SQL Backup 6 documentation

About SQL Backup

SQL Backup creates compressed, encrypted backups of Microsoft SQL Server databases.

For more information, see the SQL Backup Pro product page.

Quick links

SQL Backup 6.5 release notes

Getting started: Backing up

Errors and warnings

SQL Backup is now only available in the SQL Backup Pro edition. SQL Backup Standard and SQL Backup Lite have been retired.

If you are currently using SQL Backup Standard or SQL Backup Lite, please email dba.info@red-gate.com for a free upgrade to SQL Backup Pro.

Requirements

Microsoft SQL Server support

SQL Backup supports the following versions of Microsoft SQL Server:

- SQL Server 2008 R2 (32-bit, 64-bit, and Itanium)
- SQL Server 2008 (32-bit, 64-bit, and Itanium) - Service Pack 1 is supported but not required.
- SQL Server 2005 (32-bit, 64-bit, and Itanium) - Service Packs 1, 2 and 3 are supported but not required.
- SQL Server 2000 (32-bit, 64-bit, and Itanium) - Service Pack 3a or later is required.

The following editions of each version are supported:

- Developer Edition
- Enterprise Edition
- Standard Edition
- Web Edition
- Workgroup Edition

SQL Backup provides limited support for SQL Server Express Edition; the SQL Backup graphical user interface does not support scheduled backup jobs, scheduled restore jobs or log shipping with this edition.

Microsoft Windows support

SQL Backup supports the following Microsoft Windows operating systems:

- Windows 7
- Windows Vista Service Packs 1 and 2
- Windows Server 2008 R2
- Windows Server 2008 Service Packs 1 and 2 (Standard Edition and Windows Server 2008 Enterprise Edition)
- Windows XP Professional Service Packs 1, 2 and 3
- Windows Server 2003 Standard Edition and Windows Server 2003 Enterprise Edition:For 32-bit versions of Windows, Service Pack 1 and Service Pack 2 are supported but not required.For 64-bit versions of Windows, Service Pack 1 is required. You must install this separately for Itanium; for x64 it is installed automatically. Service Pack 2 is supported but not required.
- Windows 2000 Professional Service Pack 4
- Windows 2000 Server Service Pack 4
- Windows 2000 Advanced Server Service Pack 4

Operating system requirements

- 512MB RAM
- 100MB hard disk space

.NET framework requirements

The graphical user interface requires Microsoft .NET version 2, 3.0 or 3.5.

Platform support for x86 64-bit and x86 32-bit

If you run 32-bit and 64-bit machines, note these platform support details:

- The SQL Backup GUI is a .NET 2.0 Windows Forms application, compiled to run in 32-bit mode only. There should be no performance degradation caused by this limitation on x86 64-bit machines.
- The SQL Backup Agent service is a native 32-bit process. It cannot be run in 64-bit mode. This should not degrade performance on an x86 64-bit machine.
- SQL Backup's extended stored procedures are available in 32-bit and 64-bit versions. Use the 32-bit versions with 32-bit versions of SQL Server, the 64-bit versions with 64-bit versions of SQL Server.
- SQL Backup's command line tools are all 32-bit processes. They cannot be run in 64-bit mode. This should not degrade performance on an x86 64-bit machine.

Technical notes

- SQL Backup does not support cumulative backups in a single file.
- SQL Backup fully supports FILESTREAM storage introduced in SQL Server 2008.

- To restore from a backup file that was created with SQL Backup 6 or later to a SQL Server instance running SQL Backup 5, you must first convert the backup file.

Installing

SQL Backup comprises two main parts:

- The *SQL Backup graphical user interface (GUI)* for setting up and monitoring your backup and restore jobs.The GUI is not licensed and can be installed on as many computers as you wish. You can install only one version of the SQL Backup GUI on each computer.
  To check which version of the graphical user interface is installed, from the **Help** menu select **About SQL Backup**.
- The *SQL Backup server components* installed on each SQL Server instance (or on each node of a clustered instance).This is the licensed part of SQL Backup and contains the SQL Backup extended stored procedures and the SQL Backup Agent service. You can install only one version of the SQL Backup server components on each SQL Server instance.
  To check which version of the server components is installed, in the SQL Backup GUI select the SQL Server, then from the **Edit** menu select **Properties**.

Installation process

Before you install SQL Backup, check the Requirements to confirm that your systems are compatible with SQL Backup.

1. Install the SQL Backup graphical user interface (GUI). For details, see Installing the graphical user interface.
2. Add the SQL Server instances on which you want to back up and restore databases with SQL Backup to the GUI. For details, see Adding SQL Server instances.
3. Install the SQL Backup server components on the SQL Server instances. You can use the graphical user interface to do this, or you can run the setup wizard manually. Both methods are described in Installing the server components on a SQL Server instance.
4. Activate the licenses on the SQL Server instances as necessary. For details, see Licensing.

Installing the graphical user interface

Install the SQL Backup graphical user interface (GUI) on the computer from which you want to run the backups, using the SQL Backup setup program *SQLBackup.exe*. If an earlier version of the SQL Backup GUI is already installed on the computer, it will be upgraded.

> You cannot install more than one version of the SQL Backup GUI on the same computer.

The following files are installed in the installation folder. By default this is:

- *%Program Files%\Red Gate\SQL Backup 6* on 32-bit servers.
- *%Program Files (x86)%\Red Gate\SQL Backup 6* on 64-bit servers.

You can specify an alternative location using the installation wizard.

RedGate.SQLBackup.UI.exe	The SQL Backup graphical user interface.
SQBServerSetup.exe	The SQL Backup server components installer.
ProductActivation.exe	The program for activating SQL Backup without using the graphical user interface. For example, you can use this program to activate cluster nodes. For more information see Activating.
CompressionAnalyzer.exe	The SQL Backup program to compare the effectiveness of different compression levels on a full backup of a selected database. Read more about the Compression Analyzer.
SQBConverter.exe and SQB ConverterGUI.exe	*SQBConverter.exe* is a command-line program to convert SQL Backup files to Microsoft Tape Format (MTF) files. This program will also convert files created using SQL Backup version 6 to files compatible with SQL Backup version 5 and earlier. *SQBConverterGUI.exe* is a graphical user interface for the *SQBConverter.exe* co mmand-line program. You can distribute this program as required; you do not need the full product installation to use it. For more information see SQL Backup File Converter.
sqb2mtf.exe	A command-line program to convert SQL Backup files to Microsoft Tape Format (MTF) files. This program is superseded by the *SQBConverter.exe* program described above. sqb2mtf.exe is provided with SQL Backup 6 so that any existing scripts you may have written that use this program will continue to work. You can distribute this program as required; you do not need the full product installation to use it.
SQBMaintPlanConv.exe	This program converts maintenance plans created with SQL Server Management Studio or SQL Server Enterprise Manager. It disables native backup tasks and replaces them with tasks for SQL Backup jobs, enabling you to specify compression and encryption settings. For more information see Maintenance Plan Conversion Wizard.

Additionally, the SQBObjectLevelRecovery folder contains SQLObjectLevelRecoveryPro.exe, the executable file for the SQL Object Level Recovery Pro application.

Once you have installed or upgraded the SQL Backup GUI, add the SQL Server instances on which you want to use SQL Backup, and install the server components.

Adding SQL Server instances

To back up or restore a database, you must first add the SQL Server instance to the SQL Backup graphical user interface (GUI). SQL Backup displays the list of SQL Server instances that have been added in the Registered SQL Servers pane.

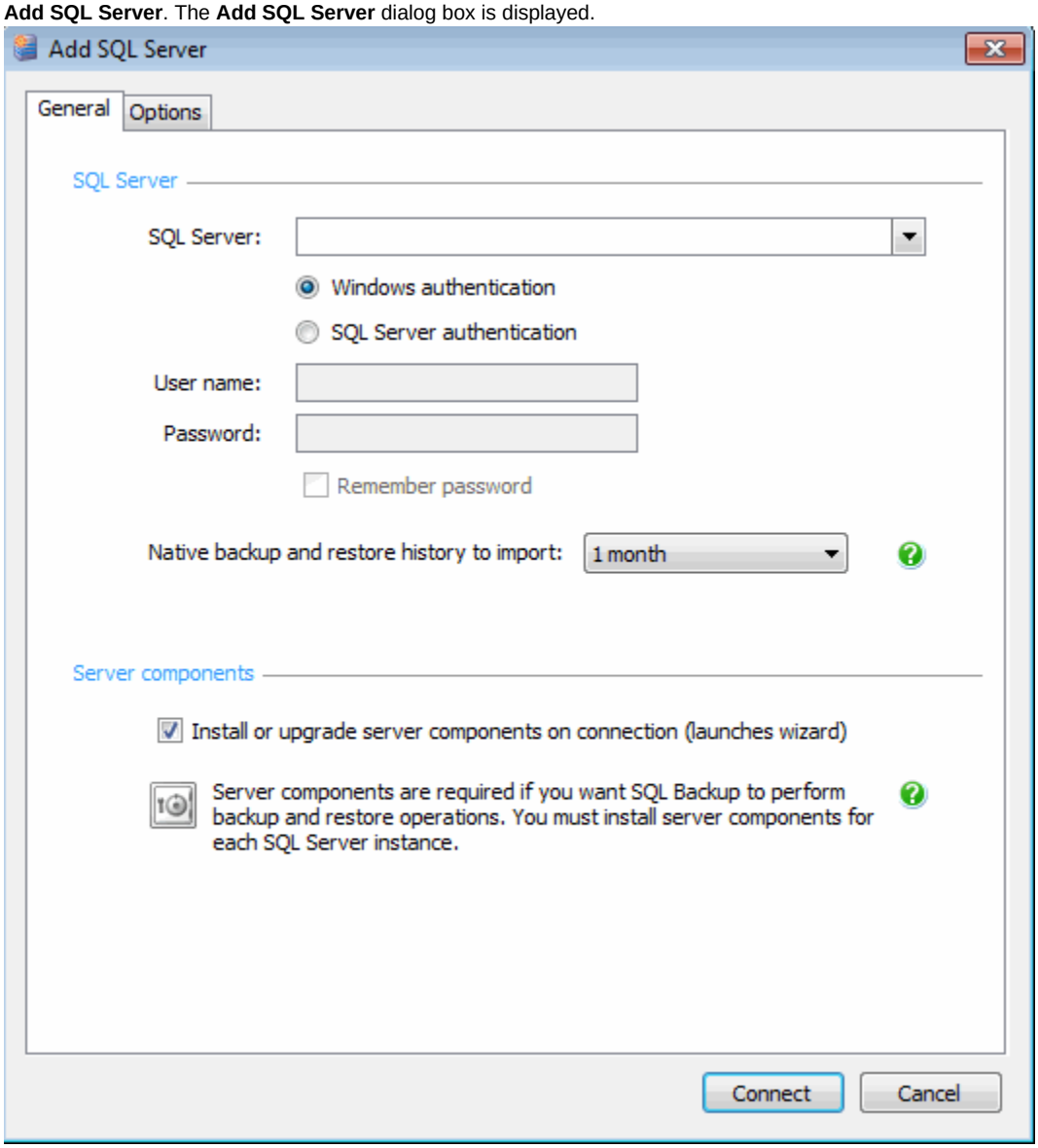
You can add the SQL Server instances by:

- specifying the instances by name
- importing registered SQL Servers from Microsoft SQL Server Management Studio, Microsoft SQL Server Enterprise Manager, or SQL Backup 4.

When you have added a SQL Server instance, you can:

- edit the registration to change the authentication or connection properties
- removing an individual SQL Server instance registration, or an entire group

Adding SQL Server instances by name

1. If you have set up more than one location and time zone, select the tab for the location in which you want to register the SQL Server instance.
2. On the **File** menu, click
   
   Add SQL Server. The **Add SQL Server** dialog box is displayed.
   

3. Type or select the name of the SQL Server instance in the **SQL Server** box.
   If you experience problems selecting a SQL Server instance that is not running on the LAN, for example if you are accessing the SQL

Server instance via an internet connection, you may need to create a SQL Server alias using TCP/IP; refer to your SQL Server documentation for details. You can then type the SQL Server alias name in the **SQL Server** box to connect to the remote SQL Server. (Note that this SQL Server alias is not the same as the SQL Backup alias, in the **Options** tab.)

4. Select the authentication method for connecting to the SQL Server.
   For Windows authentication, the credentials of the user currently connecting to the GUI are used. The user must be a member of the *sys admin* fixed server role.
   For SQL Server authentication, enter the user name and password for a user who is a member of the *sysadmin* fixed server role. The password will be encrypted. If you do not select **Remember password**, when you next start SQL Backup
   
   will be displayed next to the SQL Server instance name to indicate that SQL Backup has not attempted to connect to the SQL Server instance; double-click the icon or right-click and select **Edit** to enter the authentication details.
5. By default, SQL Backup imports the last **1 month** of history for *native* backup and restore operations into an internal cache. You can change the amount of native activity history to import, but should be aware that it may take a long time to import and display all the activity history for the SQL Server. The time taken to import the data depends on the number of databases on the SQL Server, the length of time the SQL Server has been in service, and the amount of activity on the SQL Server. You can choose **None**, **1 week**, **1 month**, **3 months**, **6 months**, **1 year**, or **All**.

   > If the SQL Server instance has previously been used with SQL Backup (version 4 or above), *all* activities that were created by SQL Backup are imported, irrespective of the time period you select for importing native activity history.

6. By default, when you add a SQL Server instance, the **Install or Upgrade Server Components** wizard is displayed so that you can install or upgrade the server components; SQL Backup requires server components to be installed on each SQL Server instance so that it can perform backup and restore operations. If you do not want to install or upgrade the server components now, clear the **Install or upgrade server components on connection** check box.
   For more information about installing the server components, see Installing the server components on a SQL Server instance.
   For more information about upgrading, see Upgrading the server components.
   For details of the permissions required by the SQL Backup Agent service, see Permissions.
7. On the **Options** tab, under **Location and group**, select the location in which the SQL Server is situated.
   For more information about locations, see Time settings and locations.
8. From the **Group** list, select the SQL Server group to which you want to add the SQL Server instance.
   For more information about groups, see Managing SQL Server groups.
9. To set up a SQL Backup alias for the SQL Server instance, in **Alias** type the alias that you want to use. The alias will be displayed throughout the SQL Backup graphical user interface.
10. To change the connection properties from their default values, under **Connection Properties** edit the values as required. For details, see Connection properties below.
11. Click **Connect**.

SQL Backup adds the SQL Server instance. This may take a few minutes. The SQL Server instance is displayed in the Registered SQL Servers pane.

If there was a problem with the authentication,

is displayed next to the SQL Server instance name. For example, this is displayed if you have entered the password incorrectly. Click the icon to correct the authentication details.

If there was a problem connecting to the SQL Server instance,

is displayed next to the SQL Server instance name. Click the icon to see further details of the problem. You can also try the following to rectify the problem:

1. Verify that the SQL Server is online and that the SQL Server name is listed in your LAN by *pinging* the address.
   For example, open a command prompt and run the following command:

   ```
   ping: <server name>
   ```

2. If the SQL Server is online, verify that you are connecting to the correct port. If your SQL Server is not running on the default port (1433), in the **Add SQL Server** dialog type the following in the **SQL Server** box: *<server name\instance>,<port>*
   For the default instance, use: *<server name>,<port>*
3. If you are sure that you are connecting to the correct port, force SQL Backup to use the TCP/IP network protocol when it makes the connection by setting the connection properties for the SQL Server (see Connection properties below).

If you chose to install or upgrade the server components, the **Install or Upgrade Server Components** wizard is displayed to guide you through the process. For more information, see Installing the server components on a SQL Server instance.

Connection properties

To change the connection properties:

1. On the **Add SQL Server** dialog box, select the **Options** tab.
2. Select the required network protocol from the **Network protocol** list.
   The available client protocols are those configured in Microsoft SQL Server using the Client Network Configuration in Computer Management.
3. In **Network packet size**, type or select the size of the network packets to be sent. The default value is 4096 bytes.
4. In **Connection time-out** type or select the number of seconds to wait for a connection to be established before timing out. The default setting is 15 seconds.
5. In **Execution time-out**, type or select the number of seconds to wait before execution of a task is stopped. The default value is zero seconds (no time-out).
6. To force the connection to be encrypted, select the **Encrypt connection** check box.

Importing SQL Server instances

To select SQL Server registrations that exist already on SQL Backup version 4, SQL Server Management Studio, or SQL Server Enterprise Manager:

1. On the **File** menu, click **Import SQL Servers**. The **Import SQL Server Registrations** dialog box is displayed showing any existing SQL Server registrations in a tree structure.
2. Select the check boxes for the SQL Server registrations that you want to import.
3. In the **Location** list, select the location into which you want to import the SQL Servers. For more information about locations, see Time settings and locations.
4. Click **Import**.

SQL Backup imports the SQL Server instances.

Editing a SQL Server instance registration

To change the SQL Server authentication or the connection properties:

1. Right-click the name of the SQL Server instance in the **Registered SQL Servers** pane and click **Edit**.
2. Edit the registration settings and connection properties as required.
3. Click **Connect**.

Removing SQL Servers

To remove a SQL Server instance from the Registered SQL Servers pane, do one of the following:

- Right-click the name of SQL Server instance in the Registered SQL Servers pane, and click **Delete**.
- Click the name of SQL Server instance in the Registered SQL Servers pane, then on the **Edit** menu, click **Delete**.

To remove all SQL Server instances in a SQL Server group and delete the group, do one of the following:

- Right-click the name of the group in the Registered SQL Servers pane, and click **Delete**.
- Click the name of the group in the Registered SQL Servers pane, then on the **Edit** menu, and click **Delete**.

> Removing SQL Server instances does not affect any SQL Backup jobs on the instances.

Installing the server components on a SQL Server instance

You can install the SQL Backup server components on a SQL Server instance in two ways:

- using the SQL Backup graphical user interface (GUI)
- installing manually

Usually, using the SQL Backup GUI is the easiest way to install the server components. You may prefer to install the server components manually if your SQL Server security does not allow you to install remotely, or if you want to choose the location of the SQL Backup files.

> If you are installing on a cluster, you should install the server components manually. See Installing the server components on a SQL Server cluster.

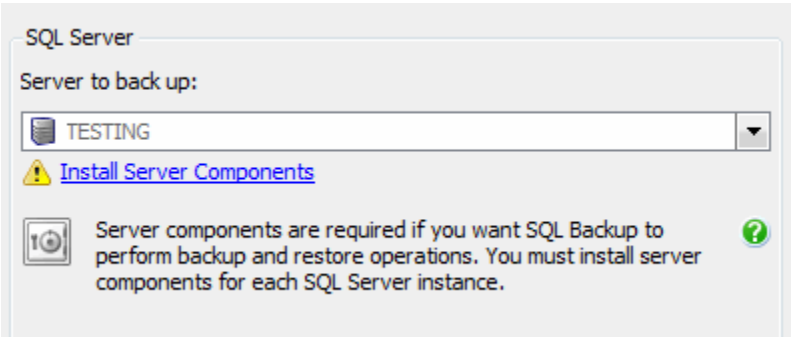
Installing the server components using the GUI

1. Open the SQL Backup graphical user interface.
2. In the Registered SQL Servers pane, select the SQL Server instance on which you want to install the server components. One of the following icons should be displayed:

   
   indicates that you have not installed the SQL Backup server components on the SQL Server.
   
   indicates that the SQL Backup version 4 (or earlier) server components are installed on the SQL Server. You must upgrade the server components to use SQL Backup 6 with this SQL Server instance. For more information, see Upgrading the server components.
   
   indicates that the SQL Backup server components are installed and licensed on this SQL Server, but the version of the server components is not up-to-date compared with your version of the graphical user interface. For more information, see Upgrading the server components.

   If the SQL Server instance is not listed, add the SQL Server instance with which you want to use SQL Backup. For more information, see Adding SQL Server instances.
3. Click the icon, or from the **Actions** menu, select **Install or Upgrade Server Components**. You will be asked to specify the account details for the SQL Backup Agent service (the startup account) and the authentication mode for the service to connect to the SQL Server instance. For details of the permissions required by the SQL Backup Agent service startup account, see Permissions.
4. Ensure that no other SQL Backup processes are currently running on the SQL Server instance, and click **Finish** to start the installation. A message dialog box shows the progress of the installation process.
   The server components are installed in the default location:
   - *%ProgramFiles%\Red Gate\SQL Backup 6\<instance name>* on 32-bit machines
   - *%ProgramFiles(x86)%\Red Gate\SQL Backup 6\<instance name>* on 64-bit machines
5. If you are installing the server components for the first time, a 14 day trial of SQL Backup will be started, during which you can evaluate the product. This is indicated by the trial icon
   
   .You can activate the server components with a SQL Backup license at any point. For more information, see Activating.

You may also be prompted to install the server components when setting up a backup or restore, or configuring log shipping. When you select a server that does not have the SQL Backup server components installed, a warning icon is displayed and you will not be able to click **Next** to proceed.



Click the **Install Server Components** link to install the components.

Installing the server components manually

You may wish to install SQL Backup on your SQL Server instances manually using the *SQBServerSetup.exe* executable if your SQL Server security does not allow you to install remotely, or if you want to customize the installation.

To install the server components manually:

1. Copy *SQBServerSetup.exe* from your SQL Backup installation folder to the server on which you want to install the components. By default, for 32-bit servers the installation folder is *%ProgramFiles%\Red Gate\SQL Backup 6*; for 64-bit servers it is *%ProgramFiles(x86)%\Red Gate\SQL Backup 6*.
2. Run *SQBServerSetup.exe* from the server. The Setup wizard guides you through the installation process. You can specify:
    - The instance on which you want to install the server components.
    - The folder in which you want to install the server components. By default this is *%ProgramFiles%\Red Gate\SQL Backup 6\<instance name>* on 32-bit systems and *%ProgramFiles(x86)%\Red Gate\SQL Backup 6\<instance name>* on 64-bit systems.
    - The startup user for the SQL Backup Agent service application. For details of the permissions required by the SQL Backup Agent service startup account, see Permissions.
    - The authentication method to be used for the SQL Server.
    - The folder in which you want to install files relating to the SQL Server Compact database, which SQL Backup uses to store information about backup and restore operations. By default this is:
        - *%PROGRAMDATA%\Red Gate\SQL Backup\Data\<instance name>* (Windows Vista, Windows 2008 and later), or
        - *%ALLUSERSPROFILE%\ Application Data\Red Gate\SQL Backup\Data\<instance name>* (Windows XP and Windows 2003).
    - The folder in which you want to store SQL Backup activity log files. SQL Backup creates a log for each backup or restore process it runs. By default this is:
        - *%PROGRAMDATA%\Red Gate\SQL Backup\Log\<instance name>* (Windows Vista, Windows 2008 and later), or
        - *%ALLUSERSPROFILE%\ Application Data\Red Gate\SQL Backup\Log\<instance name>* (Windows XP and Windows 2003).

    Note that *SQBShellExt.dll* and *SQBShellExt.reg* are always installed in *%ProgramFiles%\Common Files\Red Gate\SQL Backup* for 32-bit versions of Windows (*Win32*), or *%ProgramFiles(x86)%\Common Files\Red Gate\SQL Backup* for 64-bit versions of Windows (*x64* and *Itanium*).
3. Use the SQL Backup GUI to activate SQL Backup on the SQL Server instances as required. For more information, see Activating.

For information on installing SQL Backup unattended using the command line, see Installing the server components from the command line.

The SQL Backup server components comprise:

SQBCoreService.exe	The SQL Backup Agent service application.
xp_sqlbackup.dll	The SQL Backup extended stored procedure dynamic-link library. This will be located in a subfolder called *Win32*, *x64*, or *Itanium* depending on the SQL Server version.
SQLBackupC.exe	The SQL Backup command line interface.
zlib1.dll	The compression library used by SQBCoreService.exe and SQLBackupC.exe
RedGate.BackupReader.CryptoHelper.dll	The SQL Backup Reader dynamic-link library.
SQBShellExt.dll	The SQL Backup Windows shell extension library. This will be located in *%ProgramFiles%\Common Files\Red Gate\SQL Backup* or *%ProgramFiles(x86)%\Common Files\Red Gate\SQL Backup*.
SQBShellExt.reg	The registry information used to register the SQL Backup shell extension. This will be located in *%ProgramFiles%\Common Files\Red Gate\SQL Backup* or *%ProgramFiles(x86)%\Common Files\Red Gate\SQL Backup*.

Installing the server components on a SQL Server cluster

You can install the SQL Backup server components on all SQL Server instances in a cluster at the same time. If you have a SQL Backup multi-server license, you can then activate SQL Backup on all of the nodes.

> If you are using SQL Backup with clustered SQL Servers, you are strongly recommended to save backup files on a shared drive within the cluster. This ensures that the backups are available in the event of failure of one of the clustered SQL Servers.

When you install the server components on a cluster, the SQL Backup Agent service application is clustered so that it can run backup and restore operations on whichever cluster node is active. In addition, the following registry keys in the *HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\SQL Backup* entry are replicated:

- BackupSettingsGlobal\<instance name>
- BackupSettings\<instance name>

These registry keys contain details of saved SQL Server settings, templates, and other resources that are specific to SQL Backup.

SQL Backup scheduled backup jobs, scheduled restore jobs and log shipping jobs are replicated by the SQL Server instance.

It is also possible to install the server components on a SQL Server cluster unattended using the command line. For more information, see Installing the server components from the command line.

Prerequisites

For the installation, you will need to know:

- The account name and password of the administrator for all cluster nodes.
- The account name and password that the SQL Backup Agent service application will run as; you are recommended to use a domain account rather than a local or built-in account.

Before you install the SQL Backup server components for the first time, consider the following:

- On which clustered instance do you want to install the components?
- Do you want to use the Windows authentication of the account the SQL Backup Agent service runs as, or a separate SQL Server authentication?

The recommended method for installing the server components on a cluster involves running *SQBServerSetup.exe* from one of the clustered machines on which the components will be installed. When you install in this way, more information is provided if a problem occurs (particularly for post-installation checks). Note that, from SQL Backup 6.3, it is no longer possible to use the SQL Backup graphical user interface (GUI) to remotely install server components to clustered servers.

Upgrading from a previous release

If you are upgrading from a previous release of the SQL Backup server components to SQL Backup 6, prior to installation you must do the following:

- Ensure that no existing SQL Server Agent jobs involving SQL Backup tasks are running, and none will execute during the installation process.
- Ensure that no backup processes that use SQL Backup are running, and none will start during the installation process.
- Ensure that you have uninstalled SQL Backup version 4.x installed on any of the servers.
- In the Cluster Administrator or Failover Cluster Management tool, disable the SQL Backup Resource. This prevents the cluster from being failed over if there is a problem with the SQL Backup Agent during the upgrade process.

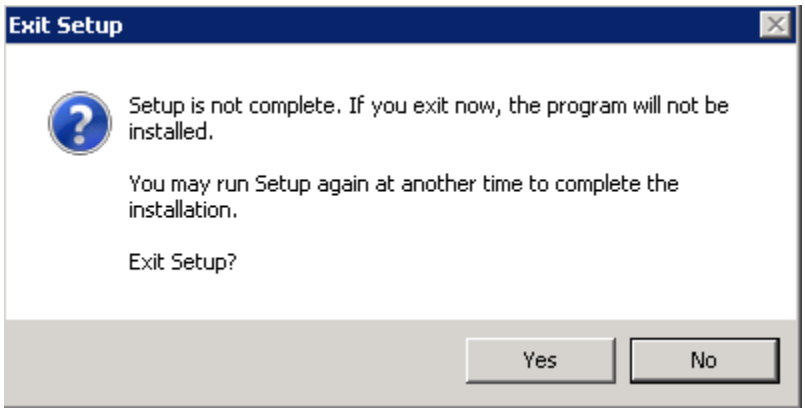
Installing using SQBServerSetup.exe

To install SQL Backup on a SQL Server cluster, copy *SQBServerSetup.exe* to the active node on the cluster, and run the program.

1. Welcome page

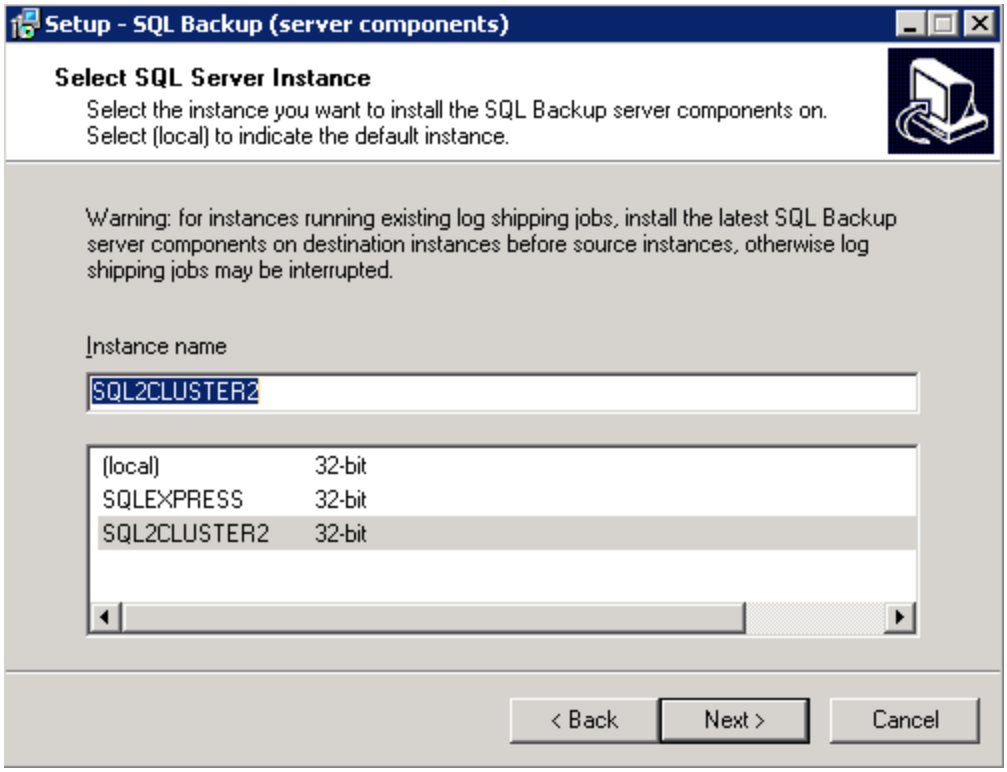
The following page is displayed:

You can cancel the installation at any time prior to the files being copied by clicking **Cancel**. The following dialog box will be displayed. Click **Yes** t o cancel the installation; no files will be installed.



2. Selecting instances

When you click **Next** on the Welcome page, a list of available instances is displayed. The list contains instances local to the computer from which you are running the setup program, and clustered instances for the SQL Server cluster that the computer is participating in.



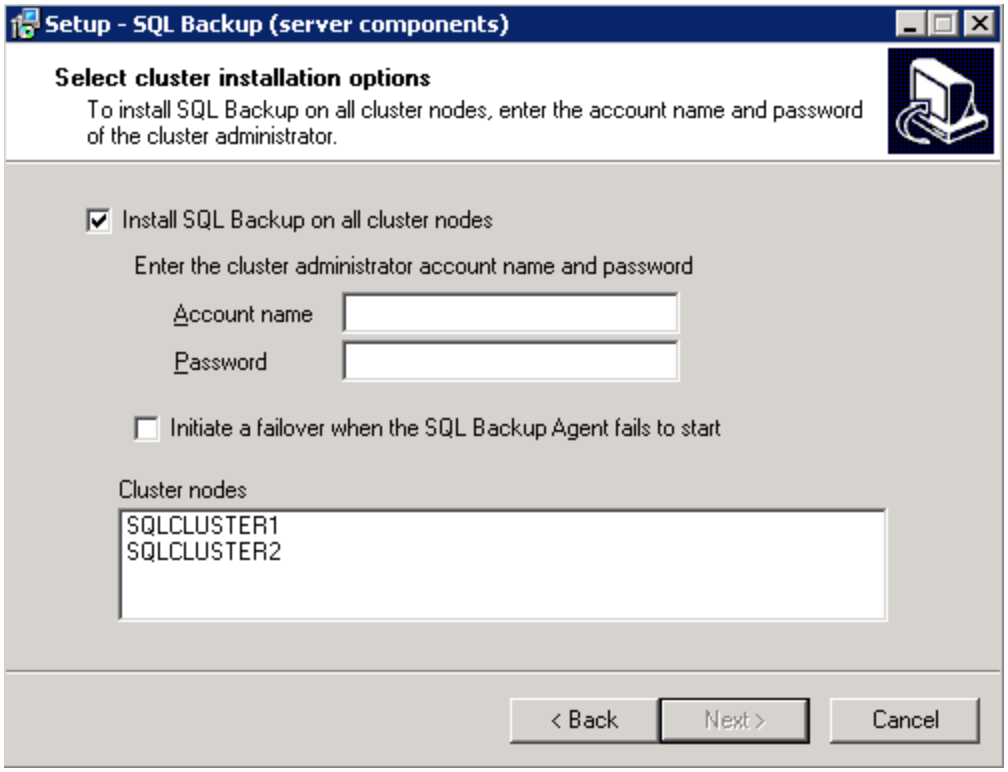
The information in the lower pane indicates which architecture the SQL Server instance is using: 32-bit, 64-bit, or Itanium. For example, a SQL Server 2000 instance running on a 64-bit computer is listed as using 32-bit architecture because the SQL Server instance runs in 32-bit mode.

If you are upgrading the components to the same major version of SQL Backup (for example, from version 6.4 to 6.5), SQL Backup remembers the previous settings and takes you straight to the Ready to Install page of the wizard (step 7) when you click **Next**.

3. Cluster installation options

The Setup wizard detects that the SQL Server is part of a cluster, and provides the option to install SQL Backup on all the cluster nodes. If the check box is cleared, the server components are not installed as a clustered resource and you must configure the cluster manually (see Installing manually on Windows Server 2003 and Installing manually on Windows Server 2008). You are recommended to select the check box and let the installer configure the server components as a clustered resource for you.

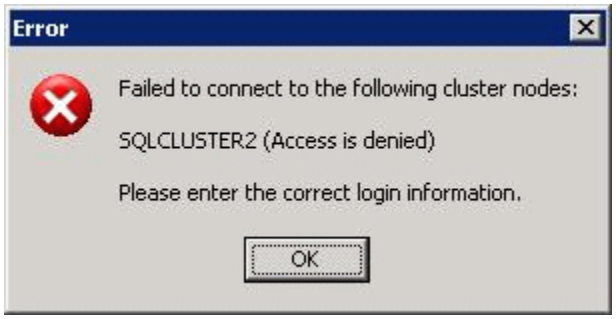
You must then enter the account name and password for a user that is an administrator on all nodes on the cluster (a cluster administrator).



The Setup wizard also provides the option to automatically fail over to another cluster node if the SQL Backup Agent service cannot be started. You are recommended to clear this check box if you do not want all the resources in the group to failover. (Note that this option is not available if you are upgrading the server components.)

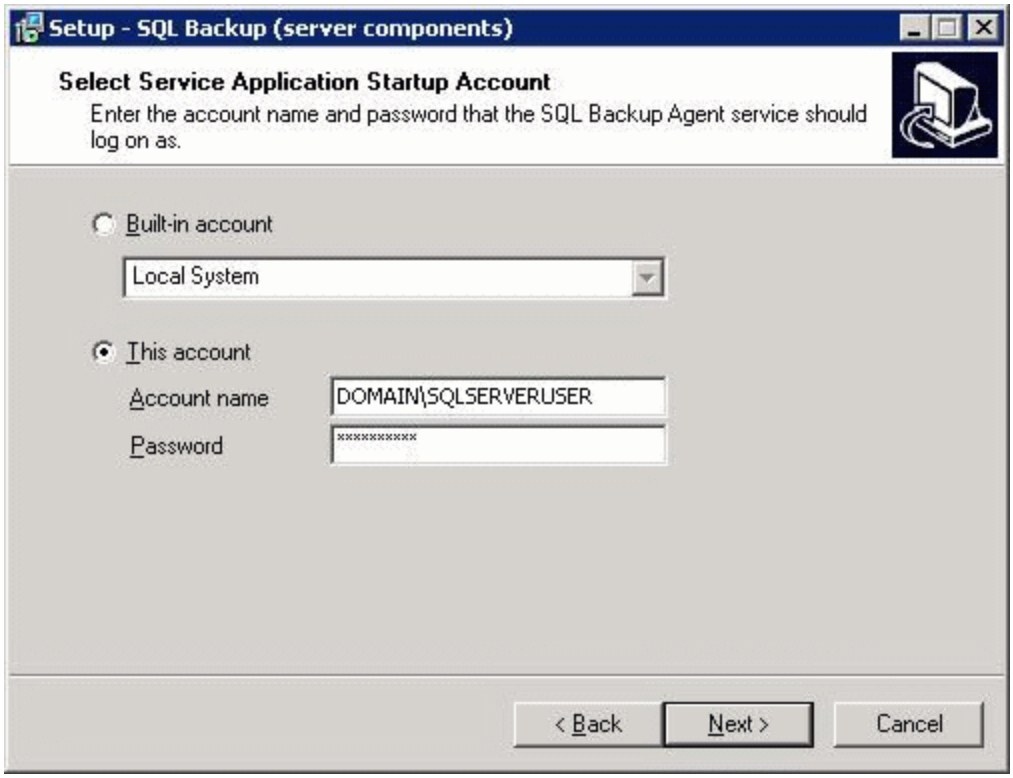
Cluster nodes lists all computers that are participating in the clustered SQL Server instance. In this example, there are two nodes: *SQLCLUSTER1* and *SQLCLUSTER2*.

When you click **Next**, the account name and password are checked on all the cluster nodes. If any of the nodes cannot be correctly authenticated, an error message is displayed. In the example below, the authentication information was valid for *SQLCLUSTER1* but not for *SQLCLUSTER2*.



4. SQL Backup Agent service application credentials

For any new installation, the installer requires information about which account the SQL Backup Agent service application will run as. The SQL Backup Agent performs SQL Backup backup and restore operations, maintains the SQL Backup activity history, and communicates information to the SQL Backup GUI.

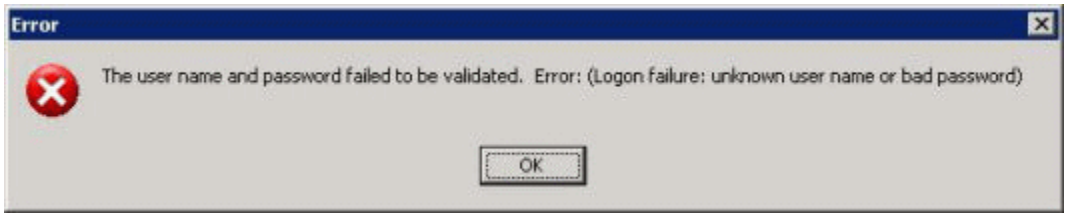


A **Built-in account** can be any account from those listed in the wizard. By default, this is *Local System*, but it can also be *Local Service* or *Networ k Service*.

For a clustered installation, you are recommended to use a domain account rather than a built-in account; select **This account** and enter the credentials. The Setup wizard checks that:

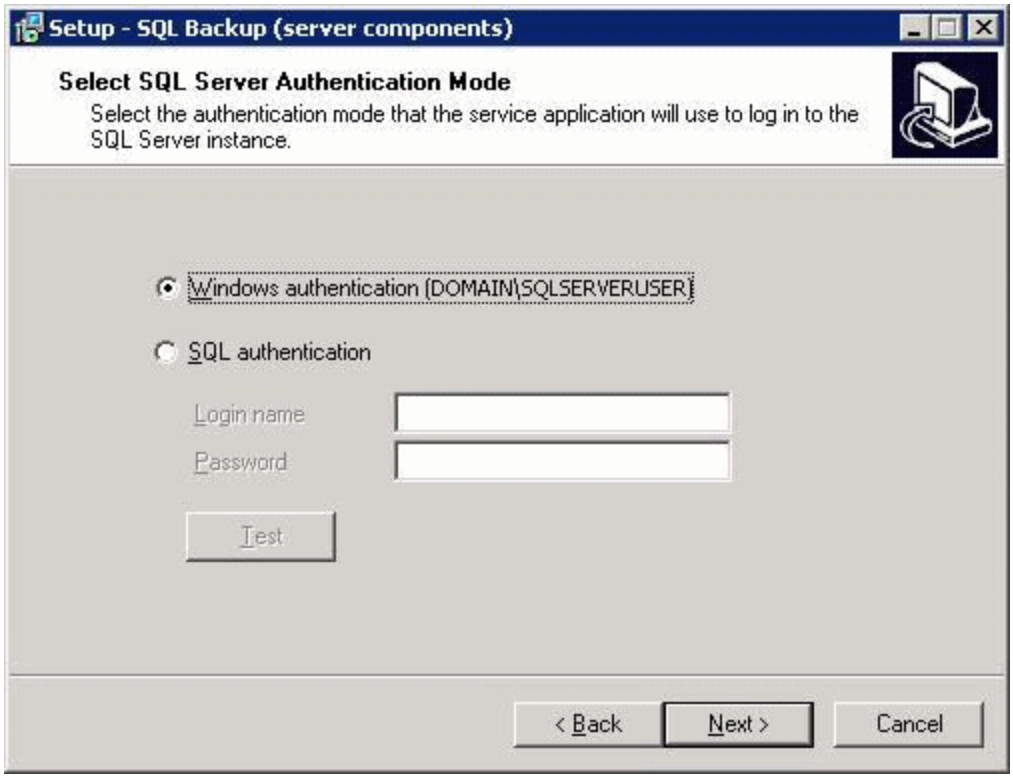
- The account name and password are correct.
- The account has sufficient permissions to run as a service, including the *Log on as a service* permission. If this permission has not been granted, the SQL Backup installer will attempt to grant the permission. For information on the permissions required for the SQL Backup Agent service, see Permissions.

If the credentials are not valid, an error message is displayed:



5. SQL Server authentication mode

For any new installation, the SQL Backup Agent service application needs to know whether to connect using the Windows authentication of the account it runs as, or to use a separate SQL Server authenticated account.

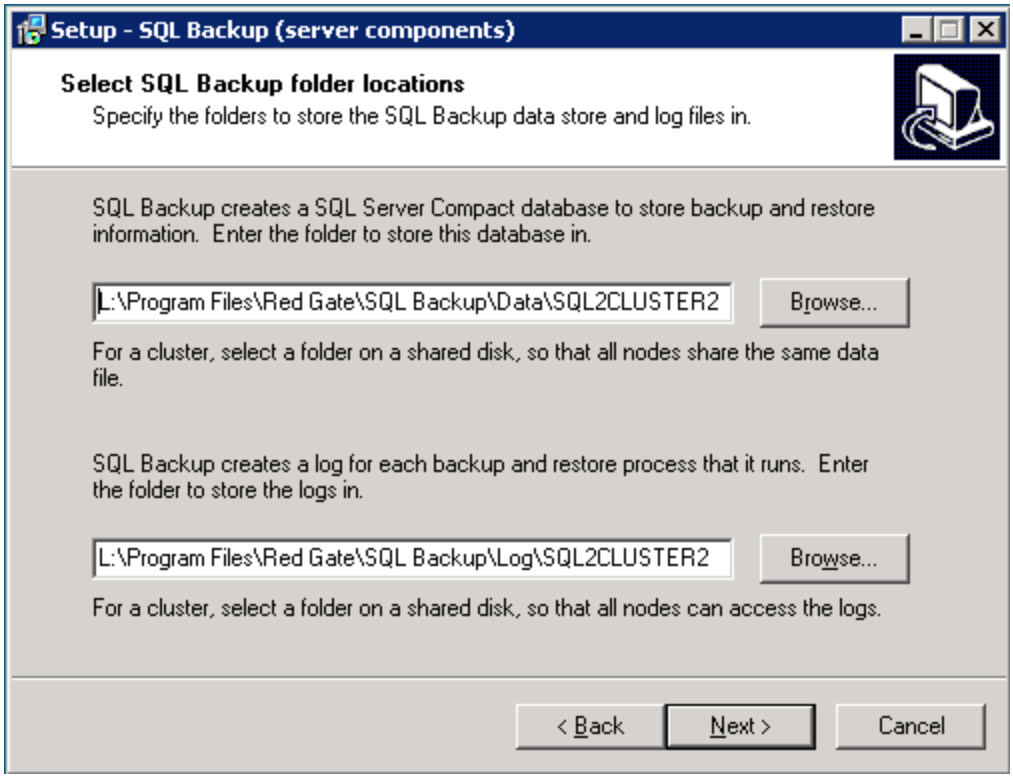


You can check the authentication details that you have entered by clicking **Test**. The Setup wizard tests that the account has sufficient SQL Server permissions (the SQL Backup Agent service application requires *sysadmin* privileges). If it has sufficient permissions, the following message is displayed.

If any other message is displayed, the credentials you have entered are incorrect, or the account does not have *sysadmin* privileges.

6. Data store location

SQL Backup stores data about backup and restore operations in a SQL Server Compact database. This information includes error and warning messages and various statistics such as compression details. SQL Backup also creates a log for each backup or restore process that it runs.

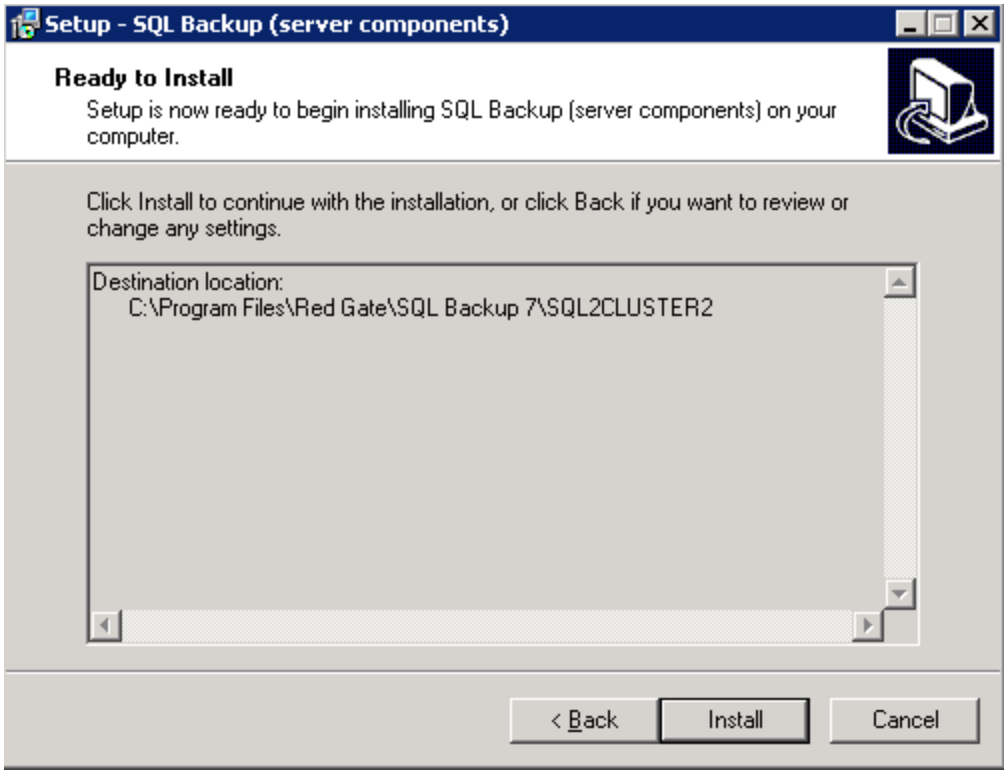


Specify locations on a shared disk that are accessible by all nodes in the cluster to store the database and logs. The location must be specified as a local path (for example, *E:\SQLBackupData\(local)*), and not as a UNC path (for example, *\\server\share\(local)*).

7. Installation location

A summary is displayed, showing the location in which the server components will be installed. By default this is *%ProgramFiles%\Red Gate\SQL Backup 6\<instance name>* or *%ProgramFiles (x86)%\Red Gate\SQL Backup 6\<instance name>*.

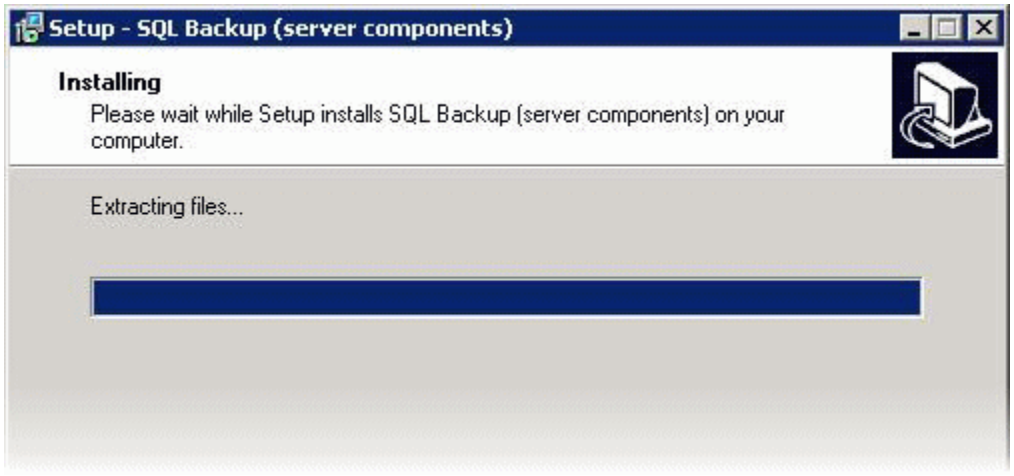
> It is not possible to specify the location in which the server components are installed from the wizard. If you want to specify the location, run *SQBServerSetup.exe* from the command line and use the /PATH parameter. For more information see Installing the server components from the command line.



This is the last point at which you can cancel the installation. When you click **Install**, the Setup wizard will install the SQL Backup server components.

8. Installation progress

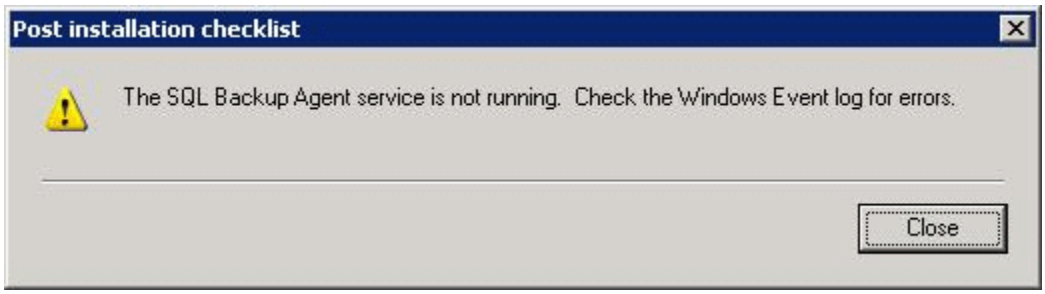
The progress of the installation is displayed.



The installer performs the following tasks:

1. If the SQL Backup Agent service application is currently running, it is taken offline. Any backup or restore operations that are in progress are stopped.
2. The server components are installed to the specified location on disk.
3. Registry settings for the SQL Backup Agent service application are generated. These can be modified later using the SQL Backup GUI.
4. The SQL Backup Agent service application is created with the supplied credentials.
5. For the cluster:
    - The installer is copied to the same location on each of the computers in the cluster in turn. The SQL Backup server components are installed with the same settings as the current computer.
    - The service is registered with the *Cluster Administrator*, and the registry keys are added to the clustered resource so that they will be copied between computers in the cluster and are always available to SQL Backup.
6. Finally, the installer attempts to start the service.

If there are any problems during installation (such as the service failing to start), a warning message is displayed to inform you of the problem. For example:



9. Installation summary

When the installation has completed, the completion page is displayed.

The server components should now be correctly installed on the cluster. If any warnings were displayed during the installation (in stage 8), you can resolve them now by following the instructions in the message.

Once the upgrade process is successfully completed, re-enable the SQL Backup Resource in the SQL Server Service.

Activating on clusters

Once you have installed the server components, you will need to activate SQL Backup on the cluster. For general information about licensing, see Licensing. In addition, note the following points:

- You must have a multi-server license key. You can review your serial numbers at http://www.red-gate.com/myserialnumbers
- You must use the Product Activation wizard. This is installed in your *%Program Files%\Red Gate\SQL Backup 6* folder. Alternatively, you can start the program from the SQL Backup GUI; on the **Tools** menu, select **Utilities** > **Product Activation**.

Installing manually on Windows Server 2003

This page applies to:

- Clusters running on Windows Server 2003 only. If your cluster is running on Windows server 2008, refer to Installing manually on Windows Server 2008.
- SQL Backup versions 5, 6 and 7.

You can install SQL Backup on a Microsoft Cluster Service without using the clustering functionality of the SQL Backup installer. For example, you may want to do this if you have already experienced problems using the installer on a cluster.

This page describes:

- the licensing requirements
- how to clean up following a previous SQL Backup installation
- how to manually install SQL Backup as a clustered resource
- how you can test the installation

Licensing

You must have a SQL Backup license for each server on which you want to install the SQL Backup server components. This means that each node (physical computer) must have its own license.

For example, a two node active-active cluster must have two licenses, and a two node active-passive cluster must also have two licenses.

Cleaning up an existing installation

If you have already attempted to install SQL Backup on the cluster but the installation did not complete successfully, you must 'clean up' the SQL Server instances.

To do this, perform the following steps on each of the SQL Server instances on which the installation failed:

1. Uninstall the server components using the **Add or Remove Programs** dialog box.
   You will be asked whether you want to remove the SQL Backup file containing the history of the backup and restore activity. If you want to save or archive the data store, ensure you click **No** on this dialog box.
2. Open the Registry Editor (regedit.exe) and navigate to the *SQL Backup* branch in the registry editor:
   For all versions of SQL Backup on 32-bit servers, and for SQL Backup 6.4 and later on 64-bit servers, the branch is in `HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate`
   For SQL Backup 6.3 and earlier on 64-bit servers, the branch is in `HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Red Gate`
3. In the next step you will delete the contents of the *SQL Backup* branch, so if you want to copy your settings back later you should export the branch; select the *SQL Backup* branch and on the **File** menu, click **Export**.
4. Delete the following subkeys from the *SQL Backup* branch:
   `BackupSettings\<instance name>`
   `BackupSettingsGlobal\<instance name>`
   `InstalledInstances\<instance name>`

When you have performed these steps on the SQL Server instances as required, you can start the manual installation.

Installing manually

> Although they are similar, this manual installation process (for SQL Backup versions 5, 6 and 7) has critical differences from version 4. If you do not observe these differences, the installation will not be fully cluster-aware.

To install the SQL Backup server components, perform the following steps **on each SQL Server instance in the cluster:**

1. To start the SQL Backup (server components) Setup Wizard, run *SQBServerSetup.exe* on the cluster node.
   a. On step 2 of the wizard, the clustered SQL Server instances will be shown in the list of available instances in the wizard. Select a clustered instance and click **Next**.
   b. On step 3, ensure you *clear* the option to **Install SQL Backup on all cluster nodes**.
   c. On step 4, specify a domain account with sufficient privileges to run as a service, including the Log on as a service permission, for the SQL Backup Agent service to run as.
   d. On step 5, choose the authentication mode for the SQL Backup Agent service to connect to the SQL Server cluster. The account must be a member of the sysadmin fixed server role.
   e. On step 6, specify a location for the SQL Backup data store which is a shared drive, accessible by all nodes in the cluster. The

location must be specified as a local path (for example, *E:\SQLBackupData\(local)*), and not as a UNC path (for example, *\\server\share\(local)*).

f. On step 7, the location for the SQL Backup server component files must be a local drive.

For more information about using *SQBServerSetup.exe* see Installing the server components on a SQL Server cluster.

2. Use the **Services** application to stop the SQL Backup Agent service if it is running. If you are stopping this service on a non-default instance, the service name will be in the form: *SQL Backup Agent-<instance>*.
3. Open the Registry Editor (regedit.exe) and navigate to the *SQL Backup* branch in the registry editor:
   For all versions of SQL Backup on 32-bit servers, and for SQL Backup 6.4 and later on 64-bit servers, the branch is in *HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate*
   For SQL Backup 6.3 and earlier on 64-bit servers, the branch is in *HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Red Gate*
4. Create or modify the following registry entries in the *SQL Backup* branch:
   a. In *BackupSettingsGlobal\<instance>*, create a string value called *DataPath* and set the value to a folder on the shared storage.
   b. In *InstalledInstances\<instance>*, create a dword value called *IsCluster* and set the value to *1*.

When you have completed these steps on all the SQL Server nodes in the cluster, create the clustered resource using the Cluster Administrator tool:

1. Right-click on the SQL Server group, click **New**, and then click **Resource**.
2. On the **New Resource** page, enter the following details:

Name:	SQL Backup Agent - <instance>
Description:	(optional)
Resource type:	Generic Service
Group:	The group to which the SQL Server instance is registered.

3. On the **Possible Owners** page, ensure that the nodes listed match those that the SQL Server instance can run on.
4. On the **Dependencies** page, you are recommended to add the SQL Server instance and the physical disk.
5. On the **Generic Service Parameters** page, ensure the **Service name** is listed as *SQLBackupAgent* for the default instance, or *SQLBackupAgent_<instance>* for a named instance.
   For a named SQL Server instance, enter the **Start parameters as** `-i <instance>`
6. On the **Registry Replication** page, add the following registry keys (do not include the *HKEY_LOCAL_MACHINE* prefix):
   For all versions of SQL Backup on 32-bit servers, and for SQL Backup 6.4 and later on 64-bit servers:
   SOFTWARE\Red Gate\SQL Backup\BackupSettings\<instance>
   SOFTWARE\Red Gate\SQL Backup\BackupSettingsGlobal\<instance>
   For SQL Backup 6.3 and earlier on 64-bit servers:
   SOFTWARE\Wow6432Node\Red Gate\SQL Backup\BackupSettings\<instance>
   SOFTWARE\Wow6432Node\Red Gate\SQL Backup\BackupSettingsGlobal\<instance>
7. Bring the Clustered Resource online.

You have now completed the manual installation.

Testing the installation

To ensure that SQL Backup is clustered correctly, you can perform the following test. This procedure includes at least one failover of the resource group.

1. Open the SQL Backup graphical user interface and use the Back Up wizard to perform a backup, ensuring that you save the backup settings to a template by selecting the **Save as template** check box on Step 6 of the wizard.
   The template will be included in the registry under *BackupSettings\<instance>*
2. In Microsoft SQL Server Management Studio, list the backup that you have just performed by typing the following query:

```
master..sqbdata 'SELECT * FROM backuphistory'
```

3. Fail over to the second node.
4. Use the Back Up wizard to perform a second backup using the template you created.
   If the registry keys have been correctly replicated, the template will be displayed in the list of templates in Step 1 of the wizard.
5. In SQL Server Management Studio, list the backups you have performed by typing the following query:

```
master..sqbdata 'SELECT * FROM backuphistory'
```

   If the data stored has been correctly replicated, this will list both backups.
6. If required, fail over to the first node.

Installing manually on Windows Server 2008

This page applies to:

- Clusters running on Windows Server 2008 only. If your cluster is running on Windows server 2003, refer to Installing manually on Windows Server 2003 instead.
- SQL Backup versions 5, 6 and 7

You can install SQL Backup on a Microsoft Cluster Service without using the clustering functionality of the SQL Backup installer. For example, you may want to do this if you have already experienced problems using the installer on a cluster.

This page describes:

- the licensing requirements
- how to clean up following a previous SQL Backup installation
- how to manually install SQL Backup as a clustered resource
- how you can test the installation

Licensing

You must have a SQL Backup license for each server on which you want to install the SQL Backup server components. This means that each node (physical computer) must have its own license.

For example, a two node active-active cluster must have two licenses, and a two node active-passive cluster must also have two licenses.

Cleaning up an existing installation

If you have already attempted to install SQL Backup on the cluster but the installation did not complete successfully, you must 'clean up' the SQL Server instances.

To do this, perform the following steps on each of the SQL Server instances on which the installation failed:

1. Uninstall the server components using the **Add or Remove Programs** dialog box.
   You will be asked whether you want to remove the SQL Backup file containing the history of the backup and restore activity. If you want to save or archive the data store, ensure you click **No** on this dialog box.
2. Open the Registry Editor (regedit.exe) and navigate to the *SQL Backup* branch in the registry editor:
   For all versions of SQL Backup on 32-bit servers, and for SQL Backup 6.4 and later on 64-bit servers, the branch is in *HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate*
   For SQL Backup 6.3 and earlier on 64-bit servers, the branch is in *HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Red Gate*
3. In the next step you will delete the contents of the *SQL Backup* branch, so if you want to copy your settings back later you should export the branch; select the *SQL Backup* branch and on the **File** menu, click **Export**.
4. Delete the following subkeys from the *SQL Backup* branch:
   BackupSettings\<instance name>
   BackupSettingsGlobal\<instance name>
   InstalledInstances\<instance name>

When you have performed these steps on the SQL Server instances as required, you can start the manual installation.

Installing manually

> Although they are similar, this manual installation process (for SQL Backup versions 5, 6 and 7) has critical differences from version 4. If you do not observe these differences, the installation will not be fully cluster-aware.

To install the SQL Backup server components, perform the following steps on each SQL Server instance in the cluster.

1. To start the SQL Backup (server components) Setup Wizard, run *SQBServerSetup.exe* on the cluster node.
   a. On step 2 of the wizard, the clustered SQL Server instances will be shown in the list of available instances in the wizard. Select a clustered instance and click **Next**.
   b. On step 3, ensure you *clear* the option to **Install SQL Backup on all cluster nodes**.
   c. On step 4, specify a domain account with sufficient privileges to run as a service, including the *Log on as a service* permission, for the SQL Backup Agent service to run as.
   d. On step 5, choose the authentication mode for the SQL Backup Agent service to connect to the SQL Server cluster. The account must be a member of the *sysadmin* fixed server role.
   e. On step 6, specify a location for the SQL Backup data store which is a shared drive, accessible by all nodes in the cluster. The location must be specified as a local path (for example, *E:\SQLBackupData\(local)*\), and not as a UNC path (for example, *\\server\share\(local)*\).

    f. On step 7, the location for the SQL Backup server component files must be a local drive.

   For more information about using *SQBServerSetup.exe* see Installing the server components on a SQL Server cluster.
2. Use the **Services** application to stop the SQL Backup Agent service if it is running. If you are stopping this service on a non-default instance, the service name will be in the form: *SQL Backup Agent-<instance>*.
3. Open the Registry Editor (regedit.exe) and navigate to the *SQL Backup* branch in the registry editor:
   For all versions of SQL Backup on 32-bit servers, and for SQL Backup 6.4 and later on 64-bit servers, the branch is in *HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate*
   For SQL Backup 6.3 and earlier on 64-bit servers, the branch is in *HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Red Gate*
4. Create or modify the following registry entries in the *SQL Backup* branch:
    a. In *BackupSettingsGlobal\<instance>*, create a string value called *DataPath* and set the value to a folder on the shared storage.
    b. In *InstalledInstances\<instance>*, create a dword value called *IsCluster* and set the value to *1*.

When you have completed these steps on all the SQL Server nodes in the cluster, create the clustered resource using the Failover Cluster Management tool:

1. In the left-hand pane, expand the cluster, then expand **Services and Applications**.
2. Right-click on the **SQL Server** group, click **Add a Resource**, then click **Generic Service**.
   The New Resource Wizard is displayed.
3. Select **SQL Backup Agent-<instance>** from the list then click **Next**.
   The list is not ordered alphabetically, so you may have to scroll through the entire list to find the **SQL Backup Agent-<instance>** resource.
   Note: If the SQL Backup Agent service is already clustered, it will not be available in the list.
4. Click **Next** on the **Confirmation** page of the wizard.
5. Click **Finish** on the **Summary** page of the wizard.
   The new resource has now been created (offline) and is listed under **Other Resources**.
6. Under **Other Resources** select **SQL Backup Agent-<instance>**, then, on the **Actions** list click **Properties** for this resource.
   The **SQL Backup Agent-<instance> Properties** dialog is displayed.
7. On the **General** tab, enter the **Startup parameters** as `-i <instance>`
8. On the **Dependencies** tab, we recommend that you add the SQL Server instance and the physical disk.
9. On the **Policies** tab, select the **If resource fails, attempt restart on current node** option.
10. On the **Advanced Policies** tab, ensure that all the nodes in the **Possible Owners** list are selected.
11. On the **Registry Replication** page, add the following registry keys (do not include the *HKEY_LOCAL_MACHINE* prefix):
    For all versions of SQL Backup on 32-bit servers, and for SQL Backup 6.4 and later on 64-bit servers:
    SOFTWARE\Red Gate\SQL Backup\BackupSettings\<instance>
    SOFTWARE\Red Gate\SQL Backup\BackupSettingsGlobal\<instance>
    For SQL Backup 6.3 and earlier on 64-bit servers:
    SOFTWARE\Wow6432Node\Red Gate\SQL Backup\BackupSettings\<instance>
    SOFTWARE\Wow6432Node\Red Gate\SQL Backup\BackupSettingsGlobal\<instance>
12. Bring the Clustered Resource online.

You have now completed the manual installation.

Testing the installation

To ensure that SQL Backup is clustered correctly, you can perform the following test. This procedure includes at least one failover of the resource group.

1. Open the SQL Backup graphical user interface and use the Back Up wizard to perform a backup, ensuring that you save the backup settings to a template by selecting the **Save as template** check box on Step 6 of the wizard.
   The template will be included in the registry under *BackupSettings\<instance>*
2. In Microsoft SQL Server Management Studio, list the backup that you have just performed by typing the following query:

```
master..sqbdata 'SELECT * FROM backuphistory'
```

3. Fail over to the second node.
4. Use the Back Up wizard to perform a second backup using the template you created.
   If the registry keys have been correctly replicated, the template will be displayed in the list of templates in Step 1 of the wizard.
5. In SQL Server Management Studio, list the backups you have performed by typing the following query:

```
master..sqbdata 'SELECT * FROM backuphistory'
```

   If the data stored has been correctly replicated, this will list both backups.
6. If required, fail over to the first node.

Installing the server components from the command line

To install the SQL Backup server components, run SQBServerSetup.exe. This file is copied to the SQL Backup program folder (*%Program Files%\Red Gate\SQL Backup <version number>* for 32-bit servers, *%Program Files (x86)%\Red Gate\SQL Backup <version number>* for 64-bit servers) when the SQL Backup user interface is installed.

To install the server components unattended, call SQBServerSetup.exe from the command line with `/VERYSILENT` and `/SUPPRESSMESSAGEBOXES`. For example:

```
SQBServerSetup.exe /VERYSILENT /SUPPRESSMSGBOXES
```

The server components will be installed on the local instance using the default settings unless you specify otherwise using the parameters described below.

SQL Backup server components installation parameters

Parameter	Description
`/VERYSILENT`	Performs an installation without launching the SQL Backup (server components) Setup wizard. All the values that would normally be entered in the wizard must be defined at the command line, otherwise the default values will be used. Use in combination with `/SUPPRESSMSGBOXES`.
`/SUPPRESSMESSAGEBOXES`	Suppresses any warning or information boxes that would appear. No information will be returned if an error arises. Use in combination with `/VERYSILENT`.
`/I <instance>`	Specifies the instance to install on. If this parameter is not specified, the default instance is used. Do not specify more than one instance.
`/PATH <folder>`	Installs the server components in the specified directory. When installing on a cluster, the directory can be a shared drive. If this parameter is not specified, the server components are installed in the default location (*%ProgramFiles%\Red Gate\SQL Backup <version number>\<instance name>* on 32-bit machines and *%ProgramFiles(x86)%\Red Gate\SQL Backup <version number>\<instance name>* on 64-bit machines).
`/SVCUSER <name> | <enc_name>`	Specifies the user account used to log on to the SQL Backup Agent service (the startup account). Use either clear-text or an encrypted format. If this parameter is not specified, the account the SQL Server instance is currently running as is used. For details of the permissions required by the SQL Backup Agent service startup account, see Permissions.
`/SVCPW <password> | <enc_password>`	Specifies the password for the user account used to log on to the SQL Backup Agent service (the startup account). Use either clear-text or an encrypted format. Must be used with `/SVCUSER`. The password may only include 7-bit ASCII characters. If the password contains any other characters, the SQL Backup Agent service will not start.
`/SQLUSER <name> | <enc_name>`	Specifies that the SQL Backup Agent service should connect to the SQL Server using the specified SQL Server authenticated account. Use either clear-text or an encrypted format. If this parameter is not specified, Windows authentication is used. For details of the permissions required by the SQL Server authenticated account, see Permissions.

`/SQLPW <password> | <enc_password>`	Specifies the password for the specified SQL Server authenticated account. Use either clear-text or an encrypted format. Must be used with `/SQLUSER`. The password may only include 7-bit ASCII characters. If the password contains any other characters, the installation will fail.
`/DATAPATH <folder>`	Specifies the location in which the SQL Server Compact database is created. The SQL Server Compact database stores SQL Backup backup and restore information. If this parameter is not specified, the Compact database is created in the default location: *%ProgramData%\Red Gate\SQL Backup\Data* (Windows Vista, Windows 2008 and later) or *%ALLUSERSPROFILE%\Application Data\Red Gate\SQL Backup\Data* (Windows XP and Windows 2003).
`/LOGPATH <folder>`	Specifies the location in which the SQL Backup logs are stored. SQL Backup creates a log for each backup or restore process it runs. If this parameter is not specified, the logs are stored in the default location:*%ProgramData%\Red Gate\SQL Backup\Log* (Windows Vista, Windows 2008 and later) or *%ALLUSERSPROFILE%\Application Data\Red Gate\SQL Backup\Log*(Windows XP and Windows 2003).
`/CLUSTERINSTALL`	Installs SQL Backup server components as a clustered service. SQL Server must be installed as a clustered resource on the host machine.
`/CLUSTERUSER <name> | <enc_name>`	Specifies the user name of the user that the installer will run as for clustering registration. Use either clear-text or an encrypted format. Use in combination with`/CLUSTERINSTALL`.
`CLUSTERPW <password> | <enc_password>`	Specifies the password of the user that the installer will run as for clustering registration. Use either clear-text or an encrypted format. Must be used with`/CLUSTERUSER`. The password may only include 7-bit ASCII characters. If the password contains any other characters, the installation will fail.
`/CLUSTERNODE`	Specifies a cluster-based install on a secondary node (rather than the primary node). The server components will not be installed on other nodes in the cluster.
`/LOG`	Writes information from the installation to a log file, which can be found in *%TEMP%* as "Setup Log yyyy-mm-dd #nnn.txt"
`/EXITCODEFILE <file>`	Logs the outcome of the installation to the specified text file using the exit codes listed below.

Example 1

The following command installs SQL Backup server components in the default location on the local instance, specifies the account the SQL Backup Agent service should run as, and specifies that it should use SQL Server authentication to connect to the local instance (the specified accounts should have sysadmin privileges):

```
SQBServerSetup.exe /VERYSILENT /SUPPRESSMSGBOXES /SVCUSER "user name" /SVCPW
"password" /SQLUSER "user name" /SQLPW "password"
```

Note that the `/SQLUSER` and `/SQLPW` parameters are only necessary if you want the SQL Backup Agent service to connect to the SQL Server instance using SQL authentication rather than Windows authentication.

Example 2

The following command installs SQL Backup server components on the SQL2008 server in D:\SQLBackup\ServerComponents, specifies the locations of the data and log stores (D:\SQLBackup\ServerComponents\SQBData and D:\SQLBackup\ServerComponents\SQBLog respectively), and specifies the account the SQL Backup Agent service should run as:

```
SQBServerSetup.exe /verysilent /SUPPRESSMSGBOXES /i (SQL2008) /Path
D:\SQLBackup\ServerComponents /DATAPATH D:\SQLBackup\ServerComponents\SQBData /LOGPATH
D:\SQLBackup\ServerComponents\SQBLog /svcuser "SQLBACKUPSERVICEUSER" /svcpw "password"
```

The SQL Backup Agent service will connect to the SQL Server using the Windows authentication of the startup account, because SQL Server authentication has not been specified.

SQL Backup exit codes

Exit code	Definition
0	Installation was successful.
Pre-installation failures	
5	Another installation is in progress. Try again later.
6000	Current user has insufficient permissions to modify Windows Services.
6010	Windows 2003 Itanium edition requires SP1 to be installed before installing SQL Backup server components.
6020	The user account specified for the SQL Backup Agent service to log on as could not be verified.
6030	The user account specified for the SQL Backup Agent service to log on as was ambiguous. Ensure the account details are fully qualified.
6040	The password for the SQL Backup Agent service user account was invalid.
6100	The user account specified for the SQL Backup Agent service to log on as does not have permission to log on as a service.
6110	Unable to grant 'log on as a service' rights.
6200	SQL authenticated user name or password is invalid.
6210	SQL authenticated account is not a member of the *sysadmin* role.
Post-installation failures	
8192	SQL Backup Agent service executable (*SQBCoreService.exe*) is not installed.
16384	The version of the SQL Backup Agent service is incorrect.
32768	The SQL Backup Agent service could not be registered correctly.
65536	The SQL Backup Agent service was unable to start within 1 minute.
131072	The SQL Backup extended stored procedure dynamic-link library (*xp_sqlbackup.dll*) was not installed correctly.
262144	The version of the SQL Backup extended stored procedure dynamic-link library (*xp_sqlbackup.dll*) is incorrect.
524288	Unable to confirm that the SQL Backup extended stored procedure dynamic-link library (*xp_sqlbackup.dll*) was registered correctly (connection error).
1048576	Some SQL Backup extended stored procedures were not registered correctly.
2097152	The installer could not set the SQL Backup Agent service's instance (-I) flag.

The exit codes can be combined to form a bitmask. For example, a value of 278528 would mean that the versions of the SQL Backup Agent service and the SQL Backup extended stored procedure dynamic-link library are incorrect.

You may find this Simple Talk article about installing SQL Backup on multiple servers using SQL Multiscript useful.

Uninstalling

SQL Backup creates separate entries for the SQL Backup graphical user interface, and the SQL Backup server components in your **Add or Remove Programs** control panel. You can add or remove each of these programs in the usual way.

Removing the server components will also result in the removal of the extended stored procedures, the SQL Backup Agent service, and the SQL Backup binaries from the SQL Server.

You will be prompted to confirm whether you want to remove the SQL Server Compact database file that contains the history of your SQL Backup backup and restore activity. If you intend to reinstall SQL Backup, you are recommended to click **No** to retain the file for future reference; click **Yes** to remove this file only if you are removing SQL Backup from your computer permanently.

Quality improvement program

> The quality improvement program is only available in SQL Backup 6.5 and later.

Redgate products can automatically send anonymous information about the features you use to Redgate. This information helps Redgate improve the reliability and performance of our products and decide which new features to add and bug fixes to prioritize.

When you use a Redgate product for the first time, you are given the option to participate in the quality improvement program. If you choose not to participate, no usage data back to Redgate.

For more information about the types of data collected by the quality improvement program, see Data collected by the Quality Improvement Program.

Changing your participation

If you later decide to change you participation in the quality improvement program, you will need to edit the *SmartAssemblyReportUsage* entry in the Windows registry.

> Serious problems might occur if you modify the registry incorrectly by using Registry Editor or by using another method. These problems might require that you reinstall the operating system. Modify the registry at your own risk.

To change your participation in the quality improvement program:

1. Open the Registry Editor (regedit.exe).
2. In the left-hand pane, navigate to *HKEY_CURRENT_USER\Software\Red Gate\<product name>*.
   In the right-hand pane, the value of the *SmartAssemblyReportUsage* key shows whether you have already accepted (*True*) or declined (*False*) to use the quality improvement program.
3. Double-click the *SmartAssemblyReportUsage* key. The **Edit String** dialog opens.
4. Edit the text in the **Value** data text box:
   a. *True* will cause the product to send quality improvement program data to Redgate.
   b. *False* will prevent the product from sending this data to Redgate.

   To force the product to display the quality improvement program dialog when you start the GUI, delete the *SmartAssemblyReportUsage* registry entry (right-click the entry, then click **Delete**).
5. Close the Registry Editor.

Permissions

This page explains the permissions required to use SQL Backup.

Using SQL Backup from the graphical user interface

Permissions required by the GUI user

The user connecting to the SQL Backup GUI requires:

- Membership of the SQL Server *sysadmin* fixed server role, if connecting to the registered SQL Servers using Windows authentication. For information on how to connect using SQL Server authentication, see Adding SQL Server instances by name (step 4).
- *Execute* permissions on the SQL Backup extended stored procedure, *sqlbackup*.
- For the compression analyzer, *execute* permissions on *sqbtest*, *sqbtestcancel* and *sqbteststatus* extended stored procedures.

Permissions required by the SQL Backup Agent service

The SQL Backup Agent service is a Windows service which SQL Backup uses to perform backup and restore operations through the GUI. You specify the user account used to log on to the SQL Backup Agent service (the startup account) when you install the server components on a SQL Server instance.

The user account used to log on to the SQL Backup Agent service (the startup account) and connect to the SQL Server requires:

- "Log on as a service" rights in order to start the service.
- If using Windows authentication to connect to the SQL Server, membership of the SQL Server *sysadmin* fixed server role. If using SQL Server authentication, the SQL Server authenticated account must be a member of the SQL Server *sysadmin* fixed server role, but the startup account does not need to be. For information on changing the authentication mode the SQL Backup Agent service startup account uses to connect to the SQL Server instance, see Changing the authentication mode below.
- Access to any network locations that will be backed up or copied to, or restored from.
- Access to the following folders:
    - The SQL Backup local data store. By default, this is installed in *%PROGRAMDATA%\Red Gate\SQL Backup\Data* (for Windows Server 2008, Windows Server 2008 R2, Windows Vista and Windows 7) or *%ALLUSERSPROFILE%\Application Data\Red Gate\SQL Backup\Data* (for Windows Server 2003 and Windows XP).
    - The SQL Backup logs folder. By default this is *%PROGRAMDATA%\Red Gate\SQL Backup\Log* (for Windows Server 2008, Windows Server 2008 R2, Windows Vista and Windows 7) or *%ALLUSERSPROFILE%\Application Data\Red Gate\SQL Backup\Log* (for Windows Server 2003 and Windows XP).
    - The SQL Backup backup settings registry folder *HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\SQL Backup\BackupSettings\<instance>*
    - The SQL Backup backup settings global registry folder *HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\SQL Backup\BackupSettingsGlobal\<instance>*
    - The Red Gate licensing registry folder *HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\Licensing\SQL Backup* or *HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\Red Gate\Licensing\SQL Backup* (for 64-bit machines).
    - The Red Gate licenses folder *%PROGRAMDATA%\Red Gate\Licenses* (for Windows Server 2008, Windows Server 2008 R2, Windows Vista and Windows 7) or *%ALLUSERSPROFILE%\Application Data\Red Gate\Licenses* (for Windows Server 2003 and Windows XP).
    - The Microsoft MSSQLServer setup registry folder *HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\MSSQLServer\Setup* (read access only).

If you encounter errors related to permissions and access rights, ensure that the startup account for the SQL Backup Agent service application has been granted the necessary permissions.

Using SQL Backup from the extended stored procedure

Permissions required by the extended stored procedure user

The user running the extended stored procedure requires:

- permission to back up, restore and drop databases
    - to back up databases, the user must be a member of the *db_backupoperator* fixed database role, or you can use the GRANT BACKUP DATABASE command to grant the permission

    - to restore or drop databases, the user must be a member of the *db_owner* fixed database role or the *dbcreator* fixed server role, or you can use the GRANT CREATE DATABASE command to grant the permission
- *execute* permissions on the SQL Backup extended stored procedure, *sqlbackup*
- for the compression analyzer, *execute* permissions on *sqbtest*, *sqbtestcancel* and *sqbteststatus* extended stored procedures

Permissions required by the SQL Backup Agent service

The SQL Backup Agent service is a Windows service which SQL Backup uses to perform backup and restore operations through the extended stored procedure. You specify the user account used to log on to the SQL Backup Agent service (the startup account) when you install the server components on a SQL Server instance.

The user account used to log on to the SQL Backup Agent service (the startup account) and connect to the SQL Server requires:

- "Log on as a service" rights in order to start the service.
- If using Windows authentication to connect to the SQL Server, membership of the SQL Server *sysadmin* fixed server role. If using SQL Server authentication, the SQL Server authenticated account must be a member of the SQL Server *sysadmin* fixed server role, but the startup account does not need to be. For information on changing the authentication mode the SQL Backup Agent service startup account uses to connect to the SQL Server instance, see Changing the authentication mode below.
- Access to any network locations that will be backed up or copied to, or restored from.
- Access to the following folders:
    - The SQL Backup local data store *%PROGRAMDATA%\Red Gate\SQL Backup\Data* (for Windows Server 2008, Windows Server 2008 R2, Windows Vista and Windows 7) or *%ALLUSERSPROFILE%\Application Data\Red Gate\SQL Backup\Data* (for Windows Server 2003 and Windows XP).
    - The SQL Backup logs folder *%PROGRAMDATA%\Red Gate\SQL Backup\Log* (for Windows Server 2008, Windows Server 2008 R2, Windows Vista and Windows 7) or *%ALLUSERSPROFILE%\Application Data\Red Gate\SQL Backup\Log* (for Windows Server 2003 and Windows XP).
    - The Red Gate licensing registry folder *HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\Licensing\SQL Backup* or *HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\Red Gate\Licensing\SQL Backup* (for 64-bit machines).
    - The Red Gate licenses folder *%PROGRAMDATA%\Red Gate\Licenses* (for Windows Server 2008, Windows Server 2008 R2, Windows Vista and Windows 7) or *%ALLUSERSPROFILE%\Application Data\Red Gate\Licenses* (for Windows Server 2003 and Windows XP).
    - The Microsoft MSSQLServer setup registry folder *HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\MSSQLServer\Setup* (read access only).

If you encounter errors related to permissions and access rights, ensure that the startup account for the SQL Backup Agent service application has been granted the necessary permissions.

Using SQL Backup from the command line

The SQL Backup command line program communicates with SQL Server directly; it does not use the SQL Backup Agent service application. To run *SQLBackupC.exe*, the user must have the SQL Server *sysadmin* fixed server role. For more information, see Using the command line.

Changing the SQL Backup Agent service credentials

When you install the SQL Backup server components, you specify:

- the user the SQL Backup Agent service logs on as, and
- the authentication mode the SQL Backup Agent service uses to connect to the SQL Server instance.

You can change the credentials used by the SQL Backup Agent service at any time, as described below.

Changing the SQL Backup Agent service startup account

> You will need to restart the SQL Backup Agent service for the change to take effect.

To change the account the SQL Backup Agent service logs on as, use the Windows Services snap-in:

1. From the Windows Control Panel, select **Administrative Tools** > **Services**, or run *services.msc*.
2. Select the SQL Backup Agent service for the SQL Server instance: *SQL Backup Agent - <instance name>*.
   Note: The SQL Backup Agent service for the local instance is called just *SQL Backup Agent*.
3. Right-click the service and select **Properties**.
4. On the **Log On** tab, specify the account you want the service to log on as. The account must have the permissions listed above.
5. Click **OK** to close.
6. Right-click the service and select **Restart** to restart the service and apply your changes.

Changing the authentication mode for the SQL Backup Agent service

To change the authentication mode the SQL Backup Agent service startup account uses to connect to the SQL Server instance, use the *sqbsetlogin* extended stored procedure:

1. Add the *sqbsetlogin* stored procedure from the SQL Backup extended stored procedure dynamic link library (*xp_sqlbackup.dll*).
2. Provide the user name and password to specify SQL Server authentication.
3. Remove the *sqbsetlogin* extended stored procedure when you have finished using it. (This step is optional but recommended.)

For example:

```
EXECUTE master..sp_addextendedproc sqbsetlogin, 'xp_sqlbackup.dll'

EXECUTE master..sqbsetlogin 'sa', 'sqbpassword'

EXECUTE master..sp_dropextendedproc sqbsetlogin
```

To revert to Windows authentication, call *sqbsetlogin* with blank values:

```
EXECUTE master..sqbsetlogin '', ''
```

> If you are using SQL Server authentication and you change the account password, you must also apply the change to the SQL Backup Agent service, otherwise backups and restores may fail. This can be done using the *sqbsetlogin* stored procedure as above, but specifying the new account credentials on step 2.

Using a different security model

You may want to use a different security model, for example if you want to back up locally but copy the backup to a locked down network share. The following procedure assumes that you are working in a single domain.

1. Create a domain account with minimal permissions. Add the domain account to a security group on the Windows server on which the SQL Server is installed; the security group must have sufficient permissions to run as a service.
2. Create a SQL Server authenticated account that has the ability to back up and restore databases.To do this, add the account to the *sysadmin* or *db_backupoperator* fixed role, or if you are using SQL Server 2005, 2008 or 2012, you can use the GRANT BACKUP command.
3. When you install the SQL Backup server components on the SQL Server:
    - For the SQL Backup Agent service credentials, select **This account** and enter the domain account you created in step 1.
    - For the SQL Server credentials, select **SQL Server authentication**, and specify the credentials for the SQL Server authenticated account you created in step 2.
4. Create the folder on the local server in which you want to create the backups, and a folder on a network share to which you want to copy the backup files.
5. Confirm that the permissions on both folders are set such that the domain user you created in step 1 can access and write to them.

To check that all the accounts have the appropriate permissions, use the Back Up wizard to create a backup in a local folder and copy it to a network share.

Alternatively, run the following query to ensure that the domain account has sufficient permissions on the network share:

```
EXECUTE master..sqbutility 999, 'RWE', '<network location>'
```

If this is successful and the SQL Backup Agent service has read (R), write (W), and execute (E) permissions, the query will return:

```
<SQBUTILITYRESULT>:1:
```

If there is a problem, the query will return a value of *0*, followed by a message, for example:

```
<SQBUTILITYRESULT>:0:Folder does not exist :
<network location>
```

Working with servers on different Windows domains

If you are working with servers which do not participate in the same Windows domain, you can still use SQL Backup to work with them as usual by setting up "matching accounts". This will be necessary if you want to copy backups to a locked down network share on a different Windows domain, or set up log shipping between servers on different domains.

1. Create accounts on each machine with identical user names and passwords.
2. Set the SQL Backup Agent service to log in to the SQL Server using the account created in step 1, using the *sqbsetlogin* extended stored procedure. For more information, see Changing the authentication mode above.
   When log shipping, the SQL Backup Agent on *both* SQL Servers must log in using the matched account.
3. Give the account on the other domain access permissions to the relevant locations.

Licensing

When you download and install SQL Backup, you have a 14-day trial period to evaluate it without purchase. If you need more time to evaluate the product, email dba.info@red-gate.com.

When you buy a license for the product, you will receive an invoice that contains your serial number to activate the product. If you can't find your invoice, you can view your serial numbers at http://www.red-gate.com/myserialnumbers. You will need to enter the email address and password you provided when you bought the product.

The SQL Backup server components are licensed per server. In this context, a server refers to a distinct system running a Windows operating system, whether physical or virtual. If you have a number of SQL Server instances running on the same physical or virtual server, you only need one license to activate the server components on each of the SQL Server instances. If your servers are clustered, you will need a license for each node running SQL Server. You do not require a license to use the SQL Backup graphical user interface.

The status of SQL Backup on the server is displayed in the Registered SQL Servers pane in the graphical user interface. You can activate SQL Backup on a server if one of the following icons is displayed to the right of the SQL Server instance name:

	SQL Backup is activated for a trial period on this server

The SQL Backup trial period has expired. To use SQL Backup with this SQL Server instance, you must activate SQL Backup for the server.

When you activate the product, the licensing and activation program sends an activation request to the Redgate activation server, using checksums of attributes from your computer. The checksums that are 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.

For more information on how to activate SQL Backup on a SQL Server, see Activating.

If you are experiencing problems with activating your products, you will be offered the option to activate manually. For example, you will need to use manual activation if:

- your computer is not connected to the internet
- your network uses a proxy server that interrupts contact between the product and the Redgate activation server
  Activation should work with proxies configured in the Internet Options applet in the Windows Control Panel. However, it will not work when:
  - the proxy settings specify an automatic configuration script
  - the proxy requires authentication, which cannot be transparently handled by Windows
  - the proxy filters out SOAP packet data or XML data
- activation is unsuccessful (see Troubleshooting licensing and activation errors)

If you are reinstalling SQL Backup on the same computer, for example following installation of a new operating system, you can reactivate the products using the same serial number. This does not affect the number of distinct activations for the serial number. For information on moving a serial number to a different computer, see below.

> You cannot automate activation using the command line. However, if you are not able to activate a server using the graphical user interface, you can use *ProductActivation.exe* which is provided with SQL Backup. For example, you may wish to use this program to activate cluster nodes. *ProductActivation.exe* is located in the folder in which you installed SQL Backup: for example, *C:\Program Files\Red Gate\SQL Backup 6*. You can also start the program from the SQL Backup GUI: from the **Tools** menu, select **Utilities > Product Activation**.

Serial numbers for bundles

If you've bought a bundle of products, your serial number activates all the products in the bundle. 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 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, from 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 machine, 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 this menu item is not available, use the deactivation tool.

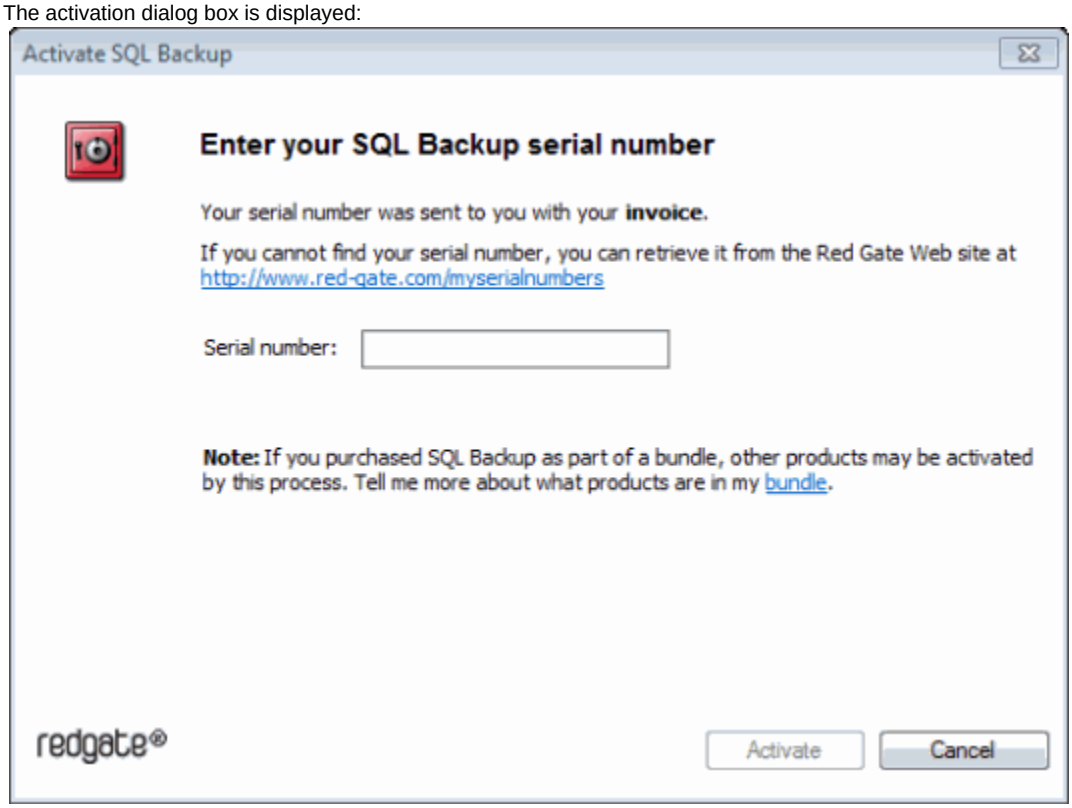
If you cannot deactivate a serial number, use the Request Extra Activations page to request more activations for your serial number. You will need to provide your serial number and the reason for the additional activations.

Activating

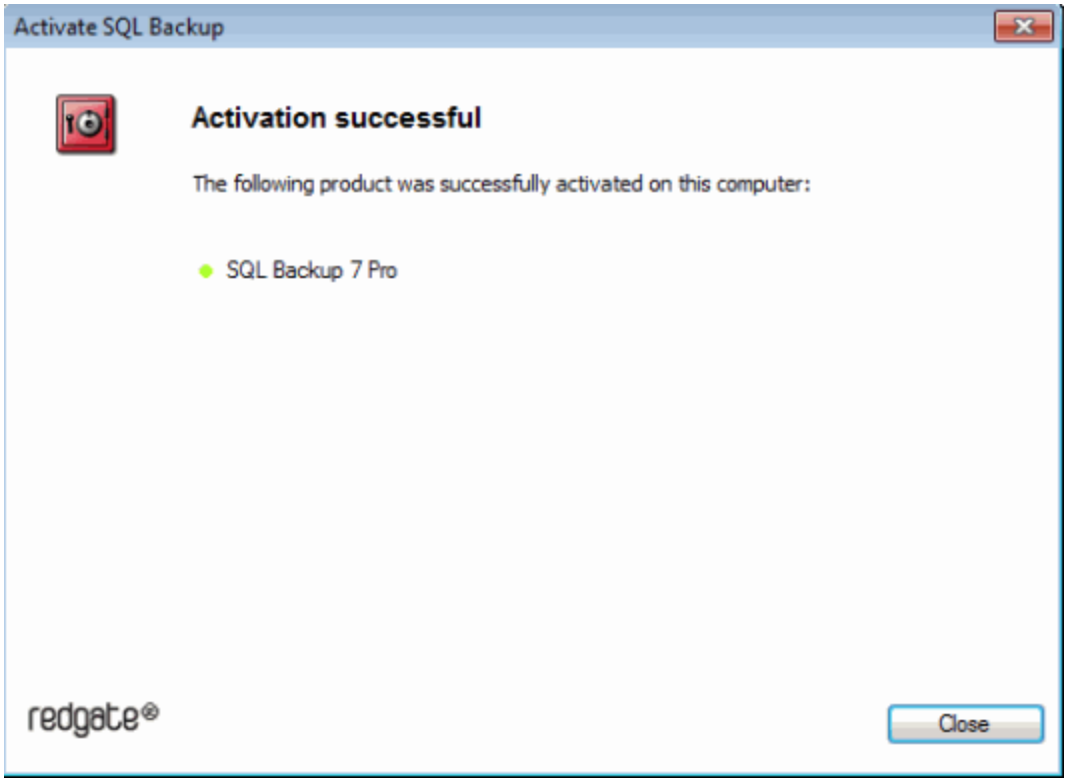
SQL Backup is licensed per server. If you have a number of SQL Server instances registered with SQL Backup that are all on the same server, you need to activate only one of the instances.

To activate SQL Backup on a server:

1. In the Registered SQL Servers pane, click the  or  icon next to the SQL Server instance you want to activate, or right-click and select **Activate License**.
   The activation dialog box is displayed:



2. Enter your serial number and click **Activate**. Your activation request is sent to the Redgate activation server.
   When your activation has been confirmed, the Activation successful page is displayed:



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.

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

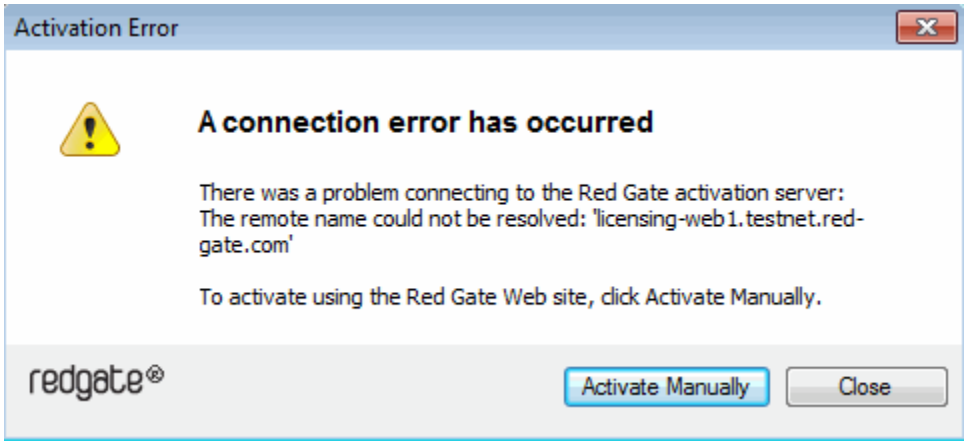
Activating multiple servers

If you have purchased a SQL Backup license for multiple servers, you can activate these servers at the same time using the Multi-Server Activation utility. From the SQL Backup GUI, open the **Tools** menu and select **Utilities** > **Multi Server Activation**.

Manual activation

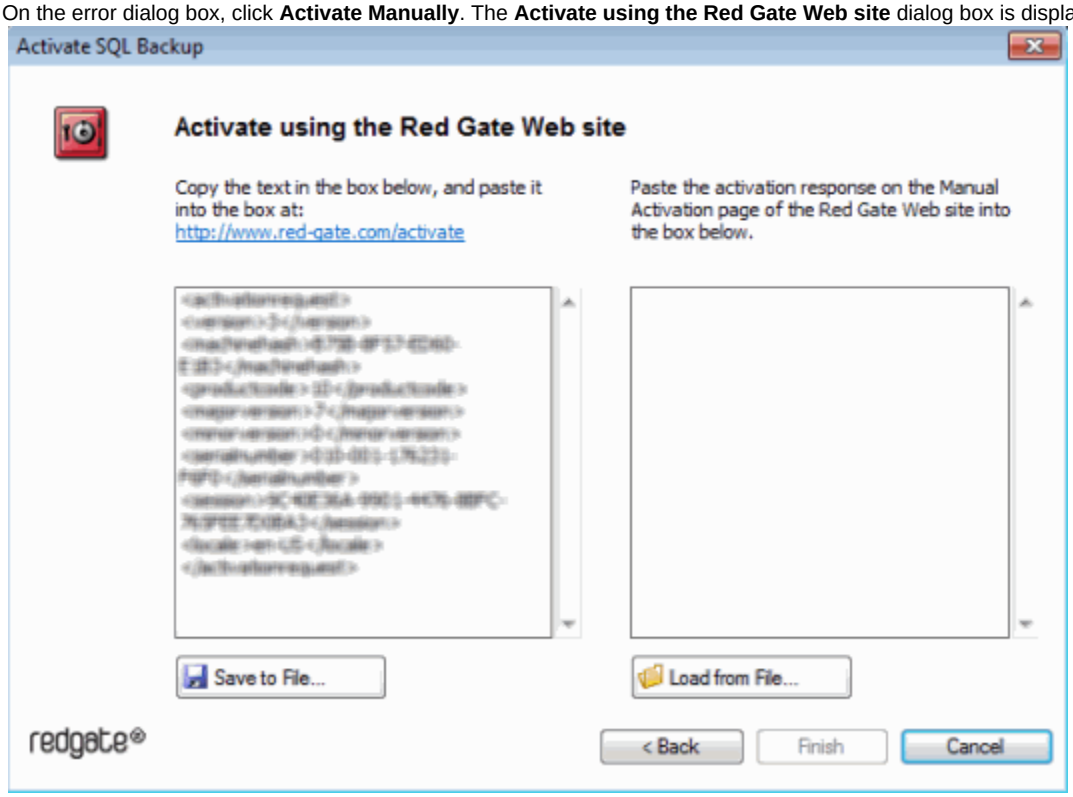
Manual activation enables you to activate products when your computer does not have an internet connection or your internet connection does not allow SOAP requests. You will need access to another computer that has an internet connection.

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

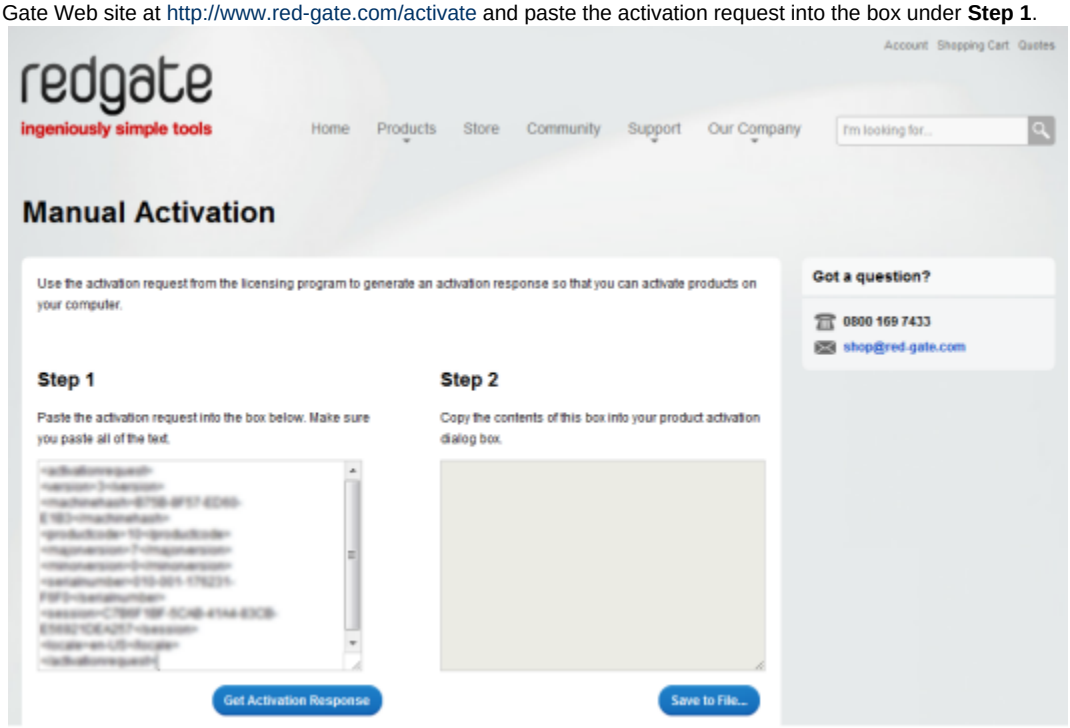


To activate manually:

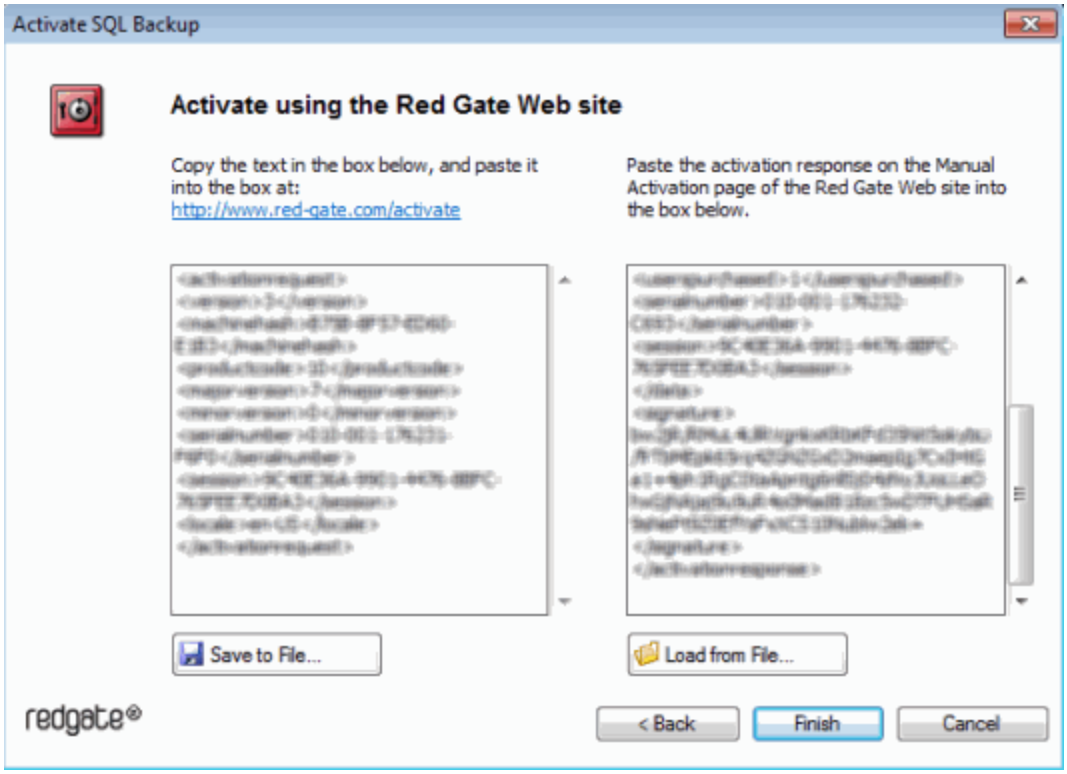
1. On the error dialog box, click **Activate Manually**. The **Activate using the Red Gate Web site** dialog box is displayed:



2. Copy all of the activation request. Alternatively you can save the activation request, for example to a location on your network or to a USB device.
3. Leaving the activation dialog box open, on a computer that has an internet connection, go to the **Manual Activation** page of the Red Gate Web site 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.
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 SQL Backup on this computer.

Re-activating SQL Servers

If you need to re-install SQL Backup server components on the same computer, for example following installation of a new operating system, you can re-activate using the same serial number. If you need to change the computer on which the server components are installed and your current serial number does not work, contact licensing@red-gate.com.

Deactivating

You may wish to deactivate SQL Backup on a server so that you can reuse the serial number on another server. You can do this using the SQL Backup GUI, or the Redgate deactivation tool.

To deactivate a serial number, the server on which the SQL Backup server components are installed must have a connection to the internet. If you cannot deactivate a serial number, you can request additional activations for your serial number. You may need to do this if:

- your computer does not have an internet connection
- your network uses a proxy server that interrupts contact between the product and the Redgate activation server
- your serial numbers are not displayed in the deactivation tool (for example, if the product installation is corrupted)

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

Deactivating SQL Backup using the GUI

In the Registered SQL Servers pane, right-click the SQL Server instance and select **Deactivate License**. If you have more than one SQL Server instance on the same computer, all the instances on that computer will be deactivated.

Note that you can deactivate a SQL Server using the GUI only if the latest server components are installed.

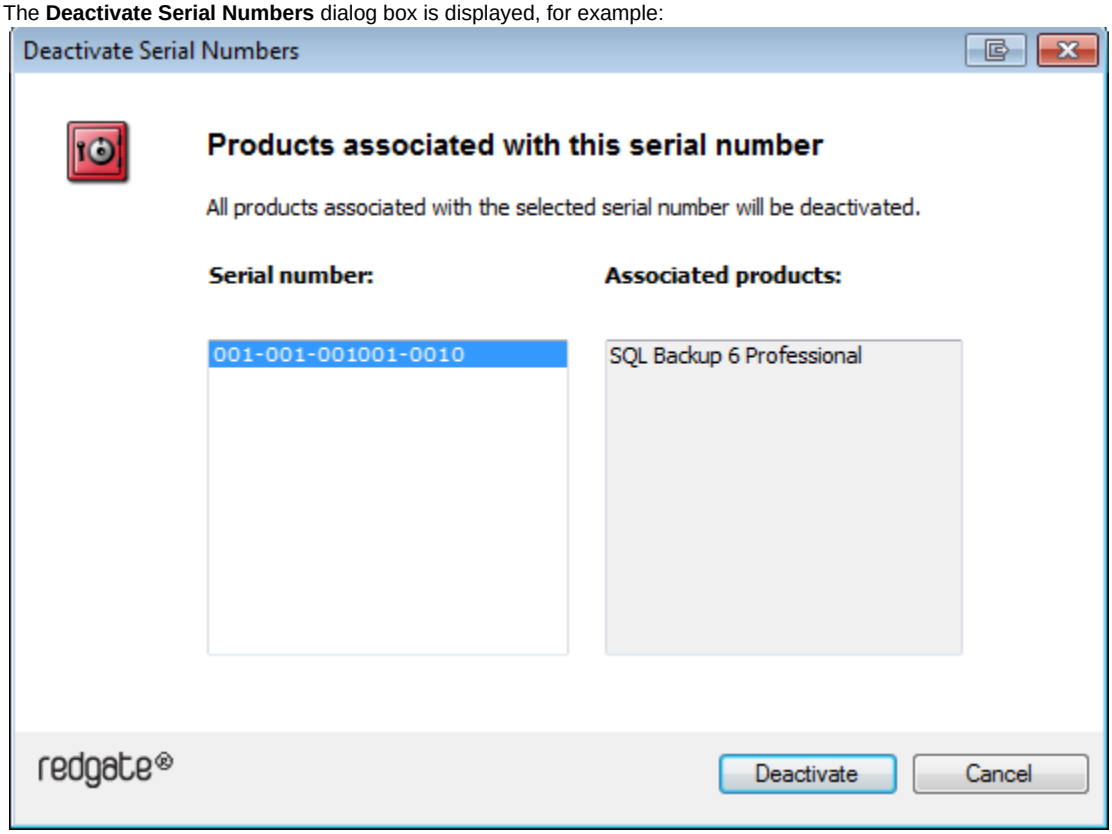
Deactivating SQL Backup using the deactivation tool

If you are unable to deactivate SQL Backup using the GUI (for example, because the SQL Server is using old SQL Backup server components), you can use the Redgate deactivation tool on the server instead.

The deactivation tool enables you to deactivate a serial number on your computer so that you can reuse it on another computer. You can also use the deactivation tool to deactivate serial numbers for products that you have uninstalled.

To deactivate your products:

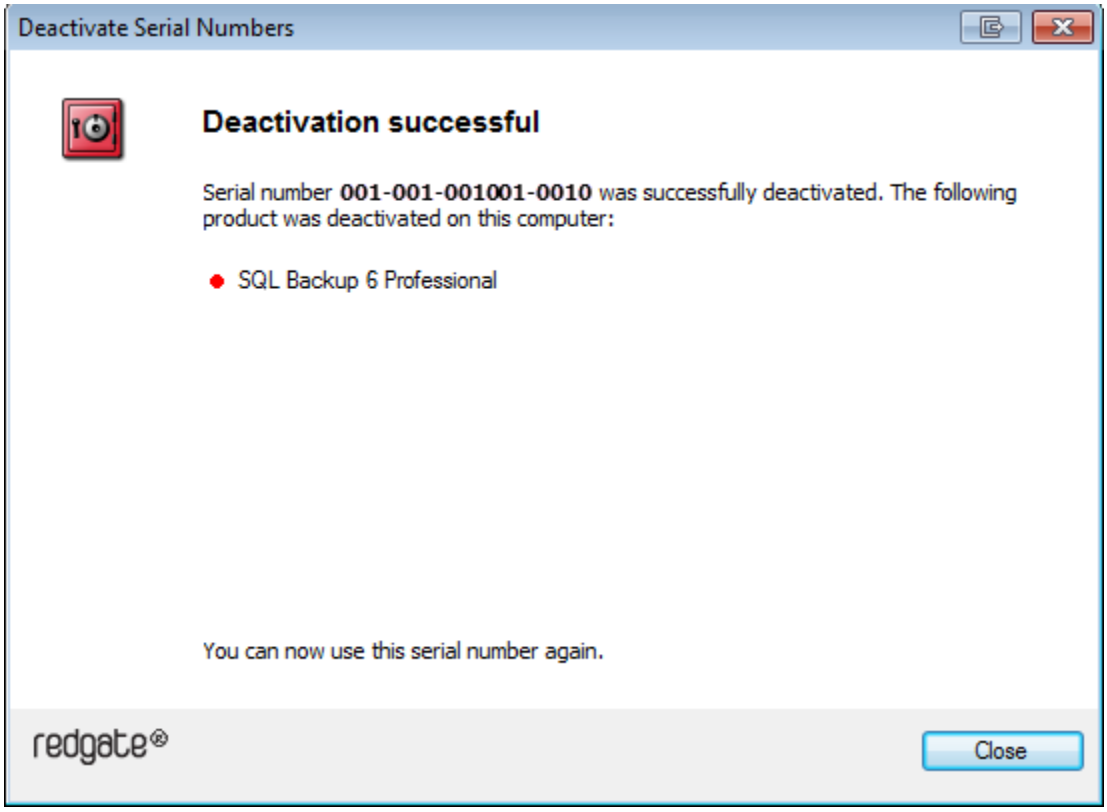
1. Download the tool on the computer where the SQL Server instance is installed and run the executable file.
   The **Deactivate Serial Numbers** dialog box is displayed, for example:



   All the serial numbers for Redgate products that have been activated on your computer are listed. 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 Redgate activation server.
3. When your deactivation has been confirmed, the **Deactivation successful** page is displayed, for example:



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

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 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.

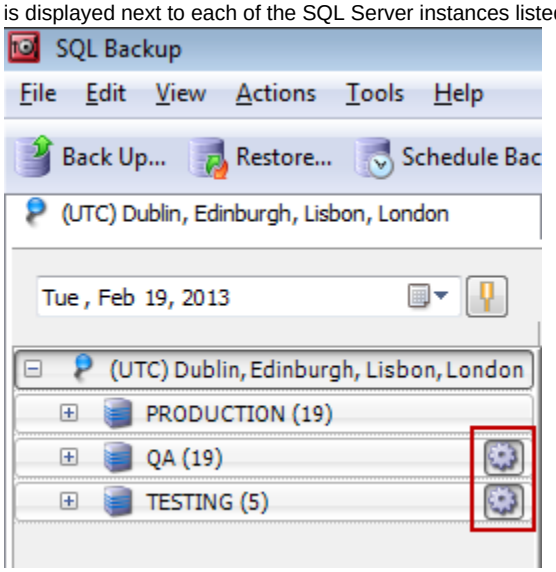
> You can install the latest *major* version of any product (other than SQL Backup Pro) on the same machine as the previous version. For example, you can run version 9 and version 10 in parallel. However, installing a *minor* release will upgrade the existing installation.
>
> To revert to an earlier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

Upgrading the server components

> For information about upgrading the server components on a SQL Server cluster, see Installing the server components on a SQL Server cluster.

The SQL Backup graphical user interface (GUI) and server components are upgraded separately:

- You can upgrade to the latest version of the GUI using **Check for Updates** on the **Help** menu.
- Once the GUI has been upgraded, the upgrade server components icon
  
  is displayed next to each of the SQL Server instances listed in the Registered SQL Servers pane.



  Click the icon or right-click the SQL Server name and select **Install or upgrade server components** to open the Upgrade Server Components wizard. Click **Finish** to confirm that you want to upgrade.
  If your SQL Server security does not allow you to upgrade remotely, you can upgrade the server components manually by copying *SQBServerSetup.exe* to the server and running it. By default *SQBServerSetup.exe* is located in *%ProgramFiles%\Red Gate\SQL Backup 6* on 32-bit servers and *%ProgramFiles(x86)%\Red Gate\SQL Backup 6* on 64-bit servers.

If you have upgraded to a new major version (for example, from version 6 to 7) and had previously activated SQL Backup on the SQL Server with a support and upgrades contract, SQL Backup will remain activated with the same serial number.

Otherwise, a 14 day trial of SQL Backup will begin, indicated by the trial license icon

. When the trial expires, this is indicated by the trial expired icon

; you must purchase a license to continue using SQL Backup on that SQL Server. To purchase a license, email sales@red-gate.com. For information about activating SQL Backup on a SQL Server, see Activating.

> When upgrading to a later version, the server components are installed in the default location and it is not possible to specify an alternative location. To install the server components in a different location, first uninstall the current server components, then install the new server components manually to specify the location.

Upgrading from SQL Backup version 5

If you are upgrading from SQL Backup version 5, you do not need to import SQL Servers that you previously added to the SQL Backup GUI. Your servers will be added automatically to the SQL Backup GUI Registered SQL Servers pane. If you have not yet upgraded to the SQL Backup 6 server components, the upgrade server components icon

is displayed to the right of the SQL Server name in the Registered SQL Servers pane. Upgrade the server components as described above. Note that when you upgrade, the server components remain in the folder they were originally installed in. For example, *%ProgramFiles%\Red Gate\SQL Backup 5\<instance name>*. To install the server components in a different location, first uninstall the current server components, then install the new server components, specifying the path.

The upgrade server components icon

may be shown in various locations within the GUI when you are working with a SQL Server instance that still has version 5 server components installed. This indicates that certain features are unavailable until you upgrade the server components to version 6.

The scripts and jobs you created using version 5 are supported in version 7, and will work in the same way.

> SQL Backup 6 introduces changes to the SQL Backup file format. When you upgrade the SQL Backup server components to the latest version 6 components, SQL Backup creates new backup files in the version 6 format. The backup file extension is unchanged (.sqb).
>
> - SQL Backup 6 is fully compatible with files created by earlier SQL Backup versions. You can work with any backup file version using SQL Backup 6.
> - SQL Backup 5 (version 5.4 and earlier) **cannot** be used to restore a version 6 backup file. To restore a version 6 backup file to a server that has not been upgraded, you must use the SQL Backup File Converter to convert the files first.

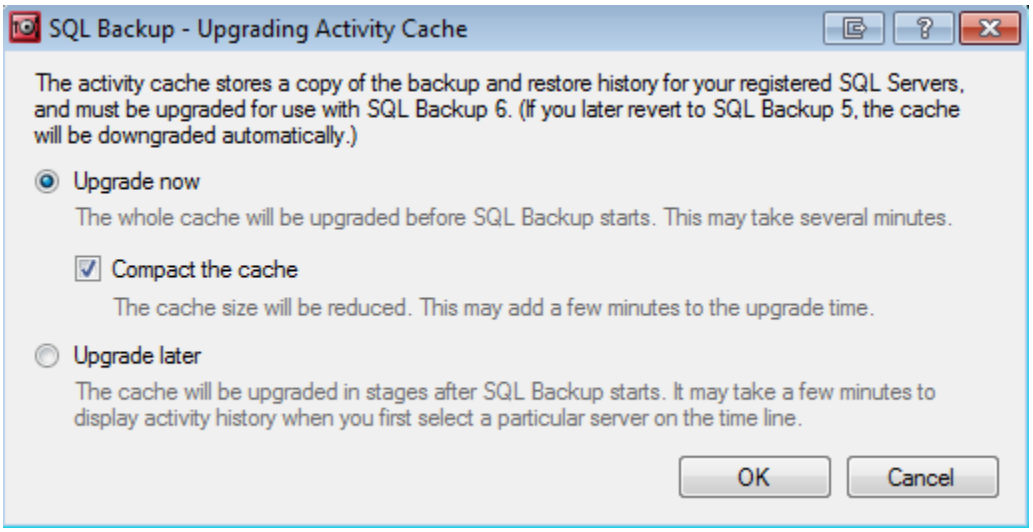
Upgrading the activity cache

The activity cache speeds up the display of backup and restore history in the SQL Backup GUI by storing a copy of the backup and restore history for your registered SQL Servers.

By default, the cache data is located on the same computer as the SQL Backup GUI in:

- *%LOCALAPPDATA%\Red Gate\SQL Backup\Server Data* (on Windows Vista, Windows 2008 and later) or
- *%USERPROFILE%\Local Settings\Application Data\Red Gate\SQL Backup\Server Data* (on Windows XP and Windows 2003)

When you start the GUI for the first time, you may be prompted to upgrade the activity cache. This will only happen if an activity cache from SQL Backup version 5 is found on your computer.



This may take several minutes, but happens only once.

Upgrading from SQL Backup version 4

If you are upgrading the SQL Backup GUI from version 4, you can import all your registered SQL Servers into SQL Backup 6; you do not have to register them all again. For details, see Importing SQL Server instances

When you add or import a SQL Server, if you have not yet installed the SQL Backup 6 server components on it,

is displayed to the right of the SQL Server name in the Registered SQL Servers pane. Click the icon to upgrade the server components.

You must then activate the SQL Server. If you have purchased a support contract, you can use the same license key as you used in version 4 (SQL Backup will remember this for you). Otherwise, you can use a trial version of SQL Backup 6 for 14 days, but then you must purchase a license. For more information, see Licensing.

If you do not upgrade the server components, SQL Backup retrieves the details of activities and jobs on these SQL Servers when you first start SQL Backup. The information is displayed in the Time Line, the Activity History, and the Jobs tab. However, this information cannot be refreshed while SQL Backup is running.

Note the following changes from version 4:

- Any options you set in version 4 are imported into version 6. However, note that backup alerts are no longer available; instead you can now use the Time Line or create a report of backup activity to see overdue backups. For more information about the options available, see File management options.
- The scripts and jobs you created using version 4 are supported in version 6, and will work in the same way.
- If you saved any backup settings in version 4, you can use them in version 6. However, in version 6 they are called *templates*.
- Some features are restricted for activities and jobs from a previous version of SQL Backup. For example, you cannot display the properties of a backup performed by SQL Backup version 4.

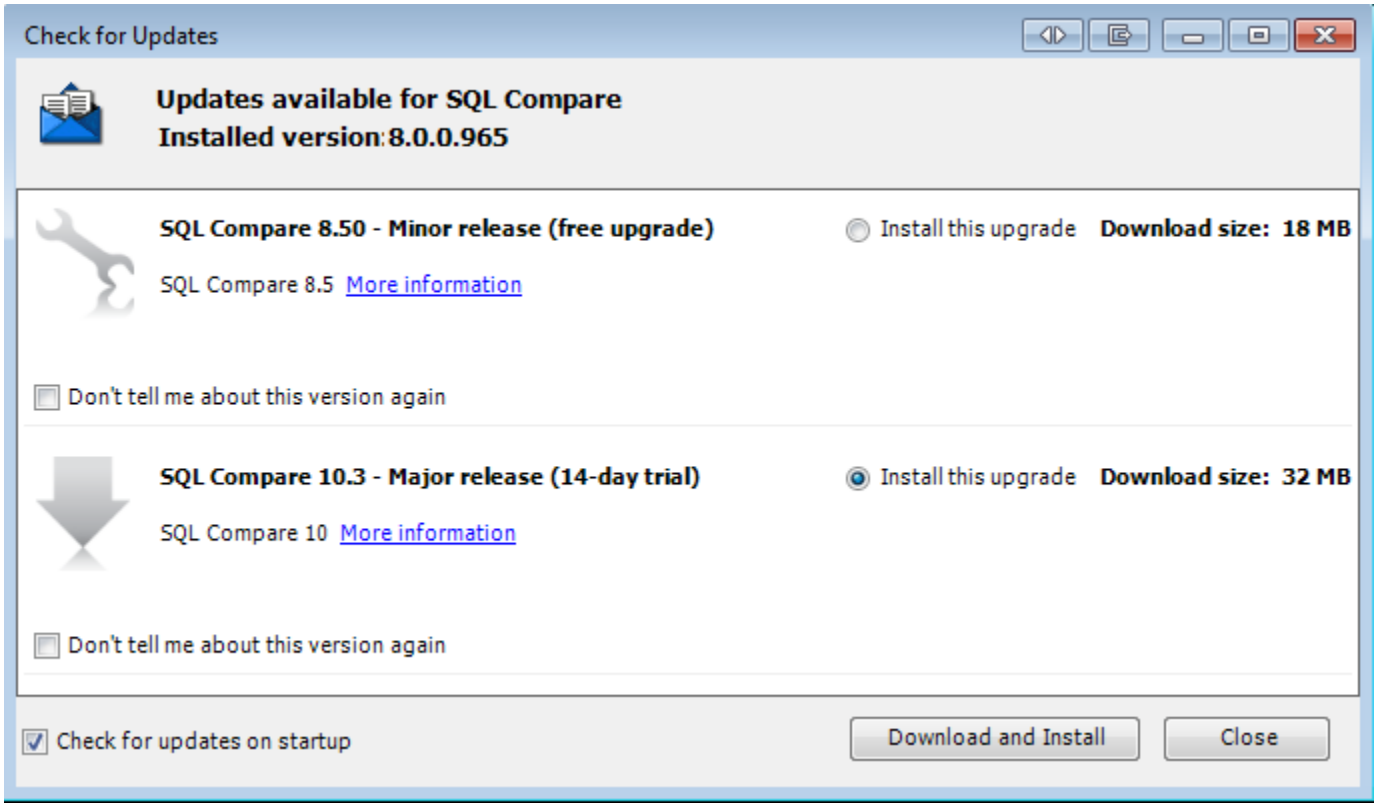
Using Check for Updates

> 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:



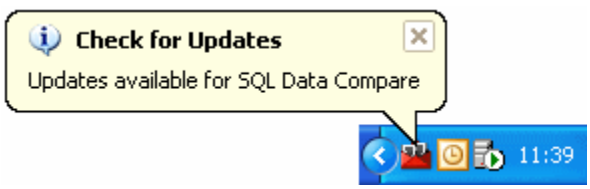
To view the full release details in your default web browser, click **More information**.

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.

Using the GUI to manage backup and restore activity

The SQL Backup graphical user interface (GUI) helps you manage your backups and restores.

To start the graphical user interface, select **SQL Backup 6** from the **Start** menu. The SQL Backup main window is displayed:

The Registered SQL Servers pane

The Registered SQL Servers pane displays a list of the SQL Server instances that are registered with SQL Backup and their databases. Use this area to select the SQL Server instance or database you want to work with.

If you are using the SQL Backup GUI for the first time, you need to add SQL Server instances to the Registered SQL Servers pane, then install the SQL Backup server components on each instance. For more information, see Adding SQL Server instances and Installing the server components on a SQL Server instance.

> If you have upgraded the GUI, you will need to upgrade the SQL Backup server components on each instance. For more information, see Upgrading the server components.

To refresh the connection, right-click the SQL Server instance and select **Refresh Connection** or press F5. This will update both the time line, and the contents of the tabs in the lower pane. While the connection to the instance is being refreshed,

is displayed. You can still interact with the time line while a connection is being refreshed.

The Time Line

The time line provides a graphical overview of all backup and restore activities: completed, in progress and scheduled. You can look at an overview of all activities for a SQL Server instance, or only activities on a specific database. The time line shows where a backup or restore activity has failed; it also shows potential conflicts for scheduled jobs.

Details of jobs and activities

In the lower pane, you can see details of activities and scheduled jobs for the selected SQL Server instance or database:

- The Activity History tab shows the backup and restore operations that have completed or have failed, including native SQL Server backups and restores. For each item, the properties and log provide additional information about any errors or warnings that were raised.
- The In Progress tab shows the SQL Backup activities that are currently in progress.
- The Log Copy Queue tab shows any transaction log backups that are due to be copied to another location (for example, to a network share during log shipping).

- The Jobs tab shows all scheduled SQL Backup jobs.

Getting help

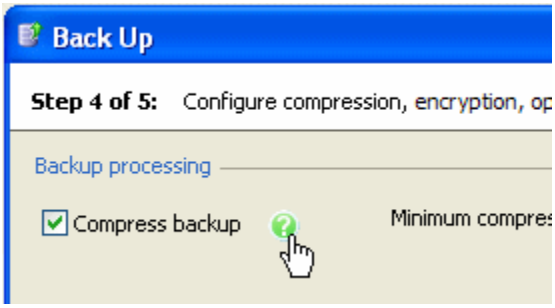
To display the SQL Backup help from the graphical user interface, do one of the following:

- press F1
- on the toolbar, click
  
- on the **Help** menu, click
  
  Contents

You can also display hints on some individual items in the graphical user interface, where you see the

icon. For example, in the Back Up wizard, click

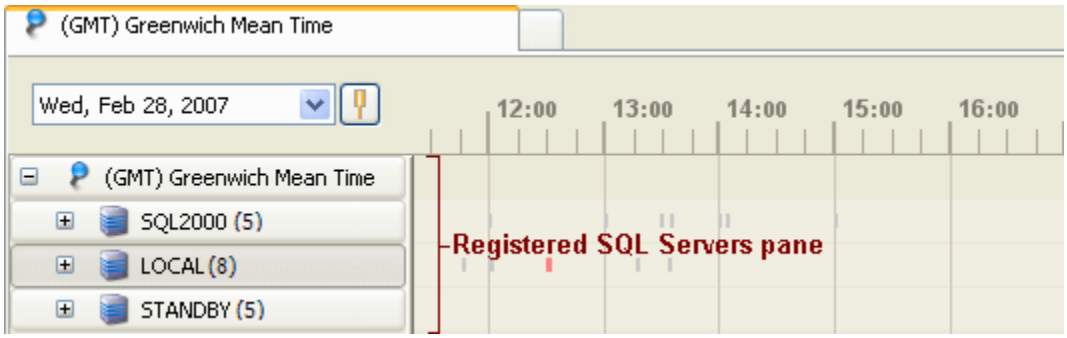
to display a hint for the **Compress backup** option:



To get SQL Server documentation, on the **Help** menu, click **SQL Server Books Online**.

The Registered SQL Servers pane

The Registered SQL Servers pane displays a list of the SQL Server instances you have added to the SQL Backup graphical user interface (GUI), and their databases, in the selected location.



You use this list to select the SQL Server instance or database that you want to back up or restore, or for which you want to view details of jobs and activities in the Time Line and tabs, such as the Activity History.

For more information about adding or importing SQL Server instances so that you can back up or restore databases on them with SQL Backup, see Adding SQL Server instances.

For more information about using time zones and locations, see Time settings and locations.

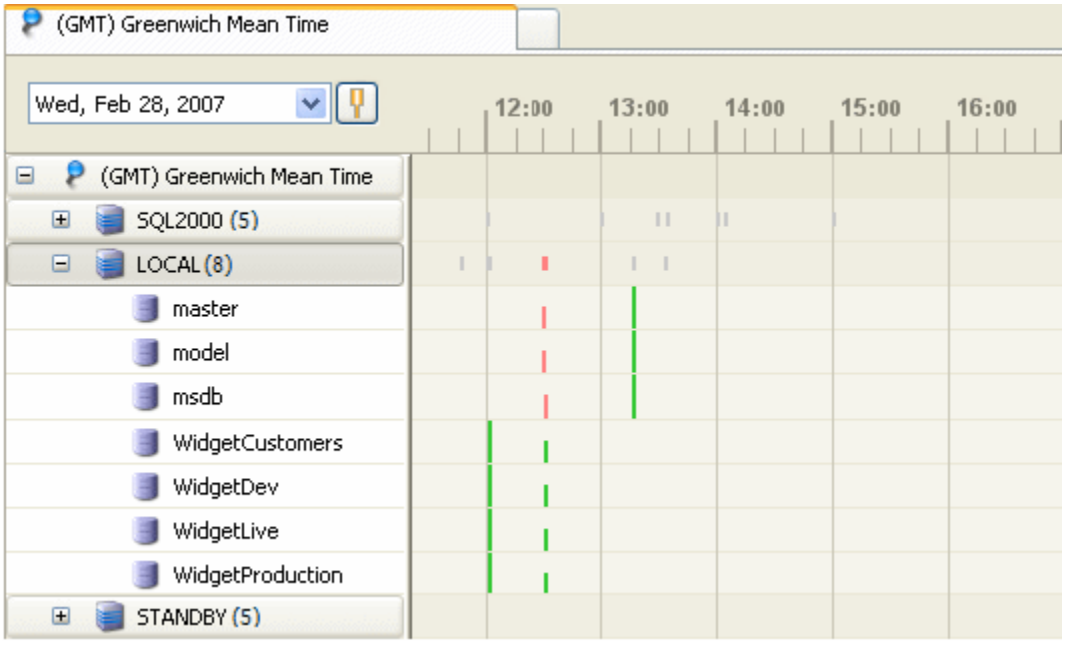
Navigating the Registered SQL Servers pane

The SQL Server instances are organized into groups. You choose the group to which you want an instance to belong when you add it. To change the group to which an instance belongs, in the **Registered SQL Servers** pane, drag the SQL Server instance to the group. For information about creating and managing groups, see Managing SQL Server groups.

The SQL Server groups, instances, and databases are displayed in a tree structure. To expand an item, click the expand button

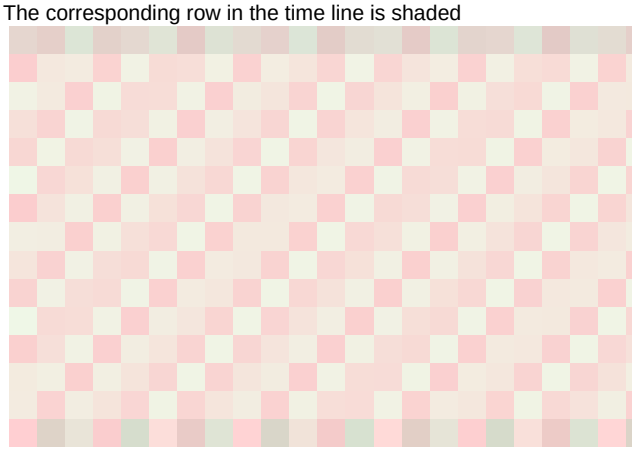
. The Time Line also expands to show activities for the databases. To collapse an item, click the collapse button

.

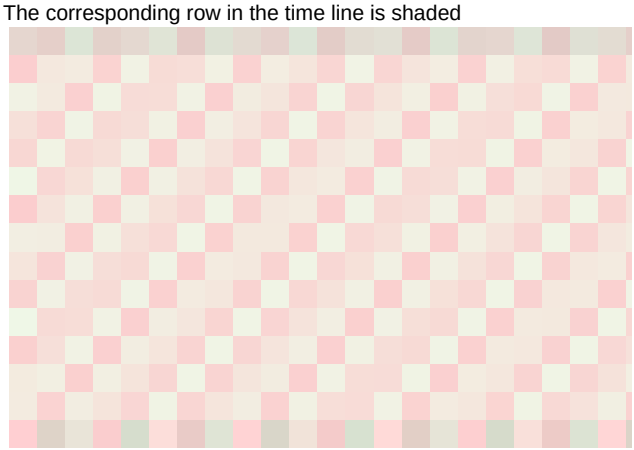


Display

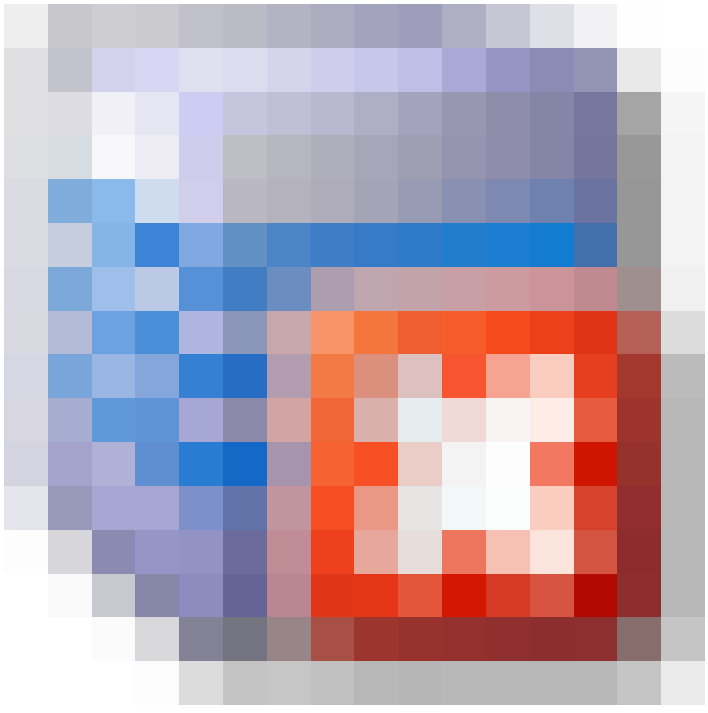
The connection status of each SQL Server instance is displayed to the left of the instance:

	Connected.
	SQL Backup has not attempted to connect to the SQL Server beca are using SQL Server authentication and you chose not to save the password when you entered the authentication details. Double-click icon or right-click and select **Edit**, and then type the password. If yo to save the password in future, select the **Remember Password** ch box. The corresponding row in the time line is shaded  indicating that there is no connection to the server.

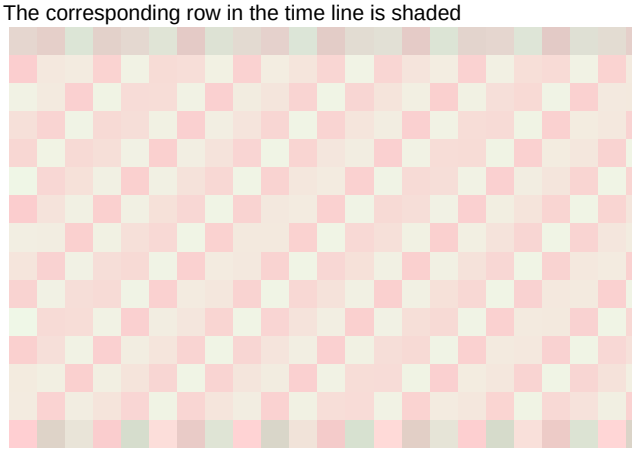
SQL Backup could not connect to the SQL Server because of an authentication error, such as an incorrect password. Right-click and **Edit** to enter the correct details.
The corresponding row in the time line is shaded



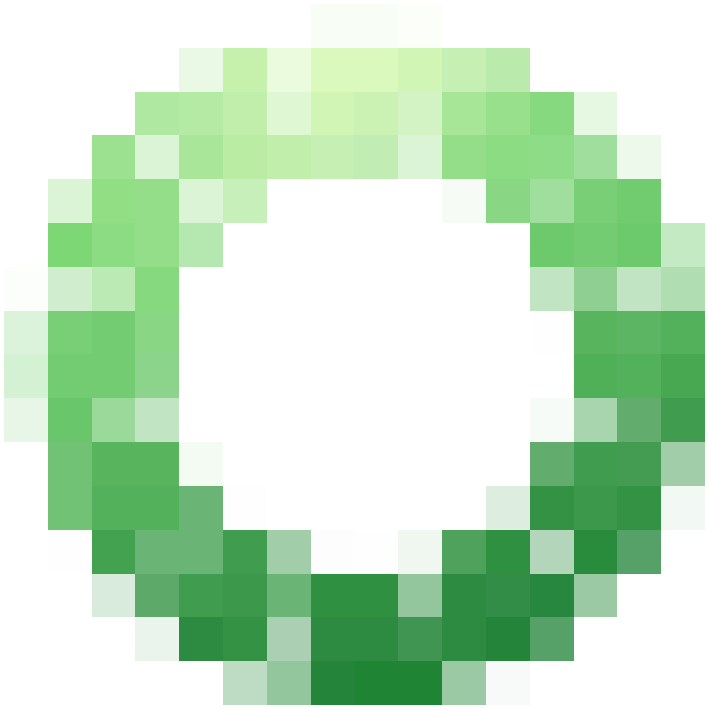
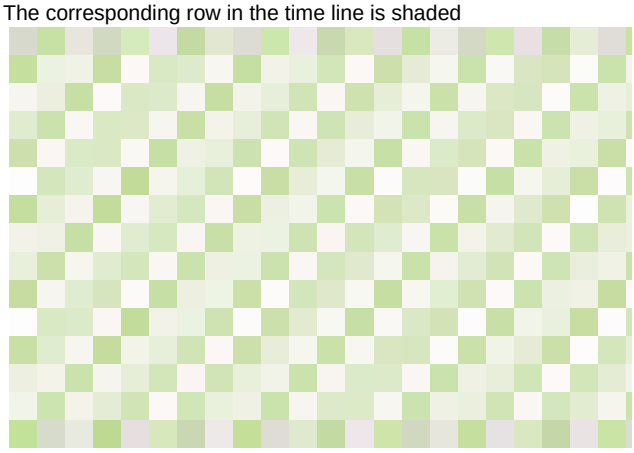
indicating that there is no connection to the server.



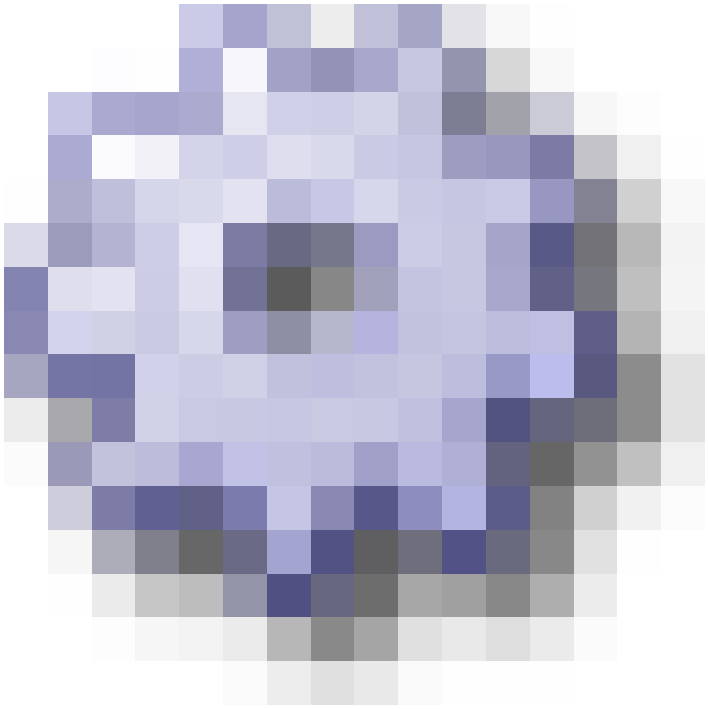
SQL Backup could not connect to the SQL Server. Click the icon to associated error messages. Note that you can specify the time peri which SQL Backup will attempt to connect to each SQL Server in th **ection properties** for that server (right-click the SQL Server and cl then select the **Options** tab). For more information see Connection properties.
The corresponding row in the time line is shaded



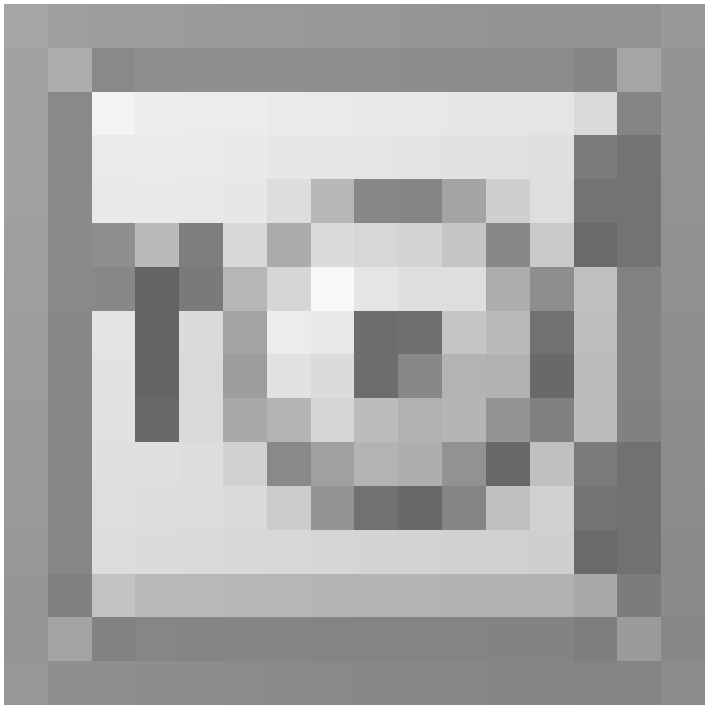
indicating that there is no connection to the server.

	SQL Backup is connecting to the SQL Server instance, or refreshin connection. You can continue to interact with the time line while a connection is being refreshed. The corresponding row in the time line is shaded  while connecting.

You may also see the following icons displayed to the right of the SQL Server instance name:

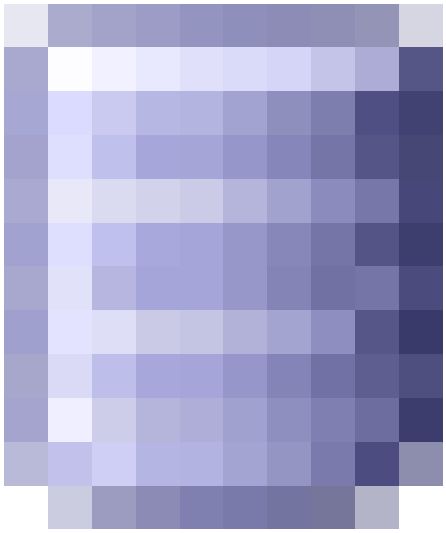
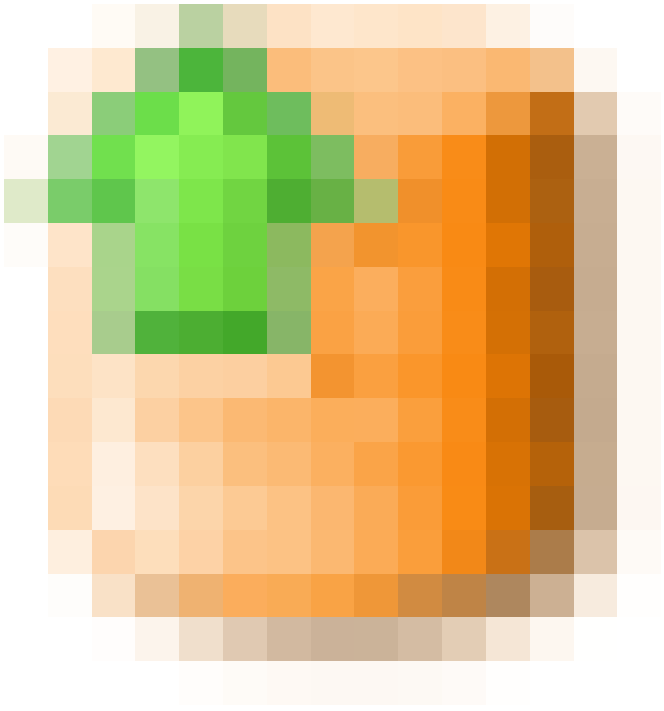
	SQL Backup is installed and activated on this SQL Server, but the version of server components is not up-to-date compared with your version of the GUI. Click the icon to upgrade the server components. For more information, see Upgrading the server components. Note that this icon does not show you when new versions of SQL Backup become available for download; you can use Using Check for Updates for this.

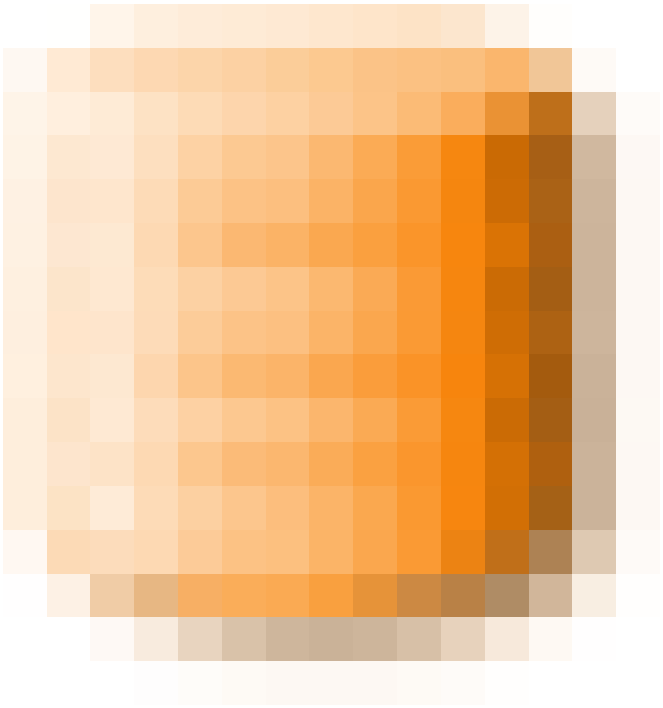
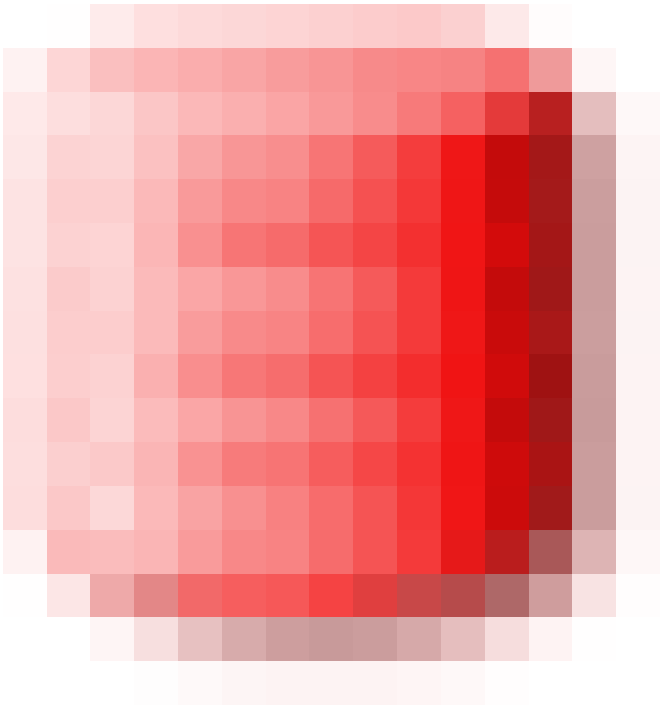
	SQL Backup is activated for a trial period on this SQL Server. Click the icon if you have purchased a license and you would like to activate it. For more information, see Licensing.
	The SQL Backup trial period has expired. To use SQL Backup with this SQL Server instance, you must activate SQL Backup for the SQL Server. Click the icon if you have purchased a license and you would like to activate it. For more information, see Licen sing.

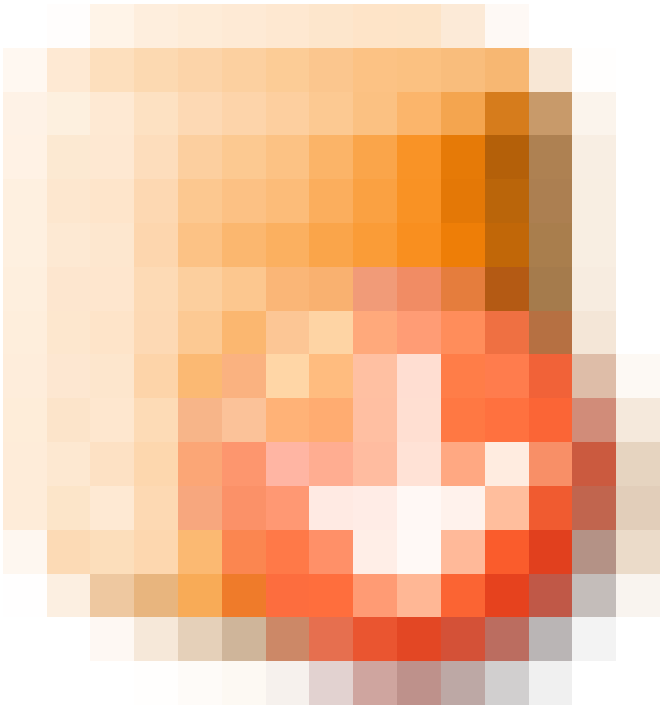
	SQL Backup version 4 or earlier is installed on the SQL Server. You must upgrade the server components to use SQL Backup 6 with this SQL Server instance. Click the icon to upgrade the version of SQL Backup on the SQL Server. For more information, see Upgrading the server components.
	You have not installed the SQL Backup server components on the SQL Server. Click the icon to install the server components. For more information, see Installing the server components on a SQL Server instance.

For each connected SQL Server instance, click the expand button

to see the databases on the instance. The status of each database is displayed:

	Operational. The database is fully usable. Transaction log backups and differential backups cannot be restored to this database.
	The database is being restored, or it has been left in a non-operational state (using `WITH NORECOVERY`) so that differential backups and transaction log backups can be restored to it.

	The database is read-only. If the database is read-only because it has been left in a standby state (using `WITH STANDBY`), transaction log backups and differential backups can be restored to this database.
	The database is unavailable because it has been set to a SUSPECT or EMERGENCY state. Refer to your SQL Server documentation for more information.

	The database is offline or in an unknown state.

Refreshing the data

For the selected SQL Server, and for all databases that are visible in the time line, SQL Backup refreshes the display every minute.

To refresh the display for a SQL Server manually, select the SQL Server in the **Registered SQL Servers** pane and press F5.


is displayed next to the SQL Server instance while SQL Backup refreshes the connection.

> If the SQL Server is busy, there may be a short delay before the display refreshes. You can continue to interact with the time line while a connection is being refreshed.

Interaction

Whenever applicable, the SQL Backup features and displays act on the item you select in the **Registered SQL Servers** pane. For example, if you have a database selected and you click

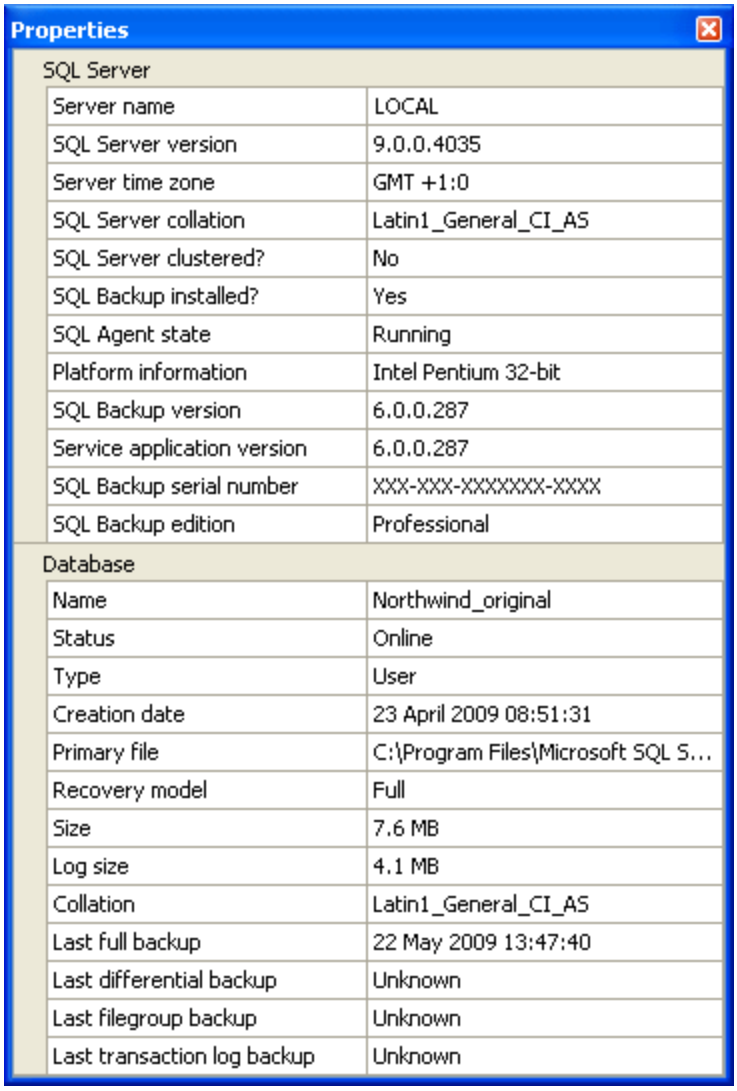

Backup, the Backup wizard pre-selects the database for you (you can change the selection in the wizard).

Similarly, the Activity History, In Progress, Log Copy Queue and Jobs tab are updated with information related to the selected item. For example, if the **Activity History** tab is displayed and you click a SQL Server instance name, all activities that have occurred on that instance are shown in the tab; if you then click a database, only those activities that occurred on the selected database are shown in the tab.

SQL Server and database properties

You can display properties for a SQL Server instance or a database. To do this, right-click the SQL Server instance or database in the Registered SQL Server pane and click **Properties**.

SQL Backup must be connected to the SQL Server for the properties to be displayed. SQL Backup displays details of the selected SQL Server instance, and database, as applicable. For example:



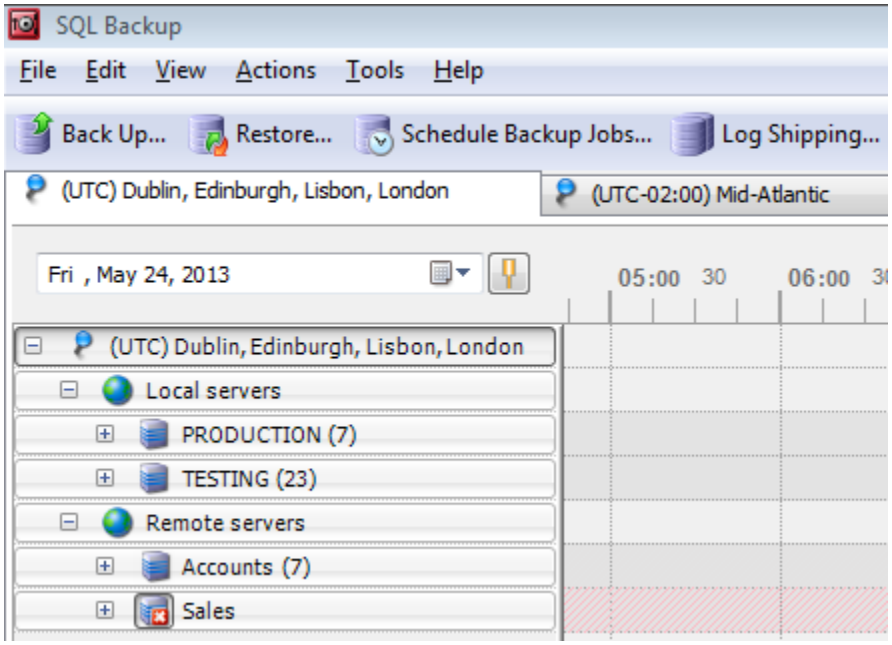
To see the properties of another SQL Server or database, click on it; you do not need to close the **Properties** window first.

You can display the properties of backup and restore activities from the Activity History. For details, see Properties.

Managing SQL Server groups

The SQL Server instances that you have added or imported to the SQL Backup graphical user interface are listed in the Registered SQL Servers pane. By default, the instances are listed at the top level, under the location heading.

You can create your own hierarchy of groups to organize the registered instances. For example:



When you add a SQL Server instance, you can allocate the instance to an existing group. For more information, see Adding SQL Server instances.

For more information about locations, see Time settings and locations.

Creating SQL Server groups

To create a new group or subgroup:

1. On the **File** menu, click  **New Group**.
2. In the **Group name** box, type the name for the new group.
3. Select the **Location** in which you want to create the group.
4. Choose to create a top-level group, or a group within a group.To create a lower-level group, click **Sub-group of**, and then select the group in which you want to create the new sub-group.
5. Click **Add Group**.

Deleting SQL Server groups

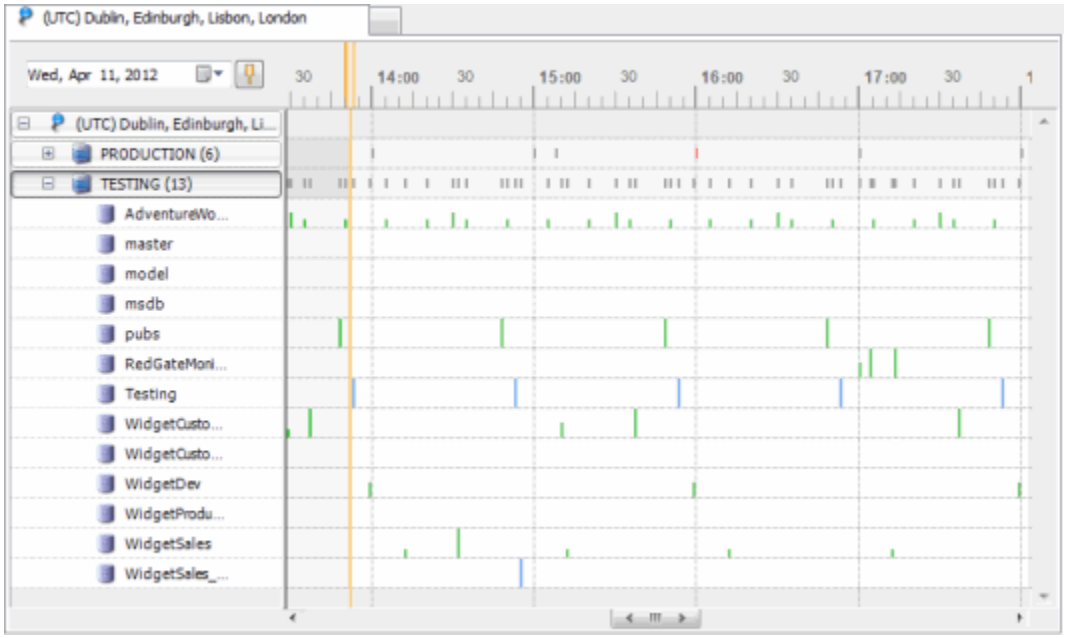
To delete a group or sub-group, do one of the following:

- Right-click the name of the group in the **Registered SQL Servers** pane, click **Delete**, and then click **Delete** on the confirmation dialog box.
- Click the name of the group in the **Registered SQL Servers** pane, in the **Edit** menu, click **Delete**, and then click **Delete** on the confirmation dialog box.

> If you delete a group that contains registered SQL Server instances, all instances in the group are removed. Any SQL Backup jobs on the instances will not be affected.

The Time Line

The time line provides a graphical view of SQL Backup backup and restore activities.

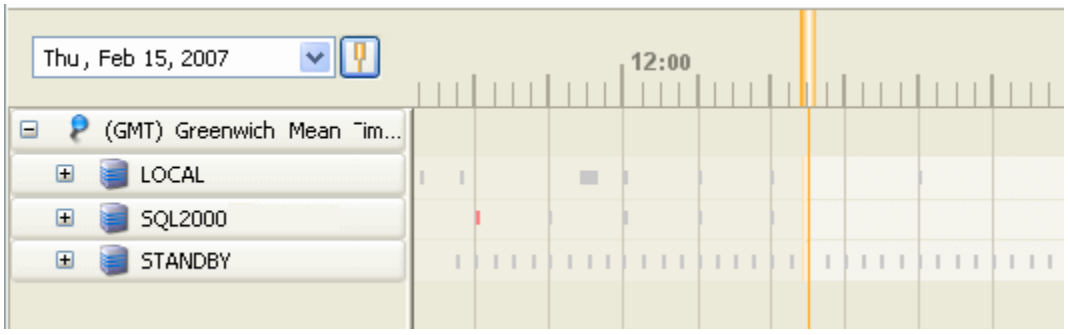


The time line shows activities that have been set up using the SQL Backup graphical user interface (GUI), and, where possible, any other SQL Backup activities, such as those initiated using the command line. However, not all SQL Backup activities set up outside of the GUI can be recognized by SQL Backup.

Object level recovery activities (performed using the SQL Object Level Recovery Pro application) are not recorded on the time line.

Display

For each registered SQL Server instance that is connected, the time line shows a summary of the activities on that SQL Server instance.



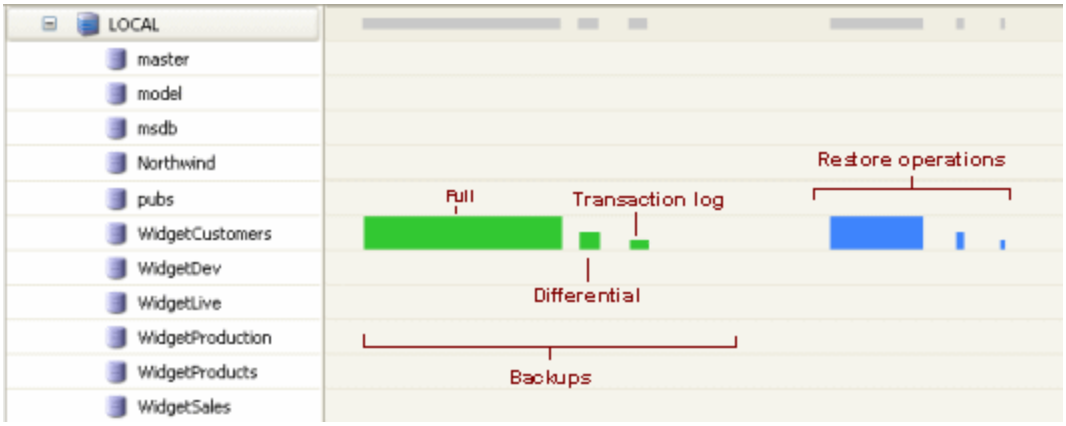
A gray block

indicates a successful activity or a scheduled job; a red block

indicates a failed activity or conflicting backup jobs.

To see the activities for the individual databases, click the expand button

to expand the SQL Server instance in the Registered SQL Servers pane so that the databases are listed. The activities that correspond to the databases are shown.



Activities that have already occurred are shown as solid blocks, as illustrated above. Backup operations are green
; restore operations are blue
. The type of backup is indicated by the height of the block. File or filegroup backups are the same height as differential backups.

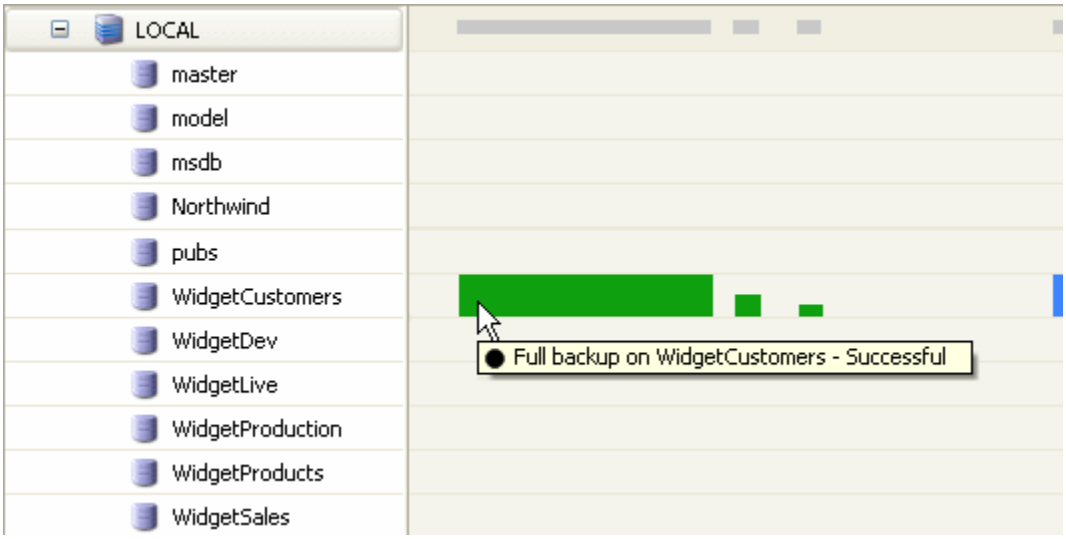
An operation that failed is shown as a red block

Jobs that are in progress are shown as striped blocks, for example:

Backup jobs scheduled for the future are shown as hollow blocks, for example:

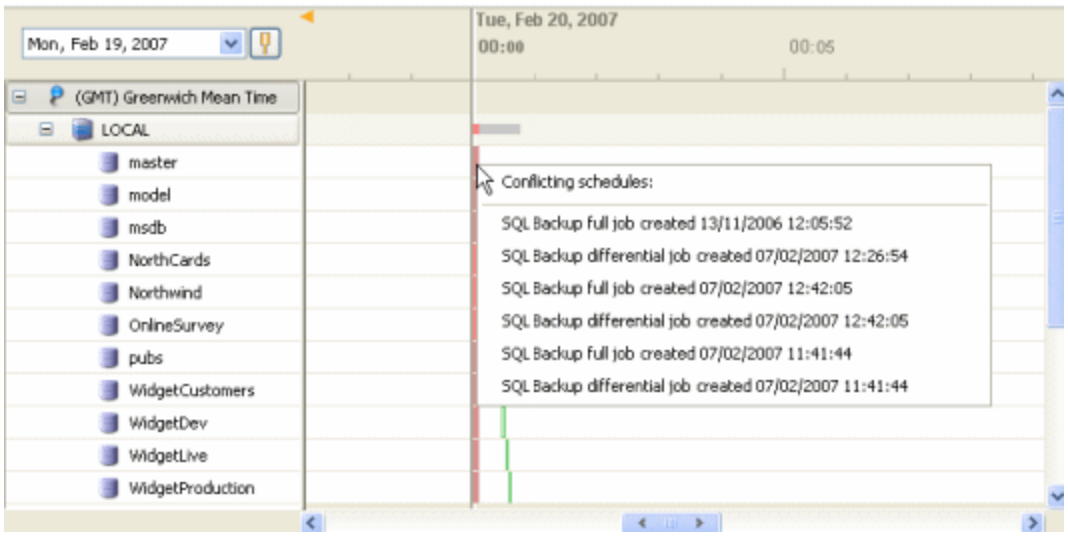
SQL Backup estimates the time duration of scheduled backup jobs based on the last successful backup on the same database and of the same type. (Note that for multi-step jobs that include native SQL Server steps, the time estimate takes account of the SQL Backup backup step only.) Full and differential restore operations that are scheduled for the future are shown using the icon for restoring a **full** backup, because SQL Backup cannot determine the backup type. Disabled jobs are not shown in the time line.

You can move your mouse pointer over a block to see more details.



If a database has two or more operations represented on the time line that are too close together to be visually distinguished, the operations are represented by a gray block
. You can zoom in to the gray block to see the individual operations. For details of how to zoom in, see Navigating the Time Line below.

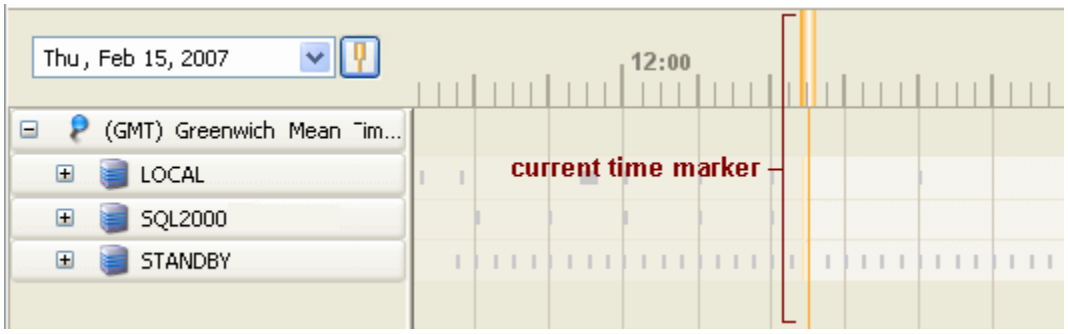
If more than one backup or restore operation is scheduled at the same time on the same SQL Server instance, the time slot in which the conflict occurs is highlighted pink, and the conflicting operations are highlighted red. To see a list of all the visible jobs that conflict, right-click in the shaded area.



If a job comprises multiple schedules, the schedule that causes the conflict is also shown in the list. Note that SQL Backup lists only those jobs for the time period that you can currently see in the time line; for a full list of all conflicting jobs for the time slot, ensure that you can see the entire time slot. For details of how to change the time line view, see Navigating the Time Line below. You can click the name of the job or schedule in the list to select it in the Jobs tab.

Navigating the Time Line

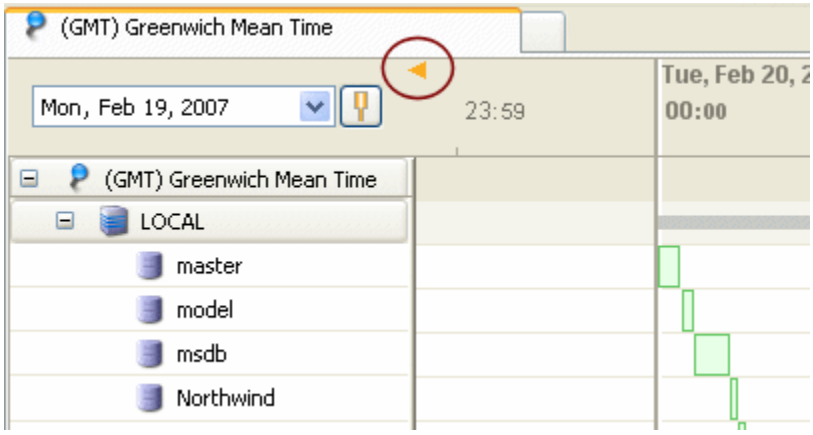
The current time is indicated with a yellow marker:



If the current time marker is not visible in the view, a small, yellow arrow is displayed in the top left

(if you are looking at future jobs) or the top right

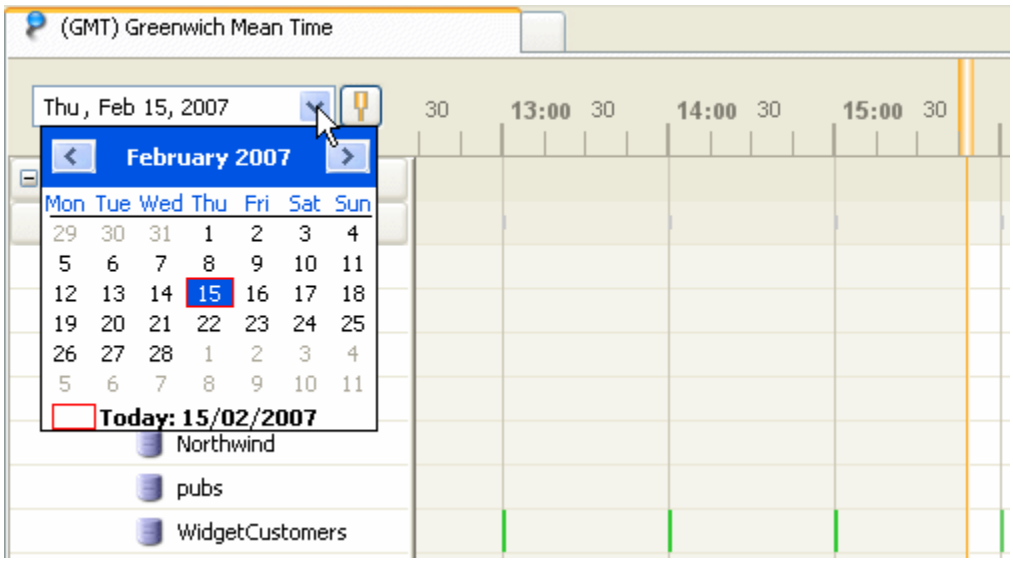
(if you are looking at activity history).



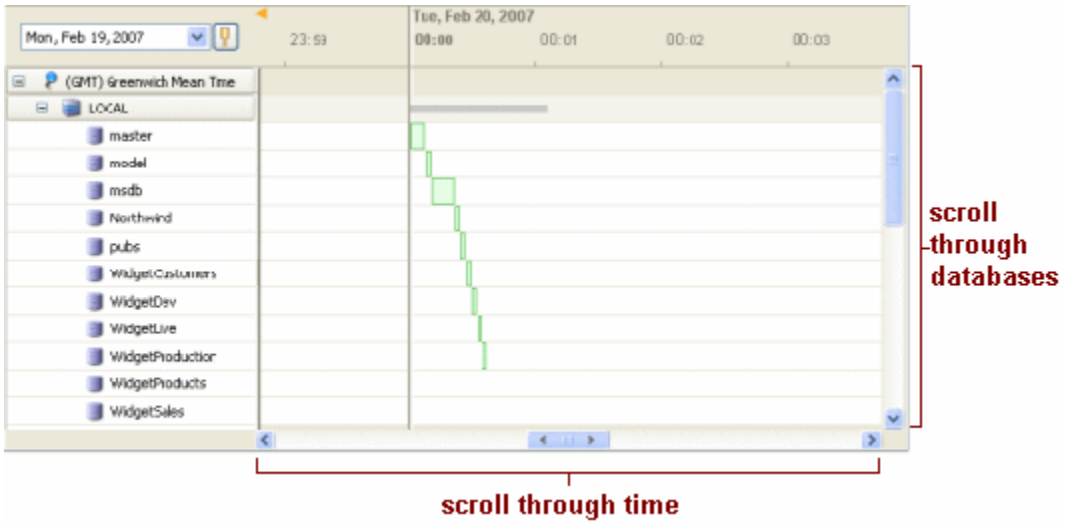
You can navigate to the current time by clicking the arrow, or by clicking

For more information on the current time marker, see Time settings and locations below.

The date corresponding to the left edge of the time line view is displayed above the **Registered SQL Servers** pane. To change the date, click the arrow and select the date you require from the calendar:



There are a number of ways to scroll through the time line:



You can scroll to a different time or date by clicking the left and right arrow buttons

to move in the direction indicated. To scroll more quickly, drag the central slider

left or right; the further you drag the slider, the faster the time line moves.

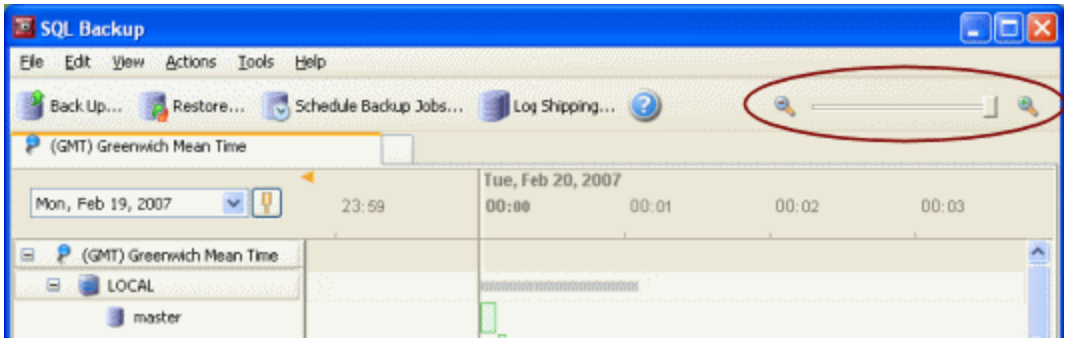
To move the time line and SQL Server Registration pane up or down so that you can see details for databases that are not shown in the view, click the up and down arrow buttons, use the scroll bar, or press CTRL and rotate your mouse wheel button.

Alternatively, you can move your mouse pointer to an area that does not show any activities and drag the time line to the required position (up, down, left, or right) using the grab hand

To zoom out, drag the **Zoom** slider to the left or click

; to zoom in, drag the **Zoom** slider to the right or click

:

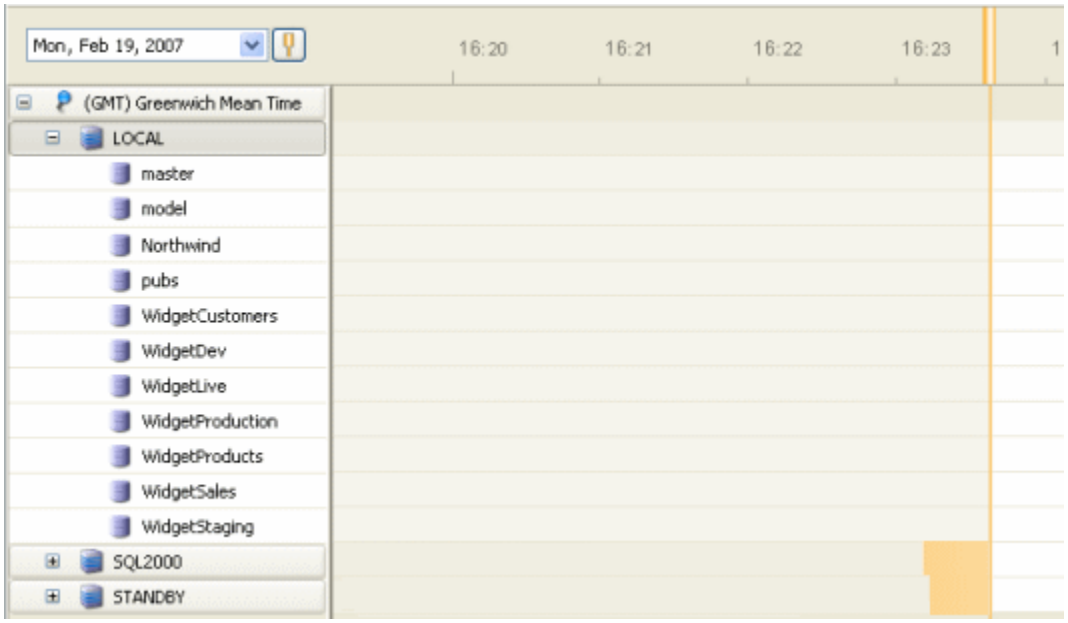


Alternatively, with the mouse pointer over the time line, rotate your mouse wheel button to zoom in or out.

When you perform a backup or restore operation, on completion, SQL Backup automatically scrolls to the activity in the time line. Similarly, when you schedule a job, on completion of the job creation SQL Backup scrolls to the first run of the new job.

Refreshing the data

For the selected SQL Server, and for all databases that are visible in the time line, SQL Backup refreshes the display every minute.



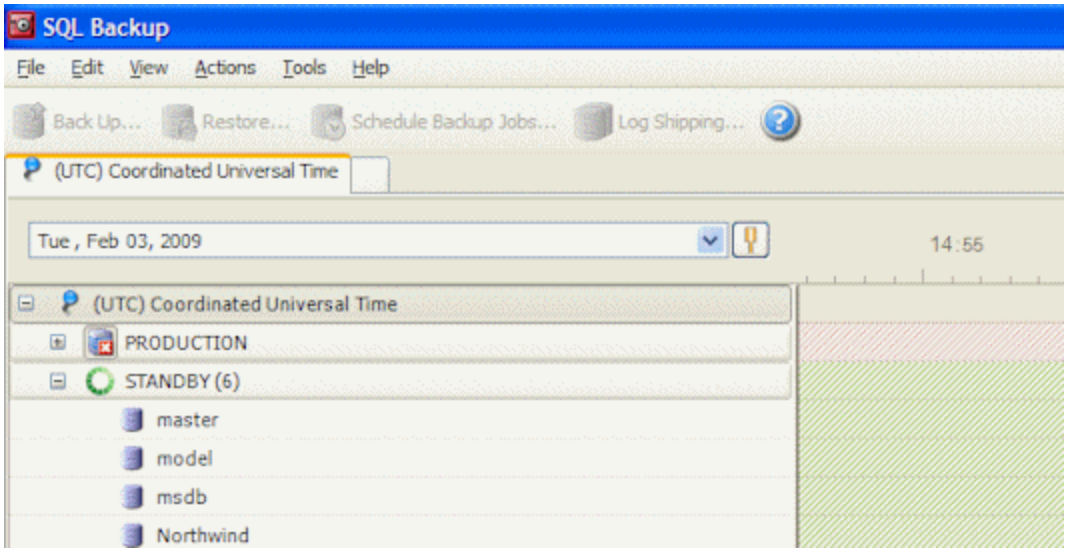
In the example above, the information for the LOCAL SQL Server instance is up-to-date, but the orange shading for SQL2000 and STANDBY shows that they have not been refreshed for approximately one minute.

You can move your mouse pointer over the shaded area to see the precise time at which the SQL Server was last updated. For more details about the different times that are provided, see Time settings and locations below.

Red shading

is used on the time line to show that the connection to a SQL Server instance has failed. Green shading

is used to show that data is being collected for that particular section of the time line.



To refresh the display for a SQL Server instance, click the instance in the **Registered SQL Servers** pane, and press F5. While the connection to the instance is being refreshed,

is displayed. You can continue to interact with the time line while a connection is being refreshed.

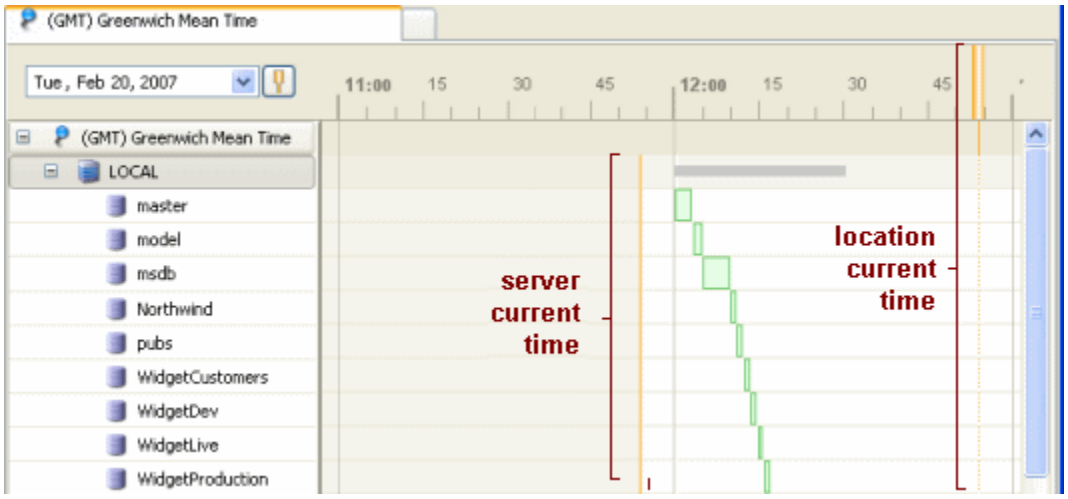
If you have registered a large number of instances and associated databases, you can reduce the time it takes to refresh the time line by creating groups and distributing your instances between them; data is not refreshed for collapsed groups.

Time settings and locations

SQL Backup uses *locations* so that you can manage SQL Servers that are situated in different time zones.

SQL Backup sets up an initial location for you to register your local SQL Server instances. This default location uses the time set on the *client* co mputer (the computer on which you are running the GUI). The time zone is reflected in the tab at the top of the time line.

If there is a difference between the location time and the time set on the server (the *server time*), SQL Backup reflects this in the time line. For example, this may occur if the clocks on your servers have not been synchronized exactly. In the example below, the time on the server has been set to an hour earlier than the location time (in reality, the time difference may often be only a few minutes).



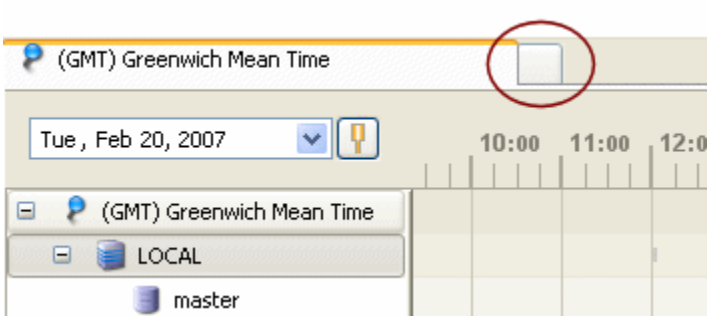
For the default location, the client time and location time are always the same.

> If your network is slow, or if your server is overloaded, you may see a difference between the server time and the client time even though their clocks are synchronized, due to the delay in SQL Backup receiving the data.

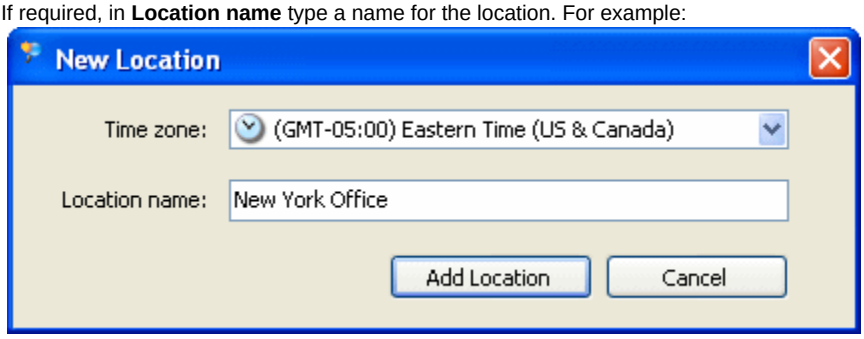
If you have servers situated in different world time zones, you can set up a separate time line for each time zone so that the current time marker shows the time local to the servers.

To create a new location:

1. On the **File** menu, click
   
   New Location, or click the blank tab at the top of the time line:

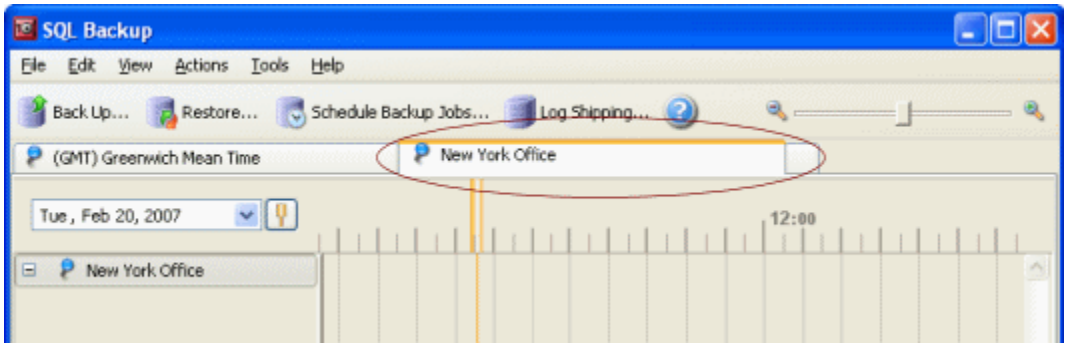


2. In the **New Location** dialog, from the **Time zone** drop-down list, select the time zone in which the servers are located.
3. If required, in **Location name** type a name for the location. For example:



4. Click **Add Location**.

The new location is displayed.



SQL Backup takes into account adjustments to time zone differences for daylight saving.

> SQL Backup uses the client time to calculate the location time by adding or subtracting the appropriate offset. Therefore, if your client time is incorrect, the location time will be incorrect by the same amount.

You can add SQL Servers to the new location in the usual way. See Adding SQL Server instances for more information.

To edit or delete a location, click the tab, and on the **Edit** menu, click **Edit** or **Delete** as required.

> If you delete a location that contains registered SQL Server instances, all instances registered to the location are removed.

Interaction

Click an activity that has completed (to the left of the current server time) to select the corresponding entry in the Activity History. Double-click a past activity in the time line to see its properties; for details, see Properties.

Click an activity that is scheduled for the future (to the right of the current server time) to select it in the Jobs tab. Note that a number of activities on the time line can correspond to a single job in the **Jobs** tab.

Double-click a future activity to edit the job. The **Edit Backup Job** wizard is displayed. This is identical to the **Schedule Backup Jobs** wizard, but contains the settings for the selected job for you to edit. For details, see Scheduling backup jobs.

> Not all jobs can be edited using the SQL Backup GUI. For example, filegroup backup jobs must be edited using a SQL application such as SQL Server Management Studio or Enterprise Manager.

You can click an activity in the Activity History, or a job in the Jobs tab, to select it in the time line; selected activities are shown in the time line with shading, for example:

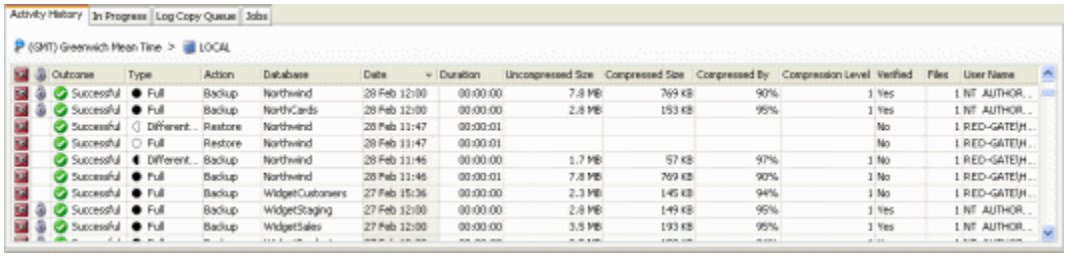
for backup operations or

for restore operations that have already occurred.

The Activity History

Select the **Activity History** tab on the main window to view a history of backup and restore operations.

Select the SQL Server instance or database for which you want to see the activity history in the Registered SQL Servers pane.



To update the details, on the **View** menu, click

Refresh Connection.

If you are experiencing problems with the performance of the Activity History, see Populating the Activity History below.

Display

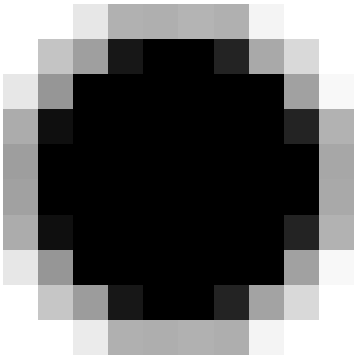
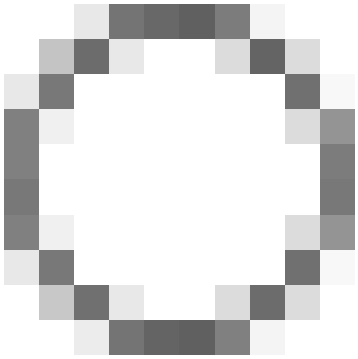
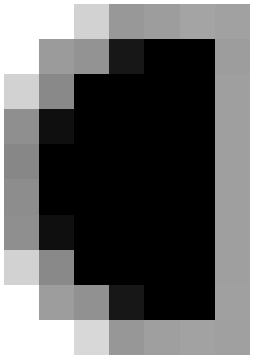
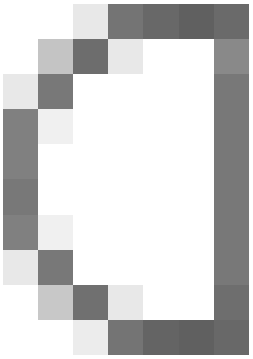
In the first column,

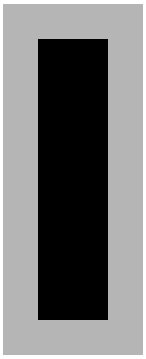
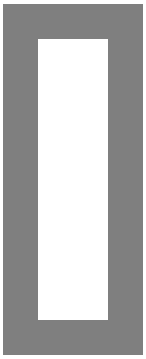
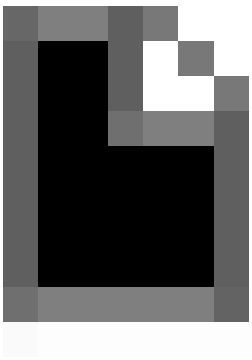
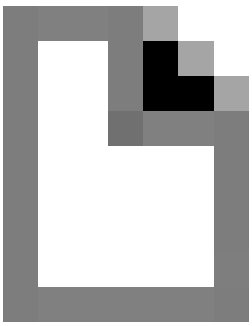
indicates that the activity was performed using SQL Backup (version 5 or later). If no icon is shown, the activity was performed using an earlier version of SQL Backup, or another program (such as SQL Server native backup).

In the second column,

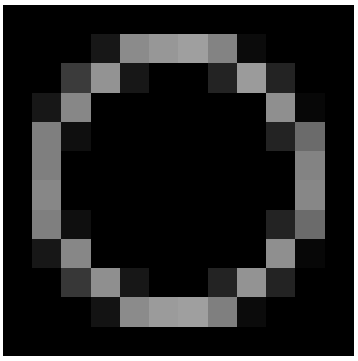
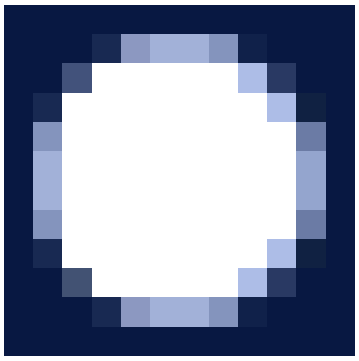
indicates that the backup is encrypted.

Type shows the type of activity. An icon is displayed so that you can scan the list easily:

	Full backup		Full restore
	Differential backup		Differential restore

	Transaction log backup		Transaction log restore
	File or filegroup backup		File or filegroup restore

If you have a dark background colour, the colours of the icons do not reverse, but the outline changes. For example:

	Full backup		Full restore

Date shows the date and time of the activity in the time zone local to the SQL Server.

Duration shows the time taken to perform the backup or restore operation. Note that this does not include the time taken for any activities done on completion, such as erasing existing files, copying backups to a network share or hosted storage, and sending notification emails.

Uncompressed Size shows the total amount of uncompressed data passed to SQL Backup by SQL Server. This may be larger than the **Data size** and **Log size** reported in the database properties because data is passed to SQL Backup in blocks of a minimum of 64 KB. For example, a 512 byte block of data from SQL Server is passed to SQL Backup as a 64 KB block. You can use the *sp_spaceused* stored procedure if you want to check how much of the data and log file allocation is used; for details, refer to your SQL Server documentation.

Compressed Size shows the size of the backup file(s) on disk.

Compressed By shows the percentage compression; SQL Backup calculates the percentage compression of a backup by comparing the size of the SQL Backup backup with the total database size. For example, if a database comprises a 10 GB data file and a 3 GB transaction log file and SQL Backup generates a full backup of the database to create a backup file that is 3 GB, the compression for this backup is calculated as 77%, [1-(3/13)]x100.

Compression Level shows the compression level used for the backup.

Files shows the total number of files created for a backup, or the number of backup files restored for a restore operation.

User Name displays the login used to access the SQL Server.

Populating the Activity History

The information displayed is pulled from the local activity cache, which in turn extracts the data from the *msdb* database and the SQL Server Compact database (created when the SQL Backup server components are installed). You can control the amount of information that is displayed as follows:

- When you add a SQL Server instance to SQL Backup, you can restrict the number of days of native backup and restore history that is imported from the *msdb* database using the **Native backup and restore history to import** option. For more information, see Adding SQL Server instances.
- Once you have installed the SQL Backup server components, you can control the number of days of backup and restore history that is kept in the *msdb* database and SQL Server Compact database using the File Management options for that SQL Server. From the **Tools** menu select **Server Options**, and on the **File Management** tab select the **Delete all backup and restore history** option. For more information, see File management options.

For more information about the activity cache, see Activity cache location.

> If you change the name of your SQL Server, SQL Backup displays only activities that have occurred since the name was changed.

Interaction

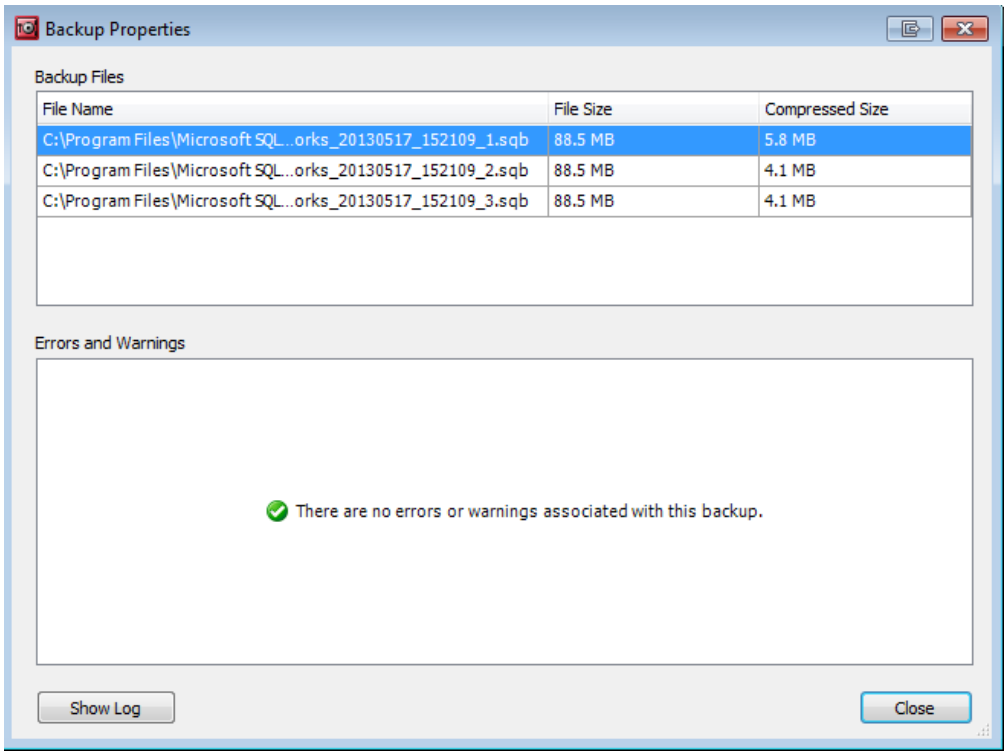
By default, the activities are listed in date and time order, with the most recent first. You can change the sort order by clicking the header for the column by which you want to sort the list. Click the header again to reverse the order. You can also resize the columns as required.

When you select an activity in the **Activity History**, the time line view changes to show the activity and it is highlighted. Similarly, when you click a completed activity in the time line, it is selected in the **Activity History**.

Properties

To display the properties of a backup or restore operation and any associated errors or warnings, right-click the activity in the **Activity History** an d select **Properties**. Alternatively, double-click the activity in the time line or **Activity History**. Activity properties are available only for activities performed by SQL Backup (version 5 or later).

Backup Properties



File Name lists the backup files created by the backup operation.

File Size displays the total amount of uncompressed data passed by the SQL Server to SQL Backup in order to create the backup. The reported size may be larger than the size of the data and log files because data is passed to SQL Backup in 64 KB blocks. For example, a 512 byte block

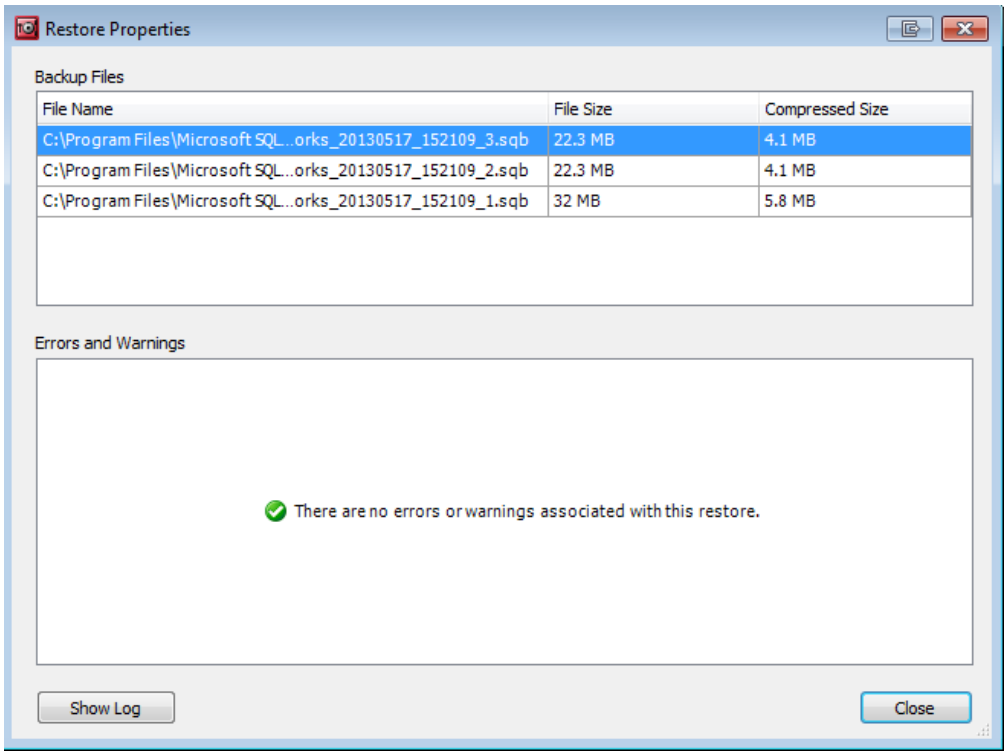
of data from SQL Server is passed to SQL Backup as a 64 KB block. To check how much of the data and log file allocation is actually used, use the **sp_spaceused** stored procedure. For details, refer to your SQL Server documentation.

Compressed Size displays the size of each backup file.

The **Errors and Warnings** panel displays any error or warning messages associated with the backup. For more information on particular errors or warnings, see the SQL Backup warnings 1 - 499 and SQL Backup errors 500 - 5292.

To view the SQL Backup log file for the backup operation, click **Show Log**. The **Activity Log** is displayed.

Restore Properties



File Name lists the files included in the restore operation.

File Size for single file backups displays the amount of uncompressed data passed to SQL Backup when the backup was created. For backups split into multiple files, **Data Size** displays the uncompressed size of each backup.

Compressed Size displays the size of each backup file in the restore operation.

The **Errors and Warnings** panel displays any error or warning messages associated with the restore. For more information on particular errors or warnings, see the SQL Backup warnings 1 - 499 and SQL Backup errors 500 - 5292.

To view the SQL Backup log file for the restore operation, click **Show Log**. The **Activity Log** is displayed.

The In Progress tab

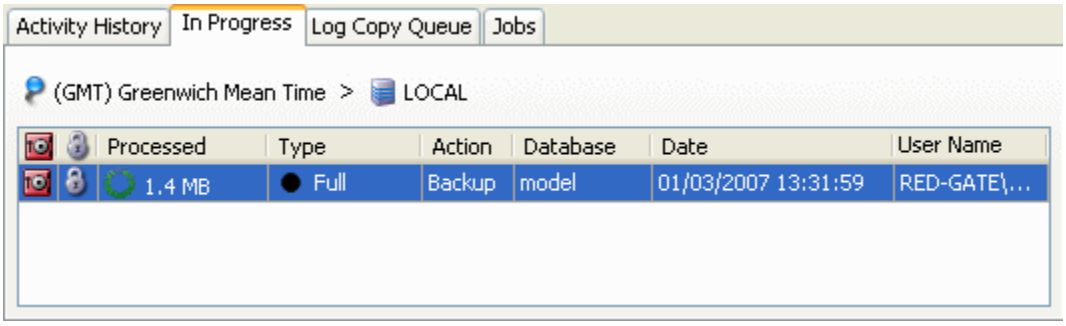
The **In Progress** tab shows the progress of SQL Backup backups and restores for whichever SQL Server instance or database is selected in the Registered SQL Servers pane. To update the details, on the **View** menu, click

Refresh Connection.

> Transaction log backups currently being copied to a network location (for example, during log shipping) are listed in the Log Copy Queue tab.

Backing up and restoring

An entry is displayed for each database that is currently being backed up or restored.



Any network resilience settings you have selected (on step 4 of the Back Up wizard, step 5 of the Schedule Backup Jobs wizard, or using the `DISKRETRYCOUNT` and `DISKRETRYINTERVAL` options) will apply when writing the backup to disk or restoring from the backup.

When the backup or restore completes, the entry is removed from the In Progress tab and an entry is added to the Activity History. For more information, see The Activity History.

> You can also use the *sqbstatus* extended stored procedure to view the active SQL Backup backup and restore processes from within your SQL application (for example, SQL Server Management Studio).

The Log Copy Queue

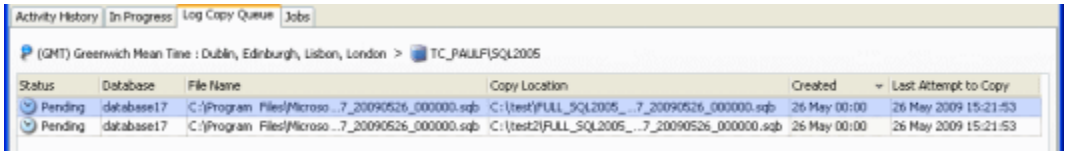
When you create a transaction log backup file using SQL Backup, *and* request that the file is copied to another location (either using the `COPYTO` keyword, or with the **Copy backup to network** option in the SQL Backup wizards), the backup files are placed in the log copy queue before they are copied. SQL Backup checks the log copy queue regularly for transaction log backups that are waiting to be copied (see below for details of the copy schedule). Backup files are then copied in the correct order; log backup files for lower LSNs are copied first.

If there is a problem during the copying process (caused by an extended network outage, for example) and the copy fails, the backup files remain on the log copy queue, ready for another copy attempt.

> To revert to the SQL Backup 5 copy behavior for transaction log backups, you must edit the backup script and specify the `USESIMPLECOPY` keyword in the `BACKUP` command. You cannot specify the old copy behavior directly from the Log Shipping wizard.

Display

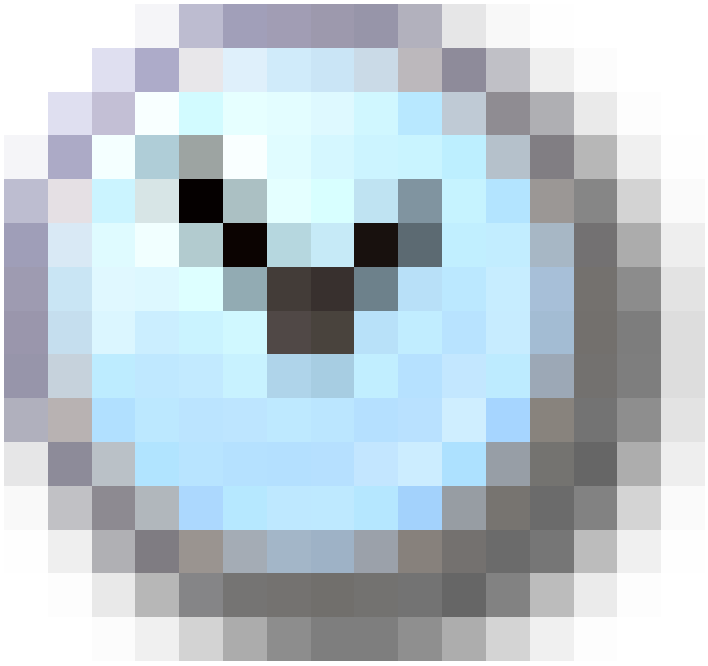
The **Log Copy Queue** tab shows every transaction log backup file in the log copy queue for the SQL Server instance or database that you have selected in the Registered SQL Servers pane.

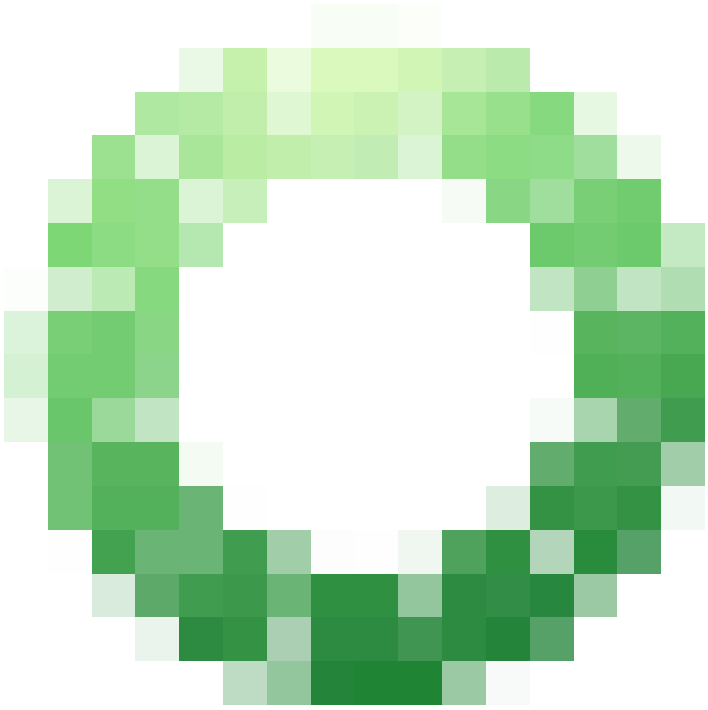


To update the details, on the **View** menu, click

Refresh Connection (or press F5).

The **Log Copy Queue** tab shows basic information such as the source database, file name, creation date, and copy location for each file on the queue. The **Status** column indicates the current copy status of each file:

 **Pending**	The file is due to be copied to the location shown in the **Copy Location** column. If a previous attempt to copy was unsuccessful, the **Last Attempt to Copy** column indicates when the last copy attempt occurred. To show a log of all copy attempts for a file, double-click the item. The **Copy attempt log** is displayed.

 **Copying**	The file is in the process of being copied to the location shown in the **Copy Location** column. Note that transaction log copy activities are not shown on the **In Progress** tab.

Interaction

By default, the transaction log backup files are listed in order of creation date, with the most recent first. You can change the sort order by clicking the header for the column by which you want to sort the list. Click the header again to reverse the order. You can also resize the columns as required.

To show a log of all copy attempts for a particular file, double-click the item. The **Copy attempt log** is displayed.

Log copy schedule

By default, SQL Backup attempts to copy each transaction log backup file as soon as it is added to the log copy queue. If the first copy attempt fails, subsequent copy attempts are made according to the following schedule:

Copy attempt	Copy attempt is made...
1	Immediately
2	2 minutes after the 1st copy attempt failed
3	4 minutes after the 2nd copy attempt failed
4	6 minutes after the 3rd copy attempt failed
5	8 minutes after the 4th copy attempt failed
6	10 minutes after the 5th copy attempt failed
All subsequent copies	10 minutes after the preceding copy attempt failed

SQL Backup will continue trying to copy the oldest transaction log backup file every 10 minutes, for up to 24 hours.

You can change the copy schedule by creating and editing registry entries in the following registry key: *HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\SQL Backup\BackupSettingsGlobal\<instance name>*

Changes to these registry settings will not take effect until you restart the SQL Backup Agent Service.

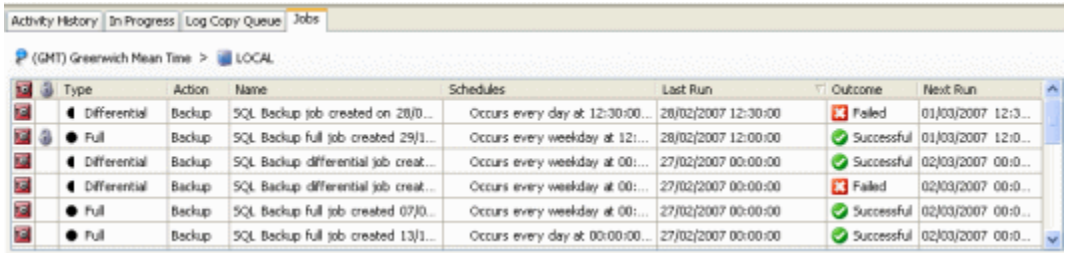
Registry entry	Type	Description
COPYTO:CopyIntervalInMinutes	DWORD	The initial interval between copy attempts (2 minutes by default).

COPYTO:MaxCopyIntervalCount	DWORD	The number of copy attempts for which the interval to the next copy attempt is increased (5 by default).
COPYTO:ExpiryIntervalInMinutes	DWORD	The maximum time to continue with copy attempts (1440 minutes by default).
COPYTO:ThreadCount	DWORD	The number of files that can be copied simultaneously (5 by default). Maximum value is 10.

The interval between copy attempts after the number of attempts specified by COPYTO:MaxCopyIntervalCount is derived by multiplying COPYTO:CopyIntervalInMinutes with COPYTO:MaxCopyIntervalCount. For example, using the default values, this is 2 x 5 = 10 minutes.

The Jobs tab

The **Jobs** tab displays details of SQL Backup jobs that have been scheduled, and jobs that have been scheduled using SQL Server.



Select the SQL Server instance or database for which you want to see the jobs in the Registered SQL Servers pane.

To update the details, on the **View** menu, click

Refresh Connection.

Display

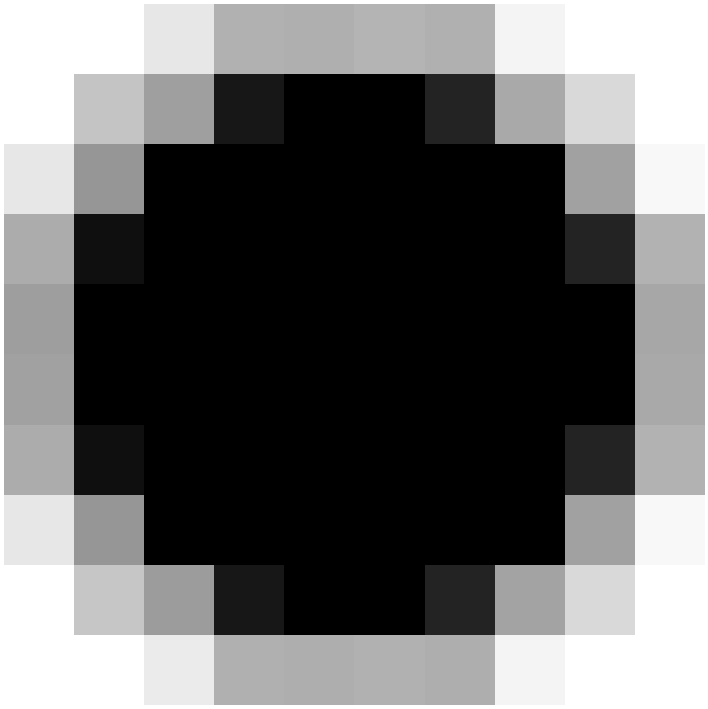
In the first column,

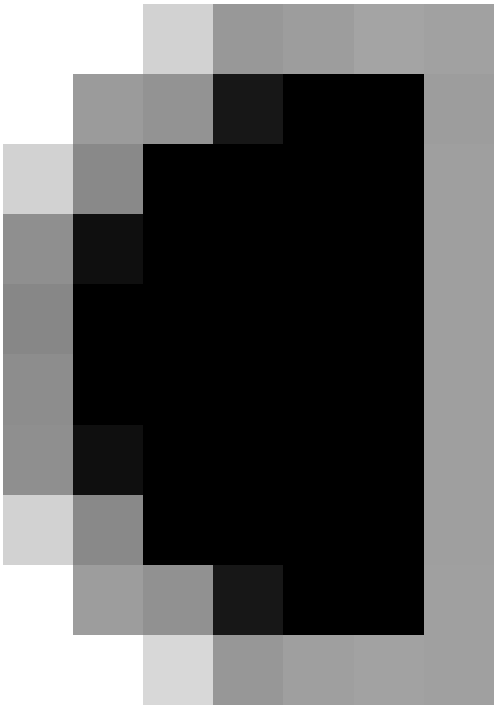
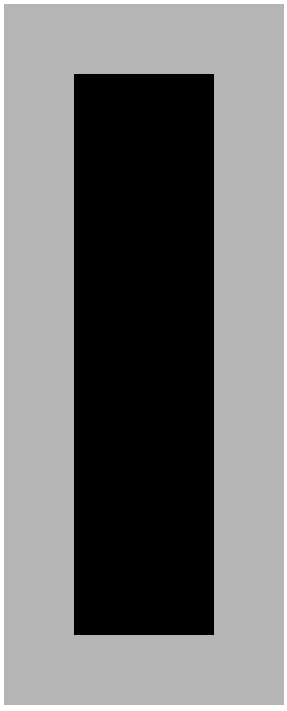
indicates that the job was created using SQL Backup (version 5 or later). If no icon is shown, the job was created by an earlier version of SQL Backup and the server components have not yet been upgraded to SQL Backup 5 or later, or the job was created by another program (such as SQL Server native backup).

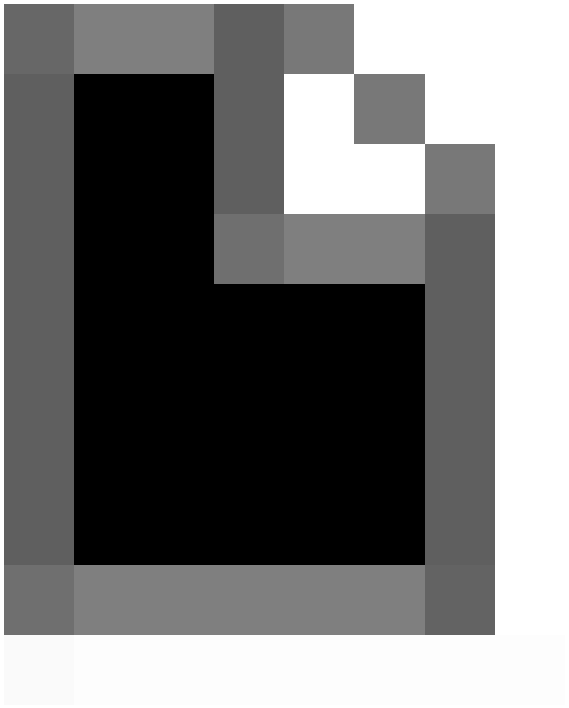
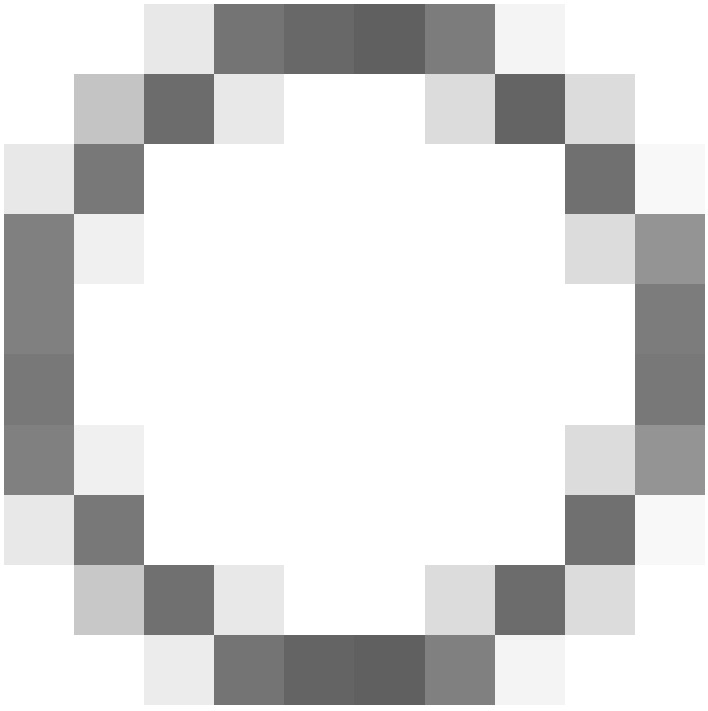
In the second column,

indicates that the backup will be encrypted.

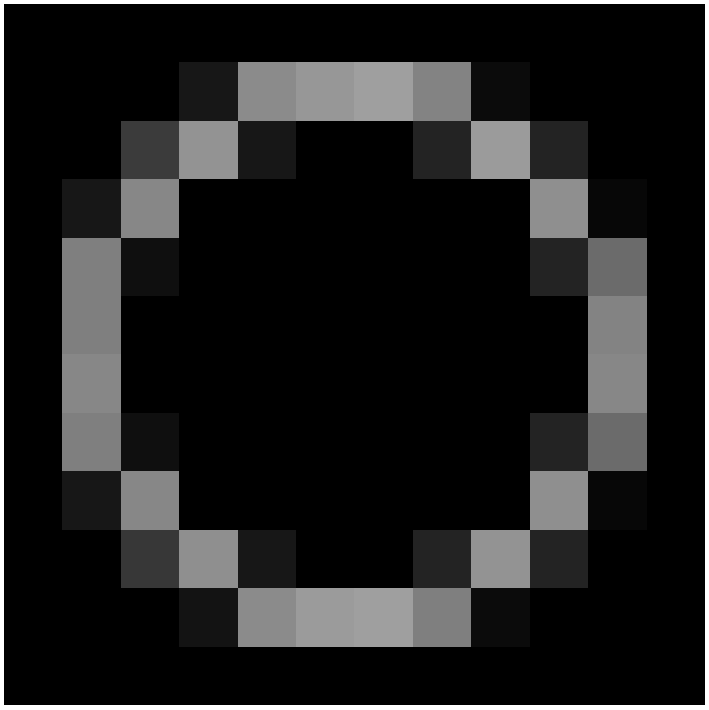
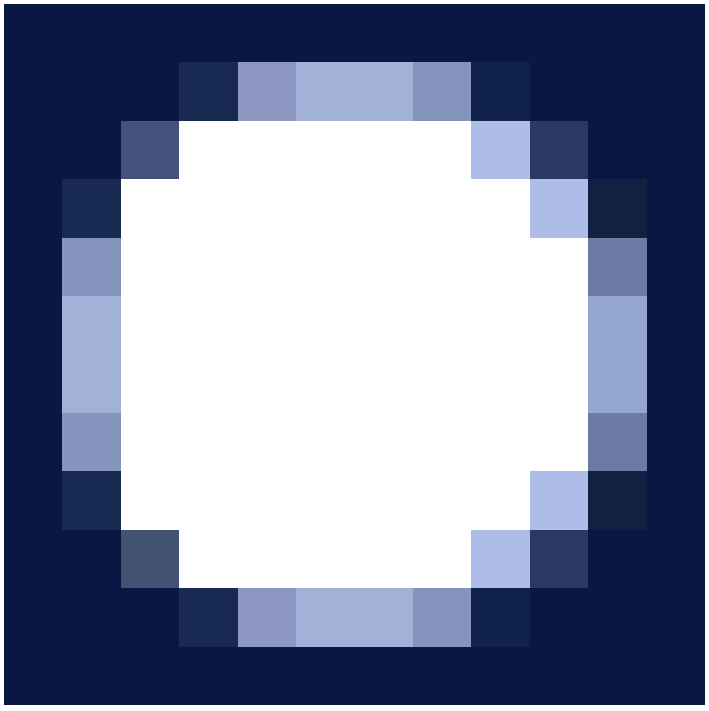
Type shows the type of activity. An icon is displayed so that you can scan the list easily:

	Full backup

	Differential backup
	Transaction log backup

	File or filegroup backup
	Full, differential, transaction log, file, or filegroup restore
All	Backup set restore (Full and differential and/or transaction log)

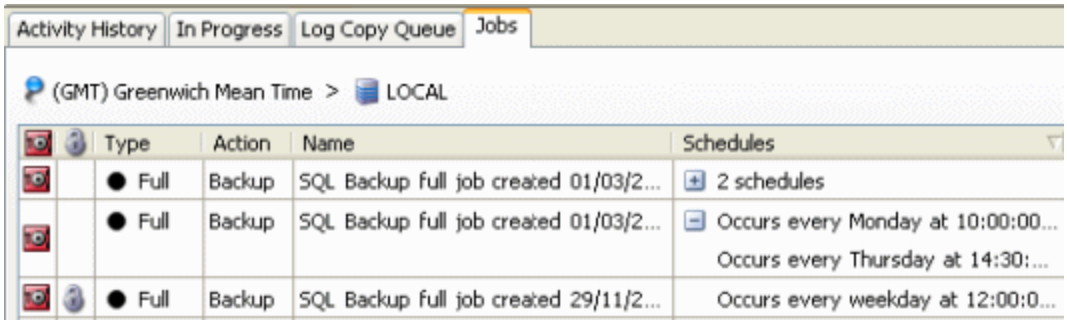
Note that if you have a dark background colour, the colours of the icons do **not** reverse. For example:

	Full backup
	Restore

Dates and times are displayed in the time zone local to the selected SQL Server.

If more than one schedule has been created for a job, the **Schedules** column shows the number of schedules. Click

to see the individual schedules. For example:



If a job has never been run, **Outcome** shows

Unknown.

Interaction

By default, the jobs are listed in order of the date and time of the last run, with the most recent first. You can change the sort order by clicking the header for the column by which you want to sort the list. Click the header again to reverse the order. You can also resize the columns as required.

When you click a job in the **Jobs** tab, the Time Line view changes to show the activities that correspond to the job and the activities are highlighted. Similarly, when you click a future activity in the time line, its corresponding job is selected in the **Jobs** tab.

To edit a SQL Backup backup job, right-click the job and click **Edit**, or double-click the job. The **Edit Backup Job** wizard is displayed. This is similar to the Schedule Backup Jobs wizard; the wizard contains the settings for the selected job. You can edit only one job at a time. The edit feature is not available for jobs that do not display the

icon. Note that some types of SQL Backup jobs (for example, filegroup backup jobs and restore jobs) cannot be edited using the graphical user interface.

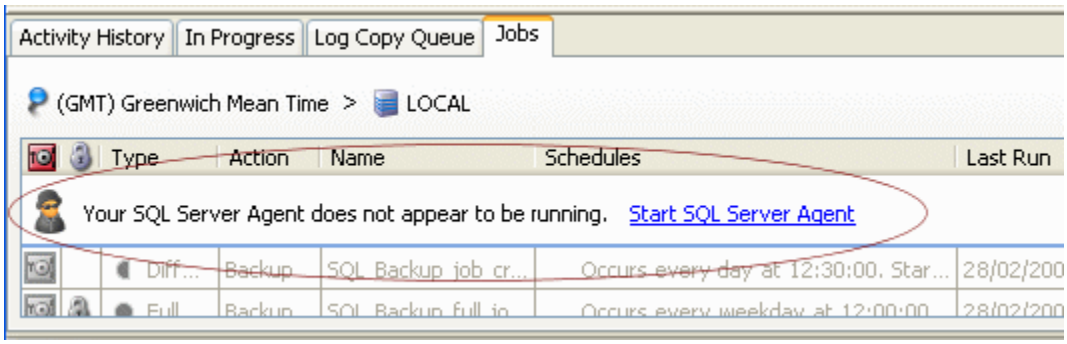
To delete a job, right-click the job and click **Delete**.

To run a job, right-click the job and click **Start**.

To stop a job that is running, right-click the job and click **Stop**. If you click **Stop** for a job that is not currently running, an error is displayed.

SQL Server Agent

If the SQL Server Agent for the selected SQL Server instance is not running, a message is displayed at the top of the list, and the list appears dimmed:



You can start the SQL Server Agent by clicking the **Start SQL Server Agent** link.

Backing up

These pages explain how to create backups with SQL Backup:

- Creating backups
- Scheduling backup jobs
- Backing up all databases on an instance

Creating backups

SQL Backup provides the Back Up wizard to guide you through the process of creating a backup. To start the Back Up wizard, click  **Back Up**.

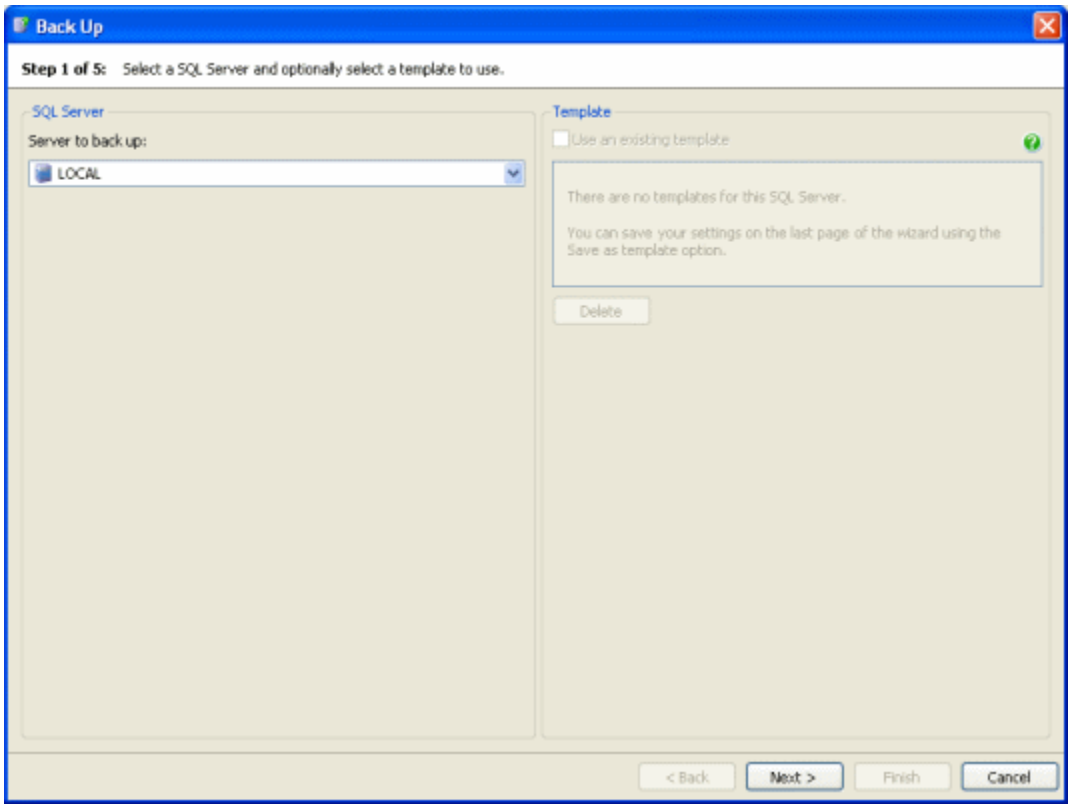
The Back Up wizard comprises the following steps:

Step 1:	Select the SQL Server, and optionally choose a previously saved template.
Step 2:	Select the type of backup, and the database that you want to back up.
Step 3:	Specify the backup file locations (including any copies to hosted storage or a network location), and set options for existing backup files.
Step 4:	Specify compression, encryption, and optimization options, and actions required on completion.
Step 5:	Review the backup summary and script, and start the backup process.

Next: specify SQL Server

Creating backups - specify SQL Server

On step 1 of the wizard, select the SQL Server you want to back up.



In the **Server to back up** list, click the name of the SQL Server instance for the databases that you want to back up. SQL Servers that are not currently available (for example, because they are disconnected) are not displayed in the list.

> If the selected SQL Server does not have the server components installed, a warning is displayed and you must install the server components to proceed.

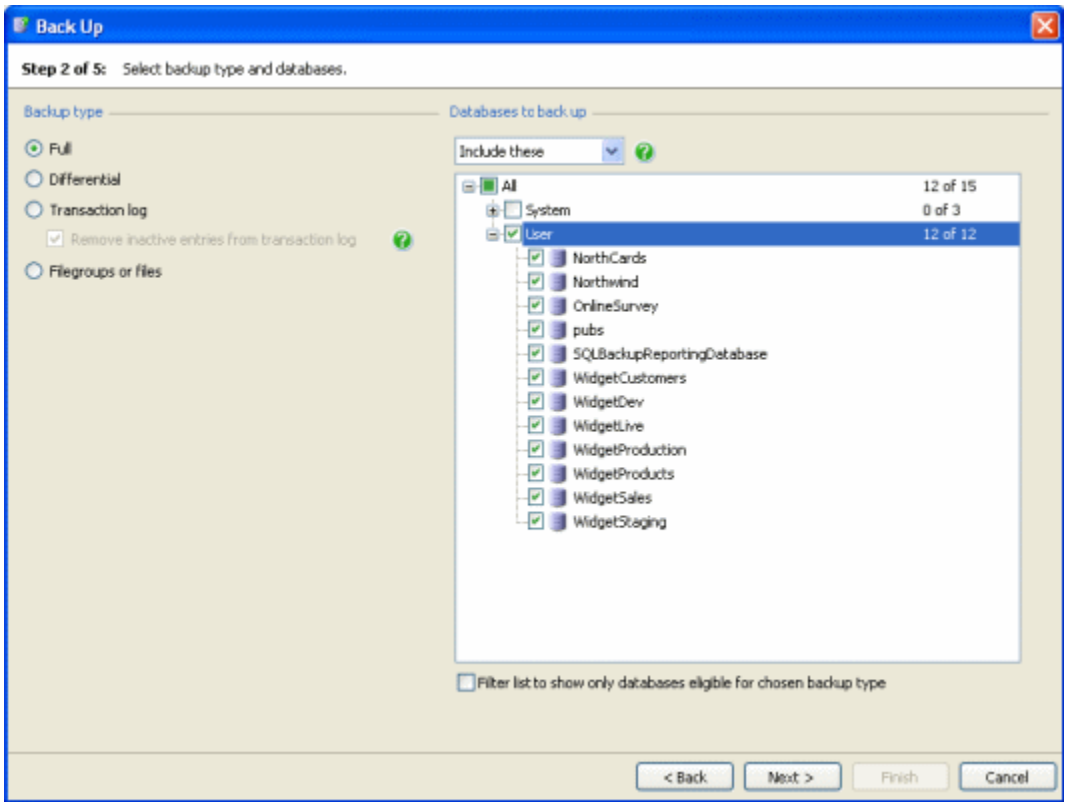
You can save the settings that you enter in the wizard as a *template* so that you can use them again. The option to save the settings as a template is displayed when you review the backup summary on step 6 of the wizard. If you have saved any templates, these are displayed in the right-hand pane.

To use a template, select the **Use an existing template** check box and click the name of the template in the list; when you click **Next**, the wizard will display those settings for you to edit if required.

Next: select the backup type and databases

Creating backups - select the backup type and databases

On step 2 of the wizard, select the type of backup you want to perform and the databases that you want to back up.



Backup type

Select the type of backup you require:

- **Full** creates a complete copy of the selected database.
- **Differential** creates a partial copy of the selected database.Only the changes since the last full backup was made are copied. You must create a full backup of the database before you can perform a differential backup.
- **Transaction log** copies all the log records that have been written to the live transaction log since the first full or differential backup and the last transaction log backup.
  You can back up the live transaction log only if the database uses the FULL or BULK LOGGED recovery model, and you have previously created a full backup. To truncate the live transaction log, select the **Remove inactive entries from transaction log** check box. The space reserved for the transaction log can then be reused by new transactions.

  > Truncating the transaction log does not reduce the size of the log file on disk; for this, refer to the SQL Server documentation on Shrinking a File.

- **Filegroups or files** creates a backup of selected files and filegroups. For example, you may want to back up a file or filegroup when the database size makes a full database backup impractical. You can create a filegroup or files backup for only one database at a time. The database must use the FULL or BULK LOGGED recovery model. You will be prompted to select the files and filegroups when you click **N ext**.

Databases to back up

For full, differential, or transaction log backups:

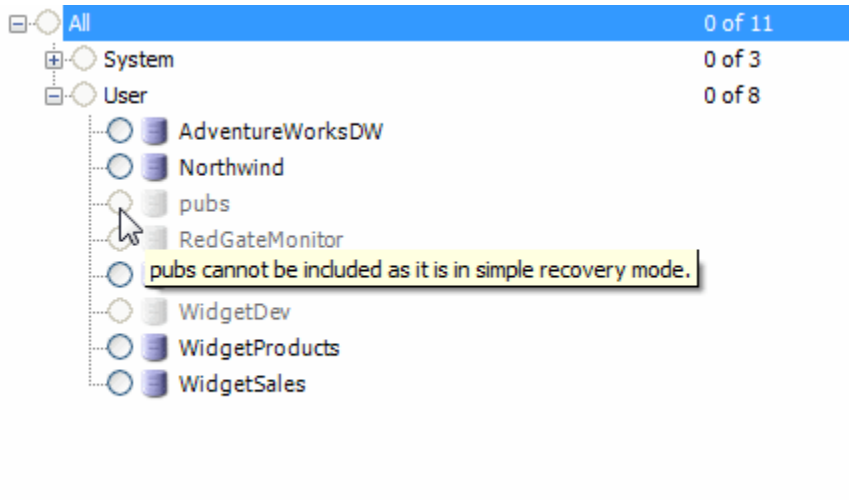
- To select the databases that you want to back up, click **Include these**, and then select the check box next to each database that you want to back up.Selecting **System** or **User** selects all databases within that group. Included databases are marked with ☑
- To back up all databases except for individual specified databases, click **Exclude these** and select the check box next to each database

that you do **not** want to back up.Excluded databases are marked with 

For filegroups or files, you can select only one database; click the database in the list.

> Only those databases that can be backed up with the backup type you selected are available. For example, if you selected **Differential**, only databases that have already had a full backup are available.

By default, all databases are displayed in the list, and those that are not available for selection given the selected backup type are displayed in gray. If you are using **Exclude these**, databases that are not available are automatically excluded. To see the reason why a database is not available, move your mouse pointer over the database name. In the example below, the **Filegroups or files** backup type has been selected.



You can filter the list so that only available databases are displayed in the list by selecting the **Filter list to show only databases eligible for chosen backup type** check box.

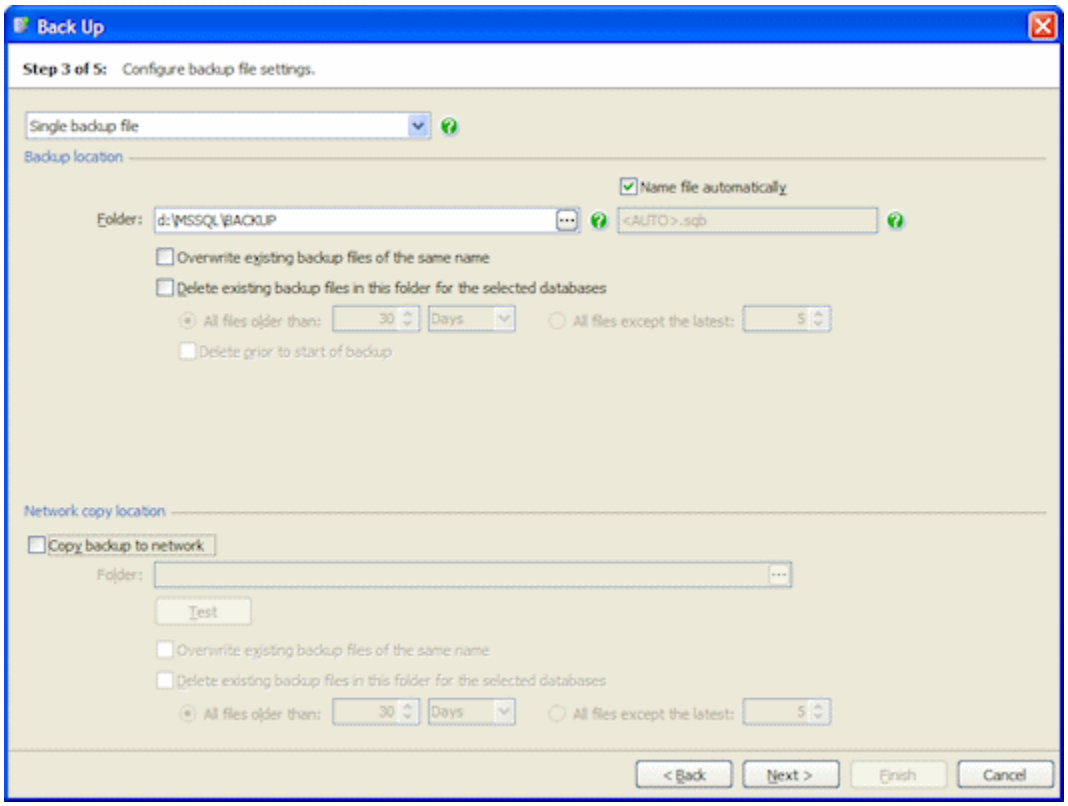
Click **Next** when you have selected the databases.

If you have chosen to back up files or filegroups, the wizard displays a page for you to select the files. The files and filegroups for the database you selected are displayed in a hierarchy. To select a file, select the appropriate check box. If you select the check box for a folder, all files and filegroups within that folder are automatically selected. When you have made your selection, click **Next**.

Next: file settings

Creating backups - file settings

On step 3 of the wizard, specify the locations and file names for your backups, including any copies to a network location, and configure settings to manage existing backup files.



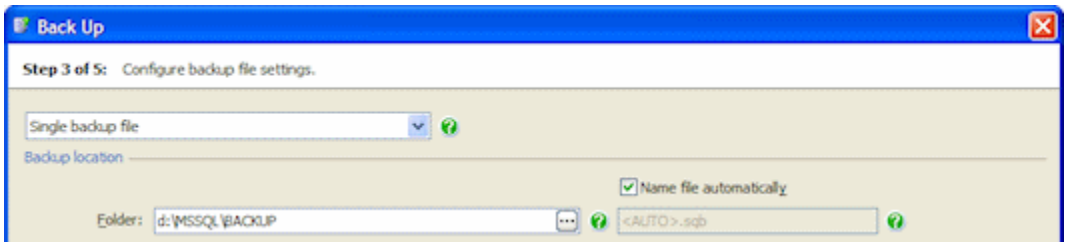
Choose:

- **Single backup file** to create each backup as a single file.
- **Single backup file, mirrored to second location** to create the backup as a single file and simultaneously make a duplicate of the backup during the backup process. The second backup is written in parallel to the first, so they will finish at the same time.This option is only available when you have selected a single database to back up. During the backup process, a warning is raised if any of the files cannot be written. However, the backup process continues as long as at least one specified backup file can be written. If none of the files can be written, an error is raised and the backup process is stopped.
- **Split backup into multiple files** to store each backup across a number of files.

Splitting the backup can speed up the backup process if backing up to a single file does not fully use the Input/Output capacity of your disks. For a single database backup, you can split (or 'stripe') your backup files across a number of disks. For multiple database backups, you can specify the number of files to split each database backup into (on the same disk). All backup files must be available when you restore the backup.

Backup location

For a **Single backup file**, if you have selected only one database to back up, a default folder and file name is displayed.



To change the file name, clear the **Name file automatically** check box, and type the new file name. To change the folder, type the new path or click

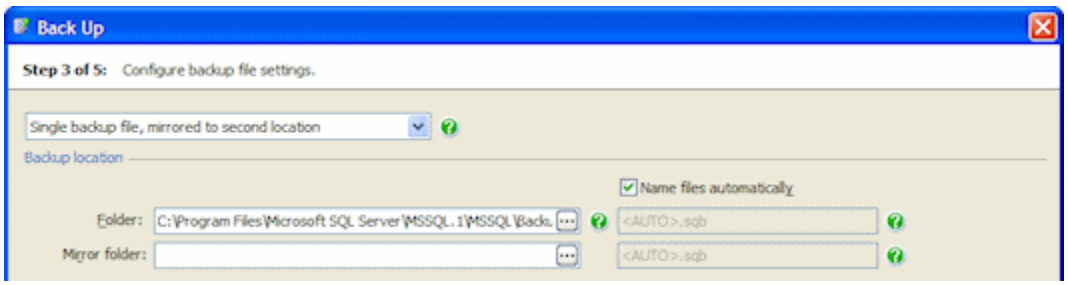

and specify the path using the folder browser. You can use tags in both the folder and file name. For more information, see File location tags.

> The file path is relative to the selected SQL Server. For example, if you back up a database on a remote SQL Server instance called *Se rverA* 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 machine.
>
> If you specify a network share as the backup file location, the SQL Backup Agent service 'log on' user must have *Full* permissions to access the location. Backing up directly to a network share can be slower than backing up locally and occasionally problematic. You are recommended to back up locally and use the **Copy backup to network** option instead. For more information, see Backing up and restoring on a network share.

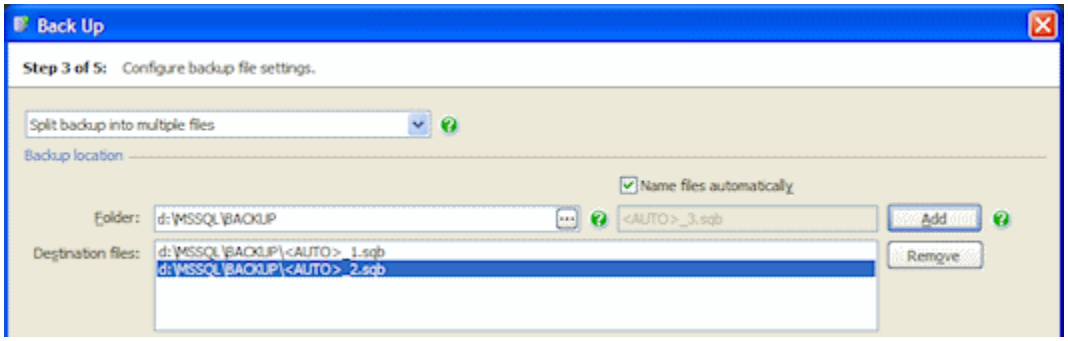
If you have selected multiple databases, the backup for each database is created in a separate file. The file names will be generated automatically using the <AUTO> tag, which uses the default file name format specified in your File management options. If you have not set up a default file name format, SQL Backup uses the SQL Server instance's default format for file names. A default folder for these files is displayed; to change the folder, type the new path or click

and specify the folder using the folder browser. If you want the backup file to be created in a different folder for each database, in the folder browser select **Create subfolder for each database**.

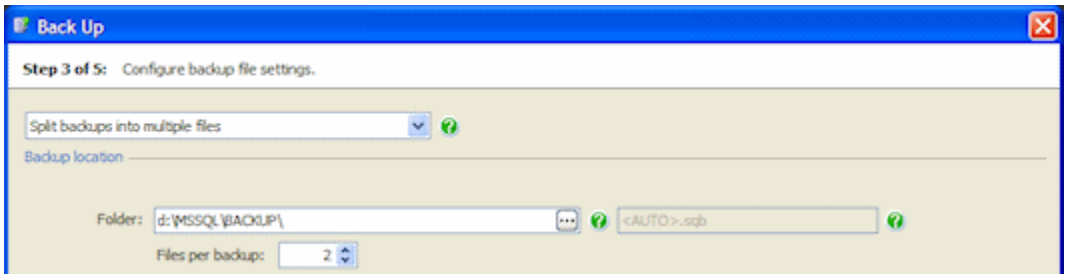
For a **Single backup file, mirrored to second location**, specify the backup file as described above, and additionally specify the folder and file name for the mirrored backups.



For **Split backup into multiple files**, if you have selected a single database, you must specify at least two destination files. Type or select each folder and file name as described above, and click **Add** to add it to the list.



If you have selected multiple databases, you can specify only one folder for the split backup files; the files will be named automatically.



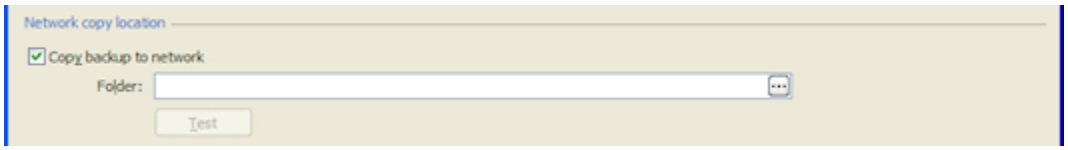
In **Files per backup**, type or select the number of files into which you want each backup to be split (maximum 32).

Network copy location

If you want a copy of the backup files to be created on a network share when the backup has completed, select **Copy backup to network**. Type the network share details in the **Folder** box or click


to use the folder browser.

The folder browser displays the local file system for the SQL Server you are backing up. Select the server you want to copy the backup to from the drop-down list, or click **Add Server** and type the server name or IP address. Other servers will only be visible to the local server if it has the appropriate permissions to write to or read from them. The name of the local server you are connected to and your user name are displayed above the **Server** list. This information may explain why some servers cannot be browsed.



If you typed the network share details, click **Test** to check that the "log on" user for the SQL Backup Agent service on the source SQL Server has permission to create files on the network share. If it does not have the appropriate permissions, an error message is displayed next to the button; if you choose to keep the network share details, a warning

is displayed in the Activity History when the backup has completed. If you browsed to the network share, this check is performed automatically.

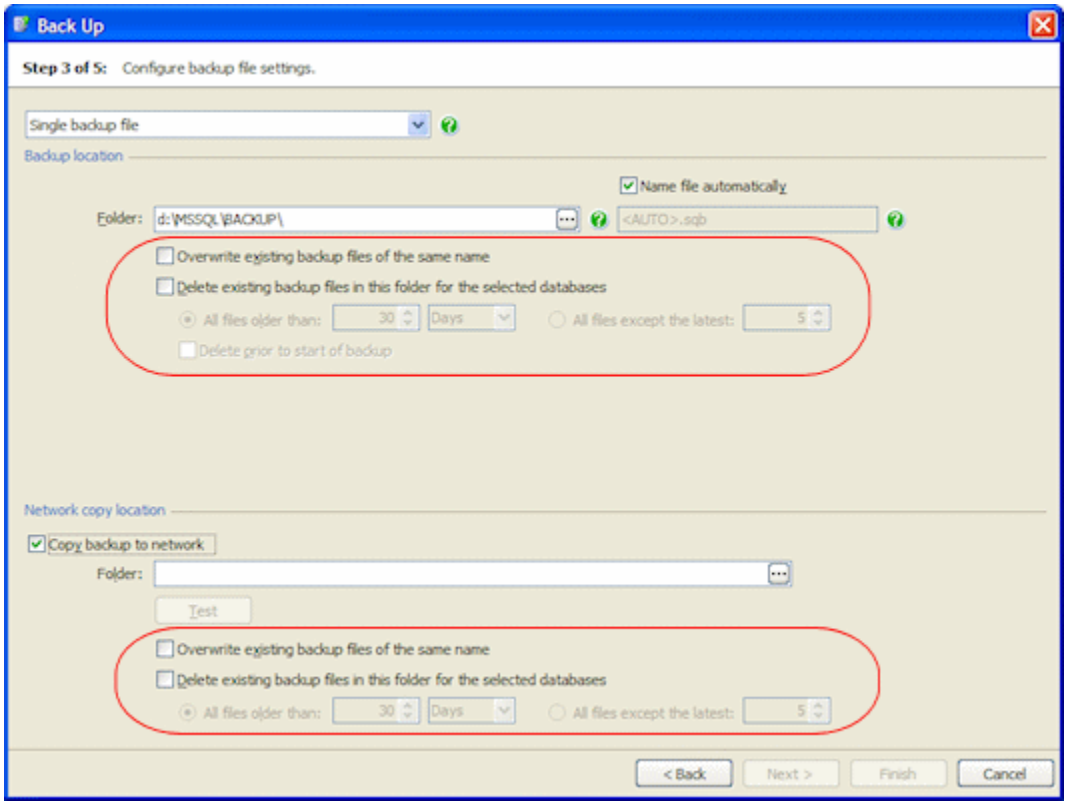
For more information about using network shares with SQL Backup, see Backing up and restoring on a network share. For details about setting up permissions to use a network share, see Permissions.

You may experience problems when you back up to network shares if your SQL Server is unable to write large data blocks (over 2 MB) to the network share. You can use the `MAXDATABLOCK` keyword with the BACKUP command to limit the data block size.

Managing backup files

SQL Backup provides a number of settings to automate the management of your existing backup files. You can specify these settings separately for the initial backup location, and the network copy location.

> If you specify a *network* path in the backup location folder, any network copy location options you specify will be applied to files at this path. Similarly, if you specify a *local* path in the network copy location folder, any backup location folder options will be applied to the copied files in the (local) network copy location.



Select **Overwrite existing backup files of the same name** if you want to overwrite any existing files with the share/path name.

If a file of the same name exists already and you have not chosen to overwrite it, the backup or network copy will fail.

Select the **Delete existing backup files in this folder for the selected databases** check box if you want SQL Backup to delete backups of the same type for the selected databases from the initial backup or network copy destination folders. You can restrict deletion of existing backup files by age (All files older than) or number (All files except the latest). By default, files are deleted when the backup process or network copy has completed. If the backup fails, the files are not deleted.

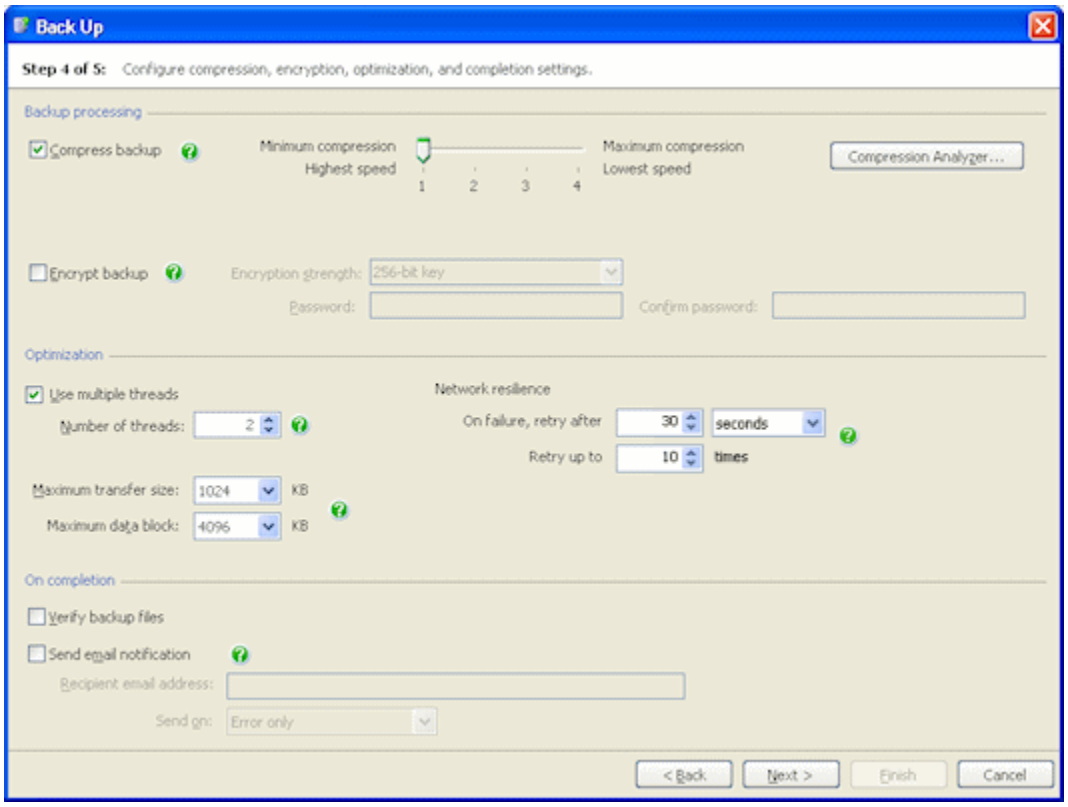
To delete the files *before* the backup is created, select the **Delete files prior to start of backup** check box. For example, you may want to do this to create space for the new backup files. However, note that if the backup fails, the old files will have been deleted; therefore, you are recommended to select this check box only if you have a copy of the existing backups. This option is available for the initial backup location only; existing network copies of files are always deleted after the copying process has completed.

If SQL Backup cannot list the contents of the folder that contains the files to be deleted, it cannot delete the files. Ensure that the user account from which you are running SQL Backup has permissions to list the folder contents.

Next: processing encryption settings

Creating backups - processing and encryption settings

On step 4 of the wizard, specify the compression, encryption, and optimization options for your backups.



Backup processing

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

> If SQL Backup Lite is installed on the SQL Server and you choose to compress your backup, compression level 1 is selected and you cannot change it.

For more information about compression levels, see Compression levels.

For full database backups, you can click **Compression Analyzer** to perform a test on the databases to check which compression level will produce the best result for your requirements. For more information about the Compression Analyzer, see Compression analyzer.

To encrypt the backup, select the **Encrypt backup** check box, then type a password for the backup in **Password**, and again in **Confirm password**.

Choose 128-bit or 256-bit encryption. **You must remember your password**; if you do not, you will not be able to access the encrypted backup.

Optimization

If you are backing up to a **Single file** or **Single backup file, mirrored to second location**, SQL Backup can use multiple threads to create the backups. This 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. For more information about using multiple threads, see Optimizing backup speed.

Maximum transfer size specifies the maximum size of each block of memory to be used when SQL Backup stores backup data. The default value is 1024 KB; you may want to change this value if a SQL Server reports that it has insufficient memory to service requests from SQL Backup.

Maximum data block specifies the maximum size of data blocks to be used when SQL Backup stores backup data. The default value is 4096 KB.

Note that if you want to specify the network transfer packet size, change the connection properties for the SQL Server. For more information, see

Connection Properties.

The **Network resilience** settings control retry behavior following a read/write error whenever SQL Backup:

- writes a backup file to disk
- copies a backup file to another disk location

Read/write errors are most likely to occur when SQL Backup is processing files through a network connection.

In most circumstances, you can leave **On failure, retry after** and **Retry up to** (n) **times** at their default values (30 seconds, and 10 times). To disable retries completely, specify **Retry up to** 0 **times**.

On completion

If you want to receive an email with a copy of the completion log, select the **Send email notification** check box, and enter the recipient's email address. This option is available only if you have entered your email settings. For more information about setting up email notification, see Email settings.

To send the log to multiple email addresses, type each address separated with a semi-colon (**;**). For example:

dba01@myco.com**;**dba02@myco.com

By default, email notifications are sent only when there are errors during the backup process (**Error only**). If you want to receive an email when an error or warning occurs, select **Error or warning**; to always receive an email on completion of the backup process (on success or failure), select **Any outcome**.

> If SQL Backup Lite is installed on the SQL Server, you cannot use email notification.

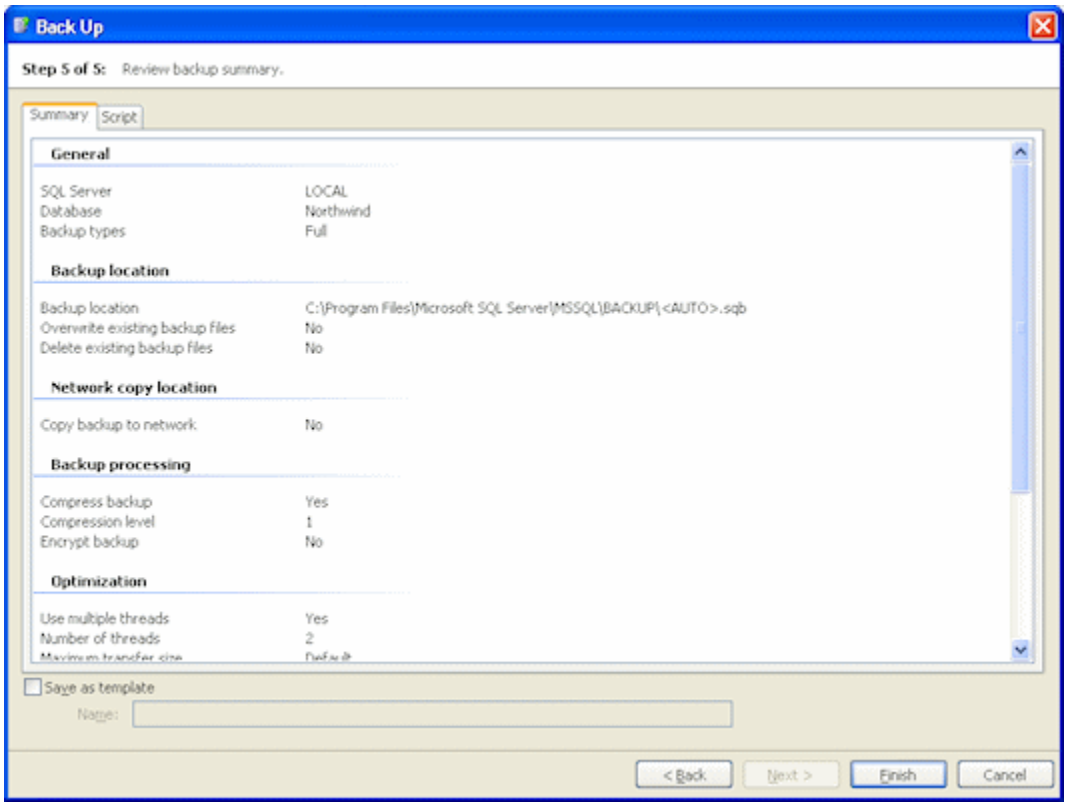
Select the **Verify backup files** check box if you want SQL Backup to run a RESTORE VERIFYONLY on the backup files when they have been created. When the backup process completes, the results of the verification are displayed in the Activity History. They are also saved in the SQL Backup completion log file.

Next: review summary

Creating backups - review summary

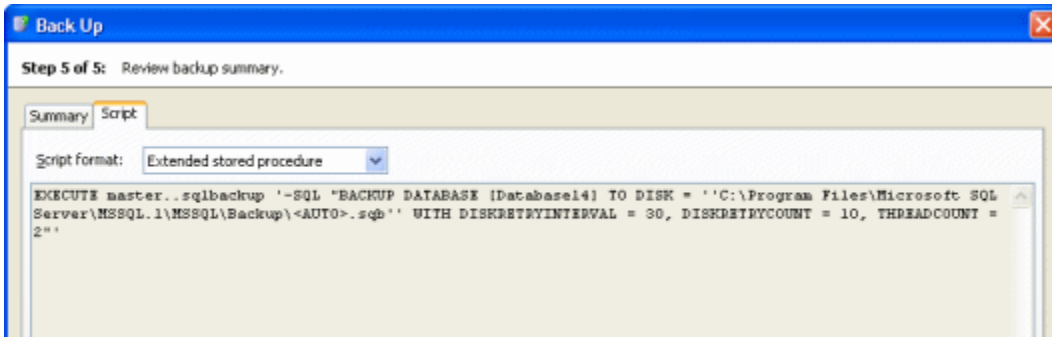
Creating backups > Specify SQL Server > Select backup type and database > File settings > Processing and encryption settings > **Review summary**

On step 5 of the wizard, review your backup settings and, if required, copy the script.



The **Summary** tab displays a simple report of the options you have set for the backup, so that you can check the details.

To see the script that will be used, click the **Script** tab.



You can then select the format in which to view the script:

- **Extended stored procedure** shows the script you can use to run the backup when you are connected to the SQL Server using an application such as Microsoft SQL Server Management Studio, or connectivity tools such as ADO, OLEDB, ODBC. For information about how you can use the SQL Backup extended stored procedure to back up databases, see Using the extended stored procedure.
- **Command line** shows the script you can use to run the backup from the command line. For more information, see Using the command line.

You can save the settings to a template. You can then use the template when you next create a backup or when you schedule backup jobs. For example, you may want to use the Back Up wizard to perform a trial run of the backup; when you are happy with the settings, you can open the template in the Schedule Backup Jobs wizard to create a scheduled job.

You can also use the template from the command line or extended stored procedure with the `-USE` parameter. For more information about the `-USE` parameter, see Scripting SQL Backup.

To save your settings to a template, select the **Save as template** check box and type a name for the template. The template will be saved when

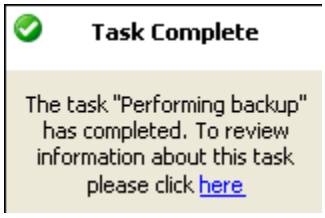
you click **Finish**. Note that if a template of the same name exists already, it will be overwritten.

When you have checked the settings, click **Finish**.

SQL Backup displays a message dialog box that shows the progress of the backup. Click **Hide** to minimize this dialog box and continue working. You can display the box again by clicking the arrow in the SQL Backup status bar.

The progress of the backup is also displayed in the In Progress tab.

When the backup process completes, if the message box is minimized, a pop-up message is displayed to inform you that the task is complete.

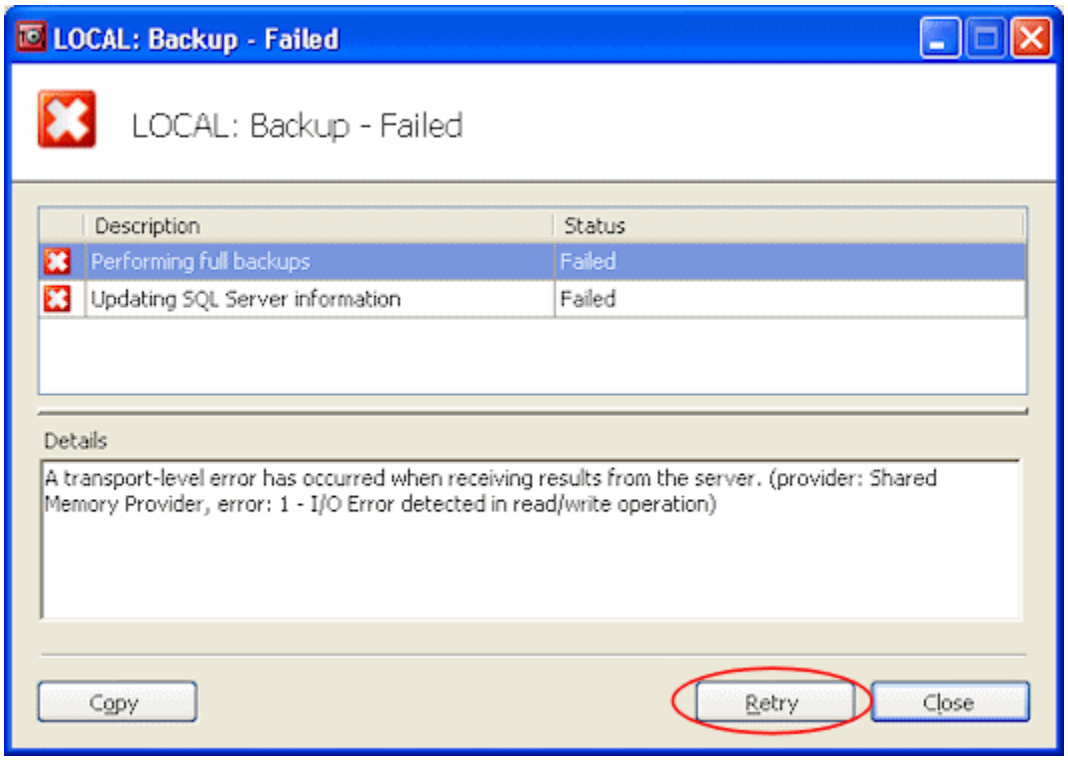


Information about the backup is then displayed in the Activity History. This information is also sent to a log file; you can view the contents of the log file by right-clicking the activity, and selecting **Show Log**. By default, log files are located in:

- *%PROGRAMDATA%\Red Gate\SQL Backup\Log\<instance name>* (Windows Vista, Windows 2008 and later), or
- *%ALLUSERSPROFILE%\ Application Data\Red Gate\SQL Backup\Log\<instance name>* (Windows XP and Windows 2003).

You can change this location in your log file options (see File management options).

If the backup process fails, you can optionally return to the final step of the wizard by clicking **Retry** in the message dialog. You can then change the settings in any of the wizard steps before starting the backup process again.



Scheduling backup jobs

SQL Backup provides the Schedule Backup Jobs wizard to guide you through the process of setting up a recurring backup job. The wizard creates a scheduled SQL Server Agent backup jobs that uses SQL Backup to perform the backup.

You can use the Schedule Backup Jobs wizard to generate a SQL script for backing up multiple databases. You can then view this script and use it as a basis for your own multiple database backup scripts.

The Schedule Backup Jobs wizard automatically sets up a job *step* using the `RAISERROR` command so that the SQL Server Agent reports an error if the SQL Backup command fails. To be notified of the error by email, select **Send email notification** in step 5 of the wizard. For information about SQL Server Agent jobs, steps, and the `RAISERROR` command, refer to your SQL Server documentation.

To start the Schedule Backup Jobs wizard, click  **Schedule Backup Jobs**. The Schedule Backup Jobs wizard comprises the following steps:

Step 1:	Select the SQL Server, and optionally choose a previously-saved template.
Step 2:	Select the types of backup you want to perform and the databases that you want to back up.
Step 3:	Define the details of the job schedules.
Step 4:	Specify the backup file locations (including any copies to hosted storage or a network location), and set options for existing backup files.
Step 5:	Specify compression, encryption, and optimization options, and actions required on completion.
Step 6:	Review the backup summary and script, and create the backup jobs.

You can edit a scheduled job by right-clicking the job in the Jobs tab, and then clicking **Edit**. The Edit Backup Job wizard is displayed. This is similar to the Schedule Backup Jobs wizard; the wizard contains the settings for the selected job. Note that you can edit only one job at a time.

Scheduling backup jobs with SQL Server Express Edition

With SQL Server Express Edition, you can use the SQL Backup GUI to take ad hoc backups and restores. However, you cannot set up scheduled backup jobs, scheduled restore jobs, or log shipping, because SQL Server Express Edition lacks the SQL Server Agent (which SQL Backup uses to schedule jobs).

You can still schedule backups using the Windows Task Scheduler (found in the Windows Control Panel) and SQL Backup's command line interface, `SQLBackupC.exe`:

1. Go through the Back Up wizard to create a template backup job for the database you want to schedule jobs for.
2. On step 6 of the wizard, choose the command line version of the script, and copy it.
3. In the Windows Control Panel, open **Task Scheduler** and create a new task.
4. On the **Actions** tab, create a new action which starts a program, and paste the command line script from the Back Up wizard into the **Program/script** box.
5. On the **Triggers** tab, create the schedule on which you want to run your backup job.
6. Click **OK** to create the scheduled task.

The backups will be listed in the **Activity History**, but the task will not be listed in the **Jobs** tab.

For more information about the command line interface, see Using the command line.

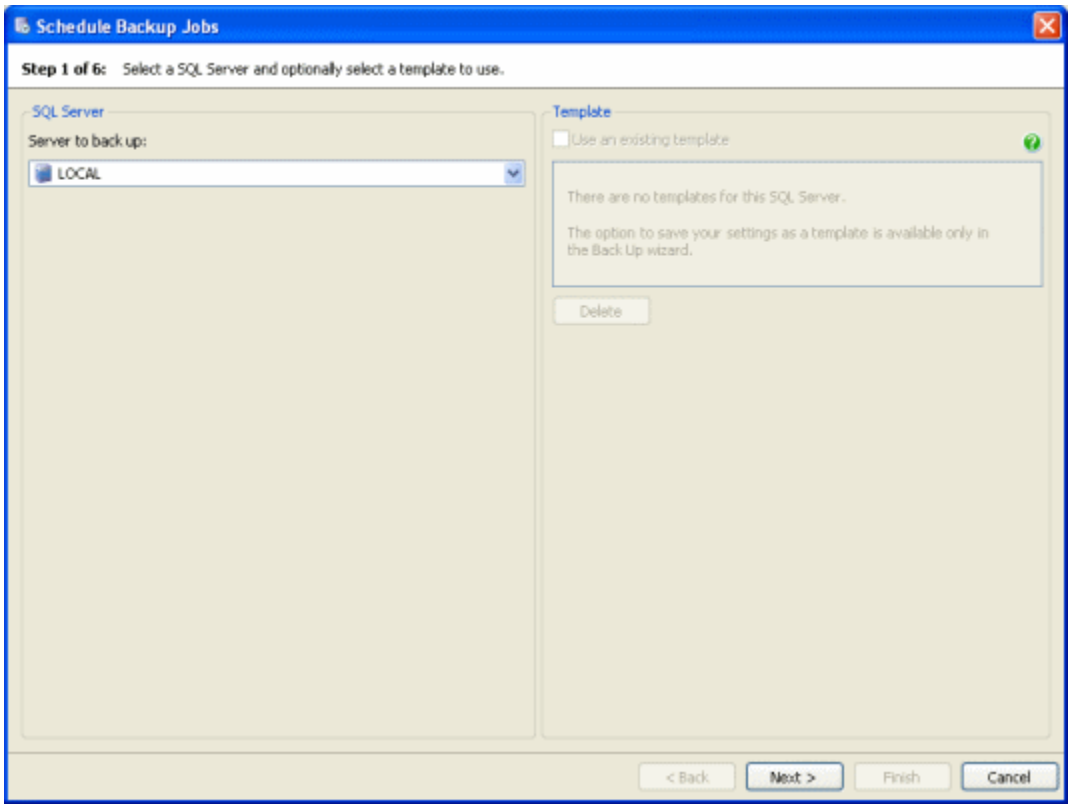
For more information about the Back Up wizard, see Creating backups.

Next: specify SQL Server

Scheduling backups - specify SQL Server

Scheduling backups > **Specify SQL Server** > Select backup type and database > Create backup schedules > File settings > Processing and encryption settings > Review summary

On step 1 of the wizard, select the SQL Server you want to back up.



In the **Server to back up** list, click the name of the SQL Server instance for the databases that you want to back up. SQL Servers that are not currently available (for example, because they are disconnected) are not displayed in the list.

> If the selected SQL Server does not have the server components installed, a warning is displayed and you must install the server components to proceed.

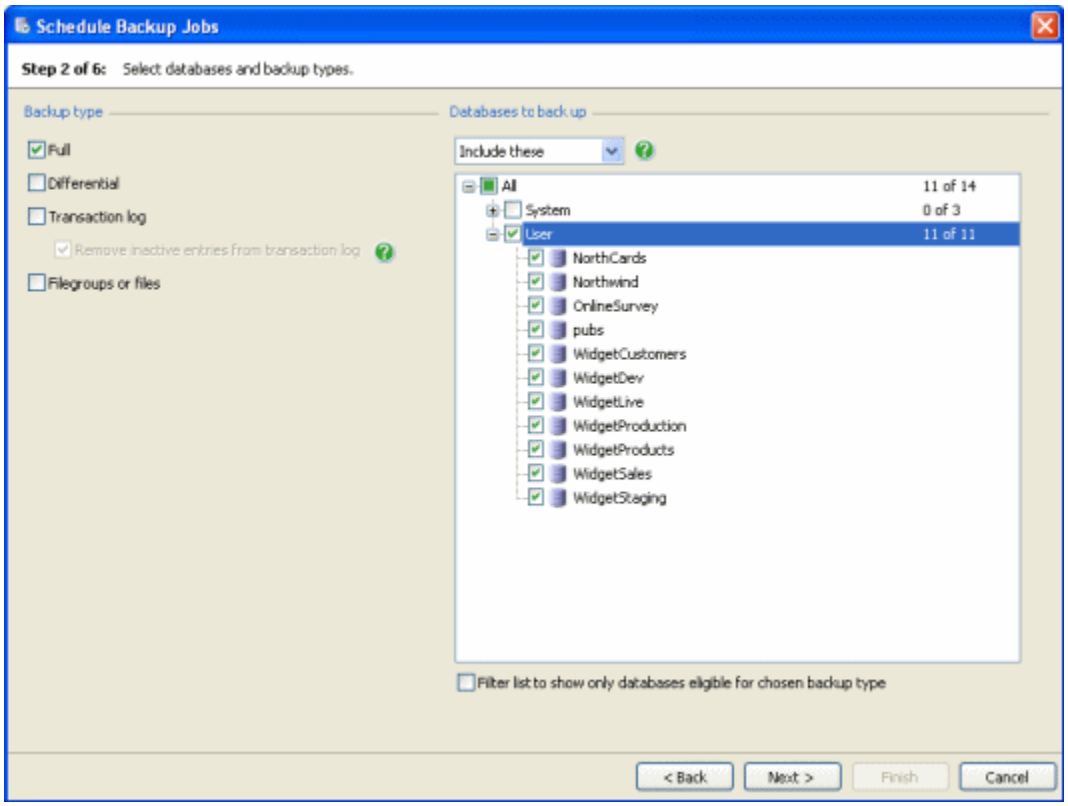
If you have previously saved any templates using the Back Up wizard, these are displayed in the right-hand pane.

To use a template, select the **Use an existing template** check box and click the name of the template in the list; when you click **Next**, the wizard will display those settings for you to edit if required. To delete a template, click the name of the template in the list and click **Delete**.

Next: select the backup type and database

Scheduling backups - select the backup type and database

On step 2 of the wizard, select the types of backup you want to perform and the databases that you want to back up.



Backup type

Select the type of backup you require:

- **Full** creates a complete copy of the selected database.
- **Differential** creates a partial copy of the selected database.Only the changes since the last full backup was made are copied. You must create a full backup of the database before you can perform a differential backup.
- **Transaction log** copies all the log records that have been written to the live transaction log since the first full or differential backup and the last transaction log backup.You can back up the live transaction log only if the database uses the FULL or BULK LOGGED recovery model, and you have previously created a full backup. To truncate the live transaction log, select the **Remove inactive entries from transaction log** check box. The space reserved for the transaction log can then be reused by new transactions.

  > Truncating the transaction log does not reduce the size of the log file on disk; for this, refer to the SQL Server documentation on Shrinking a File.

- **Filegroups or files** creates a backup of selected files and filegroups. For example, you may want to back up a file or filegroup when the database size makes a full database backup impractical. You can create a filegroup or files backup for only one database at a time. The database must use the FULL or BULK LOGGED recovery model. You will be prompted to select the files and filegroups when you click **N ext**.

Databases to back up

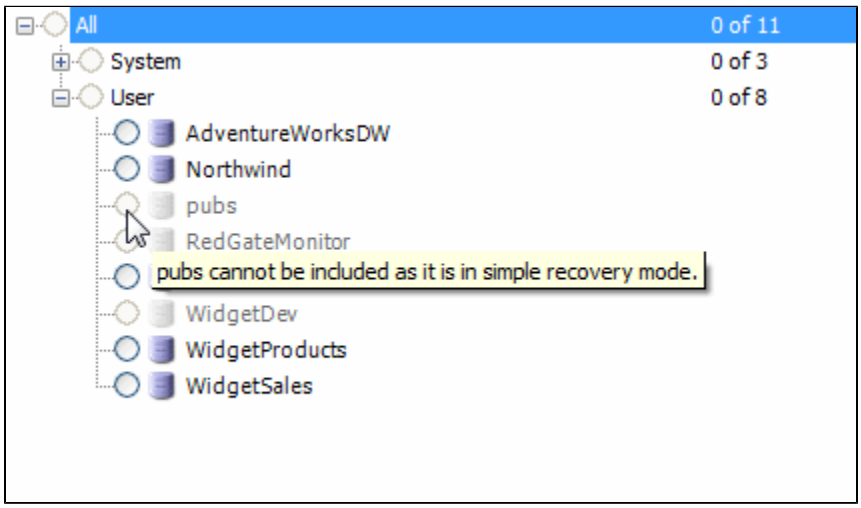
For full, differential, or transaction log backups:

- To select the databases that you want to back up, click **Include these**, and then select the check box next to each database that you want to back up.Selecting **System** or **User** selects all databases within that group. Included databases are marked with 
- To back up all databases except for individual specified databases, click **Exclude these** and select the check box next to each database that you do **not** want to back up.Excluded databases are marked with

For filegroups or files, you can select only one database; click the database in the list.

> Only those databases that can be backed up with the backup type you selected are available. For example, if you selected **Differential**, only databases that have already had a full backup are available.

By default, all databases are displayed in the list, and those that are not available for selection given the selected backup type are displayed in gray. If you are using **Exclude these**, databases that are not available are automatically excluded. To see the reason why a database is not available, move your mouse pointer over the database name. In the example below, the **Filegroups or files** backup type has been selected.



You can filter the list so that only available databases are displayed in the list by selecting the **Filter list to show only databases eligible for chosen backup type** check box.

Click **Next** when you have selected the databases.

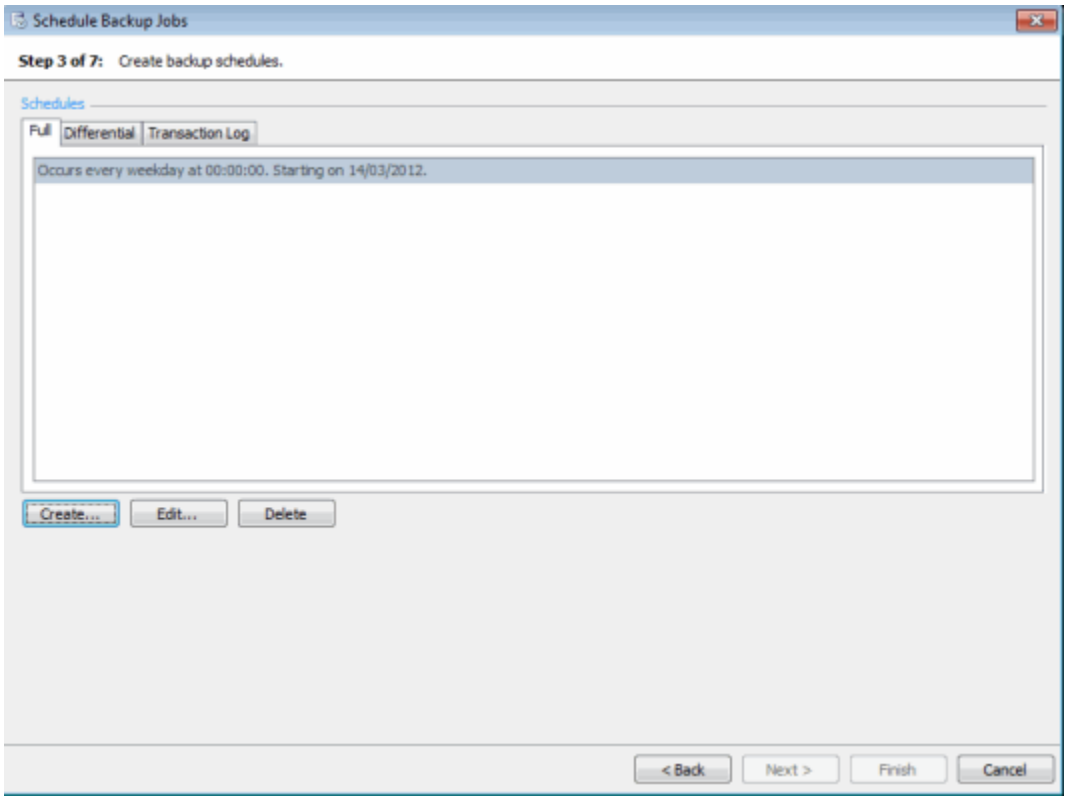
If you have chosen to back up files or filegroups, the wizard displays a page for you to select the files. The files and filegroups for the database you selected are displayed in a hierarchy. To select a file, select the appropriate check box. If you select the check box for a folder, all files and filegroups within that folder are automatically selected. When you have made your selection, click **Next**.

Next: create backup schedules

Scheduling backups - create backup schedules

Scheduling backups > Specify SQL Server > Select backup type and database > **Create backup schedules** > File settings > Processing and encryption settings > Review summary

On step 3 of the wizard, specify schedules for each of the backup types you have selected.



Each backup type (full, differential, transaction log or filegroup) must have one or more schedules. The schedules define the time and frequency of the backups. For example, a full backup could have two schedules; one to perform the backups daily, and another to perform them weekly.

To create a schedule, click the tab for the backup type for which you want to create the schedule, and click **Create**.

The **Schedule Editor** is displayed for you to define the schedule. Select:

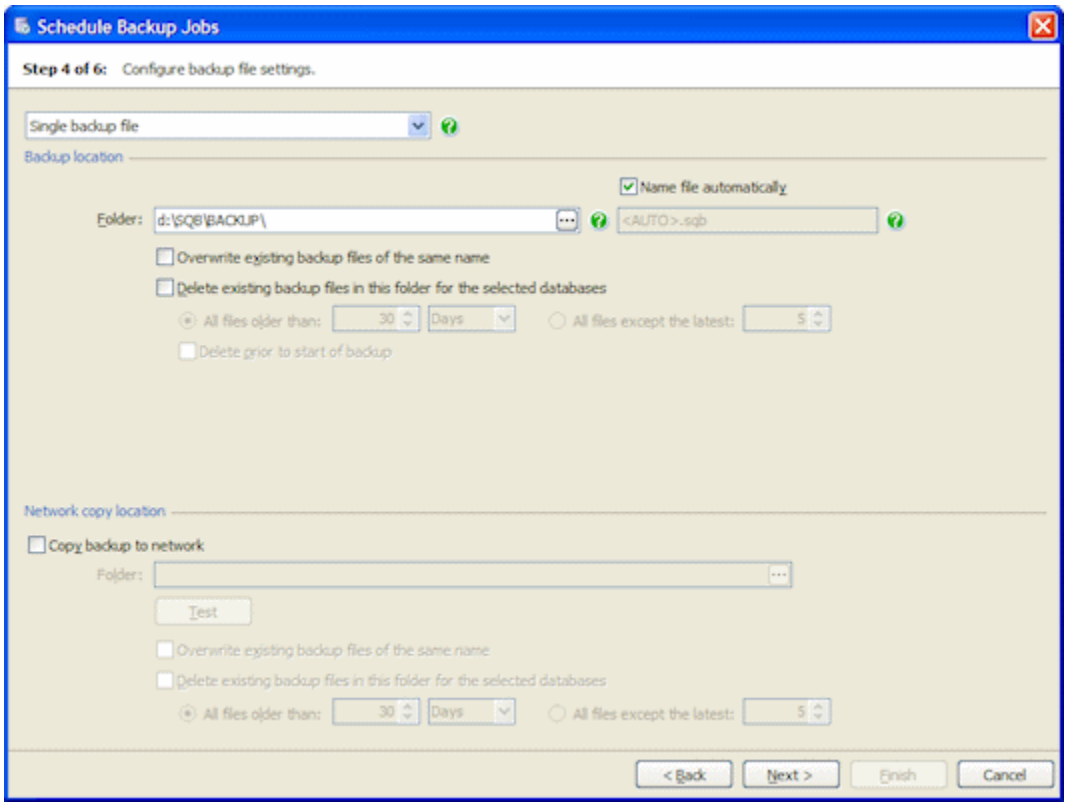
- **Recurring** to set a schedule that recurs at a specified frequency in days, weeks, or months.
- **Single occurrence** to set a single date and time for a one-off scheduled backup.
- **Occurs whenever SQL Server Agent starts** to start the backup job whenever the SQL Server Agent is started.
- **Occurs whenever the CPU becomes idle** to start the backup job whenever the CPU becomes idle.

Next: file settings

Scheduling backups - file settings

Scheduling backups > Specify SQL Server > Select backup type and database > Create backup schedules > **File settings** > Processing and encryption settings > Review summary

On step 4 of the wizard, specify the locations and file names for your backups, including any copies to a network location, and configure settings to manage existing backup files.



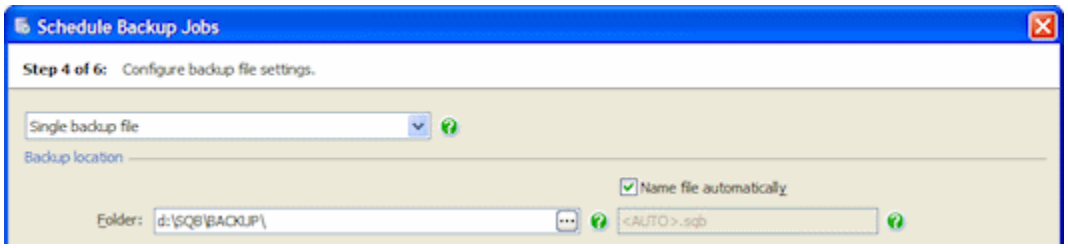
Choose:

- **Single backup file** to create each backup as a single file.
- **Single backup file, mirrored to second location** to create the backup as a single file and simultaneously make a duplicate of the backup during the backup process. The second backup is written in parallel to the first, so they will finish at the same time.This option is only available when you have selected a single database to back up. During the backup process, a warning is raised if any of the files cannot be written. However, the backup process continues as long as at least one specified backup file can be written. If none of the files can be written, an error is raised and the backup process is stopped.
- **Split backup into multiple files** to store each backup across a number of files.

Splitting the backup can speed up the backup process if backing up to a single file does not fully use the Input/Output capacity of your disks. For a single database backup, you can split (or 'stripe') your backup files across a number of disks. For multiple database backups, you can specify the number of files to split each database backup into (on the same disk). All backup files must be available when you restore the backup.

Backup location

For a **Single backup file**, if you have selected only one database to back up, a default folder and file name is displayed.



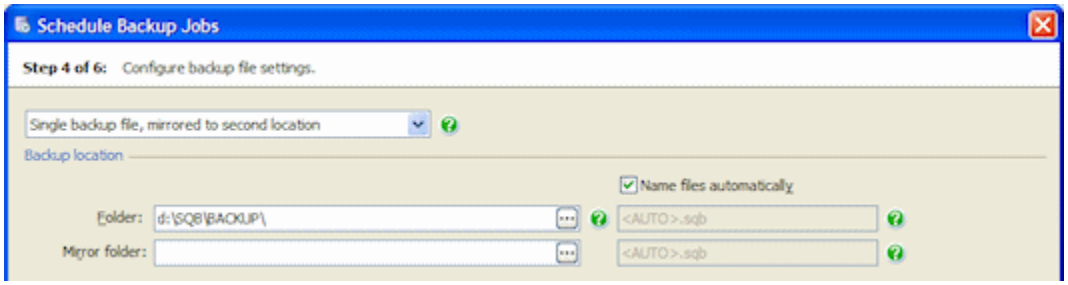
To change the file name, clear the **Name file automatically** check box, and type the new file name. To change the folder, type the new path or click

and specify the path using the folder browser. You can use tags in both the folder and file name. For more information, see File location tags.

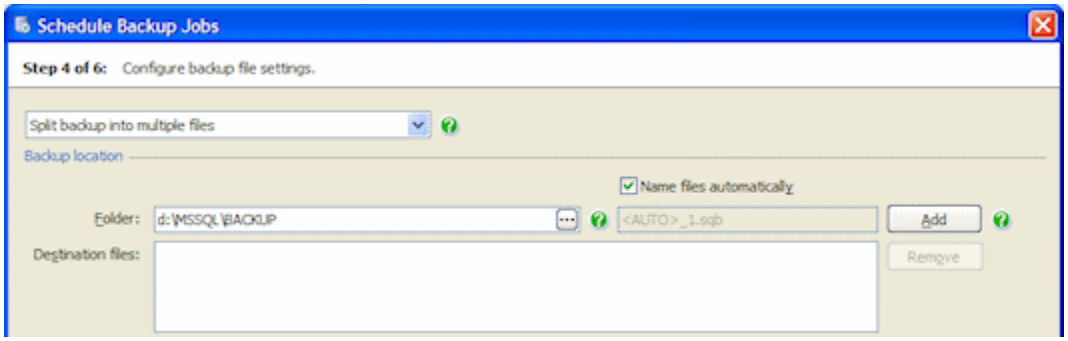
> The file path is relative to the selected SQL Server. For example, if you back up a database on a remote SQL Server instance called *ServerA* and you specify a local path such as *C:\Backups*, the backup files will be created on the C: drive on *ServerA*, not on the local machine.
>
> If you specify a network share as the backup file location, the SQL Backup Agent service 'log on' user must have *Full* permissions to access the location. Backing up directly to a network share can be slower than backing up locally and occasionally problematic. You are recommended to back up locally and use the **Copy backup to network** option instead. For more information, see Backing up and restoring on a network share.

If you have selected multiple databases, multiple backup types, or both, the backup for each database and backup type is created in a separate file. The file names will be generated automatically using the <AUTO> tag, which uses the default file name format specified in your File management options. If you have not set up a default file name format, SQL Backup uses the SQL Server instance's default format for file names. A default folder for these files is displayed; to change the folder, type the new path or click
and specify the folder using the folder browser. If you want the backup file to be created in a different folder for each database, in the folder browser select **Create subfolder for each database**.

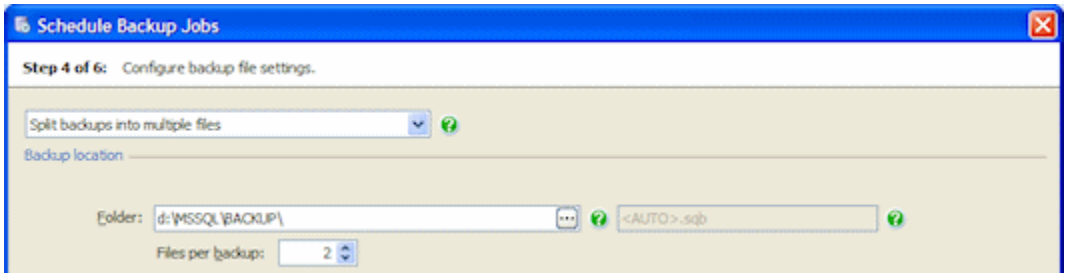
For a **Single backup file, mirrored to second location**, specify the backup file as described above, and additionally specify the folder and file name for the mirrored backups.



For **Split backup into multiple files**, if you have selected a single database and a single backup type, you must specify at least two destination files. Type or select each folder and file name as described above, and click **Add** to add it to the list.



If you have selected multiple databases, multiple backup types, or both, you can specify only one folder for the split backup files; the files will be named automatically.



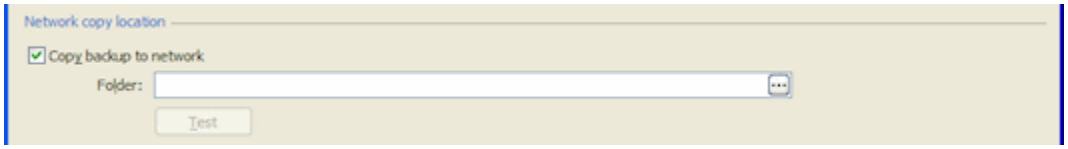
In **Files per backup**, type or select the number of files into which you want each backup to be split (maximum 32).

Network copy location

If you want a copy of the backup files to be created on a network share when the backup has completed, select **Copy backup to network**. Type the network share details in the **Folder** box or click

to use the folder browser.

The folder browser displays the local file system for the SQL Server you are backing up. Select the server you want to copy the backup to from the drop-down list, or click **Add Server** and type the server name or IP address. Other servers will only be visible to the local server if it has the appropriate permissions to write to or read from them. The name of the local server you are connected to and your user name are displayed above the **Server** list. This information may explain why some servers cannot be browsed.



If you typed the network share details, click **Test** to check that the "log on" user for the SQL Backup Agent service on the source SQL Server has permission to create files on the network share. If it does not have the appropriate permissions, an error message is displayed next to the button; if you choose to keep the network share details, a warning

is displayed in the Activity History when the backup has completed. If you browsed to the network share, this check is performed automatically.

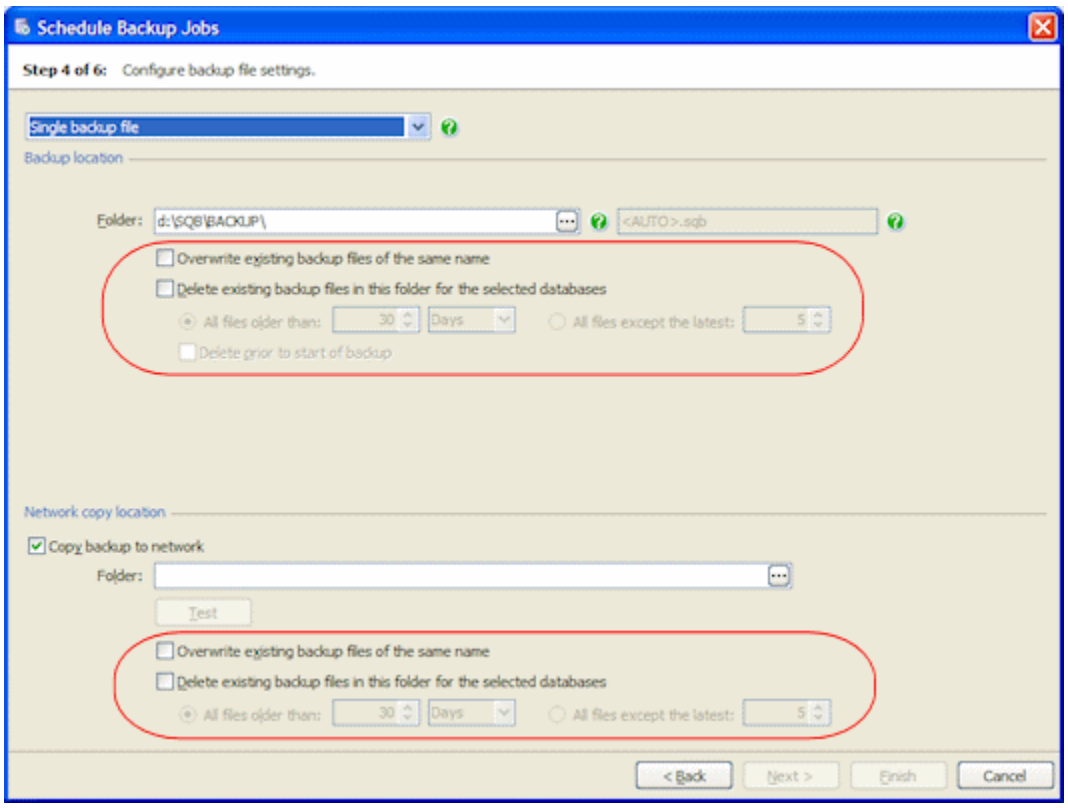
For more information about using network shares with SQL Backup, see Backing up and restoring on a network share. For details about setting up permissions to use a network share, see Permissions.

You may experience problems when you back up to network shares if your SQL Server is unable to write large data blocks (over 2 MB) to the network share. You can use the `MAXDATABLOCK` keyword with the BACKUP command to limit the data block size.

Managing backup files

SQL Backup provides a number of settings to automate the management of your existing backup files. You can specify these settings separately for the initial backup location, and the network copy location.

> If you specify a *network* path in the backup location folder, any network copy location options you specify will be applied to files at this path. Similarly, if you specify a *local* path in the network copy location folder, any backup location folder options will be applied to the copied files in the (local) network copy location.



Select the **Overwrite existing backup files of the same name** check box if you want to overwrite any existing files with the share/path name.

> If a file of the same name exists already and you have not chosen to overwrite it, the backup or network copy will fail.

Select the **Delete existing backup files in this folder for the selected databases** check box if you want SQL Backup to delete backups of the same type for the selected databases from the initial backup or network copy destination folders. You can restrict deletion of existing backup files by age (**All files older than**) or number (**All files except the latest**).

By default, files are deleted when the backup process or network copy has completed. If the backup fails, the files are not deleted. To delete the files before the backup is created, select the **Delete files prior to start of backup** check box. For example, you may want to do this to create space for the new backup files. However, note that if the backup fails, the old files will have been deleted; you are therefore recommended to select this check box only if you have a copy of the existing backups. This option is available for the initial backup location only; existing network copies of files are always deleted after the copying process has completed.

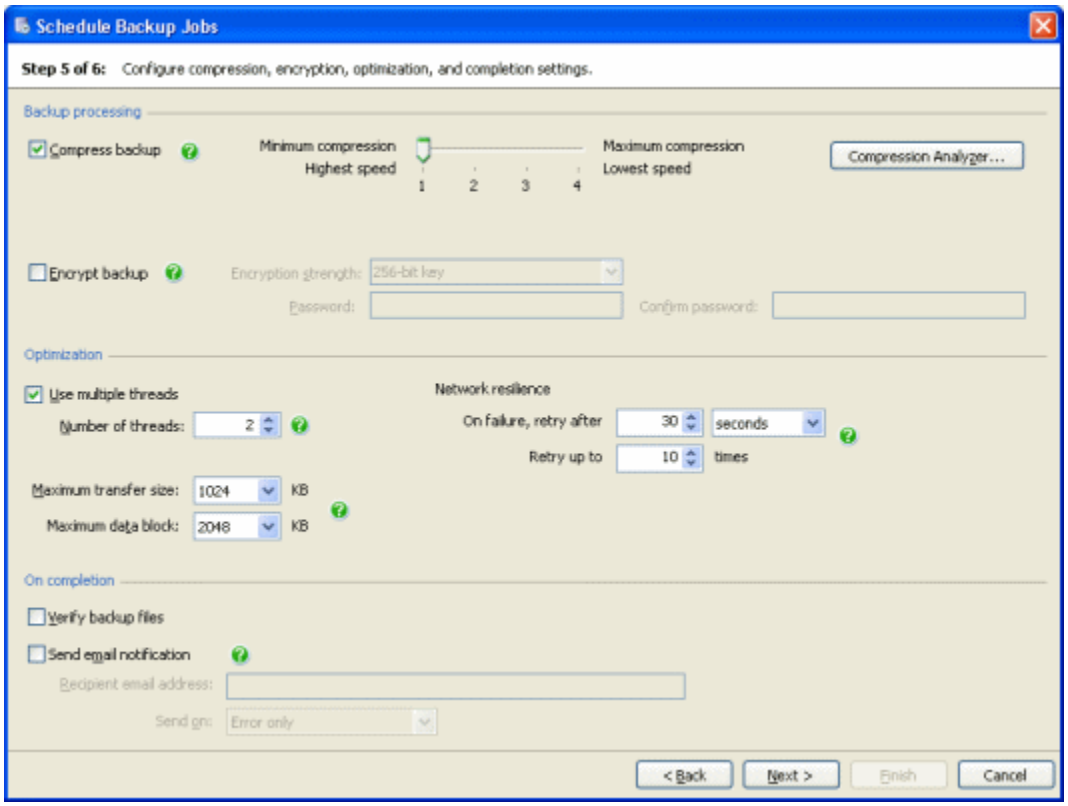
> If SQL Backup cannot list the contents of the folder that contains the files to be deleted, it cannot delete the files. Ensure that the user account from which you are running SQL Backup has permissions to list the folder contents.

Next: processing and encryption settings

Scheduling backups - processing and encryption settings

Scheduling backups > Specify SQL Server > Select backup type and database > Create backup schedules > File settings > **Processing and encryption settings** > Review summary

On step 5 of the wizard, specify the compression, encryption, and optimization options for your backups.



Backup processing

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

> If SQL Backup Lite is installed on the SQL Server and you choose to compress your backup, compression level 1 is selected and you cannot change it.

For more information about compression levels, see Compression levels.

For full database backups, you can click **Compression Analyzer** to perform a test on the databases to check which compression level will produce the best result for your requirements. For more information about the Compression Analyzer, see Compression analyzer.

To encrypt the backup, select the **Encrypt backup** check box, then type a password for the backup in **Password**, and again in **Confirm password**.

Choose 128-bit or 256-bit encryption. **You must remember your password**; if you do not, you will not be able to access the encrypted backup.

Optimization

If you are backing up to a **Single file** or **Single backup file, mirrored to second location**, SQL Backup can use multiple threads to create the backups. This 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. For more information about using multiple threads, see Optimizing backup speed.

Maximum transfer size specifies the maximum size of each block of memory to be used when SQL Backup stores backup data. The default value is 1024 KB; you may want to change this value if a SQL Server reports that it has insufficient memory to service requests from SQL Backup.

Maximum data block specifies the maximum size of data blocks to be used when SQL Backup stores backup data. The default value is 4096 KB.

Note that if you want to specify the network transfer packet size, change the connection properties for the SQL Server. For more information, see

Connection Properties.

The **Network resilience** settings control retry behavior following a read/write error whenever SQL Backup:

- writes a backup file to disk
- copies a backup file to another disk location

Read/write errors are most likely to occur when SQL Backup is processing files through a network connection.

In most circumstances, you can leave **On failure, retry after** and **Retry up to** (n) **times** at their default values (30 seconds, and 10 times). To disable retries completely, specify **Retry up to** 0 **times**.

On completion

If you want to receive an email with a copy of the completion log, select the **Send email notification** check box, and enter the recipient's email address. This option is available only if you have entered your email settings. For more information about setting up email notification, see Email settings.

To send the log to multiple email addresses, type each address separated with a semi-colon (**;**). For example:

dba01@myco.com**;**dba02@myco.com

By default, email notifications are sent only when there are errors during the backup process (**Error only**). If you want to receive an email when an error or warning occurs, select **Error or warning**; to always receive an email on completion of the backup process (on success or failure), select **Any outcome**.

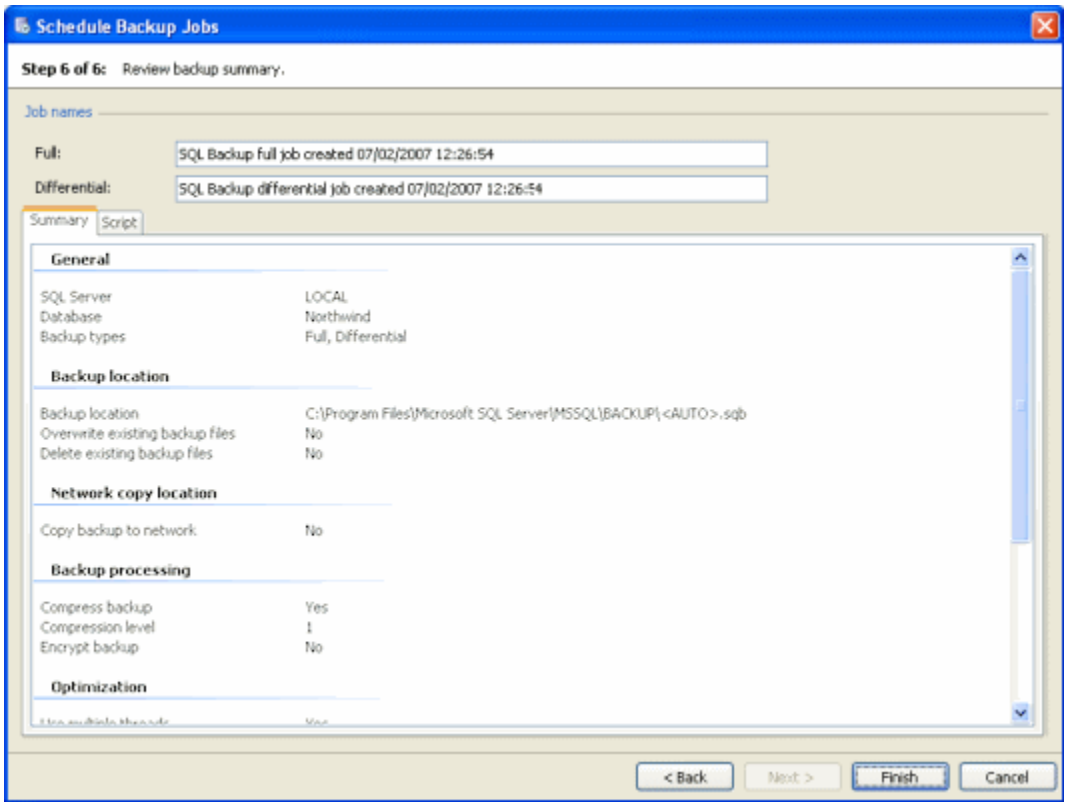
> If SQL Backup Lite is installed on the SQL Server, you cannot use email notification.

Select the **Verify backup files** check box if you want SQL Backup to run a RESTORE VERIFYONLY on the backup files when they have been created. When the backup process completes, the results of the verification are displayed in the Activity History. They are also saved in the SQL Backup completion log file.

Next: review summary

Scheduling backups - review summary

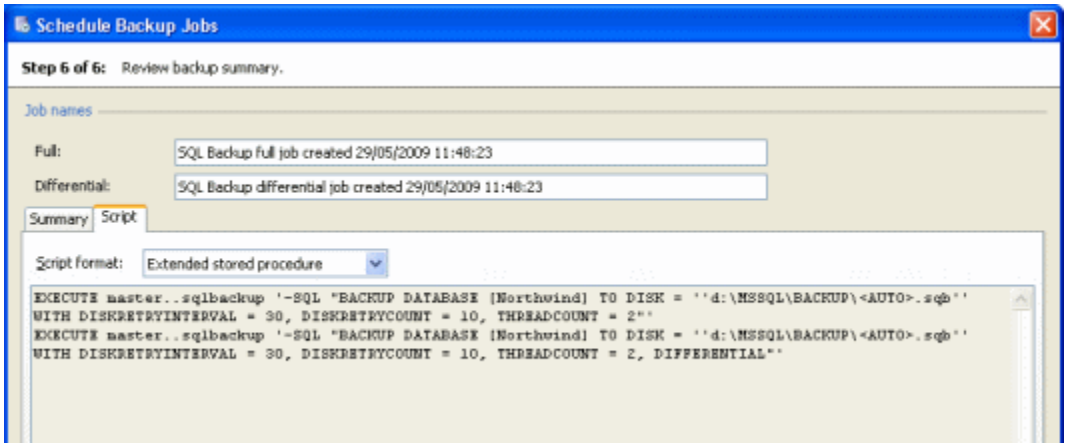
On step 6 of the wizard, review your backup settings and, if required, copy the script.



The name of each backup job that will be created is displayed. You can edit the backup job names, if required, by typing in the appropriate box.

The **Summary** tab displays a simple report of the options you have set for the backup jobs, so that you can check the details.

To see the scripts that will be used, click the **Script** tab.



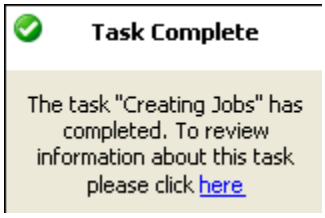
You can then select the format in which to view the scripts:

- **Extended stored procedure** shows the scripts you can use to run the backups when you are connected to the SQL Server using an application such as Microsoft SQL Server Management Studio, or connectivity tools such as ADO, OLEDB, ODBC.For information about how you can use the SQL Backup extended stored procedure to back up databases, see Using the extended stored procedure.
- **Command line** shows the scripts you can use to run the backups from the command line. For more information, see Using the command line.

When you have checked the settings, click **Finish**.

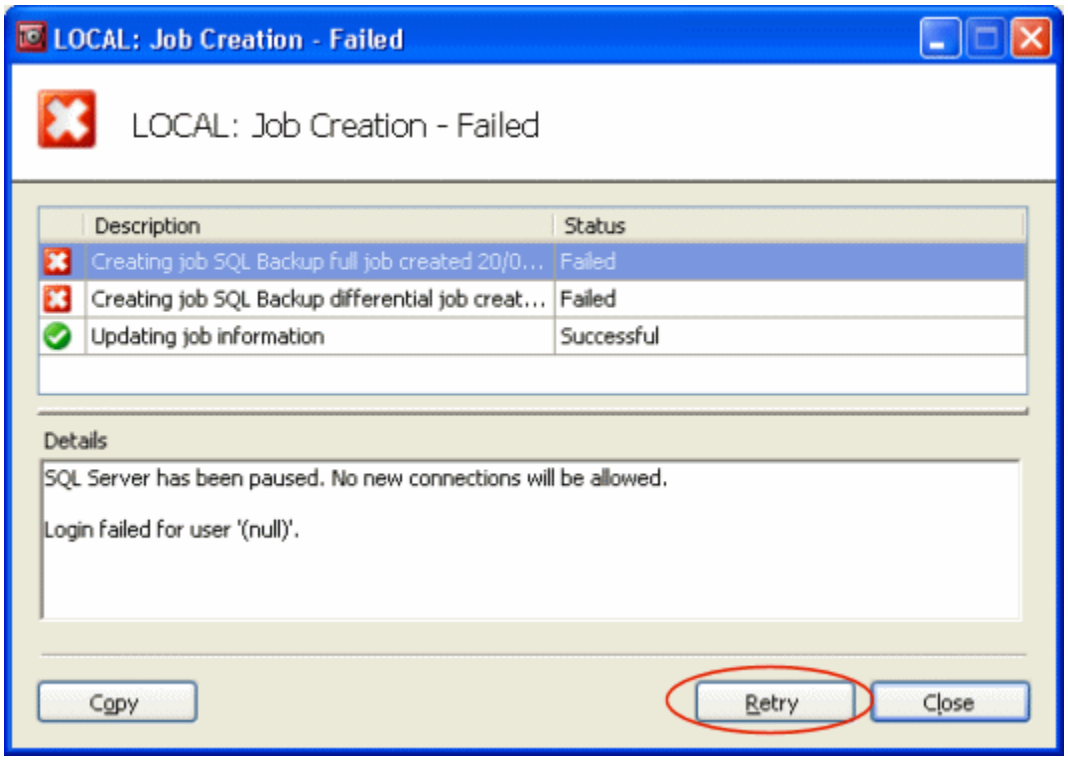
SQL Backup displays a message dialog box that shows the progress of the job creation. Click **Hide** to minimize this dialog box and continue working. You can display the box again by clicking the arrow in the SQL Backup status bar.

When the job creation process completes, if the message box is minimized, a pop-up message is displayed to inform you that the task is complete.



The created jobs are then displayed in the Jobs tab.

If the job creation process fails, you can optionally return to the final step of the wizard by clicking **Retry** in the message dialog. You can then change the settings in any of the wizard steps before starting the job creation process again.



Backing up all databases on an instance

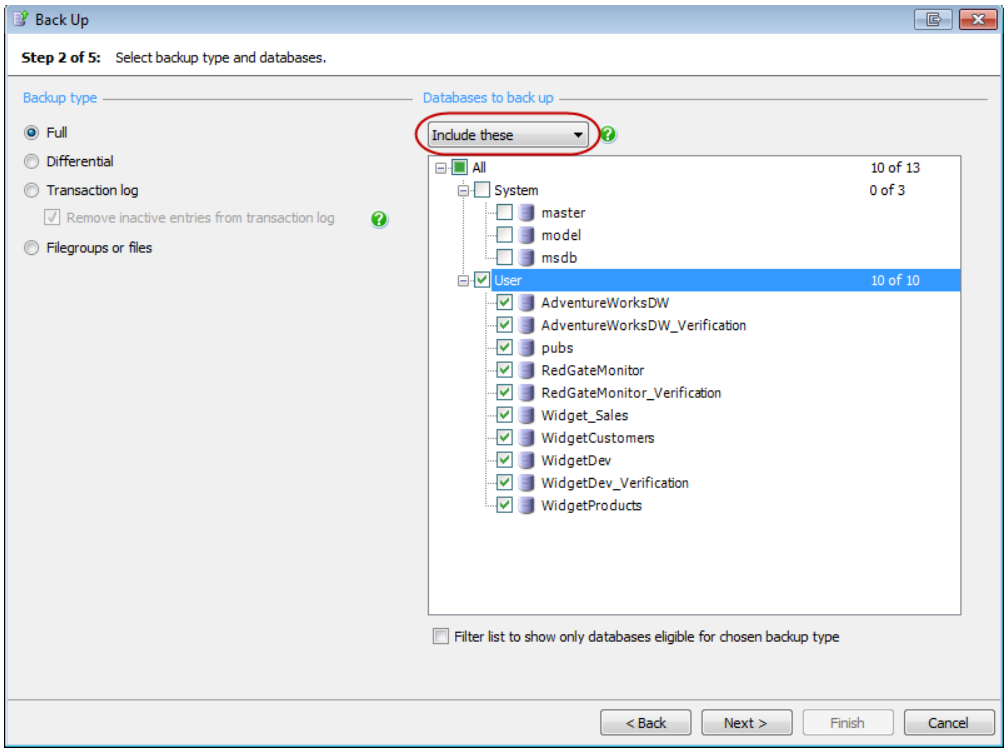
You may want to back up all the databases on an instance, or all but one or two. Both the backup wizards and the BACKUP command syntax include options to back up all databases on an instance and exclude particular databases.

Using the backup wizards

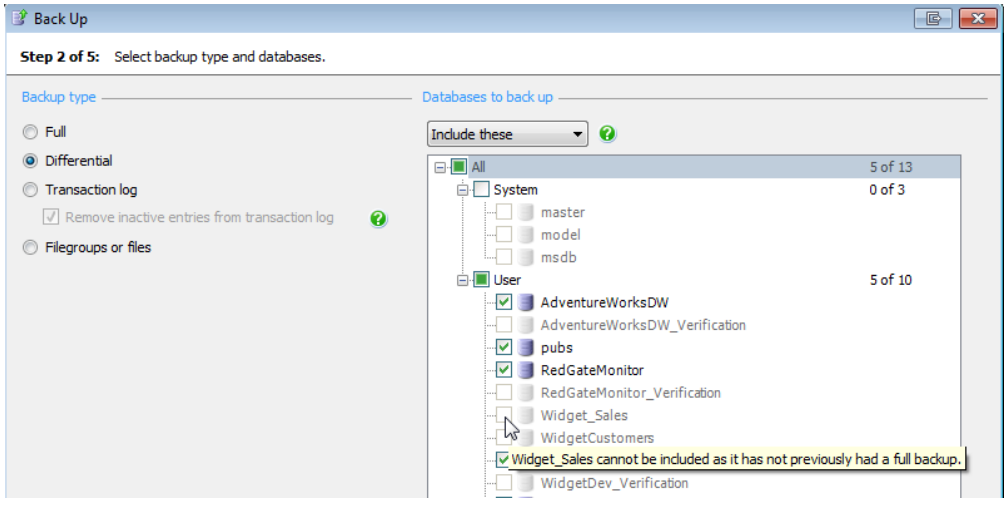
To take a single backup, open the Back Up wizard. To create a scheduled backup job, open the Schedule Backup Jobs wizard.

On step 1 of the wizard, select the SQL Server instance.

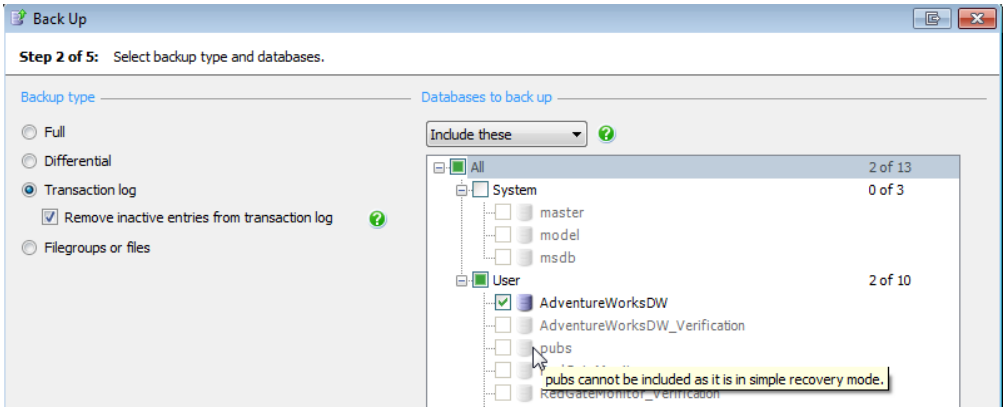
On step 2 of the wizard, select the type of backup (full, differential or transaction log), then select the databases you want to back up. To back up all databases, or all system or user databases, keep the default option of **Include these** and then select **All**, **System** or **User** as appropriate. The selected databases are marked with

If you select differential backups, only databases that have previously had a full backup taken are available:

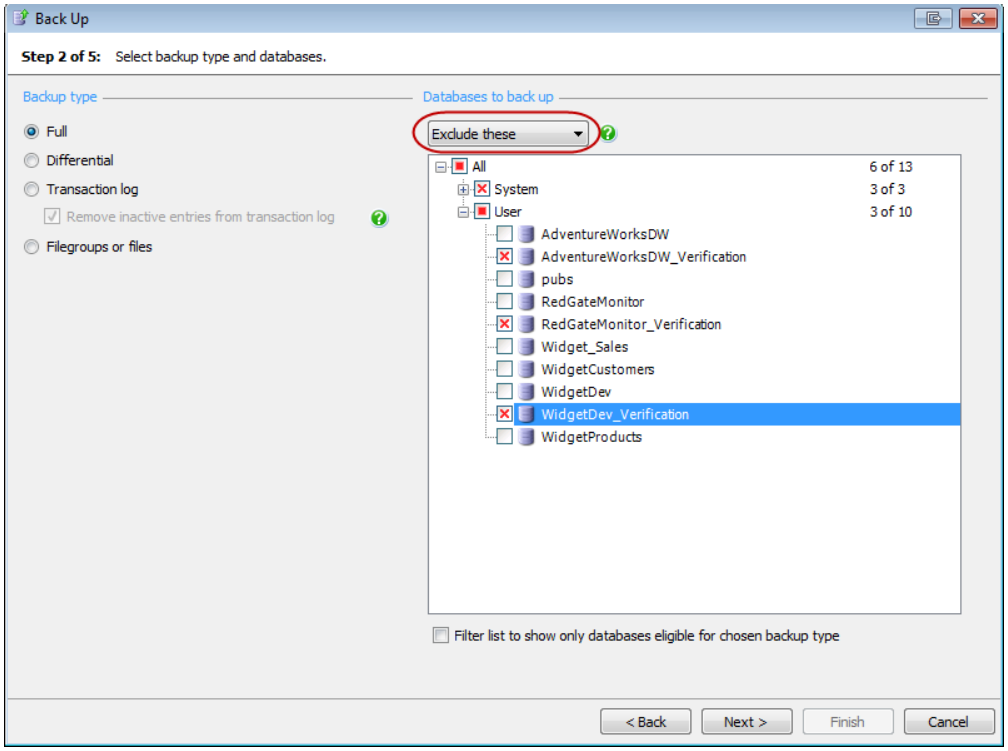


If you select transaction log backups, only databases that are in full or bulk-logged recovery mode and have previously had a full backup taken are available:

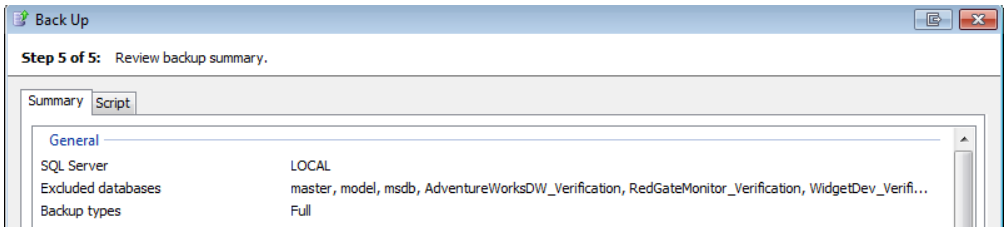


To back up all databases on the instance except for specified databases, select **Exclude these** from the drop-down list. Then select the databases that you do not want to back up. The excluded databases are marked with

All other databases that are online and operational when the job runs will be backed up.



Proceed through the rest of the wizard as normal. On the final step of the wizard you can view a summary of the options you have selected, including the databases you have included or excluded.



If you have included all databases or have selected **Exclude these**, any databases which you subsequently add to the instance will also be included in the backup job the next time it runs. This is because the backup command generated by the wizard uses the wildcard character * to select all databases on the instance.

If you add a database to an instance and do not want to include it in the job, you will need to edit the job to exclude the database. From the Jobs tab, double-click the job to open the Edit Backup Job wizard.

If you have created a differential or transaction log backup job and later add a database to the instance that you do not exclude from the job, you

should take a full backup of that database. If a full backup does not exist, the differential or transaction log backup for that database may fail with SQL Backup VDI error 1010 and SQL error 3059 (differential backups) or 4214 (transaction log backups), depending on the version of SQL Server you are using. This is because a full backup is required to restore a differential or transaction log backup. Refer to your SQL Server documentation for more information.

Using the BACKUP DATABASES command with the extended stored procedure or command line

In the following examples, the extended stored procedure version of the command is given first (beginning `EXECUTE master..sqlbackup`) followed by the command line version (beginning `SQLBackupC.exe`). For more information, see Scripting SQL Backup.

To back up all databases on the instance you are connected to, use `ALL` or a single wildcard character (*). For example:

```
EXECUTE master..sqlbackup '-SQL "BACKUP ALL DATABASES TO DISK =
''C:\Backups\<AUTO>''"'
```

```
SQLBackupC.exe -SQL "BACKUP ALL DATABASES TO DISK = 'C:\Backups\<AUTO>'"
```

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASES [*] TO DISK =
''C:\Backups\<AUTO>''"'
```

```
SQLBackupC.exe -SQL "BACKUP DATABASES [*] TO DISK = 'C:\Backups\<AUTO>'"
```

To back up all system databases (master, model and msdb) on the instance you are connected to, use `SYSTEM`. For example:

```
EXECUTE master..sqlbackup '-SQL "BACKUP SYSTEM DATABASES TO DISK =
''C:\Backups\<AUTO>''"'
```

```
SQLBackupC.exe -SQL "BACKUP SYSTEM DATABASES TO DISK = 'C:\Backups\<AUTO>'"
```

To back up all online and operational user databases on the instance you are connected to, use `USER`. For example:

```
EXECUTE master..sqlbackup '-SQL "BACKUP USER DATABASES TO DISK =
''C:\Backups\<AUTO>''"'
```

```
SQLBackupC.exe -SQL "BACKUP USER DATABASES TO DISK = 'C:\Backups\<AUTO>'"
```

Using EXCLUDE

To back up all online and operational databases (both system and user) on the instance you are connected to, apart from certain specified databases, use `EXCLUDE`. For example:

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASES EXCLUDE master, model, Testing TO
DISK = ''C:\Backups\<AUTO>''"'
```

```
SQLBackupC.exe -SQL "BACKUP DATABASES EXCLUDE master, model, Testing TO DISK =
'C:\Backups\<AUTO>'"
```

You can combine `SYSTEM` or `USER` with `EXCLUDE`. For example, the following command backs up all online and operational user databases other than *AdventureWorksDW* and *pubs*:

```
EXECUTE master..sqlbackup '-SQL "BACKUP USER DATABASES EXCLUDE AdventureWorksDW, pubs
TO DISK = ''C:\Backups\<AUTO>'' "'
```

```
SQLBackupC.exe -SQL "BACKUP USER DATABASES EXCLUDE AdventureWorksDW, pubs TO DISK =
'C:\Backups\<AUTO>'"
```

Using WITH DIFFERENTIAL

Include WITH DIFFERENTIAL to take a differential backup of the specified databases. For example:

```
EXECUTE master..sqlbackup '-SQL "BACKUP USER DATABASES EXCLUDE SalesArchive TO DISK =
''C:\Backups\<AUTO>'' WITH DIFFERENTIAL"'
```

```
SQLBackupC.exe -SQL "BACKUP USER DATABASES EXCLUDE SalesArchive TO DISK =
'C:\Backups\<AUTO>' WITH DIFFERENTIAL"
```

A full database backup is required to restore a differential backup. If a full backup is not taken before taking a differential backup, the differential backup may fail with SQL Backup error VDI 1010 and SQL error 3035. You may encounter this error if you set up a differential backup job using *, `ALL`, `SYSTEM` or `USER` and later add a database to the instance and do not take a full backup before the job runs. SQL Backup Pro 7.1 and later include a new keyword, `FULLIFREQUIRED`, which can be used to avoid this error. For more information, refer to the documentation for SQL Backup 7.1 or later.

Using the BACKUP LOGS command with the extended stored procedure or command line

In the following examples, the extended stored procedure version of the command is given first (beginning `EXECUTE master..sqlbackup`) followed by the command line version (beginning `SQLBackupC.exe`). For more information, see Scripting SQL Backup.

To back up transaction logs of all online and operational databases using the `FULL` or `BULK-LOGGED` recovery model on the instance you are connected to, use a single wildcard character (*). For example:

```
EXECUTE master..sqlbackup '-SQL "BACKUP LOGS [*] TO DISK = ''C:\Backups\<AUTO>'' "'
```

```
SQLBackupC.exe -SQL "BACKUP LOGS [*] TO DISK = 'C:\Backups\<AUTO>'"
```

Using EXCLUDE

To back up the transaction logs of all online and operational databases using the FULL or BULK-LOGGED recovery model on the instance you are connected to, apart from certain specified databases, use `EXCLUDE`. For example:

```
EXECUTE master..sqlbackup '-SQL "BACKUP LOGS EXCLUDE SalesArchive TO DISK =
''C:\Backups\<AUTO>''"'
```

```
SQLBackupC.exe -SQL "BACKUP LOGS EXCLUDE SalesArchive TO DISK = 'C:\Backups\<AUTO>'"
```

A full database backup is required to restore transaction log backups. If a full backup has not been taken before taking transaction log backups, the backups may fail with SQL Backup error VDI 1010 and SQL error 4214. You may encounter this error if you set up a transaction log backup job using `BACKUP LOGS '*'` or `BACKUP LOGS EXCLUDE '<database name>'` later add a database to the instance or change a database's recovery model from simple to full or bulk-logged, and do not take a full backup before the job runs. SQL Backup Pro 7.1 and later include a new keyword, `FULLIFREQUIRED`, which can be used to avoid this error. For more information, refer to the documentation for SQL Backup 7.1 or later.

Restoring

These pages explain how to restore backups created with SQL Backup:

- Restoring backups
- Restoring multiple backups
- Restoring the master database

Restoring backups

SQL Backup provides a wizard to guide you through the process of Restoring Backups. To start the Restore wizard, click  **Restore**, or in the Activity History, right click the backup and click **Restore**.

The Restore wizard comprises the following steps:

Step 1:	Select the SQL Server you want to restore to and the backup files to use. If necessary, enter the passwords for encrypted backups.
Step 2:	Specify the destination database and the location of the database files.
Step 3:	Configure your settings for restoring the backups.
Step 4:	Review the restore summary and script, and start the restore process.

Restoring individual database objects

If you want to restore individual objects from a SQL Backup .sqb file, use SQL Object Level Recovery Pro. This application, included with SQL Backup, can recover individual objects to a database, with potentially large savings in time and disk space. For more information, see Object level recovery.

Next: select backups

Restoring backups - select backups

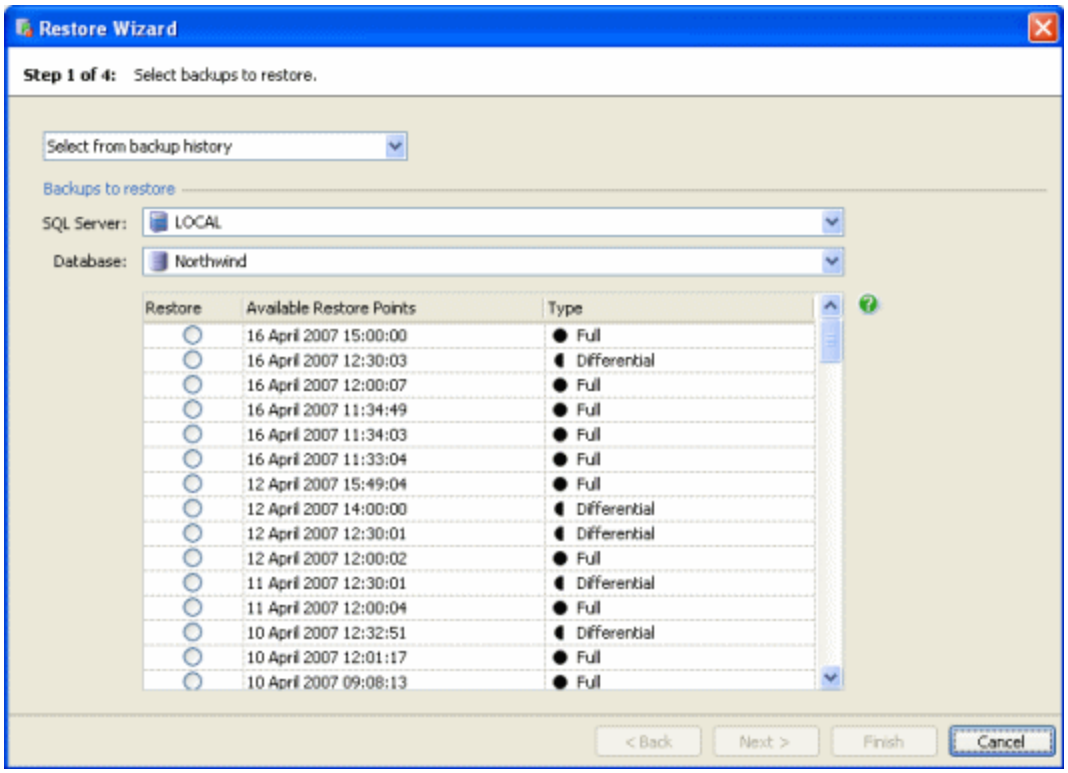
On step 1 of the wizard, select the SQL Server you want to restore to and the backup files to use.

Choose:

- **Select from backup history** to select from a list of backups that were created on the destination SQL Server (the SQL Server to which you want to restore the backup).
- **Browse for backup files to restore** to select individual backup files using the file browser. This is useful if the destination SQL Server is different from the SQL Server on which you created the backup, or if you are restoring from a copy of the backup files.

Select from backup history

Select the **SQL Server** and **Database** that was backed up.

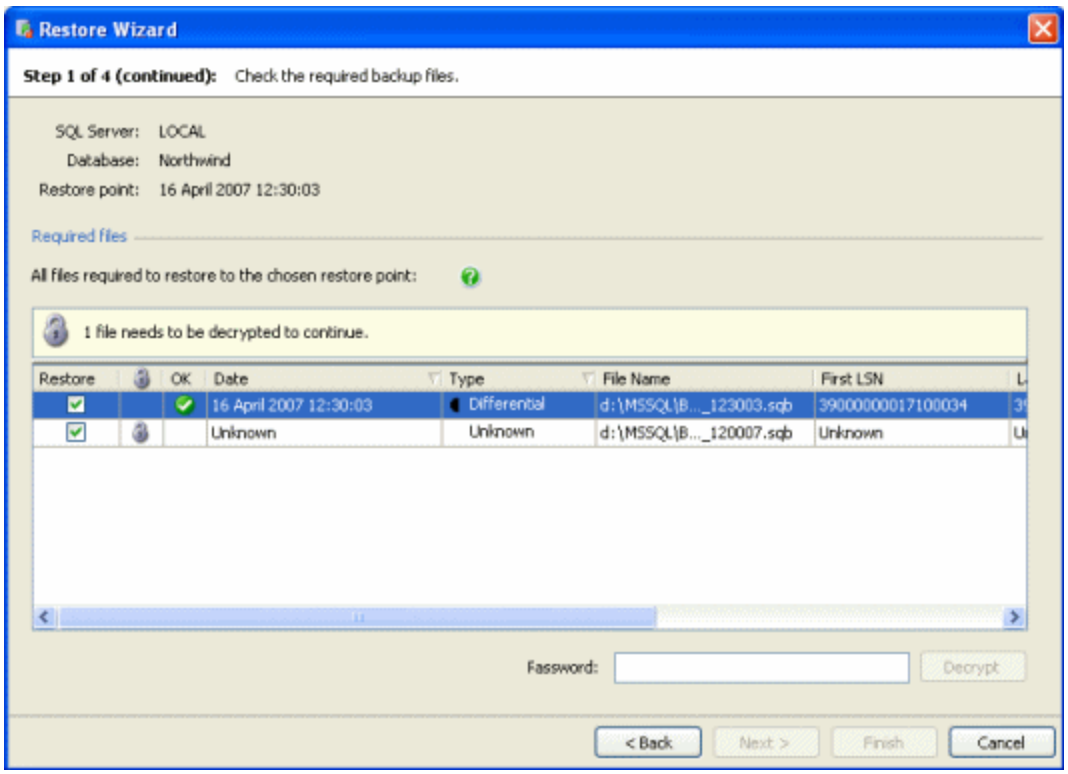


The backups for the selected database are listed. Select the backup you want to restore.

> Backups that were created using SQL Backup version 4 are not listed; you can use **Browse for backup files to restore** to select these files.

If you select a files or filegroups backup, you can select individual files or filegroups to restore at a later stage in the wizard.

When you click **Next**, SQL Backup retrieves the files for the selected backup. If you selected a transaction log backup or a differential backup from a backup set, SQL Backup automatically retrieves files for all backups in the set. For example, if you selected a transaction log backup, SQL Backup retrieves all transaction log backups since the corresponding full backup, and the full backup itself. You may see displayed while SQL Backup searches for the appropriate files.



The files are listed with the most recent first. You can re-order the list by clicking the column headers as required; this will not affect the order in which they will be restored.


is displayed next to any encrypted files. Type the password in **Password**, and click **Decrypt**. For all files that use the password you entered,

is displayed. You must enter the passwords for all selected files before you click **Next**.

If there are any problems with a file,

is displayed next to the file. For example, SQL Backup checks that all the files exist and are not corrupted. To see details of the error, move your mouse pointer over the icon.

By default, all the files that SQL Backup has identified as part of the backup set are selected to be restored. If required, you can clear the **Restore** check box for a file. For example, you may want to do this if you know that the full backup has already been restored to a database. However, note that the restore process may fail if you do not restore all required files.

Browse for backup files to restore

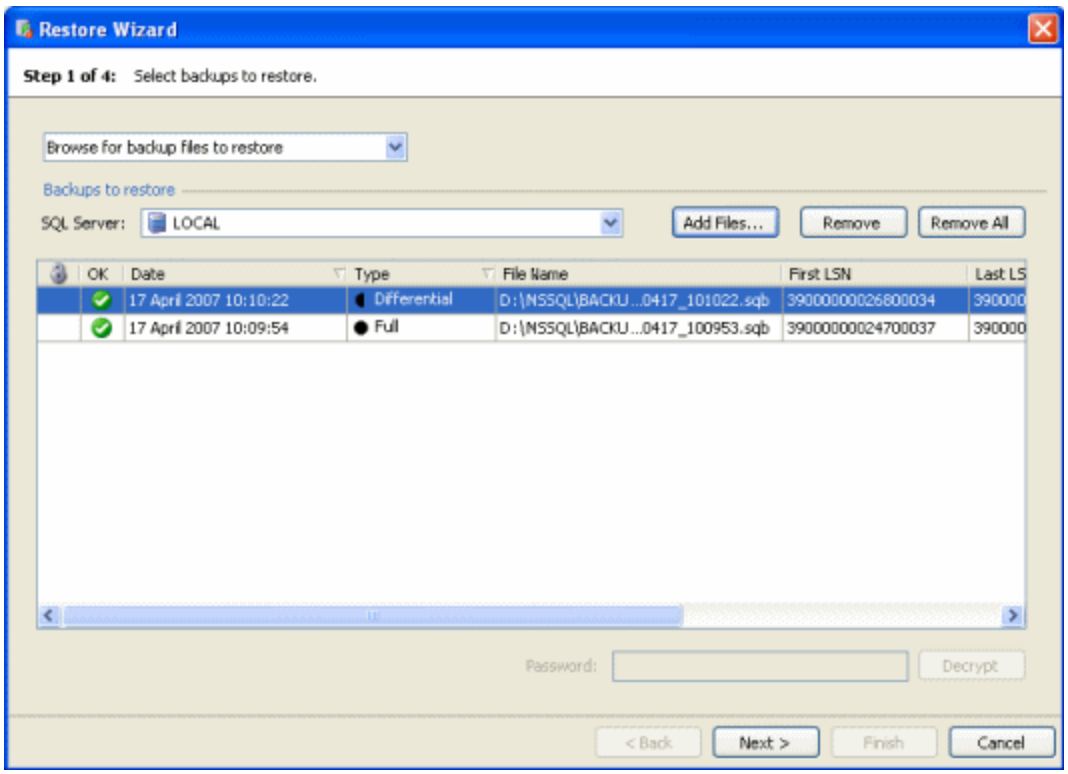
You may want to back up a database on one SQL Server, but restore the backup to a different SQL Server. Similarly, you may want to restore a backup from a copy of the backup files. (If the backup files are on a network share, you are recommended to copy the files to a local folder before restoring; for more information, see Backing up and restoring on a network share.)

In these circumstances, you must choose **Browse for backup files to restore**. (If you choose **Select from backup history**, when you select the **SQL Server** to which you want to restore the backup, you will not see the backup you require in the **Available Restore Points** list because it is not associated with the destination SQL Server.)

Select the **SQL Server** to which you want to restore the backups, and then click **Add Files**. The **File Browser** is displayed for you to select the backup files. From the **Server** drop-down list, select the server on which the backups are stored. If the server is not on the list, click **Add Server** a nd specify the name or IP address of the (physical) server. You can then select the files from the tree view.

> Other servers will only be visible to the local server if it has the appropriate permissions to write to or read from them. The name of the local server you are connected to and your user name are displayed above the **Server** list. This information may explain why some servers cannot be browsed.

When you click **OK** on the File Browser, the files you have selected are listed in the Restore wizard.



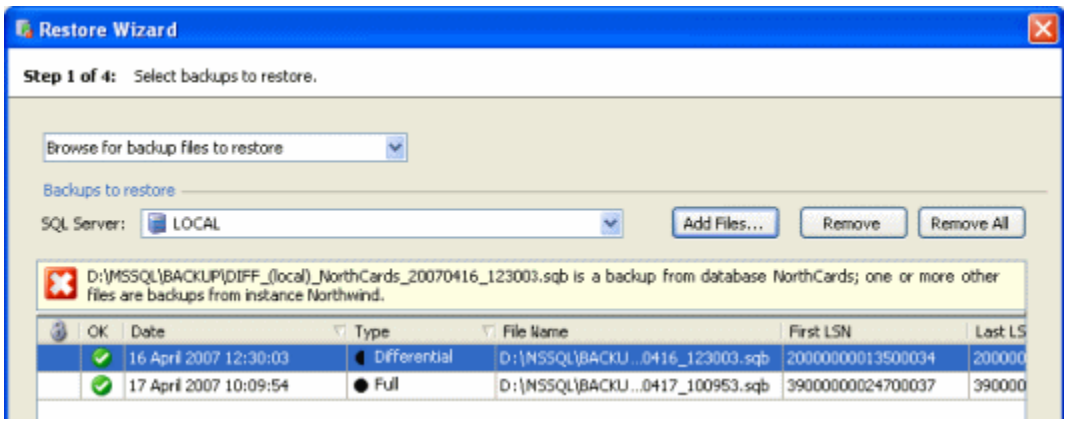

is displayed next to any encrypted files. Type the password in **Password**, and click **Decrypt**. For all files that use the password you entered,

is displayed. You must enter the passwords for all selected files before you click **Next**.

If there are any problems with a file, for example a file is missing,

is displayed. To see details of the error, move your mouse pointer over the icon.

If there are any problems with your selection, for example if you have selected backup files from different backup sets or databases, a message is displayed above the list of files.

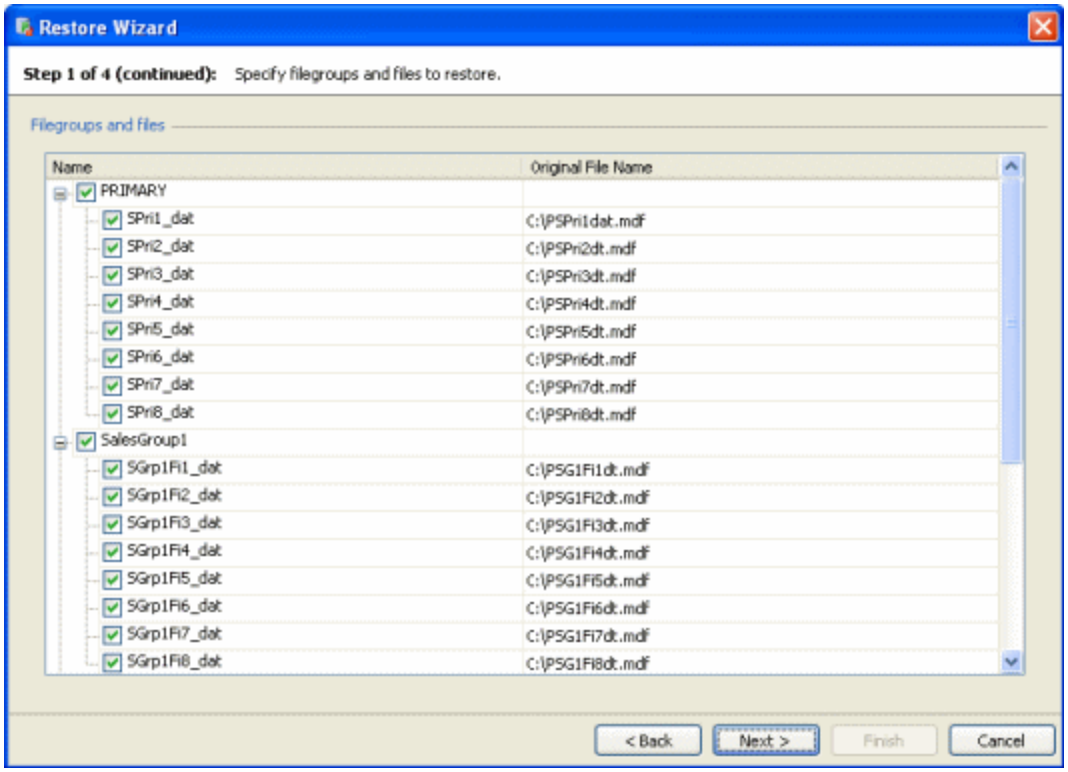


SQL Backup displays only one message at a time, therefore when you have fixed the displayed problem, you may see a second message.

To remove a file from the list, select the file and click **Remove**; to clear the list, click **Remove All**.

Files or filegroup backups

If you have selected a files/filegroups backup, when you click **Next** a tree of the available files is displayed.

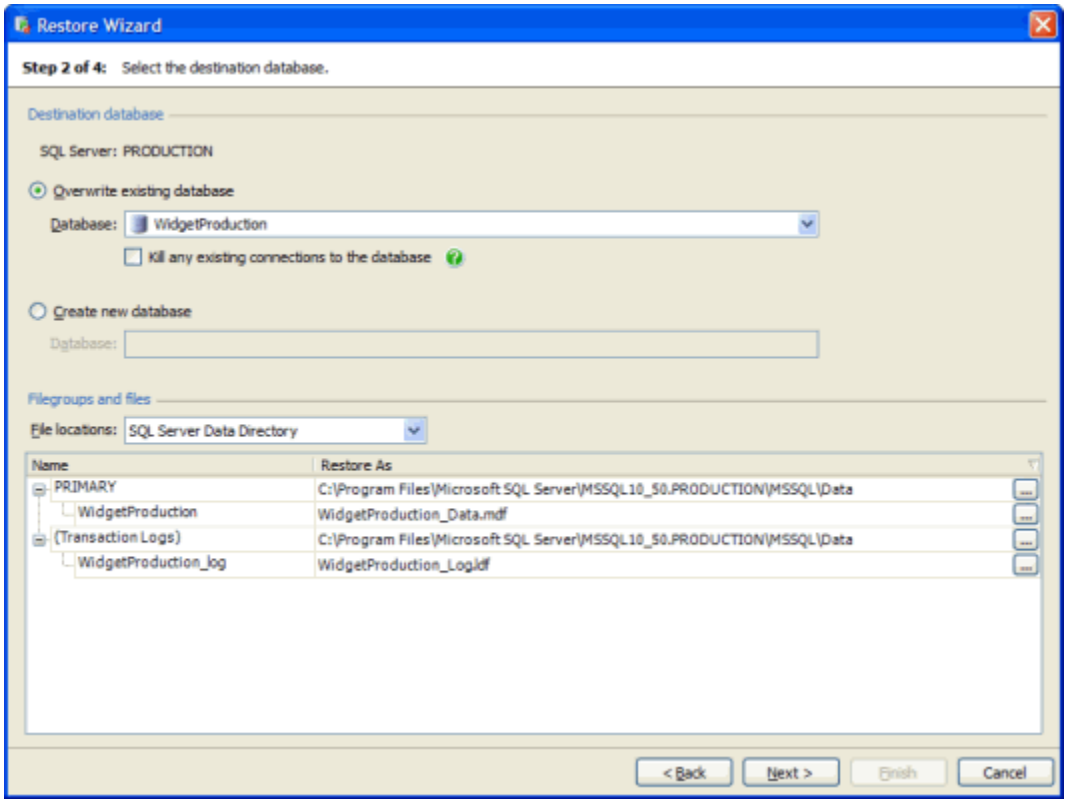


Select the check boxes for any files or filegroups that you want to restore and click **Next**.

Next: destination database

Restoring backups - destination database

On step 2 of the wizard, specify the destination database and the location of the database files.



Overwriting an existing database

To restore the backups to an existing database, select **Overwrite existing database** and select the database.

If you are restoring a full database backup or a filegroup backup, you can select any existing database. If you are restoring a differential database backup or a transaction log backup, only those databases that are in an "unrecovered" or "standby" state are available for selection.

The restore process will fail if there are existing connections to the destination database. To automatically kill any existing connections to the destination database before starting the restore process, select **Kill any existing connections to the database**. If you do not select this option, you will need to kill existing connections manually before restoring.

Transaction log tail

If you select a database that is **not** in an "unrecovered" or "standby" state but uses the Full or Bulk-Logged recovery model (full or filegroup backups only), when you click **Next** SQL Backup prompts you to back up the tail of the transaction log. Backing up the tail of the transaction log ensures that you capture all the transaction log records that have not already been backed up before overwriting the database.

- To back up the tail of the transaction log, select **Back up the transaction log before proceeding with the restore** and then click **Back Up Now**.

  > If there are existing connections to the destination database, and you have not selected **Kill any existing connections to the database** in the previous step of the wizard, the transaction log backup (and subsequent restore) will fail. Either kill connections manually before backing up the transaction log, or return to step 2 of the wizard and select the option. This will kill existing connections before the transaction log is backed up.

  A wizard is displayed for you to enter the backup settings you require. See Creating backups for details. The script to back up the tail of the transaction log is displayed only in this wizard. When you click **Finish** on the wizard, the transaction log is backed up. Click **Close** on the message box, and then click **Next** to proceed with the Restore wizard.
- If you do not want to back up the tail of the transaction log, select **Discard transactions in the tail of the transaction log** and click **Next** to proceed with the Restore wizard.

There is no requirement to back up the tail of the transaction log before restoring if you have selected an existing database that is:

- in an "unrecovered" or "standby" state
- or is **not** in an "unrecovered" or "standby" state, but uses the Simple recovery model (full or filegroup backups only)

This is equivalent to using `WITH REPLACE` in a T-SQL `RESTORE` statement. Refer to your SQL Server documentation for full details.

Filegroups and files

The data and log files (and any full text catalogs or filestream data) will be restored to the default SQL Server data and log directories unless you specify other locations:

- To restore the data and log files to the locations specified in the backup, select **Original database data and log directory** from the **File locations** list.
- To restore the data files or log file to different locations or file names, click  next to the file or folder and specify the new location or file name.

> Any files listed under **Filegroups and files** will be overwritten; other files that are not shown in the list but which previously belonged to the database, will be deleted.

Creating a new database

If you are restoring a full database backup, you can select **Create new database** to restore to a new database. Type a name for the database in the box.

Filegroups and files

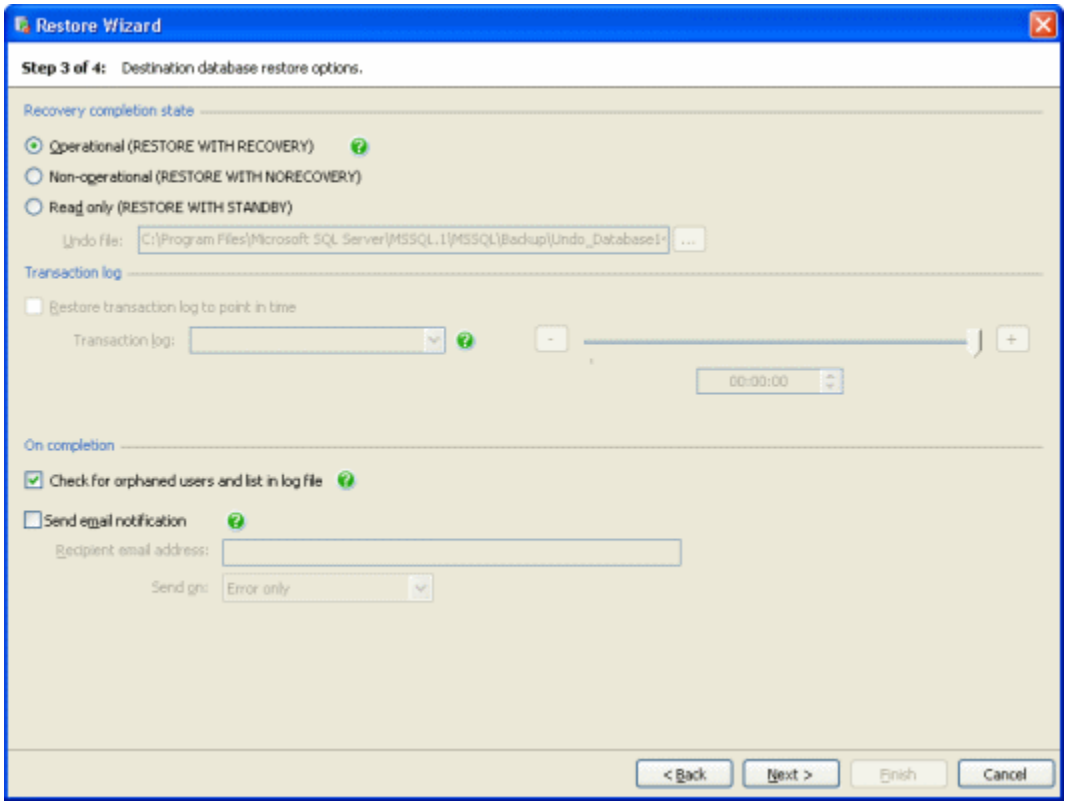
The data and log files (and any full text catalogs or filestream data) will be restored to the default SQL Server data and log directories unless you specify other locations:

- To restore the data and log files to the locations specified in the backup, select **Original database data and log directory** from the **File locations** list.
- To restore the data files or log file to different locations or file names, click  next to the file or folder and specify the new location or file name.

Next: restore options

Restoring backups - restore options

On step 3 of the wizard, configure the settings to be used when you restore the database.



Recovery completion state

Select the state in which you want the database to be left on completion of the restore operation.

- **Operational**
  The database is restored from the backup, and recovery is completed. Incomplete transactions are rolled back. The database is fully usable. Additional transaction logs or differential backups cannot be restored. This is equivalent to using the SQL Server `RESTORE WITH RECOVERY` option.
- **Non-operational**
  The database is restored but not recovered. Incomplete transactions are not rolled back. The database is not usable, but differential backups and transaction log backups can be restored to it. This is equivalent to using the SQL Server `RESTORE WITH NORECOVERY` opti on.
- **Read only**
  The database is restored and recovered. Incomplete transactions are rolled back, but recorded in the **Undo** file. The data in the database can be read, but not modified. Differential backups and transaction log backups can be restored to it. This is equivalent to using the SQL Server `RESTORE WITH STANDBY` option.
  To change the location of the Undo file, in the **Undo file** box, type the full path on the standby server for the file, or click
  
  and specify the file using the File Browser. Note that the file path is relative to the selected SQL Server. For example, if you are restoring a database on a remote SQL Server instance called *ServerA* and you specify a local path such as *C:\Undo*, the backup files will be created on the C: drive on *ServerA*, not on the local server.

Transaction log

If you are restoring transaction logs, you can specify the exact time to which you want to restore the transactions. Select the **Restore transaction log to point in time** check box, and select a transaction log from the list. The slider shows the time period covered by the selected transaction log; move the slider to select the exact time to which you want to restore the transactions.

If you have selected a transaction log backup for which SQL Backup cannot find the preceding backup, SQL Backup cannot determine the start time of the period covered by the transaction log backup. You must therefore type the time and date of the point in time to which you want to restore the backup.

On completion

These options are applied once the restore process has completed.

Check for orphaned users and list in log file

If you are restoring a database onto a different SQL Server instance, it is possible that some of the restored database user accounts will not have an associated server login on that instance. These database user accounts are known as orphaned users, and will be unable to provide access to the database unless you create their associated logins.

Select the **Check for orphaned users and list in log file** check box to run a check for orphaned users when the restore job has completed. If any orphaned users are found, a warning is included in the SQL Backup log file, along with a list of the orphaned users.

SQL Backup does not fix orphaned users automatically. If orphaned users are reported, you should consider creating the required logins on the SQL Server instance that you are restoring to.

For more information about database user accounts, logins, and fixing orphaned users, refer to your SQL Server documentation.

Send email notification

If you want to receive an email with a copy of the completion log, select the **Send email notification** check box, and enter the recipient's email address. This option is available only if you have entered your email settings. For more information about setting up email notification, see Email settings.

To send the log to multiple email addresses, type each address separated with a semi-colon (**;**). For example:

*dba01@myco.com**;**dba02@myco.com*

By default, email notifications are sent only when there are errors during the backup process (**Error only**). If you want to receive an email when an error or warning occurs, select **Error or warning**; to always receive an email on completion of the backup process (on success or failure), select **Any outcome**.

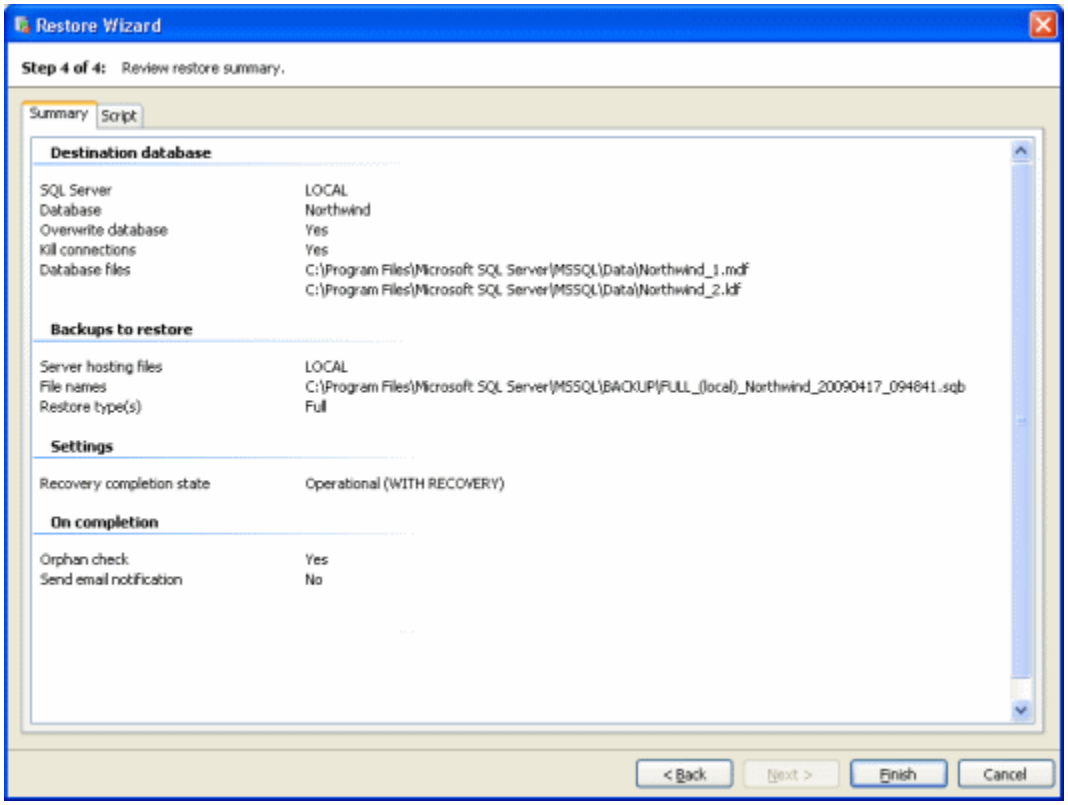
> If SQL Backup Lite is installed on the SQL Server, you cannot use email notification.

Next: review summary

Restoring backups - review summary

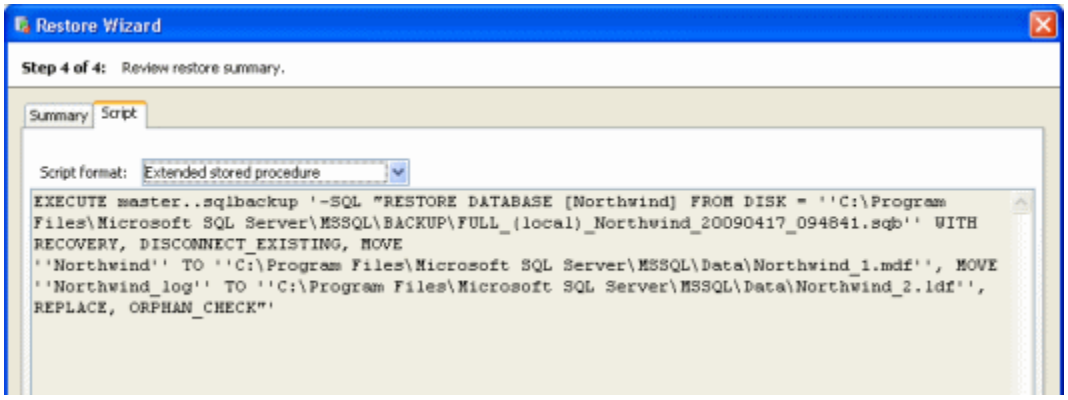
Restoring backups > Select backups > Destination database > Restore options > **Review summary**

On step 4 of the wizard, review your restore settings and, if required, copy the script.



The **Summary** tab displays a simple report of the options you have set for the restore operation, for you to check.

To see the script that will be used, click the **Script** tab.



You can then select the format in which to view the script:

- **Extended stored procedure** shows the script you can use to run the restore operation when you are connected to the server using an application such as Microsoft SQL Server Management Studio, or connectivity tools such as ADO, OLEDB, ODBC.For information about how you can use the SQL Backup extended stored procedure to restore databases, see Using the extended stored procedure.
- **Command line** shows the script you can use to run the restore operation from the command line.For information about how you can use the SQL Backup command line to restore databases, see Using the command line.

> If you chose to back up the tail of the transaction log, the script to do this is displayed on the summary page of the **Backup Transaction Log Tail** wizard.

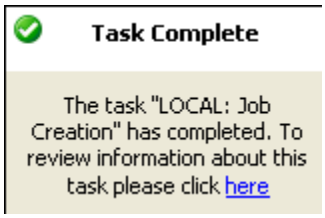
Refer to your SQL Server documentation for more information about backing up the tail of the transaction log.

When you have checked the settings, click **Finish** to start the restore process.

SQL Backup displays a message dialog box that shows the progress of the restore operation. Click **Hide** to minimize this dialog box and continue working. You can display the box again by clicking the arrow in the SQL Backup status bar.

The progress is also displayed in the In Progress tab.

When the restore process completes, if the message box is minimized, a pop-up message is displayed to inform you that the task is complete.

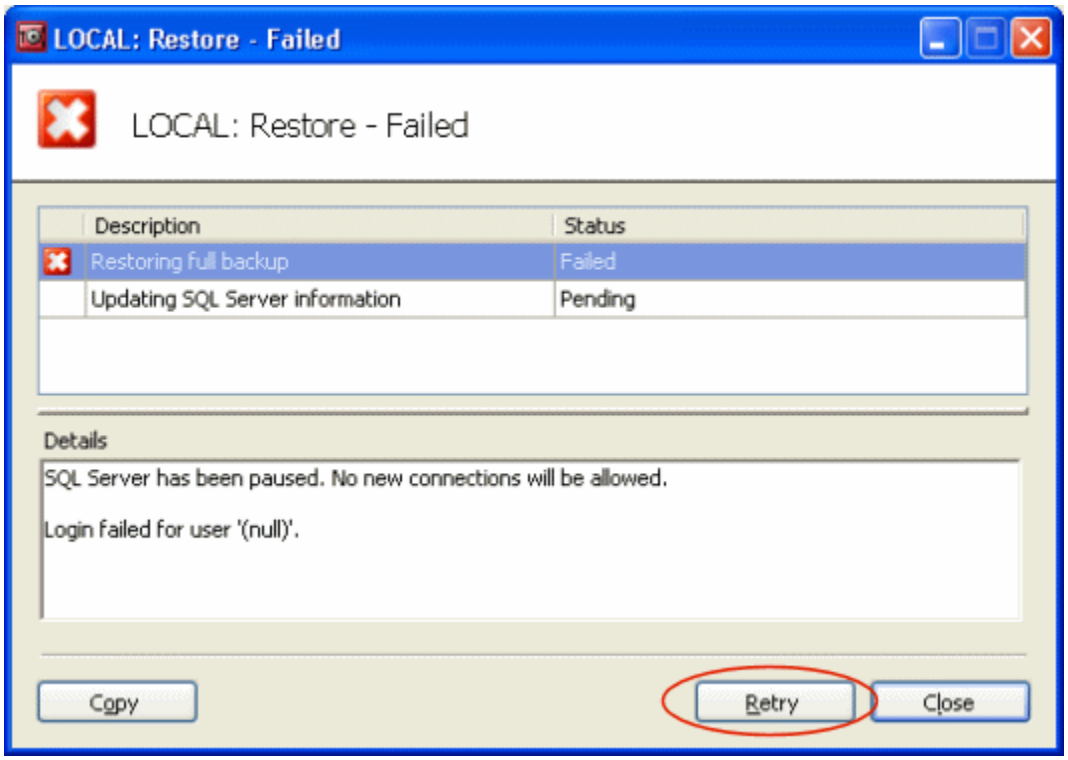


Information about the backup is then displayed in the Activity History. This information is also sent to a log file; you can view the contents of the log file by right-clicking the activity, and selecting **Show Log**. By default, log files are located in:

- *%PROGRAMDATA%\Red Gate\SQL Backup\Log\<instance name>* (Windows Vista, Windows 2008 and later), or
- *%ALLUSERSPROFILE%\ Application Data\Red Gate\SQL Backup\Log\<instance name>* (Windows XP and Windows 2003).

You can change this location in your log file options (see File management options).

If the restore process fails, you can optionally return to the final step of the wizard by clicking **Retry** in the message dialog. You can then change the settings in any of the wizard steps before starting the restore process again.



Restoring multiple backups

You can restore multiple full, differential, and transaction log backups that belong to the **same backup set** using the Restore wizard. For example, you can restore a full and differential backup for the same database at the same time. Details of how to do this are given in step 1 (select backups).

You can restore the latest set of backups for a particular database (the latest full backup, a subsequent differential backup and all available transaction log backups) using the LATEST_ALL argument when using the command line or extended stored procedure. See LATEST_ALL in Th e RESTORE command for more information.

You can restore multiple transaction log backups for a particular database using the command line or extended stored procedure. See The RESTORE command for more information.

Restoring the master database

To restore the *master* database, you must use the SQL Backup command line to ensure that the SQL Server is started in single user mode. It is not possible to restore the *master* database with the Restore wizard.

1. Stop the SQL Server service.
2. Open a Command Prompt window, and navigate to the SQL Server instance's *Binn* folder.
3. Start SQL Server in single user mode using `sqlservr.exe -c -m` for an unnamed instance, or `sqlserver -c -m -s <instance name>` for a named instance.
4. Use the SQL Backup command line interface (`SQLBackupC.exe`) to restore the *master* database.
   For example, at the command prompt, type:

   ```
   SQLBackupC.exe -SQL "RESTORE DATABASE master FROM DISK = 'C:\Backups\master
   20070101.sqb'"
   ```

   For a named instance, use the **-I** parameter to specify the instance name, for example:

   ```
   SQLBackupC.exe -SQL "RESTORE DATABASE master FROM DISK = 'C:\Backups\master
   20070101.sqb'" -I Instance2
   ```

   If you are using SQL Server authentication, use the **-U** and **-P** parameters to add the authentication details, for example:

   ```
   SQLBackupC.exe -SQL "RESTORE DATABASE master FROM DISK='C:\Backups\master
   20070101.sqb'" -I Instance2 -U sa -P MyPassword
   ```

   If you are restoring the master database to a new (or rebuilt) server you will need to add the WITH REPLACE keyword, for example:

   ```
   SQLBackupC.exe -SQL "RESTORE DATABASE master FROM DISK='C:\Backups\master
   20070101.sqb' WITH REPLACE" -I Instance2
   ```

   SQL Backup restores the *master* database, and SQL Server automatically stops the SQL Server service when the restore operation is complete.
5. You can now start the SQL Server in normal mode.

See Using the command line for more information.

Log shipping

Log shipping refers to the process of taking transaction log backups of a database on one server, shipping them to a secondary server and restoring them.

In the log shipping process:

Transaction log backups are performed on the source database on the primary SQL Server (for example the production database).

The backup files are copied to a network share that can be accessed by both SQL Servers.

The backup files are restored to the destination database on the secondary SQL Server.

Log shipping is useful if you are maintaining a "standby" server as a backup to your primary server, but do not require automatic failover. For information on manually failing over to a log-shipped standby server, see Failing over to a standby server.

Alternatives for maintaining a standby server include replication, a failover clustering solution, AlwaysOn Availability Groups, and database mirroring. For more information, see Configuring High Availability in SQL Server Books Online. For a detailed comparison of log shipping and replication, see Log Shipping vs Replication on SQLServerCentral.com.

Why use SQL Backup for log shipping?

SQL Backup simplifies the configuration process for log shipping and protects against network outages. It also makes the backup and restore process up to ten times faster than native backup and restore, and supports compression and encryption.

The SQL Backup graphical user interface provides a Log Shipping wizard for the configuration of log shipping. If you have not already set up the destination database for log shipping, the wizard can do this for you by taking a full backup of the source database and restoring it to a new or existing database on the secondary SQL Server. This ensures the destination database is consistent with the primary database and in the correct state for receiving the transaction log backups.

The wizard then sets up SQL Server Agent jobs that use SQL Backup to perform the backup and restore operations. A SQL Server Agent job on the primary server periodically backs up the transaction logs and copies the backup files to the shared folder. A job on the standby server periodically retrieves the backup files from the shared folder and restores the transaction logs.

If the copying process fails for a backup file (due to an extended network outage, for example), SQL Backup automatically attempts to copy the file to the network share at regular intervals. Any future transaction log backups that are created are queued behind the file that could not be copied. Once the copying process is working again (because the network outage has been fixed, for example), SQL Backup copies each transaction log backup to the network share, in the correct order.

For more information about using the wizard to set up log shipping, see Configuring log shipping.

Prerequisites

- You must have at least two SQL Server database engine servers or two database engine instances in your log shipping implementation.

  > You cannot ship transaction logs from a later version of SQL Server to an earlier version (for example, from SQL Server 2005 to SQL Server 2000) because of the differences in their structures. You *can* ship transaction logs from earlier versions of SQL Server to later versions (for example, from SQL Server 2005 to SQL Server 2008), but you must select the Non-operational recovery completion state (RESTORE WITH NORECOVERY). This is a SQL Server restriction.

- SQL Backup server components must be installed on both of the SQL Servers.
  However, you cannot ship transaction log backups created on a SQL Server instance that has SQL Backup 6 server components or later installed, to an instance that has an earlier version of the server components installed. This is because of differences in the SQL Backup (.sqb) file format introduced in SQL Backup 6. In this scenario, you should update the destination SQL Server instance with the latest SQL Backup server components before configuring log shipping.
- The source database must use the full or bulk-logged recovery model. Databases using the simple recovery model do not implement the transaction log in a manner compatible with log shipping and will not be displayed. Note that a number of issues may arise when changing a database recovery model; refer to your SQL Server documentation for more information.
- You must have a shared folder to copy the transaction log backups to. The SQL Server Agent service account of the primary server must have read/write access either to the shared folder or to the local NTFS folder. The SQL Server Agent account of the standby server must have read and delete access to the shared folder.
- The SQL Server Agent services must be running and configured with network credentials (such as a domain account) if you plan to use a network share as the shared folder. You can configure log shipping with SQL Server Agent services stopped, but the process does not run until the Agent is started.
- You must have *execute* permissions on the following extended stored procedures: *sqbdata*, *sqbdir*, *sqbmemory*, *sqbstatus*, *sqbutility*, *sqlbackup*.
- You must be a member of the *sysadmin* fixed server role on the participating servers.

Customizing log shipping

The Log Shipping wizard may not provide all the options you need. If required, you can use the wizard to set up the jobs, and then use SQL Server Management Studio to modify the job steps manually.

For more information about SQL Backup syntax, see Scripting SQL Backup.

Monitoring the log shipping

You can use the free Log Shipping Monitor tool for collating and summarizing SQL Server log shipping activity. The Log Shipping Monitor provides customizable alerts for key log shipping variables, allowing you to take steps to avoid log shipping failures. For example, you can set up alerts to notify you if the free space on a drive falls below a specified size, or if no transaction log backups have been created or restored on the standby server for a specified period of time.

In the event of a log shipping failure, you can use the information provided by the Log Shipping Monitor to identify possible causes so that you can avoid future failures.

You can download the Log Shipping Monitor from Redgate Labs.

Configuring log shipping

SQL Backup provides a wizard to guide you through the process of setting up *log shipping*, to ship transaction log backups from a primary server to a secondary server and restore them.

To start the Log Shipping wizard, click  **Log Shipping**. The Log Shipping wizard comprises the following steps:

Step 1:	Select the SQL Server instance database you want to back up, and the SQL Server instance database on which you want to restore the transaction log backups.
Step 2:	Specify the locations and file names for the transaction log backups, configure settings for managing existing backup files, and specify backup processing settings such as compression.
Step 3:	Configure the settings for restoring the transaction log backups.
Step 4:	Specify the network share to which the transaction log backups will be copied, and network resilience settings.
Step 5:	Specify the schedule for backing up the transaction logs, and for restoring the backups.
Step 6:	Review your log shipping settings.

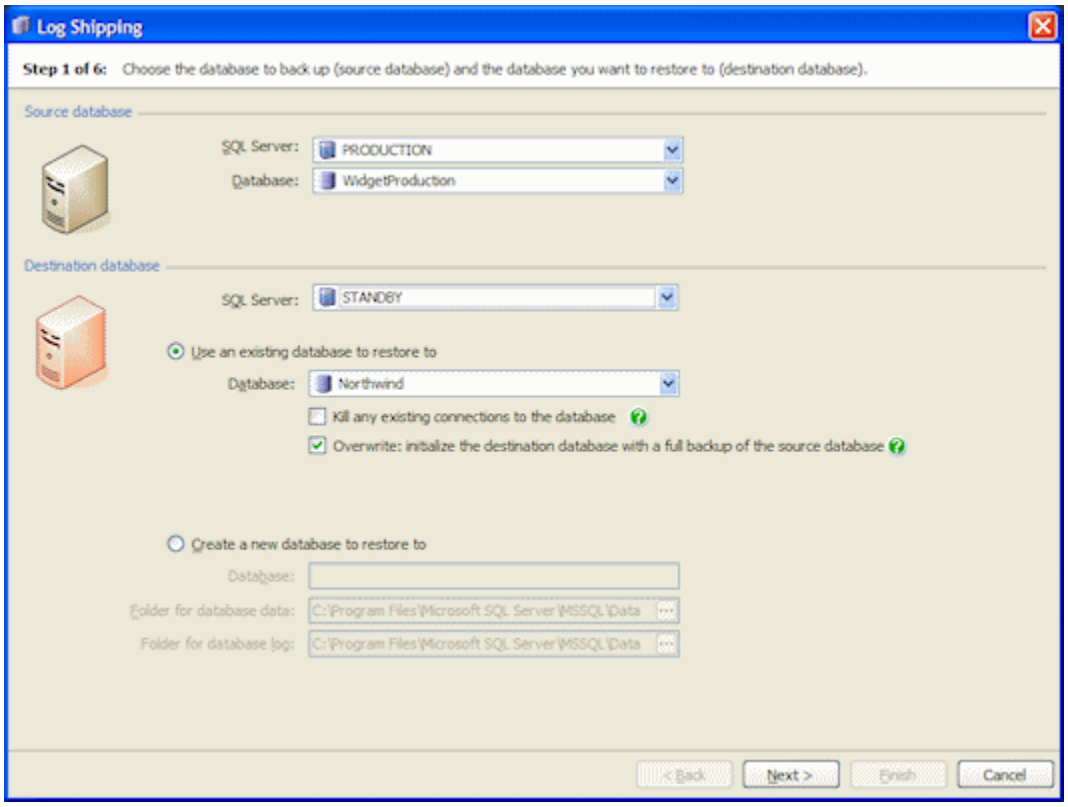
If you encounter problems, check that you have met all the prerequisites for log shipping.

Next: specify databases

Log shipping - specify source and destination database

Log shipping > **Specify databases** > Backup settings > Restore settings > Network share > Schedule > Review summary

On step 1 of the wizard, select the SQL Server instance and database that you want to back up (the source database), and the SQL Server instance and database to which you want to restore the transaction logs (the destination database).



Note that if the selected SQL Server does not have the server components installed, a warning is displayed and you must install the server components to proceed.

In step 3 of the wizard, you will specify the network share to which backup files will be copied. The user that the SQL Backup Agent service is running as on the source SQL Server must have permissions to create files on the network share you are going to use.

Similarly, the user that the SQL Backup Agent service is running as on the specified destination SQL Server must have permissions to access the network share and to erase files from the network share, so that it can move the files.

Source database

In the **SQL Server** list, select the SQL Server instance for the database for which you want to ship transaction logs.

In the **Database** list, select the database for which you want to ship transaction logs. The database cannot be in simple recovery mode or standby mode, and must not be a system database. It must also be online.

Destination database

Select the SQL Server on which you want to restore the transaction log backups.

You can then choose to restore the transaction logs to an existing database on the secondary SQL Server, or to a new database.

Restoring to an existing database

To restore the transaction logs to an existing database on the secondary SQL Server, select **Use an existing database to restore to**, and then select the name of the database.

To automatically kill any existing connections to the destination database before starting log shipping, select **Kill any existing connections to the database**. Log shipping will fail if there are existing connections to the destination database; if you do not select this option, you will need to kill existing connections manually.

By default, SQL Backup will create a full backup of the source database and restore it to the secondary SQL Server prior to creating the log shipping tasks. This effectively overwrites the destination database, and ensures that the destination database is in the correct state for restoring the transaction log backups.

If you have already prepared your destination database so that it is ready to receive transaction logs, you should clear the **Overwrite: initialize the destination database with a full backup of the source database** check box. You must ensure that the destination database is in `NO RECOVERY` or `STANDBY` mode, and is populated with the most recent backup of the source database. You must also restore any subsequent backups taken on the source database before log shipping can start.

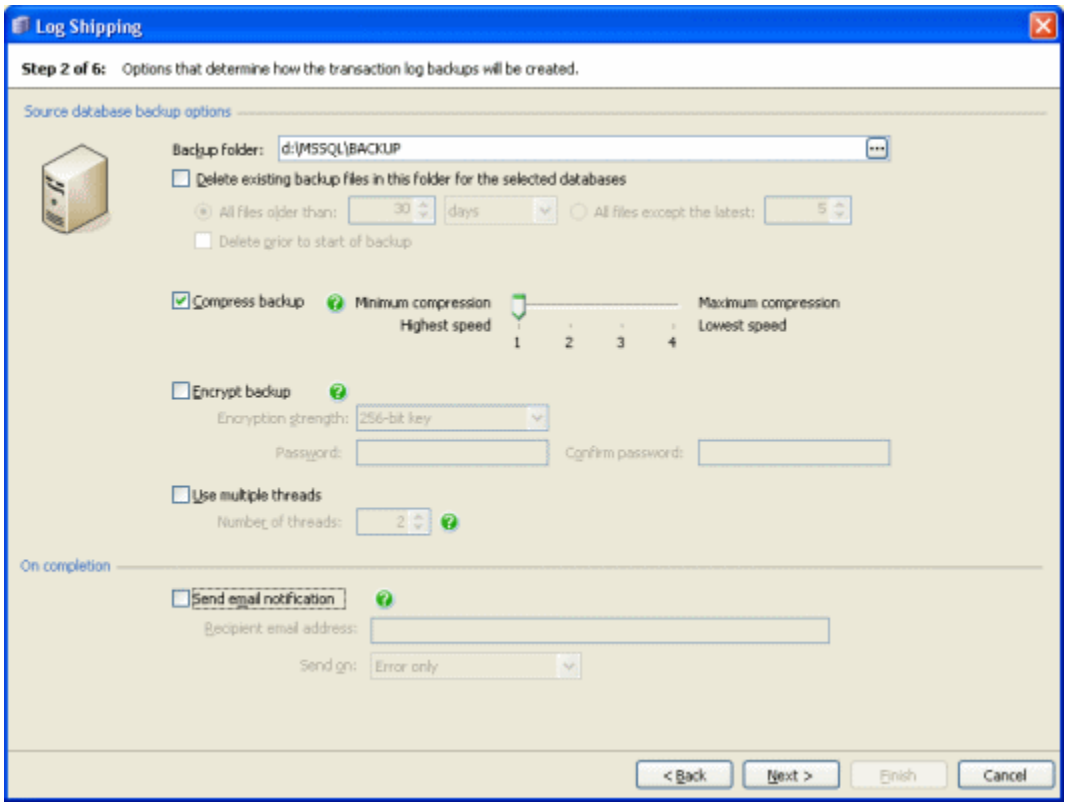
Restoring to a new database

To restore the transaction logs to a new database, select **Create a new database to restore to**. Then type a name for the new database and specify the folders in which you want the database's data and log files to be stored.

Next: backup settings

Log shipping - backup settings

On step 2 of the wizard, specify the locations and file names for the transaction log backups, configure settings for managing existing backup files, and specify backup processing settings such as compression.



Backup location

A default folder for the transaction log backups is displayed; the file names will be generated automatically. To change the folder, type the new path or click

and specify the folder using the folder browser. You can use tags in both the path and file name. For more information, see File location tags.

> The file path is relative to the SQL Server that you are backing up. For example, if you back up a transaction log 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 machine.
>
> If you specify a network share as the backup file location, the SQL Backup Agent service "log on" user must have *Full* permissions to access the location. For more information, see Backing up and restoring on a network share.

File management

Select **Delete existing backup files in this folder for the selected databases** if you want SQL Backup to delete backups of the selected database that exist in the destination backup folder. You can restrict deletion of existing backup files by age (**All files older than**) or number (**All files except the latest**).

By default, the files are deleted when the backup process has completed. If the backup fails, the files are not deleted. To delete the files before the backup is created, select the **Delete prior to start of backup** check box. For example, you may want to do this to create space for the new backup files. However, note that if the backup fails, the old files will have been deleted; you are therefore recommended to select this check box only if you have a copy of the existing backups.

> If SQL Backup cannot list the contents of the folder that contains the files to be deleted, it cannot delete the files. Ensure that the user account from which you are running SQL Backup has permissions to list the folder contents.

Backup processing

To compress the backups, select the **Compress backup** check box and select the compression level by moving the slider. For more information about compression levels, see Compression levels.

To encrypt the backup, select the **Encrypt backup** check box, then type a password for the backup in **Password**, and again in **Confirm password**. You can choose 128-bit or 256-bit encryption.

Optimization

SQL Backup can use multiple threads to create the backups. This 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. For more information about using multiple threads, see Optimizing backup speed.

On completion

If you want to receive an email with a copy of the completion log, select **Send email notification**, and enter the recipient's email address. This option is available only if you have entered your email settings. For more information about setting up email notification, see Email settings.

To send the log to multiple email addresses, type each address separated with a semi-colon (**;**). For example:

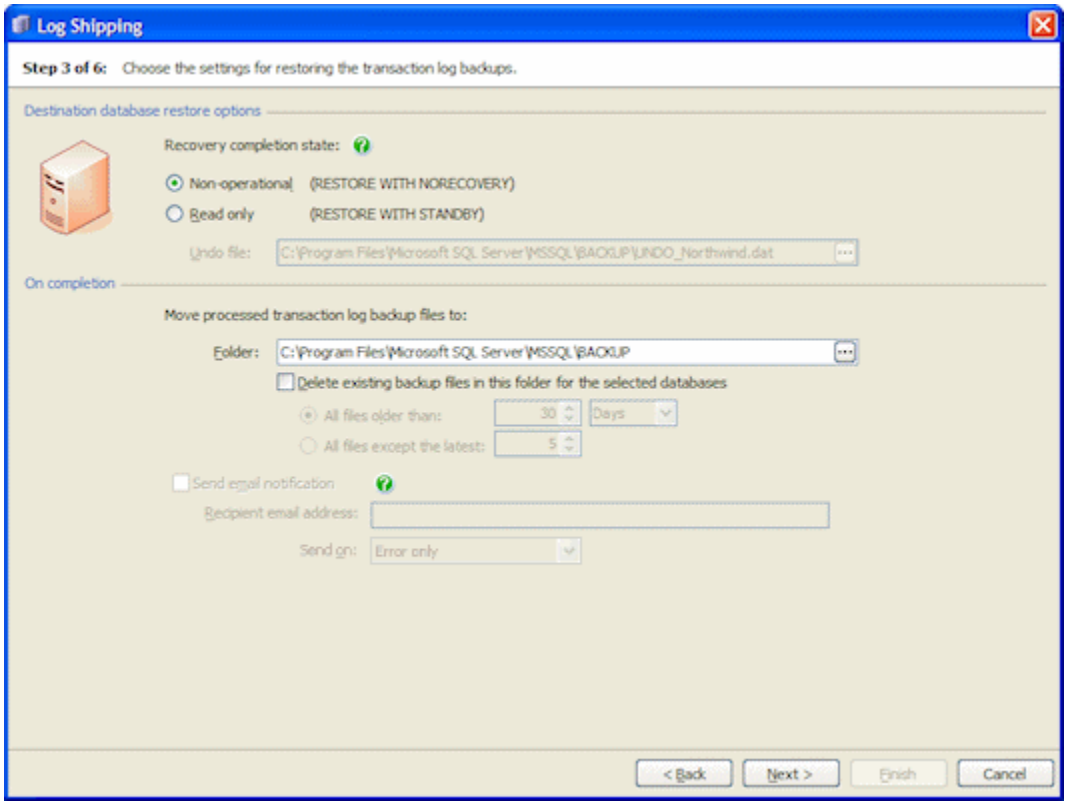
*dba01@myco.com**;**dba02@myco.com*

By default, email notifications are sent only when there are errors during the backup process (**Error only**). If you want to receive an email when an error or warning occurs, select **Error or warning**; to always receive an email on completion of the backup process (on success or failure), select **Any outcome**.

Next: restore settings

Log shipping - restore settings

On step 3 of the wizard, configure settings for restoring the transaction log backups.



Destination database restore options

Select the state in which you want the destination database to be left on completion of the restore operation. Which state you select depends on how you intend to use the destination database:

- **Non-operational** This is equivalent to using the SQL Server `RESTORE WITH NORECOVERY` option. The database is restored but not recovered. Incomplete transactions are not rolled back. The database is not usable and cannot be queried by users.
- **Read only** This is equivalent to using the SQL Server `RESTORE WITH STANDBY` option. The database is restored and recovered. Incomplete transactions are rolled back, but recorded in the **Undo file**. This allows users read-only access to the data between transaction log restores. This is useful if you want to reduce the load on the primary SQL Server by running queries on the destination SQL Server.
  To change the location of the *Undo* file, in the **Undo file** box, type the full path on the standby server for the file, or click
  
  and specify the file using the File Browser. Note that the file path is relative to the selected SQL Server. For example, if you are restoring a database on a remote SQL Server instance called *ServerA* and you specify a local path such as *C:\Undo*, the backup files will be created on the C: drive on *ServerA*, not on the local server.

On completion

When they have been restored, SQL Backup moves the transaction log backup files from the network share (which you specify in the next step) to another folder, so that they are not processed in the next run. Specify the path for the folder to which the processed transaction log backup files are to be moved.

SQL Backup displays a default path for the files. To change the path, type the full path for the folder, or click

and specify the folder using the folder browser. The file path is relative to the SQL Server that you are restoring to. For example, if you are restoring to a remote SQL Server instance called *ServerA* and you specify a local path such as *C:\ProcessedBackups*, the processed backup files will be created on the C: drive on *ServerA*, not on the local machine.

> The folder that processed backup files are moved to must not be the same location as the network share specified in step 4.

Select the **Delete existing backup files in this folder for the selected databases** check box if you want SQL Backup to delete existing backup files for the selected database in the specified folder. You can restrict deletion of existing backup files by age or number. To restrict deletion of files by age, select **All files older than** and type or select the age (in days or hours) of the files that you want to delete. To restrict deletion of files by number, select **All files except the latest** and type or select the number of the latest backup files to keep.

If you want to receive an email with a copy of the completion log, select **Send email notification**, and enter the recipient's email address. This option is available only if you have entered your email settings. For more information about setting up email notification, see Email settings.

To send the log to multiple email addresses, type each address separated with a semi-colon (**;**). For example:

*dba01@myco.com**;**dba02@myco.com*

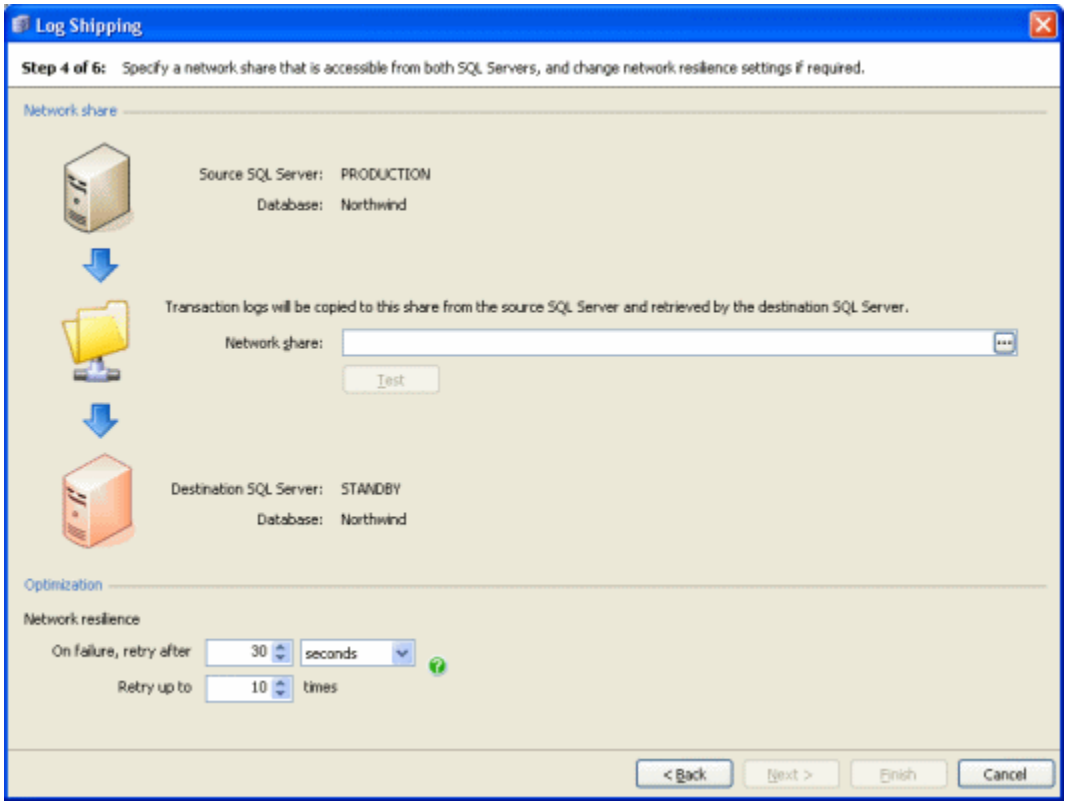
By default, email notifications are sent only when there are errors during the backup process (**Error only**). If you want to receive an email when an error or warning occurs, select **Error or warning**; to always receive an email on completion of the backup process (on success or failure), select **Any outcome**.

Next: network share

Log shipping - network share

Log shipping > Specify databases > Backup settings > Restore settings > **Network share** > Schedule > Review summary

On step 4 of the wizard, specify the network share to which the transaction log backups will be copied and network resilience settings.



When SQL Backup has backed up the source database, the backup file is copied to a folder that can be accessed by the secondary SQL Server. The SQL Backup Agent service on both the primary and secondary SQL Servers must have access to this folder.

Type the name of the network share to which you want to copy the backup files. Alternatively, click

and specify the network share using the Folder Browser: from the **Server** drop-down list, select the server you want to copy the backups to, or click **Add Server** and type the server name or IP address. Other servers will only be visible to the local server if it has the appropriate permissions to write to or read from them. The name of the local server you are connected to and your user name are displayed above the **Server** list. This information may explain why some servers cannot be browsed.

> The network share must *not* be the same location as the folder the processed transaction log backup files are moved to on completion (specified in step 3).

If you typed the network share name, click **Test** to check that the user for the SQL Backup Agent service on the primary SQL Server has permissions to create files on the network share, and that the user for the SQL Backup Agent service on the specified secondary SQL Server has permissions to access the network share and delete files. If the user for either SQL Backup Agent service does not have the appropriate permissions, an error message is displayed. If you browsed to the network share, this check is performed automatically.

For more information, see Backing up and restoring on a network share. For details about setting up permissions to use a network share, see Permissions.

Optimization

The **Network resilience** settings control retry behavior following a read/write error whenever SQL Backup:

- writes a backup file to disk
- copies a backup file to another location

Read/write errors are most likely to occur when SQL Backup is processing files through a network connection.

The **Network resilience** settings apply to all read/write operations involving the transaction log backups. In most circumstances, you can leave **On failure, retry after** and **Retry up to (n) times** at their default values (30 seconds, and 10 times).

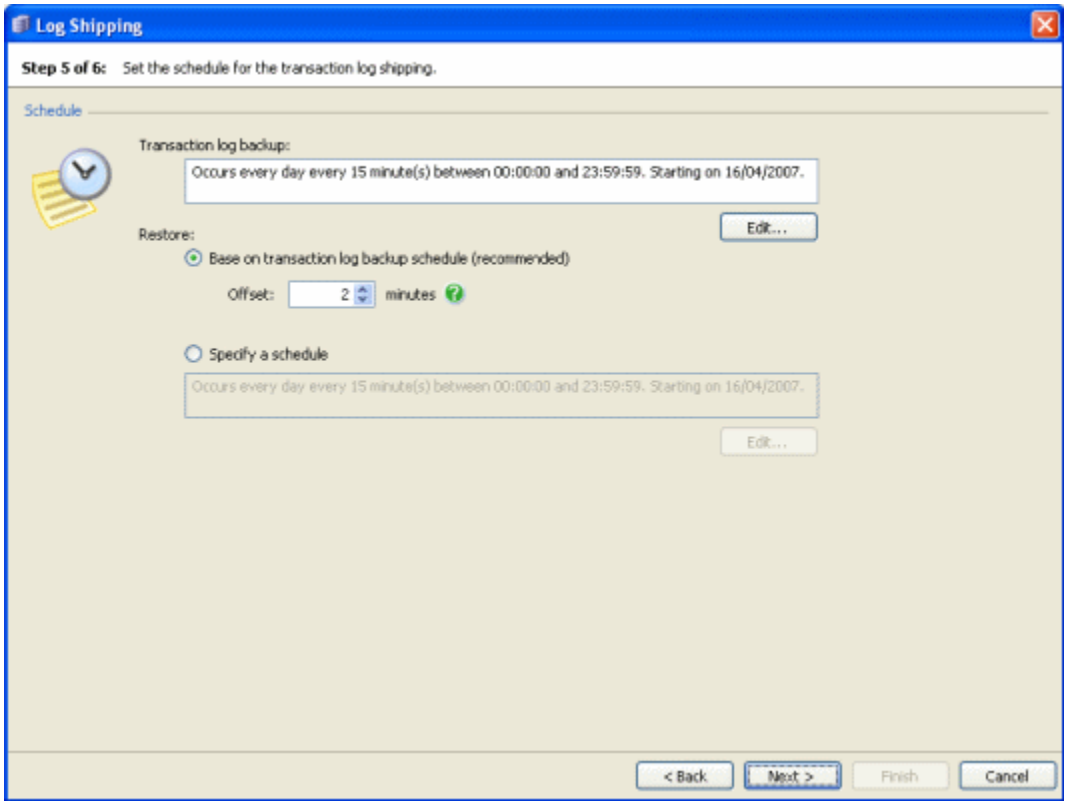
To disable retries completely, specify **Retry up to 0 times**.

Next: set the schedule

Log shipping - set the schedule

Log shipping > Specify databases > Backup settings > Restore settings > Network share > **Schedule** > Review summary

On step 5 of the wizard, specify the schedule for backing up the transaction logs, and for restoring the backups.



The schedule defines the time and frequency of the backups. For example, you may want to make transaction log backups every 5 minutes.

A default schedule is displayed for the backups. To change the schedule, click **Edit**, and on the **Schedule Editor**, specify the schedule and click **OK**.

You are recommended to use the backup schedule as a basis for the restore schedule. Select **Base on transaction log backup schedule** and in **Offset**, type or select the number of minutes by which you want to delay the restore operation after the backup has started. The offset time should be long enough to allow the backup to complete.

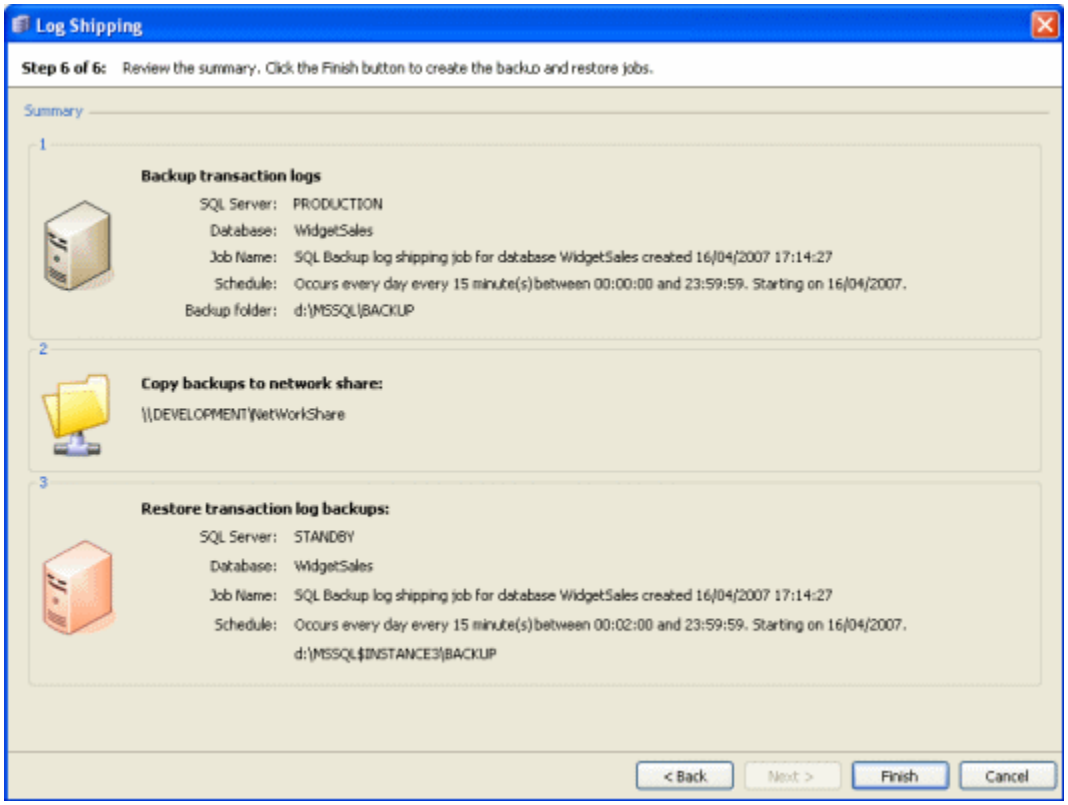
Alternatively, you can set up a separate schedule for the restore operations. Select **Specify a schedule**. To change the default schedule, click **Ed it** and on the **Schedule Editor**, specify the schedule and click **OK**.

Next: review summary

Log shipping - review summary

Log shipping > Specify databases > Backup settings > Restore settings > Network share > Schedule > **Review summary**

On step 6 of the wizard, review your log shipping settings.

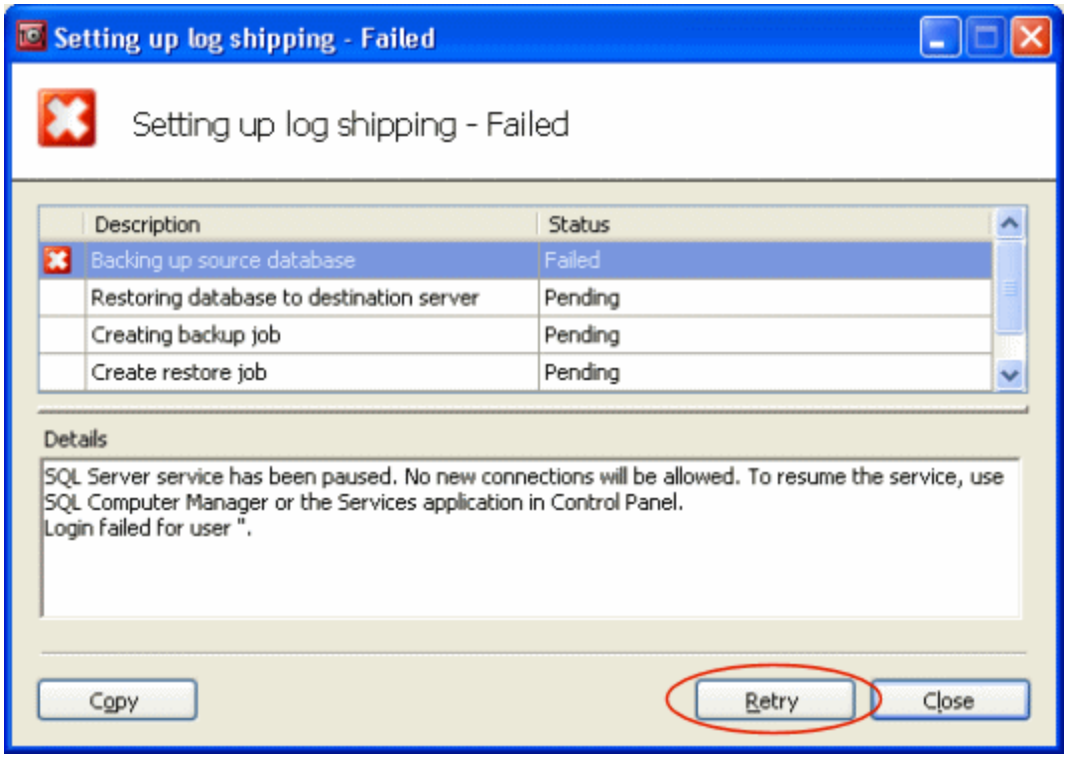


The **Summary** displays the settings you have specified. When you have checked the settings, click **Finish** to create the SQL Server Agent jobs to perform the log shipping.

The Log Shipping wizard automatically sets up job steps using the `RAISERROR` command so that the SQL Server Agent reports an error if the SQL Backup command fails. For information about SQL Server Agent jobs, steps, and the `RAISERROR` command, refer to your SQL Server documentation.

SQL Backup displays the progress of the log shipping setup. SQL Backup performs a full backup of the source database and restores it on the secondary SQL Server.

If any tasks fail, an error message is displayed, and all subsequent operations are failed. You can optionally return to the final step of the wizard by clicking **Retry** in the message dialog. You can then change the settings in any of the wizard steps before running the tasks again.


Setting up log shipping - Failed
Setting up log shipping - Failed
Description
Status
Backing up source database
Failed
Restoring database to destination server
Pending
Creating backup job
Pending
Create restore job
Pending
Details
SQL Server service has been paused. No new connections will be allowed. To resume the service, use SQL Computer Manager or the Services application in Control Panel.
Login failed for user ''.
Copy
Retry
Close

Log shipping to multiple standby servers

It is not possible to set up log shipping to multiple standby servers from the Log Shipping wizard. However, it can be set up by adapting the backup and restore scripts of an existing log shipping job as follows:

1. For each of the standby servers, set up new network share locations, ensuring that the SQL Backup Agent service startup account on each server has permission to read and write from the share.
2. Using the Log Shipping wizard, set up a log shipping job to the destination databases on your first standby server.
3. Restore a full backup of the source database to the destination databases on the remaining standby servers, using `WITH NORECOVERY` or `WITH STANDBY`. You can use the SQL Backup Restore wizard to do this.
4. Modify the backup job on the source server in SQL Server Management Studio:
    a. Open **Object Explorer** and expand **SQL Server Agent**.
    b. Open the log shipping job and select the **Steps** page.
    c. Click **Edit** and, for each of the remaining standby servers, add `COPYTO="<network share location>"` to the `WITH` clause.
5. On each of the remaining standby servers, create a new SQL Server Agent job to restore the log backups based on the restore job on the first standby server:
    a. In SQL Server Management Studio, right click **SQL Server Agent**, then select **New**, then **Jobs**.
    b. On the **Steps** page, add a new step.
    c. In the **Command** box, copy and paste the script from the restore job on the first standby server.
    d. Change the `FROM DISK` location to the appropriate network share, and the `MOVETO` location to a folder on the standby server.

Reseeding a standby database

If you encounter problems with log shipping, one solution is to take a fresh backup of the source database and restore it to the secondary server ('reseeding'). This can be done without starting the log shipping process from scratch.

1. Disable the log shipping jobs on the primary and standby servers in SQL Server Management Studio.
2. Remove all transaction log backup files from the network share location.
3. Take a full backup of the source database, and restore it to the destination database either `WITH NORECOVERY` or `WITH STANDBY`. This can be done using the SQL Backup Restore wizard. For more information, see Restoring backups.

Reseeding is now complete and the log shipping jobs on both servers can be re-enabled.

Failing over to a standby server

If you need to switch from the source database to a standby database, you must bring the standby database online manually; there is no automatic failover with the log shipping configuration.

The standby database will be non-operational or read-only, depending on the recovery completion state you selected (`WITH NORECOVERY` or `WITH STANDBY`). To make the standby database available for reading and writing, the log shipping needs to be broken and the database brought online `WITH RECOVERY`:

1. If possible, run the transaction log backup job on the primary server to perform a final backup of the the source database transaction log. You can do this from the **Jobs** tab in the SQL Backup graphical user interface (GUI).
   If the source database is damaged, the log backup may fail. In this case, edit the SQL Server Agent job (for example, using SQL Server Management Studio) and append `WITH NO_TRUNCATE` to the `BACKUP LOG` command, then run the job again.
2. If there are transaction log backup files remaining in the log shipping shared folder, restore them to the standby database, specifying the option `WITH RECOVERY` when you restore the final backup. For example:

   ```
   "RESTORE LOG [standby database] FROM DISK = 'C:\Backups\log_29022013_160012' WITH
   NORECOVERY"

   "RESTORE LOG [standby database] FROM DISK = 'C:\Backups\log_29022013_160512' WITH
   NORECOVERY"

   "RESTORE LOG [standby database] FROM DISK = 'C:\Backups\log_29022013_161012' WITH
   RECOVERY"
   ```

   You can also use the SQL Backup Restore wizard: on step 1, select **Browse for backup files to restore** and use the **File Browser** to select the backup files. For more information, see Restoring backups.

   If there are no transaction log backup files remaining, recover the standby database using the native SQL Server `RESTORE` statement, for example:

   ```
   RESTORE DATABASE [database_name] WITH RECOVERY
   ```

3. Reconfigure connecting resources to use the standby database.

If any SQL users were transferred via the transaction log backups, they will either not exist as logins on the standby SQL Server or they will use invalid security identifiers (SIDs) from the source SQL Server's login. To reconcile the SIDs:

1. Make a list of the users in the database's **Users** container (in SQL Server Management Studio, open Object Explorer and select **Server** > **Databases** > **<standby database>** > **Security** > **Users**).
2. Open the SQL Server's **Security** container and ensure that the users have corresponding logins on the SQL Server.
3. Reconcile all database users against the security identifier on the standby server using this query:

   ```
   USE [MyDatabase]
   exec sp_change_users_login 'Update_One', 'MyDBUser', 'MyDBLogin'
   ```

   where *Update_One* links the specified user (*MyDBUser)* in the database (*MyDatabase*) to the existing SQL Server login (*MyDBLogin*). The user and the login are usually the same. For more information about *sp_change_users_login*, refer to you SQL Server documentation

To re-establish log shipping, restore the standby database from a new full backup of the source database using `WITH NORECOVERY` or `WITH STANDBY`. You can use the Restore wizard to do this.

Object level recovery

> SQL Object Level Recovery Pro is only available in SQL Backup 6.2 and later.

Sometimes you may need to restore individual database objects rather than a complete backup. For example, a user may have accidentally dropped a lookup table, or deleted important data from a production database.

To restore the table from a backup, you would normally have to restore the entire database to an appropriate test server, and then produce a table creation script and data insert statements to recreate the table in the production database. For large databases, the full restore may take a long time, and could require a lot of free disk space on the test server.

SQL Object Level Recovery Pro (included with SQL Backup) enables you to extract individual database objects from a backup file, and recover them to a database of your choice.

By recovering only the objects you specify, SQL Object Level Recovery Pro can save you a considerable amount of time when compared to restoring a full backup. You can also save space, as you do not need to restore a complete database.

SQL Object Level Recovery Pro temporarily stores a recovery script on the computer it runs on, for which some disk space is required. The recovery script contains the SQL needed to recreate only those database objects that you select for recovery. If you have chosen to recover tables, this includes the table data.

- For an overview of how to use SQL Object Level Recovery Pro, see Recovering objects.
- SQL Object Level Recovery Pro has some limitations on the type of objects it can recover, and the type of backup files it can recover objects from. See Limitations for more information.

Recovering objects

> SQL Object Level Recovery Pro is only available in SQL Backup 6.2 and later.

This page describes the basic steps to recover objects from a backup file using SQL Object Level Recovery Pro, including tips to help you get the most from the application.

Starting SQL Object Level Recovery Pro

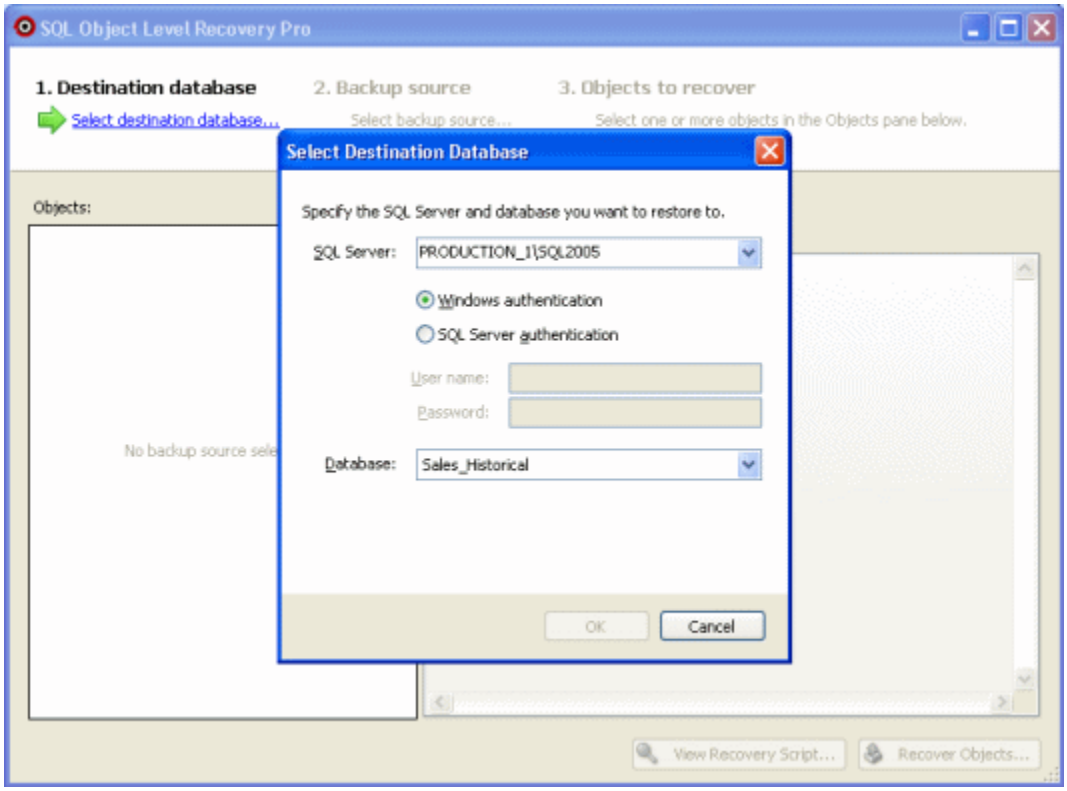
You can start SQL Object Level Recovery Pro:

- from SQL Backup
  Right-click a server or database in the **Registered SQL Servers** pane, and select **Object Level Recovery**.
  This option is not available if the server you have selected does not have a valid SQL Backup license.
- from Windows Explorer
  The application is installed to *C:\Program Files\Red Gate\SQL Backup 7\SQBObjectLevelRecovery\SQLObjectLevelRecoveryPro.exe* by default.
  You can copy the contents of the *SQBObjectLevelRecovery* folder to any server, and run SQL Object Level Recovery Pro independently of SQL Backup. This is useful, for example, if you are recovering objects from a large backup, and want to avoid streaming large amounts of data across the network. The server you are recovering objects to must have a valid SQL Backup license.
- from the Windows Start menu

You can recover objects in four steps:

1. Select a destination database
2. Select a backup source
3. Select objects to recover
4. Recover the objects

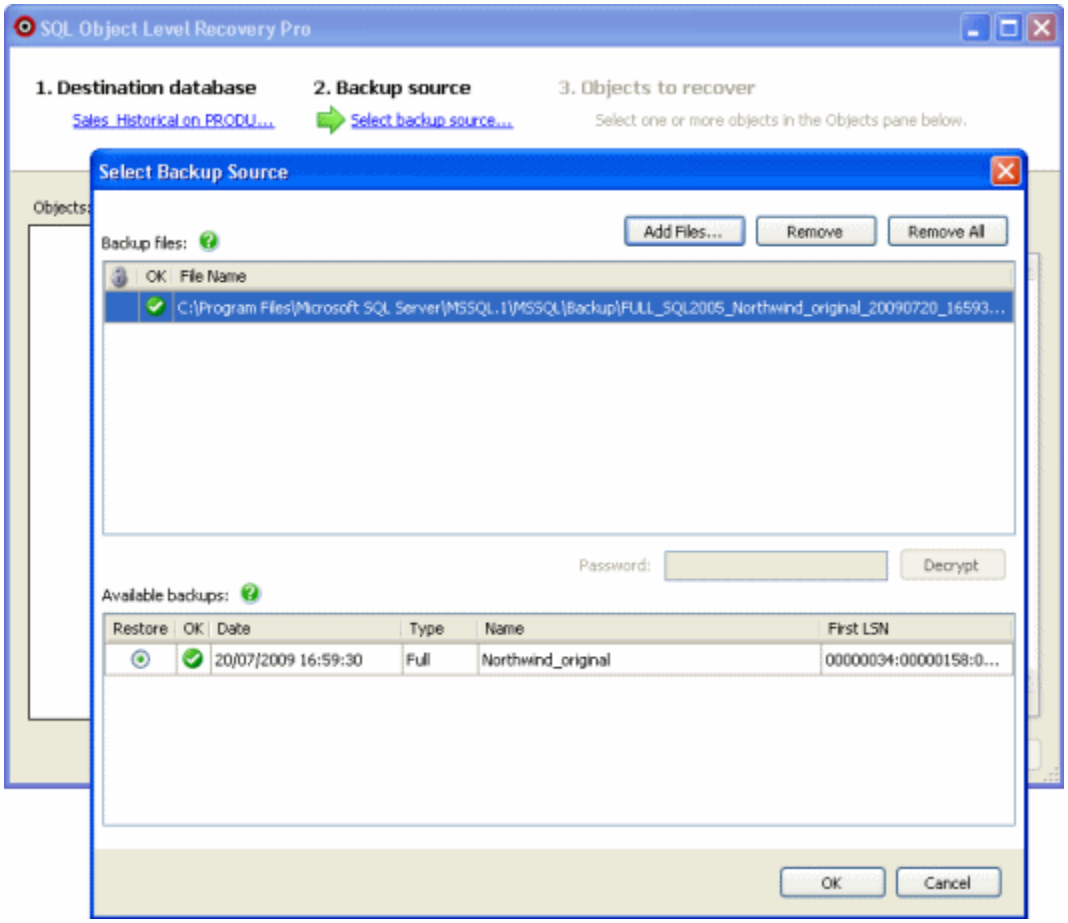
1. Select a destination database



- if you have already chosen a database in SQL Backup, this is pre-selected as the destination database; you can continue to step 2 to select a backup source
- the server you select must be licensed for use with SQL Backup
- the destination database must be open, and in a suitable state for receiving recovered objectsFor example, a database that is currently

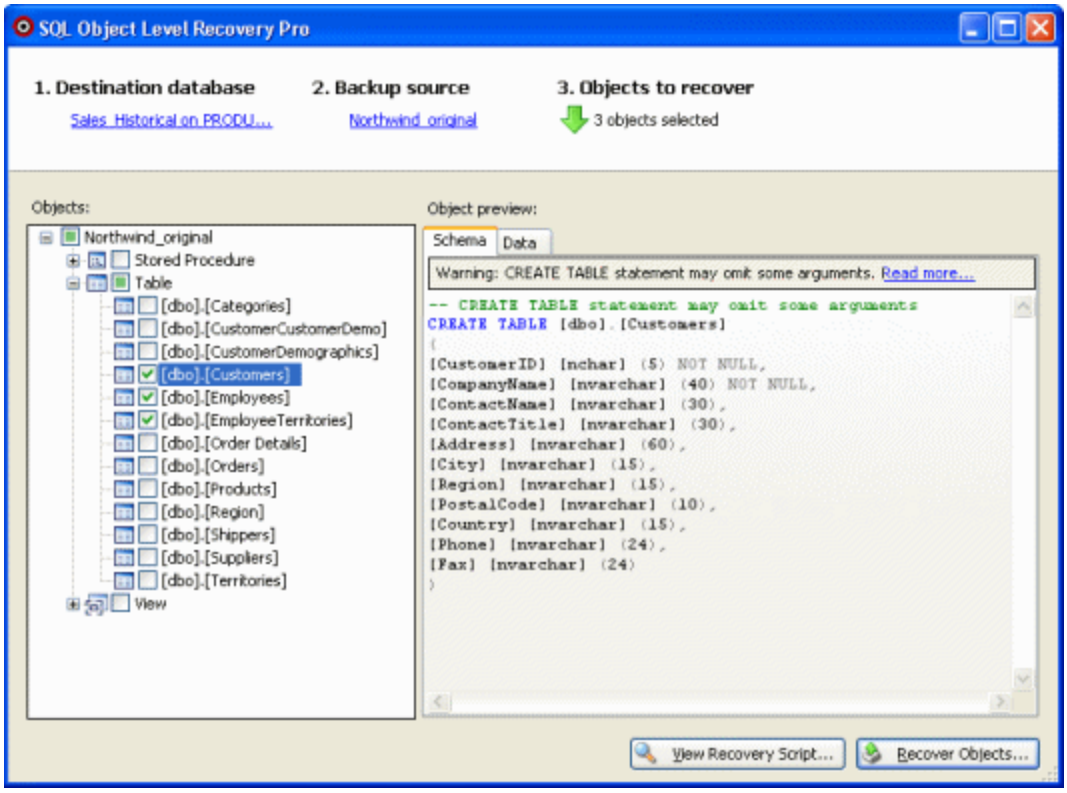
being restored cannot be used as a destination database for object level recovery.
- you must have appropriate security permissions to create objects in the destination database

2. Select a backup source



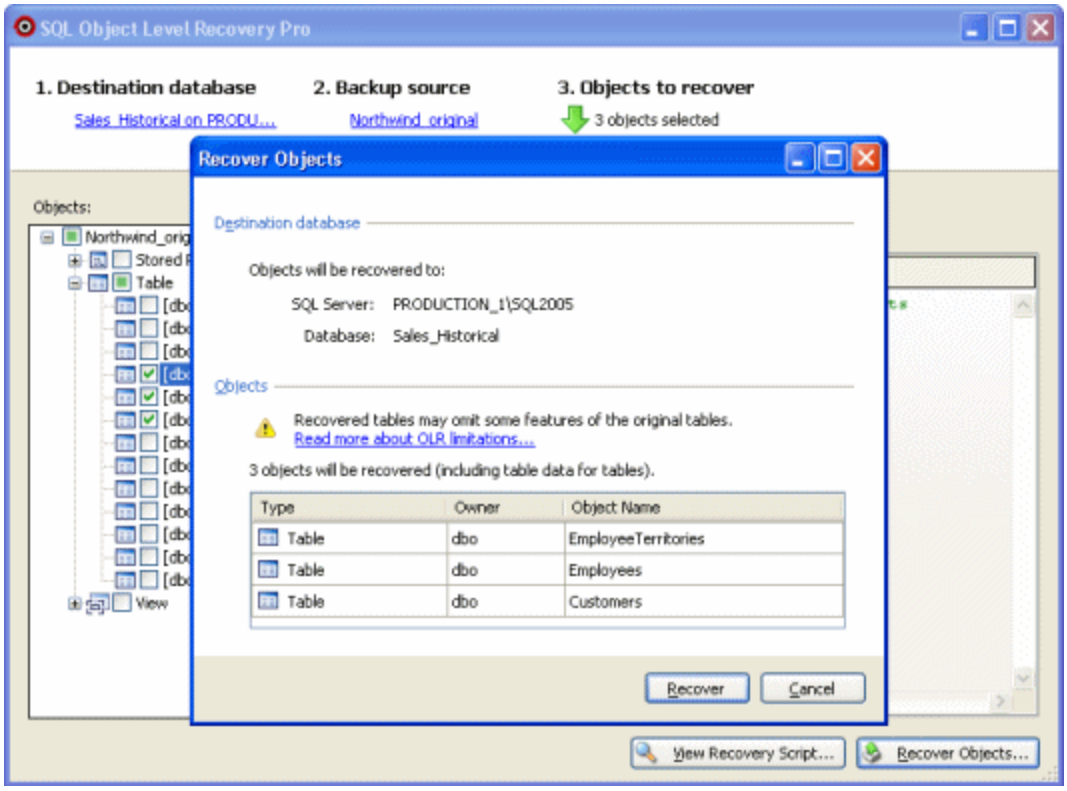
- you can add full backup files or differential backup files (SQL Backup .sqb files only)
  If you add a differential backup file, you will also need to add the related full backup file.
- you can add files for backups that are split across several files
  You must add every file from the backup set.
- if there are problems validating the file or backup (a
  
  symbol is shown in the **OK** column), select the file or backup to see a detailed error message

3. Select objects to recover



- SQL Object Level Recovery Pro does not automatically check for dependencies between objects, but objects are recovered in an order that reduces the possibility of errors caused by missing dependents.
  Depending on the contents of the destination database, you may have to select several objects to ensure that the necessary object dependencies are maintained.
- the **Data** tab (available for table objects only) shows the first 100 rows of data in a table
  Data for certain data types (for example, varchar and nvarchar) may not be shown if it is too large to display.
- **View Recovery Script** displays a complete object and data recovery script for the objects you have selected
- SQL Object Level Recovery Pro does not support all possible arguments of the CREATE TABLE statement
  For more information, see Limitations.

4. Recover the objects



- when you click **Recover**, the **Recovering Objects** dialog shows the status of each object you selected for recoverySelect objects in the Progress list to show a detailed error message for objects that could not be recovered. These objects are marked with a
- SQL Object Level Recovery Pro will not alter or overwrite existing objects in the destination database
  See Limitations for more information.
- object level recovery activities are not shown on the SQL Backup time line, or on the SQL Backup Activity History tab

Limitations

> SQL Object Level Recovery Pro is only available in SQL Backup 6.2 and later.

SQL Object Level Recovery Pro has some limitations. For example, dependencies between objects are not handled automatically, and some features of SQL Server tables are not supported. The following sections provide more detailed information about these limitations:

Refer to your SQL Server documentation for detailed information about specific object types, data types, and table arguments.

For more complex recovery scenarios, you should consider using Redgate SQL Compare and SQL Data Compare. These enable you to compare the contents (object schema, and data) of SQL Backup .sqb files with a live database, and then synchronize the database with the backup file contents while maintaining object dependencies.

Existing objects

Objects that already exist in the destination database will not be modified or overwritten by SQL Object Level Recovery Pro. Attempting to recover such objects results in an error.

It is usually safer to recover an object to a test or staging database first, and then transfer the object to its final destination database manually. If you want to recover an object directly from a backup file to its final destination database, you will have to drop the object first. Make sure you have a recent valid backup of the object before you drop it.

Object dependencies

Objects you attempt to recover may have dependencies on other objects in the destination database. For example, a view may refer to several tables; successful recovery of the view depends on these tables being present in the destination database.

SQL Object Level Recovery Pro does *not* attempt to resolve dependencies automatically. To avoid dependency errors, you may need to recover multiple dependent objects together.

If you have selected objects of more than one type, they are recovered in the following order:

1. SCHEMA
2. TYPE (user defined type)
3. XML SCHEMA COLLECTION
4. FUNCTION
5. TABLE
6. VIEW
7. PROCEDURE (stored procedure)

Recovering objects in this order reduces the possibility of failures caused by dependencies on missing objects.

Supported backup types

You can use SQL Object Level Recovery Pro with full backups and differential backups. To recover objects from a differential backup, you will also need to provide the associated full backup.

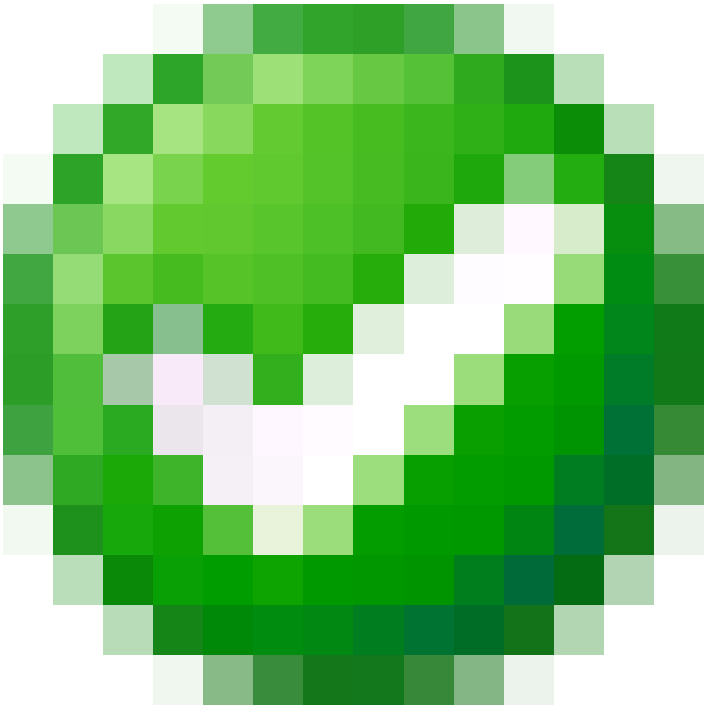
SQL Object Level Recovery Pro does not support:

- filegroup backups
- transaction log backups
- backups from databases that use Transparent Data Encryption (TDE)
- native SQL Server backups (.bak files)

Supported object types

Object types marked


can be recovered from SQL Backup .sqb files. Other object types are not supported.

Object type	Supported
ASSEMBLY	
ASYMMETRIC KEY	
CERTIFICATE	
CONTRACT	
DEFAULT	
EVENT NOTIFICATION	
FULLTEXT CATALOG	
FULLTEXT STOPLIST	
FUNCTION	
INDEX	
MESSAGE TYPE	
PARTITION FUNCTION	
PARTITION SCHEME	

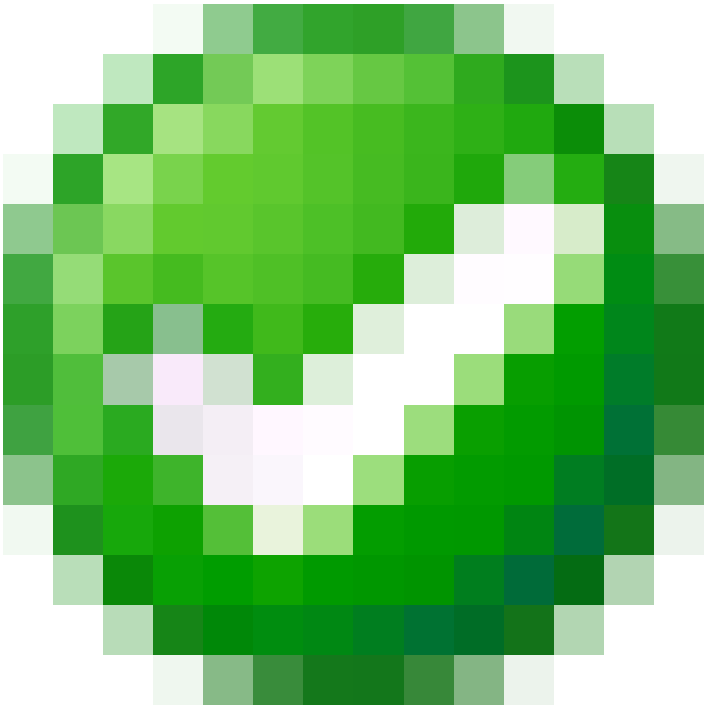
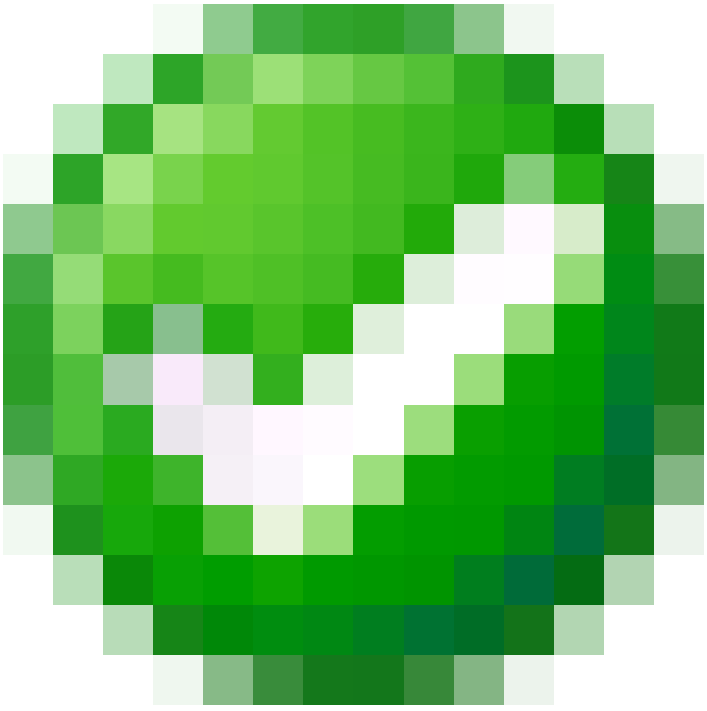
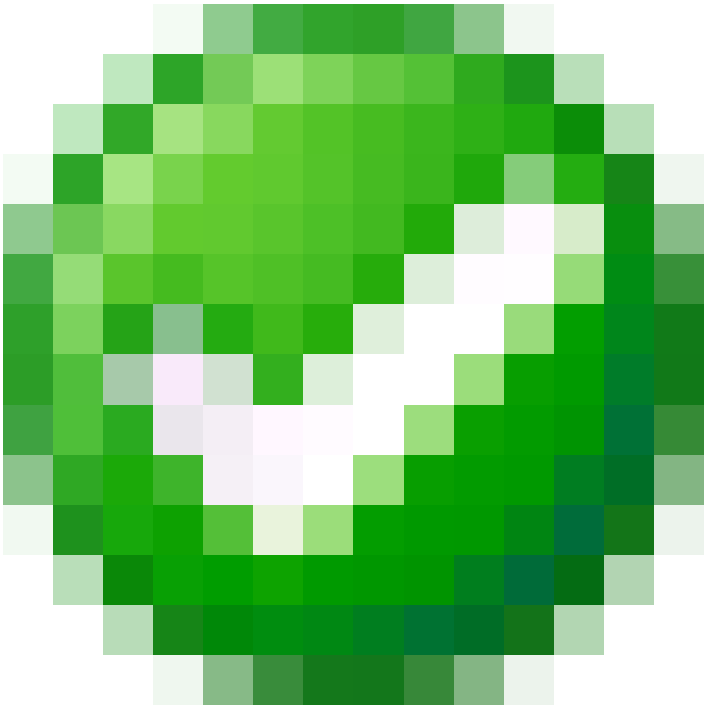
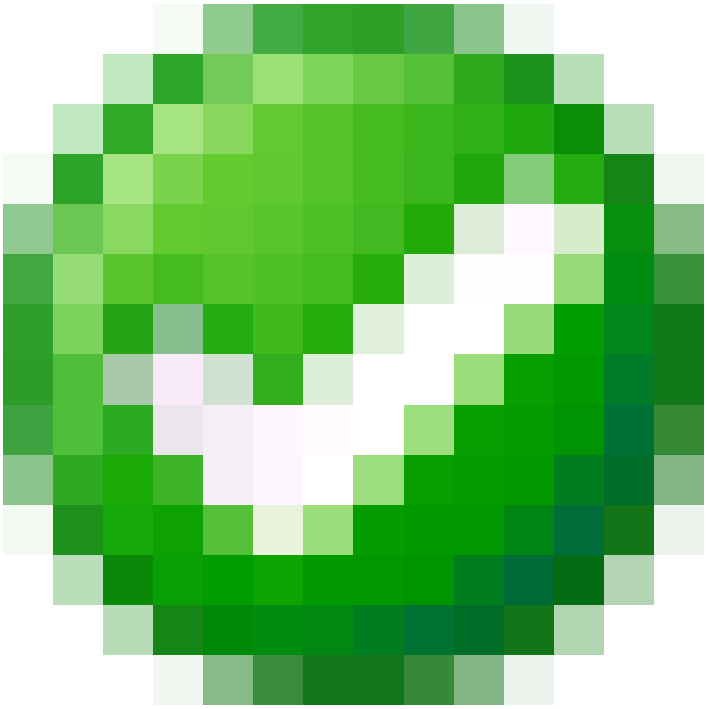
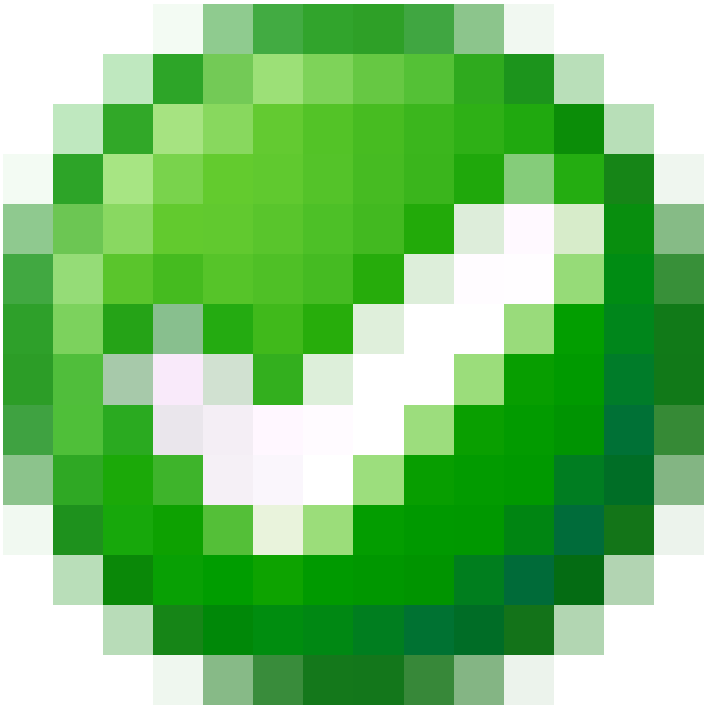
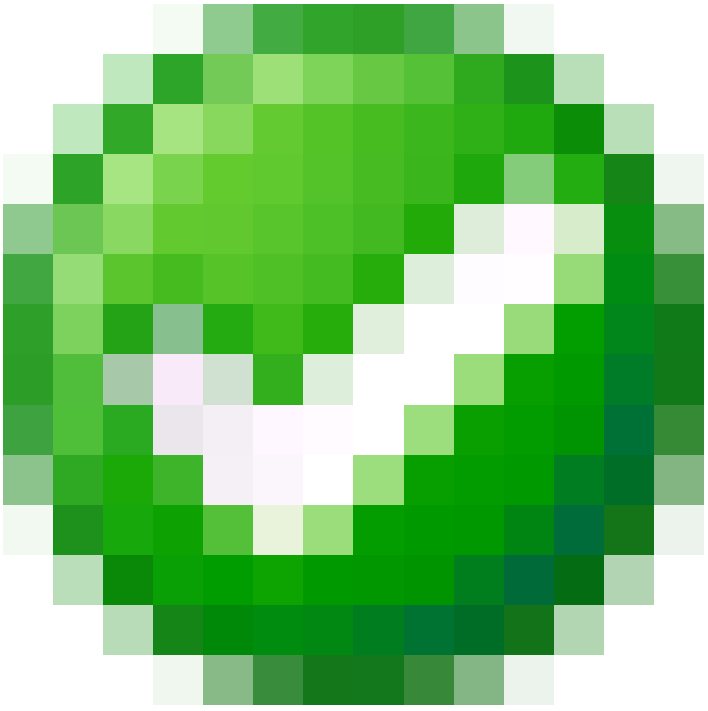
PROCEDURE (stored procedure)	
QUEUE	
REMOTE SERVICE BINDING	
ROLE	
ROUTE	
RULE	
SCHEMA	
SERVICE	
SYMMETRIC KEY	
SYNONYM	

TABLE	
TRIGGER	
TYPE (user defined type)	
USER	

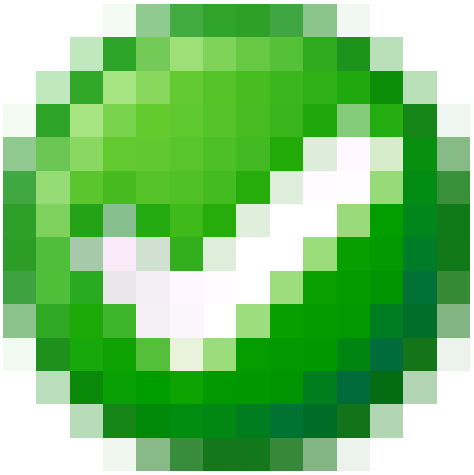
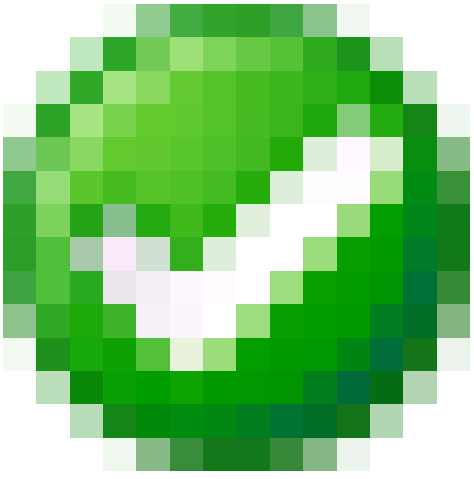
VIEW	
XML SCHEMA COLLECTION	

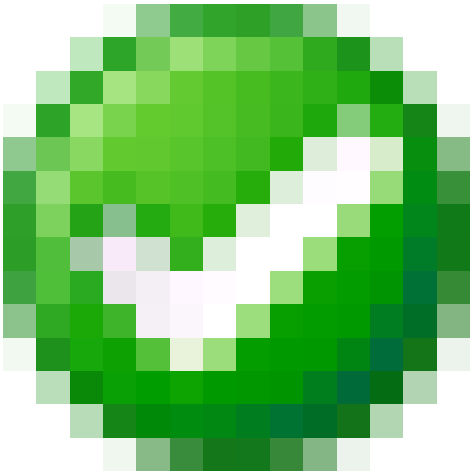
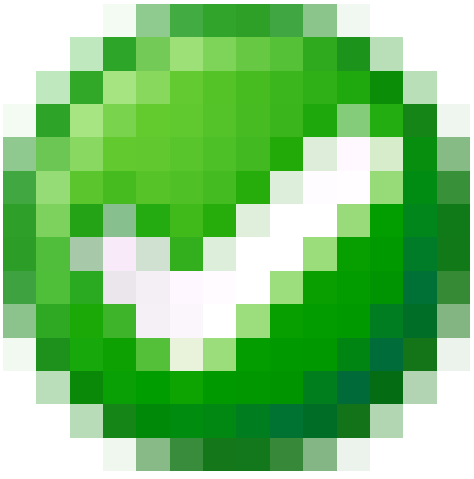
Supported data types

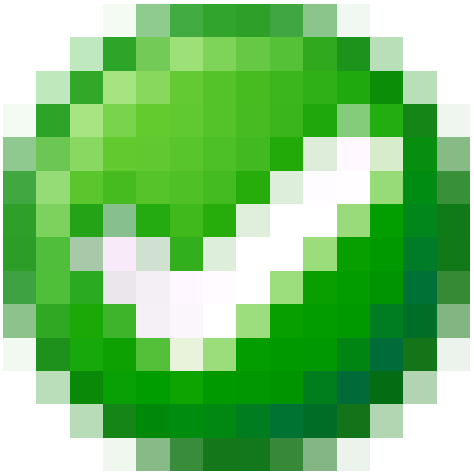
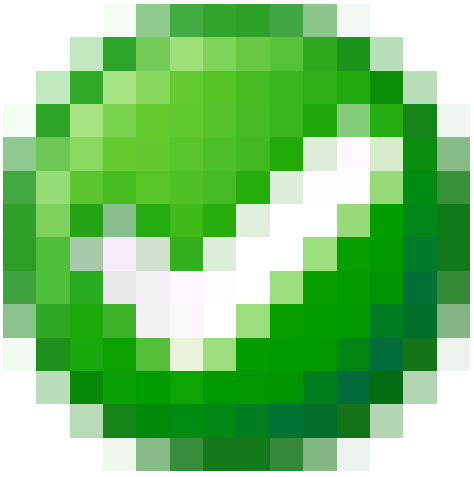
Table data with types marked

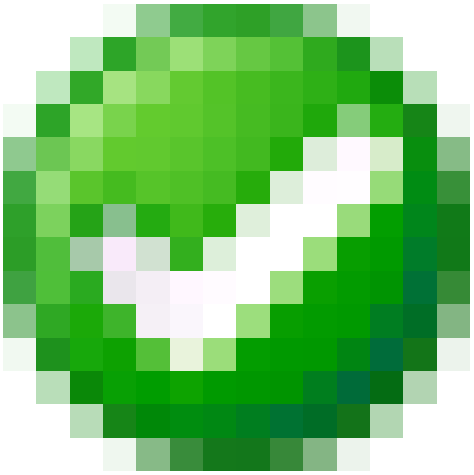
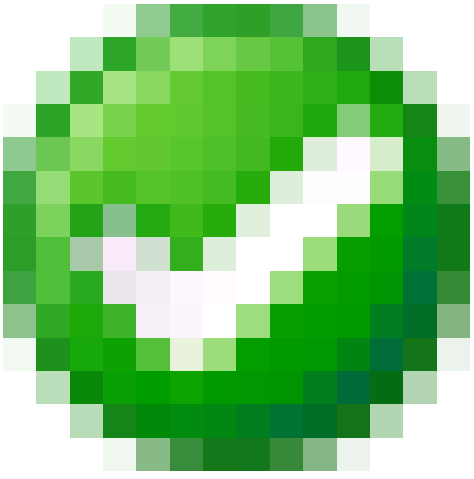
can be recovered from SQL Backup .sqb files.

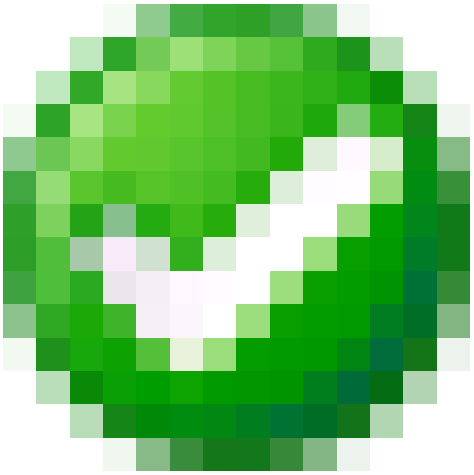
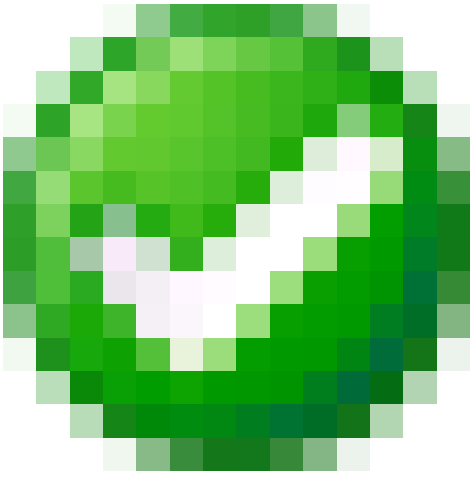
Group	Data type	Supported

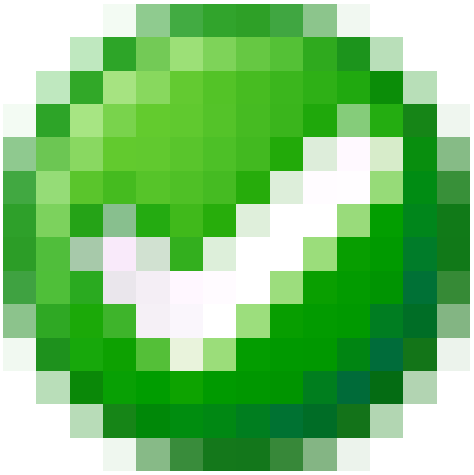
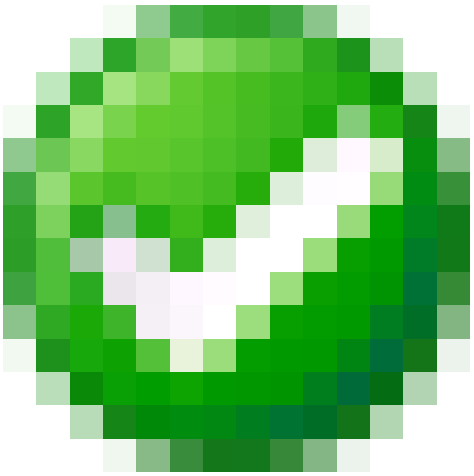
Exact numerics	bit	
	tinyint	
	smallint	

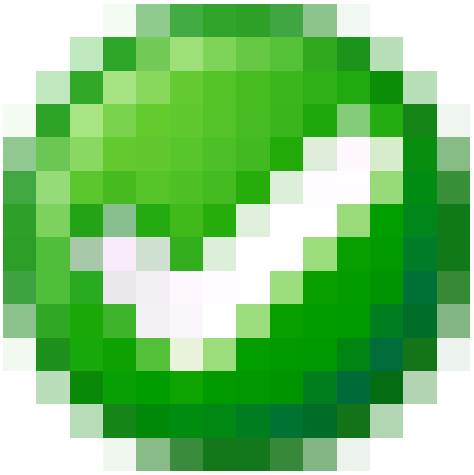
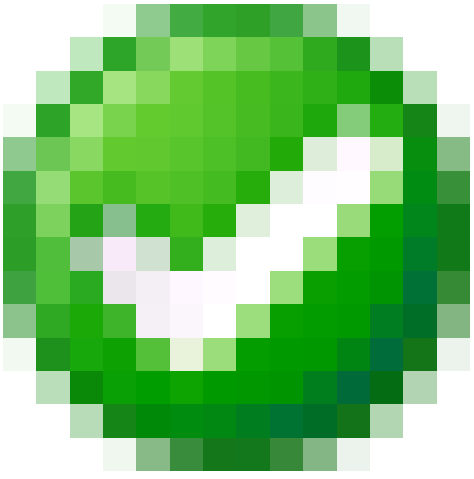
int	
bigint	
numeric	

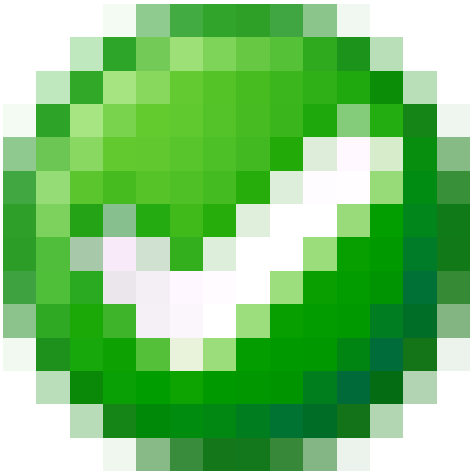
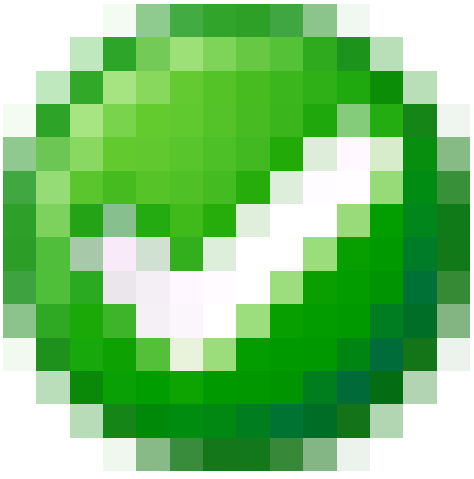
	decimal	
	smallmoney	
	money	

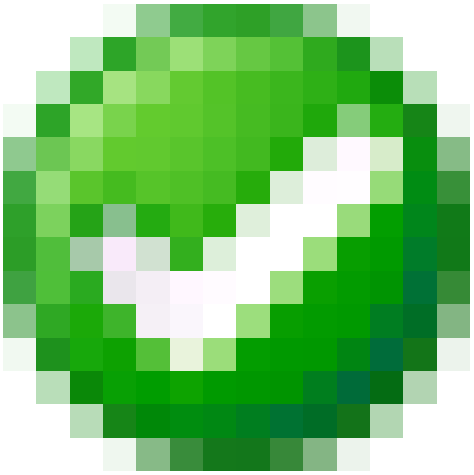
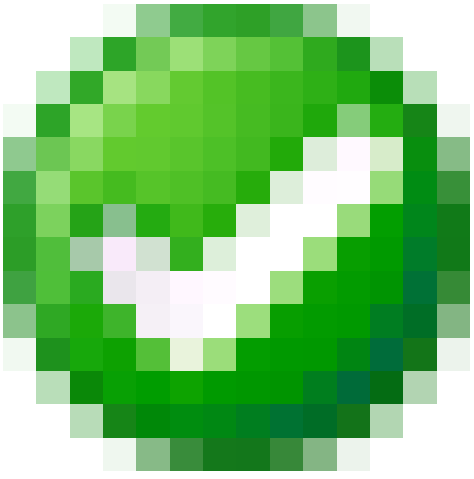
Approximate numerics	float	
	real	
Date and time	datetime	

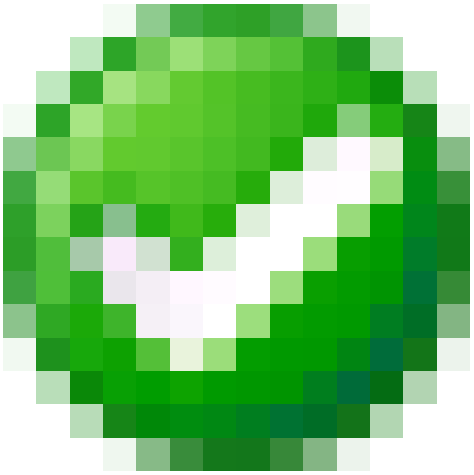
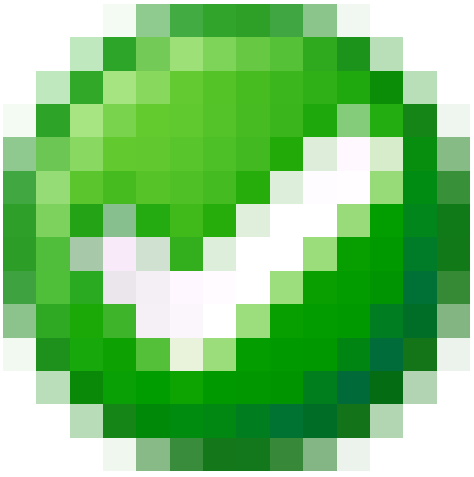
smalldatetime	
date	
time	

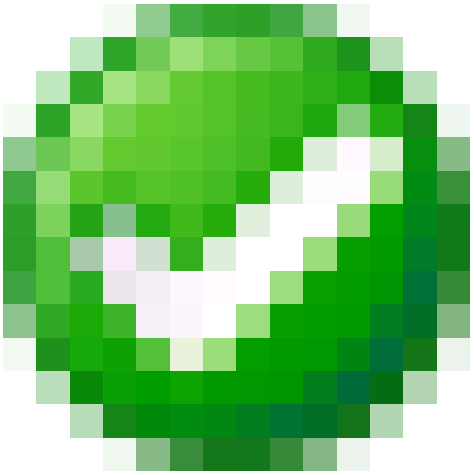
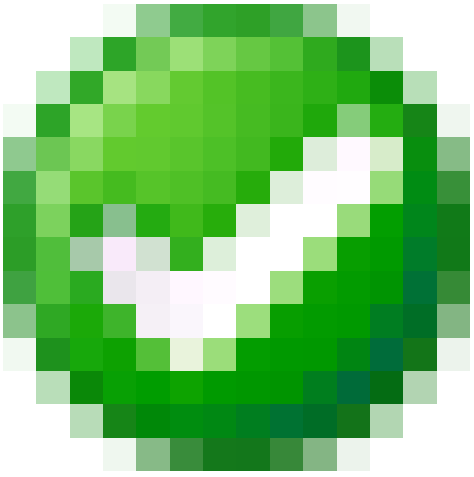
	datetimeoffset	
	datetime2	
Character strings	char	

	varchar / varchar(max)	
	text	
Unicode character strings	nchar	

	nvarchar / nvarchar(max)	
	ntext	
Binary strings	binary	

	varbinary / nvarbinary(max)	
	image	
Other data types	sql_variant	

	timestamp	
	uniqueidentifier	
	xml	

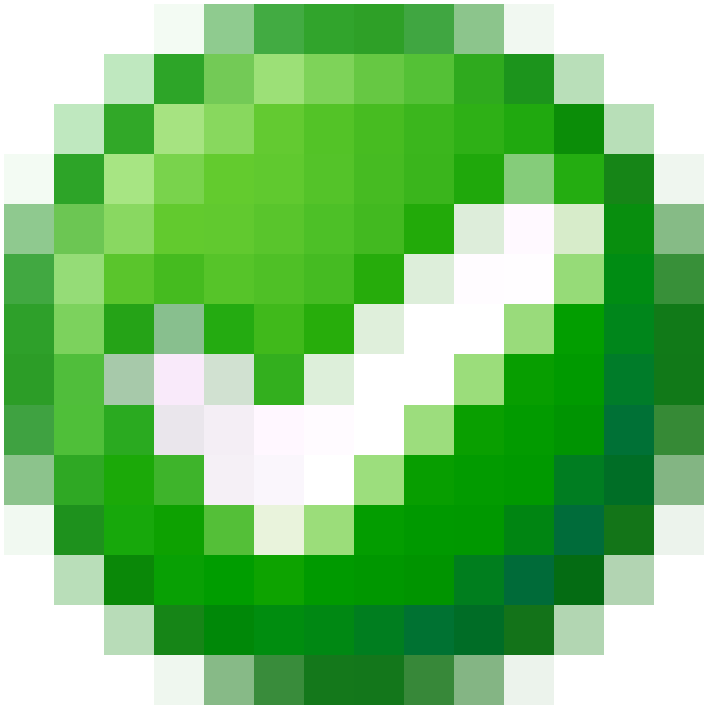
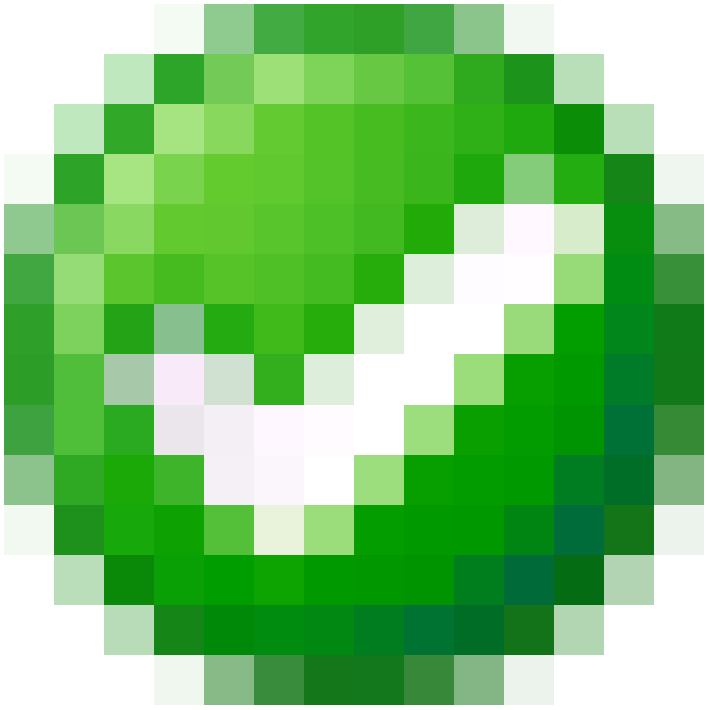
CLR data types	hierarchyid	
Spatial data types	geometry	
	geography	

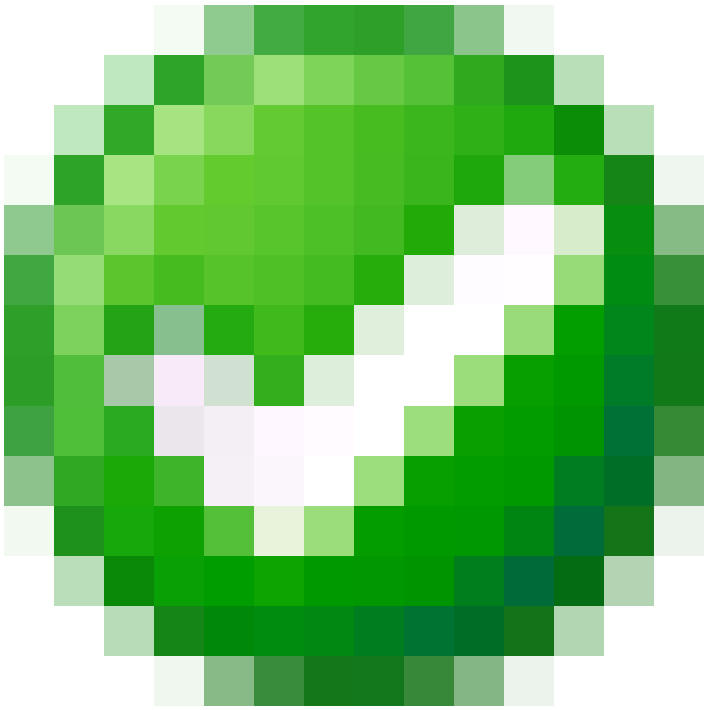
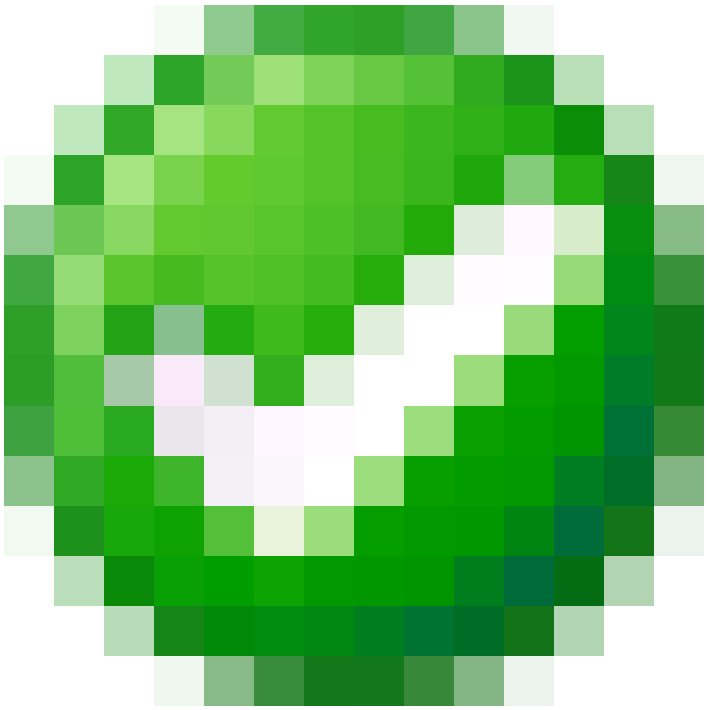
Supported CREATE TABLE arguments

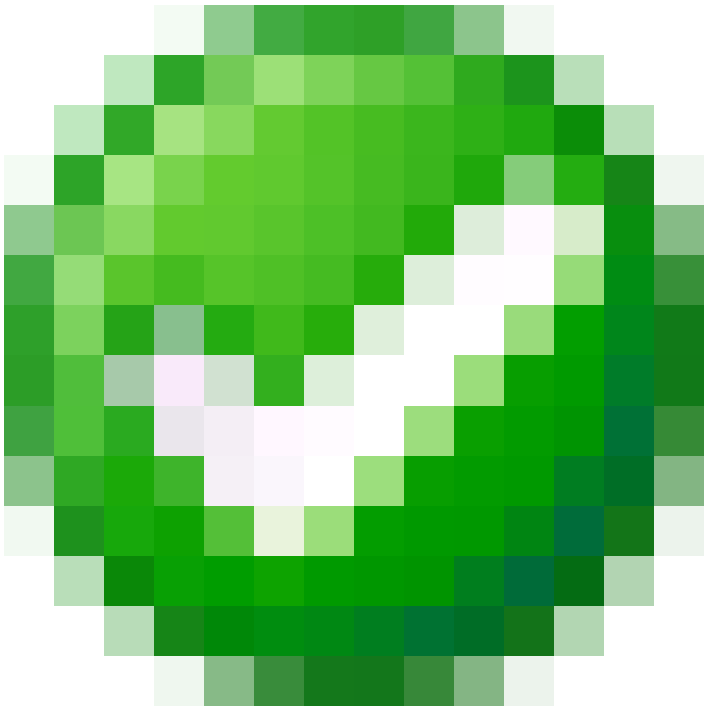
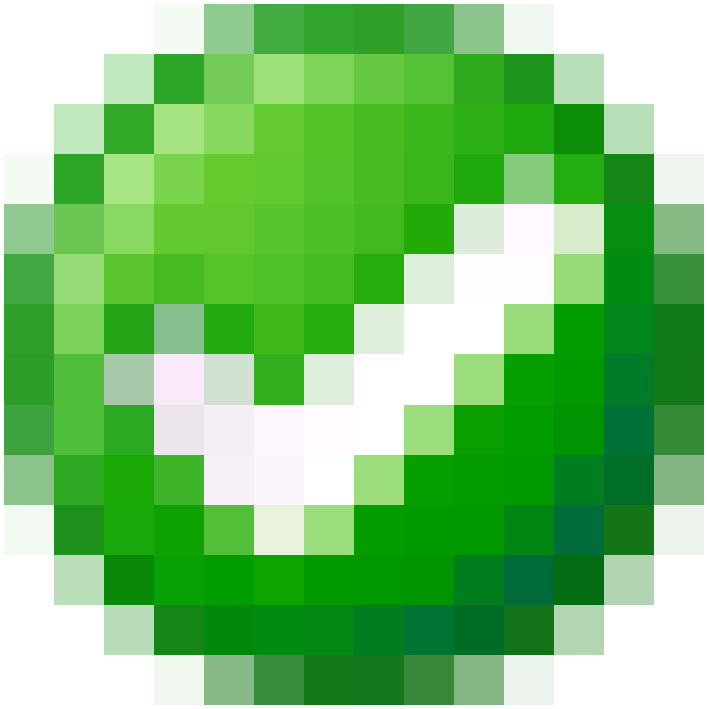
`CREATE TABLE` arguments marked

are supported. All other `CREATE TABLE` arguments are ignored. For example, if the table to be recovered includes a `FOREIGN KEY ... REFERENCES` argument, this will not be created in the recovered table.

Argument	Supported
ALLOW_PAGE_LOCKS	

ALLOW_ROW_LOCKS	
CLUSTERED	
COLLATE	
computed_column_expression	
CONSTRAINT	
CONTENT	
DATA_COMPRESSION	
DEFAULT	
DOCUMENT	
FILESTREAM_ON	

FOREIGN KEY ... REFERENCES	
IDENTITY	
IGNORE_DUP_KEY	
NONCLUSTERED	
NOT FOR REPLICATION	
NULL	
ON filegroup	
ON partition scheme	
ON DELETE	
ON UPDATE	

PAD_INDEX	
PERSISTED	
PRIMARY KEY	
RANGE	
ROWGUIDCOL	
SPARSE	
STATISTICS_NORECOMPUTE	
TEXTIMAGE_ON	
UNIQUE	
WITH FILLFACTOR	

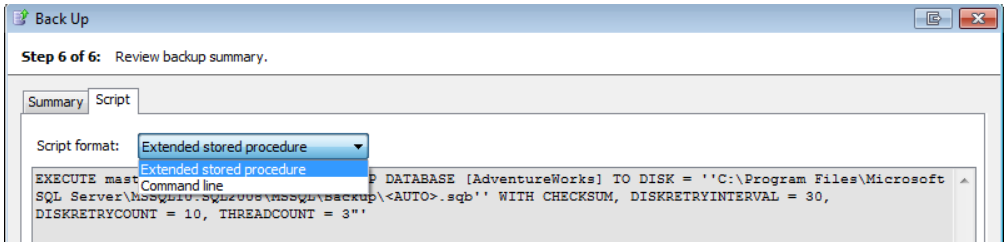
XML COLUMN_SET FOR ALL_SPARSE_COLUMNS	

Scripting SQL Backup

SQL Backup provides a command line interface and extended stored procedure so that you can run backups and restores using scripts:

- The command line executable *SQLBackupC.exe* allows you to run your backup and restore operations in script and batch files.
- The extended stored procedure, *sqlbackup*, allows you to run your backup and restore operations using an application such as Microsoft SQL Server Management Studio, or a database connectivity layer such as ADO or OLE DB.

Before you use the command line or extended stored procedure, you are recommended to use the backup and restore wizards until you are familiar with the SQL Backup features and tools. On the final step of each wizard, you can view the backup or restore script in either command line or extended stored procedure format:



> You cannot use the command line to back up or restore databases on remote servers; you must use the extended stored procedure or the wizards.

Using the command line

The SQL Backup command line executable is *SQLBackupC.exe*. It is located in the folder in which you installed the SQL Backup server components; for example, *C:\Program Files\Red Gate\SQL Backup\<instance name>*. For details, see Installing the server components on a SQL Server instance.

When running SQL Backup from the command line, you can use both connection parameters and process parameters. Use a command of the format:

```
SQLBackupC.exe <parameters>
```

The following connection and process parameters are available:

Parameter	Syntax	Description
Connection Parameters		
`-U`	`-U <SQL_Server_user_name>`	Specifies the SQL Server user name to use to log in if you are using SQL Server authentication.
`-P`	`-P <SQL_Server_password>`	Specifies the password to use to log in if you are using SQL Server authentication.
`-I`	`-I <SQL_Server_instance>`	Specifies the name of the SQL Server instance you want to log on to. If no value is entered, the default instance is used. To explicitly specify the default, use `-I (local)`
Process Parameters		
`-SQL`	`-SQL "T-SQL_statement"`	Identifies the Transact-SQL statement for the backup or restore operation. For details of the standard commands you can use, and the extensions to the standard Transact-SQL syntax, see the following pages: • The BACKUP command • The RESTORE command • The RESTORE SQBHEADERONLY command • The CONVERT command • The ERASE command

`-USE`	`-USE "template_name"`	Indicates that you want to load the configuration values from the specified template. You save the template using the Back Up wizard.

Examples

The following command restores the *pubs* database on the SQL Server default instance using Windows authentication:

```
SQLBackupC.exe -SQL "RESTORE DATABASE pubs FROM DISK ='C:\Backups\pubs.sqb' WITH
PASSWORD = 'BackupPassword'"
```

The following command takes full, encrypted backup of the *pubs* database on a named instance, *ServerInstance2*, using SQL Server authentication

```
SQLBackupC.exe -I ServerInstance2 -U sa -P MyPassword -SQL "BACKUP DATABASE pubs TO
DISK ='C:\Backups\pubs.sqb' WITH PASSWORD = 'BackupPassword'"
```

The following command loads a template for a backup job called Daily Full Backup on the SQL Server default instance using SQL Server authentication:

```
SQLBackupC.exe -USE "Daily Full Backup" -U sa -P MyPassword
```

Accessing SQLBackupC.exe from other locations

You can set up your system so that you can access the SQL Backup command line executable *SQLBackupC.exe* from your current location without typing the full path.

To make the executable accessible to all users, add the SQL Backup installation path to the System PATH environment variable; to make the executable accessible only to the current user, add the SQL Backup installation path to the User PATH environment variable. You can access the **Environment Variables** property page from the **Advanced** tab of the **System Properties**. Refer to your Microsoft Windows documentation for details.

Using the extended stored procedure

The extended stored procedure is installed when you install the SQL Backup server components on your SQL Server instance. For details, see Installing the server components on a SQL Server instance.

When running the extended stored procedure you can use both connection and process paramenters, in the form:

```
master..sqlbackup 'parameters'
```

You can use connection parameters with the extended stored procedure to back up or restore databases on a different SQL Server instance. If you do not specify connection parameters, the extended stored procedure will run on the instance you are already connected to.

The following connection and process parameters are available:

Parameter	Syntax	Description
Connection Parameters		
`-U`	`'-U <SQL_Server_user_name>'`	Specifies the SQL Server user name to use to log in if you are connecting to a different SQL Server instance using SQL Server authentication.
`-P`	`'-P <SQL_Server_password>'`	Specifies the password to use to log in if you are connecting to a different SQL Server instance using SQL Server authentication.

`-I`	`'-I <SQL_Server_instance>'`	Specifies the name of the SQL Server instance you want to log on to. If no value is entered, the SQL Server you are connected to is used. To explicitly specify the default, use `master..sqlbackup '-I (local)'`
Process Parameters		
`-SQL`	`'-SQL "T-SQL_statement"'`	Identifies the Transact-SQL statement for the backup or restore operation. For details of the standard commands you can use, and the extensions to the standard Transact-SQL syntax, see the following pages: • The BACKUP command • The RESTORE command • The RESTORE SQBHEADERONLY command • The CONVERT command • The ERASE command
`-USE`	`'-USE "template_name"'`	Indicates that you want to load the configuration values from the specified template. You save the template using the Back Up wizard.

Note that the set of parameters must be enclosed by a pair of single quotes. Therefore, wherever you would use a single quote to delimit variables when using the command line, for the extended stored procedure you must use **two** single quotes so that SQL Server does not interpret it as a string delimiter. For example:

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASE [pubs] TO DISK = ''C:\Backups\pubs.sqb'' WITH
PASSWORD = ''MyPassword'' " '
```

Alternatively, to make your code easier to read, you can use square brackets instead of two single quotes to delimit variables:

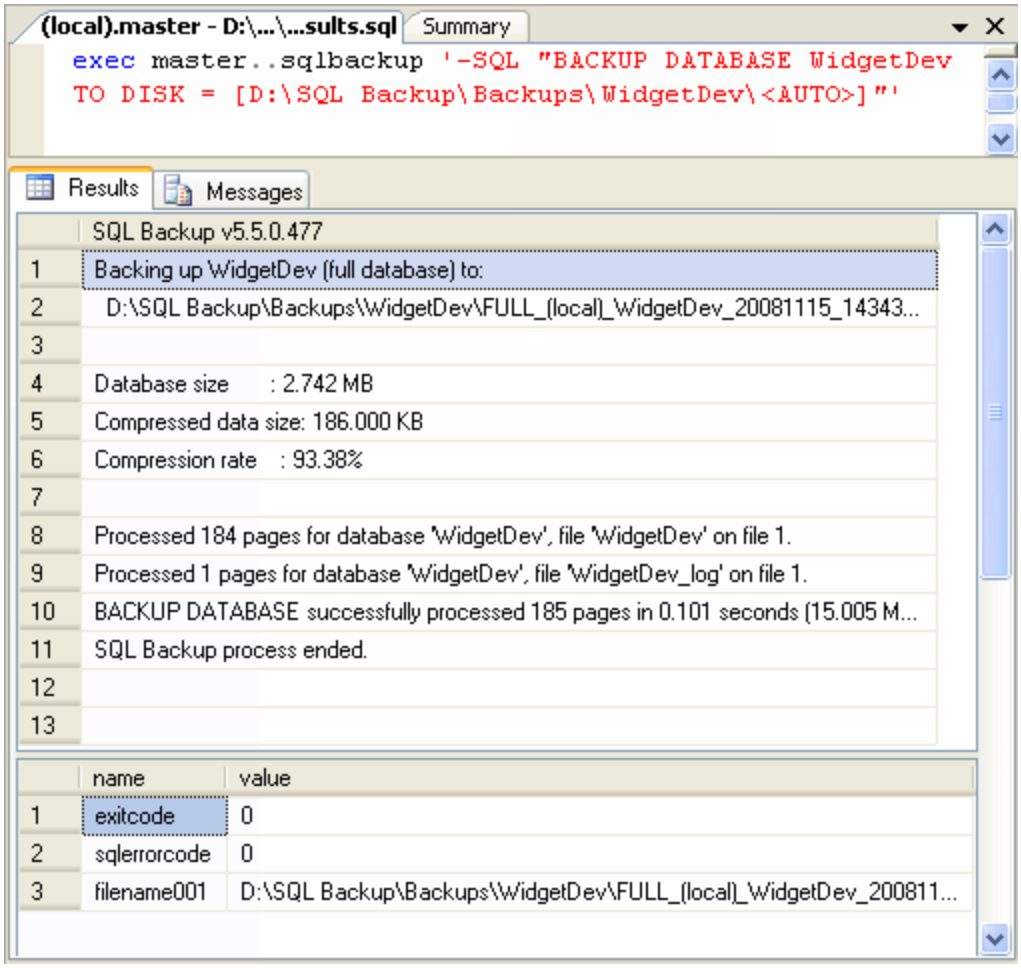
```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASE [pubs] TO DISK = [C:\Backups\pubs.sqb] WITH
PASSWORD = [MyPassword] " '
```

Feedback from the extended stored procedure

The extended stored procedure returns results in datasets:

- the SQL Backup messages are returned in the first dataset
- the SQL Backup exit code, the SQL Server error code, and the names of all the backup files are returned in the second dataset

For example:

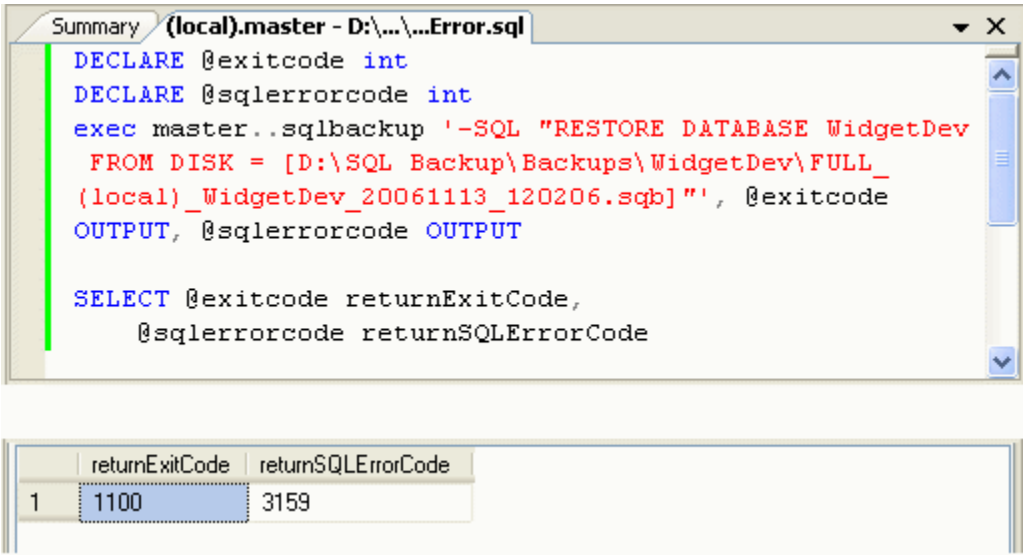


To omit the second dataset from the results, use the `SINGLERESULTSET` keyword. This is useful if you want to process the first dataset within a Transact-SQL script. Transact-SQL scripts only allow manipulation of the returned data if a single dataset is returned.

To retrieve error codes from the `sqlbackup` extended stored procedure, call the extended stored procedure with two output parameters of type *integer*. SQL Backup returns:

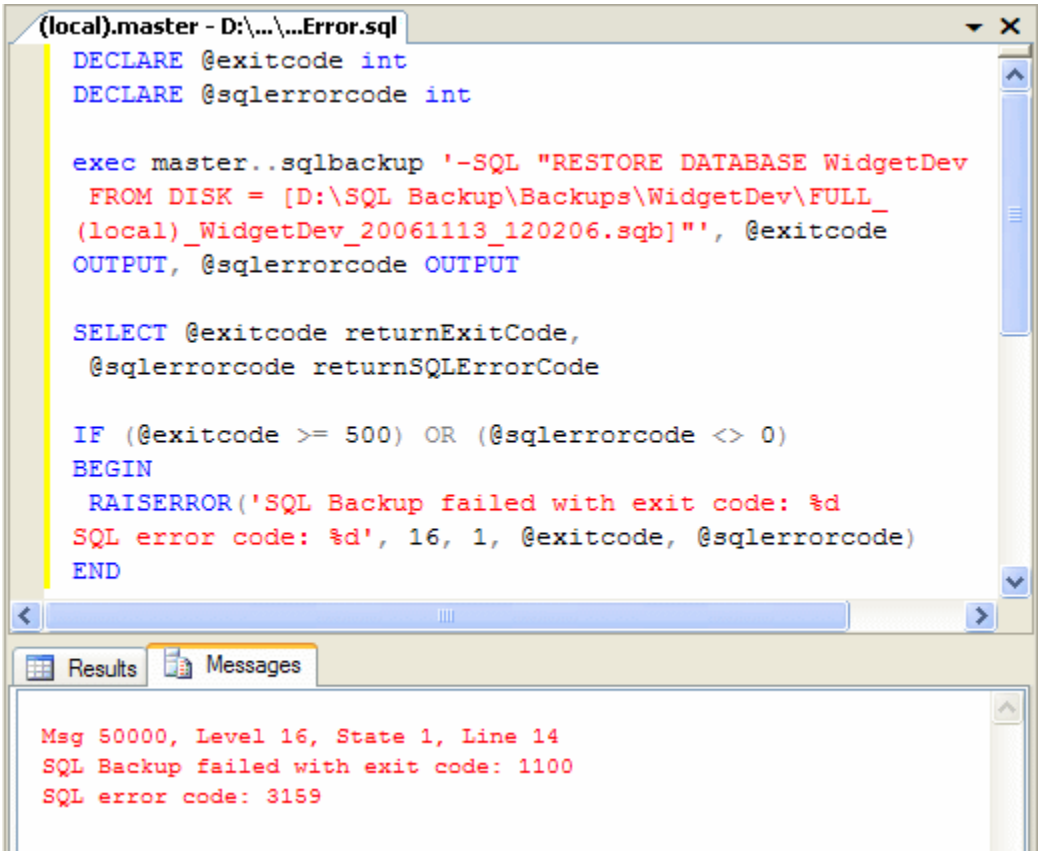
- the SQL Backup exit code in the first output parameter (or '0' if there are no SQL Backup errors)
- the SQL Server error code in the second output parameter (or '0' if there are no SQL Server errors)

For example:



Note that the `sqlbackup` extended stored procedure itself will not generate an error in the above example. To force the call to the extended

stored procedure to generate an error when a SQL Backup error or SQL Server error is returned, you must use the `RAISERROR` keyword. For example:



Now, if the `sqlbackup` extended stored procedure encounters a SQL Backup error (code 500 or higher), the query will fail.

For more information on the warning codes and error codes, see SQL Backup warnings 1 - 499 and SQL Backup errors 500 - 5292.

The BACKUP command

Use the BACKUP command with the SQL Backup -SQL parameter to back up one or more databases, transaction logs, or filegroups using the command line or extended stored procedure.

- Syntax provides the grammar for the BACKUP command.
- Arguments describes the arguments for the BACKUP command.
- WITH options describes the options that can be used in the BACKUP command.
- Examples provides examples of using the command line and extended stored procedure to create backups with a range of options.

> When using the extended stored procedure, the parameter or set of parameters (such as -SQL) must be delimited by single quotes. Therefore, wherever a single quote is used for the arguments below, for the extended stored procedure you must use **two** single quotes so that SQL Server does not interpret it as a string delimiter. See Using the extended stored procedure for more information.

Syntax

The following arguments are available only with SQL Server 2005 and later:

- CHECKSUM / NO_CHECKSUM
- CONTINUE_AFTER_ERROR / STOP_ON_ERROR
- COPY_ONLY
- READ_WRITE_FILEGROUPS

Backing up a database to a single file or split into multiple files

```
BACKUP DATABASE { database_name }
   [FILE = { 'logical_file_name' } [ ,...n ] | FILEGROUP = {'logical_filegroup_name'} ] [ ,...n ]
TO { DISK } = { 'physical_backup_device_name' | '<AUTO>' } [ ,...n ]
[ WITH
   [ [ , ] { CHECKSUM | NO_CHECKSUM } ]
   [ [ , ] COMPRESSION = { 0 | 1 | 2 | 3 | 4 } ]
   [ [ , ] { CONTINUE_AFTER_ERROR | STOP_ON_ERROR } ]
   [ [ , ] COPY_ONLY ]
   [ [ , ] COPYTO = { 'target_folder_name' } [ ,...n ] ]
   [ [ , ] DESCRIPTION = { 'text' } ]
   [ [ , ] DIFFERENTIAL ]
   [ [ , ] DISKRETRYCOUNT = { n } ]
   [ [ , ] DISKRETRYINTERVAL = { seconds } ]
   [ [ , ] ERASEFILES | ERASEFILES_ATSTART = { days | hours{h} | except latest{b} } ]
   [ [ , ] ERASEFILES_REMOTE = { days | hours{h} | except latest{b} } ]
   [ [ , ] FILECOUNT = { n } ]
   [ [ , ] FILEOPTIONS = { 1 | 2 | 3 | 4 | 5 | 6 | 7 } ]
   [ [ , ] INIT ]
   [ [ , ] KEYSIZE = { 128 | 256 } ]
   [ [ , ] LOG_ONERROR ]
   [ [ , ] LOG_ONERRORONLY ]
   [ [ , ] LOGTO = { 'target_folder_name' | 'file_name' } ]
   [ [ , ] MAILTO = { 'recipients' } ]
   [ [ , ] MAILTO_NOLOG ]
   [ [ , ] MAILTO_ONERROR = { 'recipients' } ]
   [ [ , ] MAILTO_ONERRORONLY = { 'recipients' } ]
   [ [ , ] MAXDATABLOCK = { 65536 | 131072 | ... | 2097152 } ]
   [ [ , ] MAXTRANSFERSIZE = { 65536 | 131072 | ... | 1048576 } ]
   [ [ , ] MIRRORFILE = { 'physical_backup_device_name' } ] [ ,...n ]
   [ [ , ] NAME = { 'backup_set_name' } ]
   [ [ , ] NOCOMPRESSWRITE | NOWRITE ]
   [ [ , ] NOLOG ]
   [ [ , ] PASSWORD = { 'password' } ]
   [ [ , ] SINGLERESULTSET ]
   [ [ , ] SQLCOMPRESSION ]
   [ [ , ] THREADCOUNT = { n } ]
   [ [ , ] THREADPRIORITY = { 0 | 1 | 2 | 3 | 4 | 5 | 6 } ]
   [ [ , ] USESIMPLECOPY ]
   [ [ , ] VERIFY ]
]
```

Backing up multiple databases to single files or split into multiple files

```
BACKUP
{
   [
      { [ ALL | SYSTEM | USER] } DATABASES [EXCLUDE [ list of databases ]]
      |
      DATABASES [EXCLUDE] { [ list_of_databases ] }
   ]
}
TO { DISK } = { 'physical_backup_device_name' | '<AUTO>' } [ ,...n ]
[ WITH
   [ [ , ] { CHECKSUM | NO_CHECKSUM } ]
   [ [ , ] COMPRESSION = { 0 | 1 | 2 | 3 | 4 } ]
   [ [ , ] { CONTINUE_AFTER_ERROR | STOP_ON_ERROR } ]
   [ [ , ] COPY_ONLY ]
   [ [ , ] COPYTO = { 'target_folder_name' } [ ,...n ] ]
   [ [ , ] DESCRIPTION = { 'text' } ]
   [ [ , ] DIFFERENTIAL ]
   [ [ , ] DISKRETRYCOUNT = { n } ]
   [ [ , ] DISKRETRYINTERVAL = { seconds } ]
   [ [ , ] ERASEFILES | ERASEFILES_ATSTART = { days | hours{h} | except latest{b} } ]
   [ [ , ] ERASEFILES_REMOTE = { days | hours{h} | except latest{b} } ]
   [ [ , ] FILECOUNT = { n } ]
   [ [ , ] FILEOPTIONS = { 1 | 2 | 3 | 4 | 5 | 6 | 7 } ]
   [ [ , ] INIT ]
   [ [ , ] KEYSIZE = { 128 | 256 } ]
   [ [ , ] LOG_ONERROR ]
   [ [ , ] LOG_ONERRORONLY ]
   [ [ , ] LOGTO = { 'target_folder_name' | 'file_name' } ]
   [ [ , ] MAILTO = { 'recipients' } ]
   [ [ , ] MAILTO_NOLOG ]
   [ [ , ] MAILTO_ONERROR = { 'recipients' } ]
   [ [ , ] MAILTO_ONERRORONLY = { 'recipients' } ]
   [ [ , ] MAXDATABLOCK = { 65536 | 131072 | ... | 2097152 } ]
   [ [ , ] MAXTRANSFERSIZE = { 65536 | 131072 | ... | 1048576 } ]
   [ [ , ] NAME = { 'backup_set_name' } ]
   [ [ , ] NOCOMPRESSWRITE | NOWRITE ]
   [ [ , ] NOLOG ]
   [ [ , ] PASSWORD = { 'password' } ]
   [ [ , ] SINGLERESULTSET ]
   [ [ , ] SQLCOMPRESSION ]
   [ [ , ] THREADCOUNT = { n } ]
   [ [ , ] THREADPRIORITY = { 0 | 1 | 2 | 3 | 4 | 5 | 6 } ]
   [ [ , ] USESIMPLECOPY ]
   [ [ , ] VERIFY ]
]
```

Creating a partial filegroup backup

```
BACKUP DATABASE { database_name }
   READ_WRITE_FILEGROUPS [, FILEGROUP = { 'logical_filegroup_name' } [ ,...n ] ]
TO { DISK } = { 'physical_backup_device_name' | '<AUTO>' } [ ,...n ]
[ WITH
   [ [ , ] { CHECKSUM | NO_CHECKSUM } ]
   [ [ , ] COMPRESSION = { 0 | 1 | 2 | 3 | 4 } ]
   [ [ , ] { CONTINUE_AFTER_ERROR | STOP_ON_ERROR } ]
   [ [ , ] COPY_ONLY ]
   [ [ , ] COPYTO = { 'target_folder_name' } [,...n ]
   [ [ , ] DESCRIPTION = { 'text' } ]
   [ [ , ] DIFFERENTIAL ]
   [ [ , ] DISKRETRYCOUNT = { n } ]
   [ [ , ] DISKRETRYINTERVAL = { seconds } ]
   [ [ , ] ERASEFILES | ERASEFILES_ATSTART = { days | hours{h} | except latest{b} } ]
   [ [ , ] ERASEFILES_REMOTE = { days | hours{h} | except latest{b} } ]
   [ [ , ] FILECOUNT = { n } ]
   [ [ , ] FILEOPTIONS = { 1 | 2 | 3 | 4 | 5 | 6 | 7 } ]
   [ [ , ] INIT ]
   [ [ , ] KEYSIZE = { 128 | 256 } ]
```

```
   [ [ , ] LOG_ONERROR ]
   [ [ , ] LOG_ONERRORONLY ]
   [ [ , ] LOGTO = { 'target_folder_name' | 'file_name' } ]
   [ [ , ] MAILTO = { 'recipients' } ]
   [ [ , ] MAILTO_NOLOG ]
   [ [ , ] MAILTO_ONERROR = { 'recipients' } ]
   [ [ , ] MAILTO_ONERRORONLY = { 'recipients' } ]
   [ [ , ] MAXDATABLOCK = { 65536 | 131072 | ... | 2097152 } ]
   [ [ , ] MAXTRANSFERSIZE = { 65536 | 131072 | ... | 1048576 } ]
   [ [ , ] MIRRORFILE = { 'physical_backup_device_name' } ] [ ,...n ]
   [ [ , ] NAME = { 'backup_set_name' } ]
   [ [ , ] NOCOMPRESSWRITE | NOWRITE ]
   [ [ , ] NOLOG ]
   [ [ , ] PASSWORD = { 'password' } ]
   [ [ , ] SINGLERESULTSET ]
   [ [ , ] SQLCOMPRESSION ]
   [ [ , ] THREADCOUNT = { n } ]
   [ [ , ] THREADPRIORITY = { 0 | 1 | 2 | 3 | 4 | 5 | 6 } ]
   [ [ , ] USESIMPLECOPY ]
   [ [ , ] VERIFY ]
]
```

Backing up a transaction log to a single file or split into multiple files

```
BACKUP LOG { database_name }
TO { DISK } = { 'physical_backup_device_name' | '<AUTO>' } [ ,...n ]
[ WITH
   [ [ , ] { CHECKSUM | NO_CHECKSUM } ]
   [ [ , ] COMPRESSION = { 0 | 1 | 2 | 3 | 4 } ]
   [ [ , ] { CONTINUE_AFTER_ERROR | STOP_ON_ERROR } ]
   [ [ , ] COPY_ONLY ]
   [ [ , ] COPYTO = { 'target_folder_name' } [,...n ]
   [ [ , ] DESCRIPTION = { 'text' } ]
   [ [ , ] DISCONNECT_EXISTING ]
   [ [ , ] DISKRETRYCOUNT = { n } ]
   [ [ , ] DISKRETRYINTERVAL = { seconds } ]
   [ [ , ] ERASEFILES | ERASEFILES_ATSTART = { days | hours{h} | except latest{b} } ]
   [ [ , ] ERASEFILES_REMOTE = { days | hours{h} | except latest{b} } ]
   [ [ , ] FILECOUNT = { n } ]
   [ [ , ] FILEOPTIONS = { 1 | 2 | 3 | 4 | 5 | 6 | 7 } ]
   [ [ , ] INIT ]
   [ [ , ] KEYSIZE = { 128 | 256 } ]
   [ [ , ] LOG_ONERROR ]
   [ [ , ] LOG_ONERRORONLY ]
   [ [ , ] LOGTO = { 'target_folder_name' | 'file_name' } ]
   [ [ , ] MAILTO = { 'recipients' } ]
   [ [ , ] MAILTO_NOLOG ]
   [ [ , ] MAILTO_ONERROR = { 'recipients' } ]
   [ [ , ] MAILTO_ONERRORONLY = { 'recipients' } ]
   [ [ , ] MAXDATABLOCK = { 65536 | 131072 | ... | 2097152 } ]
   [ [ , ] MAXTRANSFERSIZE = { 65536 | 131072 | ... | 1048576 } ]
   [ [ , ] MIRRORFILE = { 'physical_backup_device_name' } ] [ ,...n ]
   [ [ , ] NAME = { 'backup_set_name' } ]
   [ [ , ] NO_TRUNCATE ]
   [ [ , ] NOCOMPRESSWRITE | NOWRITE ]
   [ [ , ] NOLOG ]
   [ [ , ] { NORECOVERY | STANDBY = 'standby_file_name' } ]
   [ [ , ] PASSWORD = { 'password' } ]
   [ [ , ] SINGLERESULTSET ]
   [ [ , ] SQLCOMPRESSION ]
   [ [ , ] THREADCOUNT = { n } ]
   [ [ , ] THREADPRIORITY = { 0 | 1 | 2 | 3 | 4 | 5 | 6 } ]
   [ [ , ] USESIMPLECOPY ]
   [ [ , ] VERIFY ]
]
```

Backing up multiple transaction logs to single files or split into multiple files

```
BACKUP LOGS [EXCLUDE] { [ list_of_databases ] }
```

```
TO { DISK } = { 'physical_backup_device_name' | '<AUTO>' } [ ,...n ]
[ WITH
   [ [ , ] { CHECKSUM | NO_CHECKSUM } ]
   [ [ , ] COMPRESSION = { 0 | 1 | 2 | 3 | 4 } ]
   [ [ , ] { CONTINUE_AFTER_ERROR | STOP_ON_ERROR } ]
   [ [ , ] COPY_ONLY ]
   [ [ , ] COPYTO = { 'target_folder_name' } [,...n ]
   [ [ , ] DESCRIPTION = { 'text' } ]
   [ [ , ] DISCONNECT_EXISTING ]
   [ [ , ] DISKRETRYCOUNT = { n } ]
   [ [ , ] DISKRETRYINTERVAL = { seconds } ]
   [ [ , ] ERASEFILES | ERASEFILES_ATSTART = { days | hours{h} | except latest{b} } ]
   [ [ , ] ERASEFILES_REMOTE = { days | hours{h} | except latest{b} } ]
   [ [ , ] FILECOUNT = { n } ]
   [ [ , ] FILEOPTIONS = { 1 | 2 | 3 | 4 | 5 | 6 | 7 } ]
   [ [ , ] INIT ]
   [ [ , ] KEYSIZE = { 128 | 256 } ]
   [ [ , ] LOG_ONERROR ]
   [ [ , ] LOG_ONERRORONLY ]
   [ [ , ] LOGTO = { 'target_folder_name' | 'file_name' } ]
   [ [ , ] MAILTO = { 'recipients' } ]
   [ [ , ] MAILTO_NOLOG ]
   [ [ , ] MAILTO_ONERROR = { 'recipients' } ]
   [ [ , ] MAILTO_ONERRORONLY = { 'recipients' } ]
   [ [ , ] MAXDATABLOCK = { 65536 | 131072 | ... | 2097152 } ]
   [ [ , ] MAXTRANSFERSIZE = { 65536 | 131072 | ... | 1048576 } ]
   [ [ , ] NAME = { 'backup_set_name' } ]
   [ [ , ] NO_TRUNCATE ]
   [ [ , ] NOCOMPRESSWRITE | NOWRITE ]
   [ [ , ] NOLOG ]
   [ [ , ] { NORECOVERY | STANDBY = 'standby_file_name' } ]
   [ [ , ] PASSWORD = { 'password' } ]
   [ [ , ] SINGLERESULTSET ]
   [ [ , ] SQLCOMPRESSION ]
   [ [ , ] THREADCOUNT = { n } ]
   [ [ , ] THREADPRIORITY = { 0 | 1 | 2 | 3 | 4 | 5 | 6 } ]
   [ [ , ] USESIMPLECOPY ]
   [ [ , ] VERIFY ]
]
```

Arguments

DATABASE argument

Specifies a backup of a single database. The database name must be enclosed in square brackets if it includes reserved words or spaces; if the database name does not include reserved words or spaces, square brackets are optional. For example:

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\Full\pubs.sqb'"
```

FILE = 'logical_file_name' | FILEGROUP = 'logical_filegroup_name' [,...n]

Specifies the files or filegroups that are to be backed up. There is no limit on the number of files or filegroups that can be included in a BACKUP co mmand, provided they all belong to the same database. Identify files or filegroups using logical names. For example:

```
"BACKUP DATABASE [Database1] FILE = 'SalesF1', FILE= 'SalesF2' TO DISK =
'C:\Backups\SalesFiles\<AUTO>'"
```

READ_WRITE_FILEGROUPS [, FILEGROUP = 'logical_filegroup_name']

Specifies a partial backup of all read/write files in the database, together with any specified read-only files or filegroups. For example:

```
"BACKUP DATABASE [Database1] READ_WRITE_FILEGROUPS [, FILEGROUP = SalesArchive] TO
DISK = 'C:\Backups\SalesFiles\<AUTO>'"
```

For more information, refer to your SQL Server documentation.

DATABASES argument

Specifies multiple full or differential database backups.

'[list_of_databases]'

Is a comma-separated list of the names of the databases that are to be backed up. For example:

```
"BACKUP DATABASES [Database1, Database2, Database3] TO DISK = 'C:\Backups\<AUTO>'"
```

The list must be enclosed in square brackets [ ], and the list must **not** contain any other square brackets. You can use a single wildcard character. For example:

```
"BACKUP DATABASES [*] TO DISK = 'C:\Backups\<AUTO>'"
```

This backs up all databases that are currently online and operational, including read-only databases. Databases that are unrecovered or unavailable (for example, offline) are not backed up.

As an alternative to using a list of databases, you can specify `BACKUP ALL DATABASES`. This is equivalent to `BACKUP DATABASES [*]`

To back up all *user* databases on an instance (excluding *master*, *model*, and *msdb*), use `BACKUP USER DATABASES`.

To back up all *system* databases on an instance (only *master*, *model*, and *msdb*), use `BACKUP SYSTEM DATABASES`.

EXCLUDE

Specifies that all online databases **except** those listed (or specified by `ALL`, `USER`, or `SYSTEM`) are to be backed up. For example:

```
"BACKUP DATABASES EXCLUDE [Database1, Database2, Database3] TO DISK =
'C:\Backups\<AUTO>'"
```

This backs up all databases that are currently online and operational, except for *Database1*, *Database2*, and *Database3*. You cannot use wildcard characters in exclusion lists.

You can also combine the `EXCLUDE` keyword with the `ALL`, `SYSTEM`, or `USER` keywords. For example:

```
"BACKUP USER DATABASES EXCLUDE [Database1, Database2, Database3] TO DISK =
'C:\Backups\<AUTO>'"
```

This backs up all databases that are currently online and operational, but excludes the system databases (master, model, and msdb) as well as excluding *Database1*, *Database2*, and *Database3*.

TO { DISK } = { 'physical_backup_device_name' | '<AUTO>' } [,...n]

Creates the backups on the specified disk. You are recommended to include the `<AUTO>` parameter to ensure that each database is backed up to a unique file name. For example:

```
"BACKUP DATABASES [Database1, Database2, Database3] TO DISK = '<AUTO>'"
```

Alternatively, you could set up your own unique path using tags, such as the <DATETIME> tag. For more details, see the TO DISK argument belo w.

LOG argument

Specifies a log backup from a single database. The database must be using the FULL or BULK-LOGGED recovery model. The database name must be enclosed in square brackets if it includes reserved words or spaces; if the database name does not include reserved words or spaces, square brackets are optional. For example:

```
"BACKUP LOG [pubs] TO DISK = 'C:\Backups\Logs\pubs.sqb'"
```

For more information, refer to your SQL Server documentation.

LOGS argument

Specifies multiple transaction log backups.

'[list_of_databases]'

Is a comma-separated list of the names of the databases for which the transaction logs are to be backed up. The databases must use the FULL or BULK-LOGGED recovery model. For example:

```
"BACKUP LOGS [Database1, Database2, Database3] TO DISK = 'C:\Backups\Logs\<AUTO>'"
```

The list must be enclosed in square brackets [ ], and the list must not contain any other square brackets. Alternatively, a single wildcard character can be used. For example:

```
"BACKUP LOGS [*] TO DISK = 'C:\Backups\Logs\<AUTO>'"
```

This backs up transaction logs for all databases that are currently online, operational, and using the FULL or BULK-LOGGED recovery models.

EXCLUDE

Specifies that transaction logs of all online databases **except** those listed are to be backed up. For example:

```
"BACKUP LOGS EXCLUDE [Database1, Database2, Database3] TO DISK =
'C:\Backups\Logs\<AUTO>'"
```

This backs up the transaction logs of all databases that are currently online and operational, except for *Database1*, *Database2*, and *Database3*.

TO { DISK } = { 'physical_backup_device_name' | '<AUTO>' } [,...n]

Creates the backups on the specified disk. **Must** include the <AUTO> parameter to ensure that each transaction log is backed up to a unique file name. For details, see the `TO DISK` argument below. For example:

```
"BACKUP LOGS [*] TO DISK = 'C:\Backups\Logs\<AUTO>'"
```

TO DISK argument

You can specify multiple `DISK` values, up to a maximum of 32. For example:

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\File1.sqb', DISK =
'C:\Backups\File2.sqb'"
```

TO DISK = '<AUTO>'

If you specify <AUTO>, SQL Backup generates the backup file path and file name using the backup file locations specified in the File management options. If no backup file locations have been set up, SQL Backup uses the SQL Server instance's default backup folder, and the default format for file names. For details, see File management options.

TO DISK = 'path\<AUTO>'

If you specify a path and <AUTO>, SQL Backup uses the specified path, and generates the file name using the File management options. If no backup file locations have been set up, SQL Backup uses the default format for file names.

You can use tags in the path. For example, if you are backing up multiple databases and you want the backup files for each database to be created in a separate folder, you can specify the <DATABASE> tag. The following example creates the backups in a different folder for each database name:

```
"BACKUP DATABASES [pubs, northwind, AdventureWorks] TO DISK =
'C:\Backups\<DATABASE>\<AUTO>'"
```

You can specify more than one tag in the path. In the following example, backups are grouped by database name, date, and type of backup:

```
"BACKUP DATABASES [pubs, northwind, AdventureWorks] TO DISK =
'C:\Backups\<DATABASE>\<DATETIME yyyymmdd>\<TYPE>\<AUTO>'"
```

For details of the tags you can use, see File location tags.

TO DISK = 'file_name'

If you specify a file name with the `TO DISK` argument, SQL Backup uses the File management options to generate the backup file path, and uses the specified name for the file. If no backup file locations have been set up, SQL Backup uses the SQL Server instance's default backup folder. For example:

```
"BACKUP DATABASE [pubs] TO DISK = 'pubs_20120229.sqb'"
```

TO DISK = '<AUTO>.sqb'

If you specify <AUTO> with a file extension, SQL Backup uses the File management options to generate the backup file path and file name, but uses the specified extension. If no backup file locations have been set up, SQL Backup uses the SQL Server instance's default backup folder, and the default format for file names.

WITH options

CHECKSUM|NO_CHECKSUM

Specifies whether a backup checksum should be created. If `CHECKSUM` is included, any page checksums are validated and a backup checksum is generated and stored on the backup media for use on restore. If an invalid page checksum is encountered, the backup will fail. If `NO_CHECKSUM` i s specified, no checksums are validated or generated. This the default behavior in a `BACKUP` command. For more information, refer to your SQL Server documentation.

COMPRESSION

Specifies the compression level. The default value is 1. If you do not want to compress the backups, specify 0. For more information, see Compre ssion levels.

To use native SQL Server compression (available for SQL Server 2008 Enterprise Edition, SQL Server 2008 R2 Standard Edition and SQL Server 2012), use the `SQLCOMPRESSION` keyword. You cannot include both `COMPRESSION` and `SQLCOMPRESSION` in a command.

CONTINUE_AFTER_ERROR | STOP_ON_ERROR

If `CHECKSUM` is included in the `BACKUP` command, `CONTINUE_ON_ERROR` specifies that the backup process should continue if an invalid page checksum or torn page is encountered. `STOP_ON_ERROR` specifies that the backup process should stop if an invalid page checksum or torn page is encountered. If neither option is included, `STOP_ON_ERROR` is the default behavior.

For more information, refer to your SQL Server documentation.

COPY_ONLY

Specifies a copy-only backup. A copy-only full backup is independent of the sequence of backups and does not affect the differential base LSN. A copy-only full backup cannot be used as a base for restoring a differential backup.

A copy-only log backup is independent of the sequence of regular log backups, and does not affect the log archive point. Taking a copy-only log backup will not affect log shipping operations.

This option cannot be used with the `DIFFERENTIAL` option. For more information, refer to your SQL Server documentation.

COPYTO

Specifies that a copy of the backup files is to be created in the specified folder when the backup process completes. For example:

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<AUTO>' WITH COPYTO =
'\\BACKUPSERVER001\Folder1'"
```

You can use tags in the path. To create a copy of the backup in more than one folder, use multiple `COPYTO` arguments. For example:

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<AUTO>' WITH COPYTO =
'\\BACKUPSERVER001\<DATABASE>', COPYTO = '\\BACKUPSERVER002\<DATABASE>'"
```

You must ensure that you have sufficient rights to the specified folders.

If you want to prevent retries during copy operations, specify `USESIMPLECOPY` as well.

DESCRIPTION

Specifies the text describing the backup set. The description is limited to 255 characters. For more information, refer to your SQL Server documentation.

DIFFERENTIAL

Limits the backup to the portions of the database or file that have changed since the last full backup. For more information, refer to your SQL Server documentation.

DISCONNECT_EXISTING

Kills any existing connections to the database before starting a transaction log backup. `DISCONNECT_EXISTING` is only valid when you are backing up the tail of the transaction log (with `NORECOVERY` or `STANDBY`).

DISKRETRYCOUNT

In combination with `DISKRETRYINTERVAL`, this argument controls network resilience behavior.

`DISKRETRYCOUNT` specifies the maximum number of times to retry a failed data-transfer operation (writing or copying a backup file). If you omit this keyword, the default value of 10 is used; specify a value of 0 to prevent any retries following a failed data-transfer operation. If you specify a

value for DISKRETRYCOUNT, you should also specify a value for DISKRETRYINTERVAL.

If you have also specified the COPYTO keyword, you can prevent retries for the copying operation only, by specifying USESIMPLECOPY.

DISKRETRYINTERVAL

In combination with DISKRETRYCOUNT, this argument controls network resilience behavior.

DISKRETRYINTERVAL specifies the time interval between retries, in seconds, following a failed data-transfer operation (writing or copying a backup file). If you omit this keyword, the default value of 30 seconds is used. If you specify a value for DISKRETRYINTERVAL, you should also specify a value for DISKRETRYCOUNT.

If you have also specified the COPYTO keyword, you can prevent retries for the copying operation only, by specifying USESIMPLECOPY.

ERASEFILES

Manages deletion of existing SQL Backup backups from the primary backup folder (specified using DISK and MIRRORFILE). If multiple DISK locations are specified, the setting is applied to each folder.

The backup files are deleted only if the backup process completes successfully. **Note:** To delete backup files before the start of the backup process, use ERASEFILES_ATSTART.

You can choose to delete SQL Backup files based on:

- Age: files older than the specified number of days or hours are deleted. Specify a number for days, or type *h* after the number for hours. For example, ERASEFILES = 24 deletes files that are more than 24 days old; ERASEFILES = 24h deletes files that are more than 24 hours old. Note that a day is calculated as a period of 24 hours, and takes no account of calendar date.
- Number of backups to keep: only the latest 'x' backups will be kept. To specify the number of backups to be kept, type *b* after the number.
  For example, ERASEFILES = 5b ensures the latest 5 backups are kept; older backups are deleted.

> If you include both ERASEFILES and COPYTO in a BACKUP DATABASE command, the full or differential backup is written and old backups are deleted from the DISK location before the new backup is copied to the COPYTO location.
>
> If you include both ERASEFILES and COPYTO in a BACKUP LOG command, the transaction log backup is written and added to the log copy queue for copying to the COPYTO location before the old backups are deleted from the DISK location. SQL Backup does not wait for the transaction log backup to be copied successfully before deleting old backups from the DISK location.

Files are deleted only if the following details match the details of the database being backed up:

- The name of the SQL Server, instance (if applicable), and database recorded in the file header.
- The backup type (full, differential, transaction log).
- The backup password. If the PASSWORD option is not specified (because the backup is not encrypted), any existing encrypted backups in the DISK location will not be identified by ERASEFILES because the file header cannot be read. This may result in backups older than the specified age or in excess of the specified number being retained.

> If SQL Backup cannot list the contents of the folder that contains the files to be deleted, it cannot delete the files. Ensure the SQL Backup Agent service startup account (or, if you are using the command line, the user account from which you are running *SQLBackupC.exe*) has permissions to list the folder contents.

Examples

The following example creates a full backup in *C:\Backups\pubs*, then deletes any full backups of the *pubs* database older than 5 days from that folder.

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<DATABASE>\<AUTO>.sqb' WITH ERASEFILES =
5"
```

The following example creates a full backup in *C:\Backups\pubs*, then deletes all full backups of the *pubs* database other than the latest 7 from that folder.

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<DATABASE>\<AUTO>.sqb' WITH ERASEFILES =
7b"
```

The following example creates a full backup in a remote folder, *\\BACKUPSERVER001\pubs*, then deletes all full backups of the *pubs* database over 12 hours old from that folder.

```
 "BACKUP DATABASE [pubs] TO DISK = '\\BACKUPSERVER001\<DATABASE>\<AUTO>.sqb' WITH
ERASEFILES = 12h"
```

You can use `ERASEFILES_REMOTE` to manage deletion of files from a remote location specified using the `COPYTO` option.

> If the `DISK` location is a *remote* location and `ERASEFILES_REMOTE` is also included in the command, the `ERASEFILES_REMOTE` settin g will override the `ERASEFILES` setting.
>
> If a *local* `COPYTO` location is specified, the `ERASEFILES` setting will only apply to backups in the `COPYTO` location if `ERASEFILES_REMO TE` is also included in the command.

To prevent deletion of files that have their archive attribute set, use `FILEOPTIONS`.

To overwrite existing backups of the same name, use the `INIT` option.

ERASEFILES_ATSTART

Manages deletion of existing SQL Backup backups from the primary backup folder (specified using `DISK` and `MIRRORFILE`) *before* the backup process starts. If multiple `DISK` locations are specified, the setting is applied to each folder before the backup process starts. To delete backup files only if the backup process completes successfully, use `ERASEFILES`.

You can choose to delete SQL Backup files based on:

- Age: files older than the specified number of days or hours are deleted. Specify a number for days, or type *h* after the number for hours. For example, `ERASEFILES_ATSTART = 24` deletes files that are more than 24 days old; `ERASEFILES_ATSTART = 24h` deletes files that are more than 24 hours old. Note that a day is calculated as a period of 24 hours, and takes no account of calendar date.
- Number of backups to keep: only the latest 'x' backups will be kept. To specify the number of backups to be kept, type *b* after the number. For example, `ERASEFILES_ATSTART = 5b` ensures the latest 5 backups are kept; older backups are deleted.

Files are deleted only if the following details match the details of the database being backed up:

- The name of the SQL Server, instance (if applicable), and database recorded in the file header.
- The backup type (full, differential, transaction log).
- The backup password. If the `PASSWORD` option is not specified (because the backup is not encrypted), any existing encrypted backups in the `DISK` location will not be identified by `ERASEFILES_ATSTART` because the file header cannot be read. This may result in backups older than the specified age or in excess of the specified number being retained.

> If SQL Backup cannot list the contents of the folder that contains the files to be deleted, it cannot delete the files. Ensure the SQL Backup Agent service startup account (or, if you are using the command line, the user account from which you are running *SQLBackup C.exe*) has permissions to list the folder contents.

To prevent deletion of files that have their archive attribute set, use `FILEOPTIONS`.

To overwrite existing backups of the same name, use the `INIT` option.

You can use `ERASEFILES_REMOTE` to manage deletion of files in a remote location.

> There are known issues when `ERASEFILES_ATSTART` and `ERASEFILES_REMOTE` are used in the same `BACKUP` command. You are recommended to test the command to ensure it results in the desired behavior before using it in a production environment.

ERASEFILES_REMOTE

Manages deletion of existing SQL Backup backups from *remote* locations, whether the primary backup folder (specified using `DISK` and `MIRRORF`

ILE) or the secondary backup folder (specified using COPYTO). The backup files are deleted only if the backup process completes successfully.

You can choose to delete SQL Backup files based on:

- Age: files older than the specified number of days or hours are deleted. Specify a number for days, or type *h* after the number for hours. For example, ERASEFILES_REMOTE = 24 deletes files that are more than 24 days old; ERASEFILES_REMOTE = 24h deletes files that are more than 24 hours old. Note that a day is calculated as a period of 24 hours, and takes no account of calendar date.
- Number of backups to keep: only the latest 'x' backups will be kept. To specify the number of backups to be kept, type *b* after the number. For example, ERASEFILES_REMOTE = 5b ensures the latest 5 backups are kept; older backups are deleted.

Files are deleted only if the following details match the details of the database being backed up:

- The name of the SQL Server, instance (if applicable), and database recorded in the file header.
- The backup type (full, differential, transaction log).
- The backup password. If the PASSWORD option is not specified (because the backup is not encrypted), any existing encrypted backups in the DISK location will not be identified by ERASEFILES_REMOTE because the file header cannot be read. This may result in backups older than the specified age or in excess of the specified number being retained.

> If SQL Backup cannot list the contents of the folder that contains the files to be deleted, it cannot delete the files. Ensure that SQL Backup Agent service startup account (or, if you are using the command line, the user account from which you are running *SQLBackupC.exe*) has permissions to list the folder contents.

Examples

The following example creates a full backup in *C:\Backups\pubs*, copies the backup to a remote folder, *\\BACKUPSERVER001\pubs*, then deletes all full backups of the *pubs* database older than 5 days from the remote folder.

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<DATABASE>\<AUTO>.sqb' WITH COPYTO =
'\\BACKUPSERVER001\<DATABASE>\', ERASEFILES_REMOTE = 5"
```

The following example creates a full backup in *C:\Backups\pubs*, deletes all but the last 5 full backups of the *pubs* database from that folder, then copies the backup to a remote folder, *\\BACKUPSERVER001\pubs*, and deletes all full backups of the *pubs* database older than 10 days from the remote folder.

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<DATABASE>\<AUTO>.sqb' WITH COPYTO =
'\\BACKUPSERVER001\<DATABASE>\', ERASEFILES = 5b, ERASEFILES_REMOTE = 10"
```

The following example creates a full backup in *\\BACKUPSERVER001\pubs*, deletes all full backups of the *pubs* database older than 30 days from that folder, then copies the backup to a remote folder, *\\BACKUPSERVER002\pubs*, and deletes any full backups of the *pubs* database older than 30 days from the that folder.

```
"BACKUP DATABASE [pubs] TO DISK = '\\BACKUPSERVER001\<DATABASE>\<AUTO>.sqb' WITH
COPYTO = '\\BACKUPSERVER002\<DATABASE>\', ERASEFILES_REMOTE = 30"
```

To overwrite existing files of the same name, use FILEOPTIONS.

FILECOUNT

Specifies the number of backup files to be generated if you are splitting the backup across a number of files, where *n* is an integer between 2 and 32 inclusive. The name of the file specified in the TO DISK argument is used as the base name for the generated files. For example:

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\Pubs\FULL_20120229.sqb' WITH FILECOUNT =
4"
```

will generate four backup files:

C:\Backups\Pubs\FULL_20120229_01.sqb

C:\Backups\Pubs\FULL_20120229_02.sqb

C:\Backups\Pubs\FULL_20120229_03.sqb

C:\Backups\Pubs\FULL_20120229_04.sqb

Can be used together with the <AUTO> keyword (see the `TO DISK` argument). For example:

```
"BACKUP DATABASE [pubs] TO DISK = '<AUTO>' WITH FILECOUNT = 4"
```

In this example, SQL Backup generates the additional files using the <AUTO> naming convention, appending *_02*, *_03*, *_04* and so on to the root name.

If you are using the `FILECOUNT` keyword to create multiple files:

- You cannot use `TO DISK` multiple times.
- You cannot use the `THREADCOUNT` keyword to use multiple threads.

FILEOPTIONS

`FILEOPTIONS = 1` has been deprecated. Use `ERASEFILES_REMOTE` or `ERASEFILES_SECONDARY`.

Specifies whether old backup files are to be deleted or overwritten in the primary backup folder and any `COPYTO` folders. Specify the sum of the values that correspond to the options you require:

1	Delete old backup files in the secondary backup folders (specified using `COPYTO`) if they are older than the number of days or hours specified in `ERASEFILES` or `ERASEFILES_ATSTART`. Backups are deleted from the secondary backup folder after the backup is copied, regardless of whether `ERASEFILES` or `ERASEFILES_ATSTART` is used.
2	Delete old backup files in the primary backup folder (specified using `DISK`) if they are older than the number of days or hours specified in `ERASEFILES`, `ERASEFILES_ATSTART`, or `ERASEFILES_REMOTE` unless they have the `ARCHIVE` flag set.
4	Overwrite existing files of the same name in the `COPYTO` folder. If combined with option `1` or `ERASEFILES_REMOTE`, existing files are deleted before files of the same name are overwritten. This can result in fewer backups remaining in the folder than expected.

Valid values are **1** to **7**.

If you specify option *1* or *2* you must also set the age of the files to delete using `ERASEFILES`, `ERASEFILES_ATSTART`, or `ERASEFILES_REMOTE`. For example, to delete old backup files in the `COPYTO` folder that are older than 5 days:

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<AUTO>' WITH COPYTO =
'\\BACKUPSERVER001\<DATABASE>' , ERASEFILES = 5, FILEOPTIONS = 1"
```

To overwrite any existing files in the `COPYTO` folder and also delete old backup files in the `COPYTO` folder that are older than 5 days (values 1 + 4):

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<AUTO>' WITH COPYTO =
'\\BACKUPSERVER001\<DATABASE>' , ERASEFILES = 5, FILEOPTIONS = 5"
```

INIT

Specifies that files with the same name in the primary backup folder should be overwritten.

Because SQL Backup supports a maximum of one backup set in a media set, specifying `INIT` also overwrites any media-set data (media header) associated with the existing backup files. This is equivalent to using the `FORMAT` argument in SQL Server.

KEYSIZE

Specifies the size of the encryption key to use; you must also use the `PASSWORD` keyword to specify a password for the encrypted backups. You can specify a 128-bit key or a 256-bit key. For example:

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<AUTO>' WITH PASSWORD = 'MyPassword',
KEYSIZE = 128"
```

If you include `PASSWORD` and do not specify the key size, SQL Backup uses a 256-bit key.

LOG_ONERROR

Specifies that a log file should only be created if SQL Backup encounters an error during the backup process, or the backup completes successfully but with warnings. Use this option if you want to restrict the number of log files created by your backup processes, but maintain log information whenever warnings or errors occur. This argument controls the creation of log files on disk only; emailed log files are not affected. (See the `MAILTO` options below for details on emailing log files.)

LOG_ONERRORONLY

Specifies that a log file should only be created if SQL Backup encounters an error during the backup process. Use this option if you want to restrict the number of log files created by your backup processes, but maintain log information whenever errors occur. This argument controls the creation of log files on disk only; emailed log files are not affected. (See the `MAILTO` options below for details on emailing log files.)

LOGTO

Specifies that a copy of the log file is to be saved.

By default, the primary log file is created in the folder *%ProgramData%\Red Gate\SQL Backup\Log* (Windows Vista, Windows 2008 and later) or *%ALLUSERSPROFILE%\Application Data\Red Gate\SQL Backup\Log* (Windows XP and Windows 2003); you can change this location in your file management options.

To create a copy with the same name as the primary log file, specify the folder. For example:

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<AUTO>' WITH LOGTO = 'C:\Logs'"
```

To create a copy with a different name from the primary log file, specify the folder and file name. For example:

```
"BACKUP DATABASE pubs TO DISK = 'C:\Backups\<AUTO>' WITH LOGTO =
'C:\Logs\SQBSecondaryLog.txt'"
```

To copy the log file to more than one location, use multiple `LOGTO` commands.

MAILTO

Specifies that the outcome of the backup operation is emailed to one or more users; the email includes the contents of the log file. SQL Backup uses the settings specified in your email settings to send the email. To specify multiple recipients, separate the email addresses with a semi-colon (**;**). For example:

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<AUTO>' WITH MAILTO =
'dba01@myco.com;dba02@myco.com'"
```

If you have not defined email settings, the email will not be sent and a warning will be reported.

MAILTO_NOLOG

> MAILTO_NOLOG is only available in SQL Backup 6.5.1 and later.

Specifies that SQL Backup should not include the contents of the log file in the email. An email will still be sent to notify the specified recipients of success and/or failure, depending on which `MAILTO` parameter has been specified.

MAILTO_ONERROR

Specifies that the outcome of the backup operation is emailed to one or more users if SQL Backup encounters an error during the backup process or if the backup completes successfully but with warnings. The email includes the contents of the log file. SQL Backup uses the settings specified in your email settings to send the email. The email includes the contents of the log file. To specify multiple recipients, separate the email addresses with a semi-colon (**;**). For example:

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<AUTO>' WITH MAILTO_ONERROR =
'dba01@myco.com;dba02@myco.com'"
```

If you have not defined email settings, the email will not be sent and a warning will be reported.

MAILTO_ONERRORONLY

Specifies that the outcome of the backup operation is emailed to one or more users if SQL Backup encounters an error during the backup process. SQL Backup uses the settings specified in your email settings to send the email. To specify multiple recipients, separate the email addresses with a semi-colon (**;**). For example:

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<AUTO>' WITH MAILTO_ONERRORONLY =
'dba01@myco.com;dba02@myco.com'"
```

If you have not defined email settings, the email will not be sent and a warning will be reported.

MAXDATABLOCK

Specifies the maximum size of data blocks to be used when SQL Backup stores backup data. Valid values are integers in multiples of 65536, up to a maximum value of 2097152. For example:

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<AUTO>' WITH MAXDATABLOCK = 655360"
```

If not specified, SQL Backup uses the value defined in the following DWORD registry key:

HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\SQL Backup\BackupSettingsGlobal\<instance name>\MAXDATABLOCK

MAXTRANSFERSIZE

Specifies the maximum size of each block of memory (in bytes) to be used when SQL Backup stores backup data. You may want to specify this argument if a SQL Server reports that it has insufficient memory to service requests from SQL Backup.

Valid values are integers in multiples of 65536, up to a maximum value of 1048576.

For example:

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<AUTO>' WITH MAXTRANSFERSIZE = 262144"
```

If not specified, defaults to 1048576. However, if you have created the following DWORD registry key, SQL Backup uses the defined value as the default value:

HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\SQL Backup\BackupSettingsGlobal\<instance name>\MAXTRANSFERSIZE

MIRRORFILE

Indicates that you want to create a duplicate backup file (in addition to the backup file named in the `TO DISK` argument). Specify the value as described for the `TO DISK` argument. For example:

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<AUTO>' WITH MIRRORFILE =
'Z:\AlternativeLocation\MirrorBackup.sqb'"
```

A single backup is created, which is then written in parallel to the disk location and mirror locations. All the files will therefore finish being written at the same time.

`MIRRORFILE` is useful if you want to write backups to locations on similar devices. If you want to write to devices with different characteristics such as processor speed, it is better to use `COPYTO` to prevent unnecessary load on your SQL Server.

You cannot use `MIRRORFILE` if you are backing up multiple databases, if you specify multiple `DISK` options, or if you use the `FILECOUNT` option.

To create more than one duplicate backup file, use multiple `MIRRORFILE` arguments (up to a maximum of 32 files).

Before the backup process starts, the locations specified in the `MIRRORFILE` arguments are verified; if a location does not exist, or the file cannot be created at that location, then the backup will not start.

During the backup process, if any of the files cannot be written a warning is raised. However, the backup process continues as long as at least one specified backup file can be written. If none of the files can be written, an error is raised and the backup process is stopped.

NAME

Specifies the name of the backup set. The name is limited to 128 characters. If this option is not included, the name is left blank. For more information, refer to your SQL Server documentation.

NOCOMPRESSWRITE

Determines the maximum backup process throughput of your SQL Server. When you specify this argument, SQL Backup simulates a backup process without compression; no backup files are created.

You can compare the results obtained using `NOCOMPRESSWRITE` and `NOWRITE` to deduce the effects of compression on the backup throughput. For details, see Optimizing backup speed.

NOLOG

Prevents a log file from being created for the backup process, even if errors or warnings are generated. You may want to use this option if you are concerned about generating a large number of log files, and are certain that you will not need to review the details of errors or warnings (for example, because it's possible to run the process again without needing to know why it failed). This argument controls the creation of log files on disk only; emailed log files are not affected. (See the `MAILTO` options above for details on emailing log files.)

NORECOVERY

Backs up the tail of the transaction log and leaves the database in an unrecovered state. Incomplete transactions are not rolled back. The database is not usable, but differential and transaction log backups can be restored to it. For more information, refer to your SQL Server documentation.

NO_TRUNCATE

Specifies that the transaction log should not be truncated after a transaction log backup. For more information, refer to your SQL Server documentation.

NOWRITE

Determines the maximum backup process throughput of your SQL Server using compression. When you specify this argument, SQL Backup simulates a backup process using the specified compression level; no backup files are created.

You can compare the results obtained using `NOCOMPRESSWRITE` and `NOWRITE` to deduce the effects of compression on the backup throughput. For details, see Optimizing backup speed.

PASSWORD

Specifies the password to be used with encrypted backup files. You must supply the same password when you restore the backup. You cannot include square brackets in a password.

You can specify the size of the encryption key using the `KEYSIZE` keyword.

> In standard Transact-SQL syntax, the `PASSWORD` argument attaches a password to the backup file, but does not encrypt the file contents.

If you subsequently convert the SQL Backup file to a Microsoft Tape Format (MTF) file, you must supply the password; the MTF file will not be protected by the password.

SINGLERESULTSET

Specifies that the results returned by the `BACKUP` command should be limited to just one result set. This may be useful if you want to manipulate results using a Transact-SQL script. Such scripts can only manipulate results when a single result set is returned. The `BACKUP` command will return two result sets by default in most cases, unless you specify the `SINGLERESULTSET` keyword.

SQLCOMPRESSION

> SQLCOMPRESSION is only available in SQL Backup 6.3 and later.

Specifies that the backup should be compressed using SQL Server's native compression. This feature is available for SQL Server 2008 Enterprise Edition, SQL Server 2008 R2 Standard Edition and SQL Server 2012.

To use SQL Backup's compression algorithms, use the `COMPRESSION` keyword. Do not specify both `SQLCOMPRESSION` and `COMPRESSION`.

STANDBY

Backs up the tail of the transaction log and leaves the database in a read-only and `STANDBY` state. The `STANDBY` clause writes standby data (performing rollback, but with the option of further restores).

'undo_file_name'

Is a standby file, whose location is stored in the log of the database. This file holds the rolled back changes, which must be reversed if `RESTORE LOG` operations are to be subsequently applied.

When used with the `BACKUP LOGS` command (backing up multiple transaction logs) 'undo_file_name' must include the <DATABASE> tag.

When used with the `BACKUP LOG` command (backing up a single transaction log) 'undo_file_name' can include tags, but these are not required.

THREADCOUNT

Specifies the number of threads to be used to create the backup, where *n* is an integer between 2 and 32 inclusive. For example:

```
"BACKUP DATABASE [pubs] TO DISK = 'C:\Backups\<AUTO>' WITH THREADCOUNT = 4"
```

This example creates a single backup file using four threads.

> If you are using the `THREADCOUNT` keyword to use multiple threads:
>
> - you cannot use `TO DISK` multiple times
> - you cannot use the `FILECOUNT` keyword to create multiple files

THREADPRIORITY

Sets the SQL Backup thread priority when the backup or restore process is run. Valid values are **0** to **6**, and correspond to the following priorities:

0	Idle
1	Very low
2	Low
3	Normal
4	High
5	Very high
6	Time critical

If this value is not specified, normal priority (3) is used.

USESIMPLECOPY

Prevents retries occurring during copy operations (specified using the `COPYTO` keyword). With `USESIMPLECOPY` specified, copying behavior is the same as in SQL Backup version 5.

VERIFY

Checks that the backup set is complete and can be read, but does not verify the structure of the data inside the backup. If `WITH CHECKSUM` is included in the `BACKUP` command, both the backup checksum and any page checksums are also validated.

The process will continue if any errors (such as invalid checksums) are encountered. The results of the check (including any errors encountered) are displayed as text output when the backup completes. You can also check the results later by viewing the Activity History in the graphical user interface.

This is equivalent to running `RESTORE VERIFYONLY` on a backup file.

Deprecated WITH options

The following `WITH` options have been deprecated:

- `THREADS` (replaced by `FILECOUNT`)
- `ERASEFILEOPTIONS` (deprecated in version 4)
- `FILEOPTIONS` (value 1 only)

Examples

Back up a database to a single file

This example creates a full backup of the *pubs* database in a single file.

```
SQLBackupC.exe -I {instance name} -SQL "BACKUP DATABASE [pubs] TO DISK =
'C:\Backups\pubs_01.sqb' "
```

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASE [pubs] TO DISK =
''C:\Backups\pubs_01.sqb'' " '
```

Create a mirrored backup

This example simultaneously creates a full backup of the *pubs* database and two duplicate backup files.

```
SQLBackupC.exe -I {instance name} -SQL "BACKUP DATABASE [pubs] TO DISK =
'C:\Backups\pubs_01.sqb' WITH MIRRORFILE = 'E:\Backups\pubs_01_alt1.sqb' MIRRORFILE =
'F:\Backups\pubs_01_alt2.sqb' "
```

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASE [pubs] TO DISK =
''C:\Backups\pubs_01.sqb'' WITH MIRRORFILE = ''E:\Backups\pubs_01_alt1.sqb''
MIRRORFILE = ''F:\Backups\pubs_01_alt2.sqb'' " '
```

Split the backup file

This example creates a full backup of the *pubs* database and splits the backup across two files.

```
SQLBackupC.exe -I {instance name} -SQL "BACKUP DATABASE [pubs] TO DISK =
'C:\Backups\pubs_01.sqb', DISK = 'C:\Backups\pubs_02.sqb' "
```

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASE [pubs] TO DISK =
''C:\Backups\pubs_01.sqb'', DISK = ''C:\Backups\pubs_02.sqb'' " '
```

Create copies of the backup file in another disk location

This example creates a full backup of the *pubs* database and then copies the backup file to two remote folders on completion of the backup process.

```
SQLBackupC.exe -I {instance name} -SQL "BACKUP DATABASE [pubs] TO DISK =
'C:\Backups\pubs_01.sqb' WITH COPYTO = '\\BACKUPSERVER001\share', COPYTO =
'\\BACKUPSERVER002\share' "
```

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASE [pubs] TO DISK =
''C:\Backups\pubs_01.sqb'' WITH COPYTO = ''\\BACKUPSERVER001\share'', COPYTO =
''\\BACKUPSERVER002\share'' " '
```

Specify the compression level for a backup

This example creates a full backup of the *pubs* database using compression level 4.

```
SQLBackupC.exe -I {instance name} -SQL "BACKUP DATABASE [pubs] TO DISK =
'C:\Backups\pubs_01.sqb' WITH COMPRESSION = 4"
```

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASE [pubs] TO DISK =
''C:\Backups\pubs_01.sqb'' WITH COMPRESSION = 4" '
```

Encrypt the backup file

This example creates a full backup of the *pubs* database and encrypts the backup file with password *MyPassword* and a 256-bit key.

```
SQLBackupC.exe -I {instance name} -SQL "BACKUP DATABASE [pubs] TO DISK =
'C:\Backups\pubs_01.sqb' WITH PASSWORD = 'MyPassword', KEYSIZE = 256"
```

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASE [pubs] TO DISK =
''C:\Backups\pubs_01.sqb'' WITH PASSWORD = ''MyPassword'', KEYSIZE = 256" '
```

Check the backup file

This example creates a full backup of the *pubs* database, testing any page checksums in the process. A backup checksum is generated and tested, and the backup is checked to ensure it is complete and readable.

```
SQLBackupC.exe -I {instance name} -SQL "BACKUP DATABASE [pubs] TO DISK =
'C:\Backups\pubs_01.sqb' WITH CHECKSUM, VERIFY"
```

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASE [pubs] TO DISK =
''C:\Backups\pubs_01.sqb'' WITH CHECKSUM, VERIFY" '
```

Send email notification of the backup process

This example creates a full backup of the *pubs* database and sends an email of the completion log to two users upon completion of the backup process.

```
SQLBackupC.exe -I {instance name} -SQL "BACKUP DATABASE [pubs] TO DISK =
'C:\Backups\pubs_01.sqb' WITH MAILTO = 'dba01@myco.com;dba02@myco.com' "
```

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASE [pubs] TO DISK =
''C:\Backups\pubs_01.sqb'' WITH MAILTO = ''dba01@myco.com;dba02@myco.com'' " '
```

Back up multiple databases with exclusions

This example creates a full backup of all the databases except *master* and *model*.

```
-SQLBackupC.exe -I {instance name} -SQL "BACKUP DATABASES EXCLUDE [master, model] TO
DISK = 'C:\Backups\<AUTO>' "
```

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASES EXCLUDE [master, model] TO DISK =
''C:\Backups\<AUTO>'' " '
```

Back up multiple databases to split files

This example creates full backups of the *northwind* and *pubs* databases, splitting each into two files.

```
SQLBackupC.exe -I {instance name} -SQL "BACKUP DATABASES [northwind, pubs] TO DISK =
'C:\Backups\<AUTO>' WITH FILECOUNT = 2"
```

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASES [northwind, pubs] TO DISK =
''C:\Backups\<AUTO>'' WITH FILECOUNT = 2" '
```

Back up multiple databases to split files on different disks

This example creates full backups of the *northwind* and *pubs* databases, splitting both backups to separate files on different disks.

```
SQLBackupC.exe -I {instance name} -SQL "BACKUP DATABASES [northwind, pubs] TO DISK =
'C:\Backups\<AUTO>_01', TO DISK = 'D:\Backups\<AUTO>_02' "
```

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASES [northwind, pubs] TO DISK =
''C:\Backups\<AUTO>_01'', TO DISK = ''D:\Backups\<AUTO>_02'' " '
```

Back up transaction logs for multiple databases

This example creates transaction log backups for databases *northwind* and *pubs* in the default location.

```
SQLBackupC.exe -I {instance name} -SQL "BACKUP LOGS [northwind, pubs] TO DISK =
'<AUTO>' "
```

```
EXECUTE master..sqlbackup '-SQL "BACKUP LOGS [northwind, pubs] TO DISK = ''<AUTO>'' "
'
```

Create differential backups for multiple databases using threads

This example creates differential backups for databases *northwind* and *pubs* using two threads.

```
SQLBackupC.exe -I {instance name} -SQL "BACKUP DATABASES [northwind, pubs] TO DISK =
'C:\Backups\<AUTO>' WITH THREADCOUNT = 2, DIFFERENTIAL "
```

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASES [northwind, pubs] TO DISK =
''C:\Backups\<AUTO>'' WITH THREADCOUNT = 2, DIFFERENTIAL " '
```

Create a partial filegroup backup

This example backs up the read/write files in the *FileGroupTest* database using the standard Transact-SQL argument `READ_WRITE_FILEGROUPS`. For detailed information about this argument, refer to your SQL Server documentation about the BACKUP command.

```
SQLBackupC.exe -I {instance name} -SQL "BACKUP
DATABASE [FileGroupTest] READ_WRITE_FILEGROUPS TO DISK = 'C:\Backups\<AUTO>' "
```

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASE [FileGroupTest] READ_WRITE_FILEGROUPS
TO DISK = ''C:\Backups\<AUTO>'' " '
```

Create a partial differential filegroup backup

This example creates a differential backup of the read/write files in the *FileGroupTest* database using the standard Transact-SQL argument `READ_WRITE_FILEGROUPS`. For detailed information about this argument, refer to your SQL Server documentation about the `BACKUP` command.

```
SQLBackupC.exe -I {instance name} -SQL "BACKUP
DATABASE [FileGroupTest] READ_WRITE_FILEGROUPS TO DISK = 'C:\Backups\<AUTO>' WITH
DIFFERENTIAL"
```

```
EXECUTE master..sqlbackup '-SQL "BACKUP DATABASE [FileGroupTest] READ_WRITE_FILEGROUPS
TO DISK = ''C:\Backups\<AUTO>'' WITH DIFFERENTIAL" '
```

The RESTORE command

Use the RESTORE command with the SQL Backup -SQL parameter to restore a SQL Backup backup using the command line or extended stored procedure.

- Syntax provides the grammar for the RESTORE command.
- Arguments describes the arguments for the RESTORE command.
- WITH options describes the options that can be used in the RESTORE command.
- Examples provides examples of using the command line and extended stored procedure to restore with a range of options.

> When using the extended stored procedure the parameter or set of parameters (such as -SQL) must be delimited by single quotes. Therefore, wherever a single quote is used for the arguments below, for the extended stored procedure you must use **two** single quotes so that SQL Server does not interpret it as a string delimiter. See Using the extended stored procedure for more information.

Syntax

The following arguments are only available with SQL Server 2005 and later:

- CHECKSUM / NO_CHECKSUM
- CONTINUE_AFTER_ERROR / STOP_ON_ERROR
- PARTIAL

The KEEP_CDC argument is only available with SQL Server 2008 and later.

Restore an entire database

```
RESTORE DATABASE { database_name }
[ FROM
{
        {DISK} = { 'physical_backup_device_name' | 'file_search_pattern' } [ ,...n ]
        [
           [ LATEST_FULL | LATEST_DIFF | LATEST_ALL ]
           |
           SOURCE = 'source_database_name' { LATEST_FULL | LATEST_DIFF | LATEST_ALL }
        ]
   |
        {BACKUPHISTORY} [ = 'history_database_name' ]
        { LATEST_FULL | LATEST_DIFF | LATEST_ALL }
   }
]
[ WITH
   [ [ , ] { CHECKSUM | NO_CHECKSUM } ]
   [ [ , ] { CONTINUE_AFTER_ERROR | STOP_ON_ERROR } ]
   [ [ , ] DISCONNECT_EXISTING ]
   [ [ , ] DISKRETRYCOUNT = { n } ]
   [ [ , ] DISKRETRYINTERVAL = { n } ]
   [ [ , ] ERASEFILES = { days | hours{h} | except latest{b} } ]
   [ [ , ] ERASEFILES_REMOTE = { days | hours{h} | except latest{b} } ]
   [ [ , ] FILEOPTIONS = { 1 | 2 | 3 } ]
   [ [ , ] KEEP_CDC ]
   [ [ , ] KEEP_REPLICATION ]
   [ [ , ] LOG_ONERROR ]
   [ [ , ] LOG_ONERRORONLY ]
   [ [ , ] LOGTO = { 'target_folder_name' | 'file_name' } ] [ ,...n ]
   [ [ , ] MAILTO = { 'recipients' } ]
   [ [ , ] MAILTO_NOLOG ]
   [ [ , ] MAILTO_ONERROR = { 'recipients' } ]
   [ [ , ] MAILTO_ONERRORONLY = { 'recipients' } ]
   [ [ , ] MAXTRANSFERSIZE = { 65536 | 131072 | ... | 1048576 } ]
   [ [ , ] MOVE 'logical_file_name' TO 'operating_system_file_name' ] [ ,...n ]
   [ [ , ] MOVETO = { 'target_folder_name' } ]
   [ [ , ] NOLOG ]
   [ [ , ] { NORECOVERY | RECOVERY | STANDBY = 'standby_file_name' } ]
   [ [ , ] ORPHAN_CHECK ]
   [ [ , ] PASSWORD = { 'password' } ]
   [ [ , ] REPLACE ]
   [ [ , ] RESTRICTED_USER ]
```

```
   [ [ , ] SINGLERESULTSET ]
   [ [ , ] THREADPRIORITY = { 0 | 1 | 2 | 3 | 4 | 5 | 6 } ]
]
```

Restore part of a database

```
RESTORE DATABASE { database_name }
   [ FILE = { 'logical_file_name' } | FILEGROUP = { 'logical_filegroup_name' } | PAGE = { 'file:page' } ] [ ,...n ]
FROM DISK = { 'physical_backup_device_name' | 'file_search_pattern' } [ ,...n ]
[ WITH
   PARTIAL
   [ [ , ] { CHECKSUM | NO_CHECKSUM } ]
   [ [ , ] { CONTINUE_AFTER_ERROR | STOP_ON_ERROR } ]
   [ [ , ] DISCONNECT_EXISTING ]
   [ [ , ] DISKRETRYCOUNT = { n } ]
   [ [ , ] DISKRETRYINTERVAL = { n } ]
   [ [ , ] ERASEFILES = { days | hours{h} | except latest{b} } ]
   [ [ , ] ERASEFILES_REMOTE = { days | hours{h} | except latest{b} } ]
   [ [ , ] FILEOPTIONS = { 1 | 2 | 3 } ]
   [ [ , ] LOG_ONERROR ]
   [ [ , ] LOG_ONERRORONLY ]
   [ [ , ] LOGTO = { 'target_folder_name' | 'file_name' } ] [ ,...n ]
   [ [ , ] MAILTO = { 'recipients' } ]
   [ [ , ] MAILTO_NOLOG ]
   [ [ , ] MAILTO_ONERROR = { 'recipients' } ]
   [ [ , ] MAILTO_ONERRORONLY = { 'recipients' } ]
   [ [ , ] MOVE 'logical_file_name' TO 'operating_system_file_name' ] [ ,...n ]
   [ [ , ] MOVETO = { 'target_folder_name' } ]
   [ [ , ] NOLOG ]
   [ [ , ] NORECOVERY ]
   [ [ , ] ORPHAN_CHECK ]
   [ [ , ] PASSWORD = { 'password' } ]
   [ [ , ] REPLACE ]
   [ [ , ] RESTRICTED_USER ]
   [ [ , ] SINGLERESULTSET ]
   [ [ , ] THREADPRIORITY = { 0 | 1 | 2 | 3 | 4 | 5 | 6 } ]
]
```

Restore a transaction log

```
RESTORE LOG { database_name }
   [ FILE = { 'logical_file_name' } | FILEGROUP = { 'logical_filegroup_name' } | PAGE = { 'file:page' } ] [ ,...n ]
[ FROM { DISK } = { 'physical_backup_device_name' | 'file_search_pattern' } ] [ ,...n ]
[ WITH
   [ [ , ] { CHECKSUM | NO_CHECKSUM } ]
   [ [ , ] { CONTINUE_AFTER_ERROR | STOP_ON_ERROR } ]
   [ [ , ] DISCONNECT_EXISTING ]
   [ [ , ] DISKRETRYCOUNT = { n } ]
   [ [ , ] DISKRETRYINTERVAL = { n } ]
   [ [ , ] ERASEFILES = { days | hours{h} | except latest{b} } ]
   [ [ , ] ERASEFILES_REMOTE = { days | hours{h} | except latest{b} } ]
   [ [ , ] FILEOPTIONS = { 1 | 2 | 3 } ]
   [ [ , ] KEEP_CDC ]
   [ [ , ] KEEP_REPLICATION ]
   [ [ , ] LOG_ONERROR ]
   [ [ , ] LOG_ONERRORONLY ]
   [ [ , ] LOGTO = { 'target_folder_name' | 'file_name' } ] [ ,...n ]
   [ [ , ] MAILTO = { 'recipients' } ]
   [ [ , ] MAILTO_NOLOG ]
   [ [ , ] MAILTO_ONERROR = { 'recipients' } ]
   [ [ , ] MAILTO_ONERRORONLY = { 'recipients' } ]
   [ [ , ] MAXTRANSFERSIZE = { 65536 | 131072 | ... | 1048576 } ]
   [ [ , ] MOVE 'logical_file_name' TO 'operating_system_file_name' ] [ ,...n ]
   [ [ , ] MOVETO = { 'target_folder_name' } ]
   [ [ , ] NOLOG ]
   [ [ , ] { NORECOVERY | RECOVERY | STANDBY = 'standby_file_name' } ]
   [ [ , ] ORPHAN_CHECK ]
   [ [ , ] PASSWORD = { 'password' } ]
   [ [ , ] REPLACE ]
```

```
   [ [ , ] RESTRICTED_USER ]
   [ [ , ] SINGLERESULTSET ]
   [ [ , ] { STOPAT = { 'date_time' | @date_time_var } ]
    | STOPATMARK = { 'mark_name' | 'lsn:lsn_number' }
        [ AFTER 'datetime' ]
    | STOPBEFOREMARK = { 'mark_name' | 'lsn:lsn_number' }
        [ AFTER 'datetime']
   } ]
   [ [ , ] THREADPRIORITY = { 0 | 1 | 2 | 3 | 4 | 5 | 6 } ]
]
```

Restore file list

```
RESTORE FILELISTONLY
[ FROM { DISK } = { 'physical_backup_device_name' } ]
[ WITH
   [ [ , ] { CHECKSUM | NO_CHECKSUM } ]
   [ [ , ] MAXTRANSFERSIZE = { 65536 | 131072 | ... | 1048576 } ]
   [ [ , ] PASSWORD = { 'password' } ]
   [ [ , ] SINGLERESULTSET ]
]
```

> You cannot use wildcards in the FROM DISK argument with RESTORE FILELISTONLY.

Restore header

```
RESTORE HEADERONLY
[ FROM { DISK } = { 'physical_backup_device_name' } ]
[ WITH
   [ [ , ] { CHECKSUM | NO_CHECKSUM } ]
   [ [ , ] MAXTRANSFERSIZE = { 65536 | 131072 | ... | 1048576 } ]
   [ [ , ] PASSWORD = { 'password' } ]
   [ [ , ] SINGLERESULTSET ]
]
```

We recommend you use the SQL Backup command RESTORE SQBHEADERONLY to retrieve the header information for SQL Backup backup files, because it is much quicker than using the native command RESTORE HEADERONLY. For details, see The RESTORE SQBHEADERONLY command.

> You cannot use wildcards in the FROM DISK argument with RESTORE HEADERONLY.

Verify backup set

```
RESTORE VERIFYONLY
[ FROM { DISK } = { 'physical_backup_device_name' } ] [ ,...n ]
[ WITH
   [ [ , ] { CHECKSUM | NO_CHECKSUM } ]
   [ [ , ] MAILTO = { 'recipients' } ]
   [ [ , ] MAILTO_NOLOG ]
   [ [ , ] MAILTO_ONERROR = { 'recipients' } ]
   [ [ , ] MAILTO_ONERRORONLY = { 'recipients' } ]
   [ [ , ] MAXTRANSFERSIZE = { 65536 | 131072 | ... | 1048576 } ]
   [ [ , ] PASSWORD = { 'password' } ]
   [ [ , ] SINGLERESULTSET ]
]
```

> You cannot use wildcards in the FROM DISK argument with RESTORE VERIFYONLY.

Arguments

DATABASE argument

It is only possible to restore one database at a time. The database name must be enclosed in square brackets [ ] if it includes reserved words or spaces; if the database name does not include reserved words or spaces, square brackets are optional. For example:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\pubs\FULL_20120229.sqb'"
```

[FILE = 'logical_file_name' | FILEGROUP = 'logical_filegroup_name' | PAGE = 'file:page'] [,...n]

Specifies the files, filegroups or pages of the database that are to be restored. There is no limit on the number of files, filegroups or pages that can be restored, provided thay all belong to the same database. Identify files or filegroups using logical names. For example:

```
"RESTORE DATABASE pubs FILE = 'SalesF1', FILE = 'SalesF2' FROM DISK =
'C:\Backups\pubs\salesFiles.sqb'"
```

If the database is using the simple recovery model, the specified files or filegroups must be read-only unless `WITH PARTIAL` is specified.

Individual pages of read/write filegroups can be restored, provided the database is using the full or bulk-logged recovery model. Identify the pages to be restored in the format `PAGE = 'fileID:pageID'`. To restore multiple pages, use the format `PAGE = 'fileID:pageID, fileID:pageID'`.

For more information, refer to your SQL Server documentation.

LOG argument

To restore transaction log backups, the database must be in an unrecovered or standby state (see `NORECOVERY` and `STANDBY` below). The database name must be enclosed in square brackets [ ] if it includes reserved words or spaces; if the database name does not include reserved words or spaces, square brackets are optional. For example:

```
"RESTORE LOG [pubs] FROM DISK = 'C:\Backups\pubs\LOG__20120229_111500.sqb'"
```

To restore log backups using successive restore commands, include `WITH NORECOVERY` in each `RESTORE LOG` command. For example:

```
"RESTORE LOG [pubs] FROM DISK = 'C:\Backups\pubs\LOG_20120229_111500.sqb' WITH
NORECOVERY"
"RESTORE LOG [pubs] FROM DISK = 'C:\Backups\pubs\LOG_20120229_113000.sqb' WITH
NORECOVERY"
```

Restore the last log `WITH RECOVERY` to recover the database to a usable state. For information on restoring multiple log backups in one `RESTORE` command, see the `FROM DISK` argument below.

FILE = 'logical_file_name' | FILEGROUP = 'logical_filegroup_name' | PAGE = 'file:page'] [,...n]

Specifies the files, filegroups or pages of the database that are to be restored. There is no limit on the number of files, filegroups or pages that can be restored, provided thay all belong to the same database. Identify files or filegroups using logical names. For example:

```
"RESTORE LOG pubs FILE = 'SalesF1', FILE = SalesF2' FROM DISK =
'C:\Backups\pubs\SalesFiles.sqb'"
```

If the database is using the simple recovery model, the specified files or filegroups must be read-only unless `WITH PARTIAL` is specified.

Individual pages of read/write filegroups can be restored, provided the database is using the full or bulk-logged recovery model. Identify the pages to be restored in the format `PAGE = 'fileID:pageID'`. To restore multiple pages, use the format `PAGE = 'fileID:pageID, fileID:pageID'`.

For more information, refer to your SQL Server documentation.

FROM DISK argument

You can specify up to 32 DISK values. This is useful when you have split a backup across multiple files. You can also enter a 'file search pattern' by specifying wildcard characters in the physical backup device name. All files that match the wildcard characters must belong to the same backup set. For example, instead of:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\pubs_01.sqb', DISK =
'C:\Backups\pubs_02.sqb', DISK = 'C:\Backups\pubs_03.sqb', DISK =
'C:\Backups\pubs_04.sqb', DISK = 'C:\Backups\pubs_05.sqb'"
```

you can enter:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\pubs_*.sqb'"
```

You can also use the file search pattern to restore multiple transaction log backups. The database that the logs are restored to must be in an unrecovered state, that is, it must have been restored using the `NORECOVERY` or `STANDBY` option. For example:

```
"RESTORE LOG [pubs] FROM DISK = 'C:\Backups\Pubs\Logs*.*'"
```

SQL Backup ensures that the files are restored in the correct sequence. To specify a number of folders, use the `DISK` command repeatedly. For example:

```
"RESTORE LOG [pubs] FROM DISK = 'C:\Backups\Pubs\Logs*.*', DISK =
'E:\OtherBackups\Pubs\Logs*.*'"
```

The backup files that match the wildcard characters must belong to the same backup set. Any encrypted files must use the same password.

> You cannot use wildcard characters in the `FROM DISK` argument for `RESTORE FILELISTONLY`, `RESTORE HEADERONLY`, and `RESTORE VERIFYONLY` commands.

LATEST_FULL

> LATEST_FULL is only available in SQL Backup 6.4 and later.

You can use the optional `LATEST_FULL` keyword to select the most recent full backup of the destination database that matches the `DISK` values you specify. The `DISK` values you specify must contain the '*' wildcard. For example:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\pubs*.sqb' LATEST_FULL"
```

will find backup files of the destination database in the *C:\Backups* directory with a file name matching *pubs*.sqb*, and will then restore the latest full backup. You can specify multiple disk values; this is useful if you have split a backup across multiple files. The files that match the file search pattern must belong to the same backup set.

Note that you cannot use `LATEST_FULL` when restoring a file or filegroup backup.

LATEST_DIFF

> LATEST_DIFF is only available in SQL Backup 6.4 and later.

You can use the optional `LATEST_DIFF` keyword to select the most recent differential backup of the destination database that matches the disk values you specify. You can specify multiple locations; this is useful if you have split the backup across multiple files. The `DISK` values you specify must contain the '*' wildcard. For example:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\sales*.sqb', DISK =
'D:\Backups\sales*.sqb' LATEST_DIFF"
```

will find all differential backup files of the destination database in the *C:\Backups* and *D:\Backups* directories with a file name matching *sales*.sqb*, and will then restore the latest differential backup. The files that match the file search pattern must belong to the same backup set.

Note that if you specify `LATEST_DIFF`, you must restore to a database that has already had the most recent full backup applied. You cannot use `LATEST_DIFF` when restoring a file or filegroup backup.

LATEST_ALL

> LATEST_ALL is only available in SQL Backup 6.5 and later.

If you specify the `LATEST_ALL` keyword, the most recent full backup of the destination database will be restored, followed by the most recent differential backup (if one exists), and then finally by the most recent transaction log backups (if any exist). The `DISK` values you specify must contain the '*' wildcard. For example:

```
"RESTORE DATABASE pubs FROM DISK = 'C:\Backups\pubs*.sqb' LATEST_ALL"
```

will find backup files of the destination database in the *C:\Backups* directory with a file name matching *pubs*.sqb*, and will then restore the latest full backup, the latest differential backup (if applicable) and the latest log backups (if applicable). All the files that match the file search pattern must belong to the same backup set.

You can specify multiple disk values. This is useful if you store different backup types in separate locations or if you have split backups across multiple files. For example:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\FULL_pubs*.sqb', DISK =
'D:\Backups\DIFF_pubs*.sqb', DISK = 'E:\Backups\LOG_pubs*.sqb' LATEST_ALL"
```

> You cannot use `LATEST_ALL` when restoring a file or filegroup backup.

SOURCE

> SOURCE is only available in SQL Backup 6.4 and later.

You can restore the latest backup files taken from a database other than the destination database by including `SOURCE = 'source_database_name'`. For example:

```
"RESTORE DATABASE [Sales_Test] FROM DISK = 'D:\backups*.sqb' SOURCE = 'Sales_Prod'
LATEST_ALL"
```

will restore the latest full backup of the *Sales_Prod* database (followed by subsequent differential and transaction log backups, if applicable), to the *Sales_Test* database.

Note that you can only use `SOURCE` if `LATEST_FULL`, `LATEST_DIFF` or `LATEST_ALL` is included in the `RESTORE` statement. You cannot use `SOURCE` when restoring a file or filegroup backup.

FROM BACKUPHISTORY argument

Use FROM BACKUPHISTORY when you want to restore the latest full, latest differential, or all the latest backups (including transaction log backups) for a particular database, without having to list the individual backup file locations. SQL Backup searches its own backup history to determine which backup files to restore. You must also specify the LATEST_FULL, LATEST_DIFF, or LATEST_ALL keyword. For example:

```
"RESTORE DATABASE [sales] FROM BACKUPHISTORY LATEST_FULL WITH RECOVERY, REPLACE"
```

will search the backup history for the 'sales' database, and will then restore the latest full backup over the current *sales* database.

To search the backup history of a different database, you can specify this as part of the BACKUPHISTORY parameter. For example:

```
"RESTORE DATABASE [sales_dev] FROM BACKUPHISTORY = 'sales' LATEST_FULL WITH RECOVERY,
REPLACE"
```

will search the backup history for the 'sales' database, and will then restore the latest full backup to the *sales_dev* database.

WITH options

CHECKSUM|NO_CHECKSUM

By default, if the backup process included WITH CHECKSUM the backup checksum and any page checksums are validated on restore. If the backup does not include a backup checksum, any page checksums will not be validated. Specify NO_CHECKSUM to disable default validation of checksums. If you specify CHECKSUM, the backup checksum and any page checksums will be validated as by default, but if the backup does not include a backup checksum, an error is returned. For more information, refer to your SQL Server documentation.

CONTINUE_AFTER_ERROR|STOP_ON_ERROR

CONTINUE_AFTER_ERROR specifies that the RESTORE process should continue after an error is encountered, restoring what it can. This is the default behavior for RESTORE VERIFYONLY (see VERIFY in The BACKUP command). The RESTORE VERIFYONLY process then reports all errors it has encountered.

STOP_ON_ERROR specifies that the RESTORE process should stop if an error is encountered. This is the default behavior for RESTORE.

For more information, refer to your SQL Server documentation.

DISCONNECT_EXISTING

Kills any existing connections to the database before starting the restore. Restoring to an existing database will fail if there are any connections to the database.

DISKRETRYCOUNT

In combination with DISKRETRYINTERVAL, this argument controls network resilience behavior.

DISKRETRYCOUNT specifies the maximum number of times to retry a failed data-transfer operation (reading or moving a backup file). If you omit this keyword, the default value of 10 is used. If you specify a value for DISKRETRYCOUNT, you should also specify a value for DISKRETRYINTERVAL.

DISKRETRYINTERVAL

In combination with DISKRETRYCOUNT, this argument controls network resilience behavior.

DISKRETRYINTERVAL specifies the time interval between retries, in seconds, following a failed data-transfer operation (reading or moving a backup file). If you omit this keyword, the default value of 30 seconds is used. If you specify a value for DISKRETRYINTERVAL, you should also specify a value for DISKRETRYCOUNT.

ERASEFILES

Specifies the number of existing SQL Backup backups to be deleted from the MOVETO folder. This is useful for managing the number of backups

in the MOVETO folder when log shipping. **Note:** You must also include FILEOPTIONS.

You can choose to delete SQL Backup backups based on:

- Age: files older than the specified number of days or hours are deleted. Specify a number for days, or type *h* after the number for hours. For example, ERASEFILES = 24 deletes files that are more than 24 days old; ERASEFILES = 24h deletes files that are more than 24 hours old. Note that a day is calculated as a period of 24 hours, and takes no account of calendar date.
- Number of backups to keep: only the latest 'x' backups will be kept. To specify the number of backups to be kept, type *b* after the number.
  For example, ERASEFILES = 5b ensures the latest 5 backups are kept; older backups are deleted.

Files are deleted only if the following details match the details of the database being backed up:

- The name of the SQL Server, instance (if applicable), and database recorded in the file header.
- The backup type (full, differential, transaction log).
- The backup password. If the PASSWORD option is not specified (because the backup is not encrypted), any existing encrypted backups in the DISK location will not be identified by ERASEFILES because the file header cannot be read. This may result in backups older than the specified age or in excess of the specified number being retained.

> If SQL Backup cannot list the contents of the folder that contains the files to be deleted, it cannot delete the files. Ensure the SQL Backup Agent service startup account (or, if you are using the command line, the user account from which you are running *SQLBackupC.exe*) has permissions to list the folder contents.

Example

The following example restores a transaction log backup to the *pubs* database and leaves it in a non-operational state, then moves the backup to *C:\processed_logs* and deletes all transaction log backups of *pubs* other than the latest 10 from that folder:

```
"RESTORE LOG [pubs] FROM DISK = 'C:\shipped_logs\pubs\LOG_20120229_151009.sqb' WITH
MOVETO = 'C:\processed_logs', ERASEFILES = 10b, FILEOPTIONS = 1, NORECOVERY"
```

> If ERASEFILES and ERASEFILES_REMOTE are included in the same command, the ERASEFILES_REMOTE setting overrides the ERASEFILES setting for *remote* MOVETO locations.

ERASEFILES_REMOTE

Manages deletion of existing SQL Backup backups from *remote* MOVETO folders. This is useful for managing the number of files in the MOVETO folder when log shipping.

You can choose to delete SQL Backup files based on:

- Age: files older than the specified number of days or hours are deleted. Specify a number for days, or type *h* after the number for hours. For example, ERASEFILES_REMOTE = 24 deletes files that are more than 24 days old; ERASEFILES_REMOTE = 24h deletes files that are more than 24 hours old. Note that a day is calculated as a period of 24 hours, and takes no account of calendar date.
- Number of backups to keep: only the latest 'x' backups will be kept. To specify the number of backups to be kept, type *b* after the number. For example, ERASEFILES_REMOTE = 5b ensures the latest 5 backups are kept; older backups are deleted.

Files are deleted only if the following details match the details of the database being backed up:

- The name of the SQL Server, instance (if applicable), and database recorded in the file header.
- The backup type (full, differential, transaction log).
- The backup password. If the PASSWORD option is not specified (because the backup is not encrypted), any existing encrypted backups in the DISK location will not be identified by ERASEFILES_REMOTE because the file header cannot be read. This may result in backups older than the specified age or in excess of the specified number being retained.

> If SQL Backup cannot list the contents of the folder that contains the files to be deleted, it cannot delete the files. Ensure the SQL Backup Agent service startup account (or, if you are using the command line, the user account from which you are running *SQLBackupC.exe*) has permissions to list the folder contents.

Example

The following example restores a transaction log backup to the *pubs* database and leaves it in a non-operational state, then moves the backup to *\\Server01\processed_logs* and deletes all backups older than 6 hours from that folder:

```
"RESTORE LOG [pubs] FROM DISK = 'C:\shipped_logs\pubs\LOG_20120229_151009.sqb' WITH
MOVETO = '\\Server01\processed_logs', ERASEFILES_REMOTE = 6h, NORECOVERY"
```

To delete files from local `MOVETO` folders, use `ERASEFILES` and `FILEOPTIONS`.

FILEOPTIONS

Use in conjunction with `ERASEFILES`. Specifies whether backup files are to be deleted from the `MOVETO` folder. Specify the sum of the values that correspond to the options you require:

1	Delete backup files in the `MOVETO` folder if they are older than the number of days or hours specified in `ERASEFILES`.
2	Do not delete backup files in the `MOVETO` folder that are older than the number of days or hours specified in `ERASEFILES` if they have the `ARCHIVE` flag set.

Valid values are **1**, **2**, and **3**.

You must also set the age of the files to delete using `ERASEFILES`. For example, to delete backup files in the `MOVETO` folder that are older than 5 days:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\pubs\FULL_20120229.sqb' WITH MOVETO =
'C:\Backups\Archive\pubs', ERASEFILES = 5, FILEOPTIONS = 1"
```

To delete any existing files in the `MOVETO` folder that are older than 5 days and do not have the `ARCHIVE` flag set, (values 1 + 2):

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\pubs\FULL_20120229.sqb' WITH MOVETO =
'C:\Backups\Archive\pubs', ERASEFILES = 5, FILEOPTIONS = 3"
```

KEEP_CDC

> KEEP_CDC is only available in SQL Backup 6.5 and later.

Specifies that Change Data Capture settings are to be retained when a database or log is restored to another server.

This option cannot be included with `NORECOVERY`. Refer to your SQL Server documentation for more information.

KEEP_REPLICATION

This option is for use when log shipping is used in conjunction with replication. Specifies that replication settings are to be retained when a database or log is restored to a standby server.

This option cannot be included with `NORECOVERY`. Refer to your SQL Server documentation for more information.

LOG_ONERROR

Specifies that a log file should only be created if SQL Backup encounters an error during the restore process, or the restore completes successfully but with warnings. Use this option if you want to restrict the number of log files created by your restore processes, but maintain log information whenever warnings or errors occur. This argument controls the creation of log files on disk only; emailed log files are not affected. (See the `MAILTO` options below for details on emailing log files.)

LOG_ONERRORONLY

Specifies that a log file should only be created if SQL Backup encounters an error during the restore process. Use this option if you want to restrict the number of log files created by your restore processes, but maintain log information whenever errors occur. This argument controls the creation

of log files on disk only; emailed log files are not affected. (See the `MAILTO` options below for details on emailing log files.)

LOGTO

Specifies that a copy of the log file is to be saved.

By default, the primary log file is created in the folder *%ProgramData%\Red Gate\SQL Backup\Log* (Windows Vista, Windows 2008 and later) or *%ALLUSERSPROFILE%\Application Data\Red Gate\SQL Backup\Log* (Windows XP and Windows 2003); you can change this location in your file management options.

To create a copy with the same name as the primary log file, specify the folder. For example:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\pubs\FULL_20120229.sqb' WITH LOGTO =
'C:\Logs'"
```

To create a copy with a different name from the primary log file, specify the folder and file name. For example:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\pubs\FULL_20120229.sqb' WITH LOGTO =
'C:\Logs', LOGTO = 'C:\Logs\SQBSecondaryLog.txt'"
```

To copy the log file to more than one location, use multiple `LOGTO` commands.

MAILTO

Specifies that the outcome of the restore operation is emailed to one or more users; the email includes the contents of the log file. SQL Backup uses the settings specified in your email settings to send the email. To specify multiple recipients, separate the email addresses with a semi-colon (**;**). For example:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\pubs\FULL_20120229.sqb' WITH MAILTO =
'dba01@myco.com*;*dba02@myco.com'"
```

If you have not defined email settings, the email will not be sent and a warning will be reported.

MAILTO_NOLOG

Specifies that SQL Backup should not include the contents of the log file in the email. An email will still be sent to notify the specified recipients of success and/or failure, depending on which `MAILTO` parameter has been specified.

MAILTO_ONERROR

Specifies that that the outcome of the restore operation is emailed to one or more users if SQL Backup encounters an error during the restore process or the restore process completes successfully but with warnings. The email includes the contents of the log file. SQL Backup uses the settings specified in your email settings to send the email. To specify multiple recipients, separate the email addresses with a semi-colon (**;**). For example:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\pubs\FULL_20120229.sqb' WITH
MAILTO_ONERROR = 'dba01@myco.com;dba02@myco.com'"
```

If you have not defined email settings, the email will not be sent and a warning will be reported.

MAILTO_ONERRORONLY

Specifies that that the outcome of the restore operation is emailed to one or more users if SQL Backup encounters an error during the restore process. The email includes the contents of the log file. SQL Backup uses the settings specified in your email settings to send the email. To specify multiple recipients, separate the email addresses with a semi-colon (**;**). For example:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\pubs\FULL_20120229.sqb' WITH
MAILTO_ONERRORONLY = 'dba01@myco.com;dba02@myco.com'"
```

If you have not defined email settings, the email will not be sent and a warning will be reported.

MAXTRANSFERSIZE

Specifies the maximum size of each block of memory to be used when SQL Backup restores backup data. You may want to specify this argument if a SQL Server reports that it has insufficient memory to service requests from SQL Backup.

Valid values are integers in multiples of 65536, up to a maximum value of 1048576. For example:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\pubs\FULL_20120229.sqb' WITH
MAXTRANSFERSIZE = 262144"
```

If not specified, defaults to 1048576. However, if you have created the following DWORD registry key, SQL Backup uses the defined value as the default value:

HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\SQL Backup\BackupSettingsGlobal\<instance name>\MAXTRANSFERSIZE

MOVE '*logical_file_name*' TO '*operating_system_file_name*'

Specifies that the data file, log file, full text catalog (SQL Server 2005 only) or filestream data (SQL Server 2008 or later) identified by the logical file name should be restored to the physical location specified. The location must exist before the `RESTORE` command is executed. This option can also be used to rename the physical files. For example:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\pubs\FULL_20120229.sqb' WITH MOVE
'pubs_data' TO 'F:\Pubs02\Data\pubs_data02'"
```

MOVETO

Specifies that the backup files should be moved to another folder when the restore process completes. If the folder you specify does not exist, it will be created.

You must ensure that you have permission to delete files from the original folder, and to write to the MOVETO folder.

You can also use tags with the `MOVETO` argument, for example:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\pubs\FULL_20120229.sqb' WITH MOVETO =
'C:\Backups\Archive\<INSTANCE>\<DATABASE>\'"
```

NOLOG

Prevents a log file from being created for the restore process, even if errors or warnings are generated. You may want to use this option if you are concerned about generating a large number of log files, and are certain that you will not need to review the details of errors or warnings (for example, because it's possible to run the process again without needing to know why it failed). This argument controls the creation of log files on disk only; emailed log files are not affected. (See the `MAILTO` options above for details on emailing log files.)

NORECOVERY

Specifies that incomplete transactions are not to be rolled back on restore. The database cannot be used but differential backups and transaction log backups can be restored. For more information, refer to your SQL Server documentation.

ORPHAN_CHECK

Specifies that once the restore has completed, the database should be checked for orphaned users. Database user names are considered to be orphaned if they do not have a corresponding login defined on the SQL Server instance. Orphaned users are often created when you restore a database backup to a different SQL Server instance.

Note that `ORPHAN_CHECK` will only detect database users that are based on SQL Server logins; orphaned users based on Windows principals are not detected.

If orphaned users are detected, warning 472 is generated and each orphaned user is listed in the SQL Backup log file along with the associated SID.

PARTIAL

Specifies a partial restore of a database. The primary filegroup is restored, together with any specified secondary filegroups. See the DATABASE argument above for details on how to specify particular filegroups in a restore operation.

For more information, refer to your SQL Server documentation.

PASSWORD

Specifies the password to be used with encrypted backup files.

You cannot use the encrypted form of the password. This is to prevent unauthorized users from restoring backups if they have access to the encrypted password from the backup script.

SQL Backup version 3 allowed the use of encrypted passwords; these will no longer work. You must specify the password in unencrypted form:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\pubs\FULL_20120229.sqb' WITH PASSWORD
= 'Password'"
```

RECOVERY

Specifies that incomplete transactions are to be rolled back. Recovery is completed and the database is in a usable state. Further differential backups and transaction log backups cannot be restored.

If no recovery completion state is specified, `WITH RECOVERY` is the default behavior. For more information, refer to your SQL Server documentation.

REPLACE

Specifies that the database should be restored, even if another database of that name already exists. The existing database will be deleted. `REPLACE` is required to prevent a database of a different name being overwritten by accident.

`REPLACE` is not required to overwrite a database which matches the name recorded in the backup.

For more information, refer to your SQL Server documentation.

RESTRICTED_USER

Specifies that access to the restored database is to be limited to members of the db_owner, dbcreator or sysadmin roles. Return the database to multi-user or single-user mode using your SQL Server application. For more information, refer to your SQL Server documentation.

SINGLERESULTSET

Specifies that the results returned by the RESTORE command should be limited to just one result set. This may be useful if you want to manipulate results using a Transact-SQL script. Such scripts can only manipulate results when a single result set is returned. The `RESTORE` command will return two result sets by default in most cases, unless you specify the `SINGLERESULTSET` keyword.

STANDBY

Specifies a standby file that allows the recovery effects to be undone. The STANDBY option is allowed for offline restore (including partial restore). The option is disallowed for online restore.

Refer to your SQL Server documentation for more information about the STANDBY argument.

'standby_file_name'

Is a standby file used to keep a "copy-on-write" pre-image for pages modified during the undo pass of a RESTORE WITH STANDBY.

When used with the RESTORE DATABASE or RESTORE LOG command, '**standby_file_name**' can include tags, but these are not required.

STOPAT

Specifies a point in time to which a transaction log backup should be restored. The database will be recovered up to the last transaction commit that occurred at or before the specified time. When restoring to a point in time, include this option in each RESTORE LOG statement. For example:

```
"RESTORE DATABASE [pubs] FROM DISK = 'C:\Backups\pubs_full.sqb' WITH NORECOVERY'"
"RESTORE LOG [pubs] FROM DISK = 'C:\Backups\pubs_log_20120601093000.sqb' WITH
NORECOVERY, STOPAT = '2012-06-01T09:40:30'"
"RESTORE LOG [pubs] FROM DISK = 'C:\Backups\pubs_log_20120601094500.sqb' WITH
RECOVERY, STOPAT = '2012-06-01T09:40:30'"
```

For more information, refer to your SQL Server documentation.

STOPATMARK

Specifies the point to which a transaction log backup should be restored, using either the log sequence number or a marked transaction. The database will be recovered up to and including the log record that contains the specified LSN or the marked transaction.

AFTER can be used when specifying a marked transaction and is useful when the mark name is not unique. The database is recovered as far as the first marked transaction to have occurred on or after the specified time.

For more information, refer to your SQL Server documentation.

STOPBEFOREMARK

Specifies the point to which a transaction log backup should be restored, using either the log sequence number or a marked transaction. The database will be recovered up to but excluding the log record that contains the specified LSN or the marked transaction.

AFTER can be used when specifying a marked transaction and is useful when the mark name is not unique. The database is recovered up to the first marked transaction to have occurred on or after the specified time.

For more information, refer to your SQL Server documentation.

THREADPRIORITY

Sets the SQL Backup thread priority when the backup or restore process is run. Valid values are **0** to **6**, and correspond to the following priorities:

0	Idle
1	Very low
2	Low
3	Normal
4	High
5	Very high
6	Time critical

If this value is not specified, normal priority is used.

Examples

Restore a database from a single file

This example restores a full backup of the pubs database from a single file.

```
SQLBackupC.exe -I {instance name} -SQL "RESTORE DATABASE [pubs] FROM DISK =
'C:\Backups\pubs_01.sqb' WITH REPLACE"
```

```
EXECUTE master..sqlbackup '-SQL "RESTORE DATABASE [pubs] FROM DISK =
''C:\Backups\pubs_01.sqb'' WITH REPLACE" '
```

Restore a database from multiple (split) backup files

This example restores a full backup of the *pubs* database from two files.

```
SQLBackupC.exe -I {instance name} -SQL "RESTORE DATABASE [pubs] FROM DISK =
'C:\Backups\pubs_01.sqb', DISK = 'C:\Backups\pubs_02.sqb' WITH REPLACE"
```

```
EXECUTE master..sqlbackup '-SQL "RESTORE DATABASE [pubs] FROM DISK =
''C:\Backups\pubs_01.sqb'', DISK = ''C:\Backups\pubs_02.sqb'' WITH REPLACE" '
```

Restore a database to a new name and move the database files

This example restores a full backup of the pubs database and restores it to a new database called *pubs02*. It also renames the database data and log files and moves them to a new location.

```
SQLBackupC.exe -I {instance name} -SQL "RESTORE DATABASE [pubs02] FROM DISK =
'C:\Backups\pubs_01.sqb' WITH MOVE 'pubs' TO 'E:\Data\pubs02.mdf', MOVE 'pubs_log' TO
'E:\Data\pubs02.ldf' "
```

```
EXECUTE master..sqlbackup '-SQL "RESTORE DATABASE [pubs02] FROM DISK =
''C:\Backups\pubs_01.sqb'' WITH MOVE ''pubs'' TO ''E:\Data\pubs02.mdf'', MOVE
''pubs_log'' TO ''E:\Data\pubs02.ldf'' " '
```

Restore a database from the latest full backup on different disks

This example searches multiple disk locations for full backups of the *pubs* database that match the file name pattern and restores the latest full backup.

```
SQLBackupC.exe -I {instance name} -SQL "RESTORE DATABASE [pubs] FROM DISK =
'C:\Backups*pubs*.sqb', DISK = 'D:\Backups*pubs*.sqb', DISK = 'E:\Backups*pubs*.sqb'
LATEST_FULL WITH REPLACE"
```

```
EXECUTE master..sqlbackup '-SQL "RESTORE DATABASE [pubs] FROM DISK =
''C:\Backups*pubs*.sqb'', DISK = ''D:\Backups*pubs*.sqb'', DISK =
''E:\Backups*pubs*.sqb'' LATEST_FULL WITH REPLACE
```

Restore a database from the latest backup set on different disks

This example searches multiple disk locations for full, differential and transaction log backups of the *pubs* database that match the file name pattern, and restores the most recent full backup, followed by the most recent differential backup and the most recent transaction log backups.

```
SQLBackupC.exe -I {instance name} -SQL "RESTORE DATABASE [pubs] FROM DISK =
'C:\Backups*pubs*.sqb', DISK = 'D:\Backups*pubs*.sqb', DISK = 'E:\Backups*pubs*.sqb'
LATEST_ALL WITH REPLACE"
```

```
EXECUTE master..sqlbackup '-SQL "RESTORE DATABASE [pubs] FROM DISK =
''C:\Backups*pubs*.sqb'', FROM DISK = ''D:\Backups*pubs*.sqb'', DISK =
''E:\Backups*pubs*.sqb'' LATEST_ALL WITH REPLACE" '
```

Restore to a new database from the latest backup set and check for orphaned users

This example restores the most recent full backup of the *pubs* database, followed by the most recent differential backup and the most recent transaction log backups available from C:\Backups to a new database called pubs02 and checks for orphaned users.

```
SQLBackupC.exe -I {instance name} -SQL "RESTORE DATABASE [pubs02] FROM DISK =
'C:\Backups\pubs*.sqb' SOURCE = 'pubs' LATEST_ALL WITH ORPHAN_CHECK"
```

```
EXECUTE master..sqlbackup '-SQL "RESTORE DATABASE [pubs02] FROM DISK =
''C:\Backups\pubs*.sqb'' SOURCE = ''pubs'' LATEST_ALL WITH ORPHAN_CHECK" '
```

Restore a database from an encrypted backup file

This example restores an encrypted backup of the *pubs* database, specifying the password MyPassword.

```
SQLBackupC.exe -I {instance name} -SQL "RESTORE DATABASE [pubs] FROM DISK =
'C:\Backups\pubs_01.sqb' WITH PASSWORD = 'MyPassword' "
```

```
EXECUTE master..sqlbackup '-SQL "RESTORE DATABASE [pubs] FROM DISK =
''C:\Backups\pubs_01.sqb'' WITH PASSWORD = ''MyPassword'' "
```

Restore a database in NORECOVERY mode

This example restores a full backup of the *pubs* database, specifying that the database is to be left in an unrecovered state so that differential and transaction log backups can be restored to it.

```
SQLBackupC.exe -I {instance name} -SQL "RESTORE DATABASE [pubs] FROM DISK =
'C:\Backups\pubs_01.sqb' WITH NORECOVERY"
```

```
EXECUTE master..sqlbackup '-SQL "RESTORE DATABASE [pubs] FROM DISK =
''C:\Backups\pubs_01.sqb'' WITH NORECOVERY"
```

Restore a database in READ-ONLY mode

This example restores a full backup of the *pubs* database, specifying that the database should be left in an unrecovered, read-only state so that its data can be viewed and differential and transaction log backups can be restored to it. The location of the standby file is specified.

```
SQLBackupC.exe -I {instance name} -SQL "RESTORE DATABASE [pubs] FROM DISK =
'C:\Backups\pubs_01.sqb' WITH STANDBY = 'C:\Standby\pubs_log.DAT' "
```

```
EXECUTE master..sqlbackup '-SQL "RESTORE DATABASE [pubs] FROM DISK =
''C:\Backups\pubs_01.sqb'' WITH STANDBY = ''C:\Standby\pubs_log.DAT'' " '
```

The RESTORE SQBHEADERONLY command

Use the `RESTORE SQBHEADERONLY` command with the SQL Backup `-SQL` parameter to retrieve the header information associated with a SQL Backup backup file using the command line or extended stored procedure.

For compressed backup files, using `RESTORE SQBHEADERONLY` to view header information is much quicker than using the equivalent SQL Server command, which necessarily requires that the entire backup file is uncompressed.

> When using the extended stored procedure the parameter or set of parameters (such as `-SQL`) must be delimited by single quotes. Therefore, wherever a single quote is used for the arguments below, for the extended stored procedure you must use **two** single quotes so that SQL Server does not interpret it as a string delimiter. See Using the extended stored procedure for more information.

Syntax

Restore a backup header

```
RESTORE SQBHEADERONLY
[ FROM { DISK } = { 'physical_backup_device_name' } ]
[ WITH
   [ [ , ] PASSWORD = { 'password' } ]
   [ [ , ] SINGLERESULTSET ]
]
```

You can use wildcard characters (*) for the location of the backups (*physical_backup_device_name*) to display the headers of multiple files.

WITH options

PASSWORD

Specifies the password to be used with encrypted backup files. If you are restoring multiple headers at the same time, the encrypted files must all have the same password.

SINGLERESULTSET

Specifies that the results returned by the `RESTORE SQBHEADERONLY` command should be limited to just one result set. This may be useful if you want to manipulate results using a Transact-SQL script. Such scripts can only manipulate results when a single result set is returned. The `RESTORE SQBHEADERONLY` command will return two result sets by default unless you specify the `SINGLERESULTSET` keyword.

Examples

Restoring a header file

This example retrieves the header information for the *pubs.sqb* database backup file.

```
SQLBackupC.exe -I {instance name} -SQL "RESTORE SQBHEADERONLY FROM DISK =
'C:\Backups\pubs.sqb'"
```

```
EXECUTE master..sqlbackup '-SQL "RESTORE SQBHEADERONLY FROM DISK =
''C:\Backups\pubs.sqb''"'
```

Restoring multiple header files

This example retrieves the header information for all database backup files in the *Backups* folder.

```
SQLBackupC.exe -I {instance name} -SQL "RESTORE SQBHEADERONLY FROM DISK =
'C:\Backups*.sqb'"
```

```
EXECUTE master..sqlbackup '-SQL "RESTORE SQBHEADERONLY FROM DISK =
''C:\Backups*.sqb''"'
```

The CONVERT command

Use the `CONVERT` command with the SQL Backup `-SQL` parameter to convert SQL Backup backup files (.sqb) to Microsoft Tape Format (MTF) files (.bak) using the command line or extended stored procedure. You can use the native SQL Server `RESTORE` command to restore MTF files.

Alternatively, you can use the SQL Backup File Converter (either from the command line, or using the GUI application). This application can also convert SQL Backup v7 or v6 files to SQL Backup v5 files; both file types have extension *.sqb*. See SQL Backup File Converter for more information.

If you convert a backup file that was created with multiple threads (THREADCOUNT > 1), an individual .bak file is created for each thread. For example, if backup file *BackUpA.sqb* was created using three threads, the following files are created when you convert the backup:

BackUpA_01.bak

BackUpA_02.bak

BackUpA_03.bak

You can then restore the MTF (.bak) format backups as a group of backup files.

> When using the extended stored procedure the parameter or set of parameters (such as `-SQL`) must be delimited by single quotes. Therefore, wherever a single quote is used for the arguments below, for the extended stored procedure you must use **two** single quotes so that SQL Server does not interpret it as a string delimiter. See Using the extended stored procedure for more information.

Syntax

```
CONVERT { 'p_hysical_backup_device_name' } ] TO { 'physical_file_name' }
[ WITH
   [ [ , ] MAXDATABLOCK = { 65536 | 131072 | ... | 2097152 } ]
   [ [ , ] PASSWORD = { 'password' } ]
   [ [ , ] SINGLERESULTSET ]
]
```

Arguments

MAXDATABLOCK

Specifies the maximum size of data blocks to be used when SQL Backup stores backup data. Valid values are integers in multiples of 65536, up to a maximum value of 2097152. For example:

```
MAXDATABLOCK = 655360
```

PASSWORD

Specifies the password to be used with encrypted SQL Backup backup files. The resulting MTF (.bak) format file will not be encrypted.

SINGLERESULTSET

Specifies that the results returned by the `CONVERT` command should be limited to just one result set. This may be useful if you want to manipulate results using a Transact-SQL script. Such scripts can only manipulate results when a single result set is returned. The `CONVERT` command will return two result sets by default unless you specify the `SINGLERESULTSET` keyword.

Example

This example converts the encrypted backup file *pubs.sqb* to an unencrypted MTF format file:

```
SQLBackupC.exe -I {instance name} -SQL "CONVERT 'C:\Backups\pubs.sqb' TO
'C:\Backups\pubs.bak' WITH PASSWORD = 'MyPassword'"
```

```
EXECUTE master..sqlbackup '-SQL "CONVERT ''C:\Backups\pubs.sqb'' TO
''C:\Backups\pubs.bak'' WITH PASSWORD = ''MyPassword''"'
```

The ERASE command

> The ERASE command is only available in SQL Backup 6.3 and later.

Use the ERASE command with the SQL Backup `-SQL` parameter to delete backup files based on their age, or the number of files you want to keep using the command line or extended stored procedure. The ERASE command works in a similar way to the `ERASEFILES` parameter used with the `BACKUP` and `RESTORE` commands, but enables you to delete backup files independently of a backup or restore operation.

You must also specify the database name, backup type, and a `PASSWORD` parameter if the files are encrypted.

Only files in the directory specified in the `FROM DISK` parameter are considered for deletion. Subdirectories are not checked.

> When using the extended stored procedure the parameter or set of parameters (such as *-SQL*) must be delimited by single quotes. Therefore, wherever a single quote is used for the arguments below, for the extended stored procedure you must use **two** single quotes so that SQL Server does not interpret it as a string delimiter. See Using the extended stored procedure for more information.

Syntax

```
ERASE { FULL_BACKUPS | DIFF_BACKUPS | LOG_BACKUPS }
FOR { database_name }
FROM {DISK} = { 'physical_device_name' } [,...n]
KEEP = { days | hours{h} | latest{b}}
[ WITH
   [ [ , ] PASSWORD = { 'password' } ]
]
```

Arguments

KEEP

You can choose to delete SQL Backup files based on:

- Age: files older than the specified number of days or hours are deleted. Type a number for days, or type *h* after the number for hours. For example, `KEEP = 24` deletes files that are more than 24 days old; `KEEP = 24h` deletes files that are more than 24 hours old. Note that a 'day' is calculated as a period of 24 hours, and takes no account of calendar date.
- Number of backups to keep: only the latest 'x' backups will be kept. To specify the number of backups to be kept, type *b* after the number. For example, `KEEP = 5b` ensures the latest 5 backups are kept; older backups are deleted.

> If SQL Backup cannot list the contents of the folder that contains the files to be deleted, it cannot delete the files. Ensure that the user account from which you are running SQL Backup has permissions to list the folder contents.

PASSWORD

Specifies the password to be used with encrypted backup files. (Specifying a password does not prevent unencrypted files from being deleted.)

If the ERASE command applies to more than one backup file, and these have different passwords, then only those backup files for which the supplied password is correct will be deleted.

Example

This example keeps the 7 most recent full backups of the *pubs* database in the folder *C:\Backups*; older full backups of the *pubs* database are deleted from the folder:

```
SQLBackupC.exe -SQL "ERASE FULL_BACKUPS FOR [pubs] FROM DISK = 'C:\Backups' KEEP = 7b
WITH PASSWORD = 'MyPassword'"
```

```
EXECUTE master..sqlbackup '-SQL "ERASE FULL_BACKUPS FOR [pubs] FROM DISK =
''C:\Backups'' KEEP = 7b WITH PASSWORD = ''MyPassword''"'
```

Settings and options

You can define file management options, email settings and hosted storage settings for each SQL Server that you have added to the SQL Backup graphical user interface (GUI). From the **Tools** menu select **Server Options**, then select the SQL Server from the drop-down list. For more information see:

- File management options
- Email settings

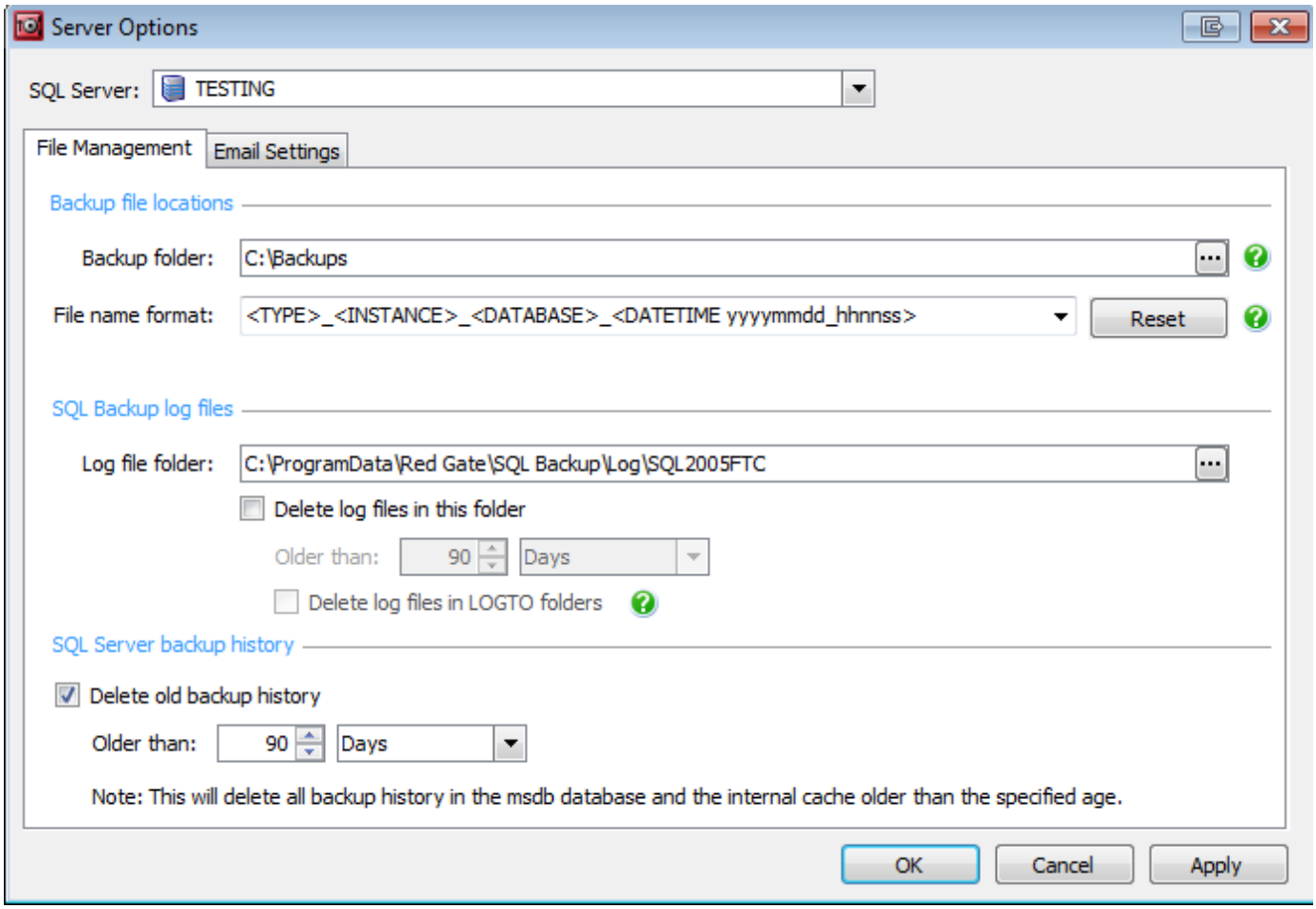
The following settings and options are not specific to each SQL Server and are not available in the **Server Options** dialog:

- Activity cache location
- File location tags
- Compression levels

File management options

You can define a number of options for file management that are used throughout SQL Backup. For example, you can specify the default folder for backup files.

On the **Tools** menu, click **Server Options**. The **File Management** tab is selected by default.



In the **SQL Server** box, select the name of the SQL Server instance for which you want to set the options. If the selected SQL Server does not have the server components installed, a warning is displayed and you must install the server components to proceed.

Backup file locations

You can set up a default folder and file name format for your backups, so that you do not have to enter the same information repeatedly when you create backups or backup jobs using the wizards. The settings are also used when you use the BACKUP command in the command line or an extended stored procedure.

In the **Backup folder** box, type the path for the folder in which you want your backups to be created by default, or click

to browse to the location. You can use file location tags in the path if required. For example, if you want the backup files for each database to be created in a separate folder, specify the <DATABASE> tag: *C:\MyBackups\<DATABASE>*

If you do not specify a location, SQL Backup uses the default SQL Server backup folder, which is usually *C:\Program Files\Microsoft SQL Server\MSSQL\BACKUP*.

In the **File name format** box, define the default file name format for backup files using file location tags. The default format is:

<TYPE><INSTANCE><DATABASE>_<DATETIME yyyymmdd_hhnnss>

For example, for a full database backup of the *SalesData* database running on the default SQL Server instance, the form of the backup file name would be *FULL_(local)_SalesData_20120229_005042.sqb* where the numbers show the date and time of the backup.

The settings are saved on your computer. You must have Full Control permissions for the following registry entries in order to make changes to

these settings:

- *HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\SQL Backup\BackupSettingsGlobal\<Instance_Name>\BackupFolder*
- *HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\SQL Backup\BackupSettingsGlobal\<Instance_Name>\BackupFileName*

For details of the tags you can use to specify the backup folder and file name format, see File location tags.

SQL Backup log files

When SQL Backup backs up or restores a database, the details of the operation are sent to a log file. You can specify the default folder for SQL Backup log files and set up SQL Backup to delete the log files at regular intervals. These settings are used whenever you run a backup or restore process.

In the **Log file folder** box, type the path for the folder in which you want your log files to be created by default, or click

to browse to the location. You can use tags to define the path. For more information on tags, see File location tags.

For optimal performance, you are recommended to use a dedicated folder for the SQL Backup log files. By default, the primary log file is created in:

- *%PROGRAMDATA%\Red Gate\SQL Backup\Log\<instance>* (on Windows Vista, Windows 2008 and later), or
- *%ALLUSERSPROFILE%\ Application Data\Red Gate\SQL Backup\Log\<instance>* (on Windows XP and Windows 2003).

If you want SQL Backup to delete old log files automatically, select the **Delete log files in this folder** check box, then type or select the age (in days or hours) of the files that you want to delete.

If you are scripting a SQL Backup job, you can automatically create a copy of the log file in a different location using the `LOGTO` argument with the `BACKUP` or `RESTORE` command. To delete these files in addition to the primary log files, select the **Delete log files in LOGTO folders** check box.

By scripting SQL Backup jobs you can also prevent *any* SQL Backup log files from being created on disk (using the `NOLOG` option), or specify that log files should only be created if errors (`LOG_ONERRORONLY` option) or warnings and errors (`LOG_ONERROR` option) occur during the backup or restore process.

For more information, see The BACKUP command and The RESTORE command.

SQL Server backup and restore history

Select **Delete old backup history** to limit the amount of history stored in the *msdb* database and the SQL Server Compact database (created when the SQL Backup server components are installed). This includes backup and restore activities that were not performed using SQL Backup.

If the *msdb* database and Compact database contain a large amount of history, it may take some time to reduce the existing history when the setting is first applied. This setting also determines the amount of data stored in the user interface activity cache; the cache is populated with data from the *msdb* and Compact databases each time the graphical user interface is started.

> This setting determines the amount of history available when you choose **Select from backup history** on step 1 of the Restore wizard. If the backup history has been deleted, you can select specific backup files by choosing **Browse for backup files to restore** from the drop-down box.

SQL Backup uses the stored procedure *msdb..sp_delete_backuphistory* to delete the history from the *msdb* database when the graphical user interface is running. If executing this stored procedure causes performance problems, you may find it helpful to add indexes to the backup and restore history tables in the *msdb* database. For more information, refer to your SQL Server documentation.

Alternatively, clear this option and run the stored procedure manually at a convenient time or as a scheduled task. For more information, see Deleting backup and restore history manually.

Email settings

SQL Backup can send an email to specified addresses whenever there are errors or warnings, or only if an error occurs, or for any outcome (success or failure).

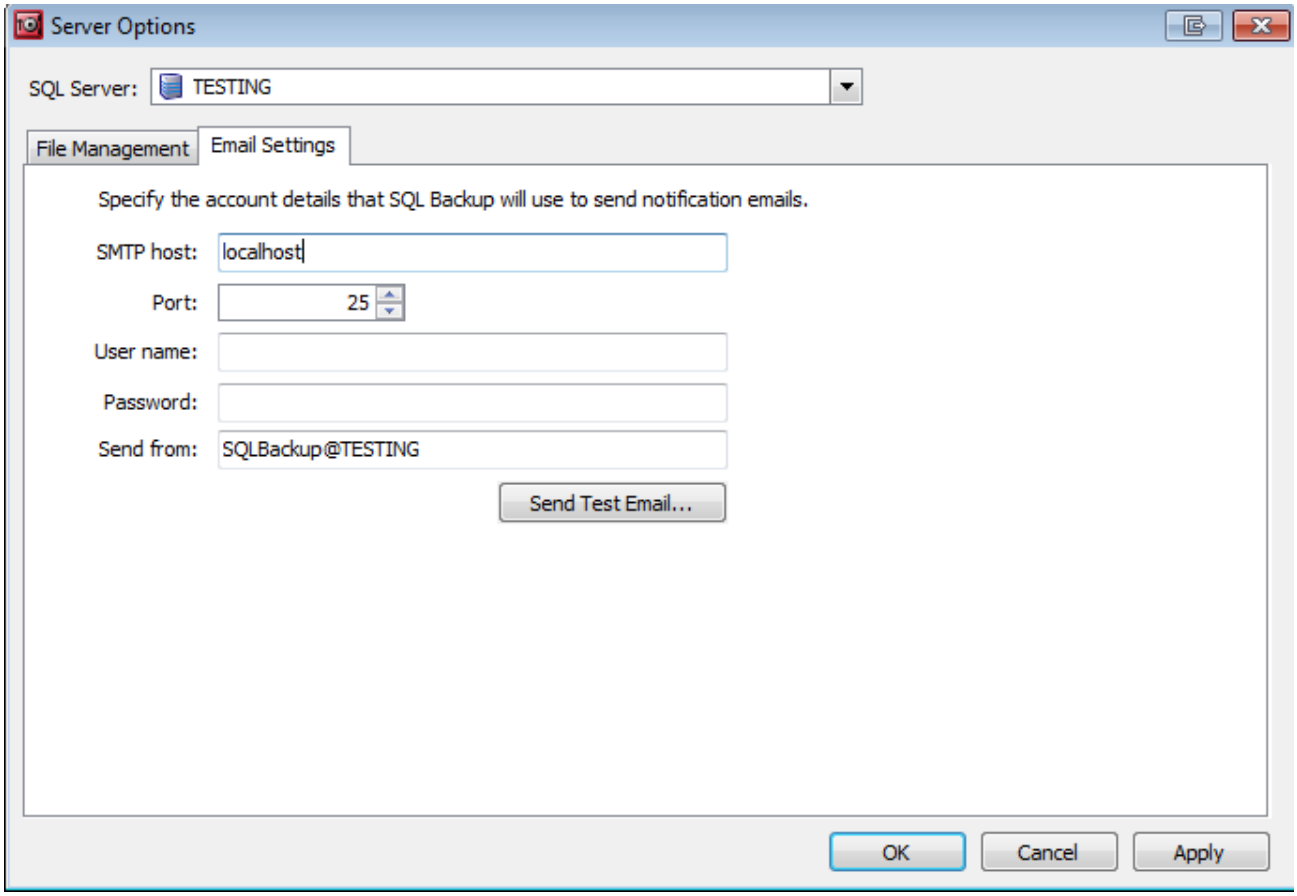
To make this option available, you must enter the email settings to be used. You can then specify the recipients of the email when you set up the backup or restore operation:

- In the graphical user interface, you specify the email addresses in the back up, scheduled backup, restore, or log shipping wizard
- When scripting a SQL Backup job, you use the `BACKUP` and `RESTORE` command options `MAILTO`, `MAILTO_ONERROR`, and `MAILTO_ONERRORONLY`. For more information see The BACKUP command and The RESTORE command.

In order to send the notification email, the server on which SQL Backup is installed must be set up to relay email to your mail server.

To edit the email notification options for a SQL Server:

1. Select the SQL Server on the **Registered SQL Servers** pane.
2. On the **Tools** menu, select **Server Options**.
3. In the **Server Options** dialog box, select the **Email Settings** tab.



4. In the **SMTP host** box, type the name of the SMTP server that SQL Backup is to use when sending emails, and in the **Port** box, type or select the number of the port on the SMTP server (defaults to 25).
5. If your server requires authentication, type the user name in **User name**, then type your password in the **Password** box. The account must have permission to send email.
6. In the **Send from** box, type the email address from whom the email will be sent.
   If you leave this blank, SQL Backup will use the default email address for the sender: *SQLBackup@ComputerName*.
7. To send a test email to ensure your settings are correct, click **Send Test Email**, and enter an email address for the recipient. When you click **OK**, the test email is sent.

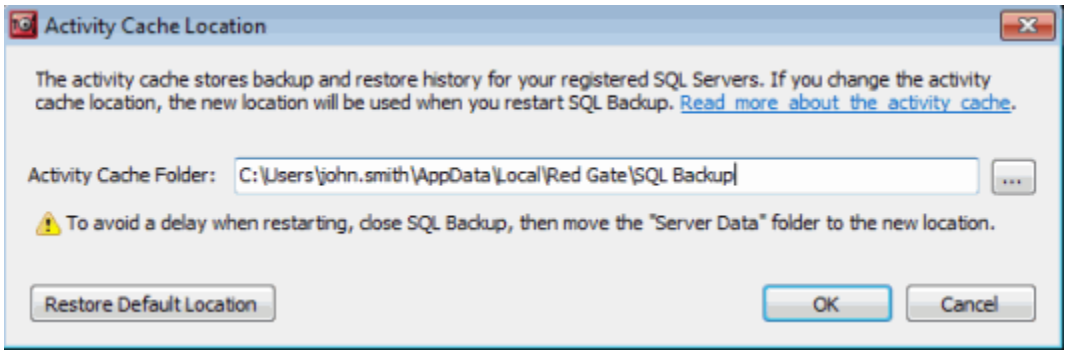
Activity cache location

The activity cache stores backup and restore history for your registered SQL Servers. The default cache location is:

- *%LOCALAPPDATA%\Red Gate\SQL Backup\Server Data* (on Windows Vista, Windows 2008 and later) or
- *%USERPROFILE%\Local Settings\Application Data\Red Gate\SQL Backup\Server Data* (on Windows XP and Windows 2003).

As the cache increases in size, this can cause problems with roaming profiles and disk quotas. You may be able to resolve these problems by changing the cache location (for example, to a network share that does not form part of your roaming profile, or to a disk on which you have a larger storage quota).

To change the location of the activity cache, on the **Tools** menu select **Activity Cache Location**. The following dialog is displayed:



When you click **OK**, SQL Backup remembers the new location for the activity cache, but will not start using this location until you restart the SQL Backup graphical user interface (GUI).

To avoid a delay when restarting, you should manually copy the contents of the *Server Data* folder to the new location.

Reducing the amount of data in the activity cache

If you cannot find a suitable location for the activity cache, you can reduce the amount of data it contains. To do this:

- When you add the SQL Server instance to SQL Backup, limit the amount of SQL Server history imported using the **Native backup and restore history to import** option. For more information, see Adding SQL Server instances.
- From the **Tools** menu select **Server Options**, and on the **File Management** tab use the **Delete all backup and restore history** option to limit the backup and restore history stored in the *msdb* database and SQL Server Compact database. The activity cache is populated from these locations each time you start the GUI. For more information, see SQL Server backup and restore history.

File location tags

SQL Backup enables you to set up file locations that are automatically generated when the backup is created, by using *tags*. For example, you can use tags to include the date and time of creation of a backup in its file name:

C:\Backups\<DATETIME yyyymmdd_hhnnss>_Backup.sqb

Tags are particularly useful when you are backing up multiple databases in one operation. For example, you could specify that the backup files for each database are to be created in a separate folder using the <DATABASE> tag in the path:

C:\Backups\<DATABASE>\Backup.sqb

You can use the following tags in paths and file names:

<TYPE>	The backup type. SQL Backup will use the following values: • FULL for full backups • DIFF for differential backups • LOG for transaction log backups • FILE for filegroup or file backups
<SERVER>	The SQL Server name.
<INSTANCE>	The SQL Server instance name.
<DATABASE>	The database name.
<DATETIME *x*>	The date and time value of the backup process, where *x* is a format string. The values that you can use in the format string are listed below, under **Date and time format**. Note that you can use only one *DATETIME* tag in the default name.
<AUTO>	If you specify <AUTO>, SQL Backup uses the backup file location options (see File management options) to generate the backup file path and file name. If no backup location options have been set up, SQL Backup uses the SQL Server instance's default backup folder, and the default format for file names. If you specify a path and <AUTO> (for example *C:\MyBackups\<AUTO>*), SQL Backup uses the specified path, and generates the file name using the Backup Location options. If no backup location options have been set up, SQL Backup uses the default format for file names. If you specify <AUTO> with a file extension (for example *<AUTO>.sqb*, SQL Backup uses the Backup Location options to generate the backup file path and file name. If no backup settings have been set up, SQL Backup uses the SQL Server instance's default backup folder, and the default format for file names. This is useful when you generate split backup files.

If you specify a format that may result in identical file names (for example, by not specifying a <DATETIME> tag), the backups will fail if you have not chosen to overwrite the backup files. For more information about overwriting backup files, see Creating backups: file settings and Scheduling backups: file settings.

Date and time format

You can use the following formats for string *x* in the <DATETIME x> tag.

d	Displays the day as a number without a leading zero (1-31).
dd	Displays the day as a number with a leading zero (01-31).
ddd	Displays the day as an abbreviation (Sun-Sat) using the strings given by the ShortDayNames global variable.
dddd	Displays the day as a full name (Sunday-Saturday) using the strings given by the LongDayNames global variable.
ddddd	Displays the date using the format given by the ShortDateFormat global variable.
dddddd	Displays the date using the format given by the LongDateFormat global variable.
m	Displays the month as a number without a leading zero (1-12). If the m specifier immediately follows an h or hh specifier, the minute rather than the month is displayed.
mm	Displays the month as a number with a leading zero (01-12). If the mm specifier immediately follows an h or hh specifier, the minute rather than the month is displayed.
mmm	Displays the month as an abbreviation (Jan-Dec) using the strings given by the ShortMonthNames global variable.

mmmm	Displays the month as a full name (January-December) using the strings given by the LongMonthNames global variable.
yy	Displays the year as a two-digit number (00-99).
yyyy	Displays the year as a four-digit number (0000-9999).
h	Displays the hour without a leading zero (0-23).
hh	Displays the hour with a leading zero (00-23).
n	Displays the minute without a leading zero (0-59).
nn	Displays the minute with a leading zero (00-59).
s	Displays the second without a leading zero (0-59).
ss	Displays the second with a leading zero (00-59).

Compression levels

SQL Backup offers four compression levels, described below. Generally, the smaller the resulting backup file, the slower the backup process.

Smaller backups save you valuable disk space. For example, if you achieve an average compression rate of 80%, you can store the backup for a 42.5 gigabyte (GB) database on a 8.5 GB DVD-R dual layer disc. Smaller files can also be transferred more quickly over the network, which is particularly useful, for example, when you want to store backups off-site.

To set the compression level:

- Using the graphical user interface, select the required option in the Backup, Schedule Backup Jobs, or Log Shipping wizard, as appropriate.
- When scripting a SQL Backup job, specify the BACKUP command COMPRESSION keyword (defaults to level 1)

The compression level used to create a backup does not noticeably affect the time necessary to restore the backup.

The compression you can achieve depends upon the type of data stored in the database; if the database contains a lot of highly-compressible data, such as text and uncompressed images, you can achieve higher compression. For full backups, you can use the Compression Analyzer to perform a test on the databases to check which compression level will produce the best result for your requirements.

Compression level 4

Compression level 4 uses the LZMA 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 3

Compression level 3 uses the *zlib* compression algorithm.

On average, the backup process is 25% to 30% faster than when compression level 4 is used, and 27% to 35% fewer CPU cycles are used. Backup files are usually 5% to 7% larger.

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.

Compression level 0

If you do not want to compress your backups, specify compression level 0 from the command line or extended stored procedure; in the graphical user interface, clear the **Compress backup** check box in the wizard. For example, you may want to do this if you require only encryption and you do not want to compress your backups.

Compression percentage

SQL Backup calculates the percentage compression of a backup by comparing the size of the SQL Backup backup with the total database size.

For example, if a database comprises a 10 GB data file and a 3 GB transaction log file and SQL Backup generates a full backup of the database to create a backup file that is 3 GB, the compression for this backup is calculated as 77%, [1-(3/13)]x100.

The compression percentage is displayed in the Activity History

Tools and utilities

The following tools and utilities are provided with SQL Backup:

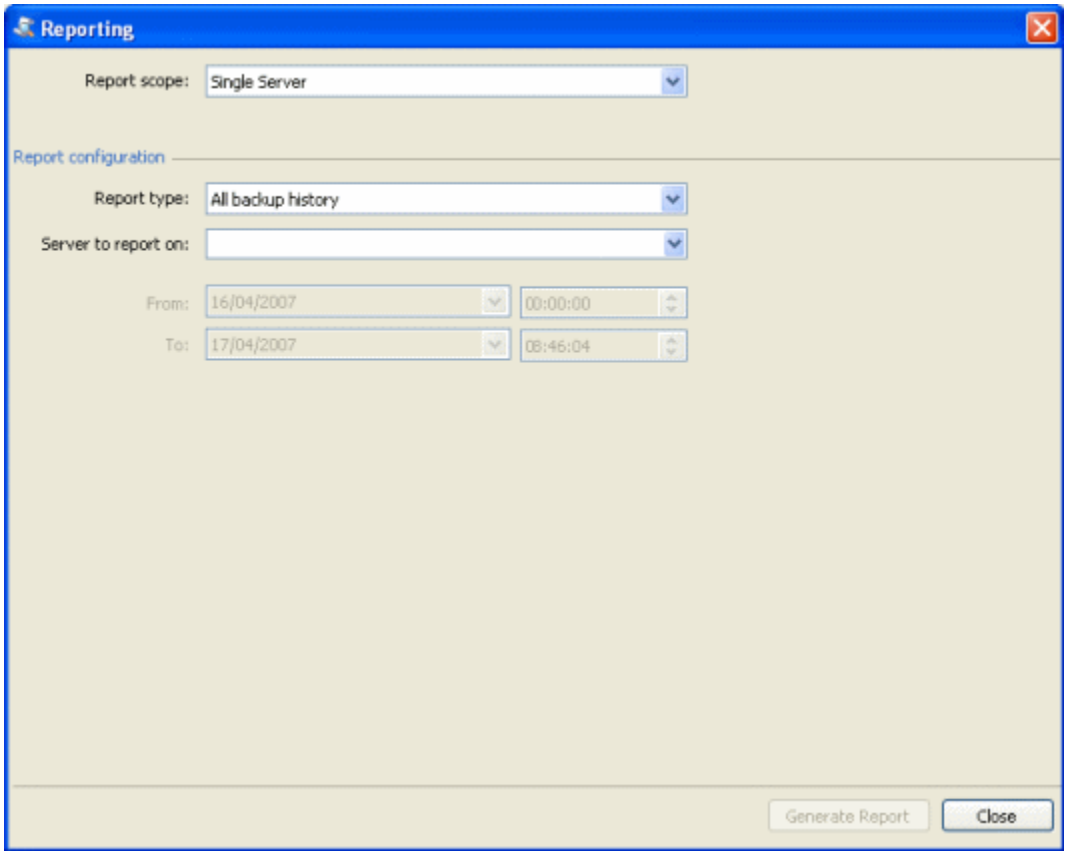
- Reporting
- Compression analyzer
- SQL Backup File Converter
- Maintenance Plan Conversion Wizard

Reporting

SQL Backup provides the **Reporting** feature for you to generate reports on completed backups that have been performed by SQL Backup version 5 or later. You can create the following reports on a selected SQL Server instance or instances, within specified dates:

- **All backup history** lists all backups that have completed.This includes successful backups, backups that have warnings associated with them, and failed backups.
  For reports on multiple SQL Servers, this includes only backups performed by SQL Backup version 5 or later.
- **Backups with warnings** lists all completed backups that have warnings associated with them.
- **Failed backups** lists all completed backups that have failed.This includes only backups performed by SQL Backup version 5 or later.

On the **Tools** menu, click **Reporting**.

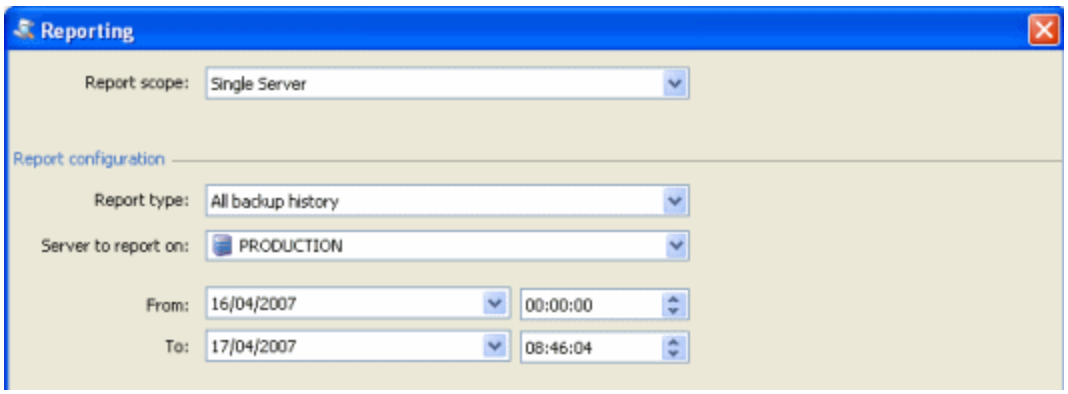


In **Report scope**, choose:

- **Single Server** to generate a report on activities that have occurred on one SQL Server.
- **Multiple Servers** to generate a combined report on a number of SQL Servers. To report on multiple SQL Servers, SQL Backup creates a *reports* database on a SQL Server that you specify.

Single server

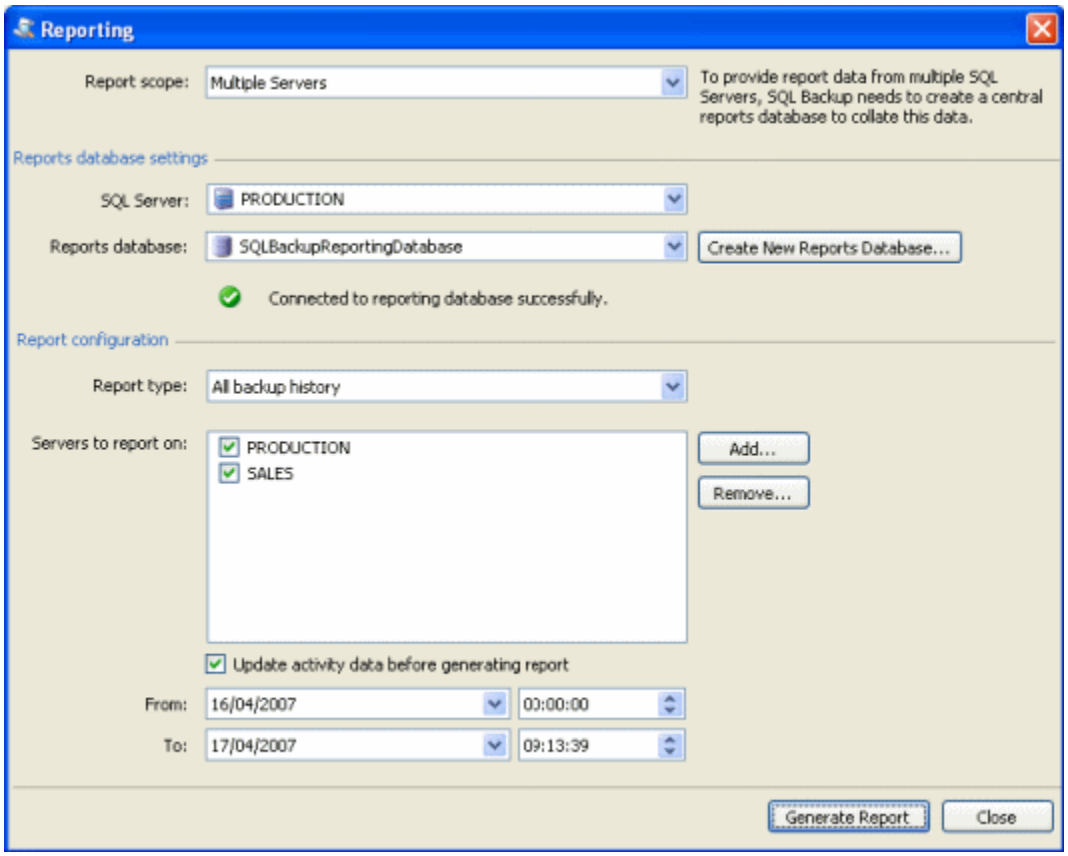
For **Single Server**, select the type of report that you want to generate, and select the SQL Server instance for which you want to create the report. Then, specify the dates (inclusive) between which you want the activities to be reported.



Click **Generate Report**. The report is displayed in the **Report Viewer**. You can print or save the report as required.

Multiple servers

For **Multiple Servers**, SQL Backup must create a reports database to collate the data from the SQL Server instances that you choose to report on.



Under **Reports database settings**, select the SQL Server for the reports database. If you have not yet created a reports database, click **Create New Reports Database**, type a name for the database, and click **Create Database**. If you have already created a reports database, you can select it from **Reports database**.

Under **Report configuration**, select the type of report you want to generate.

When you first create a reports database, the selected SQL Server is automatically added to the list of **Servers to report on**. You cannot remove this SQL Server from the list, but you can clear the check box if you do not want to include it in the report. Any "linked" SQL Servers are also listed; you can deselect or remove these if required. For more information about linked servers, refer to your SQL Server documentation.

To add another SQL Server, click **Add**. Then, in the **Add Linked Server** dialog box, type the name of the SQL Server in the **Server** box, and click **Add**.

By default, SQL Backup updates the data collated in the reports database before generating the report. If you do not want the data to be updated, for example if you know it is already up-to-date, clear the **Update activity data before generating report** check box.

Then, specify the dates (inclusive) between which you want the activities to be reported.

Click **Generate Report**. The report is displayed in the **Report Viewer**. You can print or save the report as required.

Compression analyzer

You can use the SQL Backup Compression Analyzer to compare the effectiveness of different compression levels for full database backups on a selected database.

From within the SQL Backup graphical user interface (GUI), you can launch the Compression Analyzer:

- from Step 4 of the Back Up wizard
- from Step 5 of the Schedule Backup Jobs wizard
- from the **Tools** > **Utilities** menu

SQL Backup automatically selects the SQL Server that you have already chosen in the Registered SQL Servers pane or in the wizard.

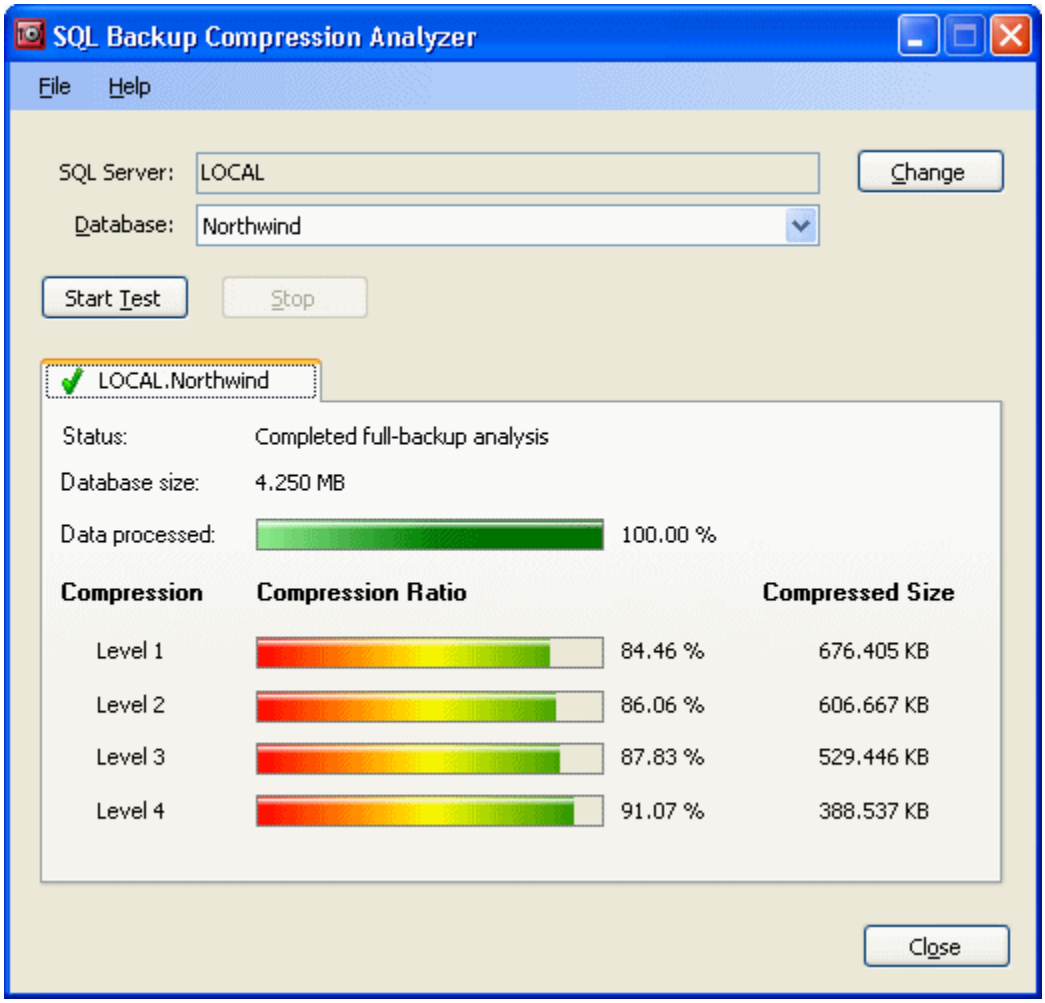
> If you are not using the GUI, you can launch the Compression Analyzer by running *CompressionAnalyzer.exe.* This is located in the SQL Backup installation folder (by default, *%ProgramFiles%\Red Gate\SQL Backup 7* on 32-bit machines and *%ProgramFiles(x86)%\Red Gate\SQL Backup 7* on 64-bit machines). You can run the program on any server on which you have installed the SQL Backup server components.

In the **Connect to SQL Server** dialog box, type or select the name of the SQL Server. Then select the authentication method, and for SQL Server authentication enter the **User name** and **Password**. For Windows authentication, the credentials of the current user are used.

The **Compression Analyzer** is displayed. Select the database for which you want to compare compression levels, and click **Start Test**.

If you want stop the analysis, for example because the database you have chosen is large and the analysis is taking too long, click **Stop**. However, note that results are based only on the data that has been sampled so far; for a fully accurate result you must allow the analysis to finish.

The results of the analysis are displayed in a tab.



If you stopped the analysis before its completion, the **Data processed** bar indicates the proportion of data that was analyzed.

For each compression level, the percentage of compression is shown under **Compression Ratio**. In the above example, Level 4 produced the

best compression. For more information about compression levels and compression ratios, see Compression levels.

You can then select another database to analyze, if required. The results for the second database are shown in a separate tab, so that you can compare the results. To delete a tab, right-click the tab header and click **Delete**.

To analyze a database from a different SQL Server, click **Change** and specify the details of the SQL Server.

SQL Backup File Converter

Use the SQL Backup File Converter (available as a command line application, or as a GUI application) to convert SQL Backup files (.sqb) to Microsoft Tape Format (MTF) files (.bak). You can use the native SQL Server RESTORE command to restore MTF files.

You can also use the SQL Backup File Converter to convert SQL Backup 6 files to SQL Backup 5 files; both file types have extension .sqb.

You can distribute the SQL Backup File Converter as required; you do not need to install SQL Backup in order to use it.

Using the SQL Backup file conversion utility (command line)

The executable for the SQL Backup conversion command-line utility is SQBConverter.exe. It is located in the SQL Backup installation folder (by default, %*ProgramFiles%**Red Gate**SQL Backup 6* on 32-bit machines and %*ProgramFiles(x86)%**Red Gate**SQL Backup 6* on 64-bit machines).

To convert a split backup to MTF format, use *SQBConverter.exe* to convert each of the backup files in turn. You can then restore the MTF format files as a group of split backup files.

> If the original SQL Backup file (.sqb) was created with multiple threads (`THREADCOUNT` > 1), SQBConverter.exe will generate a Microsoft Tape Format file (.bak) for each thread.

Syntax

To convert a SQL Backup file, use a command in the following format: `SQBConverter "inputfile" "outputfile" "password" /sqb`

Arguments	
inputfile	The file path, name and extension of the SQL Backup backup file that you want to convert.
outputfile	The file path, name and extension for the converted file.
password	The password for the SQL Backup backup file that you are converting, if the file is encrypted.
Switches	
/sqb	Convert from a SQL Backup 6 or 7 compatible file to a version 5 compatible file. Note that version 5, 6 and 7 SQL Backup files all have a .sqb extension.

Examples

The following command converts an encrypted SQL Backup 6 file to a Microsoft Tape Format file:

```
SQBConverter "C:\Backups\pubs\FULL_31012012_120000.sqb"
"C:\Backups_MTF\pubs\FULL_31012012_120000.bak" "MyPassword"
```

The following command converts a SQL Backup 6 file to a SQL Backup 5 file:

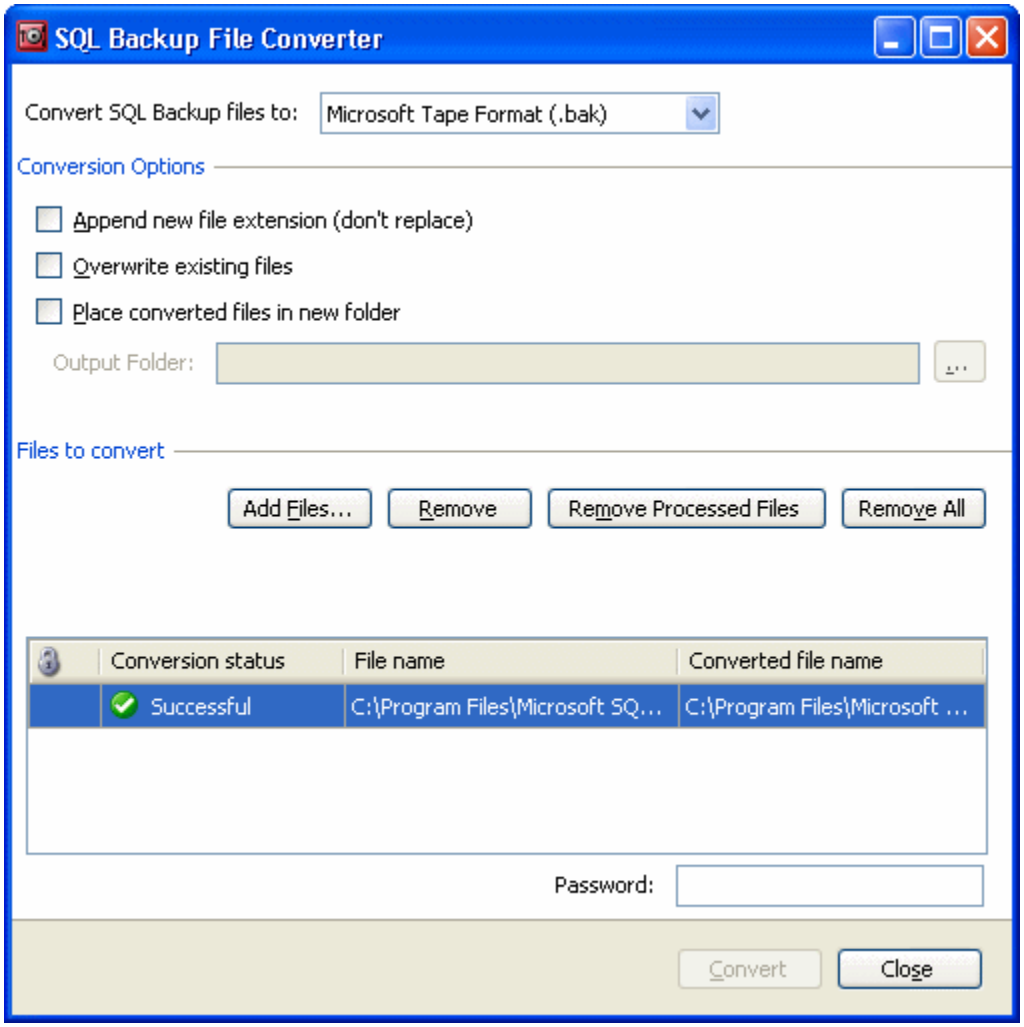
```
SQBConverter "C:\Backups\pubs\FULL_31012012_120000.sqb"
"C:\Backups_MTF\pubs\FULL_31012012_120000.sqb" /sqb
```

Using the SQL Backup file conversion utility (graphical user interface)

SQBConverterGUI.exe is a graphical user interface for the SQL Backup File Converter. It is located in the SQL Backup installation folder (by default, %*ProgramFiles%**Red Gate**SQL Backup 6* on 32-bit machines and %*ProgramFiles(x86)%**Red Gate**SQL Backup 6* on 64-bit machines).

If you want to distribute this utility, you must copy *three* files into a single directory: SQBConverterGUI.exe, RedGate.Shared.Controls.dll, and RgSqbConvHelper.dll.

You can start the user interface from the SQL Backup graphical user interface (**Tools** > **Utilities** > **SQL Backup File Converter**).



To convert SQL Backup files to SQL Backup version 5 (.sqb) or Microsoft Tape Format (.bak):

1. From the **Convert SQL Backup files to** list, select the file format that you want to convert to:
   - Choose **Microsoft Tape Format (.bak)** to convert version 5 or 6 SQL Backup files (.sqb) to native Microsoft Tape Format files (.bak).
   - Choose **SQL Backup 5 (.sqb)** to convert version 6 SQL Backup files (.sqb) to version 5 SQL Backup files (.sqb).
2. Select the required **Conversion Options** to control how the converted files are named and saved.
3. Click **Add Files** and browse for the files that you want to convert.
   Valid SQL Backup files are shown with a **Conversion status** of
   
   Pending. If there is a problem with a file, it is shown with a **Conversion status** of
   
   Error. Click the file to view more information about the problem.
   Encrypted files are marked with a padlock icon
   
   . If you have added encrypted files, you must also enter the password in the **Password** box beneath the file list. The same password will be used to decrypt every encrypted file.
4. Click **Convert**. Each file in the list is converted and saved according to the conversion options you have selected.
   Successfully converted files are shown with a **Conversion status** of
   
   Successful. Files that could not be converted are shown with a **Conversion status** of
   
   Error. Click the file to view more information about the problem.

> If the original SQL Backup file (.sqb) was created with multiple threads (`THREADCOUNT` > 1), SQBConverter.exe will generate a Microsoft Tape Format file (.bak) for each thread.

Maintenance Plan Conversion Wizard

SQL Backup provides a wizard to guide you through the process of converting maintenance plans created with SQL Server Management Studio or SQL Server Enterprise Manager so that they use SQL Backup features such as compression and encryption for backup tasks.

The wizard converts maintenance plans generated by SQL Server Management Studio or SQL Server Enterprise Manager, including legacy maintenance plans upgraded from SQL Server 2000 to SQL Server 2005 or SQL Server 2008.

> The Maintenance Plan Conversion Wizard does not support features introduced by SQL Backup 6 or later, including compression level 4, optimization and reminders.

To launch the Maintenance Plan Conversion Wizard:

- From the SQL Backup graphical user interface, select **Tools** > **Utilities** > **Maintenance Plan Conversion Wizard**.
- Run *SQBMaintPlanConv.exe.* This is located in the SQL Backup installation folder (by default, *%ProgramFiles%\Red Gate\SQL Backup 7* on 32-bit machines and *%ProgramFiles(x86)%\Red Gate\SQL Backup 7* on 64-bit machines).You can run the program on any server on which you have installed the SQL Backup server components.

The Maintenance Plan Conversion Wizard comprises the following steps:

Step 1: select the SQL Server

Step 2: select the maintenance plans

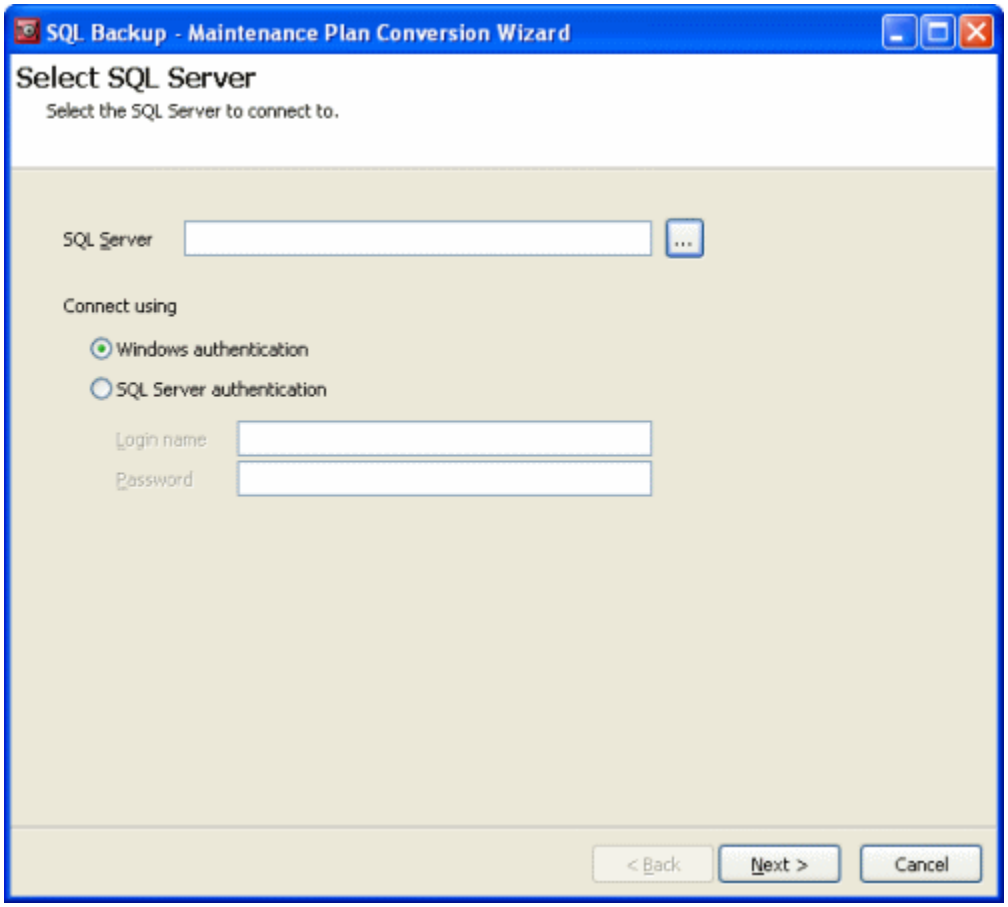
Step 3: configure settings for features provided by SQL Backup

Step 4: review and apply the changes

Step 5: verify the changes

Step 1: select the SQL Server

This page is for selecting the SQL Server instance on which the maintenance plans are located.

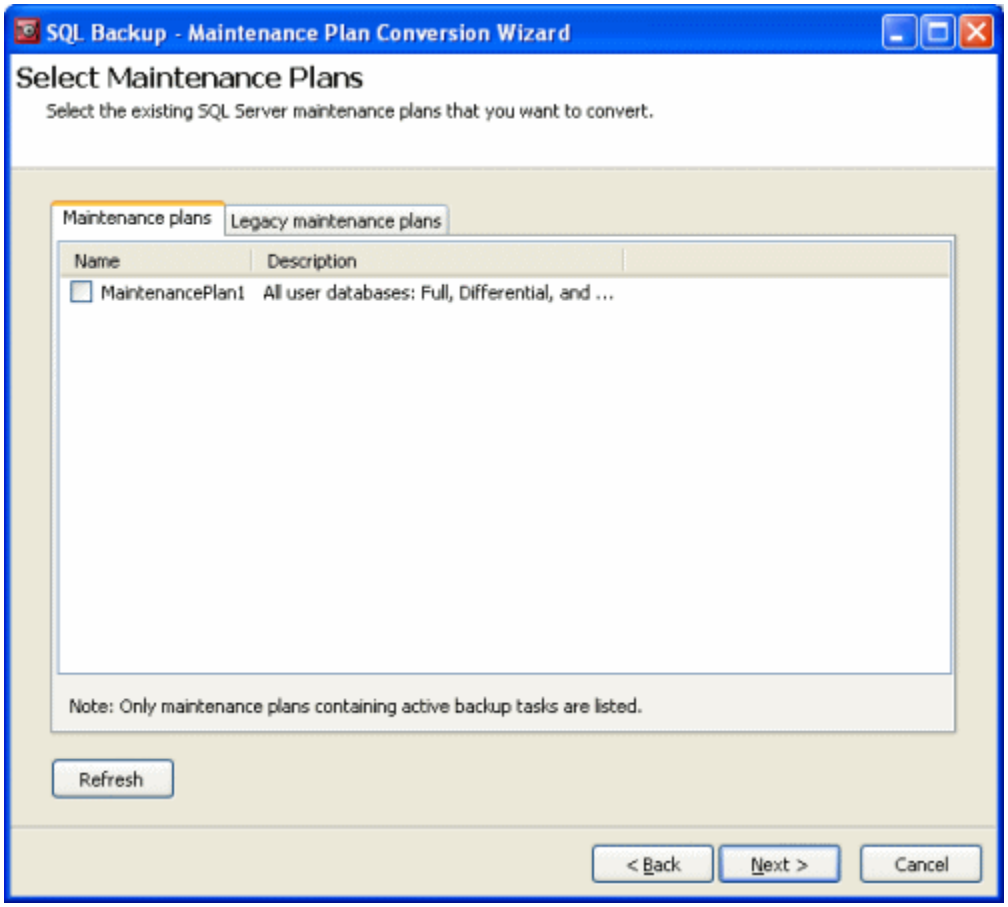


In the **SQL Server** box, type the name of the SQL Server instance, or click to browse to a SQL Server instance on the local network.

Select the authentication method to connect to the SQL Server instance. For **SQL Server authentication**, enter the **Login name** and **Password**.

Step 2: select the maintenance plans

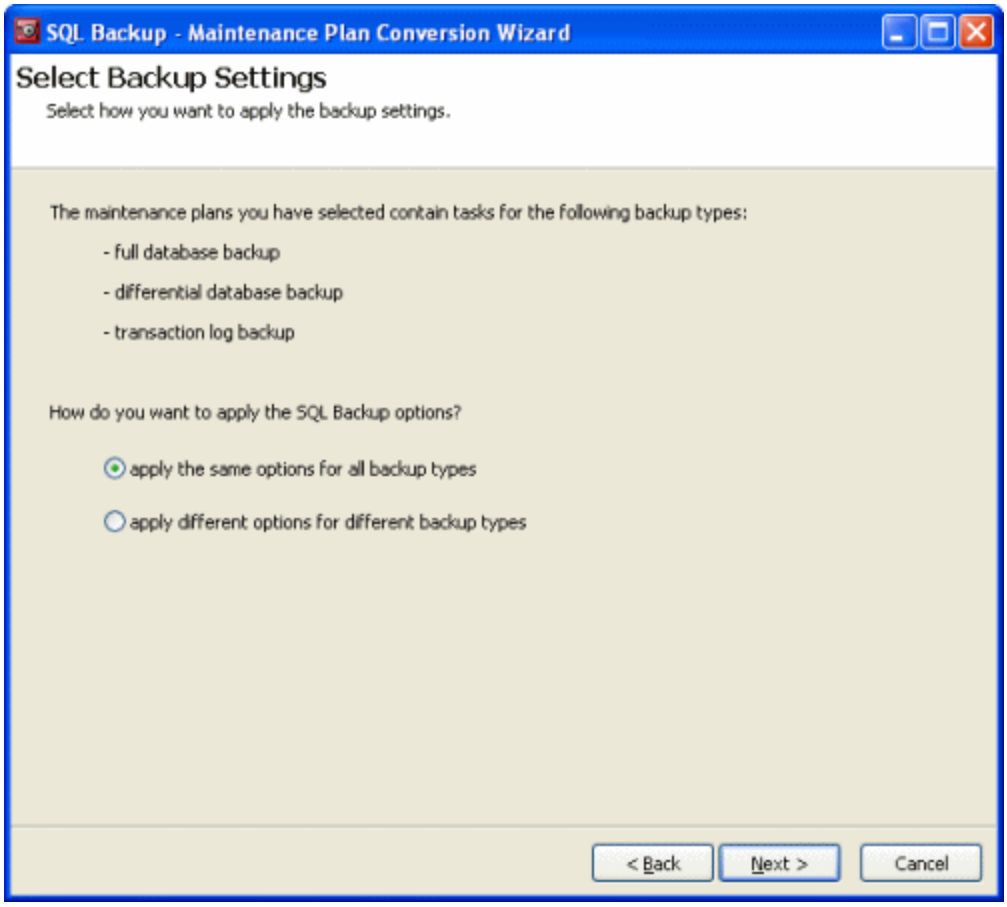
This page is for selecting the maintenance plans to convert.



A maintenance plan is displayed in the list if:

- the associated SQL Server agent jobs are enabled
- there is at least one valid native backup step
    - A step is valid if it contains one or more referenced databases.
    - For SQL Server 2005 and SQL Server 2008, the backup step must be enabled. If a maintenance plan is not displayed in the list, this may because the maintenance plan has already been converted, or because it does not contain any steps to convert.

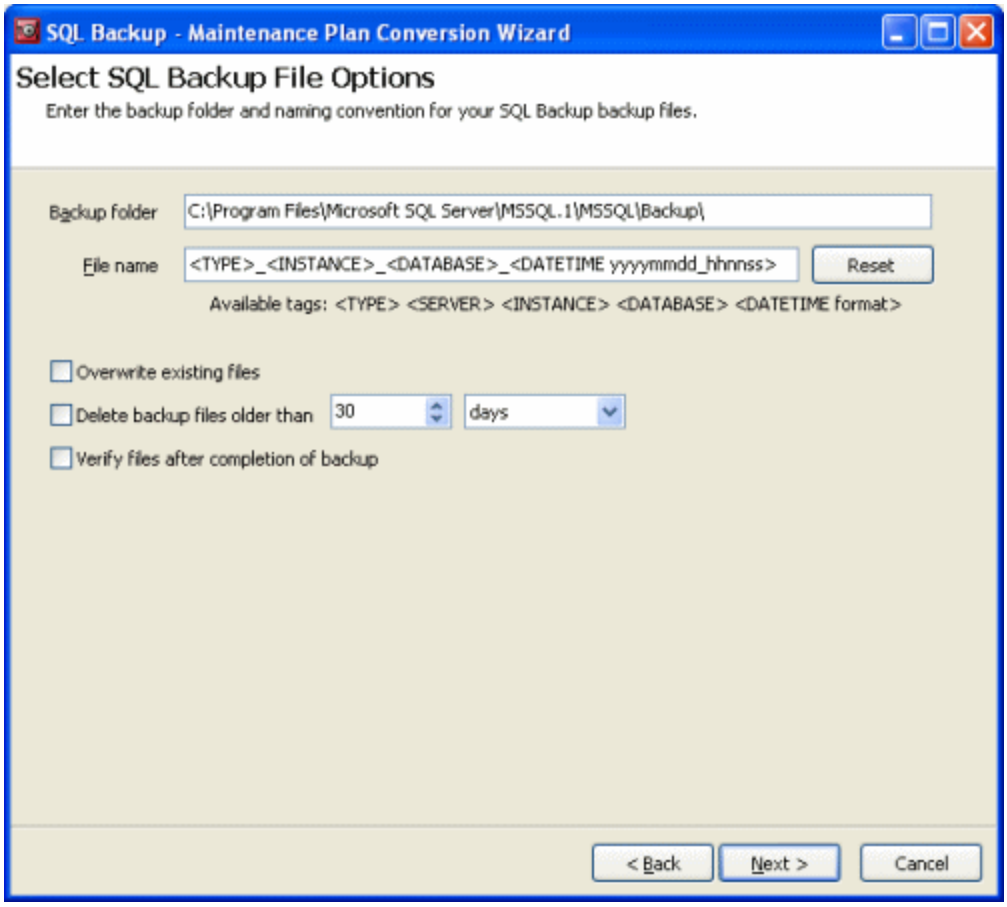
If you select a maintenance plan that has backup tasks for more than one backup type, for example full, differential, and transaction log, when you click **Next** the wizard displays a page asking you whether you want to use the same settings for all backup types or use different settings for each backup type.



If you select **apply different options for different backup types**, the options pages in the following step are repeated for each backup type.

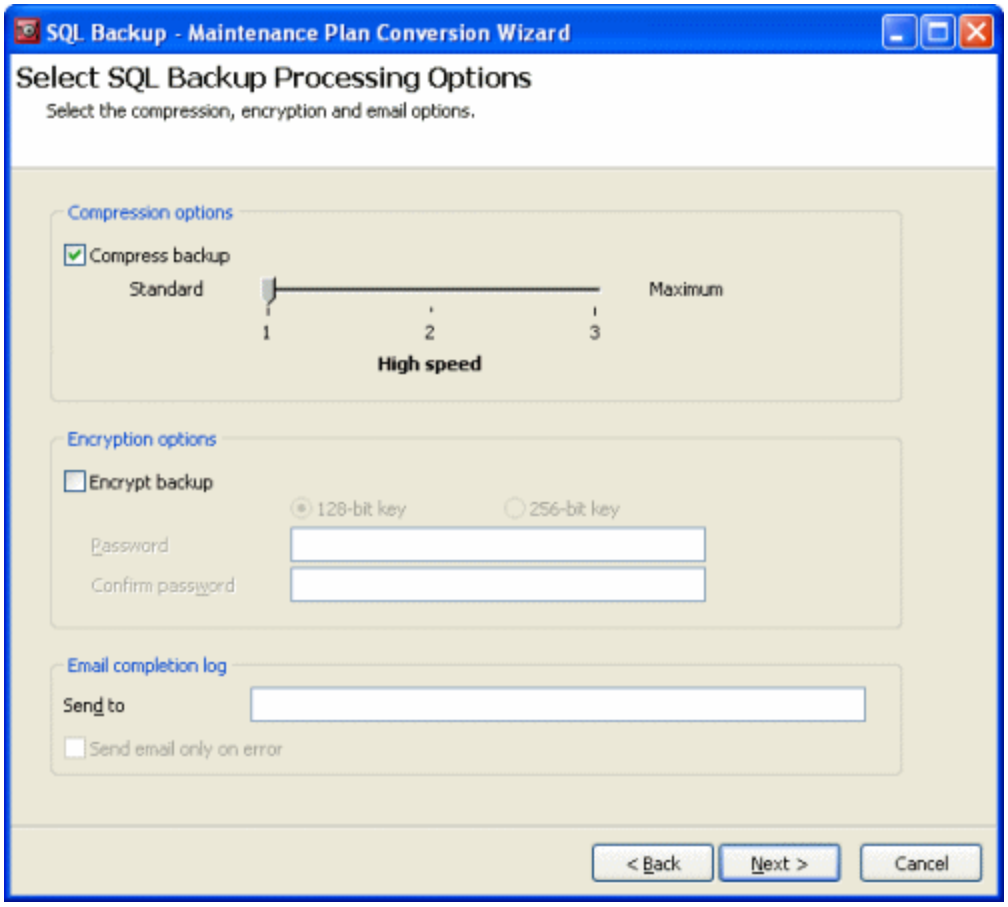
Step 3: configure settings for features provided by SQL Backup

This page is for specifying the locations and file names for backups, and for managing existing backup files.



- **Backup folder** is the name of the folder to back up to. You can optionally use the <tags> available in SQL Backup (see File location tags for more information).
- **File name** is the name of the file to back up to; this requires use of the <tags>. To reset the file name format to its default value, click **Reset**.
- **Overwrite existing files** overwrites backup files of the same name that already exist.
- **Delete backup files older than** deletes any backups of the same type, on the same server and database that are older than the specified time frame.
- **Verify files after completion of backup** performs some additional checks to confirm that the backup has not been damaged or corrupted.

The next page is for specifying compression and encryption settings.



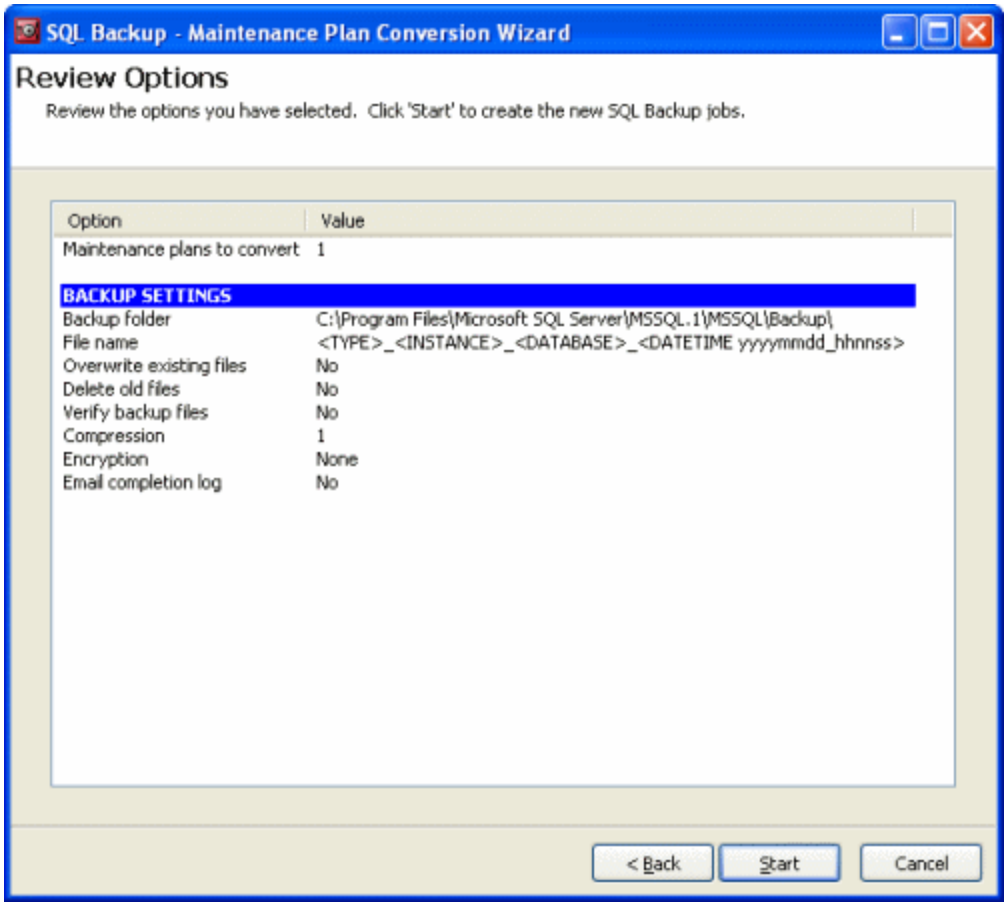
Compress backup uses compression to reduce the size of the backup file. Higher compression levels generally produce smaller files, but may take slightly longer to produce. For more information, see Compression levels.

Encrypt backup uses 128-bit or 256-bit encryption to secure the backup file against unauthorized users. The encryption levels available to you depend on the edition of SQL Backu you are using. Type a password for the backup in **Password**, and again in **Confirm password**.

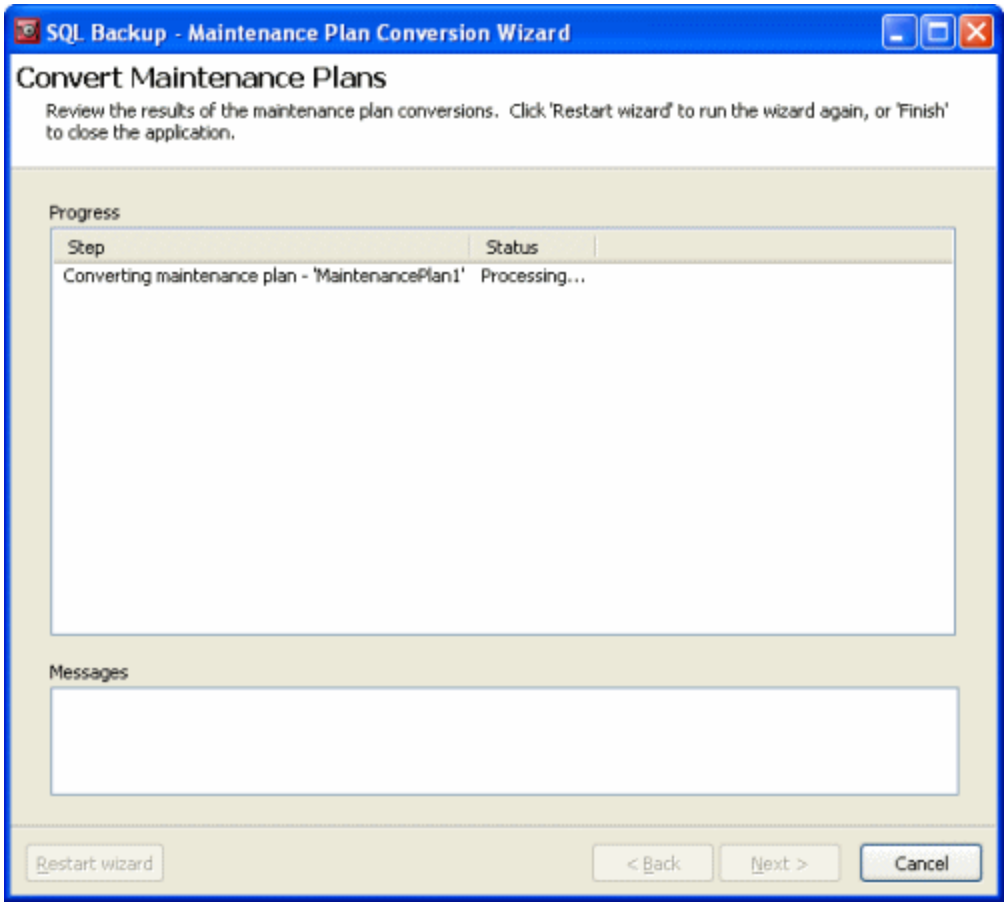
Send to sends an email when the backup is completed (on success or failure). If you select the **Send email only on error** check box, emails are sent only on failure. You must configure the email settings on the **Options** dialog box in the SQL Backup graphical user interface. For more information about configuring email settings, see Email settings.

Step 4: review and apply the changes

This page provides a summary for reviewing your settings.



When you click **Start**, the wizard displays the progress of the conversion, including any issues that occur.



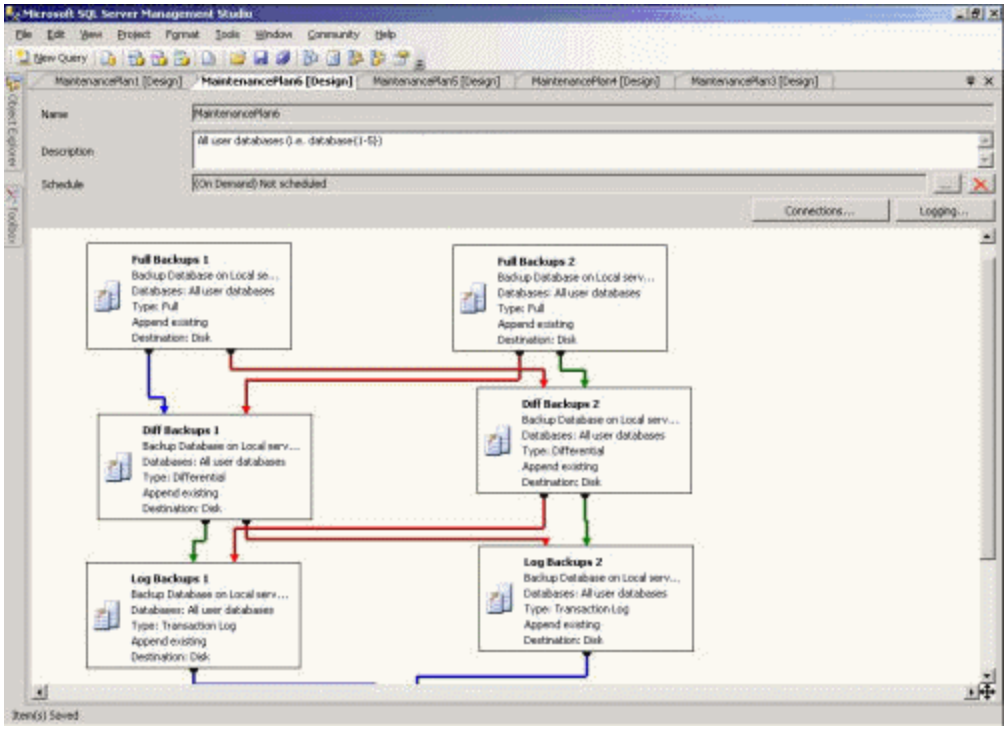
Step 5: verify the changes

When the maintenance plans have been converted, you are recommended to check them in SQL Server Management Studio or SQL Server Enterprise Manager.

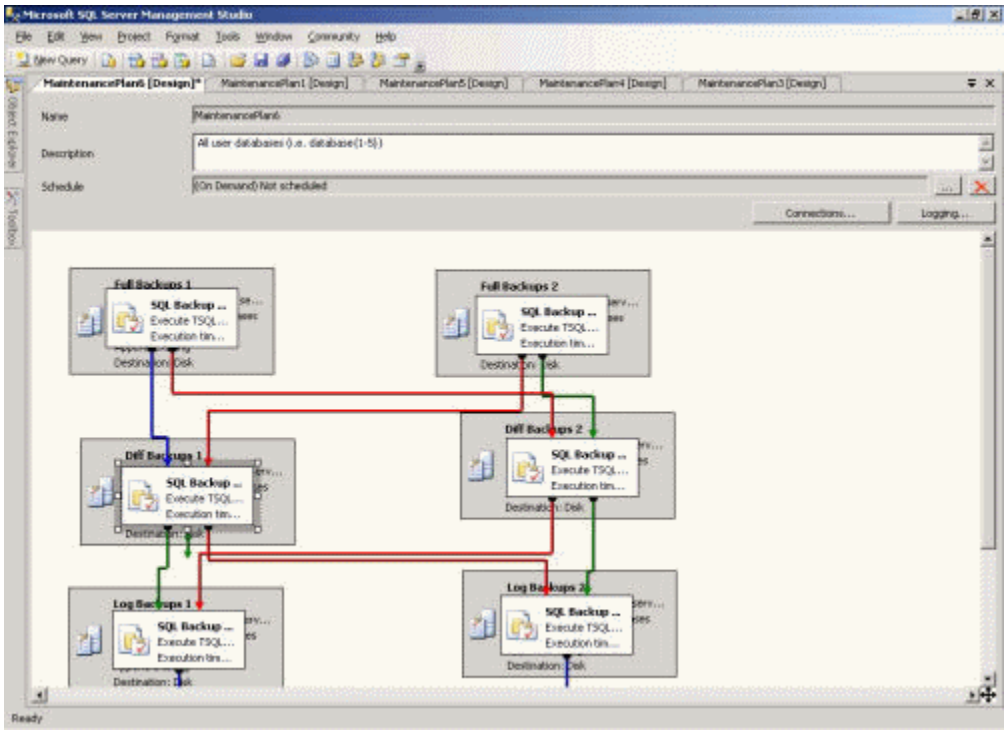
For SQL Server 2000 and legacy maintenance plans, a new maintenance plan and SQL Server Agent job is created with the settings you specified in the Maintenance Plan Conversion Wizard; the SQL Server Agent job in the old maintenance plan is disabled.

For SQL Server 2005 and SQL Server 2008, the existing maintenance plan is modified. Any backup tasks with native SQL Server backups are disabled and replaced with steps that use SQL Backup; any links or relationships between job steps should remain intact.

An example of a maintenance plan before conversion:



An example of a maintenance plan after conversion and reorganization:



Worked examples

- Backing up and restoring on a network share

Backing up and restoring on a network share

This page describes how to back up to and restore from a network share using SQL Backup. You may want to to store backups on a network share, for example, because you do not have enough disk space to store the backups on a local server. You may want to restore a database from a backup stored on another server, for example, in order to verify the backup on a non-production server.

You can use SQL Backup to:

- Back up a database locally and then copy the backup to a network share
- Back up a database directly to a network share
- Restore a database from backups stored in a network share

SQL Backup's network resilience settings can reduce the impact of network outages when backing up or restoring. If the connection is interrupted during a backup or restore, SQL Backup will attempt the operation again after 30 seconds up to 10 times by default. You can adjust these settings on step 4 of the Back Up wizard or step 5 of the Schedule Backup Jobs wizard or with by using the `DISKRETRYCOUNT` and `DISKRETRYINTERVAL` keywords in a BACKUP script run from the command line or extended stored procedure. The network resilience settings used to create the backup are applied when restoring from the backup.

Nevertheless, backing up directly to a network location can be slower than copying the backup after creation and occasionally problematic:

- More resources are required than for a local backup, resulting in increased load on the SQL Server and increased duration for the backup. Transferring compressed data after backing up locally is quicker than writing data across a network.
- By depending on external network resources, backup performance becomes less predictable.

Backing up locally and copying to a network share

It is best practice to back up locally and copy the backup to a network share. To do this, the appropriate permissions must be set up for the network share.

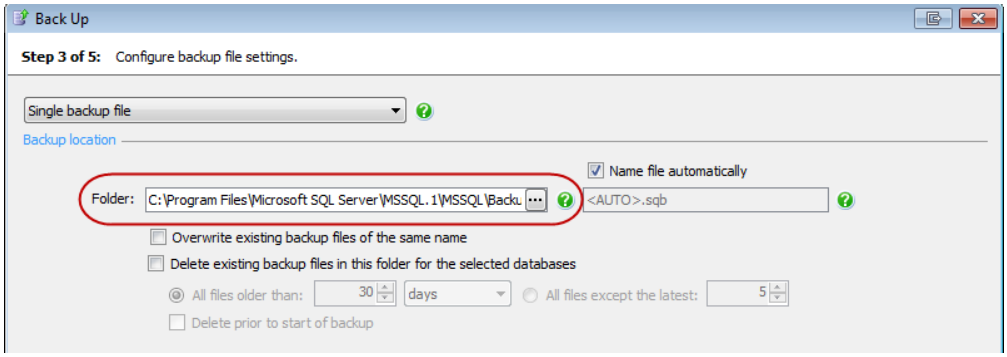
Security and permissions

- The SQL Backup process is controlled by a dedicated SQL Backup Agent application for each SQL Server instance.
- Before you can create a backup and copy it to a network share, you must ensure that the SQL Backup Agent service application permissions are set up appropriately. The user account used to start up the SQL Backup Agent service and connect to the SQL Server must have access to the folder in which the backup is to be created, and it must have BACKUP DATABASE permissions to the SQL Server.
- If you do not want to grant BACKUP DATABASE permissions to the user account used to start up the SQL Backup Agent service, for additional security you can configure a SQL Server authenticated account to connect to the SQL Server. For further information, see Permissions.
- If necessary, you can copy the backup files to a network share that is "locked down". To do this, you will need to set up a different security model. This is also documented in Permissions.

Creating the backup locally

When you have set up the permissions appropriately, use the Back Up wizard or Schedule Backup Jobs wizard to create a backup. On step 3 of the Back Up wizard or step 4 of the Schedule Backup Jobs wizard:

1. Specify a local folder for the backup location.

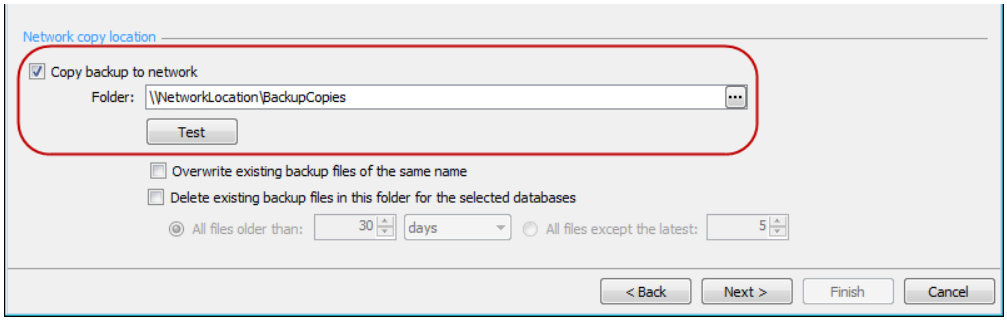


2. Select the **Copy backup to network** check box and specify a location on the network share. Type the full path, including the server name, for example: *\\ServerName\MyFolder\<DATABASE>*. Note that the file path is relative to the selected SQL Server. For example, if you have chosen to back up a 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 server.

   Alternatively, click

to open the Folder Browser. From the **Server** drop-down list, select the network share. If it is not displayed, click **Add Server** and specify the server name or IP address. Other servers will only be visible to the local server if it has the appropriate permissions to write to or read from them. The name of the local server you are connected to and your user name are displayed above the **Server** list. This information may explain why some servers cannot be browsed.

3. Click **Test** to check whether you have the necessary permissions to write to the network share.



4. Proceed through the rest of the wizard as normal.

When the job is run, SQL Backup will create the backup file or files with compression and encryption as specified, run any backup checks as specified, and then copy the files to the network share.

Backing up directly to a network share

To back up directly to a network share, the account used to start up the SQL Backup Agent service application must have full permissions on the network share address (in addition to the other permissions specified in Permissions).

Security and permissions

The following procedure describes how to grant the permissions using Microsoft Windows 7; refer to your Microsoft Windows documentation for full details for your operating system.

1. In Windows Explorer, right-click on the folder in which you want to create the backups, then click **Properties**.
2. On the **Sharing** tab, click **Advanced Sharing**.
3. In the Advanced Sharing dialog, select **Share this folder** then click **Permissions**.
4. In the Permissions dialog, click **Add** to open the **Select Users, Computers or Groups** dialog box.
5. Find or type the startup account for the SQL Backup Agent service and grant it full control.
6. Click **OK** on all of the dialog boxes.

Creating the backup on a network share

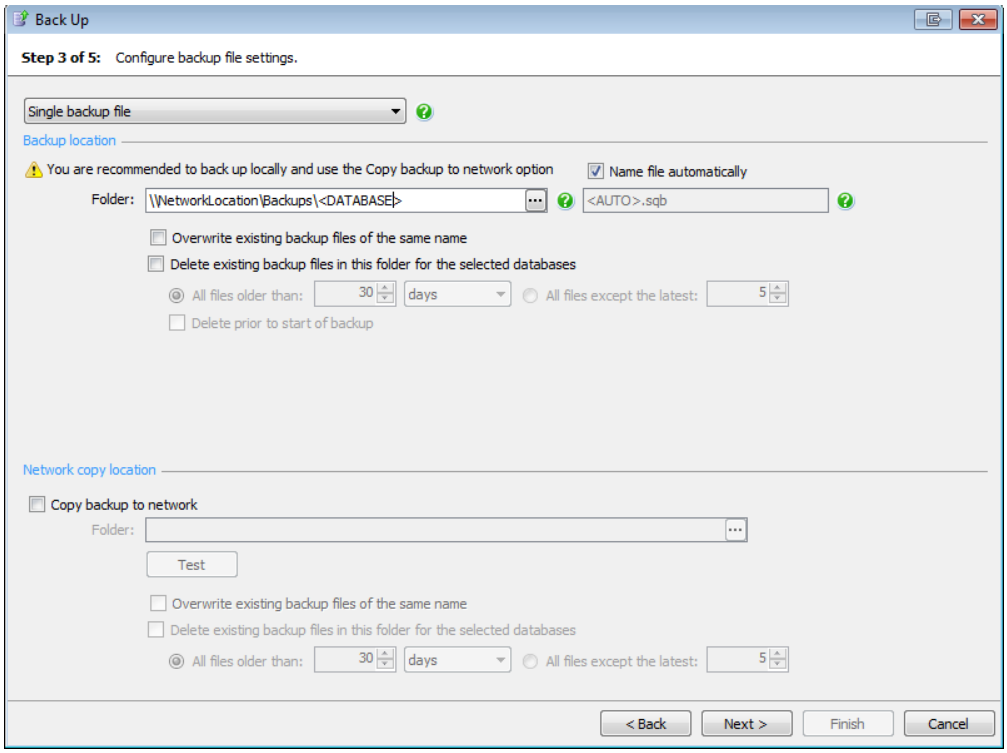
When you have set up the permissions appropriately, use the Back Up wizard or Schedule Backup Jobs wizard to create a backup. On step 3 of the Back Up wizard or step 4 of the Schedule Backup Jobs wizard:

1. Specify a location on the network share as the backup location folder. Type the full path, including the server name, for example: *Server Name**MyFolder**MyFile*. 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 server.

   Alternatively, click

   to open the Folder Browser. From the **Server** drop-down list, select the network share. If it is not displayed, click **Add Server** and specify the server name or IP address. Other servers will only be visible to the local server if it has the appropriate permissions to write to or read from them. The name of the local server you are connected to and your user name are displayed above the **Server** list. This information may explain why some servers cannot be browsed.

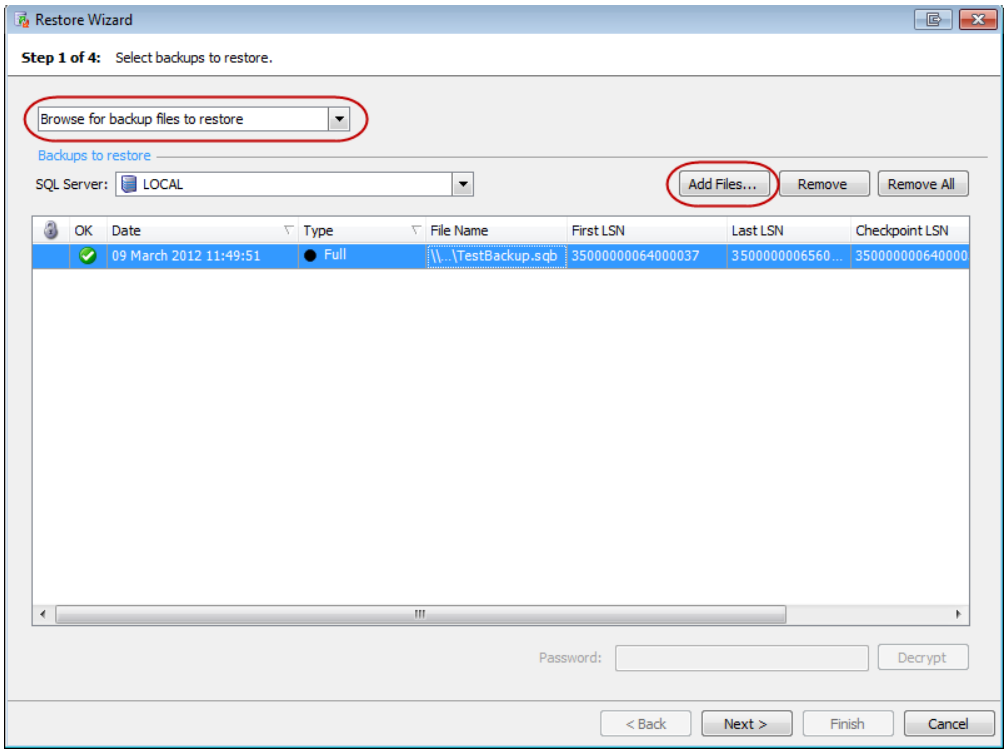


2. Leave the **Copy backup to network box** clear, unless you want to copy the backup files to a different location after they have been created on the network share.
3. Proceed through the rest of the wizard as normal.

Restoring from a network share

You can use SQL Backup to restore a database from a backup file (or set of files) on a different server by either copying the files to a local folder before the restore, or restoring directly from the network location. Note that to restore by creating a new database, the user must have CREATE DATABASE permissions.

1. Select **Browse for backup files to restore** from the drop-down list.
2. Select the SQL Server on which you want to restore the database.
3. Click **Add Files** and browse for the backup files. From the **Server** drop-down list, select the network share. If the network share is not displayed, click **Add Server** and specify the server name or IP address. Other servers will only be visible to the local server if it has the appropriate permissions to write to or read from them. The name of the local server you are connected to and your user name are displayed above the **Server** list. This information may explain why some servers cannot be browsed.



If the backup file you want to restore is incorrectly labelled "Missing", this is because the startup account for the SQL Backup Agent service does not have the correct permissions for the folder or network location it is stored in. To resolve this, grant the SQL Backup Agent service startup account read/write access to the folder or network location.

Continue through the Restore wizard as normal.

Errors and warnings

If a problem occurs during a SQL Backup backup or restore operation, one or more SQL Backup errors or warnings may be returned in addition to any SQL Server errors and Windows errors:

- If you are using the SQL Backup wizards, any errors or warnings are listed in the message dialog that is displayed when you finish the wizard.
- If you are using the extended stored procedure, any errors or warnings are returned in the second dataset by default. For information on changing the format of the output or how to retrieve error codes, see Feedback from the extended stored procedure.
- If you are using the command line, any errors or warnings are returned at the command prompt.

You can review the outcome of previous backup and restore operations in the Activity History in the SQL Backup graphical user interface. To view any errors or warnings for a backup or restore:

- Double click the row to open the Properties dialog.
- Click **Show Log** to view the Activity Log for the backup or restore.

For more information, see The Activity History.

You can also view any errors or warnings in the backup or restore log file. By default, these are created in:

- *%PROGRAMDATA%\Red Gate\SQL Backup\Log\<instance>* (on Windows Vista, Windows 2008 and later), or
- *%ALLUSERSPROFILE%\ Application Data\Red Gate\SQL Backup\Log\<instance>* (on Windows XP and Windows 2003).

For more information about log files, see SQL Backup log files.

For information on specific errors, see:

- SQL Backup warnings 1 - 499
- SQL Backup errors 500 - 5292
- SQL Server errors
- Other errors

SQL Backup warnings 1 - 499

The following warning codes may be returned by SQL Backup.

Warning code	Description
107	**File search pattern not provided. Using default search pattern of '*.sqb' for folder: <value>** When restoring from a backup or backup set, you can specify a file search pattern to identify the most recent backup in the folder. For example: `RESTORE DATABASE [AdventureWorks] FROM DISK = 'C:\Backups\*AdventureWorks*.sqb' LATEST_ALL` If no file search pattern is specified, the default pattern, *.sqb, is used. For more information, see The RESTORE command.
110	**Failed to save primary log file: <value>** The primary log file records details of backup and restore operations. Check that the startup account for the SQL Backup Agent service has rights to create files in the reported directory. Also check that the directory has sufficient free space. By default, the log files will be stored in the following folder on the machine on which the SQL Backup server components are installed: • *%PROGRAMDATA%\Red Gate\SQL Backup\Log\<instance>* on Windows Vista, Windows 2008 and later, or • *%ALLUSERSPROFILE%\ Application Data\Red Gate\SQL Backup\Log\<instance>* on Windows XP and Windows 2003. You can change the location of the log file using the file management options. For more information, see SQL Backup log files.
115	**LOGTO error: Failed to save log file: <value>** This log file, specified by the `LOGTO` option, records details of backup and restore operations. Check that the account the SQL Backup Agent service logs on as (the startup account) has rights to create files in the listed directory. Also check that the directory has sufficient free space. For more information about the `LOGTO` option, see LOGTO in The BACKUP command.
120	**LOGTO error: Folder does not exist: <value>** The LOGTO option specifies a log file for recording details of backup and restore operations. SQL Backup couldn't create the directory for this log file. Check that the account the SQL Backup Agent service logs on as (the startup account) has rights to create the directory. For more information about the LOGTO option, see LOGTO in The BACKUP command.
121	**LOGTO error: Default log file folder does not exist: <value>. Log file will be saved in folder: <value>.** The LOGTO option specifies a log file for recording details of backup and restore operations. SQL Backup couldn't create the log file in the default directory, but has successfully created the log file in the reported directory. heck that the account the SQL Backup Agent service logs on as (the startup account) has rights to create files in the default directory. For more information about the `LOGTO` option, see LOGTO in The BACKUP command.
130	**MOVETO error: Failed to move file: <value>** SQL Backup failed to move a file to the specified directory. Check the error message for the cause of the failure. In most cases, this problem occurs when the account the SQL Backup Agent service logs on as (the startup account) does not have appropriate rights. The startup account must be able to write to the specified directory, and delete the source file. If this error arises during log shipping, it is often because the `MOVETO` parameter is not moving files out of the network share. Edit the Log Restore job so that the restored log files are moved to a different folder. You will need to do this via a SQL application such as SQL Server Management Studio. See Configuring log shipping for more information.

131	**MOVETO error: Failed to execute MOVETO command.** SQL Backup failed to move the file as specified by the `MOVETO` command. Check the error message for the cause of the failure.
132	**MOVETO error: Folder does not exist: <value>** SQL Backup failed to move the file as specified by the `MOVETO` command, because the specified directory does not exist.
133	**MOVETO error: Folder does not exist: <value>**
140	**COPYTO error: Destination file already exists. Will not overwrite: <value>** SQL Backup failed to copy the file to the directory specified by the `COPYTO` command. The file already exists in this directory. To force the file to be overwritten, specify a `FILEOPTIONS` parameter that includes the bit-masked value "4". For more information, see FILEOPTIONS in The BACKUP command.
141	**COPYTO error: Unable to copy <source> to <target>** SQL Backup failed to copy the file to the target directory specified by the `COPYTO` command. In most cases, this problem occurs when the account the SQL Backup Agent service logs on as (the startup account) does not have appropriate rights. The startup account must be able to write to the target directory. For more information on permissions required by the SQL Backup Agent service, see Permissions.
142	**COPYTO error: Unable to copy <source> to <target>. Error message: <value>** SQL Backup failed to copy the file to the target directory specified by the `COPYTO` command. Check the error message for the cause of the failure.
143	**COPYTO error: Failed to copy file: <value>** SQL Backup could not process the files to be copied. Check the error message for the cause of the failure.
144	**COPYTO error: Folder does not exist: <value>**
145	**COPYTO error: Failed to create folder: <value>**
146	**COPYTO error: Source file does not exist: <value>**
147	**COPYTO made multiple attempts to copy the file: <value>** SQL Backup made multiple attempts to copy the file specified by the `COPYTO` command. The error message shows the number of copy attempts, and the reason why each copy attempt failed.
148	**COPYTO error: failed to delete temporary file: <value>**
149	**COPYTO error: failed to rename temp file <source> to <target>**
150	**MAILTO error: SMTP host name not defined.** SQL Backup could not send an email notification because the SMTP host name was not defined. To specify an SMTP host name for this SQL Server instance, open the SQL Backup GUI and from the **Tools** menu select **Server Options**, then open the **Email Settings** tab. For more information, see Email Settings.
151	**MAILTO error: Failed to send mail.** SQL Backup failed to send an email notification. Check the error message for the cause of the failure.
160	**ERASEFILES error: Failed to delete file: <value>** SQL Backup could not delete the file that was marked for deletion.
161	**ERASEFILES error: Failed to delete file: <value>. Error code: <value>** SQL Backup could not delete the file that was marked for deletion. Check the error message for the cause of the failure.
162	**Failed to delete log file: <value>. Error code: <value>** The log file records details of backup and restore operations. SQL Backup could not delete the log file that was marked for deletion. Check the error message for the cause of the failure.

164	**Failed to delete backup entries in msdb tables: <value>** SQL Backup could not delete old backup and restore records from the *msdb* history tables. Check the error message for the cause of the failure.
166	**Failed to delete old entries in local history tables: <value>** SQL Backup could not delete backup and restore records from the local SQL Server Compact database. This may be caused by corruption of the SQL Backup local data store. The local data store is a SQL Server Compact Edition database, which is used to cache backup and restore history. By default, it is installed in *%PROGRAMDATA%\Red Gate\SQL Backup\Data\<instance name>\data.sdf* for Windows Server 2008, Windows Server 2008 R2, Windows Vista, and Windows 7, or *%ALLUSERSPROFILE%\Application Data\Red Gate\SQL Backup\Data\<instance name>\data.sdf* for Windows Server 2003 and Windows XP. To fix the problem, repair the *data.sdf* file. For more information, see 'To repair a database' in How to: Maintain a Database on MSDN.
167	**Failed to get database size from server.** While calculating the compression ratio for a database, SQL Backup could not retrieve the current database size from SQL Server. Check the error message for the cause of the failure.
168	**Failed to identify filestream files.** While calculating the current database size, SQL Backup could not retrieve the details of files used to store filestream data. Check the error message for the cause of the failure.
170	**Validation error when processing the list of transaction log backups.** SQL Backup could not process the list of transaction log backups required to restore a database. Check the error message for further details. If the error message reports that the log files are not in sequence, this may be because the Log Sequence Numbers cannot be read or are missing from the header file. You can manually restore the log files, specifying `WITH NORECOVERY` or `WITH STANDBY`, using the extended stored procedure or the command line. See The RESTORE command for more information. In a log shipping scenario, this error may be due to a large log backup taking a long time to copy to the network share, allowing later (smaller) log files to be copied to the network share before it, and a restore of these attempted. This problem should sort itself out over time. You can avoid the problem by increasing the log restore job interval so that it is at least the transfer time of the largest backup. For more information, see Log shipping.
175	**Another process might be locking this file: <value> (attempt <value>) (<value>)** This is an informational message. The backup file was locked, preventing SQL Backup from updating the file at the first attempt. SQL Backup succeeded in updating the file after further attempts. Other applications may be accessing backup files. This may cause errors in the future if those applications maintain locks on the backup files.
190	**Failed to update backup history table (BACKUPSET) using backup_set_uuid: <value>**
195	**Failed to update backup history table (BACKUPMEDIAFAMILY).** SQL Backup could not update details of the backup process in the `BACKUPMEDIAFAMILY` table in the *msdb* database. Check the error message for the cause of the failure.
200	**Error creating backup file: <value>** SQL Backup could not create the backup file. Check the error message for the cause of the failure.
202	**Multiple attempts were made to create the backup file.** This is an informational message. SQL Backup could not create the backup file at the first attempt, but succeeded after further attempts. This indicates potential problems with storing backup files at the specified locations.
204	**Multiple attempts were made to restore from the backup file.** This is an informational message. SQL Backup could not restore from the backup file on the first attempt, but succeeded after further attempts.

210	**Error writing to backup file: <value>** This is an informational message. SQL Backup could not write backup data to disk at the first attempt, but succeeded after further attempts. This indicates potential problems with storing backup files at the specified locations.
213	**Error setting file pointer position: <value>** This is an informational message. SQL Backup could not continue to write backup data to disk following a previous failure, but succeeded after further attempts. This indicates potential problems with storing backup files at the specified locations.
215	**Error flushing buffer contents to file: <value>**
218	Insufficient disk space <value>: required <value>; available <value> There was insufficient disk space to create the backup file. Before attempting the backup process again, create more free space on the disk, or back up to a different location.
220	**No log files found to be restored.** SQL Backup could not find any relevant transaction log files to restore from the locations you specified. Check that you have specified the files correctly (including any wildcards in file names). Also check that the account the SQL Backup Agent service logs on as (the startup account) has permission to read the transaction log files.
225	**Files missing from backup history.** One or more of the backup files required for the restore process is missing. Check the error message for details of the missing files.
226	**Files missing from search folder.** One or more of the backup files required for the restore process is missing. Check the error message for details of the missing files.
280	**Failed to open file.**
300	**Backup failed. Retry attempt: <value>** This is an informational message. SQL Backup could not back up the database at the first attempt, but succeeded after further attempts. Check the error message for information about why each attempt failed.
320	**Validation failed.**
330	**Your license does not allow email notification to be performed.** The `MAILTO`, `MAILTO_ONERROR`, or `MAILTO_ONERRORONLY` keyword was specified, but has been ignored due to licensing restrictions. SQL Backup failed to send an email notification because you are using SQL Backup Lite edition, or a read-only licensed edition of SQL Backup. To send email notifications, you must use the Pro edition of SQL Backup. Note that from version 6.5, SQL Backup is only available in the Pro edition. SQL Backup Lite has been retired. If you are currently using SQL Backup Lite and have not yet been contacted by Redgate about a free upgrade, please go to the SQL Backup product page and download it from there. You can use your existing serial key to license the Pro version.
400	**Failed to initialize local data store: <value>** SQL Backup could not initialize the SQL Server Compact database it uses to store details of backup and restore processes. The SQL Server Compact database is created when the SQL Backup server components are installed and is used to improve the performance of the GUI. Check the error message for the cause of the failure. If the error message reports that access to the database file is not allowed, you need to ensure that the account the SQL Backup Agent service logs on as (the startup account) has permission to access it. For more information, see Permissions.

402	**Failed to open local data store: <value>** SQL Backup could not connect to the SQL Server Compact database it uses to store details of backup and restore processes. The SQL Server Compact database is created when the SQL Backup server components are installed. The account the SQL Backup Agent service logs on as (the startup account) must have permission to access the folder containing the SQL Server Compact database files (data.sdf). By default, this is: • *%PROGRAMDATA%\Red Gate\SQL Backup\Data\<instance name>* (Windows Vista, Windows 2008 and later), or • *%ALLUSERSPROFILE%\ Application Data\Red Gate\SQL Backup\Data\<instance name>* (Windows XP and Windows 2003). For more information, see Permissions.
405	**Failed to acquire handle for local data store lock (<value>): <value>**
410	**Failed to start transaction on local data store: <value>**
425	**Folder does not exist: <folder name>.** Check that all the log file locations you have specified for your log shipping restore job are available. If necessary, create the missing directories, or alter your log shipping restore job so that it refers to the correct directories. Also, check that the account the SQL Backup Agent service logs on as (the startup account) has permission to read the directories.
435	**Get exclusive local data store access failed - timeout.** SQL Backup could not acquire a lock on the SQL Server Compact database it uses to store details of backup and restore processes. Another process was accessing the database for more than 15 seconds. The SQL Server Compact database may be too large. Review your backup history deletion settings: 1. Open the SQL Backup GUI and select the SQL Server you were trying to back up or restore to. 2. From the **Tools** menu select **Server Options**. 3. On the **File Management** tab, review the settings under **SQL Server backup and restore history**. Consider reducing the amount of backup history you retain. For more information, see File Management Options.
436	**Get exclusive local data store access failed - abandoned.** SQL Backup could not acquire a lock on the SQL Server Compact database it uses to store details of backup and restore processes. This was due to a termination request. This usually happens because the SQL Backup Agent service is in the process of shutting down. You can restart the SQL Backup Agent service from the Windows Services panel.
438	**Get exclusive local data store access failed - invalid lock handle.** SQL Backup could not acquire a lock on the SQL Server Compact database it uses to store details of backup and restore processes. This was probably caused by an internal error. Try restarting the SQL Backup Agent service. To do this, open the Windows Services panel, select the SQL Backup Agent service for the instance, right-click and select **Restart**.
440	**Invalid DB handle.** SQL Backup could not write data to the SQL Server Compact database it uses to store details of backup and restore processes. The database was not available. Try restarting the SQL Backup Agent service. To do this, open the Windows Services panel, select the SQL Backup Agent service for the instance, right-click and select **Restart**.
442	**Get exclusive local data store access failed - invalid lock handle.** SQL Backup could not release a lock on the SQL Server Compact database it uses to store details of backup and restore processes. This was probably caused by an internal error. Try restarting the SQL Backup Agent service. To do this, open the Windows Services panel, select the SQL Backup Agent service for the instance, right-click and select **Restart**.
445	**Local data store error: <value>** SQL Backup could not write data to the SQL Server Compact database it uses to store details of backup and restore processes. Check the error message for the cause of the failure.

450	**Local data store SQL error: <value>** SQL Backup could not write data to the SQL Server Compact database it uses to store details of backup and restore processes. There was an error preparing the SQL command used to write the data. Check the error message for the cause of the failure.
452	**Failed to store list of files to copy into copy queue.**
460	**Encryption key size is specified, but password is empty.** The `KEYSIZE` keyword has been specified, but the `PASSWORD` keyword has not been specified, or it is specified with a zero-length password.
462	**Short passwords may not be secure.** The `KEYSIZE` keyword has been specified, but the supplied `PASSWORD` is seven characters or fewer.
470	**Transaction log has already been restored.** This is an informational message. One or more transaction log backup files has already been restored, and will not be restored again. This may indicate that the `MOVETO` option is not moving restored transaction log files to the specified directory. Alternatively, another process may be copying old transaction log backup files to the directory SQL Backup is using for newly created transaction log backup files.
471	**Failed to run sp_change_users_login report while checking for orphaned users.** The `ORPHAN_CHECK` option was included in a `RESTORE` command, but SQL Backup could not run the *sp_change_users_login* stored procedure. Check the error message for the cause of the failure.
472	**Orphaned users have been detected during a RESTORE operation. Check the log file for the list of orphaned users and associated SIDs.**
473	**The current log file contains records created after the STOPAT time.**
480	**Failed to create output folder: <value>**
481	**The file name (<file name>) contains an invalid character (<invalid character>).** This warning arises if the <DATABASE> tag is used, either directly or via the <AUTO> tag, to generate a backup file name from the database name, and the database name includes an invalid character. If no backups are created, error code 505 is also returned. For information on using tags to generate file names, see File location tags. For information on the <AUTO> tag see File management options.
485	**File does not exist: <Filename .sqb>** This warning may arise during a restore operation. The restore process will perform a check on the file to ensure it is accessible before the restore begins. If there is a temporary network interruption or a delay in the initial response being received, this warning will occur, but the second attempt to check the file will succeed. The warning can be ignored in these circumstances.

SQL Backup errors 500 - 5292

The following error codes may be returned by SQL Backup.

Error code	Description
500	**General SQL Backup error.**
505	**No valid SQL Backup file name entered.** Check that the length of the path name does not exceed the number of characters for your operating system. If you are generating a backup file name using the <DATABASE> or <AUTO> tags, check that the database name does not contain invalid characters. For more information about tags, see File location tags. If you are backing up directly to a network share (not recommended), ensure the network share already exists, and that the account the SQL Backup Agent service logs on as (the startup account) has full permissions for the network share. To check the SQL Backup Agent service permissions, run the following command: `execute master..sqbutility 999, 'RWE', '\\testserver\backuplocation'` If the SQL Backup Agent service has read (R), write (W), and execute (E) permissions, the command returns 1; 0 indicates failure. For more information about backing up to a network share, see Backing up and restoring on a network share.
506	**No valid backup sets found from backup history.**
507	**No valid backup sets found from provided folder(s).**
508	**No valid full backup sets found from provided folder(s) for latest differential backup.**
510	**Backup file exists. Will not overwrite. File name: <value>** Try specifying a file name format that does not result in identical file names. For more information about using tags to generate file names, see File location tags. To overwrite existing backup files, use the `INIT` argument.
512	**Backup file name (<value>) contains an invalid character (<value>)**
515	**Multi database backup file names must contain the <AUTO>, <DATABASE> or <DATETIME> tags.** If you are backing up multiple databases, the backup file names must be unique. Use the <AUTO>, <DATABASE>, or <DATETIME> tags to achieve this. For more information about using tags to generate file names, see File location tags.
516	**Multi database operations cannot use MIRRORFILE option.** If you are backing up multiple databases, you cannot use the `MIRRORFILE` argument. For more information, see MIRRORFILE in The BACKUP command.
518	**File name exceeds limit of 259 characters: <value>**
519	**Multi-database undo file names must contain the <DATABASE> tag.** If you are backing up logs for multiple databases and have specified the `STANDBY` keyword, the undo file names must be unique. You must use the <DATABASE> tag to do this. You can use other tags too, but the <DATABASE> tag must be present. For more information about using tags to generate file names, see File location tags.
520	**Output file exists. Will not overwrite. File name: <value>** When you use the `CONVERT` command, you must specify a unique file name for the destination file. For more information, see The CONVERT command.
525	**Folder does not exist. Folder name: <value>**
526	**Cannot find saved setting.** Check that the SQL Backup Agent service startup account has read access to the following registry key and its values: *HKEY_LOCAL_MACHINE\Software\Red Gate\SQL Backup\BackupSettings*

527	**None of the specified folders exist.** Check the transaction log folder locations you have specified for this log shipping restore job.
530	**Non-existent backup file.** Check the path of the backup file. If you are restoring from a backup file on a network share, ensure the startup user for the SQL Backup Agent service has permissions for the network share. For more information, see Backing up and restoring on a network share.
540	**No file name entered for the Microsoft Tape Format (MTF) file.** When you use the `CONVERT` command, you must specify the path for the SQL Backup files (.sqb) and the path for the MTF file. For more information, see The CONVERT command.
550	**Cannot have the same file names for the SQL Backup file and the converted file.**
560	**File does not exist: <value>** Check the path of the backup file in the `RESTORE SQBHEADERONLY` command. If the backup file is on a network share, ensure the startup user for the SQL Backup Agent service has permissions for the network share. For more information, see Backing up and restoring on a network share.
562	**Object is a folder, not a file: <value>**
564	**Error reading file.**
566	**Error reading files not in SQB path.**
570	**File is not a SQL Backup file.** Check the file you are restoring from is not a log file or native backup file.
580	**Failed to open file. Message: <value>** The file may be in use or there may be a permissions problem; ensure the startup user for the SQL Backup Agent service application has been granted the necessary permissions. For more information on the permissions required, see Permissions.
581	**Failed to disconnect existing connections to database.**
584	**System databases must be restored with full recovery using the RECOVERY option.**
585	**Cannot use multiple file names with the THREADCOUNT option.**
586	**Failed to connect to SQL Server instance: <value>**
587	**Backup and restore operations are not allowed on database tempdb.**
588	**You can only perform a full backup of the master database.** Use `BACKUP DATABASE` to back up the entire master database.
589	**Cannot back up the log of the master database.** Use `BACKUP DATABASE` to back up the entire master database.
590	**Insufficient space to perform process. Space required: <value>**
591	**Cannot disconnect existing users from the model database.**
592	**The REPLACE option must be used when using the DISCONNECT_EXISTING option on an active database.**
595	**Failed to create MTF file: <value>** Ensure the startup user for the SQL Backup Agent service application has been granted the necessary permissions. For more information on the permissions required, see Permissions.
596	**File is a native SQL Server backup file.**
597	**Failed to read MTF header data.**
600	**Multiple log file validation error for restore.**

605	**Error creating backup file(s). No files could be created.** Ensure the startup user for the SQL Backup Agent service application has been granted the necessary permissions. For more information on the permissions required, see Permissions.
610	**Error creating file handle for file: <value>** Ensure the startup user for the SQL Backup Agent service application has been granted the necessary permissions. For more information on the permissions required, see Permissions.
612	**Error reading file: unexpected end-of-file reached.**
613	**Compressed data block not found in file.**
620	**Error writing to backup file(s).**
621	**Error writing to backup file(s) due to insufficient disk space.**
622	**Error flushing contents of backup file(s).**
623	**Error setting file pointer position for backup file(s).**
624	**Error writing header (LSN data) to backup file (<value>): <value>.**
625	**Error writing header (backup size) to backup file (<value>): <value>.**
627	**Failed to set file pointer to beginning of file (<value>) to write backup size: <value>.**
628	**Failed to set file pointer to beginning of file (<value>) to read header data: <value>.**
629	**Failed to set file pointer to beginning of file (<value>) to write LSN data: <value>.**
630	**Cannot run a multi device backup with the MIRRORFILE option.** If you are backing up to multiple devices, you cannot use the `MIRRORFILE` option. For more information, see MIRRORFILE in The BACKUP command.
632	**Failed to set file pointer to current position of file.**
636	**BACKUP WITH CONTINUE_AFTER_ERROR successfully generated a backup of the damaged database.**
637	**RESTORE WITH CONTINUE_AFTER_ERROR was successful but some damage was encountered.**
638	**The backup was written with damaged data by a BACKUP WITH CONTINUE_AFTER_ERROR.**
640	**Cannot specify more than 32 devices.** You can specify up to 32 devices in a backup or restore operation. See The BACKUP command and The RESTORE command.
642	**Invalid STOPAT value (<value>): <value>.** For more information on using `STOPAT` when restoring from a transaction log backup, see STOPAT in The RESTORE command.
650	File already flushed to disk; cannot service another flush request.
651	File already flushed to disk; cannot process additional backup data.
653	File already flushed to disk, but buffer not empty.
660	**VDI library not registered.** This may be because the VDI library has become unregistered when other SQL Server components have been uninstalled. To register this library manually, first locate the *sqlvdi.dll* file (for 64-bit versions of SQL Server, you need to locate the 32-bit version of the file). You can then use the *regsvr32* command, for example: `regsvr32 sqlvdi.dll` If you are using more than one version of SQL Server, locate the latest copy of the *sqlvdi.dll* file; later versions of this file are compatible with earlier versions of SQL Server.

662	**The version of the VDI library used does not match the version of SQL Server. Please install the correct version of sqlvdi.dll on this server.** For example, this may occur if you have reinstalled SQL Server 2000, and you are attempting to back up a SQL Server 2005 database.
665	**Beta trial period has expired.**
667	**Trial period has expired.** If you need more time to complete your evaluation, email dba.info@red-gate.com.
668	**64-bit servers require Professional license.** Please contact dba.info@red-gate.com for more information.
680	**The media set for database '<value>' has <value> family members but only <value> are provided. All members must be provided.** The backup set that you are trying to restore is split across multiple files; the number of files you have specified does not match the number of files in the backup set. If this error occurs when restoring transaction log backups, it may be because SQL Backup cannot put the files in the correct order because it cannot read the Log Sequence Numbers from the header files or the information is not there. You can restore these files one at a time in date order, specifying either `WITH NORECOVERY` or `WITH STANDBY`. This will negate the need for the SQL Backup header. See also SQL Backup warning code 170. Alternatively, the SQBHeaderFix utility can be used to repair the files by recreating the SQL Backup header using information from the native SQL header. You can download the utility from: ftp://support.red-gate.com/Patches/SQL_Backup/SQBHeaderFix.zip
700	**No process type entered.**
710	**Wrong password entered.** The `RESTORE` command contains an incorrect password, or if you are restoring from an encrypted backup file, the `PASSWORD` option is missing.
715	**Backup file is not encrypted.** You do not need to specify the `PASSWORD` option in the `RESTORE` command.
716	**None of the backup files are encrypted.** You do not need to specify the `PASSWORD` option in the `RESTORE LOG` command.
720	**No database name entered.**
740	**Failed to get LSN data from server.**
742	**Failed to get LSN data from server (0 rows returned).**
743	**Failed to get file position value from server.**
760	**LSN data from server is blank.**
765	**Failed to read file header (<value>): <value>. Read <value> bytes.**
775	**Failed to open file to write LSN data. Another process might be locking this file: <value>** SQL Backup writes a header block to each backup file upon completion of the backup. In some cases, antivirus applications may lock the backup file immediately after creation, denying SQL Backup access.
780	**General conversion error.**
782	**Browse via SQL: registry value not set.**
784	**Browse via SQL: not sysadmin.**
790	**General thread error or SQL Server error.** Refer to your SQL Server documentation.
800	**File requires SQL Backup version <value> or higher to restore. Current version is <value>.**

810	**Error retrieving SQL Server login name: <value>** Check that the SQL Backup Agent service startup user has read access to the following registry key and its values: *HKEY_LOCAL_MACHINE\Software\Red Gate\SQL Backup\BackupSettings*.
820	**Validation of all backup files failed.**
822	**Encrypted files require a password to be successfully restored.**
828	**Failed to recover database.**
830	**Cannot disconnect existing users when the log is not backed up with the NORECOVERY or STANDBY options.**
840	**File validation failed.**
850	**SQL Backup syntax error.** For more information on SQL Backup syntax, see Scripting SQL Backup.
870	**No command passed to SQL Backup.** The command is empty.
880	**BACKUP DATABASE permission denied in database: <value>** Check that you have backup rights to the relevant database. Ensure that the account the SQL Backup Agent service logs on as (the startup account) is a Windows Authenticated account with *sysadmin* privileges on the SQL Server. For more information about permissions required, see Permissions. If the error persists, turn off the user rights checks by creating a registry entry called *SkipChecks*: 1. Open Registry Editor and locate *HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\SQL Backup\BackupSettingsGlobal\(local)* or *<instance name>* 2. Create a new DWORD type registry entry called *SkipChecks* and set the data value to *1*. 3. From the Windows Services panel, stop and restart the SQL Backup Agent service. To reinstate user rights checks, set the data value to 0 or delete the *SkipChecks* entry.
890	**RESTORE DATABASE permission denied for database: <value>** Check that you have the appropriate restore rights.
900	**Failed to read backup file header (0 rows).**
910	**Failed to read file list details (0 rows).**
915	**Failed to update backup history table (BACKUPSET).**
916	**Failed to update restore history table (RESTOREHISTORY).**
917	**Failed to update backup history table (BACKUPSET). Media set ID is blank.**
918	**Failed to retrieve restore history id.**
920	**Failed to update backup history table (BACKUPMEDIAFAMILY).**
930	**File read error for compressed data. Backup file is incomplete or corrupted (FileBufferRemainingBytes: <value>)**
932	**File read error for compressed data size. Backup file is incomplete or corrupted (CompressedDataSize: <value>)**
934	**File read error for device index. Backup file is incomplete or corrupted (FileBufferRemainingBytes: <value>)**
940	**File read error for <value> bytes (balance <value> bytes). Backup file is incomplete or corrupted.**
942	**File read error for <value> padding bytes (balance <value> bytes). Backup file is incomplete.**
950	**Data decompression error: <value>**
960	**Failed to service command. Error code: <value>**
970	**Backup file does not exist. File name: <value> (for restores).**
980	**Your license does not allow restoring files encrypted with 256-bit keys.**

985	**Error creating critical section: <value>.**
990	**Invalid number of virtual devices: <value>**
995	**Error creating event for virtual device: <value>**
1000*	**Failed to create virtual device. Error code: <value>** If the secondary error is *-2139684857: Failed to recognize SQL Server instance name*, make sure that the SQL Backup Agent service startup account (the account the service logs on as) has read access to the SQL Server service. You can do this using Registry Editor, or with the Windows `sc sdset` and `sc sdshow` commands: • Open Registry Editor (regedit.exe) and locate the *HKEY_LOCAL_MACHINE\Software\Microsoft\MSSQLServer\Setup* subkey. Right-click the subkey and select **Permissions**. In the **Permissions for Setup** dialog, ensure the SQL Backup Agent service startup account has at least read permissions. • Run the commands `sc <server name> sdshow <SQL Backup Agent service name>` to find out which permissions the service user has, then use `sc <server name> sdset <SQL Backup Agent service name>` to set the permissions. For more information, refer to your Microsoft Windows documentation. You can find out which user the SQL Backup Agent service logs on as from the Windows Services panel (services.msc). Note that a SQL Backup Agent service is created for each SQL Server instance that SQL Backup is installed on. This error may also arise if the account the SQL Backup Agent service logs on as is specified in the Windows Domain format (DOMAIN\USERNAME) rather than the User Principal Name format (username@domain). To change the account name format: 1. Open the Windows Services panel (services.msc) and find the SQL Backup Agent service for the affected SQL Server instance, *SQL Backup Agent-<instance name>*. The SQL Backup Agent service for the local instance is called *SQL Backup Agent.* 2. Right-click the service and select **Properties**. 3. On the **Log On** tab, change the format of the account name to UPN format. 4. Restart the SQL Backup Agent service.
1010*	**Failed to get configuration from server. Check that the SQL Server instance is running and that you have the SQL Server System Administrator server role. Error code: <value>. [Also check that the database is not currently in use.]** For certain timeout conditions, a more detailed description may be presented with this error code: **Failed to get the configuration from the server because the timeout interval has elapsed. Check that the SQL Server instance is running, that you have the SQL Server System Administrator server role, and that no other processes are blocking the backup or restore process. If the problem recurs, try increasing the value of the VDITimeout registry setting in HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\SQL Backup\BackupSettingsGlobal\<instance name>.** or, **Failed to get the configuration from the server because the timeout interval has elapsed. SQL Backup required <bytes> bytes of free SQL Server memory, which was not available. You can reduce the memory requirements by reducing the number of backup files/threads used in the backup.** When backing up or restoring, SQL Backup will wait up to 30 seconds for a response from the SQL Server, after which the backup or restore request will timeout. To change the timeout interval, configure the *VDITimeout* registry entry: 1. Open Registry Editor and locate the SQL Backup BackupSettingsGlobal key for the relevant SQL Server instance: *HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\SQL Backup\BackupSettingsGlobal\<instance name>* 2. Create a DWORD entry called "VDITimeout". 3. Set the value to a duration that you feel is adequate for the SQL Server to respond. The duration must be entered in seconds (in hexadecimal).
1020*	**Failed to open virtual device. Error code: <value>**

1030*	**Failed to create VirtualDeviceSet component. Error code: <value>** SQL Backup uses the SQL Virtual Device Interface library to perform backups and restores. This library is contained in a file named *sqlvdi.dll* and is installed and registered when you install SQL Server. This error may arise when backing up and restoring databases for two reasons: • A problem with permissions. The SQL Backup Agent service startup account must be a member of the SQL Server sysadmin role in order to access VDI libraries. For more information about required permissions, see Permissions. • The *sqlvdi.dll* file is missing or has been unregistered, or an incorrect version of SQL Server has been installed: 1. Check the version of the *sqlvdi.dll* file installed on your SQL Server. a. Open Registry Editor and open *HKEY_CLASSES_ROOT\CLSID* (32-bit edition of Windows) or *HKEY_CLASSES_ROOT\Wow6432Node\CLSID* (64-bit edition of Windows). b. Locate the key {40700425-0080-11d2-851f-00c04fc21759}. The key's value *InprocServer32* contains the path to *sqlvdi.dll*. 2. Ensure that the file exists and that the SQL Server startup account has access to it. A 64-bit edition of SQL Server will have two copies of *sqlvdi.dll*, one for 32-bit installations and one for 64-bit installations. Because SQL Backup is a 32-bit application, the 32-bit VDI will be used by SQL Backup while SQL Server will use the 64-bit version. Therefore, it is important that the two copies of *sqlvdi.dll* are the same version. Using the file locations indicated by the **InprocServer32** key, examine the properties of both *sqlvdi.dll* files to ensure the version numbers match.
1040*	**Process terminated unexpectedly.**
1050	**Failed to close VDI.**
1100	**SQL Server error.** More information about selected SQL Server errors is provided in SQL Server errors. For all other errors, refer to your SQL Server documentation.
2010	**Command exceeds permitted length.**
5000	**General inter-process communication (IPC) error.**
5100	**Error creating SQB service event: <value>**
5102	**Error creating SQB service event (already exists): <value>**
5110	**Error opening SQB service event: <value>**
5120	**Error setting SQB service event: <value>**
5130	**Error creating client event: <value>**
5140	**Error creating mutex: <value>**
5142	**Error creating mutex (already exists): <value>**
5145	**Failed to start SQL Backup Agent service.** To start the SQL Backup Agent service manually, use the Windows Services panel.
5146	**Inadequate rights to start SQL Backup Agent service. Please start the SQL Backup Agent service manually.** To start the SQL Backup Agent service manually, use the Windows Services panel.
5148	**SQL Backup cluster resource is stopped.** The SQL Backup Agent service is stopped on the active node in the cluster. This may be because a failover is in progress (in which case the service should restart automatically within a few minutes), or because the service has failed, or has been deliberately stopped. You can re-enable the service with the Windows Cluster Administrator tool (Windows Server 2003) or Failover Cluster Management tool (Windows Server 2008).
5150	**Error opening mutex: <value>**

5160	**Error acquiring mutex: <value>** A mutex is a communication channel between SQL Backup and the SQL Server processes. In certain circumstances a mutex can become orphaned. When SQL Backup attempts to create a new mutex, the request will be rejected by Windows because a mutex of the same name already exists. Rebooting the operating system should resolve this problem. Alternatively, use Process Explorer to identify the process which is causing the problem: 1. Download Process Explorer from http://technet.microsoft.com/en-us/sysinternals/bb896653.aspx and run it. 2. On the **Find** menu click **Find Handle or DLL** to open the Process Explorer **Search** dialog box. 3. Type *SQBMutex* into the **Handle or DLL substring** box and click **Search**. 4. For a SQL Server instance that is functioning correctly, the handles *\BaseNamedObjects\SQBMutex_* and *\BaseNamedObjects\SQBMutex_data_* will exist as part of the process SQBCoreService.exe (the SQL Backup Agent). For instances other than the default, the instance name will be appended to the handle (e.g. *\BaseNamedObjects\SQBMutex_SQL2005*). 5. Select the problematic instance from the list to highlight the entry in the Process Explorer main window. 6. Right click this entry and select **Close Handle**. repeat this for the corresponding data handle. You should now be able to start the SQL Backup Agent service and connect to the server.
5170	**Error releasing mutex: <value>**
5200	**Error creating memory mapped file: <value>**
5202	**Error creating memory mapped file (already exists): <value>**
5210	**Error opening memory mapped file: <value>**
5220	**Error mapping to memory mapped file: <value>**
5230	**Error creating named pipe: <value>**
5240	**SQB service did not acknowledge receipt of data: <value>**
5250	**Failed to connect to Service Control Manager: <value>**
5260	**Failed to connect to SQL Backup service: <value>**
5270	**Error: SQL Backup Agent service is no longer running.**
5280	**Error creating client event.**
5290	**Error creating local data store mutex (already exists): <value>**
5292	**Error creating local data store mutex: <value>**

*VDI errors 1000, 1010, 1020, 1030, 1040

If SQL Backup encounters VDI errors 1000, 1010, 1020, 1030, or 1040, SQL Backup attempts the backup again, reducing the `MAXTRANSFERSIZE` option in case the error was caused by an insufficient amount of contiguous memory on the SQL Server. SQL Backup attempts the backup five times in total, using the `MAXTRANSFERSIZE` values 1048576, 524288, 262144, 131072, 65536 respectively. For more information about the `MAXTRANSFERSIZE` option, see The BACKUP command. For more information about SQL Backup's memory requirements, see Configuring SQL Server memory.

SQL Server errors

You may encounter the following SQL Server errors when using SQL Backup:

- SQL error -1 - SQL Network Interfaces
- SQL Server error 3007 - The backup of the file or filegroup sysft_FTX_MyDatabase is not permitted because it is not online
- SQL Server error 3035 - Cannot perform a differential backup for database <DBname>, because a current database backup does not exist
- SQL Server error 3241 - The media family on device ... is incorrectly formed
- SQL Server error 4214 - BACKUP LOG cannot be performed because there is no current database backup
- SQL Server error 18456 - login failed for user

For information about other SQL Server errors, refer to your SQL Server documentation.

SQL error -1 - SQL Network Interfaces

When performing a backup on a 64-bit edition of SQL Server, you may encounter the following error:

```
SQL error -1: SQL Network Interfaces: Error getting enabled protocols list from registry [xFFFFFFFF].
```

SQL Backup is a 32-bit application and requires access to a 32-bit network client. This error arises if no 32-bit network client is available.

To make a 32-bit network client available:

1. Open SQL Server Configuration Manager.
2. Expand **SQL Native Client Configuration (32bit)** and select **Client Protocols**.
3. Make sure that one or more of the protocols inside the group is enabled.

SQL Server error 3007 - The backup of the file or filegroup sysft_FTX_MyDatabase is not permitted because it is not online

When performing a full backup of a SQL Server 2005 database, the following error may occur:

> SQL error 3007: The backup of the file or filegroup "sysft_FTX_MyDatabase" is not permitted because it is not online.

This error may be caused by problems with the full-text catalog.

1. Open SQL Server Management Studio and, in the **Object Explorer**, expand the relevant database.
2. Expand **Storage** > **Full Text Catalogs**.
3. Right-click the relevant catalog and select **Rebuild**.

SQL Server should now consider the full-text catalog "online" and you can take a full backup of the database.

For more information about repairing full-text catalogs, see Microsoft Support (KB 923355).

SQL Server error 3035 - Cannot perform a differential backup for database <DBname>, because a current database backup does not exist

When attempting a differential backup of a database, the following error may occur:

> SQL error 3035: Cannot perform a differential backup for database <DBname>, because a current database backup does not exist.

This error may occur if:

- The recovery model of the database has been changed to *Simple*.
- The recovery model of the database has been changed to *Full*, but no backup has been performed since the change.
- A job using `BACKUP DATABASES [*]` or `BACKUP ALL DATABASES` has run when a full backup does not exist for one or more databases on the server.
  This may occur if you have added a database to the server since setting up the job. For more information, see Backing up all databases on an instance.

To resolve this problem, perform a full backup on the database in question.

SQL Server error 3241 - The media family on device ... is incorrectly formed

When restoring a database, the following error may occur:

> SQL error 3013: RESTORE FILELIST is terminating abnormally.
>
> SQL error 3241: The media family on device <device name> is incorrectly formed.

There are two possible causes for this error:

- This error arises if you attempt to restore a database from a backup created on a later version of SQL Server to an earlier version of SQL Server. You can only restore to the same or a later version of SQL Server. This is a SQL Server limitation.
- This error may also arise if the SQL Backup Agent service doesn't have the necessary permissions. For example, if you are restoring across a network, the SQL Backup Agent service must have read permission to access the folder containing the backups. For more information on the permissions required by the SQL Backup Agent service, see Permissions.

SQL Server error 4214 - BACKUP LOG cannot be performed because there is no current database backup

When attempting a log backup of a database, the following error may occur:

> BACKUP LOG cannot be performed because there is no current database backup.

This error may occur if:

- The recovery model of the database has been changed to *Simple*.
- The recovery model of the database has been changed to *Full*, but no backup has been performed since the change.
- A job using `BACKUP LOGS [*]` has run when a full backup does not exist for one or more databases on the server.
  This may occur if you have added a database to the server since setting up the job. For more information, see Backing up all databases on an instance.

To resolve this problem, perform a full backup on the database in question.

SQL Server error 18456 - login failed for user

A backup or restore task may fail with the following error:

> SQL error 18456: Login failed for user <value>

There are two possible causes for this error.

If you have ***changed the authentication mode*** used by the SQL Backup Agent service from SQL Server authentication to Windows authentication, you may need to clear the *ServiceLogin* registry entry.

> You will need to stop the SQL Server service briefly to do this.

1. From the Windows Control Panel, select **Administrative Tools**, then **Services**.
2. Find the SQL Server service for the relevant instance of SQL Server, called "SQL Server (<instance name>)".
3. Right-click the SQL Server service and select **Stop**.
4. Open the Registry Editor and locate the *ServiceLogin* registry entry:
    a. 32-bit systems: *HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\SQL Backup\BackupSettingsGlobal\<instance name>*
    b. 64-bit systems: *HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Red Gate\SQL Backup\BackupSettingsGlobal\<iInstance name>*
5. Clear the data value for the *ServiceLogin* registry entry, but do not delete the entry itself.
6. In **Services**, right-click the SQL Server service and select **Start**.

If the SQL Backup Agent service connects using SQL Server authentication, and you have ***changed the SQL Server account password***, you need to update the password for the SQL Backup Agent service.

1. Add the *sqbsetlogin* extended stored procedure from the SQL Backup extended stored procedure dynamic link library *xp_sqlbackup.dll*.
2. Execute *sqbsetlogin* and specify the user name and new password.
3. Remove the *sqbsetlogin* extended stored procedure when you have finished using it. (This step is optional but recommended.)

For example:

```
EXECUTE master..sp_addextendedproc sqbsetlogin, 'xp_sqlbackup.dll'
EXECUTE master..sqbsetlogin 'sa', 'newpassword'
EXECUTE master..sp_dropextendedproc sqbsetlogin
```

For more information on the *sqbsetlogin* extended procedure, see Permissions.

Other errors

You may encounter the following errors when using SQL Backup:

- Error - Cannot load the DLL xp_sqlbackup.dll, or one of the DLLs it references
- Error - Could not find procedure master..sqbutility
- Error - IO error on backup or restore restart-checkpoint file
- Error - The database disk image is malformed
- System error 32 - The process cannot access the file because it is being used by another process

Error - Cannot load the DLL xp_sqlbackup.dll, or one of the DLLs it references

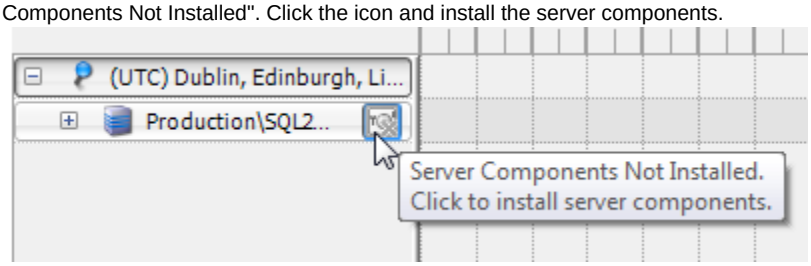
When a backup is performed using the SQL Backup extended stored procedure, the following error may occur:

> Cannot load the DLL xp_sqlbackup.dll, or one of the DLLs it references.
>
> Reason: 1114 (A dynamic link library (DLL) initialization routine failed.).

This error occurs when the old *xp_sqlbackup.dll* file has not been freed. To fix the problem, uninstall and reinstall the SQL Backup server components:

> You will need to stop the SQL Server service briefly to fix the problem.

1. Close the SQL Backup graphical user interface.
2. From the Windows Control Panel, select **Administrative Tools**, then **Services**.
3. Find the SQL Server service for the affected SQL Server instance, *SQL Server (<instance name>)*.
4. Right-click the SQL Server service and select **Stop**. Then right-click the SQL Server service and select **Start**.
5. From the Windows Control Panel, select **Programs and Features**.
6. Find the server components for the affected SQL Server instance, called "SQL Backup <version>-<instance name> (server components)". Right-click the server components and select **Uninstall**.
   Click **Yes** when asked if you want to completely uninstall the server components.
   Click **No** when asked if you want to remove the SQL Backup file containing the history of the backup and restore activity.
7. Open the SQL Backup GUI. In the Registered SQL Servers pane on the left, the affected instance has an icon next to it indicating "Server Components Not Installed". Click the icon and install the server components.



> Do not uninstall and reinstall the server components more than once, or the SQL Backup Agent service may be removed. If this happens, install the server components manually by running *SQBServerSetup.exe*. For more information, see Installing the server components manually.

You may also need to find the SQL Backup Agent service in **Services**, as above, and enable and restart it manually:

1. Find the SQL Backup Agent service for the affected SQL Server instance, *SQL Backup Agent-<instance name>*. The SQL Backup Agent service for the local instance is called *SQL Backup Agent*.
2. Right-click the service and select **Properties**.
3. Change the **Startup type** to **Automatic** and click **OK**.
4. Right-click the service and select **Start**.

Error - Could not find procedure master..sqbutility

When installing the SQL Backup server components on a SQL Server instance using the SQL Backup graphical user interface, the following error may arise:

```
Could not find procedure 'master...sqbutility'
```

To resolve this issue, install the server components manually:

1. Open the SQL Backup installation folder. By default this is:
    - *%Program Files%\Red Gate\SQL Backup 6* on 32-bit servers
    - *%Program Files (x86)%\Red Gate\SQL Backup 6* on 64-bit servers
2. Locate SQBServerSetup.exe and copy it to the machine on which you want to install the server components.
3. Run SQBServerSetup.exe and follow the instructions in the wizard. For more information, see Installing the server components manually.

If the error persists, install the *sqbutility* extended stored procedure manually by running the following command against the *master* database of the SQL Server:

```
execute sp_addextendedproc sqbutility, "xp_sqlbackup.dll"
```

If *xp_sqlbackup.dll* is not found, copy the file from the SQL Backup server components installation folder to the SQL Server *Binn* folder, then run the command again. By default, *xp_sqlbackup.dll* is installed in:

- *%ProgramFiles%\Red Gate\SQL Backup 6\<instance name>\<platform>* on 32-bit machines
- *%ProgramFiles(x86)%\Red Gate\SQL Backup 6\<instance name>\<platform>* on 64-bit machines

Note that the extended stored procedure is platform specific.

By default, the SQL Server *Binn* folder is found at *%ProgramFiles%\Microsoft SQL Server\MSSQL.x\MSSQL\Binn*

Error - IO error on backup or restore restart-checkpoint file

When backing up or restoring, the following error may occur:

> I/O error on backup or restore restart-checkpoint file 'C:\Program Files\Microsoft SQL Server\MSSQL\backup\MyDatabase.ckp'.
> Operating system error 3(The system cannot find the path specified.). The statement is proceeding but is non-restartable.

This error occurs when SQL Server's default backup directory does not exist.

1. Open Registry Editor (regedit.exe).
2. Locate the *BackupDirectory* registry entry in the the SQL Server registry folder:
   HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Microsoft SQL Server\MSSQL.n\MSSQLServer\BackupDirectory
3. Make sure that the *BackupDirectory* registry entry points to a location that exists. The SQL Server and SQL Server Agent services should have read/write permissions to the location.

Error - The database disk image is malformed

On starting SQL Backup, the following error may occur:

> The database disk image is malformed.

This is caused by a corruption of the activity cache, which stores backup and restore history.

To fix this problem, close the SQL Backup graphical user interface (GUI) and delete the activity cache files from the machine running the SQL Backup GUI. By default these are stored in:

- *%USERPROFILE%\Local Settings\Application Data\Red Gate\SQL Backup\Server Data* (Windows 2003 and Windows XP), or
- *%LOCALAPPDATA%\Red Gate\SQL Backup\Server Data* (Windows 2008, Windows 2008 R2, and Windows 7)

When you next open the SQL Backup GUI, the activity cache is populated from the *msdb* database and the SQL Server Compact database for each SQL Server that you have added to the GUI. This may take some time.

System error 32 - The process cannot access the file because it is being used by another process

When creating a backup, the following error may occur:

> Warning: System error 32 (The process cannot access the file because it is being used by another process).

This error may arise if another application is accessing the backup file and preventing SQL Backup from writing to it.

A common cause of this error is an on-access scanner running as part of an anti-virus system. An on-access scanner checks a file for viruses when the file is first written or opened. This prevents SQL Backup from writing the backup file. To avoid this problem, exclude the backup location from the on-access scan.

If you do not have an on-access scanner, use Process Explorer to identify the process that is accessing the backup file:

1. Find the path and file name that SQL Backup Pro attempted to write the backup to. If you used tags to generate the file name, you can find the file name in the Activity Log:
    - In the SQL Backup graphical user interface, open the Activity History and locate the failed backup. Right-click and select **Show Log** to display the Activity Log for the backup.
    - Alternatively, open the relevant log file. By default, these are created in: %PROGRAMDATA%\Red *Gate**SQL Backup**Log*\<*instance name*> (Windows Vista, Windows 2008 and later) or *%ALLUSERSPROFILE%*\ *Application Data**Red Gate**SQL Backup**Log*\<*instance name*> (Windows XP and Windows 2003).
2. Download Process Explorer from the Microsoft website and run it on the computer on which you tried to create the backup.
3. On the **Find** menu, select **Find Handle or DLL**.
4. In the **Handle or DLL substring** box, type the full backup path and file name, then click **Search**.
5. Process Explorer displays the applications that are currently accessing the file. Stop the application from accessing the file, or run the backup at a time when the application is not accessing the file.

Troubleshooting

- Log files
- Cannot run backup or restore operations
- Stopping a backup or restore job while in progress
- Slow backup or restore operations
- Deleting backup and restore history manually
- Browsing SQL Servers with SQL authentication
- Cannot access resource when browsing SQL Servers with Windows authentication
- The SQL Backup Agent service cannot start, or is taking too long to start up
- Configuring SQL Server memory
- Optimizing backup speed
- Performance expectations
- Specifying file paths

Log files

SQL Backup produces two different types of log file, application log files and activity log files. These files are useful if you have encountered a problem and are working with Redgate support to resolve it.

Application log files

Application log files collect information about the application while you are using it.

By default, application log files are stored in:

- *%LOCALAPPDATA%\Red Gate\Logs\SQL Backup* (Windows Vista, Windows 2008 and later)
- *%UserProfile%\Local Settings\Application Data\Red Gate\Logs\SQL Backup* (Windows XP and Windows 2003)

Activity log files

Activity log files collect information about each backup and restore activity performed in SQL Backup.

For more detailed information on activity log files, see SQL Backup log files in File management options.

Cannot run backup or restore operations

If you cannot run backup or restore operations, check that:

- The SQL Backup server components are correctly installed on the SQL Server instance. The server components include the SQL Backup Agent service and the SQL Backup extended stored procedure dynamic-link library:
  - The SQL Backup Agent service application file, *SQBCoreService.exe*, should be located in the server components installation folder. By default this is *%ProgramFiles%\Red Gate\SQL Backup 6\<instance name>* on 32-bit machines and *%ProgramFiles(x86)%\Red Gate\SQL Backup 6\<instance name>* on 64-bit machines.
    The SQL Backup Agent service for the SQL Server instance should be created as part of the installation. Use the Windows Services panel to check the service exists and is started.
  - The SQL Backup extended stored procedure dynamic-link library, *xp_sqlbackup.dll*, should be located in the server components installation folder. By default this is *%ProgramFiles%\Red Gate\SQL Backup 6\<instance name>* on 32-bit machines and *%ProgramFiles(x86)%\Red Gate\SQL Backup 6\<instance name>* on 64-bit machines. The dynamic-link library should also be copied to the SQL Server instance's *Binn* folder. By default this is *%ProgramFiles%\Microsoft SQL Server\<SQL Server version>\MSSQL\Binn*.

  If the server components are not installed, run SQBServerSetup.exe to install them manually. For more information, see Installing the server components manually.
- The SQL Backup Agent service startup account has the correct permissions. For more information, see Permissions.
- You have activated the SQL Backup server components on the instance that you are trying to back up or restore to, or you have a valid trial. For more information see Licensing.
- Your SQL Server instance is running. Ensure that you have started the Microsoft SQL Server services.
  If you are using the command line and use named instances instead of default instances, ensure that you have set the `-I` command line parameter to the named instance. For more information, see Using the command line.

Stopping a backup or restore job while in progress

To stop a backup or restore job while it is in progress, stop the SQL Backup Agent service. You will need to restart the service to allow other backup and restore jobs to run.

1. From the Windows Control Panel, select **Administrative Tools**, then **Services**.
2. Find the SQL Backup Agent service for the relevant instance of SQL Server, for example *SQL Backup Agent-<instance name>*. The SQL Backup Agent service for the local instance is called *SQL Backup Agent*.
3. Right-click the service and select **Stop.** To restart the service, right-click it and select **Start.**

It is not possible to stop the SQL Backup Agent service using the SQL Backup graphical user interface.

Slow backup or restore operations

Backup or restore operations may take longer than expected. This may be caused by a number of processes run by SQL Backup or by the SQL Server.

To check the progress of any SQL Backup backup or restore operations currently executing on a SQL Server, run the *sqbstatus* extended stored procedure against the *master* database:

```
exceute master..sqbstatus
```

The results are listed per database:

- *Processed (bytes)* represents the uncompressed size of the data that has been processed.
- *Compressed (bytes)* represents the compressed size of the data that has been processed.

If *sqbstatus* reports *0* bytes, this indicates that SQL Server is performing tasks that must be completed before the backup or restore operation can begin, and is not currently able to accept data from SQL Backup. For information on identifying the tasks SQL Server is performing, see SQL Server processes below.

Deleting old backup files

If you include the `ERASEFILES`, `ERASEFILES_ATSTART` or `ERASEFILES_REMOTE` option in a `BACKUP` command, or select the option to **Delete existing backup files in this folder for selected database** in the Back Up or Schedule Backup Jobs wizard, SQL Backup deletes backups of the same database and type from the backup folder as part of the backup operation. To identify the files to delete, SQL Backup reads the headers of all the files in the backup folder. The larger the number of files stored in the folder, the longer it will take SQL Backup to read all of the file headers.

To reduce the number of file headers SQL Backup has to read each time, store backups in folders according to database name and backup type. You can do this automatically, by including the <DATABASE> and <TYPE> tags in folder paths. For example, the following command will create full backups of Database 1, Database 2 and Database 3 in *C:\Backups\Database 1\Full*, *C:\Backups\Database 2\Full* and *C:\Backups\Database 3\Full* respectively:

```
"BACKUP DATABASES [Database 1, Database 2, Database 3] TO DISK =
'C:\Backups\<DATABASE>\<TYPE>\<DATETIME ddmmyy>.sqb' WITH ERASEFILES = 7"
```

The backup file names are based on the date and time of the backup

To avoid having to write out the tags each time, you can include tags in the default backup location used by SQL Backup, then use the <AUTO> tag in the `BACKUP` command. When using the Back Up or Schedule Backup Jobs wizard, the backup location is populated with the default location.

- For more information about setting the default backup location, see File management options.
- For more information about the tags you can use, see File location tags.

Deleting backup and restore history

If you have selected the option to **Delete all backup and restore history older than <n Days | Hours>** (available from **Tools** > **Server Options** > **File Management**), SQL Backup uses the stored procedure *msdb..sp_delete_backuphistory* to delete history from the *msdb* database when the graphical user interface is running. For more information about this option, see File management options.

If the *msdb* database contains a lot of history, or if multiple SQL Backup jobs are completing at the same time, deleting the history can slow down backup and restore operations. You may find it helpful to add indexes to the backup and restore history tables in the *msdb* database. For more information, refer to your SQL Server documentation. Alternatively, clear this option and run the stored procedure manually at a convenient time or as a scheduled task. For more information, see Deleting backup and restore history manually.

SQL Server processes

To find out which processes are being run by SQL Server as part of a backup or restore, and how long those processes take, set a trace flag on the SQL Server *master* database:

```
DBCC TRACEON (3004, 3605, -1)
```

Setting the trace flag causes additional information to be written to the SQL Server logs. To view the logs, open SQL Server Management Studio and in **Object Explorer** open **Management** > **SQL Server Logs** > **Current**.

Deleting backup and restore history manually

The SQL Backup graphical user interface (GUI) provides the option to delete the backup and restore history from the *msdb* database and SQL Server Compact database for each SQL Server (**Tools** > **Server Options** > **File Management**). For more information, see File management options.

Selecting this option can cause performance problems when running backup or restore operations, or when using the SQL Backup GUI. If this is the case, you may find it helpful to clear the option and delete backup and restore history manually or as a scheduled task.

The following script deletes backup and restore history older than the specified number of days from the *msdb* database, the SQL Server Compact database and the SQL Backup GUI activity cache:

```
--------------------------------------------------------------------------------
--------------
-- Delete backup history from SQL Backup's SQL Server Compact database
--------------------------------------------------------------------------------
--------------
SET NOCOUNT ON
DECLARE @backup_id INT ,
    @command NVARCHAR(4000) ,
    @retain_date NVARCHAR(64) ,
    @IDList VARCHAR(2000)

-- number of days of history that you want to retain (currently 30 days)
SET @retain_date = GETDATE() - 30

-- temp table for ids we're deleting.
IF OBJECT_ID('tempdb..#DeleteTheseIDs') IS NOT NULL
    DROP TABLE #DeleteTheseIDs
CREATE TABLE #DeleteTheseIDs ( id INT )

-- get total from the SQL Server Compact database (this will also show an error if its
corrupted)
EXEC [master]..sqbdata N'select COUNT(*) AS TotalBackupRows from backuphistory'
EXEC [master]..sqbdata N'select COUNT(*) AS TotalRestoreRows from restorehistory '

--------------------------------------------------------------------------------
--------------
-- Delete backup history from msdb database
--------------------------------------------------------------------------------
--------------
-- first do a standard delete
EXEC msdb.dbo.sp_delete_backuphistory @retain_date

--------------------------------------------------------------------------------
--------------
-- Delete backup history from SQL Backup GUI activity cache
--------------------------------------------------------------------------------
--------------

 --get all of the ids that we want to delete
SET @command = 'select id from backuphistory where backup_end < '''
    + @retain_date + ''''
TRUNCATE TABLE #DeleteTheseIDs
INSERT  INTO #DeleteTheseIDs
        EXEC [master]..sqbdata @command
SELECT  'Deleteing # Backup ids ' = COUNT(*)
FROM    #DeleteTheseIDs

 --loop until they're all deleted
```

```
WHILE EXISTS ( SELECT   1
               FROM     #DeleteTheseIDs )
    BEGIN

  --get next set of ids (do not get too many at a time, otherwise the list will be
truncated)
        SET @IDList = ''
        SELECT TOP 150
                @IDList = @IDList + CASE WHEN @IDList = '' THEN ''
                                         ELSE ','
                                    END + CONVERT(VARCHAR(10), id)
        FROM    #DeleteTheseIDs
        ORDER BY id

  --delete history
        SET @command = 'delete from backupfiles where backup_id IN ('
            + @IDList + ')'
        EXEC [master]..sqbdata @command
        SET @command = 'delete from backuplog where backup_id IN (' + @IDList
            + ')'
        EXEC [master]..sqbdata @command
        SET @command = 'delete from backuphistory where id IN (' + @IDList
            + ')'
        EXEC [master]..sqbdata @command

  --delete from temp file
        SET @Command = 'DELETE FROM #DeleteTheseIDs where id IN (' + @IDList
            + ')'
        EXEC (@Command)
    END

-------------------------------------------------------------------------------------
--------------
-- Delete restore history from SQL Backup GUI activity cache
-------------------------------------------------------------------------------------
--------------
 --get all of the ids that we want to delete
SET @command = 'select id from restorehistory where restore_end < '''
    + @retain_date + ''''
TRUNCATE TABLE #DeleteTheseIDs
INSERT  INTO #DeleteTheseIDs
        EXEC [master]..sqbdata @command
SELECT  'Deleting # Restore ids ' = COUNT(*)
FROM    #DeleteTheseIDs

 --loop until they're all deleted
WHILE EXISTS ( SELECT   1
               FROM     #DeleteTheseIDs )
    BEGIN

 -- get next set of ids
        SET @IDList = ''
        SELECT TOP 200
                @IDList = @IDList + CASE WHEN @IDList = '' THEN ''
                                         ELSE ','
                                    END + CONVERT(VARCHAR(10), id)
        FROM    #DeleteTheseIDs
        ORDER BY id
```

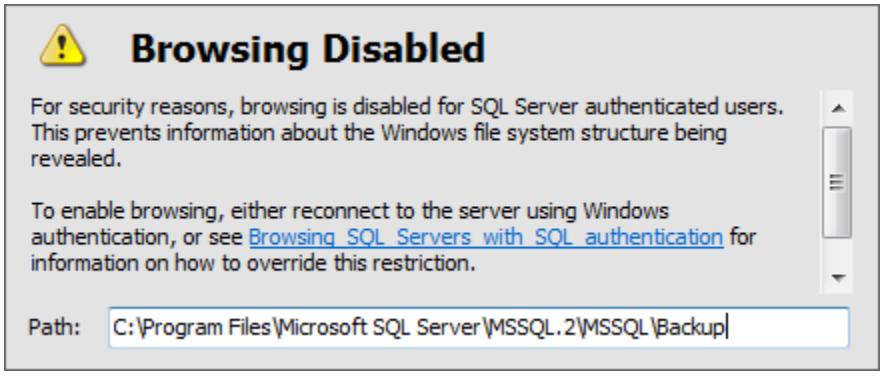
```
--delete history
      SET @command = 'delete from restorefiles where restore_id IN ('
          + @IDList + ')'
      EXEC [master]..sqbdata @command
      SET @command = 'delete from restorelog where restore_id IN ('
          + @IDList + ')'
      EXEC [master]..sqbdata @command
      SET @command = 'delete from restorehistory where id IN (' + @IDList
          + ')'
      EXEC [master]..sqbdata @command

--delete from temp file
      SET @Command = 'DELETE FROM #DeleteTheseIDs where id IN (' + @IDList
          + ')'
```

```
        EXEC (@Command)
    END
SELECT  'Done.'
```

Browsing SQL Servers with SQL authentication

When using the Folder or File Browser in the SQL Backup graphical user interface (GUI), you may encounter a Browsing Disabled error:



This error occurs if you are using SQL Server authentication to connect to a SQL Server through the SQL Backup graphical user interface (GUI). Browsing to folders or files on the SQL Server is disabled by default to prevent information about the file system structure being revealed to SQL authenticated users who do not have permissions to browse the file system on the SQL Server.

To connect to the SQL Server using Windows authentication of the user currently using the GUI:

1. Ensure the user connecting to the GUI is a member of the SQL Server *sysadmin* fixed server role.
2. In the SQL Backup GUI, right-click the SQL Server and select **Edit**.
3. On the **General** tab, select **Windows authentication**, then click **Connect**.

To enable folder and file browsing by SQL Authenticated users, create the *AllowSQLBrowsing* registry entry for the SQL Server instance.

1. Open the Registry Editor (regedit.exe).
2. Navigate to the following registry key:
   HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\SQL Backup\BackupSettingsGlobal\<instance name>
   where *<instance name>* is the SQL Server whose file system you want to browse.
3. Create a registry entry of type DWORD called *AllowSQLBrowsing* and set the value to *1*. You must have administrator privileges to do this.

This setting will not take effect until you refresh the connection to the SQL Server instance in the SQL Backup GUI. To do this, right-click the SQL Server and select **Refresh Connection** or press F5.

Cannot access resource when browsing SQL Servers with Windows authentication

If you are using Windows authentication to connect to a SQL Server through the SQL Backup graphical user interface and try to access a network share through the Folder Browser or File Browser, you may encounter the following error:

> Cannot access resource. Check that you have the correct permissions to view this resource.

This is caused by Windows security restrictions that prevent a "double hop" of credentials from one server to another, using an intermediate server. There are two possible solutions to this problem:

- Browsing network shares using the SQL Backup Agent service startup account
- Chaning the Windows authentication method to Kerberos

Browsing network shares using the SQL Backup Agent service startup account

This solution is only available if the SQL Backup Agent service startup account (the account the service logs on as) is a domain account with access to the network resources.

The SQL Backup Agent service is created for each SQL Server instance when you install the SQL Backup server components. The name of the service is in the format: *SQL Backup Agent-<instance name>* You can change the account the service uses to log on from the Windows Services panel.

To allow a user to browse a network share using the SQL Backup Agent service startup account:

1. Open Registry Editor (regedit.exe).
2. Navigate to the following registry key:
   HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\SQL Backup\BackupSettingsGlobal\<instance>
   where *<instance>* is the SQL Server you are connecting to through the SQL Backup GUI.
3. Create a Multi-String Value entry called *BrowsingUserList*.
4. In the Value data, list each of the Windows users that you want to allow to browse the network shares using the SQL Backup Agent service startup account. Each user should be listed on a new line in the format *DOMAIN\USER*.

Changing the Windows authentication method to Kerberos

This solution is only available if you have domain administrator privileges and access to the **Active Directory Users and Computers** applet on the Windows Active Directory server containing the computer account.

To change the Windows authentication method from NTLM to Kerberos you need to register a Service Principal Name for the SQL Server Service, then use Active Directory to trust the computer and any accounts for delegation.

Registering a Service Principal Name

To register a Service Principal name for the SQL Server service:

1. Ensure the server has a fully-qualified DNS entry, by pinging the IP address of the machine. (You can use *ipconfig* to get the IP address.)

   ```
   ping -a <IP address>
   ```

   The ping should return a full-qualified domain name for the server. If it does not, contact your network administrator to get the server added to your network's Domain Name Server.
2. For Windows Server 2003, install the SetSPN utility on the SQL Server.

   > Microsoft has issued an update to SetSPN for Windows 2003. For more information, see Microsoft Technet, and to download, see Microsoft Support.

   For Windows 2008, SetSPN is a built-in command line tool, available if you have the Active Directory Domain Services server role installed. For more information, see Microsoft Technet.
3. Open an elevated command prompt and change directory to the folder where the utility is installed, then run this command:

   ```
   SetSPN -S MSSQLSvc/server.domain.local:1433 SERVER
   ```

   where: *server.domain.local* is the fully-qualified domain name of the server, and *SERVER* is the NETBIOS name of the server.

Trusting the computer and accounts for delegation

To trust the computer and any accounts for delegation you must have domain administrator privileges and access to the **Active Directory Users and Computers** applet on the Windows Active Directory server containing the computer account.

- If the SQL Server service runs as the *Local System* account, follow the steps below to trust the computer for delegation.
- If the SQL Server service runs as a domain account, follow the steps below to trust both the computer and domain account for delegation.

To trust the computer for delegation:

1. Open **Active Directory Users and Computers** and locate the computer account of the SQL Server. Right-click and select **Properties**.
2. Open the **Delegation** tab and select **Trust this computer for delegation to specified services only**. (On Windows 2000, this option is on the **General** tab.)
3. Click **Add**, then **Users or Computers** and select the computer running the SQL Server.
4. Select the **MSSQLSvc** service type and click **OK**, then **OK** again.

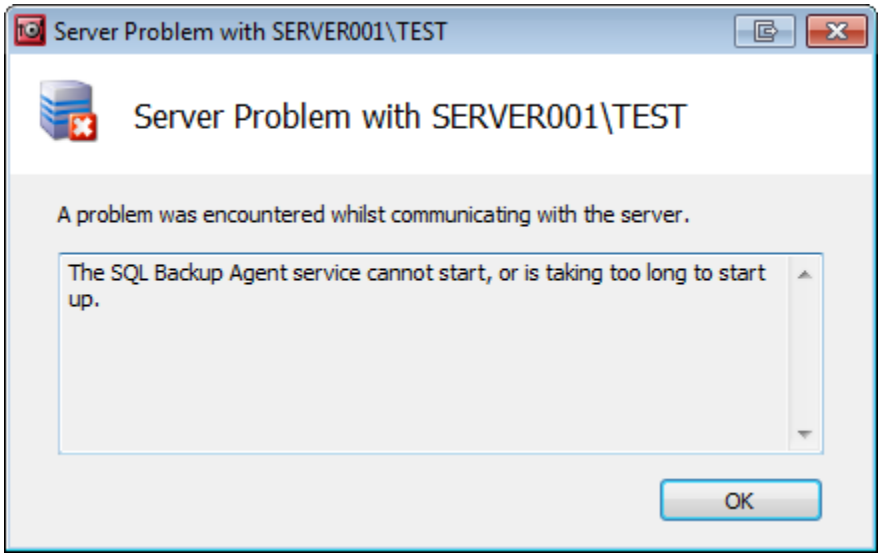
> On SQL Server clusters, this procedure must be done on the computer accounts of all nodes in the cluster.

To trust the domain account for delegation:

1. Open **Active Directory Users and Computers** and locate the domain account used by the SQL Server service. Right-click and select **Pr operties**.
2. Open the **Account** tab and:
    a. Ensure that **Account is sensitive and cannot be delegated** is not selected.
    b. Select **Account is trusted for delegation**.

The SQL Backup Agent service cannot start, or is taking too long to start up

You may encounter the following error when attempting to connect to a SQL Server in the Registered SQL Servers pane:



This problem may arise after uninstalling the SQL Backup Pro server components. The uninstall process can leave the server components in a partially uninstalled state.

To use SQL Backup Pro on the SQL Server instance:

1. Install the server components manually by running SQBServerSetup.exe on the server running the SQL Server instance. For more information, see Installing the server components manually.
   Note: It is not possible to install the server components from the SQL Backup Pro GUI in this situation.
2. Start the SQL Backup Agent service for the instance:
   a. Open the Windows Services panel (services.msc).
   b. Locate the SQL Backup Agent service for the SQL Server instance. For example, *SQL Backup Agent-<instance name>*. The SQL Backup Agent service for the local instance is called *SQL Backup Agent*.
   c. Right-click the service and select **Properties**.
   d. Set the **Startup type** to **Automatic** then click **Start** to restart the service.
3. Return to the SQL Backup Pro GUI, select the SQL Server instance and from the **View** menu select Refresh Connection. You should now be able to connect to the SQL Server instance from the SQL Backup Pro GUI and perform backup and restore operations.

If you no longer want to use SQL Backup Pro on the SQL Server instance, you can delete the instance from the Registered SQL Servers pane, delete the SQL Backup Agent service for the instance and drop the SQL Backup extended stored procedures from the instance.

- To delete the SQL Backup Agent service, use the `sc delete <service name>` command. For example, at the command prompt type:

  ```
  sc delete SQLBackupAgent_PRODUCTION
  ```

  You can find out the service name from the Windows Services panel (services.msc).
  For more information about the sc delete command, see Microsoft Technet documentation.
- To delete the SQL Backup Pro extended stored procedures from the instance, use SQL Server Management Studio's Object Explorer (Databases > System Databases > master > Programmability > Extended Stored Procedures) or `sp_dropextendedproc`. For example:

  ```
  USE master
  GO
  sp_dropextendedproc 'dbo.sqlbackup'
  ```

  The following extended stored procedures are installed when you install the SQL Backup Pro server components:
  - sqbdata
  - sqbdir
  - sqbmemory
  - sqbsetlogin
  - sqbstatus

- sqbtest
- sqbtestcancel
- sqbteststatus
- sqbutility
- sqlbackup

Configuring SQL Server memory

A backup or restore may fail because of insufficient SQL Server memory. SQL Backup errors 1000, 1010, 1020, 1030 or 1040 may also be returned.

How SQL Backup uses SQL Server memory

SQL Server provides a limited amount of memory outside the SQL Sever memory pool for loading stored procedures and OLE objects. SQL Backup uses this memory to create the Virtual Device Interface (VDI), which SQL Backup uses to communicate with SQL Server in order to perform backups and restores:

- When SQL Backup performs a backup, the backup data is written by SQL Server to the VDI. SQL Backup then processes the data and writes the backup files.
- When SQL Backup performs a restore, it reads the data from the backup file and sends it to the VDI for SQL Server to process.

In order to create the VDI, SQL Backup requires, on average, contiguous memory equivalent to 6 times the `MAXTRANSFERSIZE` value for each thread used to perform the backup or restore.

- The number of threads used to create a backup is determined by:
  - The `THREADCOUNT` option when backing up one or more databases to single files. You can specify this option from the Back Up or Schedule Backup Jobs wizards, or add it to the BACKUP command.
  - The number of `DISK` arguments when backing up a single database and splitting the backup into multiple files. You can specify this option from the Back Up or Schedule Backup Jobs wizards, or add it to the BACKUP command.
  - The `FILECOUNT` option when backing up multiple databases and splitting each backup into multiple files. You can specify this option from the Back Up or Schedule Backup Jobs wizards, or add it to the BACKUP command.
- You can specify the `MAXTRANSFERSIZE` value for creating backups from the Back Up or Schedule Backup Jobs wizards, or by adding the option to a BACKUP command.
- When restoring from a backup, the number of threads used to create the backup and the default `MAXTRANSFERSIZE` value of 1048576 bytes (1MB) are applied, unless you specify otherwise by adding the `MAXTRANSFERSIZE` option to the RESTORE command (or change the default value in the registry).

If SQL Backup error 1000, 1010, 1020, 1030, or 1040 is raised during a backup operation, SQL Backup attempts the backup up to 5 more times, using the `MAXTRANSFERSIZE` values 1048576, 524288, 262144, 131072, 65536 respectively, in order to reduce the amount of contiguous memory required.

Dealing with memory errors

If a backup or restore operation fails with SQL Backup error 1000, 1010, 1020, 1030 or 1040, attempt the operation again, writing the backup to a single file without multiple threads, or restoring from a single file created without multiple threads, and set the `MAXTRANSFERSIZE` value to 65536. This will reduce as much as possible the amount of contiguous memory SQL Backup requires.

If a backup or restore operation fails despite setting the `MAXTRANSFERSIZE` value to 65536 and not using multiple threads, it is likely that SQL Server does not have enough contiguous memory to perform the operation. To check the amount of contiguous memory available to the SQL Server, use the SQL Backup *sqbmemory* extended stored stored procedure:

```
EXECUTE master..sqbmemory
```

The *Maximum* value for *Free* memory indicates how much contiguous memory is available for SQL Backup. To make more memory available, reduce the number of other processes running on the SQL Server.

You can use the SQL Server `-g` startup option to control how much memory SQL Server leaves available for loading items such as extended stored procedure .dll files. For more information, see the Microsoft documentation on Database Engine Service Startup Options.

For more information on allocating minimum and maximum memory for SQL Server, see the Microsoft documentation on Server Memory Server Configuration Options.

Optimizing backup speed

The backup process can be separated into three distinct stages:

- Stage 1: SQL Server backup engine reads the data and log files.This incurs disk Input/Output (I/O) reads.
- Stage 2: SQL Backup compresses, and optionally encrypts, the data.This uses CPU cycles.
- Stage 3: SQL Backup writes the resulting compressed data files to disk.This incurs disk I/O writes.

To optimize the speed of your backup, you optimize each of these stages in turn:

1. Use the SQL Backup command line or extended stored procedure to run a backup using the `NOCOMPRESSWRITE` argument (see The BACKUP command).
   When you use `NOCOMPRESSWRITE`, SQL Backup simulates a backup process without compression, and no backup files are created. This simulates Stage 1.
   The throughput for a backup process is shown on the SQL Server report. When you use the `NOCOMPESSWRITE` argument, this shows the maximum possible backup throughput attainable on your system.
   If you are using a multi-processor system, test the effect of using multiple threads. 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.
   The limiting factor for Stage 1 is usually the speed at which your disk can read the data from the disks.
2. Use SQL Backup to run a backup using the NOWRITE argument (see The BACKUP command).
   When you use the NOWRITE argument, SQL Backup simulates a backup process using the specified compression level and no backup files are created. This simulates Stage 1 and Stage 2.
   You can change the compression level to see the effect that the different levels have on the backup speed.
   To improve backup throughput, use multiple threads until you achieve a throughput close to that you achieved with the `NOCOMPRESSWRITE` argument.
3. Run the backup process to completion, using the optimum number of threads found at Stage 2.
   To optimize Stage 3, store the backup data on a different set of disks from the disks used to store the data and log files. Depending upon the type of disk controllers, you have, you may need to reduce the number of backup devices so that you do not overload the disk I/O writes. Your aim is to balance the number of backup devices with the number of disks you can back up, to achieve a throughput close to that achieved at Stage 2. The Current Disk Queue Length performance counter tells you when you have reached the maximum capacity of your disks. Generally, the number should not exceed twice the number of drives on your disk array. If you achieve this, your backup process has been optimized.

Example

1. A backup was run using `NOCOMPRESSWRITE`. SQL Server reported a throughput of 103 MB/sec. The number of threads was increased to two. Throughput remained at 103 MB/sec. Therefore, the limiting factor for this setup is the disk read speed, at 103 MB/sec.
2. The same backup was run using `NOWRITE`, with compression level 1.
   SQL Server reported a throughput of 60 MB/sec. The number of threads was increased to two, and throughput increased to 99 MB/sec. The number of threads was then increased to three, and throughput increased to 103 MB/sec. This is the maximum disk read speed found at Stage 1.
   The process was then repeated for compression level 2. With one thread, throughput was 80 MB/sec; with two and three threads, throughput was 103 MB/sec.
   Therefore, optimum backup speed is achieved using compression level 2 with two threads.
3. The same backup was then run to completion (without the `NOCOMRESSWRITE` or `NOWRITE` arguments) using compression level 2 with two threads.
   Throughput was 77 MB/sec. The number of threads was increased to 3, and SQL Server reported a throughput drop to 71 MB/sec. Therefore, two threads is optimum choice for this particular backup on this system. The *Current Disk Read Queue* performance counter averaged 13 for a 4 disk array.

Performance expectations

This page compares the native SQL Server backup and restore performance against similar operations performed by SQL Backup.

The numbers shown here should be used only as a broad indicator of expected performance. The performance you can achieve using SQL Backup depends upon your own setup, such as hard disk configuration and throughput. For details of how you can optimize the speed of your backups, see Optimizing backup speed.

Results for a 22 GB database

This test was run on a four processor server with 6 GB of RAM. A single backup file was created using eight threads.

	Native backup	SQL Backup (compression level 1)
Backup		
Speed	49.660 MB/sec	101.517 MB/sec
File size	19.781 GB	7.766 GB
Elapsed time	7 min 7 sec	3 min 29 sec
Restore		
Speed	17.447 MB/sec	43.465 MB/sec
Elapsed time	20 min 15 sec	8 min 9 sec

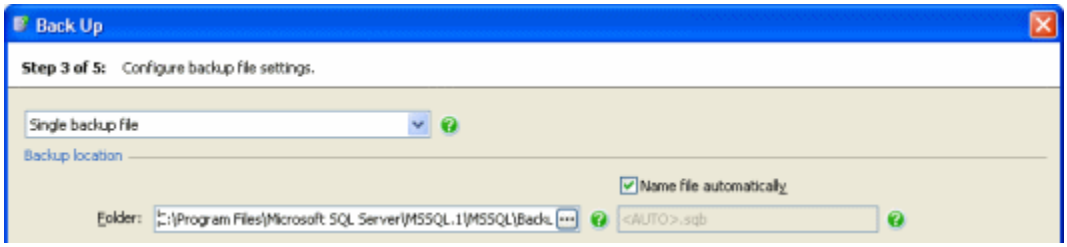
Results for a 306 GB database

This test was run on an eight processor server with 12 GB of RAM. A single backup file was created using sixteen threads.

	Native backup	SQL Backup (compression level 1)
Backup		
Speed	83.849 MB/sec	164.274 MB/sec
File size	306.561 GB	115.746 GB
Elapsed time	7 min 7 sec	3 min 29 sec
Restore		
Speed	29.682 MB/sec	51.105 MB/sec
Elapsed time	184 min 50 sec	107 min 20 sec

Specifying file paths

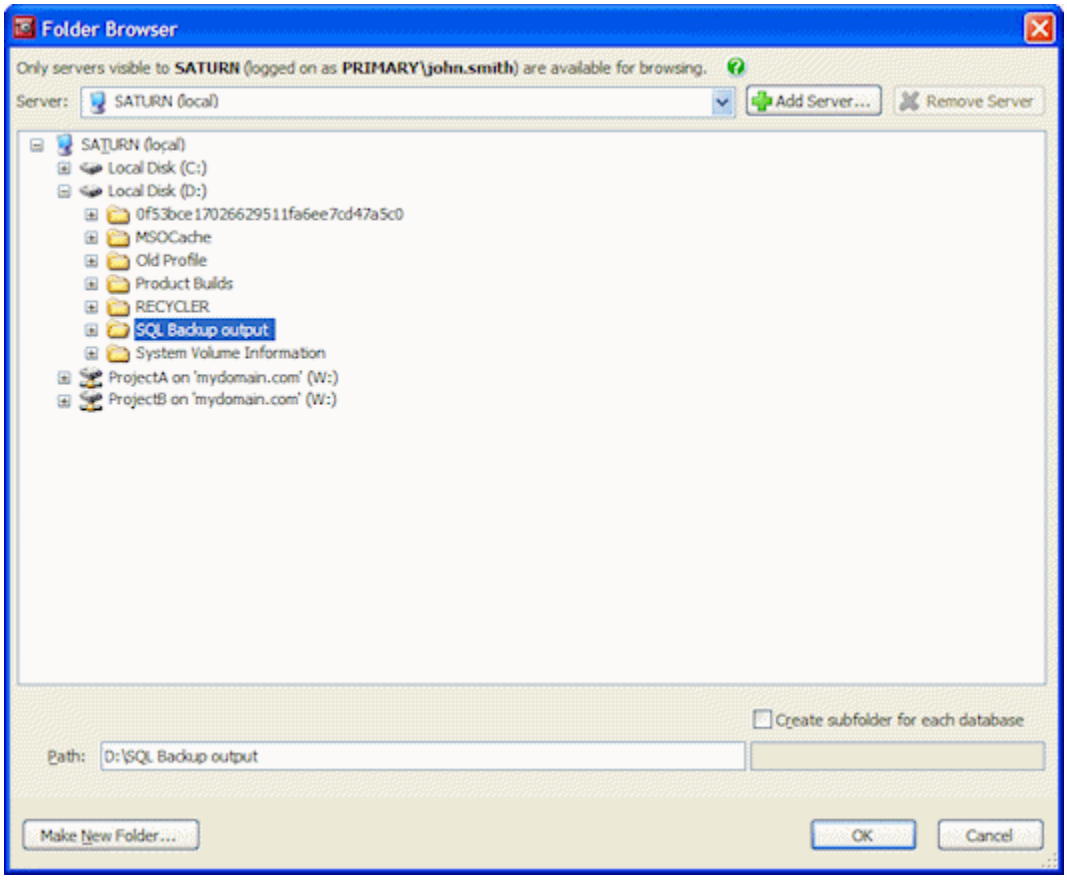
SQL Backup often requires you to specify a file path or name, as, for example, in the Back Up wizard:



You can type the file path and name in the boxes. To specify a network path, type the full path, including the server name, for example *\\ServerName\MyFolder\MyFile*

> 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 server.

Alternatively, you can click  to browse to the folder or file you require. For example:



SQL Backup displays the local file system for the selected SQL Server. By default this is the local server, which you have chosen to back up to or restore from.

You can also browse for resources on other servers, provided they are visible to the local server. Other servers will only be visible to the local server if it has the appropriate permissions to write to or read from them. The name of the local server you are connected to and your user name are displayed above the **Server** list. This information may explain why some servers cannot be browsed.

To see the file system on other servers:

1. Click the **Server** list to see if the server you want to access is listed (the local server is always available).

2. Select the server from the list. If the server you require is not displayed in the list, click **Add Server** and type the name or IP address of the server to add.

   To remove a server from the list, select it and and click **Remove Server**.

Browsing SQL Servers with SQL Server authentication

If you are using SQL Server authentication on a SQL Server, for security reasons browsing to folders or files on the SQL Server is disabled by default. This is to prevent information about the file system structure being revealed to SQL Authenticated users who do not have permissions to browse the file system on the SQL Server.

You can override this restriction. However, you should do this only after careful consideration.

To enable file browsing by SQL Authenticated users, you must edit the registry of the SQL Server whose file system you want to browse. In the registry folder HKEY_LOCAL_MACHINE\SOFTWARE\Red Gate\SQL Backup\BackupSettingsGlobal\<instance name> create the registry key **AllowSQLBrowsing** as type **DWORD**, and set the value to *1*. You must have administrator privileges to do this. This setting will not take effect until you refresh the connection to the instance.

For information about switching from from SQL Server authentication to Windows authentication see Editing a SQL Server instance registration.

Release notes and other versions

Version 7.7 (current)	July 8th, 2014	Release notes	Documentation
Version 7.6	December 4th, 2013	Release notes	
Version 7.5	November 29th, 2013	Release notes	
Version 7.4	June 20th, 2013	Release notes	
Version 7.3	March 19th, 2013	Release notes	
Version 7.2	September 4th, 2012	Release notes	Help file (CHM format) | Documentation
Version 7.1	June 21st, 2012	Release notes	Help file (CHM format) | Documentation
Version 7.0	April 24th, 2012	Release notes	Help file (CHM format) | Documentation
Version 6.5	May 9th, 2011	Release notes	Help file (CHM format) | Documentation
Version 6.4	March 10th, 2011	Release notes	Help file (CHM format) | Documentation
Version 6.3	November 19th, 2009	Release notes	Help file (CHM format) | Documentation
Version 6.2	August 11th, 2009	Release notes	Help file (CHM format) | Documentation
Version 6.1	July 16th, 2009	Release notes	Documentation
Version 6.0	July 9th, 2009	Release notes	Help file (CHM format) | Documentation
Version 5.4	November 20th, 2008	Release notes	Help file (CHM format)
Version 5.3	February 26th, 2008	Release notes	Help file (CHM format)
Version 5.2	October 1st, 2007	Release notes	*Documentation not available*
Version 5.1	May 25th, 2007	Release notes	
Version 5.0	May 1st, 2007	Release notes	
Version 4.6	August 17th 2006	Release notes	
Version 4.5	July 3rd, 2006	Release notes	
Version 4.2	May 3rd, 2006	Release notes	
Version 4.1	February 9th, 2006	Release notes	
Version 4.0	January 17th, 2006	Release notes	
Version 3.2	May 10th, 2005	Release notes	
Version 3.1	April 6th, 2005	Release notes	
Version 3.0	February 17th, 2005	Release notes	

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

CHM help files may appear empty after downloading; for help see How to view CHM files

SQL Backup 6.5 release notes

May 9th, 2011

New features

- Ability to choose "existing database file location" as well as the SQL Backup suggested defaults.

Fixes

- Improvements to purging backup files in the following cases:
  - when folders contain spaces in the name.
  - when the restored database name differs from the original database name.
- Support for `KEEP_CDC`.
- `LATEST_ALL` keyword fixes for restoring the latest full backup and and all subsequent log and differential backups, in particular:
  - handling large backup sets.
  - handling SQB files created with earlier versions of SQL Backup.
- Network resilience bug fixes.
- Product activation improvements.
- Improved debugging information and error handling:
  - Improved timeout for VDI functions and threaded SQL connections.
  - Debug information for decompression.
  - More exception handling code when writing process data for the GUI.
  - When creating new folders.
  - Partially created backup files when running of disk space.
  - Detect if transaction log backup file has previously been restored during log shipping..

Other changes

- Quality improvement program - users now have the option to send usage information back to Redgate to help us improve future versions.

SQL Backup 6.4 release notes

March 10th, 2010

New features

- Keyword support for restoring a database using the latest backups, including all available transaction log files.
- Keyword support for backing up all user databases or all system databases on an instance.
- Keyword support for specifying a different source database when restoring from disk (with `LATEST` keywords).

Fixes

- `DISCONNECT_EXISTING` can now be used with RESTORE on databases that are already in a "Recovering" state.
- Database name is now case-insensitive when restoring with `LATEST_FULL`.
- Fixes to automatic deletion of old SQL Backup log files.
- Fixes to prevent event-log flooding with Microsoft Cluster Service errors on active/active clusters.
- Miscellaneous minor bug fixes in the engine and GUI.

SQL Backup 6.3 release notes

November 19th, 2009

Fixes

- New keyword support for network resilience to restore commands.
- New support for erasing backup files (with options) independently from a backup command.
- New keyword support for backups to use native SQL Server 2008 Enterprise Edition's compression level.
- New keyword support for restore jobs to restore the most recent backup file.
- New dialog to activate serial numbers on multiple servers.
- Bug fix to be able to import SQL Server registrations into SQL Backup from SQL Server 2008 Management Studio.
- Improvements to the file browser for 64-bit systems.
- Improvements around handling the conversion of *sqb* files.
- Better error handling for certain situations, such as when deduplication is used or if the disk is running out of space.
- Miscellaneous minor bug fixes in engine and GUI.

SQL Backup 6.2 release notes

August 11th, 2009

New features

- New object level recovery functionality (SQL Object Level Recovery Pro) to enable the recovery of individual database objects from SQL Backup (.sqb) backups.

Fixes

- Minor bug fixes.

SQL Backup 6.1 release notes

July 16th, 2009

Fixes

- When backing up multiple databases transaction logs, backup files could fail to copy.
- Verifying log backups could result in it failing to copy.

SQL Backup 6.0 release notes

July 9th, 2009

New features

- Network resilience for backups to protect against temporary network outages (Pro edition).
- Network resilience for log shipping to protect against more prolonged network outages (Pro edition).
- New compression level 4 (Pro edition).
- Kill existing connections as part of the restore process (Pro edition).
- Log shipping now includes an option to skip the initialization of the destination database (Pro edition).
- Check for orphaned usersas part of the restore process (Pro edition).
- Enhanced time line responsiveness.
- Ability to step back through the wizards to re-run an operation if a problem occurs.
- Manage existing backup files with new purge options.
- Ability to view SQL Backup log files from the user interface.
- Ability to deactivate licenses from the user interface.
- Improved file browser.

Fixes

- Various bug fixes.

SQL Backup 5.4 release notes

November 20th, 2008

New features

- Support for Windows Server 2008.

Fixes

- Fix for MDAC error on Windows Server 2008.
- Fix for restore hanging when the backup file is in use.
- Fix for `ERASEFILES_ATSTART` not deleting the backup files when used with `FILEOPTIONS`.

SQL Backup 5.3 release notes

February 26th, 2008

New features

- Support for the `WITH PARTIAL` keyword.
- Inclusion of new maintenance plan conversion tool.
- Option to send email notifications when a warning is generated.
- Different purge options for local and network drives.
- Option to limit the data imported from the *msdb* database when registering a server.
- Choice of location for the data.sdf at install time.
- Optional failover if the SQL Backup service fails to start in a clustered environment.

Fixes

- Usability updates to the Back Up wizard.
- Performance improvement for the `ERASEFILES` and `ERASEFILES_ATSTART` options.
- Improved stability for compression level 3.
- Improved handling of timeouts.

SQL Backup 5.2 release notes

October 1st, 2007

New features

- Application changes for easier and faster installation of server components.

Fixes

- Various bug fixes.

SQL Backup 5.1 release notes

May 25th, 2007

Fixes

- Various bug fixes.

SQL Backup 5.0 release notes

May 1st, 2007

New features

- Complete redesign of the GUI.
- Timeline Monitoring™: a central planning and monitoring system for activities from within the GUI.
- Enterprise and server-level reporting from Timeline Monitoring.
- Integrated support for clusters.
- Multiple threads in the engine to optimize backup processes.
- New keywords `THREADCOUNT` and `FILECOUNT` in the `BACKUP` command (replace `THREADS`).
- Support for `RESTRICTED_USER`, `STOPATMARK`, and `STOPBEFOREMARK` keywords in the `RESTORE` command.
- Support for `REPLACE` in the `RESTORE LOG` command.
- Support for Microsoft Windows Vista.

Fixes

- Various bug fixes.

SQL Backup 4.6 release notes

August 17th, 2006

New features

- Support for the `KEEP_REPLICATION` and `PAGE` keywords.
- Enhanced `COPYTO` argument for the `BACKUP` command – tags can now be used to specify the folder.

Fixes

- Various bug fixes.

SQL Backup 4.5 release notes

July 3rd, 2006

New features

- Compatible with 64-bit versions of SQL Server.
- Runs the same setup file to install SQL Backup on 32-bit or 64-bit SQL Servers. SQL Backup will automatically install the appropriate files for your configuration.
- New keyword `MAXTRANSFERSIZE` to specify the maximum size of each block of memory to be used when SQL Backup transfers data.
- Help file shipped as part of installation.
- Minor feature enhancements and bug fixes.

SQL Backup 4.2 release notes

May 3rd, 2006

New features

- Support to SQL 2005 SP1.
- New keyword `MAXDATABLOCK` to instruct SQL Backup to write out data to the network shares in smaller blocks.

Fixes

- Various bug fixes.

SQL Backup 4.1 release notes

February 9th, 2006

New features

- Improved installation – you can select the SQL Backup Agent service startup account and SQL Server authentication type for new installations.
- Added support for new SQL Server 2005 backup/restore options:
  - `CHECKSUM`
  - `NO_CHECKSUM`
  - `CONTINUE_AFTER_ERROR`
  - `STOP_ON_ERROR`
  - `COPY_ONLY`
  - `READ_WRITE_FILEGROUPS`
- Added option to specify the folder in which to store the SQL Backup log files.
- Added shell extension library to provide more information about the *.sqb* files.

Fixes

- Various bug fixes.

SQL Backup 4.0 release notes

January 17th, 2006

New features

- Pro and standard editions split.
- Remote back up and restore (Pro only).
- 256-bit encryption (Pro only; standard version still has 128-bit encryption).
- Support to SQL Server 2005.
- Improved compression algorithm.
- Back up file/filegroup.
- Schedule Backup Wizard (can select multiple databases to script in one go).
- New UI.
- Log Shipping Wizard.
- Backup and restore history data will be saved in the msdb table instead (as per native SQL Server) as opposed to in a file on disk. This makes it possible to get backup/restore history data from remote servers. The new Activity history grid will display the last few backups and restores, with SQB backups/restores highlighted and includes statistics of the compression achieved and the speed.
- You can determine the age of old backups, and choose when they should be deleted (before or after running the next backup) in hours as well as days.
- You can set to delete SQL Backup log files (stored on disk), older than a certain age.
- Extended stored procedure can now return last SQB error code and last SQL error code.
- Unicode outputs ok in extended stored procedure. This means if you run remotely to a Chinese SQL Server, for example, all outputs in Chinese will be displayed correctly on an English OS if the necessary fonts are installed.
- Can be set to send an email only when there's an error.
- Can script to back up multiple databases using, for example, `BACKUP DATABASES [*]` or `BACKUP DATABASES [a, b, c]` (similarly for log backups, `BACKUP LOGS [*]`).
- New keyword `EXCLUDE` (so you can say, for example, back up all databases except these ones).
- New keyword `THREADS` to automatically back up to a given number of files, so that the user does not need to name every file in a split backup e.g. `BACKUP DATABASE pubs TO DISK = [<AUTO>] WITH THREADS = 2`.
- New Job Failures tab, to see scheduled jobs which have caused an error.
- Facility to highlight databases which have not been backed up within a specified amount of time – backup alerts.
- Removed Tools dialog (MD5# and verify files).
- Scripts are displayed with syntax highlighters.
- A reset button to go back to the original file name format.
- Improved support for usage of tags in backup file names.
- Improved extended stored procedure feedback.
- Added management dialog for extended stored procedure rights.
- Added display of server properties.
- Added support for Unicode text.

SQL Backup 3.2 release notes

May 10th, 2005

New features

- Added new keyword `THREADPRIORITY` to allow setting of SQL Backup thread priorities.
- SQL Server login passwords are now encrypted.
- <AUTO> tag now works with `MIRRORFILE` option.
- Wildcards can now be used in place of filenames during `RESTORE` operations.
- Extended stored procedure library is now installed in the SQL Server *Binn* folder to allow for easier upgrades.

Fixes

- Various bug fixes.

SQL Backup 3.1 release notes

April, 2005

New features

- Added new keyword `MAILTO_ONERROR` to send email notification only on errors.
- Fixed `MOVETO` command which was incorrectly moving corrupted files.

Fixes

- Minor bug fixes.

SQL Backup 3.0 release notes

February 17th, 2005

First release by Redgate.