Rivera', N'01784
976829', N'07885 861269', N'54 Church Street', N'London E', N'Northumberland',
'20070914 16:09:22.353', N'Bethany.Rivera@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (83, N'Laura Hayes', N'01982
956405', N'07109 544766', N'56 Station Road', N'Brighton', N'Devon', '20070914
16:09:22.370', N'Laura.Hayes@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (84, N'Anna Griffin', N'01231
471798', N'07493 384582', N'40 Windsor Road', N'Guildford', N'Cornwall', '20070914
16:09:22.383', N'Anna.Griffin@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (85, N'Samantha Jackson',
N'01231 471798', N'07493 384582', N'14 New Street', N'Llandudno', N'County Durham',
'20070914 16:09:22.400', N'Samantha.Jackson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (86, N'Kyle Sanchez', N'01978
657197', N'07610 445571', N'3 Windsor Road', N'Bromley', N'Oxfordshire', '20070914
16:09:22.400', N'Kyle.Sanchez@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (87, N'Shannon King', N'01624
725814', N'07773 220491', N'56 King Street', N'Paisley', N'Gloucestershire', '20070914
16:09:22.417', N'Shannon.King@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (88, N'Joseph Gray', N'01211
864653', N'07375 760438', N'56 South Street', N'London E', N'Gloucestershire',
'20070914 16:09:22.430', N'Joseph.Gray@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (89, N'Courtney Perry', N'01210
611735', N'07709 761071', N'13 St. John''s Road', N'York', N'Berkshire', '20070914
16:09:22.447', N'Courtney.Perry@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (90, N'Matthew Campbell',
N'01743 360803', N'07394 612512', N'68 Queen Street', N'Romford', N'Essex', '20070914
16:09:22.463', N'Matthew.Campbell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (91, N'Alexis Cook', N'01624
725814', N'07773 220491', N'36 Queen Street', N'Falkirk', N'Norfolk', '20070914
16:09:22.463', N'Alexis.Cook@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (92, N'Jamie Mitchell', N'01776
202627', N'07185 505540', N'90 High Street', N'Kingston upon Thames', N'Warwickshire',
'20070914 16:09:22.477', N'Jamie.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (93, N'Chloe Morris', N'01725
222331', N'07526 574605', N'97 Main Street', N'Bradford', N'Cambridgeshire', '20070914
16:09:22.493', N'Chloe.Morris@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (94, N'Lauren Morris', N'01914
784723', N'07264 945002', N'44 Church Street', N'Durham', N'Warwickshire', '20070914
16:09:22.510', N'Lauren.Morris@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (95, N'Olvia Barnes', N'01811
269842', N'07316 264705', N'78 Stanley Road', N'Lincoln', N'Herefordshire', '20070914
16:09:22.510', N'Olvia.Barnes@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (96, N'Olvia Hall', N'01571
657822', N'07694 711784', N'14 Windsor Road', N'Coventry', N'Middlesex', '20070914
16:09:22.523', N'Olvia.Hall@example.com')
```

```
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (97, N'Scott Turner', N'01555
443077', N'07439 738948', N'87 Station Road', N'Salisbury', N'Surrey', '20070914
16:09:22.540', N'Scott.Turner@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (98, N'Rebecca Hayes', N'01802
342584', N'07296 315860', N'94 George Street', N'Bournemouth', N'Somerset', '20070914
16:09:22.557', N'Rebecca.Hayes@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (99, N'Kyle Perry', N'01653
610021', N'07363 920914', N'82 The Grove', N'Derby', N'Gloucestershire', '20070914
16:09:22.570', N'Kyle.Perry@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (100, N'Kieran Patterson',
N'01389 465402', N'07266 836025', N'3 St. John''s Road', N'Bolton', N'County Durham',
'20070914 16:09:22.570', N'Kieran.Patterson@example.com')
SET IDENTITY_INSERT [dbo].[Contacts] OFF

-- Populate WidgetPurchases with lots of data
DECLARE @purchaseCount int
SET @purchaseCount = 0
SET NOCOUNT ON
DECLARE @temp float
-- First the base set which will be common with WidgetLive
SET @temp = RAND(1)
WHILE (@purchaseCount < 97425) BEGIN
 INSERT INTO [dbo].[WidgetPurchases] ([WidgetPriceID], [Quantity], [InvoiceNumber],
[Date]) VALUES (
  CONVERT (int, RAND() * 3) + 1, -- WidgetPriceID
  CONVERT (int, RAND() * 100) + 1, -- Quantity
  'WIDG' + convert (nvarchar(10), convert (int, RAND() * 10000) + 1), -- InvoiceNumber
  CONVERT (datetime, -- Date
   CONVERT (nvarchar (50),
    '2007-' +
    CONVERT (nvarchar (2), CONVERT (int, RAND() * 11) + 1) + '-' +
    CONVERT (nvarchar (2), CONVERT (int, RAND () * 27) + 1)
   )
  )
 )
 SET @purchaseCount = @purchaseCount + 1
END
-- Now some more, which won't be, to simulate "development" data entry
SET @temp = RAND(314159)
SET @purchaseCount = 0
WHILE (@purchaseCount < 854) BEGIN
 INSERT INTO [dbo].[WidgetPurchases] ([WidgetPriceID], [Quantity], [InvoiceNumber],
[Date]) VALUES (
  CONVERT (int, RAND() * 3) + 1, -- WidgetPriceID
  CONVERT (int, RAND() * 100) + 1, -- Quantity
  'WIDG' + convert (nvarchar(10), convert (int, RAND() * 10000) + 1), -- InvoiceNumber
  CONVERT (datetime, -- Date
   CONVERT (nvarchar (50),
    '2007-' +
    CONVERT (nvarchar (2), CONVERT (int, RAND() * 11) + 1) + '-' +
    CONVERT (nvarchar (2), CONVERT (int, RAND () * 27) + 1)
   )
  )
 )
 SET @purchaseCount = @purchaseCount + 1
```

```
END
SET NOCOUNT OFF

COMMIT TRANSACTION
GO

-- ************************************* Insert data
*************************************

USE WidgetLive
GO

BEGIN TRANSACTION
SET IDENTITY_INSERT [dbo].[Widgets] ON
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (1, 'Red
widget', 'RW')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (2, 'Extended
widget2', 'EW')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (3, 'Dark
widget', 'DW')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (4,
'Grow-your-own widget', 'GW')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (5, 'Widget
access kit', 'AW')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (6, 'Test kit',
'TK')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (7, 'Exploded
widget diagram', 'ED')
SET IDENTITY_INSERT [dbo].[Widgets] OFF
INSERT INTO [dbo].[WidgetReferences] ([WidgetID], [Reference]) VALUES (1, 'Widget 1')
INSERT INTO [dbo].[WidgetReferences] ([WidgetID], [Reference]) VALUES (2, 'Widget 2')
SET IDENTITY_INSERT [dbo].[WidgetPrices] ON
INSERT INTO [dbo].[WidgetPrices] ([RecordID], [WidgetID], [Price], [DateValidFrom],
[DateValidTo], [Active]) VALUES (1, 1, 100, '2000-01-01 00:00:00.000', '2002-01-01
00:00:00.000', 'Y')
INSERT INTO [dbo].[WidgetPrices] ([RecordID], [WidgetID], [Price], [DateValidFrom],
[DateValidTo], [Active]) VALUES (2, 2, 50, '2000-01-01 00:00:00.000', '2002-01-01
00:00:00.000', 'Y')
INSERT INTO [dbo].[WidgetPrices] ([RecordID], [WidgetID], [Price], [DateValidFrom],
[DateValidTo], [Active]) VALUES (3, 3, 25, '2000-01-01 00:00:00.000', '2002-01-01
00:00:00.000', 'Y')
SET IDENTITY_INSERT [dbo].[WidgetPrices] OFF
INSERT INTO [dbo].[WidgetDescriptions] ([WidgetID], [ShortDescription], [Description],
[Picture]) VALUES (1, N'A widget', N'<description
xmlns="http://www.red-gate.com/widgets/WidgetDescriptionSchema">
  <descriptiveText>A widget</descriptiveText>
  <manufacturer>Acme Incorporated</manufacturer>
  <countryOfOrigin>United Kingdom</countryOfOrigin>
  <size length="12" width="4" height="7" />
</description>',
0x474946383961db00d800e643000000000698495465863cccccc010f1b003366000b143333336666669999
99ffffff032d51808080021e3600333334424aafafaf01162884a5bae0e0e07b9aae586e7c111618669999
1a21252020201f1f1f4f4f4fb0b0b0dfdfdf4f636f7f7f7f6079883d4d57505050021a2f00003303294ac6
d8e399cccc84a9c1b7cedc739bb61e517de9f0f494b5cad4e1ea336699e2ebf15a85a5517d9f7b878d3651
65144876b0c9d8333366afb3b613395a0916204d5d678fa3af2d4252bed3df7f8386d0dae3cddce68db1c7
ffffff000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
```

```
00000000000000000000021f90401000043002c00000000db00d8000007ff804382838485868788898a8b
8c8d8e8f4309070803909697989999a9b9c9d9e890900a20019081c9fa8a9aaabacad8708a21a1aa3a22209
0aaeb9babbbcabb000070a0a031f07b4c00913bdcbcccdce0a22a21bc2d40a1d091bc7a595ceddedf9a0a
c60008d5e6c209bfb4a6b8e0eeeff0e2a2e5e7f5c308b3b4b6caf0fdfebb0346ddb2477098a463071870fb
c7b0e1a680a206169c782d1bad52a71c6adca888c3280813439a83a06e14825b1c536e0c256a80c897e606
e043984ca54d782c01b884c9b3dab571a3262dbc49b4d7af603d939ec3a6cd54d1a7ae8e2a9d5a8f643e51
07a06afd248f1cd5aff5208ada4a36dc387a60d30a030aa0ac5b4813ffceaa552b76c482b66ff32a122b71
2ed59c0e0adcd54bd8105fbf60d515583c3891020efc0a13cdb913f1548b061633c67b48ec819a925352b6
4c751c09cd9b1189a5f53972e87f0cb05626cdb32eead4863cb62449d3f56b7052692785302af0ed020e38
0f1a4d8d376bd0bfbb0517ce33e782e3c8c915627ece7950e8d17775454bfda562ecc913149a5ed03b56f0
e1578d2fcff35766ec1100a827c45ea47b64bec5d7c9001988f2017d308d13017605e4b79f200afc32cd54
ffb526e08002212852070a32e82021f3a555217c173a729886138965dc711f0e12a25f2306582222dca168
8f8a0c92a01321038cd3976531ce08ca28b3d9b8d4280c1660c0ff8e839c485f90420e7254074612945392
4b0ee5248a5096d85f95d55cc9609684d4586585198517215660dac3d27dd81970c05066b6a9807bf1bd68
27356f26094c3b82b0a441917bf23956747a16da27837ff2c766a1f6e070682f1c5808508690a2230a9cc7
352ae5a3999a03440e93eab2e677327eb2a5a29bfa198ca37f865acd003fd4709772aef4184b6faae80600
48a142c4e96d5eb92897acd40494d9adbbe4444f979bd469670748a2a79db1f3209b2c00f731eb0a341f75
5712809a486be7380c26870088c76aabac66deb24220a8f640dbc897ac02b0226aeaf258a07eda0af32ebc
a5a6129b8130d98b08be851a4380b5eb3689a9bbdcde162fff2a71c542a87fe35a6a48a2a18a52c271e9f1
3831b203a376b127be4ef815b420870cc075a8b52888581b439ab2ca0573a2ce8f60c598b157010b234a03
35eb57269145efcc33ae9ae80a4cce228e7b72c0bf186773244c07ecf4d39e38aba17b40070c948e0f72a0
0ed57b7e0df626e08a02ac8d107cc036b2161dc3dadd6dbafd7626f3024365d1eee6fd1ce17eff7d494e07
12eef80090433e78d315274930d48e98ebf8e65e576ef9669837121b527c736e7adf9e7f2e58cf8e8cde02
0ff3cc7dfaec76268eddca8f8c2e841012547056e9b4073f97edb7b32e3a30bb274f81071800a00125c247
4f1bf1c587ce88eec9674f410816389fc0e4d2873f15fff5d567827df6e85fc03d32e08bef7e8aa9abae38
24e7a36f3f080f4853f6fbfc0f13bffc97cb044bec47c0ece14f1ab2eb9ffbc89724dc994814058ce0ee7a
7796042a30780c6ca0f1180111097a907715681e398077c160fd0f80f37b44073ff8c1e535ef7925345d06
2de7c046ac90852cbc40f3f617c34ccd90861b5cc40d71e8c10a10ad879d1b160acb87892112b180f9e321
12db76c2255a2c887b81e0132318c52926d18aaaab2107b5b845fb75d18b28ab221853f8400094117d1268
1e09d188a01f02508c4224e31be3a8133acaca8e77c462229cf8440a6060507e34a112d7c8203c66d18d6f
bc40ac12a9333532f28ad6cb2324b72849a450b2ff50805ca22307a9471c0660929fac9d252f8949c095f2
8346244f2ac1144a2b8e5235af94602c6749c545b25283997c24113d0080c6f19296abfc652b9b984b029e
f19855aae51a6fd99966a2ef99d0b49134a729485c02e002129480312c984dfa6c939bc15484280210413e
ceb19c6939273a35b1ce02ba139eda4ca6321bd94d44d4d37e3a9c133e51244f4652f310ffcc5e27076aa3
0f706b5ffb44e1410d91d0dd2d94a174fb0509201a5120a633111505c111316a24e79060641df5283d01c0
cedded92a47b620a002280d2942e1313ff7c294c217510008cc0a66c74443d4300b09dcaea27a2f8a94d27
5a887562d3a87f9c8901387a49a612421433ff282a5411679f9a56b59f8669c956677710655a7508be6a1f
a43a50b70f24e09dbc7401ef00803456e6e7a3dbe95ab00c370a63c254aebb130000ec3a0f4b080aaed4c9
89dee406d314644f0274656472aef50886d949b18b652c431d8b3ec1ae71b2afaa2cbd3285d9cc8a82a127
202064a9fa39363d22663135ed62a548c9d44271b04b24d26bdb152a87ca566f7ee5250b5a20410ae80b85
4b62522386264b4859edb75ae55c2836405b9ec080b81ecc1f0093fba03146445bcffdad2739e7d008fc0e
2c806de1713f77d7ee6a32ba990aef6f67078bc0484002cc73de07d42a92203c51bb96bb6bc41aa1b936f1
15ba62355d7d0b400001244f7dddfb0cff4f384b44e3b2363b94654481c1c41604279873c6d04c83ef973f
7290d31c3678238049e65a026b4c5b1df63062ab1462113b58b5be731ef4ea61db325a183b2d668465cfe5
e163cc589b0718d6888b2b8008f785053d56316e51a3db7b8db69245a6c59109fabf257ff00225a6ee70df
a8bdf52e26b943710c6f7d98652dcbf0845ec6618e01400132676fc5dc6d046c55d9e60f3b2e140bc24e9c
8928010b3cc0ceca9b59832262c3710497a77df633e1009da44113d1887546b4100e4000013bc247da2aad
8c151cd94adff88d86d6b4102ed0bd0c2fc218cdcd579fb7aca1055bced24414293811ad021d78aa11c6d8
010c6425dfdfd21a41b6bef5a9dfff788003f09ac1006000248ce16013c437d27d04f1943f876b229eb2a5
658cc1620810ed6903e0c62770c19e8a2ddb63d3a7c6f2eb36119bfd461988b8dc8fa076f64e606d1a63db
ddd41980067cc920799b92a55b7c016ac82ded7c9fdb7ea7dcc0892d13e3516fee8706c721bd718802852f
1cdf9f7e38fa4e190b04f0772e154730c0851390875931e32cfc360b51b082e330dcdccbde1dc90f8798d5
b479e5b4a1341861cec28d4b500535b739c8812df2eced5c6f3b068bcfb30c74d2087de83927b3244170f4
4a2f9d11fa1eb978ab5b90a917b9ea964936d655bdbb0758208217b8f5d75fddf4e43d5db60898b895b09d
6dc7a97ded6c376e0508286effb9373ce459bffb6f35b0df9788dae28ecbc6850148741cbadd7e1e373cce
211e690917c4b791d67ba6e0cdcacab350f0d9cbbce61d9e78be8fb01eec962dd9f744fad2671dd1971702
cd293f774584dde9ae7f4f35626fdad9a38ee080f72005022001f55660f7bc3f3cd35b1f7c934020e50836
3e32919ffc0282a0d5c0c874040fd08397f73e11bfb77bf5d70f5f6d11e734293578fe025d8225edba8007
a82b0a6fce7aceb3dff5da572557d751f2664410a523cd877ffa177d9b2776ff876d016824034880b73757
```

```
4a452c87a680e6277d6057773af78091168136f277145840c4942e74a6814bc47f88e77f20586422882224
588204840117d8291ea0ff82fb777e88907e1ff8821ee63245630c5e0554b8065934831fce4640f9b7810d
087c40586451272bb507540c767ba76439c9a1830cd87f0e188545a601df93291cb26d56386eb7674440c4
85f1c68387e08342a078608860d4d54b676873b727002ec74f6ca83a2c387d2e38877d36851a5250f19686
6678457dc86d6e6808702887821886638820137887684840599824 25b049e8d3842bd88885f08891587d75
985833658982767bc6653923b084f6e3893bc88174477da3587d84a816a17083a868630564010b482c21b0
88abd7825f588bd5278627971433b88bdd4654fc747f9df88b7e088a84208ac6f880a54815cbc88c15f863
36e78aaf288d8c28ff8bbee78171788d40788b22610c49b88b7818419e751bf9018dd1e8845e0885e8f882
c8982089e88e0b57816d67663a3278e3278ec30888c5988f2f286613010bfea86c1e2458a360015c2741b0
d885c4888f0a1985ec740227e00326109229e09142a0870f696a1f2401015001e0669106e975e4887ee608
891bc97e2d4940267992a9c876d973916d18933d389335098637693f39a993b7617a1af79205478d83608d
43f98245d9597b8894bcc893bbe393d308946f2894512995115995567985589995 4c29684e290850f995ff
478f46299656a9944577964ac7958ee8956cf97f1f749463499665a995e3f884ea97977a19967d79955809
984932597619ff8a784998d5b79770899472f9418a692dbf86901a0999ae074bfda89395e941978949a175
8f83c999c1f741997898a1e99200340aa9528eb4889a7ca79a9f7992ad5990f2536596b096b41969b67998
97f89774c99bbdf998bfd967c1299cb9298c9a81669ae09bc99965cb79980b207e3c399a79169dc8399d45
569d8759966689258cb609d2e99d08069e56991c0f5091d969903aa2349c709ee8f95beaa99393350aedc9
7680e9699d409ff599596fe741ab899477d56cc7f0006ef9445ae99fffd99d016a5ae05840057a92078a0b
13c000ff220a1610020b6a99bf380285f509001aa108719fee78a18520131b0a001dfaa15c085aa950a226
1a14288aff8a2aaa1a08d0a22f3a979a015a80e209345aa3c843a0b779860e2a443b4a0b18e001d8198e8b
01a4aa30a4353aa19878a44095a48ea036c7d0a44f3a9e05200a1910a4240aa144ca1a376a855a6a095cca
a44eda9348639ca840a5266aa59ce78fe4e66a9ad0a6a3800115d07c34909f6936a7667aa65891a629955c
03860a1c100dac7103723aa3e6284e869a59762a76bba8a8df92008e4a0bee35a5e6b83bdb53a9276aa4a8
a8a9e2c1a9e5e90a70883e2144aa452a41156a53dbb90c0a109b840a900a5562677aa9c07787b52a19e320
005faa5a1e007e11aaab1f7886c12a198af500538989bc5a9fca7a8e56d8aca131012561017f7a7a4d46ad
88caff480251220a90002d1a02c54a4020807d7959adb3fa4be31a2503d0a9c010adf633aabfe9ae586a45
f11a2582a0adb460011e9080b0c4ae1ba9af1dd5affeea223d250a0f90ae0035ad5f89b0fb14a90bdb2425
7100eee941bd83ac3549b1f0da1217bb08191ab0036b4a129b8f207b49163bb287d0b000a0a0382401dfaa
902b0b469b28b22eab42194b902cb4ae2a1bae5ad8b23bab080aa0a113290004ab7c12598b378b42825ab4
e5c216d07a6906cb7e3e1b41efaa3a512bb50f51127e5a61290b82f6aa7e6034591930a85e8b09e5daa2c4
3ab3af0a8465bbac4b24a56bcba8543bb74e37b6c1a7b75b8b829979b79f00b013d9ad2c845f1eeb7a7ebb
ffaf3f1a6482ab0a6d4b0be8ea6d578b608b0b40221ab88fbb0af3ca1a7aab3c358b6d97ab3a39bab9ba40
b81c7ab27256b98b35ba01e6b8a66b2ae64a0b0f5b6144d566aeeb21b01bbbbc2013acb1b11c1bb7968ba8
77250264cabbcb50b213a9ba39c4b7c790bbf238a2c80b0e302bb3337bacb205bd9ab1a6d3eb0dbe1b1459
ab9a7cabbd8ba6a7dd0b0e47bba116a0b41516bad6f4ab4929bde7fb0fd50ba3ea0a14316786a83abf0ef1
bda210b6b6cb895abb6dfbcbbf1a11b9a3f0b6e2891a056cc01c5129b4fbb944f49cf2ebc02a81ba00e0a7
4b6b6767b6aa164c14082c0a938b684ae2c11ffc148deab9642601d87ac24f81c1dcbac11c2b429fea39c2
5a11c2310bb10a45c3365c189d1b14c0fb8535dcc36ea1bca94bb02a995f264cc48501b396aab64cecc3cf
8511514cae910339c7ebc28100003b)
INSERT INTO [dbo].[WidgetDescriptions] ([WidgetID], [ShortDescription], [Description],
[Picture]) VALUES (2, N'A medium-sized widget', N'<description
xmlns="http://www.red-gate.com/widgets/WidgetDescriptionSchema">
  <descriptiveText>This is a medium-sized widget, ideally suited for the home and
garden environment</descriptiveText>
  <manufacturer>Magic Widget Company</manufacturer>
  <countryOfOrigin>USA</countryOfOrigin>
  <size length="15" width="12" height="2" />
</description>',
0x474946383961d100ce00e64000000000ffffff465863cccccc6984953333339999996666660 32d51010f
1bdfdfdf84a5ba4f4f4f1f1f1f7f7f7fafafaf003366111618586e7c0116287b9aae003333202020e0e0e0
000b1434424a8080806699991a2125b0b0b03d4d575050504f636f607988021e36e2ebf1b7cedc84a9c194
b5ca99cccc03294ae9f0f40000336b93b01e517dd4e1ea336699739bb65a85a5021a2fcddce6c6d8e3bed3
df97aebc336666a2abb1617d8f11191f164a7849769a666699dbe6ed1a242c8db1c7ffffff000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000021f90401000040002c00000000d100ce000007ff8040828384858687888 98a8b
8c8d8e8f90918701061f059797071a1d01929e9fa0a1a2a3a4a3010500a9aaab0007a5afb0b1b2b3a20316
```

a90c0ebaba0ca8a91ab4c1c2c3c4a103aa0601cacbcba9058703c5d2d3d4b106aa03ccda07bf85a7a90717
d5e3e4e589d7a9d9dacbdcdd83dfabe1e6f3f4d2ed050aebcbbe1309c082b6521968074e5cbd83084dddd3
b72fd50408fe00225be660d5bf8418333282778061800 1be44408008ec98c07515535988a6b1a54b20015b
793409a0c24892e800a8d347b0e3cb9f07692663f84095cd9b097ce9f4a86c4003672c814aad96f381c79c
376faa2ab0936900821da68a2d46b0eb365559476270e6755d518163ffe3cafacad6633b0c69216c6dab0f
2438b9804df962c0945f5e551df932f4e52ab0e3483113f77d0ae0615ac48a0bff7dcc7991d099460f83cb
ecd59781cea80d55bdaa0a85680043496bd699ba7659bb685fc796cd5465a7da8fe90228c0f46ede0aaaac
f26e6bf203c20b9c803f8247d8a32f15c7b12d579c32aa390d88c34a574453f23aa70eb3a7db9ef9a98579
1794aab2206f7ca10e1319d23c9a15f970b3ec3165d269e4e087cb0006dcb25532f601911380cbe4e45a5a
13d4156066a83c538d70b031330041a97c209e6da93400a1322901901704150e77a16c2679474c4c5c11c5
002b07c808183cc4e106005e79b558dd8ba43dd5583139997795ff7c161860905cf18de6d18d0024b0e25a
3211295b4ac5c0b354660a1c4099330c8a459303b3599616964a6ac9578cc31808c090bc7d88e3882fe5b4
db3a86e585259a6ef206172d1ceeb99d0154aa549f46e0ad675d2afc65955fa0b2dd78642c7efd17a80206
8c395c990709878f8090ae3829a5a455a4a13598a1da14880088580f8f6da564ea49aeca864e2c017c905c
aeda3c9028383a4a13195fe8448a8076c0ae938f7ea9c042538fcdae6300934e52f399626326a002965f56
bbcc31ca9d176d2904192aee32617afa01a8c1ac9659a6ab98b86e84738206002999da7baf80b0e6184c3b
fec2e8c001b9947b2f378042bbaf2849fe9bd9b5abd0f724ff29b44a4ca402a8a8a36e006f8922a7c21ab7
c5a9a705c0fb499459961ce031a3720cc0b3daa80a712af99ce8b27eb0cafac9993b07784d62309fc8cdaa
3fa7620209b8901cb447c2e258ac227a3ebd1d37435d43279f005c9434003ffcb080048c396db5b5d85e9c
48a3e19ead985fea30ec552a787a6252d8782f000207003490a3db1eb5bbcabbbf19222acd80b75554cca8
28778ca16fa92dc9dd78574e810011f46d00e2892ff340c0d910425de7dc6559b432d750cbccd1a3505ef9
eb1b789079caa4a35def01392bd866edebdc9875cbdcec1e8091ada7f2faf1788790012e1f037ef22a4a35
cffb471d2b8375431fc3598bf1c877ff4308654fff2aff2b3a4fbf783e32ab733a4337be573cd8de7b3f36
63e59f2d2c03f5d75e11d1c33dab757183da1efce2173f0aecad6f7f135f0014d00ab391ce7728021e001a
a60f5f38875fdc23a0062920bb4f71ae73146b80033ee836b8290382ca68dc6c2e25400dba106fb1838af8
0447bbc49d8f7a5f5a5f5f54e1b5f7bdf08762cb9c033bf739629d6d7fcab82125f2053564602a83408cdf
02f8963f10a24284245c170a9168bd0932e541b2705d14bbb78108144c81ceba87f45065c200a050664314
ce4a6621c631c2ce4568541c9512582d0554a78d3ad4079588438b3ada316c1260621e1513c2cdad4b89ff
0b1c630a1746281ef2072068d922e7d50e06acf1ff425ca40b05cde58e6018728ccbfbe426a1b6c72a6ea9
430b54216b02684a4b467101a850e52a99f2bc1aba497d9af2488aa6168b0ba46203879c62db76f9b24ec6
913791f4510124178c5410c08e653c2333898428235ee87ab3f900258961cd316e008fdb4495021cf0145f
f2a67afa6ac538ca09c4446e2d9d6c6cc7019ec9932ca2ce89f304c0357f9849e1e1135521c49d9686590e
7aba30950795d83adba94b6952531a0e95622e23ea323bb5c295dab0e038ab91d1ee2913a41c75550847f8
36c4d4a3a4c7cb264a539a2b1aae514f078169e5cea93a9ada4f9f102a0b42748a4845fad48a9aa3d9202f
4a0ea216f4a8e2f368477c41c884e814a250ff45e303ac023d9599a3a4b884655617e9d1f0bc54a0953be9
58b7d9cdf92c2aa003fd814cd78acfe72d68a4e444ab5cd149d783963544751b4639edd9d7a3462d1ec484
85351369d0c21e9462f3c95635052a56c766557000ad64652d4bd7b2e25580cb74db0376e1809972561f27
a040294b3132d2a5a85ea33cad62527082b02d0f009ff5c4833a472f56c850b6b33501de54dbb552ec3671
34f12df980db961608b7721e4805531fb110de2af7ba3d65ae365a80bc056c0614d545ee75c7cb4fced2c0
7b0290ee277a850bd22577bcac88ad764920c5ef428223ee85ef782b7a3606f0514bb525607a0130dd494c
1284fa1def2a51910304fa93340126a0ff77e5f9886d252e27095e6e1e65068115707076fcd5460a9eebc2
0127961016061c8633cc2c341e032f088800de08705b06949719238ee2ec1a71dc0bb3d8b7a6ede88f46b2
acd7cdcf9b5ee1ee1813491baab51870b0fa71904b760d3515197906cc5c031cc910195c72c7e728d19481
1565168f596255ceca95bd17c339198abe9764f2d4c29bb83267f8ccffba460cd2b2660282cf9b70bee40f
c05c083a034e3e3f0e2defb81129220fd08563e39bdf048d37398beec09d43b49417c9e815f5f9871c0480
04281d36beb90f08f8259da6cdbc485420604510f8341033206352130 02ea946eeaaefdc6a15c13ad68f8e
a26a474d6a54bc6740f94df42a42ffdc5154fcdad182a635a97f80837498a405d655f6a9a6f7e26743fb92
aa1500a54bc0026b1b4f06807b8bb637cbbb6e7b1bd8944eef022ef98291981b7e2738db8a13cd6c2a57e6
dddfbea477c56dc715684527622441d0f6fde37e6b2ccd008777bc0130ef28c2405208b7e5096ec0e575d9
39c30e975845441271890b7ae051dcc165321eecb00dd893e2fa788243feaf4e97dce48296f70f5d209a01
9c72c08879708064ae5f9a2fac26370ff8c9014070029680e73dffb972dd49a45d833c8fce4ebad2734e71
a7b300d6f7461ed095fb5ff6587de658f7b5d6717ef20878207e2fd0c1afc37ebcb15f77cb429fcdba8d2e
2ea70069ed6c17742229d0bdff7a3f9beeafb33b7c6d0ca3756fbb76ee067ce0051d810c20efe2de467ce5
149f608532c7f1b88aea9025bf754a0ffe7598cf3ccbc5ee78aa6be3bdcae67bb5204efac953def27883fa
bb358f37ce27baec4b04bdec9b45fbdacbdaf40020fcd34bce7b9783be5eb16178c3d168f3da97bebb0420
40c509587972dfbcf93ff0bde3fdfbfce19319e9d6bfc9f1d37adb547860fbdd13800fbefefdd5d7fdf9f8
f7adf97355fdf4db3e6ce7847ec8c101f05777569274e0277ef9177b689475fea77e2d97371cf0775a5100
f1230007587f3e674be1b780f9275f99a6760f787dce0776c4267619c87cf697781ef87c8d753691378224
f80300407241526b28a8ff7509d8828ef78256138332687b21208287814c3988802bb8793ca86dd9454455

```
12847c16813fe0012978184d97785508703bb884da067cfaf66f50a86652580036b82213608147a88152c7
85eb86452a0686610881dd837eb026026868803a9884bdc786cfe77a1ad37f71687b00f06abf6687f19785
bba787cec787f8e78531478781488283f86c8698862ab88111a8808cf8636e782f373221912887c8c30165
9817958887488889acb7891ee88794e280a1288ac753006ab22209f0769618715bc88a2de8882f028bb148
82e9757827888579a88af7c78b4bd889440284c1687be74488d95180bd8788aa878c2ca88c6ce88aa4e18c
cfb87e3f5000d608ff00b8978b5aa8881da88d8ce88b5e710dd8f18c79018eb7d668101003c9875ed67878
e8a889ea987fcc882c70088f627881902802a27681f93877fbd88fcac88d1e129002298bf1f72322609007
8990c7b8860ca98c3540033330022019923d300327706b11e9695268391970091e407802969060b7901ba9
8c46e83d26799251386dc6988a1a39939ba84137899312a9931898911cc88f3e997f40498438098e975494
3c799449c98a4b299403a993d568949938953f49404169954e794850a98652c9957c58955639834f0993b7
828d4a68966c889669199676349697d89370d98272099629b9964848607899970bb8974249976364978948
1b812998f847ff984dd99762c9963781252cb1988c097a8e799286194588f96b58422030910ad4889497c9
629919919b09449d7925b4040430459aa59960a72990a9f943ab991609d09aaea9576f199bcfc701b3098f
b5f942b779132dc24282f09abe897f7768934c89932b809559f96bc78908cab99ca0d79cddf39556199dd2
197ef9688f147608d7899deba69dc8c39d27e91f191002d2799bfe813486509ee6a96ce8793cea2990feb1
0a1e5093398788f1995bbb19577b589f4c189ca1b89f07a020a9e076fe79985918a08b409f06ca62f7f93a
f9198cfb59990bba0a0e5a9759e81b8c40a1159a60175a39199aa0ce20391fc2a000e0762e499c55880d8d
40a2250aff5f273a63cfa9a22e421e1daa0a1c0002318a9107d76423ca9b057aa31996a36193a27128a18d
d001b012a4438a8a7a910e8f60a34a3a75081a84500a0952ca0a547a845812588aa0a55bca0a4cfa034e1a
842d52559f10a6d02301f057949f1909689aa65bd1a50ff8a6020aa6be32a7f3c6032870a7788aa48ba8a7
e37585dbb9a36eca16bc52096a6a032dd2438e005343a8a88bcaa7d6e7a7c14009811a0f9fa0537a93399a
ca0a8c9a9e8edaa7f6250ca0daaa9040544dda7ea79aaaf8b9aae9579dd570012736a1888a659853ab9c0a
78ba2a17ce50a5def3677a6aab188aaba4a702b02a16d0e39e2e146a5bcaac28eaac80c7268f71016cf3a2
2040ff8ddd23017c53a2d8aaa350c8ada80159a990010fca66b48a9de7daa4da9a74ea5a1b7f5500c52845
1260aacb39af6c5aaf258725969a1adecaa0112000e29a9ef1ca9800dba6f6aa9be3c1aee4f8ae2615ac97
f9b002fb6e86da2085f0571cb0aff1a3ac7929b2b7ea7f1debb186a3012e2a00c8da3dd66a9604daa8e997
592a9b081d201f1930b3f143365cc9b3275b7b367bb38a70012012b2a0d6b0ea08b4cd2ab4124bb4894009
2eda922f14691bc9b4d94a7a430bb58ed001a13a1c588b9f4acb88614baf9247a35cbb32201201745ab520
e0af9b58b6010b78689bb6a020b5fcf9b2dd43b27c28b7107b2b466ab7b5f0b50540add50a9bdae6b71b0b
ff6c582ab8afe0ad1e1aae3fe4b34ba8b837b7a18e3b0b149b017a8b65d1d582961b71989bb9850422fafa
43569b7fa1fb6ea34bbac100b9aa90b00b8b3c1b30b63fb6bacfd6baae3b0c9b6bb126f5b68ee74210fba5
bb5b0cf96ab2367976d725bcb84abcc52b0d01c0b2abe0b2a086b119c6bcb90ba9cf5b0e9b2bb795c3b7e3
85bdb0e6bcdb3b0e46bb0a480b6a9f1bbea7e9a9e53b2b0992b7f554aebe25be14a2bdef8b10390b3dde0b
430deb42e71424f89bbf0971beb1dbb66eebaf2f3489c639c0048c1178ab0a54fb43340699782382c5fac0
2f3100846bb8d2991519acc13f61c00d2ab958d9c0e329c262b1b99d1b452311c22a6c26a68bbc40045511
d11ac36201bb0daab0c974af38fc18bd7b4be58a9c3fec18c75bb5435cc4d211bd084bbdde430149acc4f6
d1bdaf4301cab3b5520c1c24ec0c0540bfaa60a659bcc4f13b755e15c61e8b201a90c61dd0abf91b08003b
)
INSERT INTO [dbo].[WidgetDescriptions] ([WidgetID], [ShortDescription], [Description],
[Picture]) VALUES (3, N'A small widget', N'<description
xmlns="http://www.red-gate.com/widgets/WidgetDescriptionSchema">
  <descriptiveText>This widget has been discontinued. It was last sold in
2006.</descriptiveText>
  <manufacturer>Magic Widget Company</manufacturer>
  <countryOfOrigin>United Kingdom</countryOfOrigin>
  <size length="3" width="2" height="1" />
</description>',
0x474946383961d100ce00e64000000000ffffff465863cccccc6984953333339999996666660032d51010f
1bdfdfdf84a5ba4f4f4f1f1f1f7f7f7fafafaf003366111618586e7c0116287b9aae003333202020e0e0e0
000b1434424a8080806699991a2125b0b0b03d4d575050504f636f607988021e36e2ebf1b7cedc84a9c194
b5ca99cccc03294ae9f0f40000336b93b01e517dd4e1ea336699739bb65a85a5021a2fcddce6c6d8e3bed3
df97aebc336666a2abb1617d8f11191f164a7849769a666699dbe6ed1a242c8db1c7ffffff000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000021f90401000040002c00000000d100ce000007ff804082838485868788898a8b
8c8d8e8f90918701061f059797071a1d01929e9fa0a1a2a3a4a3010500a9aaab0007a5afb0b1b2b3a20316
a90c0ebaba0ca8a91ab4c1c2c3c4a103aa0601cacbcba9058703c5d2d3d4b106aa03ccda07bf85a7a90717
```

d5e3e4e589d7a9d9dacbdcdd83dfabe1e6f3f4d2ed050aebcbbe1309c082b6521968074e5cbd83084dddd3
b72fd50408fe00225be660d5bf84183332827780618001be44408008ec98c07515535988a6b1a54b20015b
793409a0c24892e800a8d347b0e3cb9f07692663f84095cd9b097ce9f4a86c4003672c814aad96f381c79c
376faa2ab0936900821da68a2d46b0eb365559476270e6755d518163ffe3cafacad6633b0c69216c6dab0f
2438b9804df962c0945f5e551df932f4e52ab0e3483113f77d0ae0615ac48a0bff7dcc7991d099460f83cb
ecd59781cea80d55bdaa0a85680043496bd699ba7659bb685fc796cd5465a7da8fe90228c0f46ede0aaaac
f26e6bf203c20b9c803f8247d8a32f15c7b12d579c32aa390d88c34a574453f23aa70eb3a7db9ef9a98579
1794aab2206f7ca10e1319d23c9a15f970b3ec3165d269e4e087cb0006dcb25532f601911380cbe4e45a5a
13d4156066a83c538d70b031330041a97c209e6da93400a1322901901704150e77a16c2679474c4c5c11c5
002b07c808183cc4e106005e79b558dd8ba43dd5583139997795ff7c161860905cf18de6d18d0024b0e25a
3211295b4ac5c0b354660a1c4099330c8a459303b3599616964a6ac9578cc31808c090bc7d88e3882fe5b4
db3a86e585259a6ef206172d1ceeb99d0154aa549f46e0ad675d2afc65955fa0b2dd78642c7efd17a80206
8c395c990709878f8090ae3829a5a455a4a13598a1da14880088580f8f6da564ea49aeca864e2c017c905c
aeda3c9028383a4a13195fe8448a8076c0ae938f7ea9c042538fcdae6300934e52f399626326a002965f56
bbcc31ca9d176d2904192aee32617afa01a8c1ac9659a6ab98b86e84738206002999da7baf80b0e6184c3b
fec2e8c001b9947b2f378042bbaf2849fe9bd9b5abd0f724ff29b44a4ca402a8a8a36e006f8922a7c21ab7
c5a9a705c0fb499459961ce031a3720cc0b3daa80a712af99ce8b27eb0cafac9993b07784d62309fc8cdaa
3fa7620209b8901cb447c2e258ac227a3ebd1d37435d43279f005c9434003ffcb080048c396db5b5d85e9c
48a3e19ead985fea30ec552a787a6252d8782f000207003490a3db1eb5bbcabbbf19222acd80b75554cca8
28778ca16fa92dc9dd78574e810011f46d00e2892ff340c0d910425de7dc6559b432d750cbccd1a3505ef9
eb1b789079caa4a35def01392bd866edebdc9875cbdcec1e8091ada7f2faf1788790012e1f037ef22a4a35
cffb471d2b8375431fc3598bf1c877ff4308654fff2aff2b3a4fbf783e32ab733a4337be573cd8de7b3f36
63e59f2d2c03f5d75e11d1c33dab757183da1efce2173f0aecad6f7f135f0014d00ab391ce7728021e001a
a60f5f38875fdc23a0062920bb4f71ae73146b80033ee836b8290382ca68dc6c2e25400dba106fb1838af8
0447bbc49d8f7a5f5a5f5f54e1b5f7bdf08762cb9c033bf739629d6d7fcab82125f2053564602a83408cdf
02f8963f10a24284245c170a9168bd0932e541b2705d14bbb78108144c81ceba87f45065c200a050664314
ce4a6621c631c2ce4568541c9512582d0554a78d3ad4079588438b3ada316c1260621e1513c2cdad4b89ff
0b1c630a1746281ef2072068d922e7d50e06acf1ff425ca40b05cde58e6018728ccbfbe426a1b6c72a6ea9
430b54216b02684a4b467101a850e52a99f2bc1aba497d9af2488aa6168b0ba46203879c62db76f9b24ec6
913791f4510124178c5410c08e653c2333898428235ee87ab3f900258961cd316e008fdb4495021cf0145f
f2a67afa6ac538ca09c4446e2d9d6c6cc7019ec9932ca2ce89f304c0357f9849e1e1135521c49d9686590e
7aba30950795d83adba94b6952531a0e95622e23ea323bb5c295dab0e038ab91d1ee2913a41c75550847f8
36c4d4a3a4c7cb264a539a2b1aae514f078169e5cea93a9ada4f9f102a0b42748a4845fad48a9aa3d9202f
4a0ea216f4a8e2f368477c41c884e814a250ff45e303ac023d9599a3a4b884655617e9d1f0bc54a0953be9
58b7d9cdf92c2aa003fd814cd78acfe72d68a4e444ab5cd149d783963544751b4639edd9d7a3462d1ec484
85351369d0c21e9462f3c95635052a56c766557000ad64652d4bd7b2e25580cb74db0376e1809972561f27
a040294b3132d2a5a85ea33cad62527082b02d0f009ff5c4833a472f56c850b6b33501de54dbb552ec3671
34f12df980db961608b7721e4805531fb110de2af7ba3d65ae365a80bc056c0614d545ee75c7cb4fced2c0
7b0290ee277a850bd22577bcac88ad764920c5ef428223ee85ef782b7a3606f0514bb525607a0130dd494c
1284fa1def2a51910304fa93340126a0ff77e5f9886d252e27095e6e1e65068115707076fcd5460a9eebc2
0127961016061c8633cc2c341e032f088800de08705b06949719238ee2ec1a71dc0bb3d8b7a6ede88f46b2
acd7cdcf9b5ee1ee1813491baab51870b0fa71904b760d3515197906cc5c031cc910195c72c7e728d19481
1565168f596255ceca95bd17c339198abe9764f2d4c29bb83267f8ccffba460cd2b2660282cf9b70bee40f
c05c083a034e3e3f0e2defb81129220fd08563e39bdf048d37398beec09d43b49417c9e815f5f9871c0480
04281d36beb90f08f8259da6cdbc485420604510f8341033206352130002ea946eeaaefdc6a15c13ad68f8e
a26a474d6a54bc6740f94df42a42ffdc5154fcdad182a635a97f80837498a405d655f6a9a6f7e26743fb92
aa1500a54bc0026b1b4f06807b8bb637cbbb6e7b1bd8944eef022ef98291981b7e2738db8a13cd6c2a57e6
dddfbea477c56dc715684527622441d0f6fde37e6b2ccd008777bc0130ef28c2405208b7e5096ec0e575d9
39c30e975845441271890b7ae051dcc165321eecb00dd893e2fa788243feaf4e97dce48296f70f5d209a01
9c72c08879708064ae5f9a2fac26370ff8c9014070029680e73dffb972dd49a45d833c8fce4ebad2734e71
a7b300d6f7461ed095fb5ff6587de658f7b5d6717ef20878207e2fd0c1afc37ebcb15f77cb429fcdba8d2e
2ea70069ed6c17742229d0bdff7a3f9beeafb33b7c6d0ca3756fbb76ee067ce0051d810c20efe2de467ce5
149f608532c7f1b88aea9025bf754a0ffe7598cf3ccbc5ee78aa6be3bdcae67bb5204efac953def27883fa
bb358f37ce27baec4b04bdec9b45fbdacbdaf40020fcd34bce7b9783be5eb16178c3d168f3da97bebb0420
40c509587972dfbcf93ff0bde3fdfbfce19319e9d6bfc9f1d37adb547860fbdd13800fbefefdd5d7fdf9f8
f7adf97355fdf4db3e6ce7847ec8c101f05777569274e0277ef9177b689475fea77e2d97371cf0775a5100
f1230007587f3e674be1b780f9275f99a6760f787dce0776c4267619c87cf697781ef87c8d753691378224
f80300407241526b28a8ff7509d8828ef78256138332687b21208287814c3988802bb8793ca86dd9454455
12847c16813fe0012978184d97785508703bb884da067cfaf66f50a86652580036b82213608147a88152c7

```
85eb86452a0686610881dd837eb026026868803a9884bdc786cfe77a1ad37f71687b00f06abf6687f19785
bba787cec787f8e78531478781488283f86c8698862ab88111a8808cf8636e782f373221912887c8c30165
9817958887488889acb7891ee88794e280a1288ac753006ab22209f0769618715bc88a2de8882f028bb148
82e9757827888579a88af7c78b4bd889440284c1687be74488d95180bd8788aa878c2ca88c6ce88aa4e18c
cfb87e3f5000d608ff00b8978b5aa8881da88d8ce88b5e710dd8f18c79018eb7d668101003c9875ed67878
e8a889ea987fcc882c70088f627881902802a27681f93877fbd88fcac88d1e129002298bf1f72322609007
8990c7b8860ca98c3540033330022019923d300327706b11e9695268391970091e407802969060b7901ba9
8c46e83d26799251386dc6988a1a39939ba84137899312a9931898911cc88f3e997f40498438098e975494
3c799449c98a4b299403a993d568949938953f49404169954e794850a98652c9957c58955639834f0993b7
828d4a68966c889669199676349697d89370d98272099629b9964848607899970bb8974249976364978948
1b812998f847ff984dd99762c9963781252cb1988c097a8e799286194588f96b58422030910ad4889497c9
629919919b09449d7925b4040430459aa59960a72990a9f943ab991609d09aaea9576f199bcfc701b3098f
b5f942b779132dc24282f09abe897f7768934c89932b809559f96bc78908cab99ca0d79cddf39556199dd2
197ef9688f147608d7899deba69dc8c39d27e91f191002d2799bfe813486509ee6a96ce8793cea2990feb1
0a1e5093398788f1995bbb19577b589f4c189ca1b89f07a020a9e076fe79985918a08b409f06ca62f7f93a
f9198cfb59990bba0a0e5a9759e81b8c40a1159a60175a39199aa0ce20391fc2a000e0762e499c55880d8d
40a2250aff5f273a63cfa9a22e421e1daa0a1c0002318a9107d76423ca9b057aa31996a36193a27128a18d
d001b012a4438a8a7a910e8f60a34a3a75081a84500a0952ca0a547a845812588aa0a55bca0a4cfa034e1a
842d52559f10a6d02301f057949f1909689aa65bd1a50ff8a6020aa6be32a7f3c6032870a7788aa48ba8a7
e37585dbb9a36eca16bc52096a6a032dd2438e005343a8a88bcaa7d6e7a7c14009811a0f9fa0537a93399a
ca0a8c9a9e8edaa7f6250ca0daaa9040544dda7ea79aaaf8b9aae9579dd570012736a1888a659853ab9c0a
78ba2a17ce50a5def3677a6aab188aaba4a702b02a16d0e39e2e146a5bcaac28eaac80c7268f71016cf3a2
2040ff8ddd23017c53a2d8aaa350c8ada80159a990010fca66b48a9de7daa4da9a74ea5a1b7f5500c52845
1260aacb39af6c5aaf258725969a1adecaa0112000e29a9ef1ca9800dba6f6aa9be3c1aee4f8ae2615ac97
f9b002fb6e86da2085f0571cb0aff1a3ac7929b2b7ea7f1debb186a3012e2a00c8da3dd66a9604daa8e997
592a9b081d201f1930b3f143365cc9b3275b7b367bb38a70012012b2a0d6b0ea08b4cd2ab4124bb4894009
2eda922f14691bc9b4d94a7a430bb58ed001a13a1c588b9f4acb88614baf9247a35cbb32201201745ab520
e0af9b58b6010b78689bb6a020b5fcf9b2dd43b27c28b7107b2b466ab7b5f0b50540add50a9bdae6b71b0b
ff6c582ab8afe0ad1e1aae3fe4b34ba8b837b7a18e3b0b149b017a8b65d1d582961b71989bb9850422fafa
43569b7fa1fb6ea34bbac100b9aa90b00b8b3c1b30b63fb6bacfd6baae3b0c9b6bb126f5b68ee74210fba5
bb5b0cf96ab2367976d725bcb84abcc52b0d01c0b2abe0b2a086b119c6bcb90ba9cf5b0e9b2bb795c3b7e3
85bdb0e6bcdb3b0e46bb0a480b6a9f1bbea7e9a9e53b2b0992b7f554aebe25be14a2bdef8b10390b3dde0b
430deb42e71424f89bbf0971beb1dbb66eebaf2f3489c639c0048c1178ab0a54fb43340699782382c5fac0
2f3100846bb8d2991519acc13f61c00d2ab958d9c0e329c262b1b99d1b452311c22a6c26a68bbc40045511
d11ac36201bb0daab0c974af38fc18bd7b4be58a9c3fec18c75bb5435cc4d211bd084bbdde430149acc4f6
d1bdaf4301cab3b5520c1c24ec0c0540bfaa60a659bcc4f13b755e15c61e8b201a90c61dd0abf91b08003b
)

-- Populate Contacts
SET IDENTITY_INSERT [dbo].[Contacts] ON
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (1, N'Christopher Martin',
N'01446 175923', N'07634 717173', N'17 High Street', N'Darlington', N'Cheshire',
'20070914 16:09:21.383', N'Christopher.Martin@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (2, N'Joseph Bennett', N'01147
771824', N'07956 758471', N'37 Green Lane', N'Portsmouth', N'County Durham', '20070914
16:09:21.400', N'Joseph.Bennett@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (3, N'Joshua Scott', N'01940
919059', N'07331 757163', N'33 George Street', N'Llandudno', N'County Durham',
'20070914 16:09:21.400', N'Joshua.Scott@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (4, N'Daniel Powell', N'01659
216066', N'07272 134093', N'93 North Street', N'Newport', N'Cheshire', '20070914
16:09:21.417', N'Daniel.Powell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (5, N'Jennifer Washington',
N'01502 976179', N'07225 483640', N'43 The Grove', N'Hull', N'Berkshire', '20070914
```

```
16:09:21.430', N'Jennifer.Washington@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (6, N'Anthony Lee', N'01200
660059', N'07696 849723', N'97 Stanley Road', N'Newport', N'Berkshire', '20070914
16:09:21.430', N'Anthony.Lee@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (7, N'Rebecca Foster', N'01340
503829', N'07960 669272', N'79 Springfield Road', N'Salisbury', N'Licolnshire',
'20070914 16:09:21.447', N'Rebecca.Foster@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (8, N'Ava Walker', N'01978
657197', N'07610 445571', N'29 North Street', N'Southall', N'Oxfordshire', '20070914
16:09:21.463', N'Ava.Walker@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (9, N'Matthew Ward', N'01324
940147', N'07903 833764', N'71 Kingsway', N'London EC', N'Staffordshire', '20070914
16:09:21.477', N'Matthew.Ward@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (10, N'Matthew Mitchell',
N'01661 433253', N'07822 453924', N'70 Cowley Street', N'Worcester', N'Rutland',
'20070914 16:09:21.477', N'Matthew.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (11, N'Megan Stewart', N'01653
610021', N'07363 920914', N'59 Broadway', N'Chelmsford', N'Berkshire', '20070914
16:09:21.493', N'Megan.Stewart@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (12, N'Mia Robinson', N'01594
360107', N'07295 889232', N'22 The Avenue', N'Wigan', N'Norfolk', '20070914
16:09:21.510', N'Mia.Robinson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (13, N'Cameron Davis', N'01594
360107', N'07295 889232', N'35 Windsor Road', N'Watford', N'Rutland', '20070914
16:09:21.523', N'Cameron.Davis@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (14, N'Emma Howard', N'01788
915553', N'07736 274237', N'43 Park Avenue', N'Wigan', N'Derbyshire', '20070914
16:09:21.540', N'Emma.Howard@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (15, N'Liam Taylor', N'01286
103534', N'07939 391131', N'47 York Road', N'Harrow', N'Shropshire', '20070914
16:09:21.540', N'Liam.Taylor@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (16, N'Ross Ramirez', N'01945
464335', N'07126 593543', N'33 Kingsway', N'London WC', N'Herefordshire', '20070914
16:09:21.557', N'Ross.Ramirez@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (17, N'Sarah Lewis', N'01287
924924', N'07149 593067', N'91 George Street', N'Bristol', N'Cumberland', '20070914
16:09:21.570', N'Sarah.Lewis@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (18, N'Emma Taylor', N'01287
924924', N'07149 593067', N'92 Queen Street', N'Dundee', N'Kent', '20070914
16:09:21.587', N'Emma.Taylor@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (19, N'Anthony Rivera', N'01788
915553', N'07736 274237', N'97 Church Road', N'Southampton', N'Huntingdonshire',
'20070914 16:09:21.587', N'Anthony.Rivera@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (20, N'Ellie King', N'01945
```

```
464335', N'07126 593543', N'29 George Street', N'Lincoln', N'Lancashire', '20070914
16:09:21.603', N'Ellie.King@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (21, N'Megan Barnes', N'01469
486350', N'07759 520189', N'92 West Street', N'York', N'Surrey', '20070914
16:09:21.620', N'Megan.Barnes@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (22, N'Scott Watson', N'01630
240971', N'07886 967048', N'91 Mill Road', N'Chester', N'Norfolk', '20070914
16:09:21.633', N'Scott.Watson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (23, N'Emily Anderson', N'01487
403649', N'07449 889487', N'100 George Street', N'Dorchester', N'Hertfordshire',
'20070914 16:09:21.633', N'Emily.Anderson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (24, N'Nicole Martin', N'01721
599988', N'07525 825680', N'62 Kingsway', N'Colchester', N'Hertfordshire', '20070914
16:09:21.650', N'Nicole.Martin@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (25, N'David Perry', N'01212
359994', N'07270 368607', N'59 Queen Street', N'London North', N'Cheshire', '20070914
16:09:21.667', N'David.Perry@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (26, N'Abbie Rogers', N'01120
243659', N'07237 275885', N'62 Manchester Road', N'Kirkcaldy', N'Cornwall', '20070914
16:09:21.680', N'Abbie.Rogers@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (27, N'Liam Kelly', N'01140
882420', N'07354 721715', N'88 St. John''s Road', N'Manchester', N'Northamptonshire',
'20070914 16:09:21.680', N'Liam.Kelly@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (28, N'Christopher Walker',
N'01889 655242', N'07332 289915', N'73 North Street', N'Paisley', N'Yorkshire',
'20070914 16:09:21.697', N'Christopher.Walker@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (29, N'Matthew Collins',
N'01305 665775', N'07352 869339', N'60 York Road', N'Outer Hebrides',
N'Staffordshire', '20070914 16:09:21.713', N'Matthew.Collins@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (30, N'Elidh Peterson', N'01183
454781', N'07295 215265', N'59 Albert Road', N'Norwich', N'Cheshire', '20070914
16:09:21.727', N'Elidh.Peterson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (31, N'Jamie Price', N'01624
725814', N'07773 220491', N'40 Manor Road', N'Kingston upon Thames', N'Rutland',
'20070914 16:09:21.743', N'Jamie.Price@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (32, N'Katie Sanchez', N'01192
635557', N'07695 694205', N'50 New Street', N'Hemel Hempstead', N'Cornwall', '20070914
16:09:21.743', N'Katie.Sanchez@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (33, N'Erin Rodriguez', N'01701
154719', N'07699 485122', N'8 York Road', N'Swindon', N'Derbyshire', '20070914
16:09:21.760', N'Erin.Rodriguez@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (34, N'Abbie Brooks', N'01287
924924', N'07149 593067', N'73 Springfield Road', N'Ipswich', N'County Durham',
'20070914 16:09:21.773', N'Abbie.Brooks@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
```

```
[Address2], [Address3], [JoiningDate], [Email]) VALUES (35, N'Lauren Simmons', N'01442
337633', N'07986 644216', N'98 London Road', N'London E', N'Cheshire', '20070914
16:09:21.790', N'Lauren.Simmons@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (36, N'Emily Bennett', N'01907
200803', N'07636 884168', N'90 West Street', N'Salisbury', N'Herefordshire', '20070914
16:09:21.790', N'Emily.Bennett@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (37, N'Christopher Flores',
N'01198 529461', N'07207 549127', N'74 Queens Road', N'Crewe', N'Cheshire', '20070914
16:09:21.807', N'Christopher.Flores@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (38, N'Alexander Rivera',
N'01120 243659', N'07237 275885', N'25 Grange Road', N'Dudley', N'Huntingdonshire',
'20070914 16:09:21.820', N'Alexander.Rivera@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (39, N'Rachel Williams',
N'01512 965727', N'07889 215998', N'62 Victoria Road', N'Dundee', N'Hampshire',
'20070914 16:09:21.837', N'Rachel.Williams@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (40, N'Olvia Howard', N'01547
693549', N'07341 808038', N'17 Kings Road', N'Guildford', N'Warwickshire', '20070914
16:09:21.837', N'Olvia.Howard@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (41, N'Elizabeth Flores',
N'01231 471798', N'07493 384582', N'64 The Green', N'Bournemouth', N'Staffordshire',
'20070914 16:09:21.853', N'Elizabeth.Flores@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (42, N'Kyle Griffin', N'01558
179557', N'07522 904690', N'21 Manchester Road', N'Kirkcaldy', N'Hampshire', '20070914
16:09:21.870', N'Kyle.Griffin@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (43, N'Laura Evan', N'01495
153619', N'07390 958114', N'68 Station Road', N'St Albans', N'Berkshire', '20070914
16:09:21.883', N'Laura.Evan@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (44, N'Katie Martinez', N'01285
948244', N'07424 397578', N'98 Windsor Road', N'Maidstone', N'Kent', '20070914
16:09:21.883', N'Katie.Martinez@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (45, N'Sophie Griffin', N'01965
513379', N'07605 294144', N'10 The Green', N'Sheffield', N'Surrey', '20070914
16:09:21.900', N'Sophie.Griffin@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (46, N'Elidh Wood', N'01198
529461', N'07207 549127', N'88 North Road', N'Northampton', N'Essex', '20070914
16:09:21.917', N'Elidh.Wood@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (47, N'Abigail Hill', N'01555
443077', N'07439 738948', N'86 The Drive', N'Sunderland', N'Hampshire', '20070914
16:09:21.930', N'Abigail.Hill@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (48, N'Scott Bell', N'01469
486350', N'07759 520189', N'25 Park Avenue', N'Harrogate', N'Surrey', '20070914
16:09:21.930', N'Scott.Bell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (49, N'Chloe Martin', N'01329
343662', N'07783 161081', N'33 New Street', N'Telford', N'Dorset', '20070914
16:09:21.947', N'Chloe.Martin@example.com')
```

```
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (50, N'Lauren Bryant', N'01752
908316', N'07676 584363', N'70 School Lane', N'Swindon', N'Middlesex', '20070914
16:09:21.963', N'Lauren.Bryant@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (51, N'Lucy Rivera', N'01287
924924', N'07149 593067', N'31 The Grove', N'Rochester', N'Cumberland', '20070914
16:09:21.977', N'Lucy.Rivera@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (52, N'Chloe Davis', N'01346
526218', N'07291 685237', N'99 The Crescent', N'Coventry', N'Shropshire', '20070914
16:09:21.977', N'Chloe.Davis@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (53, N'Joseph Thompson',
N'01237 661433', N'07302 871821', N'75 Grove road', N'Truro', N'Northumberland',
'20070914 16:09:21.993', N'Joseph.Thompson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (54, N'William Wright', N'01417
502667', N'07177 363555', N'57 Station Road', N'Romford', N'Middlesex', '20070914
16:09:22.010', N'William.Wright@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (55, N'William Hughes', N'01407
629329', N'07309 884547', N'34 High Street', N'Torquay', N'Dorset', '20070914
16:09:22.023', N'William.Hughes@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (56, N'Samantha Miller',
N'01333 824728', N'07634 606994', N'60 Park Lane', N'Cardiff', N'Surrey', '20070914
16:09:22.040', N'Samantha.Miller@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (57, N'William Ramirez',
N'01626 530682', N'07768 951632', N'95 Grange Road', N'Cardiff', N'Berkshire',
'20070914 16:09:22.040', N'William.Ramirez@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (58, N'Lauren Green', N'01811
269842', N'07316 264705', N'28 Main Street', N'Huddersfield', N'Somerset', '20070914
16:09:22.057', N'Lauren.Green@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (59, N'Jordan Henderson',
N'01624 725814', N'07773 220491', N'67 The Drive', N'Tonbridge', N'Surrey', '20070914
16:09:22.070', N'Jordan.Henderson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (60, N'Michael Richardson',
N'01346 526218', N'07291 685237', N'76 Highfield Road', N'Kilmarnock', N'County
Durham', '20070914 16:09:22.087', N'Michael.Richardson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (61, N'Caitlin Washington',
N'01291 914147', N'07913 286110', N'55 School Lane', N'Peterborough', N'Licolnshire',
'20070914 16:09:22.103', N'Caitlin.Washington@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (62, N'Anna Long', N'01131
669331', N'07113 804523', N'41 The Avenue', N'Stevenage', N'Cornwall', '20070914
16:09:22.103', N'Anna.Long@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (63, N'Ethan Parker', N'01940
919059', N'07331 757163', N'76 York Road', N'Blackpool', N'Middlesex', '20070914
16:09:22.120', N'Ethan.Parker@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (64, N'Lauren Ramirez', N'01417
502667', N'07177 363555', N'74 Albert Road', N'Crewe', N'Kent', '20070914
```

```
16:09:22.133', N'Lauren.Ramirez@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (65, N'Joshua Price', N'01404
421474', N'07720 314778', N'24 The Green', N'Southend on Sea', N'Yorkshire', '20070914
16:09:22.150', N'Joshua.Price@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (66, N'Kyle Botton', N'01291
914147', N'07913 286110', N'25 Kingsway', N'Aberdeen', N'Lancashire', '20070914
16:09:22.150', N'Kyle.Hall@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (67, N'Bethany Hughes', N'01210
611735', N'07709 761071', N'6 New Street', N'Stevenage', N'Buckinghamshire', '20070914
16:09:22.167', N'Bethany.Hughes@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (68, N'Kyle Smith', N'01555
443077', N'07439 738948', N'90 Main Street', N'Cardiff', N'Buckinghamshire', '20070914
16:09:22.180', N'Kyle.Smith@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (69, N'Caitlin Griffin',
N'01469 486350', N'07759 520189', N'100 Mill Road', N'Oxford', N'Sussex', '20070914
16:09:22.197', N'Caitlin.Griffin@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (70, N'Jamie Perez', N'01572
459202', N'07858 711874', N'86 Queens Road', N'Nottingham', N'Staffordshire',
'20070914 16:09:22.213', N'Jamie.Perez@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (71, N'Jennifer Turner',
N'01192 635557', N'07695 694205', N'46 London Road', N'Brighton', N'Derbyshire',
'20070914 16:09:22.213', N'Jennifer.Turner@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (72, N'Alexander Barnes',
N'01281 554917', N'07800 642581', N'78 North Street', N'London W', N'Lancashire',
'20070914 16:09:22.227', N'Alexander.Barnes@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (73, N'Megan Campbell', N'01205
396727', N'07198 312473', N'20 Park Avenue', N'Blackburn', N'Suffolk', '20070914
16:09:22.243', N'Megan.Campbell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (74, N'Lewis Collins', N'01237
661433', N'07302 871821', N'10 Victoria Road', N'Southall', N'Norfolk', '20070914
16:09:22.260', N'Lewis.Collins@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (75, N'Bethany Peterson',
N'01784 976829', N'07885 861269', N'43 The Crescent', N'Worcester', N'Surrey',
'20070914 16:09:22.260', N'Bethany.Peterson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (76, N'Erin Martin', N'01428
428418', N'07377 251946', N'39 The Avenue', N'Milton Keynes', N'Suffolk', '20070914
16:09:22.273', N'Erin.Martin@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (77, N'Joseph Coleman', N'01788
915553', N'07736 274237', N'95 Park Avenue', N'Wolverhampton', N'Leicestershire',
'20070914 16:09:22.290', N'Joseph.Coleman@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (78, N'Jordan Scott', N'01500
424473', N'07824 778422', N'12 Stanley Road', N'Galashiels', N'Cambridgeshire',
'20070914 16:09:22.307', N'Jordan.Scott@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (79, N'Kieran Kelly', N'01633
```

```
392324', N'07960 870808', N'91 London Road', N'Twickenham', N'Northumberland',
'20070914 16:09:22.320', N'Kieran.Kelly@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (80, N'Tony Brown', N'01231
471798', N'07493 384582', N'41 Queensway', N'Falkirk', N'Berkshire', '20070914
16:09:22.337', N'Anthony.Brown@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (81, N'Joshua Sanders', N'01902
365096', N'07414 740869', N'12 Albert Road', N'Dundee', N'Hampshire', '20070914
16:09:22.353', N'Joshua.Sanders@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (82, N'Bethany Rivera', N'01784
976829', N'07885 861269', N'54 Church Street', N'London E', N'Northumberland',
'20070914 16:09:22.353', N'Bethany.Rivera@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (83, N'Laura Hayes', N'01982
956405', N'07109 544766', N'56 Station Road', N'Brighton', N'Devon', '20070914
16:09:22.370', N'Laura.Hayes@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (84, N'Anna Griffin', N'01345
471798', N'07493 384582', N'40 Windsor Road', N'Guildford', N'Cornwall', '20070914
16:09:22.383', N'Anna.Griffin@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (85, N'Samantha Jackson',
N'01231 471798', N'07493 384582', N'14 New Street', N'Llandudno', N'County Durham',
'20070914 16:09:22.400', N'Samantha.Jackson@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (86, N'Kyle Sanchez', N'01978
657197', N'07610 445571', N'3 Windsor Road', N'Bromley', N'Oxfordshire', '20070914
16:09:22.400', N'Kyle.Sanchez@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (87, N'Shannon King', N'01624
725814', N'07773 220491', N'56 King Street', N'Paisley', N'Gloucestershire', '20070914
16:09:22.417', N'Shannon.King@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (88, N'Joseph Gray', N'01211
864653', N'07375 760438', N'56 South Street', N'London E', N'Gloucestershire',
'20070914 16:09:22.430', N'Joseph.Gray@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (89, N'Courtney Perry', N'01210
611735', N'07709 761071', N'13 St. John''s Road', N'York', N'Berkshire', '20070914
16:09:22.447', N'Courtney.Perry@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (90, N'Matthew Campbell',
N'01743 360803', N'07394 612512', N'68 Queen Street', N'Romford', N'Essex', '20070914
16:09:22.463', N'Matthew.Campbell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (91, N'Alexis Cook', N'01624
725814', N'07773 220491', N'36 Queen Street', N'Falkirk', N'Norfolk', '20070914
16:09:22.463', N'Alexis.Cook@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (92, N'Jamie Mitchell', N'01776
202627', N'07185 505540', N'90 High Street', N'Kingston upon Thames', N'Warwickshire',
'20070914 16:09:22.477', N'Jamie.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (93, N'Chloe Morris', N'01725
222331', N'07526 574605', N'97 Main Street', N'Bradford', N'Cambridgeshire', '20070914
16:09:22.493', N'Chloe.Morris@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
```

```
[Address2], [Address3], [JoiningDate], [Email]) VALUES (94, N'Lauren Morris', N'01914
784723', N'07798 945002', N'44 Church Street', N'Durham', N'Warwickshire', '20070914
16:09:22.510', N'Lauren.Morris@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (95, N'Olvia Barnes', N'01811
269842', N'07316 264705', N'78 Stanley Road', N'Lincoln', N'Herefordshire', '20070914
16:09:22.510', N'Olvia.Barnes@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (96, N'Olvia Hall', N'01571
657822', N'07694 711784', N'14 Windsor Road', N'Coventry', N'Middlesex', '20070914
16:09:22.523', N'Olvia.Hall@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (97, N'Scott Turner', N'01555
443077', N'07439 738948', N'87 Station Road', N'Salisbury', N'Surrey', '20070914
16:09:22.540', N'Scott.Turner@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (98, N'Rebecca Hayes', N'01802
342584', N'07296 315860', N'94 George Street', N'Bournemouth', N'Somerset', '20070914
16:09:22.557', N'Rebecca.Hayes@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (99, N'Kyle Perry', N'01653
610021', N'07363 920914', N'82 The Grove', N'Derby', N'Gloucestershire', '20070914
16:09:22.570', N'Kyle.Perry@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (100, N'Kieran Patterson',
N'01389 465402', N'07266 836025', N'3 St. John''s Road', N'Bolton', N'County Durham',
'20070914 16:09:22.570', N'Kieran.Patterson@example.com')
SET IDENTITY_INSERT [dbo].[Contacts] OFF

-- Populate WidgetPurchases with lots of data
DECLARE @purchaseCount int
SET @purchaseCount = 0
SET NOCOUNT ON
DECLARE @temp float
SET @temp = RAND(1)
WHILE (@purchaseCount < 108435) BEGIN
 INSERT INTO [dbo].[WidgetPurchases] ([WidgetPriceID], [Quantity], [InvoiceNumber],
[Date]) VALUES (
  CONVERT (int, RAND() * 3) + 1, -- WidgetPriceID
  CONVERT (int, RAND() * 100) + 1, -- Quantity
  'WIDG' + convert (nvarchar(10), convert (int, RAND() * 10000) + 1), -- InvoiceNumber
  CONVERT (datetime, -- Date
   CONVERT (nvarchar (50),
    '2007-' +
    CONVERT (nvarchar (2), CONVERT (int, RAND() * 11) + 1) + '-' +
    CONVERT (nvarchar (2), CONVERT (int, RAND () * 27) + 1)
   )
  )
 )
 SET @purchaseCount = @purchaseCount + 1
END
SET NOCOUNT OFF
COMMIT TRANSACTION

-- ************************************* Insert data
*************************************

USE WidgetTest
GO
```

```
BEGIN TRANSACTION
SET IDENTITY_INSERT [dbo].[Widgets] ON
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (1, 'Red
widget', 'RW')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (2, 'Extended
widget2', 'EW')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (3, 'Dark
widget', 'DW')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (4,
'Grow-your-own widget', 'GW')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (5, 'Widget
access kit', 'AW')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (6, 'Test kit',
'TK')
INSERT INTO [dbo].[Widgets] ([RecordID], [Description], [SKU]) VALUES (7, 'Exploded
widget diagram', 'ED')
SET IDENTITY_INSERT [dbo].[Widgets] OFF
INSERT INTO [dbo].[WidgetReferences] ([WidgetID], [Reference]) VALUES (1, 'Widget 1')
INSERT INTO [dbo].[WidgetReferences] ([WidgetID], [Reference]) VALUES (2, 'Widget 2')
SET IDENTITY_INSERT [dbo].[WidgetPrices] ON
INSERT INTO [dbo].[WidgetPrices] ([RecordID], [WidgetID], [Price], [DateValidFrom],
[DateValidTo], [Active]) VALUES (1, 1, 100, '2000-01-01 00:00:00.000', '2002-01-01
00:00:00.000', 'Y')
INSERT INTO [dbo].[WidgetPrices] ([RecordID], [WidgetID], [Price], [DateValidFrom],
[DateValidTo], [Active]) VALUES (2, 2, 50, '2000-01-01 00:00:00.000', '2002-01-01
00:00:00.000', 'Y')
INSERT INTO [dbo].[WidgetPrices] ([RecordID], [WidgetID], [Price], [DateValidFrom],
[DateValidTo], [Active]) VALUES (3, 3, 25, '2000-01-01 00:00:00.000', '2002-01-01
00:00:00.000', 'Y')
SET IDENTITY_INSERT [dbo].[WidgetPrices] OFF
INSERT INTO [dbo].[WidgetDescriptions] ([WidgetID], [ShortDescription], [Description],
[Picture]) VALUES (1, N'A widget', N'<description
xmlns="http://www.red-gate.com/widgets/WidgetDescriptionSchema">
  <descriptiveText>A widget</descriptiveText>
  <manufacturer>Acme Incorporated</manufacturer>
  <countryOfOrigin>United Kingdom</countryOfOrigin>
  <size length="12" width="4" height="7" />
</description>',
0x474946383961db00d800e643000000000698495465863cccccc010f1b003366000b143333336666669999
99ffffff032d51808080021e3600333334424aafafaf01162884a5bae0e0e07b9aae586e7c111618669999
1a21252020201f1f1f4f4f4fb0b0b0dfdfdf4f636f7f7f7f6079883d4d57505050021a2f00003303294ac6
d8e399cccc84a9c1b7cedc739bb61e517de9f0f494b5cad4e1ea336699e2ebf15a85a5517d9f7b878d3651
65144876b0c9d8333366afb3b613395a0916204d5d678fa3af2d4252bed3df7f8386d0dae3cddce68db1c7
ffffff000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000021f90401000043002c00000000db00d8000007ff804382838485868788898a8b
8c8d8e8f43090708039096979899 9a9b9c9d9e890900a20019081c9fa8a9aaabacad8708a21a1aa3a22209
0aaeb9babbbcabb000070a0a031f07b4c00913bdcbcccdce0a22a21bc2d40a1d091bc7a595ceddedf9a0a
c60008d5e6c209bfb4a6b8e0eeeff0e2a2e5e7f5c308b3b4b6caf0fdfebb0346ddb2477098a463071870fb
c7b0e1a680a206169c782d1bad52a71c6adca888c3280813439a83a06e14825b1c536e0c256a80c897e606
e043984ca54d782c01b884c9b3dab571a3262dbc49b4d7af603d939ec3a6cd54d1a7ae8e2a9d5a8f643e51
07a06afd248f1cd5aff5208ada4a36dc387a60d30a030aa0ac5b4813ffceaa552b76c482b66ff32a122b71
2ed59c0e0adcd54bd8105fbf60d515583c3891020efc0a13cdb913f1548b061633c67b48ec819a925352b6
4c751c09cd9b1189a5f53972e87f0cb05626cdb32eead4863cb62449d3f56b7052692785302af0ed020e38
0f1a4d8d376bd0bfbb0517ce33e782e3c8c915627ece7950e8d17775454bfda562ecc913149a5ed03b56f0
e1578d2fcff35766ec1100a827c45ea47b64bec5d7c9001988f2017d308d13017605e4b79f200afc32cd54
```

ffb526e08002212852070a32e82021f3a555217c173a729886138965dc711f0e12a25f2306582222dca168
8f8a0c92a01321038cd3976531ce08ca28b3d9b8d4280c1660c0ff8e839c485f90420e7254074612945392
4b0ee5248a5096d85f95d55cc9609684d4586585198517215660dac3d27dd81970c05066b6a9807bf1bd68
27356f26094c3b82b0a441917bf23956747a16da27837ff2c766a1f6e070682f1c5808508690a2230a9cc7
352ae5a3999a03440e93eab2e677327eb2a5a29bfa198ca37f865acd003fd4709772aef4184b6faae80600
48a142c4e96d5eb92897acd40494d9adbbe4444f979bd469670748a2a79db1f3209b2c00f731eb0a341f75
5712809a486be7380c26870088c76aabac66deb24220a8f640dbc897ac02b0226aeaf258a07eda0af32ebc
a5a6129b8130d98b08be851a4380b5eb3689a9bbdcde162fff2a71c542a87fe35a6a48a2a18a52c271e9f1
3831b203a376b127be4ef815b420870cc075a8b52888581b439ab2ca0573a2ce8f60c598b157010b234a03
35eb57269145efcc33ae9ae80a4cce228e7b72c0bf186773244c07ecf4d39e38aba17b40070c948e0f72a0
0ed57b7e0df626e08a02ac8d107cc036b2161dc3dadd6dbafd7626f3024365d1eee6fd1ce17eff7d494e07
12eef80090433e78d315274930d48e98ebf8e65e576ef9669837121b527c736e7adf9e7f2e58cf8e8cde02
0ff3cc7dfaec76268eddca8f8c2e841012547056e9b4073f97edb7b32e3a30bb274f81071800a00125c247
4f1bf1c587ce88eec9674f410816389fc0e4d2873f15fff5d567827df6e85fc03d32e08bef7e8aa9abae38
24e7a36f3f080f4853f6fbfc0f13bffc97cb044bec47c0ece14f1ab2eb9ffbc89724dc994814058ce0ee7a
7796042a30780c6ca0f1180111097a907715681e398077c160fd0f80f37b44073ff8c1e535ef7925345d06
2de7c046ac90852cbc40f3f617c34ccd90861b5cc40d71e8c10a10ad879d1b160acb87892112b180f9e321
12db76c2255a2c887b81e0132318c52926d18aaaab2107b5b845fb75d18b28ab221853f8400094117d1268
1e09d188a01f02508c4224e31be3a8133acaca8e77c462229cf8440a6060507e34a112d7c8203c66d18d6f
bc40ac12a9333532f28ad6cb2324b72849a450b2ff50805ca22307a9471c0660929fac9d252f8949c095f2
8346244f2ac1144a2b8e5235af94602c6749c545b25283997c24113d0080c6f19296abfc652b9b984b029e
f19855aae51a6fd99966a2ef99d0b49134a729485c02e002129480312c984dfa6c939bc15484280210413e
ceb19c6939273a35b1ce02ba139eda4ca6321bd94d44d4d37e3a9c133e51244f4652f310ffcc5e27076aa3
0f706b5ffb44e1410d91d0dd2d94a174fb0509201a5120a633111505c111316a24e79060641df5283d01c0
cedded92a47b620a002280d2942e1313ff7c294c217510008cc0a66c74443d4300b09dcaea27a2f8a94d27
5a887562d3a87f9c8901387a49a612421433ff282a5411679f9a56b59f8669c956677710655a7508be6a1f
a43a50b70f24e09dbc7401ef00803456e6e7a3dbe95ab00c370a63c254aebb130000ec3a0f4b080aaed4c9
89dee406d314644f0274656472aef50886d949b18b652c431d8b3ec1ae71b2afaa2cbd3285d9cc8a82a127
202064a9fa39363d22663135ed62a548c9d44271b04b24d26bdb152a87ca566f7ee5250b5a20410ae80b85
4b62522386264b4859edb75ae55c2836405b9ec080b81ecc1f0093fba03146445bcffdad2739e7d008fc0e
2c806de1713f77d7ee6a32ba990aef6f67078bc0484002cc73de07d42a92203c51bb96bb6bc41aa1b936f1
15ba62355d7d0b400001244f7dddfb0cff4f384b44e3b2363b94654481c1c41604279873c6d04c83ef973f
7290d31c3678238049e65a026b4c5b1df63062ab1462113b58b5be731ef4ea61db325a183b2d668465cfe5
e163cc589b0718d6888b2b8008f785053d56316e51a3db7b8db69245a6c59109fabf257ff00225a6ee70df
a8bdf52e26b943710c6f7d98652dcbf0845ec6618e01400132676fc5dc6d046c55d9e60f3b2e140bc24e9c
8928010b3cc0ceca9b59832262c3710497a77df633e1009da44113d1887546b4100e4000013bc247da2aad
8c151cd94adff88d86d6b4102ed0bd0c2fc218cdcd579fb7aca1055bced24414293811ad021d78aa11c6d8
010c6425dfdfd21a41b6bef5a9dfff788003f09ac1006000248ce16013c437d27d04f1943f876b229eb2a5
658cc1620810ed6903e0c62770c19e8a2ddb63d3a7c6f2eb36119bfd461988b8dc8fa076f64e606d1a63db
ddd41980067cc920799b92a55b7c016ac82ded7c9fdb7ea7dcc0892d13e3516fee8706c721bd718802852f
1cdf9f7e38fa4e190b04f0772e154730c0851390875931e32cfc360b51b082e330dcdccbde1dc90f8798d5
b479e5b4a1341861cec28d4b500535b739c8812df2eced5c6f3b068bcfb30c74d2087de83927b3244170f4
4a2f9d11fa1eb978ab5b90a917b9ea964936d655bdbb0758208217b8f5d75fddf4e43d5db60898b895b09d
6dc7a97ded6c376e0508286effb9373ce459bffb6f35b0df9788dae28ecbc6850148741cbadd7e1e373cce
211e690917c4b791d67ba6e0cdcacab350f0d9cbbce61d9e78be8fb01eec962dd9f744fad2671dd1971702
cd293f774584dde9ae7f4f35626fdad9a38ee080f72005022001f55660f7bc3f3cd35b1f7c934020e50836
3e32919ffc0282a0d5c0c874040fd08397f73e11bfb77bf5d70f5f6d11e734293578fe025d8225edba8007
a82b0a6fce7aceb3dff5da572557d751f2664410a523cd877ffa177d9b2776ff876d016824034880b73757
4a452c87a680e6277d6057773af78091168136f277145840c4942e74a6814bc47f88e77f20586422882224
588204840117d8291ea0ff82fb777e88907e1ff8821ee63245630c5e0554b8065934831fce4640f9b7810d
087c40586451272bb507540c767ba76439c9a1830cd87f0e188545a601df93291cb26d56386eb7674440c4
85f1c68387e08342a078608860d4d54b676873b727002ec74f6ca83a2c387d2e38877d36851a5250f19686
6678457dc86d6e6808702887821886638820137887684840599824 25b049e8d3842bd88885f08891587d75
985833658982767bc6653923b084f6e3893bc88174477da3587d84a816a17083a868630564010b482c21b0
88abd7825f588bd5278627971433b88bdd4654fc747f9df88b7e088a84208ac6f880a54815cbc88c15f863
36e78aaf288d8c28ff8bbee78171788d40788b22610c49b88b7818419e751bf9018dd1e8845e0885e8f882
c8982089e88e0b57816d67663a3278e3278ec30888c5988f2f286613010bfea86c1e2458a360015c2741b0
d885c4888f0a1985ec740227e00326109229e09142a0870f696a1f2401015001e0669106e975e4887ee608
891bc97e2d4940267992a9c876d973916d18933d389335098637693f39a993b7617a1af79205478d83608d

```
43f98245d9597b8894bcc893bbe393d308946f2894512995115995567985589995 4c29684e290850f995ff
478f46299656a9944577964ac7958ee8956cf97f1f749463499665a995e3f884ea97977a19967d79955809
984932597619ff8a784998d5b79770899472f9418a692dbf86901a0999ae074bfda89395e941978949a175
8f83c999c1f741997898a1e99200340aa9528eb4889a7ca79a9f7992ad5990f2536596b096b41969b67998
97f89774c99bbdf998bfd967c1299cb9298c9a81669ae09bc99965cb79980b207e3c399a79169dc8399d45
569d8759966689258cb609d2e99d08069e56991c0f5091d969903aa2349c709ee8f95beaa99393350aedc9
7680e9699d409ff599596fe741ab899477d56cc7f0006ef9445ae99fffd99d016a5ae05840057a92078a0b
13c000ff220a1610020b6a99bf380285f509001aa108719fee78a18520131b0a001dfaa15c085aa950a226
1a14288aff8a2aaa1a08d0a22f3a979a015a80e209345aa3c843a0b779860e2a443b4a0b18e001d8198e8b
01a4aa30a4353aa19878a44095a48ea036c7d0a44f3a9e05200a1910a4240aa144ca1a376a855a6a095cca
a44eda9348639ca840a5266aa59ce78fe4e66a9ad0a6a3800115d07c34909f6936a7667aa65891a629955c
03860a1c100dac7103723aa3e6284e869a59762a76bba8a8df92008e4a0bee35a5e6b83bdb53a9276aa4a8
a8a9e2c1a9e5e90a70883e2144aa452a41156a53dbb90c0a109b840a900a5562677aa9c07787b52a19e320
005faa5a1e007e11aaab1f7886c12a198af500538989bc5a9fca7a8e56d8aca131012561017f7a7a4d46ad
88caff480251220a90002d1a02c54a4020807d7959adb3fa4be31a2503d0a9c010adf633aabfe9ae586a45
f11a2582a0adb460011e9080b0c4ae1ba9af1dd5affeea223d250a0f90ae0035ad5f89b0fb14a90bdb2425
7100eee941bd83ac3549b1f0da1217bb08191ab0036b4a129b8f207b49163bb287d0b000a0a0382401dfaa
902b0b469b28b22eab42194b902cb4ae2a1bae5ad8b23bab080aa0a113290004ab7c12598b378b42825ab4
e5c216d07a6906cb7e3e1b41efaa3a512bb50f51127e5a61290b82f6aa7e6034591930a85e8b09e5daa2c4
3ab3af0a8465bbac4b24a56bcba8543bb74e37b6c1a7b75b8b829979b79f00b013d9ad2c845f1eeb7a7ebb
ffaf3f1a6482ab0a6d4b0be8ea6d578b608b0b40221ab88fbb0af3ca1a7aab3c358b6d97ab3a39bab9ba40
b81c7ab27256b98b35ba01e6b8a66b2ae64a0b0f5b6144d566aeeb21b01bbbbc2013acb1b11c1bb7968ba8
77250264cabbcb50b213a9ba39c4b7c790bbf238a2c80b0e302bb3337bacb205bd9ab1a6d3eb0dbe1b1459
ab9a7cabbd8ba6a7dd0b0e47bba116a0b41516bad6f4ab4929bde7fb0fd50ba3ea0a14316786a83abf0ef1
bda210b6b6cb895abb6dfbcbbf1a11b9a3f0b6e2891a056cc01c5129b4fbb944f49cf2ebc02a81ba00e0a7
4b6b6767b6aa164c14082c0a938b684ae2c11ffc148deab9642601d87ac24f81c1dcbac11c2b429fea39c2
5a11c2310bb10a45c3365c189d1b14c0fb8535dcc36ea1bca94bb02a995f264cc48501b396aab64cecc3cf
8511514cae910339c7ebc28100003b)
INSERT INTO [dbo].[WidgetDescriptions] ([WidgetID], [ShortDescription], [Description],
[Picture]) VALUES (2, N'A medium-sized widget', N'<description
xmlns="http://www.red-gate.com/widgets/WidgetDescriptionSchema">
  <descriptiveText>This is a medium-sized widget, ideally suited for the home and
garden environment</descriptiveText>
  <manufacturer>Magic Widget Company</manufacturer>
  <countryOfOrigin>USA</countryOfOrigin>
  <size length="15" width="12" height="2" />
</description>',
0x474946383961d100ce00e640000000000ffffff465863cccccc6984953333339999996666660 32d51010f
1bdfdfdf84a5ba4f4f4f1f1f1f7f7f7fafafaf003366111618586e7c0116287b9aae003333202020e0e0e0
000b1434424a8080806699991a2125b0b0b03d4d575050504f636f607988021e36e2ebf1b7cedc84a9c194
b5ca99cccc03294ae9f0f40000336b93b01e517dd4e1ea336699739bb65a85a5021a2fcddce6c6d8e3bed3
df97aebc336666a2abb1617d8f11191f164a7849769a666699dbe6ed1a242c8db1c7ffffff000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000021f90401000040002c00000000d100ce000007ff8040828384858687888 98a8b
8c8d8e8f90918701061f059797071a1d01929e9fa0a1a2a3a4a3010500a9aaab0007a5afb0b1b2b3a20316
a90c0ebaba0ca8a91ab4c1c2c3c4a103aa0601cacbcba9058703c5d2d3d4b106aa03ccda07bf85a7a90717
d5e3e4e589d7a9d9dacbdcdd83dfabe1e6f3f4d2ed050aebcbbe1309c082b6521968074e5cbd83084dddd3
b72fd50408fe00225be660d5bf8418333282778061800 1be44408008ec98c07515535988a6b1a54b20015b
793409a0c24892e800a8d347b0e3cb9f07692663f84095cd9b097ce9f4a86c4003672c814aad96f381c79c
376faa2ab0936900821da68a2d46b0eb365559476270e6755d518163ffe3cafacad6633b0c69216c6dab0f
2438b9804df962c0945f5e551df932f4e52ab0e3483113f77d0ae0615ac48a0bff7dcc7991d099460f83cb
ecd59781cea80d55bdaa0a85680043496bd699ba7659bb685fc796cd5465a7da8fe90228c0f46ede0aaaac
f26e6bf203c20b9c803f8247d8a32f15c7b12d579c32aa390d88c34a574453f23aa70eb3a7db9ef9a98579
1794aab2206f7ca10e1319d23c9a15f970b3ec3165d269e4e087cb0006dcb25532f601911380cbe4e45a5a
13d4156066a83c538d70b031330041a97c209e6da93400a1322901901704150e77a16c2679474c4c5c11c5
002b07c808183cc4e106005e79b558dd8ba43dd5583139997795ff7c161860905cf18de6d18d0024b0e25a
```

3211295b4ac5c0b354660a1c4099330c8a459303b3599616964a6ac9578cc31808c090bc7d88e3882fe5b4
db3a86e585259a6ef206172d1ceeb99d0154aa549f46e0ad675d2afc65955fa0b2dd78642c7efd17a80206
8c395c990709878f8090ae3829a5a455a4a13598a1da14880088580f8f6da564ea49aeca864e2c017c905c
aeda3c9028383a4a13195fe8448a8076c0ae938f7ea9c042538fcdae6300934e52f399626326a002965f56
bbcc31ca9d176d2904192aee32617afa01a8c1ac9659a6ab98b86e84738206002999da7baf80b0e6184c3b
fec2e8c001b9947b2f378042bbaf2849fe9bd9b5abd0f724ff29b44a4ca402a8a8a36e006f8922a7c21ab7
c5a9a705c0fb499459961ce031a3720cc0b3daa80a712af99ce8b27eb0cafac9993b07784d62309fc8cdaa
3fa7620209b8901cb447c2e258ac227a3ebd1d37435d43279f005c9434003ffcb080048c396db5b5d85e9c
48a3e19ead985fea30ec552a787a6252d8782f000207003490a3db1eb5bbcabbbf19222acd80b75554cca8
28778ca16fa92dc9dd78574e810011f46d00e2892ff340c0d910425de7dc6559b432d750cbccd1a3505ef9
eb1b789079caa4a35def01392bd866edebdc9875cbdcec1e8091ada7f2faf1788790012e1f037ef22a4a35
cffb471d2b8375431fc3598bf1c877ff4308654fff2aff2b3a4fbf783e32ab733a4337be573cd8de7b3f36
63e59f2d2c03f5d75e11d1c33dab757183da1efce2173f0aecad6f7f135f0014d00ab391ce7728021e001a
a60f5f38875fdc23a0062920bb4f71ae73146b80033ee836b8290382ca68dc6c2e25400dba106fb1838af8
0447bbc49d8f7a5f5a5f5f54e1b5f7bdf08762cb9c033bf739629d6d7fcab82125f2053564602a83408cdf
02f8963f10a24284245c170a9168bd0932e541b2705d14bbb78108144c81ceba87f45065c200a050664314
ce4a6621c631c2ce4568541c9512582d0554a78d3ad4079588438b3ada316c1260621e1513c2cdad4b89ff
0b1c630a1746281ef2072068d922e7d50e06acf1ff425ca40b05cde58e6018728ccbfbe426a1b6c72a6ea9
430b54216b02684a4b467101a850e52a99f2bc1aba497d9af2488aa6168b0ba46203879c62db76f9b24ec6
913791f4510124178c5410c08e653c2333898428235ee87ab3f900258961cd316e008fdb4495021cf0145f
f2a67afa6ac538ca09c4446e2d9d6c6cc7019ec9932ca2ce89f304c0357f9849e1e1135521c49d9686590e
7aba30950795d83adba94b6952531a0e95622e23ea323bb5c295dab0e038ab91d1ee2913a41c75550847f8
36c4d4a3a4c7cb264a539a2b1aae514f078169e5cea93a9ada4f9f102a0b42748a4845fad48a9aa3d9202f
4a0ea216f4a8e2f368477c41c884e814a250ff45e303ac023d9599a3a4b884655617e9d1f0bc54a0953be9
58b7d9cdf92c2aa003fd814cd78acfe72d68a4e444ab5cd149d783963544751b4639edd9d7a3462d1ec484
85351369d0c21e9462f3c95635052a56c766557000ad64652d4bd7b2e25580cb74db0376e1809972561f27
a040294b3132d2a5a85ea33cad62527082b02d0f009ff5c4833a472f56c850b6b33501de54dbb552ec3671
34f12df980db961608b7721e4805531fb110de2af7ba3d65ae365a80bc056c0614d545ee75c7cb4fced2c0
7b0290ee277a850bd22577bcac88ad764920c5ef428223ee85ef782b7a3606f0514bb525607a0130dd494c
1284fa1def2a51910304fa93340126a0ff77e5f9886d252e27095e6e1e65068115707076fcd5460a9eebc2
0127961016061c8633cc2c341e032f088800de08705b06949719238ee2ec1a71dc0bb3d8b7a6ede88f46b2
acd7cdcf9b5ee1ee1813491baab51870b0fa71904b760d3515197906cc5c031cc910195c72c7e728d19481
1565168f596255ceca95bd17c339198abe9764f2d4c29bb83267f8ccffba460cd2b2660282cf9b70bee40f
c05c083a034e3e3f0e2defb81129220fd08563e39bdf048d37398beec09d43b49417c9e815f5f9871c0480
04281d36beb90f08f8259da6cdbc485420604510f834103320635213002ea946eeaaefdc6a15c13ad68f8e
a26a474d6a54bc6740f94df42a42ffdc5154fcdad182a635a97f80837498a405d655f6a9a6f7e26743fb92
aa1500a54bc0026b1b4f06807b8bb637cbbb6e7b1bd8944eef022ef98291981b7e2738db8a13cd6c2a57e6
dddfbea477c56dc715684527622441d0f6fde37e6b2ccd008777bc0130ef28c2405208b7e5096ec0e575d9
39c30e975845441271890b7ae051dcc165321eecb00dd893e2fa788243feaf4e97dce48296f70f5d209a01
9c72c08879708064ae5f9a2fac26370ff8c9014070029680e73dffb972dd49a45d833c8fce4ebad2734e71
a7b300d6f7461ed095fb5ff6587de658f7b5d6717ef20878207e2fd0c1afc37ebcb15f77cb429fcdba8d2e
2ea70069ed6c17742229d0bdff7a3f9beeafb33b7c6d0ca3756fbb76ee067ce0051d810c20efe2de467ce5
149f608532c7f1b88aea9025bf754a0ffe7598cf3ccbc5ee78aa6be3bdcae67bb5204efac953def27883fa
bb358f37ce27baec4b04bdec9b45fbdacbdaf40020fcd34bce7b9783be5eb16178c3d168f3da97bebb0420
40c509587972dfbcf93ff0bde3fdfbfce19319e9d6bfc9f1d37adb547860fbdd13800fbefefdd5d7fdf9f8
f7adf97355fdf4db3e6ce7847ec8c101f05777569274e0277ef9177b689475fea77e2d97371cf0775a5100
f1230007587f3e674be1b780f9275f99a6760f787dce0776c4267619c87cf697781ef87c8d753691378224
f80300407241526b28a8ff7509d8828ef78256138332687b21208287814c3988802bb8793ca86dd9454455
12847c16813fe0012978184d97785508703bb884da067cfaf66f50a86652580036b82213608147a88152c7
85eb86452a0686610881dd837eb026026868803a9884bdc786cfe77a1ad37f71687b00f06abf6687f19785
bba787cec787f8e78531478781488283f86c8698862ab88111a8808cf8636e782f373221912887c8c30165
9817958887488889acb7891ee88794e280a1288ac753006ab22209f0769618715bc88a2de8882f028bb148
82e9757827888579a88af7c78b4bd889440284c1687be74488d95180bd8788aa878c2ca88c6ce88aa4e18c
cfb87e3f5000d608ff00b8978b5aa8881da88d8ce88b5e710dd8f18c79018eb7d668101003c9875ed67878
e8a889ea987fcc882c70088f627881902802a27681f93877fbd88fcac88d1e129002298bf1f72322609007
8990c7b8860ca98c3540033330022019923d300327706b11e9695268391970091e407802969060b7901ba9
8c46e83d26799251386dc6988a1a39939ba84137899312a9931898911cc88f3e997f40498438098e975494
3c799449c98a4b299403a993d568949938953f49404169954e794850a98652c9957c58955639834f0993b7
828d4a68966c8896691996763496 97d89370d98272099629b9964848607899970bb8974249976364978948

```
1b812998f847ff984dd99762c9963781252cb1988c097a8e799286194588f96b58422030910ad4889497c9
629919919b09449d7925b4040430459aa59960a72990a9f943ab991609d09aaea9576f199bcfc701b3098f
b5f942b779132dc24282f09abe897f7768934c89932b809559f96bc78908cab99ca0d79cddf39556199dd2
197ef9688f147608d7899deba69dc8c39d27e91f191002d2799bfe813486509ee6a96ce8793cea2990feb1
0a1e5093398788f1995bbb19577b589f4c189ca1b89f07a020a9e076fe79985918a08b409f06ca62f7f93a
f9198cfb59990bba0a0e5a9759e81b8c40a1159a60175a39199aa0ce20391fc2a000e0762e499c55880d8d
40a2250aff5f273a63cfa9a22e421e1daa0a1c0002318a9107d76423ca9b057aa31996a36193a27128a18d
d001b012a4438a8a7a910e8f60a34a3a75081a84500a0952ca0a547a845812588aa0a55bca0a4cfa034e1a
842d52559f10a6d02301f057949f1909689aa65bd1a50ff8a6020aa6be32a7f3c6032870a7788aa48ba8a7
e37585dbb9a36eca16bc52096a6a032dd2438e005343a8a88bcaa7d6e7a7c14009811a0f9fa0537a93399a
ca0a8c9a9e8edaa7f6250ca0daaa9040544dda7ea79aaaf8b9aae9579dd570012736a1888a659853ab9c0a
78ba2a17ce50a5def3677a6aab188aaba4a702b02a16d0e39e2e146a5bcaac28eaac80c7268f71016cf3a2
2040ff8ddd23017c53a2d8aaa350c8ada80159a990010fca66b48a9de7daa4da9a74ea5a1b7f5500c52845
1260aacb39af6c5aaf258725969a1adecaa0112000e29a9ef1ca9800dba6f6aa9be3c1aee4f8ae2615ac97
f9b002fb6e86da2085f0571cb0aff1a3ac7929b2b7ea7f1debb186a3012e2a00c8da3dd66a9604daa8e997
592a9b081d201f1930b3f143365cc9b3275b7b367bb38a70012012b2a0d6b0ea08b4cd2ab4124bb4894009
2eda922f14691bc9b4d94a7a430bb58ed001a13a1c588b9f4acb88614baf9247a35cbb32201201745ab520
e0af9b58b6010b78689bb6a020b5fcf9b2dd43b27c28b7107b2b466ab7b5f0b50540add50a9bdae6b71b0b
ff6c582ab8afe0ad1e1aae3fe4b34ba8b837b7a18e3b0b149b017a8b65d1d582961b71989bb9850422fafa
43569b7fa1fb6ea34bbac100b9aa90b00b8b3c1b30b63fb6bacfd6baae3b0c9b6bb126f5b68ee74210fba5
bb5b0cf96ab2367976d725bcb84abcc52b0d01c0b2abe0b2a086b119c6bcb90ba9cf5b0e9b2bb795c3b7e3
85bdb0e6bcdb3b0e46bb0a480b6a9f1bbea7e9a9e53b2b0992b7f554aebe25be14a2bdef8b10390b3dde0b
430deb42e71424f89bbf0971beb1dbb66eebaf2f3489c639c0048c1178ab0a54fb43340699782382c5fac0
2f3100846bb8d2991519acc13f61c00d2ab958d9c0e329c262b1b99d1b452311c22a6c26a68bbc40045511
d11ac36201bb0daab0c974af38fc18bd7b4be58a9c3fec18c75bb5435cc4d211bd084bbdde430149acc4f6
d1bdaf4301cab3b5520c1c24ec0c0540bfaa60a659bcc4f13b755e15c61e8b201a90c61dd0abf91b08003b
)
INSERT INTO [dbo].[WidgetDescriptions] ([WidgetID], [ShortDescription], [Description],
[Picture]) VALUES (3, N'A small widget', N'<description
xmlns="http://www.red-gate.com/widgets/WidgetDescriptionSchema">
  <descriptiveText>This widget has been discontinued. It was last sold in
2006.</descriptiveText>
  <manufacturer>Magic Widget Company</manufacturer>
  <countryOfOrigin>United Kingdom</countryOfOrigin>
  <size length="3" width="2" height="1" />
</description>',
0x474946383961d100ce00e64000000000ffffff465863cccccc6984953333339999996666666032d51010f
1bdfdfdf84a5ba4f4f4f1f1f1f7f7f7fafafaf003366111618586e7c0116287b9aae003333202020e0e0e0
000b1434424a8080806699991a2125b0b0b03d4d575050504f636f607988021e36e2ebf1b7cedc84a9c194
b5ca99cccc03294ae9f0f40000336b93b01e517dd4e1ea336699739bb65a85a5021a2fcddce6c6d8e3bed3
df97aebc336666a2abb1617d8f11191f164a7849769a666699dbe6ed1a242c8db1c7ffffff000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000021f90401000040002c00000000d100ce000007ff8040828384858687888898a8b
8c8d8e8f90918701061f059797071a1d01929e9fa0a1a2a3a4a3010500a9aaab0007a5afb0b1b2b3a20316
a90c0ebaba0ca8a91ab4c1c2c3c4a103aa0601cacbcba9058703c5d2d3d4b106aa03ccda07bf85a7a90717
d5e3e4e589d7a9d9dacbdcdd83dfabe1e6f3f4d2ed050aebcbbe1309c082b6521968074e5cbd83084dddd3
b72fd50408fe00225be660d5bf84183332827780618001be44408008ec98c07515535988a6b1a54b20015b
793409a0c24892e800a8d347b0e3cb9f07692663f84095cd9b097ce9f4a86c4003672c814aad96f381c79c
376faa2ab0936900821da68a2d46b0eb365559476270e6755d518163ffe3cafacad6633b0c69216c6dab0f
2438b9804df962c0945f5e551df932f4e52ab0e3483113f77d0ae0615ac48a0bff7dcc7991d099460f83cb
ecd59781cea80d55bdaa0a85680043496bd699ba7659bb685fc796cd5465a7da8fe90228c0f46ede0aaaac
f26e6bf203c20b9c803f8247d8a32f15c7b12d579c32aa390d88c34a574453f23aa70eb3a7db9ef9a98579
1794aab2206f7ca10e1319d23c9a15f970b3ec3165d269e4e087cb0006dcb25532f601911380cbe4e45a5a
13d4156066a83c538d70b031330041a97c209e6da93400a1322901901704150e77a16c2679474c4c5c11c5
002b07c808183cc4e106005e79b558dd8ba43dd5583139997795ff7c161860905cf18de6d18d0024b0e25a
3211295b4ac5c0b354660a1c4099330c8a459303b3599616964a6ac9578cc31808c090bc7d88e3882fe5b4
```

db3a86e585259a6ef206172d1ceeb99d0154aa549f46e0ad675d2afc65955fa0b2dd78642c7efd17a80206
8c395c990709878f8090ae3829a5a455a4a13598a1da14880088580f8f6da564ea49aeca864e2c017c905c
aeda3c9028383a4a13195fe8448a8076c0ae938f7ea9c042538fcdae6300934e52f399626326a002965f56
bbcc31ca9d176d2904192aee32617afa01a8c1ac9659a6ab98b86e84738206002999da7baf80b0e6184c3b
fec2e8c001b9947b2f378042bbaf2849fe9bd9b5abd0f724ff29b44a4ca402a8a8a36e006f8922a7c21ab7
c5a9a705c0fb499459961ce031a3720cc0b3daa80a712af99ce8b27eb0cafac9993b07784d62309fc8cdaa
3fa7620209b8901cb447c2e258ac227a3ebd1d37435d43279f005c9434003ffcb080048c396db5b5d85e9c
48a3e19ead985fea30ec552a787a6252d8782f000207003490a3db1eb5bbcabbbf19222acd80b75554cca8
28778ca16fa92dc9dd78574e810011f46d00e2892ff340c0d910425de7dc6559b432d750cbccd1a3505ef9
eb1b789079caa4a35def01392bd866edebdc9875cbdcec1e8091ada7f2faf1788790012e1f037ef22a4a35
cffb471d2b8375431fc3598bf1c877ff4308654fff2aff2b3a4fbf783e32ab733a4337be573cd8de7b3f36
63e59f2d2c03f5d75e11d1c33dab757183da1efce2173f0aecad6f7f135f0014d00ab391ce7728021e001a
a60f5f38875fdc23a0062920bb4f71ae73146b80033ee836b8290382ca68dc6c2e25400dba106fb1838af8
0447bbc49d8f7a5f5a5f5f54e1b5f7bdf08762cb9c033bf739629d6d7fcab82125f2053564602a83408cdf
02f8963f10a24284245c170a9168bd0932e541b2705d14bbb78108144c81ceba87f45065c200a050664314
ce4a6621c631c2ce4568541c9512582d0554a78d3ad4079588438b3ada316c1260621e1513c2cdad4b89ff
0b1c630a1746281ef2072068d922e7d50e06acf1ff425ca40b05cde58e6018728ccbfbe426a1b6c72a6ea9
430b54216b02684a4b467101a850e52a99f2bc1aba497d9af2488aa6168b0ba46203879c62db76f9b24ec6
913791f4510124178c5410c08e653c2333898428235ee87ab3f900258961cd316e008fdb4495021cf0145f
f2a67afa6ac538ca09c4446e2d9d6c6cc7019ec9932ca2ce89f304c0357f9849e1e1135521c49d9686590e
7aba30950795d83adba94b6952531a0e95622e23ea323bb5c295dab0e038ab91d1ee2913a41c75550847f8
36c4d4a3a4c7cb264a539a2b1aae514f078169e5cea93a9ada4f9f102a0b42748a4845fad48a9aa3d9202f
4a0ea216f4a8e2f368477c41c884e814a250ff45e303ac023d9599a3a4b884655617e9d1f0bc54a0953be9
58b7d9cdf92c2aa003fd814cd78acfe72d68a4e444ab5cd149d783963544751b4639edd9d7a3462d1ec484
85351369d0c21e9462f3c95635052a56c766557000ad64652d4bd7b2e25580cb74db0376e1809972561f27
a040294b3132d2a5a85ea33cad62527082b02d0f009ff5c4833a472f56c850b6b33501de54dbb552ec3671
34f12df980db961608b7721e4805531fb110de2af7ba3d65ae365a80bc056c0614d545ee75c7cb4fced2c0
7b0290ee277a850bd22577bcac88ad764920c5ef428223ee85ef782b7a3606f0514bb525607a0130dd494c
1284fa1def2a51910304fa93340126a0ff77e5f9886d252e27095e6e1e65068115707076fcd5460a9eebc2
0127961016061c8633cc2c341e032f088800de08705b06949719238ee2ec1a71dc0bb3d8b7a6ede88f46b2
acd7cdcf9b5ee1ee1813491baab51870b0fa71904b760d3515197906cc5c031cc910195c72c7e728d19481
1565168f596255ceca95bd17c339198abe9764f2d4c29bb83267f8ccffba460cd2b2660282cf9b70bee40f
c05c083a034e3e3f0e2defb81129220fd08563e39bdf048d37398beec09d43b49417c9e815f5f9871c0480
04281d36beb90f08f8259da6cdbc485420604510f834103320635213002ea946eeaaefdc6a15c13ad68f8e
a26a474d6a54bc6740f94df42a42ffdc5154fcdad182a635a97f80837498a405d655f6a9a6f7e26743fb92
aa1500a54bc0026b1b4f06807b8bb637cbbb6e7b1bd8944eef022ef98291981b7e2738db8a13cd6c2a57e6
dddfbea477c56dc715684527622441d0f6fde37e6b2ccd008777bc0130ef28c2405208b7e5096ec0e575d9
39c30e975845441271890b7ae051dcc165321eecb00dd893e2fa788243feaf4e97dce48296f70f5d209a01
9c72c08879708064ae5f9a2fac26370ff8c9014070029680e73dffb972dd49a45d833c8fce4ebad2734e71
a7b300d6f7461ed095fb5ff6587de658f7b5d6717ef20878207e2fd0c1afc37ebcb15f77cb429fcdba8d2e
2ea70069ed6c17742229d0bdff7a3f9beeafb33b7c6d0ca3756fbb76ee067ce0051d810c20efe2de467ce5
149f608532c7f1b88aea9025bf754a0ffe7598cf3ccbc5ee78aa6be3bdcae67bb5204efac953def27883fa
bb358f37ce27baec4b04bdec9b45fbdacbdaf40020fcd34bce7b9783be5eb16178c3d168f3da97bebb0420
40c509587972dfbcf93ff0bde3fdfbfce19319e9d6bfc9f1d37adb547860fbdd13800fbefefdd5d7fdf9f8
f7adf97355fdf4db3e6ce7847ec8c101f05777569274e0277ef9177b689475fea77e2d97371cf0775a5100
f1230007587f3e674be1b780f9275f99a6760f787dce0776c4267619c87cf697781ef87c8d753691378224
f80300407241526b28a8ff7509d8828ef78256138332687b21208287814c3988802bb8793ca86dd9454455
12847c16813fe0012978184d97785508703bb884da067cfaf66f50a86652580036b82213608147a88152c7
85eb86452a0686610881dd837eb026026868803a9884bdc786cfe77a1ad37f71687b00f06abf6687f19785
bba787cec787f8e78531478781488283f86c8698862ab88111a8808cf8636e782f373221912887c8c30165
9817958887488889acb7891ee88794e280a1288ac753006ab22209f0769618715bc88a2de8882f028bb148
82e9757827888579a88af7c78b4bd889440284c1687be74488d95180bd8788aa878c2ca88c6ce88aa4e18c
cfb87e3f5000d608ff00b8978b5aa8881da88d8ce88b5e710dd8f18c79018eb7d668101003c9875ed67878
e8a889ea987fcc882c70088f627881902802a27681f93877fbd88fcac88d1e129002298bf1f72322609007
8990c7b8860ca98c3540033330022019923d300327706b11e9695268391970091e407802969060b7901ba9
8c46e83d26799251386dc6988a1a39939ba84137899312a9931898911cc88f3e997f40498438098e975494
3c799449c98a4b299403a993d568949938953f49404169954e794850a98652c9957c58955639834f0993b7
828d4a68966c8896691996763496 97d89370d98272099629b9964848607899970bb8974249976364978948
1b812998f847ff984dd99762c9963781252cb1988c097a8e799286194588f96b58422030910ad4889497c9

```
629919919b09449d7925b4040430459aa59960a72990a9f943ab991609d09aaea9576f199bcfc701b3098f
b5f942b779132dc24282f09abe897f7768934c89932b809559f96bc78908cab99ca0d79cddf39556199dd2
197ef9688f147608d7899deba69dc8c39d27e91f191002d2799bfe813486509ee6a96ce8793cea2990feb1
0a1e5093398788f1995bbb19577b589f4c189ca1b89f07a020a9e076fe79985918a08b409f06ca62f7f93a
f9198cfb59990bba0a0e5a9759e81b8c40a1159a60175a39199aa0ce20391fc2a000e0762e499c55880d8d
40a2250aff5f273a63cfa9a22e421e1daa0a1c0002318a9107d76423ca9b057aa31996a36193a27128a18d
d001b012a4438a8a7a910e8f60a34a3a75081a84500a0952ca0a547a845812588aa0a55bca0a4cfa034e1a
842d52559f10a6d02301f057949f1909689aa65bd1a50ff8a6020aa6be32a7f3c6032870a7788aa48ba8a7
e37585dbb9a36eca16bc52096a6a032dd2438e005343a8a88bcaa7d6e7a7c14009811a0f9fa0537a93399a
ca0a8c9a9e8edaa7f6250ca0daaa9040544dda7ea79aaaf8b9aae9579dd570012736a1888a659853ab9c0a
78ba2a17ce50a5def3677a6aab188aaba4a702b02a16d0e39e2e146a5bcaac28eaac80c7268f71016cf3a2
2040ff8ddd23017c53a2d8aaa350c8ada80159a990010fca66b48a9de7daa4da9a74ea5a1b7f5500c52845
1260aacb39af6c5aaf258725969a1adecaa0112000e29a9ef1ca9800dba6f6aa9be3c1aee4f8ae2615ac97
f9b002fb6e86da2085f0571cb0aff1a3ac7929b2b7ea7f1debb186a3012e2a00c8da3dd66a9604daa8e997
592a9b081d201f1930b3f143365cc9b3275b7b367bb38a70012012b2a0d6b0ea08b4cd2ab4124bb4894009
2eda922f14691bc9b4d94a7a430bb58ed001a13a1c588b9f4acb88614baf9247a35cbb32201201745ab520
e0af9b58b6010b78689bb6a020b5fcf9b2dd43b27c28b7107b2b466ab7b5f0b50540add50a9bdae6b71b0b
ff6c582ab8afe0ad1e1aae3fe4b34ba8b837b7a18e3b0b149b017a8b65d1d582961b71989bb9850422fafa
43569b7fa1fb6ea34bbac100b9aa90b00b8b3c1b30b63fb6bacfd6baae3b0c9b6bb126f5b68ee74210fba5
bb5b0cf96ab2367976d725bcb84abcc52b0d01c0b2abe0b2a086b119c6bcb90ba9cf5b0e9b2bb795c3b7e3
85bdb0e6bcdb3b0e46bb0a480b6a9f1bbea7e9a9e53b2b0992b7f554aebe25be14a2bdef8b10390b3dde0b
430deb42e71424f89bbf0971beb1dbb66eebaf2f3489c639c0048c1178ab0a54fb43340699782382c5fac0
2f3100846bb8d2991519acc13f61c00d2ab958d9c0e329c262b1b99d1b452311c22a6c26a68bbc40045511
d11ac36201bb0daab0c974af38fc18bd7b4be58a9c3fec18c75bb5435cc4d211bd084bbdde430149acc4f6
d1bdaf4301cab3b5520c1c24ec0c0540bfaa60a659bcc4f13b755e15c61e8b201a90c61dd0abf91b08003b
)

-- Populate Contacts
SET IDENTITY_INSERT [dbo].[Contacts] ON
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (1, N'Christopher Martin',
N'01446 175923', N'07634 717173', N'17 High Street', N'Darlington', N'Cheshire',
'20070914 16:09:21.383', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (2, N'Joseph Bennett', N'01147
771824', N'07956 758471', N'37 Green Lane', N'Portsmouth', N'County Durham', '20070914
16:09:21.400', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (3, N'Joshua Scott', N'01940
919059', N'07331 757163', N'33 George Street', N'Llandudno', N'County Durham',
'20070914 16:09:21.400', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (4, N'Daniel Powell', N'01659
216066', N'07272 134093', N'93 North Street', N'Newport', N'Cheshire', '20070914
16:09:21.417', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (5, N'Jennifer Washington',
N'01502 976179', N'07225 483640', N'43 The Grove', N'Hull', N'Berkshire', '20070914
16:09:21.430', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (6, N'Anthony Lee', N'01200
660059', N'07696 849723', N'97 Stanley Road', N'Newport', N'Berkshire', '20070914
16:09:21.430', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (7, N'Rebecca Foster', N'01340
503829', N'07960 669272', N'79 Springfield Road', N'Salisbury', N'Licolnshire',
'20070914 16:09:21.447', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (8, N'Ava Walker', N'01978
```

```
657197', N'07610 445571', N'29 North Street', N'Southall', N'Oxfordshire', '20070914
16:09:21.463', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (9, N'Matthew Ward', N'01324
940147', N'07903 833764', N'71 Kingsway', N'London EC', N'Staffordshire', '20070914
16:09:21.477', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (10, N'Matthew Mitchell',
N'01661 433253', N'07822 453924', N'70 Cowley Street', N'Worcester', N'Rutland',
'20070914 16:09:21.477', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (11, N'Megan Stewart', N'01653
610021', N'07363 920914', N'59 Broadway', N'Chelmsford', N'Berkshire', '20070914
16:09:21.493', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (12, N'Mia Robinson', N'01594
360107', N'07295 889232', N'22 The Avenue', N'Wigan', N'Norfolk', '20070914
16:09:21.510', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (13, N'Cameron Davis', N'01594
360107', N'07295 889232', N'35 Windsor Road', N'Watford', N'Rutland', '20070914
16:09:21.523', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (14, N'Emma Howard', N'01788
915553', N'07736 274237', N'43 Park Avenue', N'Wigan', N'Derbyshire', '20070914
16:09:21.540', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (15, N'Liam Taylor', N'01286
103534', N'07939 391131', N'47 York Road', N'Harrow', N'Shropshire', '20070914
16:09:21.540', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (16, N'Ross Ramirez', N'01945
464335', N'07126 593543', N'33 Kingsway', N'London WC', N'Herefordshire', '20070914
16:09:21.557', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (17, N'Sarah Lewis', N'01287
924924', N'07149 593067', N'91 George Street', N'Bristol', N'Cumberland', '20070914
16:09:21.570', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (18, N'Emma Taylor', N'01287
924924', N'07149 593067', N'92 Queen Street', N'Dundee', N'Kent', '20070914
16:09:21.587', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (19, N'Anthony Rivera', N'01788
915553', N'07736 274237', N'97 Church Road', N'Southampton', N'Huntingdonshire',
'20070914 16:09:21.587', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (20, N'Ellie King', N'01945
464335', N'07126 593543', N'29 George Street', N'Lincoln', N'Lancashire', '20070914
16:09:21.603', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (21, N'Megan Barnes', N'01469
486350', N'07759 520189', N'92 West Street', N'York', N'Surrey', '20070914
16:09:21.620', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (22, N'Scott Watson', N'01630
243241', N'07886 967048', N'91 Mill Road', N'Chester', N'Norfolk', '20070914
16:09:21.633', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
```

```
[Address2], [Address3], [JoiningDate], [Email]) VALUES (23, N'Emily Anderson', N'01487
403649', N'07449 889487', N'100 George Street', N'Dorchester', N'Hertfordshire',
'20070914 16:09:21.633', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (24, N'Nicole Martin', N'01721
599988', N'07525 825680', N'62 Kingsway', N'Colchester', N'Hertfordshire', '20070914
16:09:21.650', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (25, N'David Perry', N'01212
359994', N'07270 368607', N'59 Queen Street', N'London North', N'Cheshire', '20070914
16:09:21.667', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (26, N'Abbie Rogers', N'01120
243659', N'07237 275885', N'62 Manchester Road', N'Kirkcaldy', N'Cornwall', '20070914
16:09:21.680', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (27, N'Liam Kelly', N'01140
882420', N'07354 721715', N'88 St. John''s Road', N'Manchester', N'Northamptonshire',
'20070914 16:09:21.680', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (28, N'Christopher Walker',
N'01889 655242', N'07332 289915', N'73 North Street', N'Paisley', N'Yorkshire',
'20070914 16:09:21.697', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (29, N'Matthew Collins',
N'01305 665775', N'07352 869339', N'60 York Road', N'Outer Hebrides',
N'Staffordshire', '20070914 16:09:21.713', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (30, N'Elidh Peterson', N'01183
454781', N'07295 215265', N'59 Albert Road', N'Norwich', N'Cheshire', '20070914
16:09:21.727', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (31, N'Jamie Price', N'01624
725814', N'07773 220491', N'40 Manor Road', N'Kingston upon Thames', N'Rutland',
'20070914 16:09:21.743', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (32, N'Katie Sanchez', N'01192
635557', N'07695 694205', N'50 New Street', N'Hemel Hempstead', N'Cornwall', '20070914
16:09:21.743', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (33, N'Erin Rodriguez', N'01701
154719', N'07699 485122', N'8 York Road', N'Swindon', N'Derbyshire', '20070914
16:09:21.760', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (34, N'Abbie Brooks', N'01287
924924', N'07149 593067', N'73 Springfield Road', N'Ipswich', N'County Durham',
'20070914 16:09:21.773', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (35, N'Lauren Simmons', N'01442
337633', N'07986 644216', N'98 London Road', N'London E', N'Cheshire', '20070914
16:09:21.790', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (36, N'Emily Bennett', N'01907
200803', N'07636 884168', N'90 West Street', N'Salisbury', N'Herefordshire', '20070914
16:09:21.790', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (37, N'Christopher Flores',
N'01198 529461', N'07207 549127', N'74 Queens Road', N'Crewe', N'Cheshire', '20070914
16:09:21.807', N'Matt.Mitchell@example.com')
```

```
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (38, N'Alexander Rivera',
N'01120 243659', N'07237 275885', N'25 Grange Road', N'Dudley', N'Huntingdonshire',
'20070914 16:09:21.820', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (39, N'Rachel Williams',
N'01512 965727', N'07889 215998', N'62 Victoria Road', N'Dundee', N'Hampshire',
'20070914 16:09:21.837', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (40, N'Olvia Howard', N'01547
693549', N'07341 808038', N'17 Kings Road', N'Guildford', N'Warwickshire', '20070914
16:09:21.837', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (41, N'Elizabeth Flores',
N'01231 471798', N'07493 384582', N'64 The Green', N'Bournemouth', N'Staffordshire',
'20070914 16:09:21.853', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (42, N'Kyle Griffin', N'01558
179557', N'07522 904690', N'21 Manchester Road', N'Kirkcaldy', N'Hampshire', '20070914
16:09:21.870', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (43, N'Laura Evan', N'01495
153619', N'07390 958114', N'68 Cowley Road', N'St Albans', N'Berkshire', '20070914
16:09:21.883', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (44, N'Katie Martinez', N'01285
948244', N'07424 397578', N'98 Windsor Road', N'Maidstone', N'Kent', '20070914
16:09:21.883', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (45, N'Sophie Griffin', N'01965
513379', N'07605 294144', N'10 The Green', N'Sheffield', N'Surrey', '20070914
16:09:21.900', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (46, N'Elidh Wood', N'01198
529461', N'07207 549127', N'88 North Road', N'Northampton', N'Essex', '20070914
16:09:21.917', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (47, N'Abigail Hill', N'01555
443077', N'07439 738948', N'86 The Drive', N'Sunderland', N'Hampshire', '20070914
16:09:21.930', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (48, N'Scott Bell', N'01469
486350', N'07759 520189', N'25 Park Avenue', N'Harrogate', N'Surrey', '20070914
16:09:21.930', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (49, N'Chloe Martin', N'01329
343662', N'07783 161081', N'33 New Street', N'Telford', N'Dorset', '20070914
16:09:21.947', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (50, N'Lauren Bryant', N'01752
908316', N'07676 584363', N'70 School Lane', N'Swindon', N'Middlesex', '20070914
16:09:21.963', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (51, N'Lucy Rivera', N'01287
924924', N'07149 593067', N'31 The Grove', N'Rochester', N'Cumberland', '20070914
16:09:21.977', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (52, N'Chloe Davis', N'01346
526218', N'07291 685237', N'99 The Crescent', N'Coventry', N'Shropshire', '20070914
```

```
16:09:21.977', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (53, N'Joseph Thompson',
N'01237 661433', N'07302 871821', N'75 Grove road', N'Truro', N'Northumberland',
'20070914 16:09:21.993', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (54, N'William Wright', N'01417
502667', N'07177 363555', N'57 Station Road', N'Romford', N'Middlesex', '20070914
16:09:22.010', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (55, N'William Hughes', N'01407
629329', N'07309 884547', N'34 High Street', N'Torquay', N'Dorset', '20070914
16:09:22.023', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (56, N'Samantha Miller',
N'01333 824728', N'07634 606994', N'60 Park Lane', N'Cardiff', N'Surrey', '20070914
16:09:22.040', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (57, N'William Ramirez',
N'01626 530682', N'07768 951632', N'95 Grange Road', N'Cardiff', N'Berkshire',
'20070914 16:09:22.040', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (58, N'Lauren Green', N'01811
269842', N'07316 264705', N'28 Main Street', N'Huddersfield', N'Somerset', '20070914
16:09:22.057', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (59, N'Jordan Henderson',
N'01624 725814', N'07773 220491', N'67 The Drive', N'Tonbridge', N'Surrey', '20070914
16:09:22.070', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (60, N'Michael Richardson',
N'01346 526218', N'07291 685237', N'76 Highfield Road', N'Kilmarnock', N'County
Durham', '20070914 16:09:22.087', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (61, N'Caitlin Washington',
N'01291 914147', N'07913 286110', N'55 School Lane', N'Peterborough', N'Licolnshire',
'20070914 16:09:22.103', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (62, N'Anna Long', N'01131
669331', N'07113 804523', N'41 The Avenue', N'Stevenage', N'Cornwall', '20070914
16:09:22.103', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (63, N'Ethan Parker', N'01940
919059', N'07331 757163', N'76 York Road', N'Blackpool', N'Middlesex', '20070914
16:09:22.120', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (64, N'Lauren Ramirez', N'01417
502667', N'07177 363555', N'74 Albert Road', N'Crewe', N'Kent', '20070914
16:09:22.133', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (65, N'Joshua Price', N'01404
421474', N'07720 314778', N'24 The Green', N'Southend on Sea', N'Yorkshire', '20070914
16:09:22.150', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (66, N'Kyle Botton', N'01291
914147', N'07913 286110', N'25 Kingsway', N'Aberdeen', N'Lancashire', '20070914
16:09:22.150', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (67, N'Bethany Hughes', N'01210
```

```
611735', N'07709 761071', N'6 New Street', N'Stevenage', N'Buckinghamshire', '20070914
16:09:22.167', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (68, N'Kyle Smith', N'01555
443077', N'07439 738948', N'90 Main Street', N'Cardiff', N'Buckinghamshire', '20070914
16:09:22.180', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (69, N'Caitlin Griffin',
N'01469 486350', N'07759 520189', N'100 Mill Road', N'Oxford', N'Sussex', '20070914
16:09:22.197', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (70, N'Jamie Perez', N'01572
459202', N'07858 711874', N'86 Queens Road', N'Nottingham', N'Staffordshire',
'20070914 16:09:22.213', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (71, N'Jennifer Turner',
N'01192 635557', N'07695 694205', N'46 London Road', N'Brighton', N'Kent', '20070914
16:09:22.213', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (72, N'Alexander Barnes',
N'01281 554917', N'07800 642581', N'78 North Street', N'London W', N'Lancashire',
'20070914 16:09:22.227', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (73, N'Megan Campbell', N'01205
396727', N'07198 312473', N'20 Park Avenue', N'Blackburn', N'Suffolk', '20070914
16:09:22.243', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (74, N'Lewis Collins', N'01237
661433', N'07302 871821', N'10 Victoria Road', N'Southall', N'Norfolk', '20070914
16:09:22.260', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (75, N'Bethany Peterson',
N'01784 976829', N'07885 861269', N'43 The Crescent', N'Worcester', N'Surrey',
'20070914 16:09:22.260', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (76, N'Erin Martin', N'01428
428418', N'07377 251946', N'39 The Avenue', N'Milton Keynes', N'Suffolk', '20070914
16:09:22.273', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (77, N'Joseph Coleman', N'01788
915553', N'07736 274237', N'95 Park Avenue', N'Wolverhampton', N'Leicestershire',
'20070914 16:09:22.290', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (78, N'Jordan Scott', N'01500
424473', N'07824 778422', N'12 Stanley Road', N'Galashiels', N'Cambridgeshire',
'20070914 16:09:22.307', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (79, N'Kieran Kelly', N'01633
392324', N'07960 870808', N'91 London Road', N'Twickenham', N'Northumberland',
'20070914 16:09:22.320', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (80, N'Tony Brown', N'01231
471798', N'07493 384582', N'41 Queensway', N'Falkirk', N'Berkshire', '20070914
16:09:22.337', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (82, N'Bethany Rivera', N'01784
976829', N'07885 861269', N'54 Church Street', N'London E', N'Northumberland',
'20070914 16:09:22.353', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
```

```
[Address2], [Address3], [JoiningDate], [Email]) VALUES (83, N'Laura Hayes', N'01982
956405', N'07109 544766', N'56 Station Road', N'Brighton', N'Devon', '20070914
16:09:22.370', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (84, N'Anna Griffin', N'01345
471798', N'07493 384582', N'40 Windsor Road', N'Guildford', N'Cornwall', '20070914
16:09:22.383', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (85, N'Samantha Jackson',
N'01231 471798', N'07493 384582', N'14 New Street', N'Llandudno', N'County Durham',
'20070914 16:09:22.400', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (86, N'Kyle Sanchez', N'01978
657197', N'07610 445571', N'3 Windsor Road', N'Bromley', N'Oxfordshire', '20070914
16:09:22.400', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (88, N'Joseph Gray', N'01211
864653', N'07375 760438', N'56 South Street', N'London E', N'Gloucestershire',
'20070914 16:09:22.430', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (89, N'Courtney Perry', N'01210
611735', N'07709 761071', N'13 St. John''s Road', N'York', N'Berkshire', '20070914
16:09:22.447', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (90, N'Matthew Campbell',
N'01743 360803', N'07394 612512', N'68 Queen Street', N'Romford', N'Essex', '20070914
16:09:22.463', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (91, N'Alexis Cook', N'01624
725814', N'07773 220491', N'36 Queen Street', N'Falkirk', N'Norfolk', '20070914
16:09:22.463', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (92, N'Jamie Mitchell', N'01776
202627', N'07185 505540', N'90 High Street', N'Kingston upon Thames', N'Warwickshire',
'20070914 16:09:22.477', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (93, N'Chloe Morris', N'01725
222331', N'07526 574605', N'97 Main Street', N'Bradford', N'Cambridgeshire', '20070914
16:09:22.493', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (94, N'Lauren Morris', N'01914
784723', N'07798 945002', N'44 Church Street', N'Durham', N'Warwickshire', '20070914
16:09:22.510', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (95, N'Olvia Barnes', N'01811
269842', N'07316 264705', N'78 Stanley Road', N'Lincoln', N'Herefordshire', '20070914
16:09:22.510', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (96, N'Olvia Hall', N'01571
657822', N'07694 711784', N'14 Windsor Road', N'Coventry', N'Middlesex', '20070914
16:09:22.523', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (97, N'Scott Turner', N'01555
443077', N'07439 738948', N'87 Station Road', N'Salisbury', N'Surrey', '20070914
16:09:22.540', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (98, N'Rebecca Hayes', N'01802
342584', N'07296 315860', N'94 George Street', N'Bournemouth', N'Somerset', '20070914
16:09:22.557', N'Matt.Mitchell@example.com')
```

```
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (99, N'Kyle Perry', N'01653
610021', N'07363 920914', N'82 The Grove', N'Derby', N'Gloucestershire', '20070914
16:09:22.570', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (100, N'Kieran Patterson',
N'01389 465402', N'07266 836025', N'3 St. John''s Road', N'Bolton', N'County Durham',
'20070914 16:09:22.570', N'Matt.Mitchell@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (101, N'Tony Botton', N'01467
283712', N'07123 873472', N'56 Albert Square', N'Greater London', N'London', '20070919
09:16:00.000', N'Tony.Botton@example.com')
INSERT INTO [dbo].[Contacts] ([ID], [Name], [PhoneWork], [PhoneMobile], [Address1],
[Address2], [Address3], [JoiningDate], [Email]) VALUES (102, N'Robert Brown', N'01325
348273', N'07263 764373', N'652 Long Road', N'Northampton', N'Kent', '20070919
09:17:00.000', N'Robert.Brown@example.com')SET IDENTITY_INSERT [dbo].[Contacts] OFF

-- Populate WidgetPurchases with lots of data
DECLARE @purchaseCount int
SET @purchaseCount = 0
SET NOCOUNT ON
DECLARE @temp float
SET @temp = RAND(1)
WHILE (@purchaseCount < 108435) BEGIN
 INSERT INTO [dbo].[WidgetPurchases] ([WidgetPriceID], [Quantity], [InvoiceNumber],
[Date]) VALUES (
  CONVERT (int, RAND() * 3) + 1, -- WidgetPriceID
  CONVERT (int, RAND() * 100) + 1, -- Quantity
  'WIDG' + convert (nvarchar(10), convert (int, RAND() * 10000) + 1), -- InvoiceNumber
  CONVERT (datetime, -- Date
   CONVERT (nvarchar (50),
    '2007-' +
    CONVERT (nvarchar (2), CONVERT (int, RAND() * 11) + 1) + '-' +
    CONVERT (nvarchar (2), CONVERT (int, RAND () * 27) + 1)
   )
  )
 )
 SET @purchaseCount = @purchaseCount + 1
END
SET NOCOUNT OFF
```

```
COMMIT TRANSACTION
GO
```