SQL Response 1.3

Learning topics

redgate®

ingeniously simple tools

Contents

Getting started

SQL Response monitors your SQL Server instances, and raises alerts to warn you about issues such as failed jobs, blocked processes, low memory problems and connection errors.

When a problem occurs on one of your monitored SQL Servers, the relevant alert is raised, containing details of exactly what has happened, and at what time. Each alert contains useful diagnostic data to help you identify the cause of the problem.

SQL Response also recommends maintenance actions to take on your databases, based on the data it collects about each SQL Server.

SQL Response works with any combination of SQL Server 2000, SQL Server 2005 and SQL Server 2008.

Using SQL Response

You can:

- investigate alerts by looking at performance counters (page 51), captured SQL Profiler trace statements, and system processes
- change the level (page 66) at which an alert is raised (alerts can be raised at high, medium or low level)
- adjust the thresholds for each alert type (page 8)
- disable alerts (page 66) on one or more SQL Servers
- set up configuration templates (page 60) to manage alerts across a number of instances at once
- filter the list of alerts (page 42) by SQL Server, by groups of servers, by alert type and by timeframe
- create your own set of filters and save them as a view
- add comments (page 55) when clearing alerts
- set up email (page 70) to be sent automatically whenever an alert is raised
- view recommendations (page 14) about databases on your monitored SQL Servers

Getting more information

- What's new in this version of SQL Response? (http://www.red-gate.com/supportcenter/Content.aspx?p=SQL%20Response&amp;c=SQL_Response%5carticles%5cversion_1xx_SQLResponse.htm)
- Find out about activating SQL Response (page 32).
- How can I check for updates to SQL Response? (http://www.red-gate.com/support/page?c=all_products%5carticles%5ccheck_for_updates.htm)
- Find out more about Red Gate. (http://www.red-gate.com/about/index.htm)

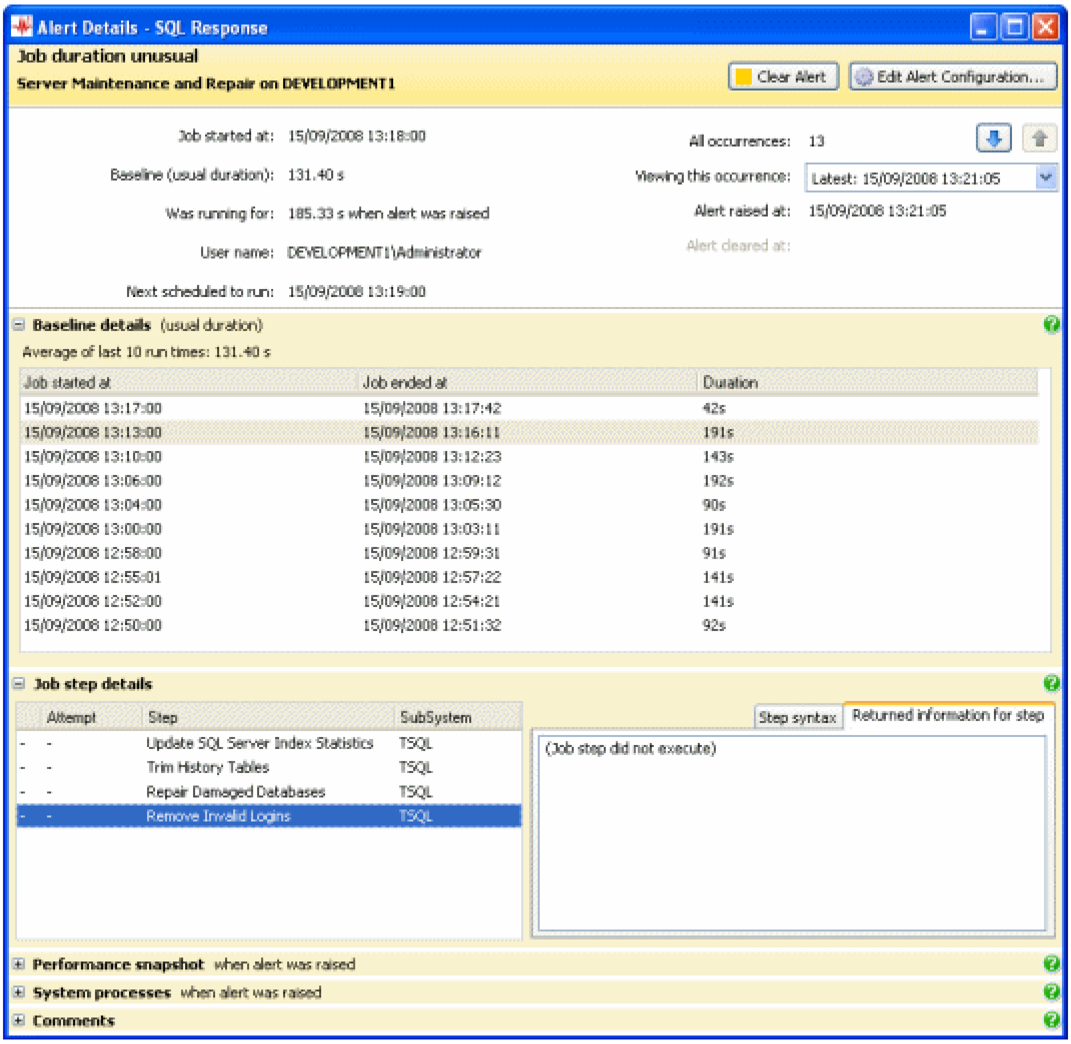
About alerts and recommendations

SQL Response informs you about issues on your monitored SQL Servers in two different ways:

Alerts are raised only when a specific issue occurs, based on thresholds you can configure; for example when a block or deadlock occurs, a job fails or does not start, or available disk space falls below a specified percentage.

Recommendations offer advice concerning some best practice issues for managing your databases, based on certain conditions SQL Response checks for at set intervals.

Example of an alert



Alert Details - SQL Response
Clear Alert
Edit Alert Configuration...
Performance snapshot when alert was raised
System processes when alert was raised
Comments

Job duration unusual

Server Maintenance and Repair on DEVELOPMENT1

Job started at: 15/09/2008 13:18:00

Baseline (usual duration): 131.40 s

Was running for: 185.33 s when alert was raised

User name: DEVELOPMENT1\Administrator

Next scheduled to run: 15/09/2008 13:19:00

All occurrences: 13

Viewing this occurrence: Latest: 15/09/2008 13:21:05

Alert raised at: 15/09/2008 13:21:05

Alert cleared at:

Baseline details (usual duration)

Average of last 10 run times: 131.40 s

Job started at	Job ended at	Duration
15/09/2008 13:17:00	15/09/2008 13:17:42	42s
15/09/2008 13:13:00	15/09/2008 13:16:11	191s
15/09/2008 13:10:00	15/09/2008 13:12:23	143s
15/09/2008 13:06:00	15/09/2008 13:09:12	192s
15/09/2008 13:04:00	15/09/2008 13:05:30	90s
15/09/2008 13:00:00	15/09/2008 13:03:11	191s
15/09/2008 12:58:00	15/09/2008 12:59:31	91s
15/09/2008 12:55:01	15/09/2008 12:57:22	141s
15/09/2008 12:52:00	15/09/2008 12:54:21	141s
15/09/2008 12:50:00	15/09/2008 12:51:32	92s

Job step details

	Attempt	Step	SubSystem
-	-	Update SQL Server Index Statistics	TSQL
-	-	Trim History Tables	TSQL
-	-	Repair Damaged Databases	TSQL
-	-	Remove Invalid Logins	TSQL

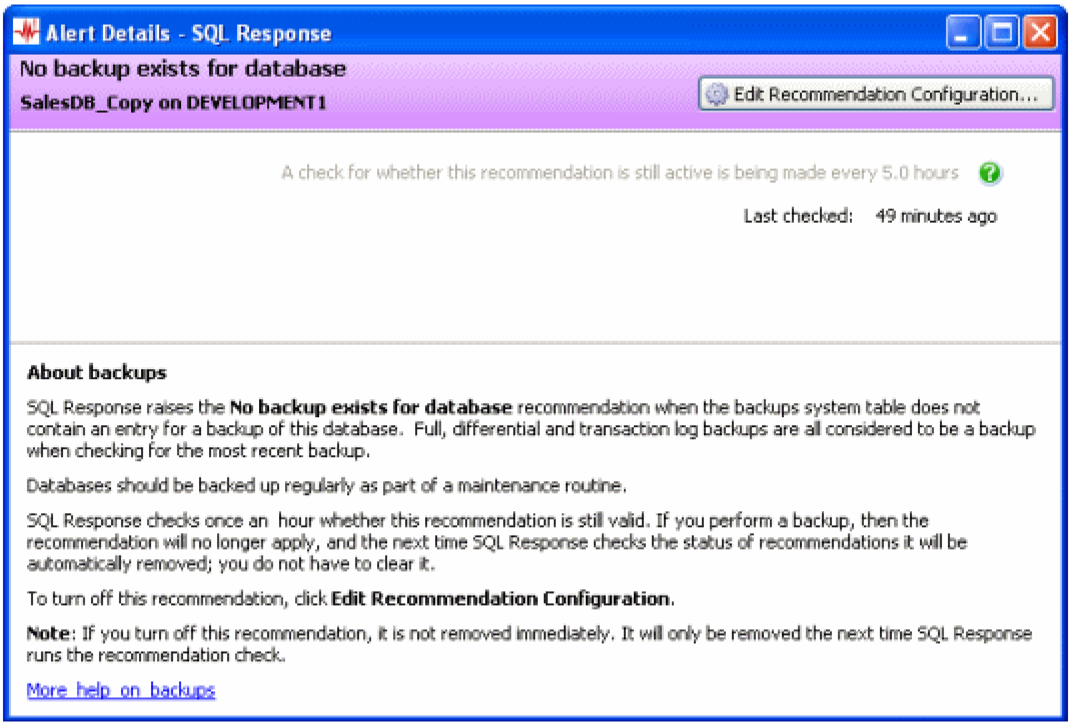
Step syntax | Returned information for step

(Job step did not execute)

This is a **Job duration unusual** type alert, for the Server Maintenance and Repair job on the DEVELOPMENT1 SQL Server.

It has been raised because the job was running for longer than usual, based on the average duration of the last ten runs.

Example of a recommendation



Alert Details - SQL Response

No backup exists for database

SalesDB_Copy on DEVELOPMENT1

Edit Recommendation Configuration...

A check for whether this recommendation is still active is being made every 5.0 hours

Last checked: 49 minutes ago

About backups

SQL Response raises the **No backup exists for database** recommendation when the backups system table does not contain an entry for a backup of this database. Full, differential and transaction log backups are all considered to be a backup when checking for the most recent backup.

Databases should be backed up regularly as part of a maintenance routine.

SQL Response checks once an hour whether this recommendation is still valid. If you perform a backup, then the recommendation will no longer apply, and the next time SQL Response checks the status of recommendations it will be automatically removed; you do not have to clear it.

To turn off this recommendation, click **Edit Recommendation Configuration**.

Note: If you turn off this recommendation, it is not removed immediately. It will only be removed the next time SQL Response runs the recommendation check.

More help on backups

This is a **No backup exists for database** type recommendation for the SalesDB_Copy database on the DEVELOPMENT1 SQL Server. It has been raised because SQL Response cannot find any backup entry for this particular database in the backups system table.

It was last checked 49 minutes ago, and will be checked again in 4 hours and 11 minutes time. If a backup is found for this database when SQL Response next checks, the recommendation will be removed.

Differences between alerts and recommendations

- **Alerts** display a number of sections containing detailed information about the problem, performance snapshot data, and any comments added when the alert was last cleared.
- **Recommendations** display some text to describe the generic issue, and a link to further information on the Red Gate Support website.
- **Alerts** have a status of active or cleared, and can be raised at a number of levels (low, medium, or high).
- **Recommendations** do not have different levels, and cannot be manually cleared; they are automatically removed when the issue has been resolved.

Actions that apply only to alerts

- view different occurrences of an alert; for example, if a job fails several timse in a day, you can view each separate failure
- clear alerts; clearing an alert changes its status from active to cleared
- add comments to an alert
- configure alerts to be raised at low, medium, or high level
- filter alerts by timeframe; for example, to view only alerts from the last hour, the last 12 hours, the last week, month and so on
- set up an email to be sent when an alert is raised
- view historical alerts

Actions that apply to both alerts and recommendations

- filter by SQL Server, and by type of alert or recommendation
- sort by any column (for example Type, Detail or Last Occurred)
- save custom filters as a view; the views available for alerts are separate from the views available for recommendations.
- enable or disable them; disabled recommendations are only removed the next time SQL Response checks their status; disabled alerts will not be raised in future, but existing alerts of that type are still available to view.
- configure thresholds (not all alerts or recommendations have configurable thresholds)
- open in a new window

The presence of alerts and recommendations on a specific SQL Server are both indicated by a colored bar next to the SQL Server name in the **Servers to show** area.

List of alerts

SQL Response raises the following types of alert:

Computer unreachable

Raised when:	Computer cannot be contacted on the network
Configurable thresholds:	None
Possible causes:	• Computer is turned off • WMI (the interface through which Windows computers provide information and notification) is not enabled

Computer login failure

Raised when:	Login to computer unsuccessful
Configurable thresholds:	None
Possible causes:	• Incorrect Windows account credentials specified • Disabled Windows account

SQL Server unreachable

Raised when:	SQL Server instance not running or cannot be contacted This alert will always be raised following the **Computer unreachable** alert for any SQL Server instances on the computer that cannot be contacted
Configurable thresholds:	None
Possible causes:	SQL Server instance has been stopped

SQL Server login failure

Raised when:	Login to SQL Server instance unsuccessful
Configurable thresholds:	None

Possible causes:

- Incorrect credentials specified (username invalid, unauthorised user, or password incorrect)
- SQL Server is in single user mode
- Alert Repository service account is not sysadmin

Job failure

Raised when: Job does not complete successfully (returns error code)

Configurable thresholds: None

Possible causes: Various

Check the **Job step details** area of the alert to review the returned message for each job step

Job did not start

Raised when: A scheduled job did not start at the expected time

Configurable thresholds: None

Possible causes:

- Previous execution of this job was overrunning
- SQL Server Agent was not running (a **SQL Server Agent not running** alert is raised when SQL Response first detects that SQL Server Agent is not running)
- No target servers defined

Job duration unusual

Raised when: The job execution time has exceeded or fallen below the baseline duration (the average of the last ten runs) by a specified percentage

Alert indicates whether the job has underrun or overrun

Configurable thresholds:	% longer than baseline % shorter than baseline
Possible causes:	• High system load • Suboptimal SQL execution plan • Job is waiting on an event or blocked
Notes:	SQL Response calculates the baseline duration by using the job history to find the last ten successful run times SQL Response will not raise the **Job duration unusual** alert until the job history contains at least ten successful runs

SQL Server Agent not running

Raised when:	No SQL Server Agent service running
Configurable thresholds:	None
Possible causes:	SQL Server Agent has been stopped, or encountered a problem and failed

Low disk space

Raised when:	Amount of space available on a disk drive has fallen below a specified level
Configurable thresholds:	% of used space on disk
Possible causes:	• Database and log files may be growing too large without frequent backups • Other applications may be using the disk drive for file storage

Low physical memory

Raised when:	Physical memory usage has exceeded a specified level
Configurable thresholds:	% of used physical memory
Possible causes:	Computer may be overloaded

Long running query

Raised when:	Query has been running for longer than a specified duration
Configurable thresholds:	Query duration (seconds)
Possible causes:	• Complex query • Insufficient physical memory • CPU over-utilized

Blocked SQL process

Raised when:	A SQL connection has been waiting for another process to release its blocking lock for longer than a specified duration
Configurable thresholds:	Block duration (seconds)
Possible causes:	• Long-running queries. • Using Insert, Update or Delete on large numbers of records in a single transaction. • Canceling queries, but not rolling them back

SQL deadlock

Raised when:	Deadlock detected
Configurable thresholds:	None
Possible causes:	• Inefficient application code • Application accesses objects in a different order each time • User input during transactions • Lengthy transactions • Locks not being released as early as possible

SQL Server error log entry

Raised when:	An error message has been written to the SQL Server error log with a severity level above a specified value
Configurable thresholds:	None

Possible causes:	Various Check the **SQL Server error log entry** area of the alert to see the error message text

Cluster failover

Raised when:	The active node for the cluster has changed
Configurable thresholds:	None
Possible causes:	The previously active node has failed or been manually switched to a different node.

Database state changed

Raised when:	Database has been created, removed, restored, or has changed state to: • offline • suspect • not recovered yet • in recovery • crashed while loading
Configurable thresholds:	None
Possible causes:	As above

CPU utilization unusual

Raised when:	Processor activity has exceeded or fallen below a specified percentage for longer than a specified duration Alert indicates whether utilization is unusually high or low
Configurable thresholds:	% CPU utilization considered high Duration of high CPU utilization (seconds) % CPU utilization considered low Duration of low CPU utilization (seconds)

Possible causes:

- Other processes running on the CPU - check the **System processes** area of the alert
- CPU-intensive SQL queries - if trace is turned on for the SQL Server, check the trace SQL statements in the **Performance snapshot** area of the alert

Note:

SQL Response only collects CPU usage data at 5-second intervals:

- the minimum value for the duration is 5 seconds
- when you configure the alert, change the duration value by 5 second increments

Managing alerts

For each type of alert, you can:

- disable it, so the alert will not be raised in future
- change the level at which it is raised, to either low, medium, or high
- change the thresholds that trigger the alert to be raised

You can edit the alert settings for a single SQL Server or across a number of SQL Servers at once (by editing a template (page 60)). For job-related, disk-related or database-related alerts, you can edit the alert for a specific job, disk or database.

When an alert is raised, you can change its settings by clicking **Edit Alert Configuration** in the alert details pane. To review the settings for alerts, or edit one or more alerts, click **Configuration** in the main toolbar.

List of recommendations

Recommendations are raised when certain conditions exist in any database on a monitored SQL Server instance.

SQL Response checks the status of these databases at set intervals and updates the list of active recommendations. Any recommendations that no longer apply are automatically removed.

SQL Response raises the following types of recommendation:

No backup exists for database

Raised when:	No entry for a backup exists in the backups system table
Configurable thresholds:	None
Default check:	Every 5 hours
Resolution:	Back up database
More information:	About database backups (page 79)

Full backup overdue

Raised when:	The backups system table does not contain an entry for a full backup, or the last full backup is older than a specified time
Configurable thresholds:	Time since last full backup (days)
Default check:	Every 5 hours
Resolution:	Run full backup on database
More information:	About full backups (page 78)

Log backup overdue

Raised when:	The backups system table does not contain a transaction log backup entry for this database, or the last transaction log backup is older than a specified time (Only raised for databases using the the full or bulk-logged recovery model.)
Configurable thresholds:	None

Default check:	Every 5 hours
Resolution:	Back up database
More information:	About log backups (page 83)

Integrity check overdue

Raised when:	The database error log file does not contain an entry for an integrity check, or the last integrity check is older than a specified time
Configurable thresholds:	Time since last integrity check (days)
Default check:	Every 5 hours
Resolution:	Run database integrity check
More information:	About integrity checks (page 80)

Fragmented index

Raised when:	The fragmentation of any index in the database with more than a specified number of pages exceeds a specified percentage
Configurable thresholds:	% fragmentation
	Minimum number of pages in index
Default check:	Every 5 hours
Resolution:	Rebuild index
More information:	About fragmented indexes (page 76)

Page verification is off

Raised when:	Torn Page Detection or Page Checksum (SQL Server 2005 only) is not enabled for the database
Configurable thresholds:	None
Default check:	Every 5 hours
Resolution:	Enable Torn Page Detection/Page Checksum
More information:	About page verification (page 86)

Transaction log has excessive free space

Raised when:	The percentage of the total transaction log size that is not utilized exceeds a specified value, and the log size is above a specified value
Configurable thresholds:	% free space in transaction log Minimum transaction log size (MB)
Default check:	Every 5 hours
Resolution:	Shrink transaction log to reclaim disk space
More information:	About free space in log files (page 89)

Data file has excessive free space

Raised when:	The percentage of free space in a database's data file exceeds a specified value, and the data file size is above a specified value
Configurable thresholds:	% free space in data file Miniumum data file size (MB)
Default check:	Every 5 hours
Resolution:	Shrink data file to reclaim disk space
More information:	About free space in data files (page 88)

Managing recommendations

For each type of recommendation, you can:

- disable it, so the recommendation will not be raised in future
- change the thresholds (for some recommendation types only)

You can edit the recommendation settings for a single SQL Server or across a number of SQL Servers at once (by editing a template).

When a recommendation is raised, you can change its settings (for example, to disable it in future) by clicking **Edit Recommendation Configuration** in the recommendation details pane. To review the settings for recommendations, or to edit one or more recommendations, click **Configuration** in the main toolbar.

Installation overview

SQL Response does not require any components to be installed on the SQL Servers you want to monitor. SQL Response comprises two main components:

- the **SQL Response client**, installed on your local computer

  The SQL Response client communicates with the Alert Repository to retrieve and display all alerts.

- the **Alert Repository**, typically installed on a network server

  The Alert Repository is a Windows service that continuously monitors your SQL Servers and generates alerts when problems occur.

To install and activate SQL Response

1. Install the SQL Response client.
2. Install the Alert Repository, and connect to it.

   The Alert Repository installer is part of the SQL Response client; when you have installed the SQL Response client, you will be prompted to install the Alert Repository. You need to connect to the Alert Repository before you can add SQL Servers to monitor.

   For more details, see Installing and connecting to the Alert Repository (page 18).
3. Add SQL Server instances to monitor.

   Select the SQL Servers you want to monitor, and set the properties for each.

   For more details, see Adding SQL Servers to monitor (page 27).
4. License SQL Servers for use.

   All SQL Servers you monitor are licensed for a 14 day trial period from the time you install the Alert Repository. To continue monitoring after this period, you need to apply a serial number to activate each SQL Server.

   For more details, see Activating SQL Servers in SQL Response (page 32).

Installing and connecting to the Alert Repository

When you first run SQL Response, you are prompted to install an Alert Repository. The Alert Repository is required by SQL Response to monitor SQL Servers.

What is the Alert Repository?

The Alert Repository is a Windows service that runs on a network computer; it communicates continuously with a list of specified SQL Server instances to collect information about their performance and store issues relating to job failures, performance problems and so on.

The Alert Repository service must be run under an administrator account; it also requires local administrator privileges on all SQL Servers it monitors.

The SQL Response client runs on your local computer and retrieves alert information from the Alert Repository for each monitored SQL Server instance. When the client is not running, the Alert Repository continues to monitor your SQL Servers and collect alert information.

Multiple SQL Response clients

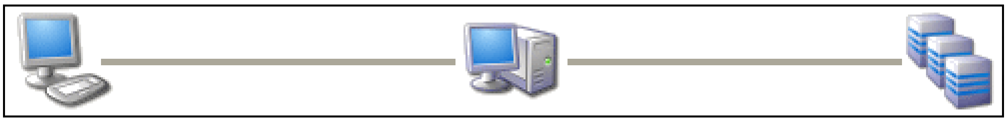
You can install multiple SQL Response clients connecting to a single Alert Repository. This allows several users to access the alerts raised on that Alert Repository. Any updates made by one client would be reflected in all other clients within a short period of time.

Multiple Alert Repositories

You can install several Alert Repositories; this may be useful if you have a large number of SQL Servers to monitor. We recommend that you monitor no more than 50 SQL Servers on a single Alert Repository. When using multiple Alert Repositories with a single client, you would have to disconnect from one Alert Repository and connect to the other in order to view alerts across all your monitored SQL Servers.

Where to install the Alert Repository

To monitor your SQL Servers, we recommend you install the Alert Repository on a network server.

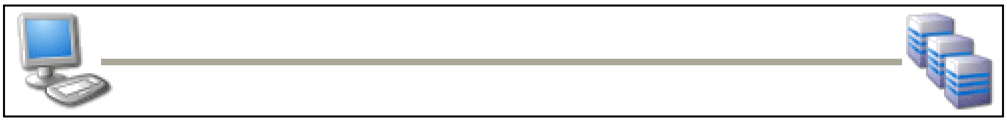


Local computer
SQL Response client

Server on network
Alert Repository

Monitored SQL
Servers

If you want to evaluate SQL Response by monitoring only a small number of servers, you can install the Alert Repository on your local computer. Note that if you turn this computer off, for example when you go home in the evening, monitoring will stop and no alerts will be raised by SQL Response during that period.



Local computer
SQL Response client and Alert Repository

Monitored SQL
Servers

Note: Windows 2000 is not a supported operating system for the computer running the Alert Repository.

Installing the Alert Repository on a network computer

1. If the **Alert Repository Setup** wizard is not displayed, from the **Alert Repository** menu, select **Alert Repository Setup**.
2. Select **Install a new Alert Repository**.
3. Select **On a server somewhere on your network**.
4. Click **Next**.

   To install on a network server, you need to copy an MSI installer file to the computer, and then log in to that computer to run the installer.
5. Click **Save Installer File to Server** and copy the Alert Repository installer (MSI) file to a folder on your network server.

   If you can connect remotely from your local computer, click **Launch Windows Remote Desktop** to connect to the server.
6. Run the installer file locally on the network computer.

The **SQL Response Alert Repository Setup** wizard is launched.

Installing the Alert Repository on the local computer

Alternatively, you can install the Alert Repository on the same computer as the SQL Response client. This may be suitable for evaluation purposes.

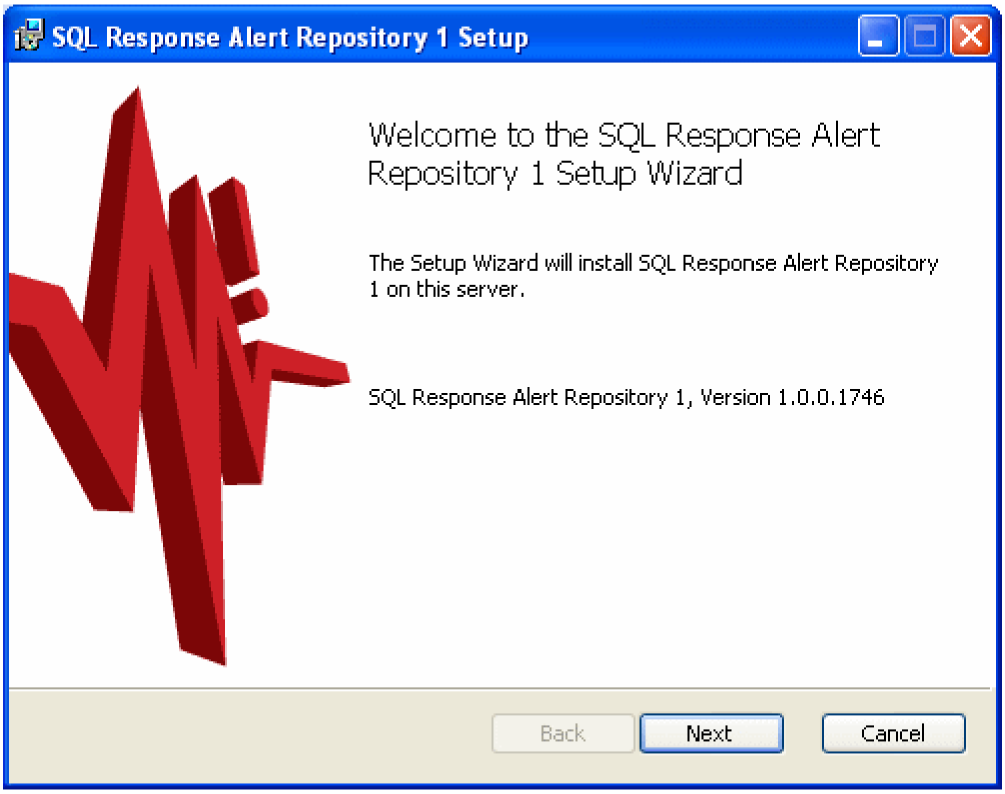
To do this:

1. In the **Alert Repository Setup** wizard, select to install **On this server**.
2. Click **Next**.

The **SQL Response Alert Repository Setup** wizard is launched.

Using the SQL Response Alert Repository Setup wizard

Follow the steps in the Wizard to specify the settings for installing and running the Alert Repository.



Welcome to the SQL Response Alert Repository 1 Setup Wizard

The Setup Wizard will install SQL Response Alert Repository 1 on this server.

SQL Response Alert Repository 1, Version 1.0.0.1746

Choosing a data store folder

Select a folder to store the alert files created by the Alert Repository.

Every time an alert is raised, one or more alert files are created containing details about the alert. These alert files are stored on the same computer that runs the Alert Repository service.

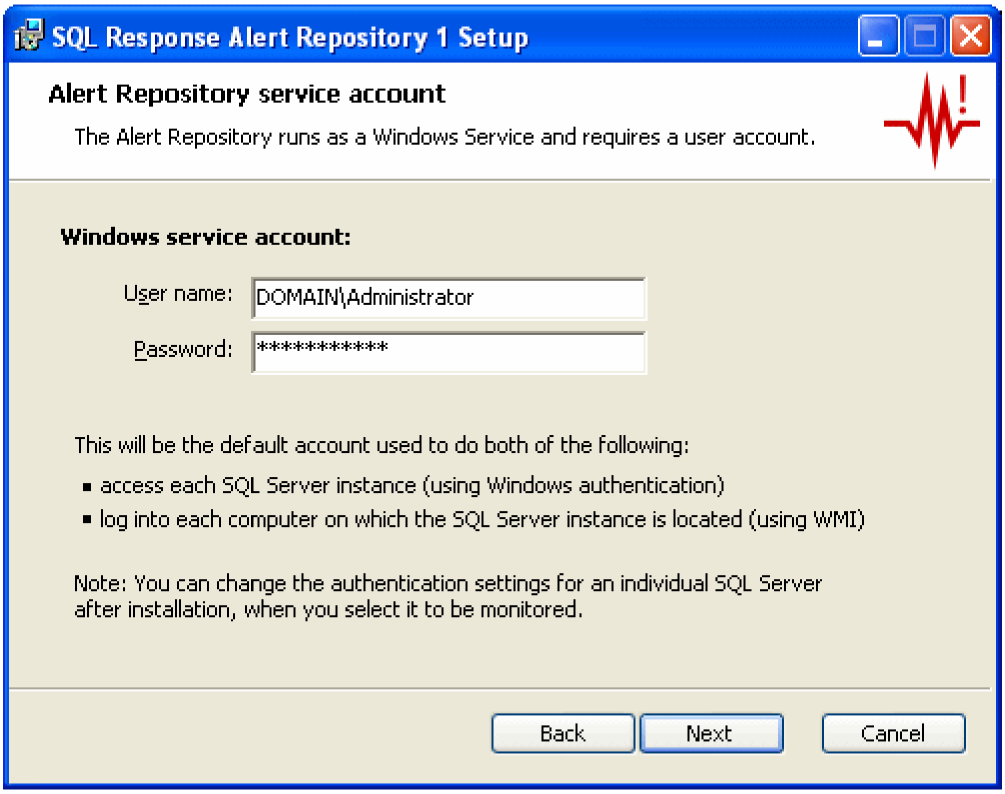
When SQL Response has been running for several days or weeks, the number of alert files stored in the data store folder could grow to be quite large. This takes up a lot of disk space, and may adversely affect the performance of SQL Response.

You should select a folder on a disk with as much space as possible. We recommend a minimum of 1GB per SQL Server instance being monitored.

Note: You can configure SQL Response to automatically delete alert files from disk after a specified time. See Managing the size of the Alert Repository data files on disk (page 91).

Specifying the Alert Repository service account

The Alert Repository service requires an administrator account, with administrator privileges on all the SQL Servers you want to monitor:



SQL Response Alert Repository 1 Setup

Alert Repository service account

The Alert Repository runs as a Windows Service and requires a user account.

Windows service account:

User name: DOMAIN\Administrator

Password: ************

This will be the default account used to do both of the following:

- access each SQL Server instance (using Windows authentication)
- log into each computer on which the SQL Server instance is located (using WMI)

Note: You can change the authentication settings for an individual SQL Server after installation, when you select it to be monitored.

When you connect to the Alert Repository and add SQL Servers to monitor in the SQL Response client, you will see that the Alert Repository service account is used by default to connect to each SQL Server and also to the computer hosting the SQL Server.

Enter the user name and password for the account, then click **Next**. SQL Response validates the credentials.

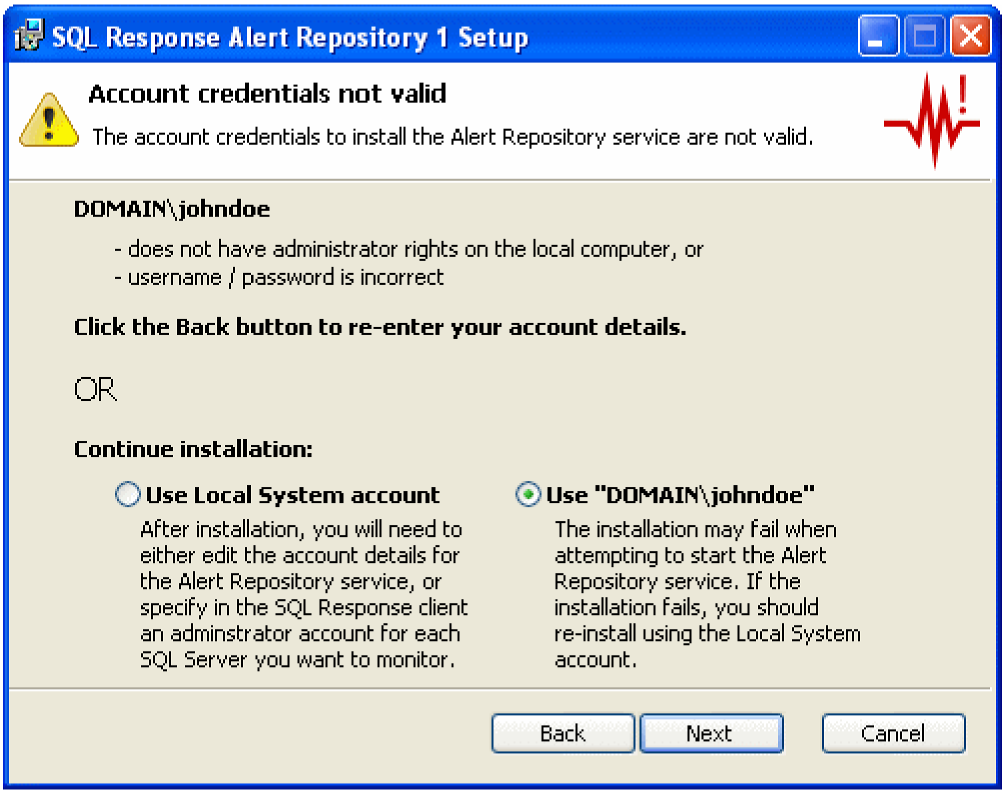
If you subsequently change the account name or password after SQL Response has been installed (for example, if this is an account that requires a change of password every six months), you will need to specify the new details in the properties of the Alert Repository service using Windows Services. You cannot update the credentials within the SQL Response interface.

Note: Windows 2000 is not a supported operating system for the computer running the Alert Repository. (Windows 2000 is still supported for the SQL Response client and for monitored SQL Servers.)

To subsequently specify a different account to log in to one of your monitored servers, see Changing the Windows login credentials for a specific computer (page 92).

Account credentials not valid?

When you enter the account credentials and click **Next**, SQL Response checks that the account you entered can be used to run the Alert Repository service. If the credentials are not valid, a page is displayed:



Account credentials not valid

The account credentials to install the Alert Repository service are not valid.

DOMAIN\johndoe

- does not have administrator rights on the local computer, or
- username / password is incorrect

Click the Back button to re-enter your account details.

OR

Continue installation:

○ **Use Local System account**
After installation, you will need to either edit the account details for the Alert Repository service, or specify in the SQL Response client an adminstrator account for each SQL Server you want to monitor.

● **Use "DOMAIN\johndoe"**
The installation may fail when attempting to start the Alert Repository service. If the installation fails, you should re-install using the Local System account.

Do one of the following:

- Click **Back** and enter a different account or password
- Continue the installation process using the **Local System account**

  This allows you to complete the installation of the Alert Repository. However, the Alert Repository service will not be able to connect to any SQL Servers you want to monitor using the Local System account; you will need to either:

  - change the account used to run the Alert Repository service after installation (use Windows Services to edit the properties of the service), or

  - specify an account to use to connect to each SQL Server when you add SQL Servers to monitor in the SQL Response client. See Adding SQL Servers to monitor (page 27).
- Continue the installation process using the account you specified

  This allows you to continue installing the Alert Repository. However, the installation may fail at its final step, when it attempts to start the Alert Repository using this account. If the installation fails, you will need to re-install the Alert Repository from the beginning, using a different account.

Choosing the TCP port

The Alert Repository service requires a TCP port to communicate with the SQL Response client.

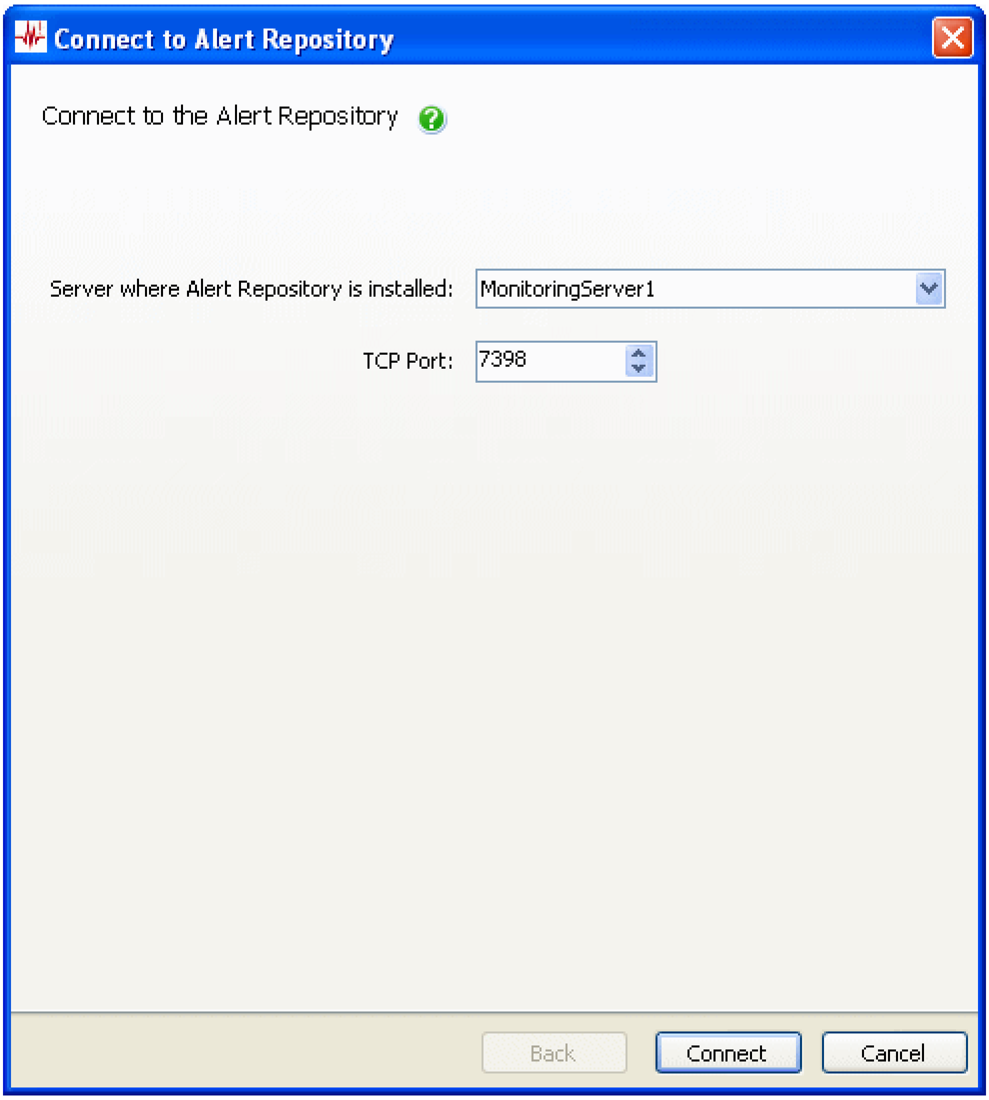
The default port number is 7398.

Choose a port number between 1 and 65535 that is not in use and allows incoming connections.

Connecting to the Alert Repository

When you have finished installing the Alert Repository, you need to connect to it from SQL Response:

- If you are still running the Alert Repository Setup wizard, click **Next** to choose the Alert Repository to connect to.
- Otherwise, from the **Alert Repository** menu in SQL Response, click **Connect to Alert Repository**.



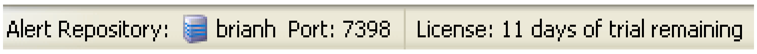
Connect to the Alert Repository

Server where Alert Repository is installed: MonitoringServer1

TCP Port: 7398

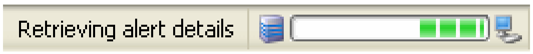
Type the name or IP address of the computer where you installed the Alert Repository, and click **Connect**.

The connection status to the Alert Repository is shown in the status bar at the bottom left of the application window. When you are connected to the Alert Repository, the name of the computer hosting the Alert Repository, the port number, and licensing status, is displayed:



Alert Repository: brianh Port: 7398 License: 11 days of trial remaining

Whenever SQL Response communicates with the Alert Repository for a specific purpose, for example to refresh the display of alerts or retrieve alert details, the task status is displayed in the status bar at the bottom right of the application window:



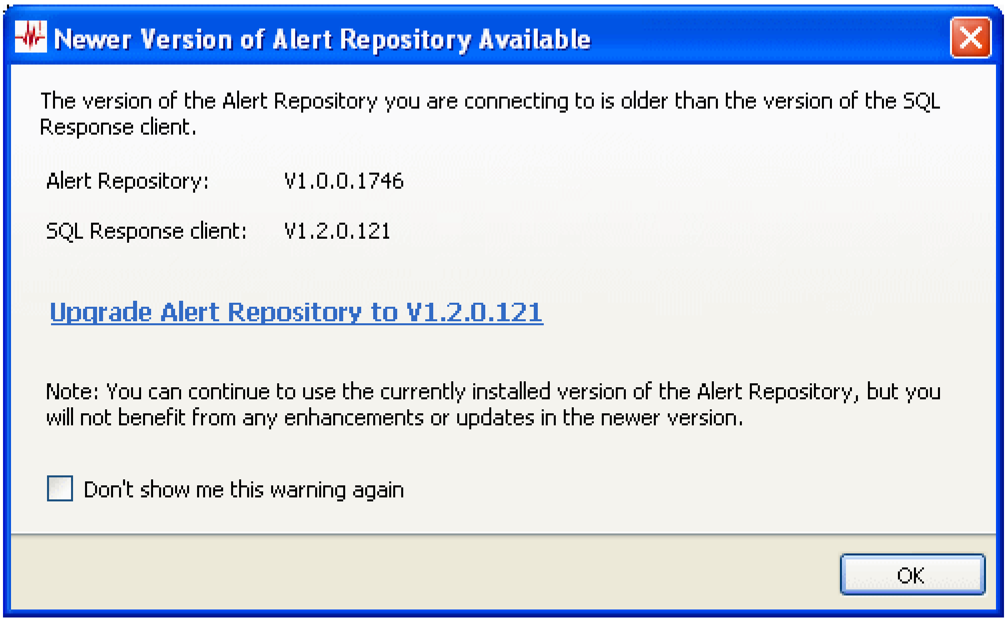
Retrieving alert details

When you have successfully connnected to the Alert Repository, you need to select which SQL Servers you want to monitor. See Adding SQL Servers to monitor (page 27).

Upgrading the Alert Repository

If you have installed a newer version of the SQL Response client, you should upgrade the version of your Alert Repository version to match the client. The installer files to upgrade the Alert Repository are automatically installed with the new version of the client.

You will be prompted to upgrade your Alert Repository to the new version when you next connect to the Alert Repository:



Newer Version of Alert Repository Available

The version of the Alert Repository you are connecting to is older than the version of the SQL Response client.

Alert Repository: V1.0.0.1746

SQL Response client: V1.2.0.121

Upgrade Alert Repository to V1.2.0.121

Note: You can continue to use the currently installed version of the Alert Repository, but you will not benefit from any enhancements or updates in the newer version.

□ Don't show me this warning again

Click the **Upgrade Alert Repository** link to start the Upgrade Alert Repository installation wizard. If you don't want to upgrade the Alert Repository, click **OK** to close the dialog box.

You can also upgrade at any time when you are connected to your Alert Repository by selecting **Upgrade Alert Repository** from the **Alert Repository** menu.

What happens when I upgrade?

All your existing alerts will remain available. No data is altered or removed.

Your current configuration will remain unchanged and you will be able to connect to the Alert Repository without changing your existing credentials.

Note: If you have several SQL Response clients and some have not been upgraded, these older versions of the client will still be able to connect to the newer version of the Alert Repository.

What happens if I don't upgrade?

SQL Response will still continue to work with the older version of the Alert Repository, but you will not benefit from any enhancements or bug fixes available in the newer version.

For a list of what's new in the latest version, see the release notes for the latest version (http://www.red-gate.com/supportcenter/Content.aspx?p=SQL%20Response&amp;c=SQL_Response%5carticles%5cversion_1xx_SQLResponse.htm).

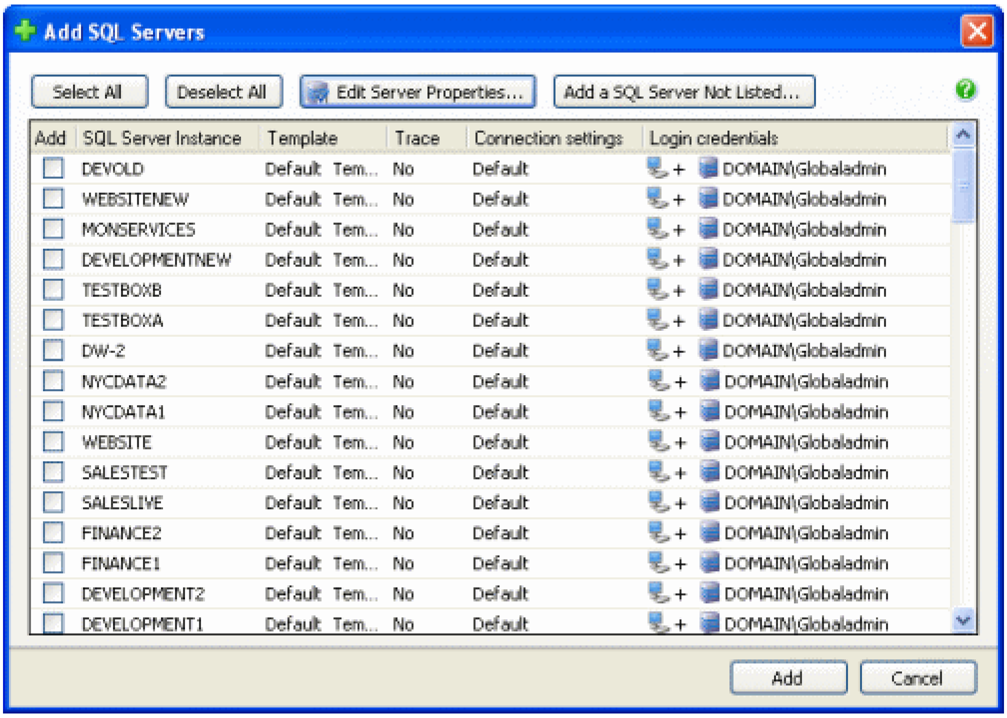
Adding SQL Servers to monitor

To monitor SQL Servers in SQL Response, you need to have installed and then connected to the SQL Response Alert Repository. If you are not connected to the Alert Repository, see Installing and connecting to the Alert Repository (page 18).

Selecting the SQL Servers to monitor

1. Click ✚ **Add SQL Servers**.

   The **Add SQL Servers** dialog box is displayed.



Add	SQL Server Instance	Template	Trace	Connection settings	Login credentials
□	DEVOLD	Default Tem...	No	Default	+ DOMAIN\Globaladmin
□	WEBSITENEW	Default Tem...	No	Default	+ DOMAIN\Globaladmin
□	MONSERVICES	Default Tem...	No	Default	+ DOMAIN\Globaladmin
□	DEVELOPMENTNEW	Default Tem...	No	Default	+ DOMAIN\Globaladmin
□	TESTBOXB	Default Tem...	No	Default	+ DOMAIN\Globaladmin
□	TESTBOXA	Default Tem...	No	Default	+ DOMAIN\Globaladmin
□	DW-2	Default Tem...	No	Default	+ DOMAIN\Globaladmin
□	NYCDATA2	Default Tem...	No	Default	+ DOMAIN\Globaladmin
□	NYCDATA1	Default Tem...	No	Default	+ DOMAIN\Globaladmin
□	WEBSITE	Default Tem...	No	Default	+ DOMAIN\Globaladmin
□	SALESTEST	Default Tem...	No	Default	+ DOMAIN\Globaladmin
□	SALESLIVE	Default Tem...	No	Default	+ DOMAIN\Globaladmin
□	FINANCE2	Default Tem...	No	Default	+ DOMAIN\Globaladmin
□	FINANCE1	Default Tem...	No	Default	+ DOMAIN\Globaladmin
□	DEVELOPMENT2	Default Tem...	No	Default	+ DOMAIN\Globaladmin
□	DEVELOPMENT1	Default Tem...	No	Default	+ DOMAIN\Globaladmin

   The **Add SQL Servers** dialog box shows all the SQL Server instances that are:

   - in the same domain as the Alert Repository, and
   - in the same IP range as the Alert Repository, and
   - not yet added to SQL Response

2. In the **Add** column, select the check box next to each SQL Server instance you want to monitor.

   If the instance you want to monitor is not listed (for example if it is in a different domain or a sub-domain), click **Add a SQL Server Not Listed** and type the full path to the SQL Server instance. It is added to the top of the list and selected.

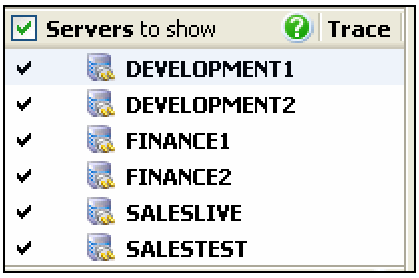
3. Before you click **Add** to start monitoring the selected SQL Servers, you can change the following settings for each SQL Server:

- log-in credentials
- whether SQL Profiler trace is enabled
- configuration template
- connection settings, such as connection time out value

For more information about changing any of these settings, see the sections below.

4. Click the **Add** button to start monitoring all the selected SQL Servers.

   The list of monitored instances is displayed in the **Servers to show** area of the Filter pane on the left of the main application window:

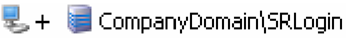


☑ Servers to show	Trace
✓ DEVELOPMENT1	
✓ DEVELOPMENT2	
✓ FINANCE1	
✓ FINANCE2	
✓ SALESLIVE	
✓ SALESTEST	

   When you first add SQL Servers to monitor, the icon next to the SQL Server name indicates that SQL Response is attempting to connect and log in to that instance. If the connection or login is unsuccessful, the icon will be updated. For an explanation of SQL Server status icons, see Monitoring the status of SQL Servers (page 40) .

Changing the credentials used to log in to a SQL Server instance

The account listed under **Login credentials** in the **Add SQL Servers** dialog box is, by default, the Alert Repository service account that was entered during the Alert Repository installation process. This account is used to connect to both the SQL Server (for collecting SQL-related data) and also to the computer hosting the SQL Server (for checking drive space and general server issues).



+ CompanyDomain\SRLogin

Indicates that the same credentials are used to log in to the computer and to connect to SQL Server on this instance.

Changing the login credentials for a single SQL Server instance

1. Click the SQL Server to select it in the list.
2. Click **Edit Server Properties**.
3. In the Server Properties dialog box, under **Login credentials**, you can:
   - Specify a different Windows account to log in to the computer hosting the SQL Server (on the left side).

   If you have several SQL Servers running on the same computer, the same credentials will be used for each SQL Server.

   - Use SQL Server authentication instead of Windows authentication for the SQL Server login (on the right side)

Note: You cannot change the Windows authentication settings for SQL Server login in this dialog box. These credentials are specified when you install the Alert Repository, and are used for connecting to all monitored SQL Servers. To change the Alert Repository service account, you will need to edit the properties of the Alert Repository Windows service to run it using a different account.

Changing the login credentials for multiple SQL Server instances

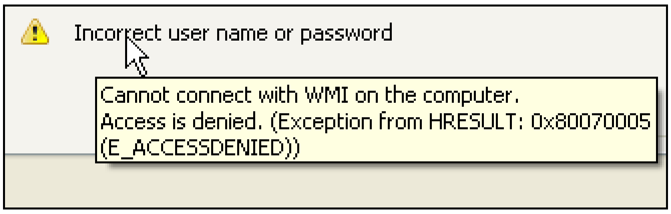
You can change the credentials for multiple SQL Servers at the same time by selecting several SQL Servers in the list. The names of the SQL Servers you are editing are listed in the title bar of the **Add SQL Servers** dialog box.

Testing the login settings

Click **Test Credentials** to attempt to connect to the SQL Server(s) using the account settings you entered. SQL Response displays a message beneath the relevant account if it cannot connect using the specified details.

When editing the settings for multiple SQL Servers, SQL Response will stop testing the credentials after the first connection failure. The SQL Servers are tested in the order in which you selected them from the **Add SQL Servers** dialog box; this order is listed in the title bar.

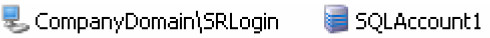
Tip: When SQL Response returns an error message after the credentials test, you can view more information about the problem by moving your mouse pointer over the message to view a tooltip:



Incorrect user name or password

Cannot connect with WMI on the computer.
Access is denied. (Exception from HRESULT: 0x80070005 (E_ACCESSDENIED))

When you have changed the default credentials for the computer login or the SQL Server login, the entered credentials are shown separately in the **Add SQL Servers** dialog box:



CompanyDomain\SRLogin SQLAccount1

Shows both the computer login credentials and the SQL Server authentication

Enabling SQL Profiler trace data

Enabling SQL Profiler trace data on a monitored SQL Server means that SQL Response continuously collects trace data from that server. When an alert on that SQL Server is raised, trace data for the period immediately before the alert is displayed as part of the alert details. This allows you to inspect in detail what was happening on that Server at the time the alert was raised, and can help you to diagnose the problem.

To enable the collection of SQL profiler trace data on a SQL Server:

1. In the **Add SQL Servers** dialog box, click to select the SQL Server in the list.

2. Click **Edit Server Properties**
3. In the Server properties dialog box, select **Enable collection of trace data**.

Note: We recommend that you enable trace on a maximum of eight to ten SQL Servers, as the continuous collection of trace data can lead to performance issues. SQL Response will still raise alerts on SQL Servers that do not have trace enabled, and performance data other than Profiler trace is provided as part of the alert details.

For more details, see Collecting SQL Profiler trace data (page 38).

Choosing a configuration template

Choosing a configuration template is an optional step. If you do not select a template, SQL Response will use the Default Template for added SQL Servers. The Default Template contains the settings and thresholds for each type of alert and recommendation that SQL Response raises.

Once you have added the SQL Servers to monitor, you can edit the Default Template or create new templates. See Working with templates (page 60).

Changing the connection settings

1. 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.
2. In **Network packet size**, type or select the size of the network packets to be sent.

   The default value is 4096 bytes.
3. 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.
4. In **Execution time-out**, type or select the number of seconds to wait before execution of a task is stopped.

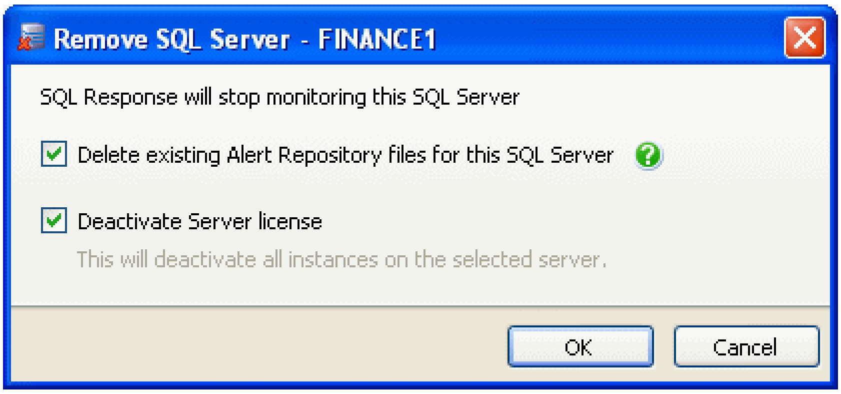
   The default value is zero seconds (no time-out).

To force the connection to be encrypted, select the **Encrypt connection** check box.

Removing a SQL Server

To stop monitoring a SQL Server, right-click it in the **Servers to show** area of the filter pane, and click **Remove SQL Server**.

You can also press the Delete key to remove the currently selected SQL Server.



Remove SQL Server - FINANCE1

SQL Response will stop monitoring this SQL Server

☑ Delete existing Alert Repository files for this SQL Server

☑ Deactivate Server license

This will deactivate all instances on the selected server.

Select **Delete existing Alert Repository files for this SQL Server** to permanently remove all the alert files for this SQL Server stored on disk on the Alert Repository computer. This frees up space on the Alert Repository computer and can improve the performance of SQL Response.

Clear this option only if you intend to start monitoring this SQL Server again in the future, and want to preserve all the historical alerts.

Note: There may be a slight delay deleting the files from the Alert Repository data folder when the SQL Server is removed. It can take up to ten minutes before the files are deleted.

Select **Deactivate Server license** to deactivate SQL Response on this SQL Server instance, and on any other instances hosted on the same computer. If you are deactivating a SQL Server instance that is part of a cluster, you will be prompted to deactivate all the other nodes in the cluster.

For more information, see Deactivating your products (http://www.red-gate.com/Products/Licensing/v_2/DeactivationHelp).

Activating SQL Response

When you download and install a product from Red Gate Software, you have a 14 day trial period in which you can evaluate the product. When your trial period expires, you are invited to purchase the product. If you need more time to evaluate the product, contact licensing@red-gate.com (mailto:licensing@red-gate.com).

When you purchase the product, you are sent an invoice that contains your serial number. You use the serial number to activate the product. If you cannot find your invoice, you can review your serial numbers at http://www.red-gate.com/myserialnumbers (http://www.red-gate.com/myserialnumbers).

Licensing SQL Server instances in SQL Response

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

Before activating a SQL Server instance:

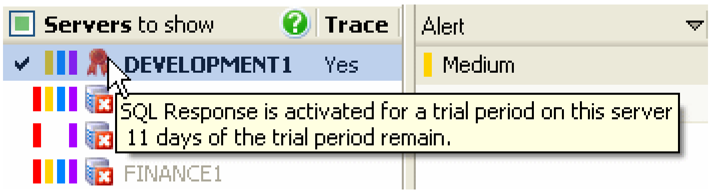
1. If you have not done so already, install the Alert Repository (page 18).
2. Connect to the Alert Repository.
3. Add the SQL Server (page 27) instance you want to monitor.

The status of SQL Response on a SQL Server instance is displayed in the Servers to show (page 42) area in the Filter pane, at the top left of the application window. You can activate SQL Response on a SQL Server if one of the following icons is displayed to the right of the name of the SQL Server instance:

- SQL Response is activated for a trial period on all unlicensed SQL Servers.
- The SQL Response trial period has expired for all unlicensed SQL Servers. To monitor this SQL Server instance using SQL Response, you must activate this SQL Server.

Note: You cannot activate a SQL Server instance until SQL Response can successfully connect to the SQL Server. If you have a connection error, indicated by or next to the instance name then you will not be able to activate this SQL Server.

To see how many days you have left in the trial period, move the mouse pointer over any of the unlicensed SQL Servers in the Servers to show area:



SQL Response is activated for a trial period on this server
11 days of the trial period remain.

The status bar at the bottom of the application window indicates how many days of the trial period remain for all unlicensed SQL Servers, or indicates that the trial period has expired.

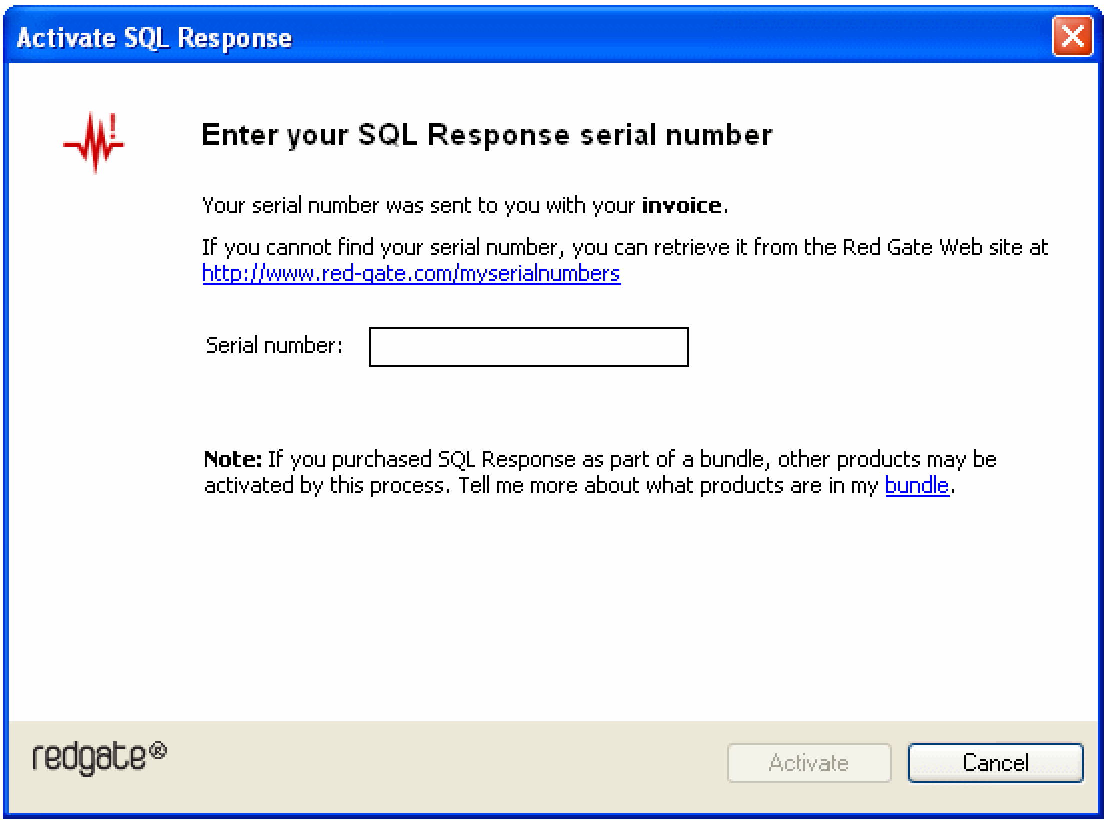
The trial period starts from the time you install the Alert Repository service.

Note: If you have installed multiple Alert Repositories, then you cannot share licenses between them. You will need a new license for any SQL Server you want to monitor using a different Alert Repository.

To activate a SQL Server

1. Right-click the name of the SQL Server instance and click **Activate Server License**.

   The product activation dialog box is displayed:



Activate SQL Response

Enter your SQL Response serial number

Your serial number was sent to you with your **invoice**.

If you cannot find your serial number, you can retrieve it from the Red Gate Web site at http://www.red-gate.com/myserialnumbers

Serial number:

Note: If you purchased SQL Response as part of a bundle, other products may be activated by this process. Tell me more about what products are in my bundle.

2. Enter your serial number and click **Activate**.

   Your activation request is sent to the Red Gate activation server.

   When your activation has been confirmed, the **Activation successful** page is displayed.

   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 Licensing and activation errors (/support/page?c&#61;all_products%5carticles%5clicensing_error_messages.htm). Depending on the error, you may want to try manual activation (page 34).

3. Click **Close**.

   You can now continue to monitor the SQL Server instance in SQL Response.

Note: The serial number you enter licenses all the SQL Server instances that run on the same computer.

Deactivating a SQL Server

Right-click the SQL Server instance and click **Deactivate Server License.**

If you have more than one instance on the same computer, then all the instances on that computer will be deactivated.

Re-activating SQL Servers

If you need to re-install the product on the same computer, for example following installation of a new operating system, you can re-activate the product using the same serial number. If you need to change the computer on which the product is installed and your current serial number does not work, contact licensing@red-gate.com (mailto:licensing@red-gate.com).

Activating and deactivating SQL Servers on a cluster

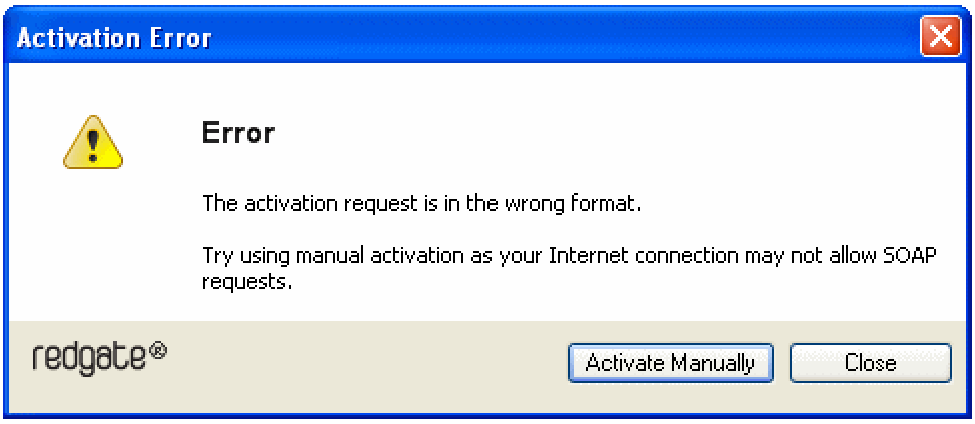
To activate a cluster, you will be prompted to enter a serial number for each node in the cluster. For example, if you have four nodes, each on a separate computer, the activation screen will be displayed four times. You will need a license for each node.

Similarly, when you deactivate a cluster, you will be prompted to deactivate once for each separate node.

Manual activation

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

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



Activation Error

Error

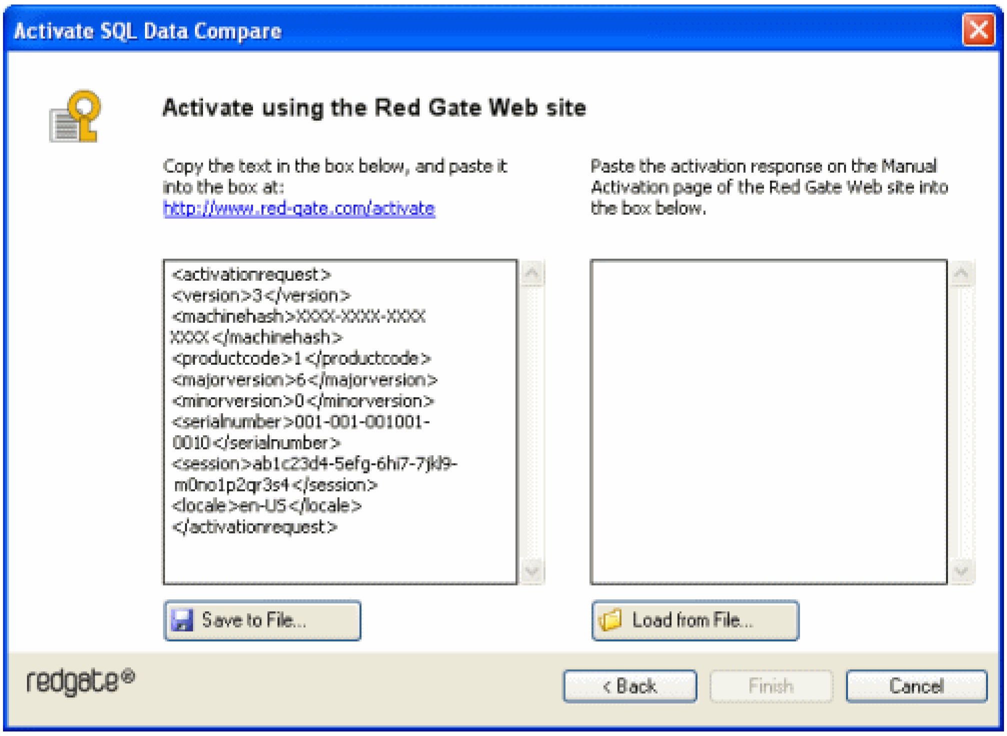
The activation request is in the wrong format.

Try using manual activation as your Internet connection may not allow SOAP requests.

To activate manually:

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

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



Activate SQL Data Compare

Activate using the Red Gate Web site

Copy the text in the box below, and paste it into the box at: http://www.red-gate.com/activate

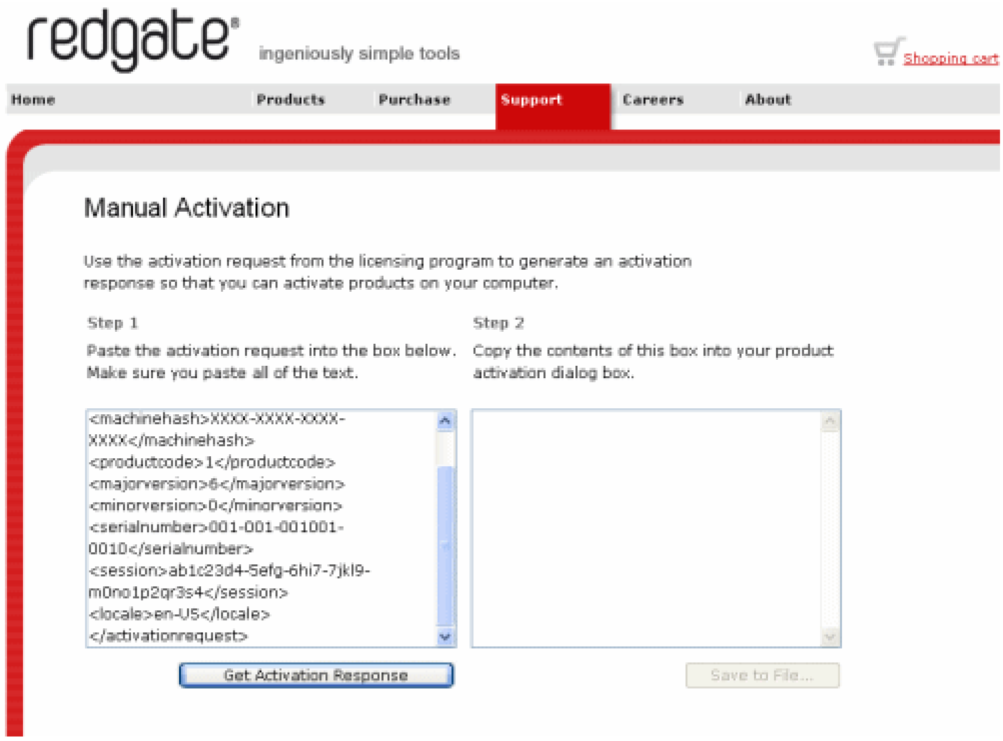
Paste the activation response on the Manual Activation page of the Red Gate Web site into the box below.

```
<activationrequest>
<version>3</version>
<machinehash>XXXX-XXXX-XXXX
XXXX</machinehash>
<productcode>1</productcode>
<majorversion>6</majorversion>
<minorversion>0</minorversion>
<serialnumber>001-001-001001-
0010</serialnumber>
<session>ab1c23d4-5efg-6hi7-7jkl9-
m0no1p2qr3s4</session>
<locale>en-US</locale>
</activationrequest>
```

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. 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 (http://www.red-gate.com/activate) and paste the activation request into the box under **Step 1**.



Manual Activation

Use the activation request from the licensing program to generate an activation response so that you can activate products on your computer.

Step 1

Paste the activation request into the box below. Make sure you paste all of the text.

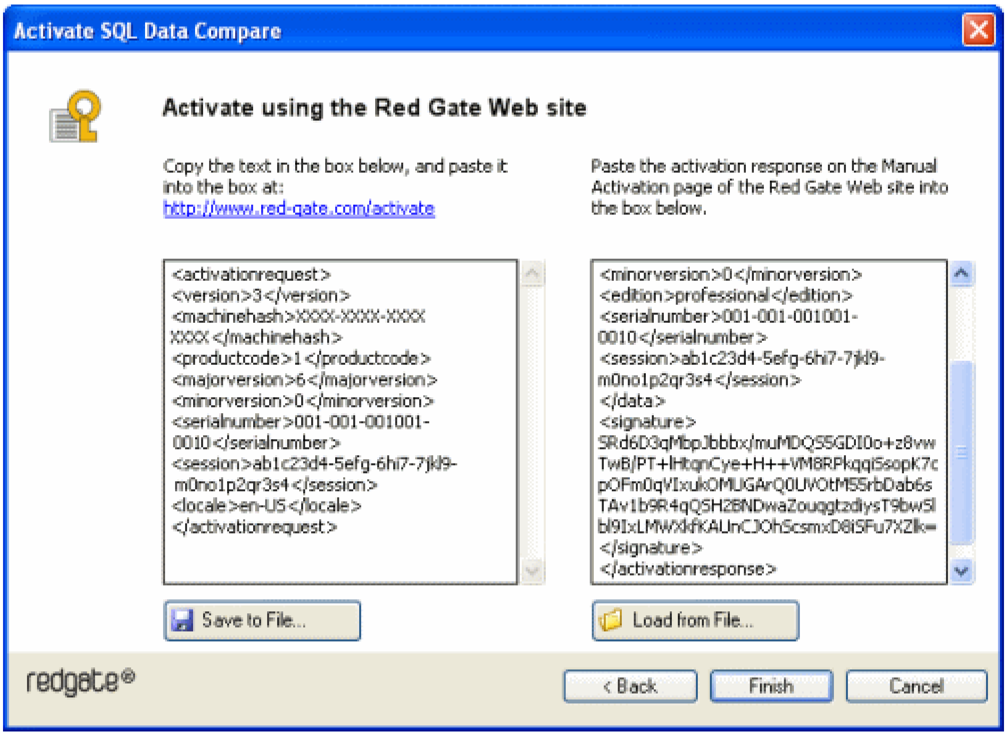
```
<machinehash>XXXX-XXXX-XXXX-
XXXX</machinehash>
<productcode>1</productcode>
<majorversion>6</majorversion>
<minorversion>0</minorversion>
<serialnumber>001-001-001001-
0010</serialnumber>
<session>ab1c23d4-5efg-6hi7-7jkl9-
m0no1p2qr3s4</session>
<locale>en-US</locale>
</activationrequest>
```

Step 2

Copy the contents of this box into your product activation dialog box.

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.



Activate SQL Data Compare

Activate using the Red Gate Web site

Copy the text in the box below, and paste it into the box at: http://www.red-gate.com/activate

```
<activationrequest>
<version>3</version>
<machinehash>XXXX-XXXX-XXXX
XXXX</machinehash>
<productcode>1</productcode>
<majorversion>6</majorversion>
<minorversion>0</minorversion>
<serialnumber>001-001-001001-
0010</serialnumber>
<session>ab1c23d4-5efg-6hi7-7jkl9-
m0no1p2qr3s4</session>
<locale>en-US</locale>
</activationrequest>
```

Paste the activation response on the Manual Activation page of the Red Gate Web site into the box below.

```
<minorversion>0</minorversion>
<edition>professional</edition>
<serialnumber>001-001-001001-
0010</serialnumber>
<session>ab1c23d4-5efg-6hi7-7jkl9-
m0no1p2qr3s4</session>
</data>
<signature>
SRd6D3qMbpJbbbx/muMDQS5GDI0o+z8vw
TwB/PT+lHtqnCye+H++VM8RPkqqiSsopK7c
pOFm0qVIxukOMUGArQ0UVOtM55rbDab6s
TAv1b9R4qQSH2BNDwaZouqgtzdiysT9bw5l
bl9IxLMWXkfKAUnCJOhScsmxD8iSFu7XZlk=
</signature>
</activationresponse>
```

7. Click **Finish**.

   The **Activation successful** page is displayed.

8. Click **Close**.

   You can now continue to use your product.

Collecting SQL Profiler trace data

SQL Response can continuously collect SQL Profiler trace data on selected SQL Servers you are monitoring.

Enabling the collection of trace data allows you to review the key SQL statements that were executing around the time of a raised alert; captured SQL Profiler data from just before and after the alert is displayed as part of the diagnostic data provided with each raised alert on the Server.

Setting up SQL Profiler trace collection on a SQL Server

In the **Trace** column of the **Servers to show** area in the Filter pane, SQL Servers that are collecting trace data are labelled *Yes*.

To turn on the collection of SQL Profiler trace data for a SQL Server:

1. Right-click on the SQL Server in the **Servers to show** area of the Filter pane, and click **Server properties.**
2. In the SQL Server Properties dialog box, select **Enable collection of trace data**.

Alerts that are raised on that SQL Server from this point on will include trace data in the **Performance snapshot** section of the alert details pane. See Viewing performance snapshot data (page 51).

If you turn off the collection of SQL Profiler trace data for a SQL Server, this will only affect alerts that are raised from that point on. Existing raised alerts will still include the trace data.

For a short period after you enable trace on a SQL Server, the server icon in the **Servers to show** area indicates that SQL Response is attempting to connect to that SQL Server. SQL Response needs to reinitialize the connection to the SQL Server when you turn on collection of SQL Profiler trace data.

What trace data is collected?

The following trace information is collected (SQL Server event number in brackets) :

RPC_Starting	(11)
SQL_BatchStarting	(13)
Audit_Login	(14)
Attention	(16)
SQL Exception	(33)

Note: The Profiler trace events that are collected are hard-wired in SQL Response, and cannot be customised.

Effect of running trace

Enabling the collection of SQL Profiler trace data on a SQL Server will increase processor activity on that SQL Server (typically, by up to 5%). Enabling collection of trace data on numerous computers can therefore reduce performance of the Alert Repository.

For that reason, we recommend that you enable trace data only on problematic or important SQL Servers.

Where are trace data files stored?

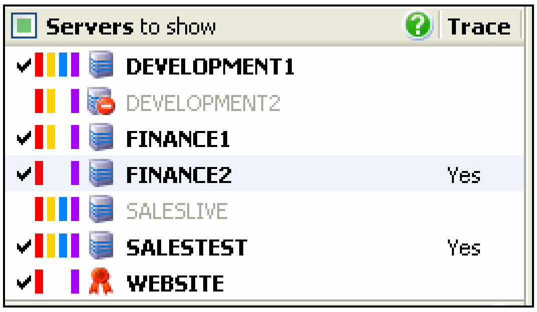
When SQL Profiler trace data is collected on a SQL Server, it is stored in the log folder for that SQL Server. SQL Response then collects this trace data from each monitored SQL Server at intervals of one minute, and stores it in the data folder on the Alert Repository server. About 20 minutes worth of trace data is stored in the data folder on the Alert Repository at any time; this requires roughly 1 gigabyte of disk space. Older trace files are automatically deleted.

What happens if I turn off collection of trace data?

SQL Response provides some diagnostic information for each alert even when the collection of trace data is turned off. CPU utilization, memory and disk performance, and data from various other SQL-based counters are still collected.

Monitoring the status of SQL Servers

You can see all the SQL Servers you are currently monitoring and the connection status of each SQL Server in the **Servers to show** area in the Filter pane, at the top left of the application window:



	Servers to show	Trace
✓	DEVELOPMENT1	
	DEVELOPMENT2	
✓	FINANCE1	
✓	FINANCE2	Yes
	SALESLIVE	
✓	SALESTEST	Yes
✓	WEBSITE	

If the Filter pane is not displayed, from the **View** menu, select **Filter pane**.

Yes in the **Trace** column indicates that SQL Response is collecting SQL Profiler trace data (page 38) on that SQL Server.

A check mark next to a SQL Server name indicates that alerts or recommendations are being displayed for this SQL Server. See Viewing and filtering alerts and recommendations (page 42).

Status of monitored SQL Servers

The icon next to a SQL Server name indicates the current connection status and licensing status of the monitored SQL Server:

SQL Response is activated on this SQL Server, and can connect to it.

SQL Response is activated for a trial period on this SQL Server.

Move the mouse pointer over the icon to see the number of days left in the trial.

The SQL Response license on this SQL Server has expired.

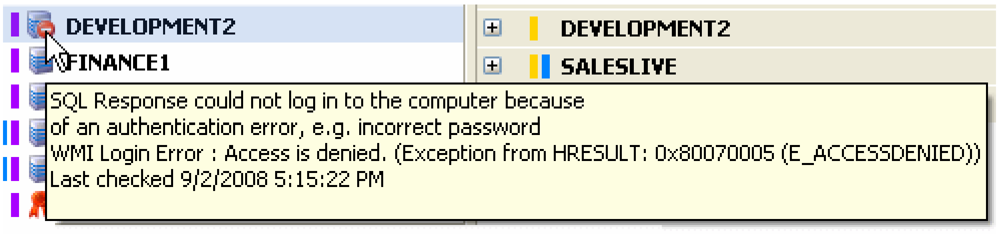
To monitor this SQL Server with SQL Response, you must activate SQL Response (page 32).

SQL Response cannot connect to the SQL Server or to the computer, because of an authentication error such as an incorrect password.

SQL Response cannot connect to the SQL Server as the computer is unreachable or the SQL Server instance is not running.

SQL Response is connecting to the SQL Server for the first time after it has been added to the list of SQL Servers to monitor.

Move the mouse pointer over the icon next to a SQL Server name to view a tooltip containing more detailed information about the connection or licensing status of that SQL Server:



SQL Response could not log in to the computer because
of an authentication error, e.g. incorrect password
WMI Login Error : Access is denied. (Exception from HRESULT: 0x80070005 (E_ACCESSDENIED))
Last checked 9/2/2008 5:15:22 PM

Alert bars on monitored SQL Servers

The colored alert bars next to a SQL Server name indicate whether there are any uncleared high level, medium level or low level alerts, or any recommendations, on this SQL Server:

🟥 High level alert

🟨 Medium level alert

🟦 Low level alert

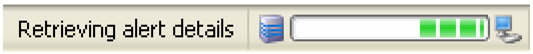
🟪 Recommendation

The level at which an alert is raised is part of the alert's configuration, which you can change. If you have not edited the Default Template, the level at which each alert has been raised is determined by the configuration for those alert types in the Default Template. See Working with templates (page 60).

Note: the existence of cleared alerts on a SQL Server is not indicated by an alert bar, but cleared alerts can still be viewed. See Clearing alerts and adding comments (page 55).

How can I see what SQL Response is doing?

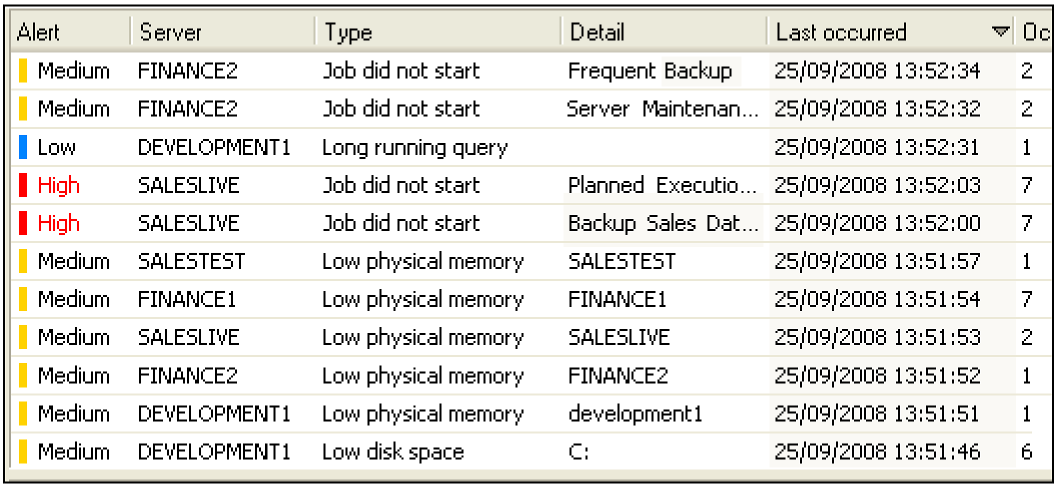
Whenever SQL Response communicates with the Alert Repository for a specific task, for example to refresh the display of alerts or retrieve alert details, the task status is displayed in the status bar at the bottom right of the application window:



Retrieving alert details

Viewing and filtering alerts and recommendations

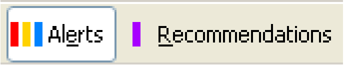
When SQL Response has been running for a short period, you should start to see a number of alerts being raised in the main list.



Alert	Server	Type	Detail	Last occurred ▽	Oc
🟨 Medium	FINANCE2	Job did not start	Frequent Backup	25/09/2008 13:52:34	2
🟨 Medium	FINANCE2	Job did not start	Server Maintenan...	25/09/2008 13:52:32	2
🟦 Low	DEVELOPMENT1	Long running query		25/09/2008 13:52:31	1
🟥 High	SALESLIVE	Job did not start	Planned Executio...	25/09/2008 13:52:03	7
🟥 High	SALESLIVE	Job did not start	Backup Sales Dat...	25/09/2008 13:52:00	7
🟨 Medium	SALESTEST	Low physical memory	SALESTEST	25/09/2008 13:51:57	1
🟨 Medium	FINANCE1	Low physical memory	FINANCE1	25/09/2008 13:51:54	7
🟨 Medium	SALESLIVE	Low physical memory	SALESLIVE	25/09/2008 13:51:53	2
🟨 Medium	FINANCE2	Low physical memory	FINANCE2	25/09/2008 13:51:52	1
🟨 Medium	DEVELOPMENT1	Low physical memory	development1	25/09/2008 13:51:51	1
🟨 Medium	DEVELOPMENT1	Low disk space	C:	25/09/2008 13:51:46	6

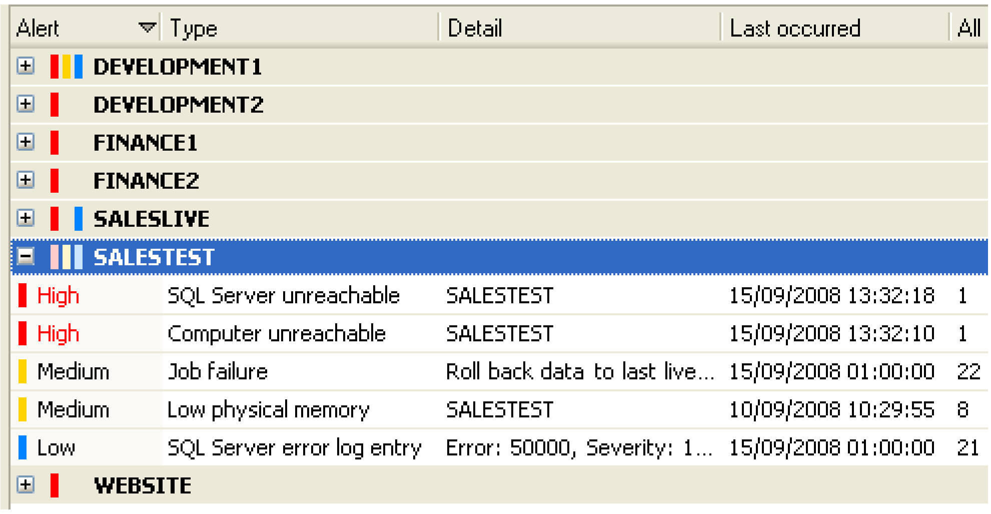
Note: All the examples in this topic refer to alerts; the list of recommendations can also be grouped, sorted, and filtered in the same way.

Switch between viewing alerts and recommendations by using the buttons in the main toolbar:



Grouping alerts by SQL Server

Select the **Group by server** check box on the main toolbar to collapse the list of raised alerts into SQL Server groups.



Alert	Type	Detail	Last occurred	All
⊞ DEVELOPMENT1				
⊞ DEVELOPMENT2				
⊞ FINANCE1				
⊞ FINANCE2				
⊞ SALESLIVE				
⊟ SALESTEST				
High	SQL Server unreachable	SALESTEST	15/09/2008 13:32:18	1
High	Computer unreachable	SALESTEST	15/09/2008 13:32:10	1
Medium	Job failure	Roll back data to last live...	15/09/2008 01:00:00	22
Medium	Low physical memory	SALESTEST	10/09/2008 10:29:55	8
Low	SQL Server error log entry	Error: 50000, Severity: 1...	15/09/2008 01:00:00	21
⊞ WEBSITE				

To view the alerts in a group, click ⊞, or double-click the group heading. Colored alert bars in the group heading indicate whether there are any alerts of high, medium or low level for that SQL Server, with the current selection of filters (including the current timeframe).

When there are no alerts to display for a SQL Server, given the current selection of filters, that SQL Server group is not displayed.

You can sort by column, within a group. Each expanded group is sorted separately.

Sorting alerts

Click a column header to sort the list of alerts by that column.

Filtering alerts

You can filter the list of alerts by selecting or clearing check marks for items in the Filter pane.

✓ **FINANCE1** - alerts displayed

FINANCE1 - alerts filtered

You can select any combination of filters.

To filter alerts by:	Do this:
SQL Server	Select or clear check marks in the **Servers to show** area.
Alert level	Select or clear check marks in the **Alert levels to show** area.

Type of alert

Select or clear check marks in the following areas:

Server alerts to show

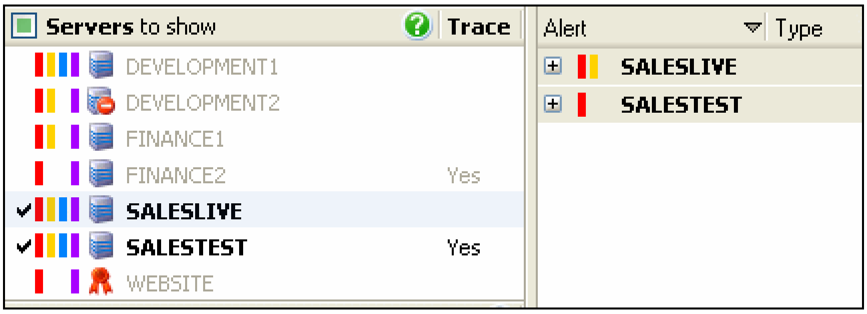
Job alerts to show

Process alerts to show

Timeframe

Select the time in the **From the last** drop-down list in the main toolbar

The alert bars next to the SQL Server name in the **Servers to show** area are unaffected by any current filters that may be applied, including the timeframe filter.



Servers to show	Trace
DEVELOPMENT1	
DEVELOPMENT2	
FINANCE1	
FINANCE2	Yes
✓ SALESLIVE	
✓ SALESTEST	Yes
WEBSITE	

Alert	Type
⊞ SALESLIVE	
⊞ SALESTEST	

In the above example, the alert bars in the Filter pane on the left indicate there are uncleared alerts at high, medium and low level on the two selected SQL Servers.

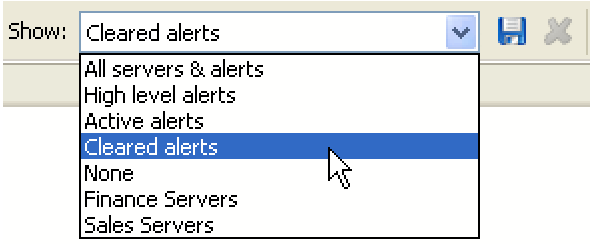
The alert bars in the list on the right reflect the current selection of filters, including the timeframe filter, and so indicate that there are just high and medium level alerts to view.

Note: switching the view from alerts to recommendations may change the selection of SQL Server filters; the SQL Server filters are saved separately for alerts and recommendations.

Using views

A view saves the current state of the Filter pane, that is, which check marks are selected. When you select the view from the **Show** box at a later date, these filters are instantly reapplied. You maintain one set of views for filtering alerts and another for filtering recommendations.

SQL Response has several predefined views available in the **Show** list, but you can create your own custom views. The predefined views cannot be deleted.



Show: Cleared alerts

All servers & alerts
High level alerts
Active alerts
Cleared alerts
None
Finance Servers
Sales Servers

Note: The default views such as *All servers & alerts* or *High level alerts* always select all the monitored SQL Servers.

To save a set of filters as a view:

1. In the Filter pane, select the filters you want to save You can choose any combination of filters, for example, to show only high level Job type alerts on a specifed SQL Server, or to show all Blocked SQL process alerts across all your monitored SQL Servers.

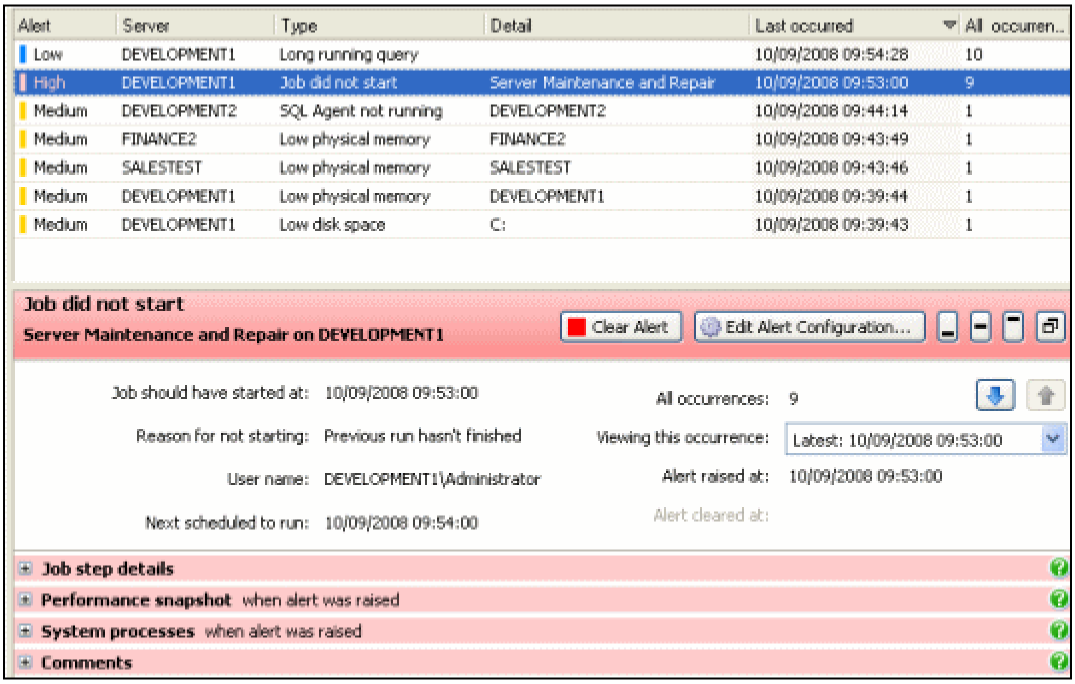
   The **Show** list shows the last selected view with an asterisk, indicating that the selection of filters has been modified.
2. Click the save view button 💾 in the main toolbar.
3. In the **Save View** dialog box, type the name for the view. When you save a modified view, you can either save it with the same name to overwrite it, or change the name to create a new view. You cannot save a custom view with the name of a predefined view, for example *Active alerts*.

The view does not save the current timeframe, or the **Group by server** setting.

If you save a view that has all the SQL Servers selected, any new instances you subsequently add to SQL Response will not automatically be included in that view.

Viewing alert details

Click on an alert to view more information about it in the alert details pane below the list of raised alerts:



Alert	Server	Type	Detail	Last occurred	All occurren..
Low	DEVELOPMENT1	Long running query		10/09/2008 09:54:28	10
High	DEVELOPMENT1	Job did not start	Server Maintenance and Repair	10/09/2008 09:53:00	9
Medium	DEVELOPMENT2	SQL Agent not running	DEVELOPMENT2	10/09/2008 09:44:14	1
Medium	FINANCE2	Low physical memory	FINANCE2	10/09/2008 09:43:49	1
Medium	SALESTEST	Low physical memory	SALESTEST	10/09/2008 09:43:46	1
Medium	DEVELOPMENT1	Low physical memory	DEVELOPMENT1	10/09/2008 09:39:44	1
Medium	DEVELOPMENT1	Low disk space	C:	10/09/2008 09:39:43	1

Job did not start

Server Maintenance and Repair on DEVELOPMENT1

Job should have started at: 10/09/2008 09:53:00

Reason for not starting: Previous run hasn't finished

User name: DEVELOPMENT1\Administrator

Next scheduled to run: 10/09/2008 09:54:00

All occurrences: 9

Viewing this occurrence: Latest: 10/09/2008 09:53:00

Alert raised at: 10/09/2008 09:53:00

Alert cleared at:

⊞ **Job step details**

⊞ **Performance snapshot** when alert was raised

⊞ **System processes** when alert was raised

⊞ **Comments**

The alert details pane shows a summary of the alert, and provides a number of expandable sections. These sections contain further information about the alert, and diagnostic data, to help you identify what was happening on the SQL Server at the time the alert was raised. Click on a section heading to expand or collapse that section.

The first expandable section is specific to the type of alert raised. For job-related alerts, for example, the **Job step details** section lists the steps of the job, and if appropriate, the outcome of each step. For **Long running query**, **Blocked SQL process**, and **SQL deadlock** alerts, the first section lists the processes involved and includes a query fragment for each process.

All data for the alert is relative to the time the alert was raised. For example, the **Next scheduled to run** time for job type alerts shows the next run after the displayed alert time; if you are viewing an alert raised several hours or days previously, the time of the next run may be in the past.

You can adjust the height of the alert details pane by dragging it up or down, or by using the ▁ ▬ ▔ buttons to move the pane to a fixed height.

Click ⧉ to open the alert in a new window.

Time zones of alert times

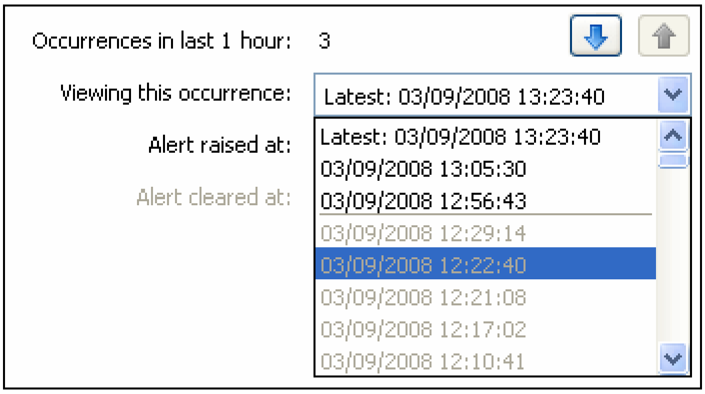
All times shown in SQL Response are local to the SQL Server on which the alert was raised.

For example, if you are monitoring SQL Servers in Moscow on a computer running SQL Response in New York, then the alert times displayed in SQL Response will all be eight hours ahead. If you are monitoring SQL Servers across numerous time zones, this means that sorting the list of alerts by the **Last occurred** column may not give you a chronological list of the most recent alerts.

Viewing alert occurrences

When you view an alert, the latest (most recent) occurrence is displayed. You can review earlier occurrences of the alert by doing one of the following:

- click the previous occurrence button ⬇
- from the **Viewing this occurrence** list, select the occurrence you want to view.



Occurrences in last 1 hour: 3

Viewing this occurrence: Latest: 03/09/2008 13:23:40

Alert raised at:

Alert cleared at:

Latest: 03/09/2008 13:23:40
03/09/2008 13:05:30
03/09/2008 12:56:43
03/09/2008 12:29:14
03/09/2008 12:22:40
03/09/2008 12:21:08
03/09/2008 12:17:02
03/09/2008 12:10:41

The most recent occurrences are listed first, but you can scroll down to view occurrences not within the current timeframe (shown in grey).

How alerts are grouped

A new occurrence of an alert is raised when the same problem happens again on a SQL Server; that is, the basic details of the alert are identical. For example, when the same job fails twice in an hour, an additional occurrence of the **Job failure** alert is raised for that specific job; the job name is shown in the **Detail** column. If a different job fails, a separate **Job failure** alert is raised.

- For connection type alerts (**SQL Server unreachable**, **Computer unreachable**, **SQL Server login failure**, and **Computer login failure**), a second occurrence of the alert is raised only when the connection has first been re-established and then fails for a second time. If the SQL Server remains offline or unavailable, only the initial occurrence of the alert will be listed.
- For **SQL deadlock**, **Blocked SQL process** and **Long running query** alerts, repeated instances of the problem are grouped into a single alert, rather than into separate alerts for each process or query. Select an occurrence of the alert to view the details for the specific problem.

  For example, a number of different queries overrun on a SQL Server within a half-hour period, resulting in 20 occurrences of a single **Long running query** alert, rather than 20 separate alerts in the main list. Use the **Viewing this occurence** list in the alert details pane to review the query involved in each separate overrun.

When you clear an alert, you clear the status of the alert as a whole; you cannot clear an individual occurrence of the alert. Clearing an alert identifies the current status of the problem rather than being a property of an individual occurrence. See Clearing alerts and adding comments (page 55).

Viewing the performance snapshot

The **Performance snapshot** section provides a snapshot of SQL Server and CPU activity leading up to the time the alert was raised, and for a short period afterwards. As SQL Response is monitoring your SQL Servers continuously, this data is already captured, and can be included in the details of the alert. If you have enabled the collection of SQL Profiler trace (page 38) on a monitored SQL Server, then you can also view the Transact-SQL statements for each connection to that SQL Server in the period leading up to the alert.

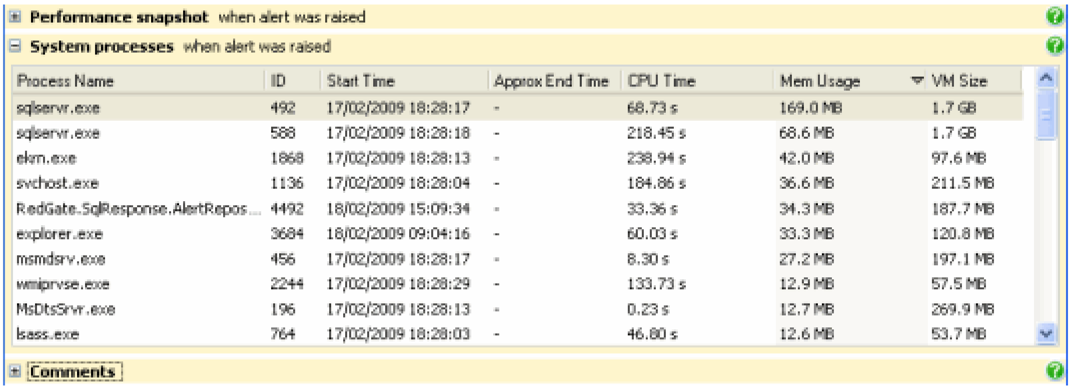
See Viewing performance snapshot data (page 51).

Performance snapshot information is available for all alerts except the following:

- Computer login failure, SQL Server login failure
- Computer unreachable, SQL Server unreachable
- SQL Agent not running

Viewing system processes

The **System processes** section lists the Windows processes that were running at the time the alert was raised.



⊞ **Performance snapshot** when alert was raised

⊟ **System processes** when alert was raised

Process Name	ID	Start Time	Approx End Time	CPU Time	Mem Usage	▼ VM Size
sqlservr.exe	492	17/02/2009 18:28:17	-	68.73 s	169.0 MB	1.7 GB
sqlservr.exe	588	17/02/2009 18:28:18	-	218.45 s	68.6 MB	1.7 GB
ekrn.exe	1868	17/02/2009 18:28:13	-	238.94 s	42.0 MB	97.6 MB
svchost.exe	1136	17/02/2009 18:28:04	-	184.86 s	36.6 MB	211.5 MB
RedGate.SqlResponse.AlertRepos...	4492	18/02/2009 15:09:34	-	33.36 s	34.3 MB	187.7 MB
explorer.exe	3684	18/02/2009 09:04:16	-	60.03 s	33.3 MB	120.8 MB
msmdsrv.exe	456	17/02/2009 18:28:17	-	8.30 s	27.2 MB	197.1 MB
wmiprvse.exe	2244	17/02/2009 18:28:29	-	133.73 s	12.9 MB	57.5 MB
MsDtsSrvr.exe	196	17/02/2009 18:28:13	-	0.23 s	12.7 MB	269.9 MB
lsass.exe	764	17/02/2009 18:28:03	-	46.80 s	12.6 MB	53.7 MB

⊞ Comments

For each process, the following data is provided:

- **ID**: The numerical ID assigned to the process by Windows while it was running.
- **Start Time**: When the process started.
- **Approx End Time**: When the process ended. This column is empty when the process was still running at the time the alert was raised.
- **CPU Time**: The total processor time, in seconds, used by the process since it was started, at the time the alert was raised.
- **Mem Usage**: The amount of main memory used by the process, at the time the alert was raised
- **VM Size**: The amount of virtual memory, or address space, committed to a process, at the time the alert was raised.

These performance measures are the same as those available in Windows Task Manager.

Click on a column heading to sort the processes by that column.

The data provided is a snapshot of Windows process information from around the time the alert was raised. This data could be from up to ten seconds before the exact time the alert was raised; process data is captured every ten seconds.

Viewing comments

Comments can be added to an alert only when that alert is cleared; they allow you to add notes about the causes of the problem and describe any actions taken to resolve the issue. When comments are added to an alert, the time and user name are added automatically.

Comments are not attached to any specific occurrence of an alert. If an alert is cleared, re-raised, and then cleared again several times, each additional comment will be appended to the list. All comments for an alert are always available to view.

To view the comments made on alerts that are currently at Cleared status, ensure that the *Cleared alert* filter check mark is selected in the **Alert levels to show** area of the Filter pane.

See Clearing alerts and adding comments (page 55).

Selecting multiple alerts

You can select several alerts in the main list at once.

- To open each selected alert in a new window, click 🗗
- Click **Clear Selected Alerts** to clear multiple alerts at once. Adding a comment to the cleared alerts will add the same comment to each alert.

Viewing recommendation details

Recommendation details do not contain multiple sections. Each recommendation contains basic information about the data that caused the recommendation to be raised, and a description of the issue.

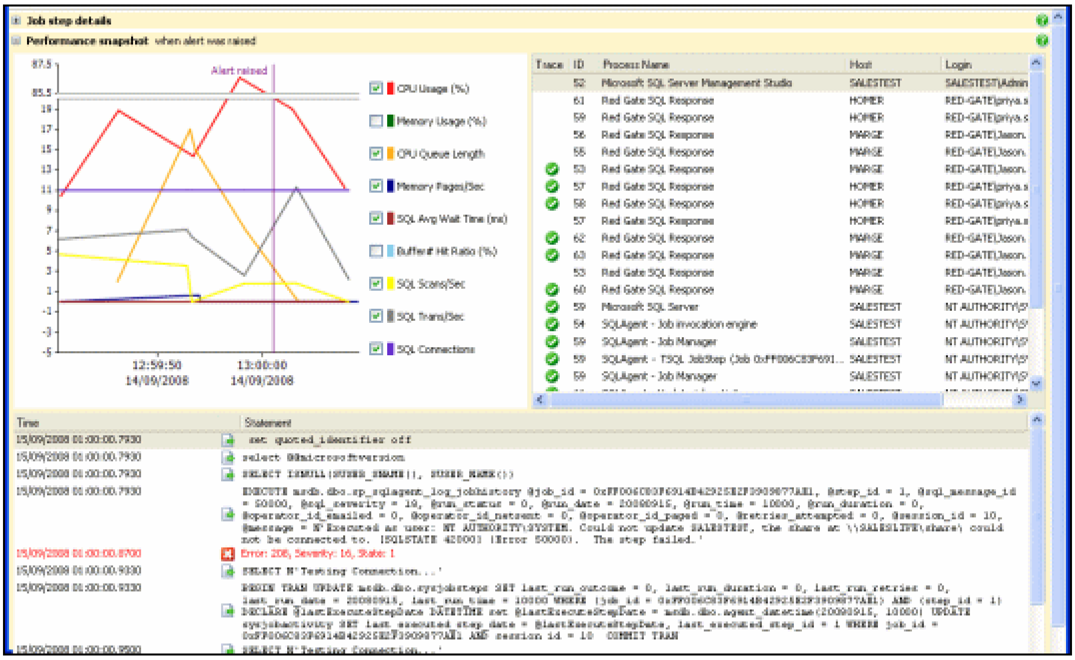
See About alerts and recommendations (page 5).

Viewing performance snapshot data

The Performance snapshot section of an alert shows:

- a graph of various performance counter values, such as CPU Usage and SQL Scans/Sec, in the period leading up to the alert and for a short period afterwards
- all SQL processes being executed in that period
- the T-SQL statements for each process (only available for SQL Servers on which you have enabled SQL Profiler trace data collection)

The performance snapshot is not a live view of current activity on your SQL Server.



Trace	ID	Process Name	Host	Login
	52	Microsoft SQL Server Management Studio	SALESTEST	SALESTEST\Admin
	61	Red Gate SQL Response	HOMER	RED-GATE\priya.s
	59	Red Gate SQL Response	HOMER	RED-GATE\priya.s
	56	Red Gate SQL Response	MARGE	RED-GATE\Jason.
	55	Red Gate SQL Response	MARGE	RED-GATE\Jason.
✓	53	Red Gate SQL Response	MARGE	RED-GATE\Jason.
✓	57	Red Gate SQL Response	HOMER	RED-GATE\priya.s
✓	58	Red Gate SQL Response	HOMER	RED-GATE\priya.s
	57	Red Gate SQL Response	HOMER	RED-GATE\priya.s
✓	62	Red Gate SQL Response	MARGE	RED-GATE\Jason.
✓	63	Red Gate SQL Response	MARGE	RED-GATE\Jason.
	53	Red Gate SQL Response	MARGE	RED-GATE\Jason.
✓	60	Red Gate SQL Response	MARGE	RED-GATE\Jason.
✓	59	Microsoft SQL Server	SALESTEST	NT AUTHORITY\S
✓	54	SQLAgent - Job invocation engine	SALESTEST	NT AUTHORITY\S
✓	59	SQLAgent - Job Manager	SALESTEST	NT AUTHORITY\S
✓	59	SQLAgent - TSQL JobStep (Job 0xFF006C83F691...	SALESTEST	NT AUTHORITY\S
✓	59	SQLAgent - Job Manager	SALESTEST	NT AUTHORITY\S

Time	Statement
15/09/2008 01:00:00.7930	set quoted_identifier off
15/09/2008 01:00:00.7930	select @@microsoftversion
15/09/2008 01:00:00.7930	SELECT ISNULL(SUSER_SNAME(), SUSER_NAME())
15/09/2008 01:00:00.7930	EXECUTE msdb.dbo.sp_sqlagent_log_jobhistory @job_id = 0xFF006C83F6914B4292SE2F3909077AE1, @step_id = 1, @sql_message_id = 50000, @sql_severity = 10, @run_status = 0, @run_date = 20080915, @run_time = 10000, @run_duration = 0, @operator_id_emailed = 0, @operator_id_netsent = 0, @operator_id_paged = 0, @retries_attempted = 0, @session_id = 10, @message = N'Executed as user: NT AUTHORITY\SYSTEM. Could not update SALESTEST, the share at \\SALESLITE\share\ could not be connected to. [SQLSTATE 42000] (Error 50000). The step failed.'
15/09/2008 01:00:00.8700	Error: 208, Severity: 16, State: 1
15/09/2008 01:00:00.9330	SELECT N'Testing Connection...'
15/09/2008 01:00:00.9330	BEGIN TRAN UPDATE msdb.dbo.sysjobsteps SET last_run_outcome = 0, last_run_duration = 0, last_run_retries = 0, last_run_date = 20080915, last_run_time = 10000 WHERE (job_id = 0xFF006C83F6914B4292SE2F3909077AE1) AND (step_id = 1) DECLARE @lastExecuteStepDate DATETIME set @lastExecuteStepDate = msdb.dbo.agent_datetime(20080915, 10000) UPDATE sysjobactivity SET last_executed_step_date = @lastExecuteStepDate, last_executed_step_id = 1 WHERE job_id = 0xFF006C83F6914B4292SE2F3909077AE1 AND session_id = 10 COMMIT TRAN
15/09/2008 01:00:00.9500	SELECT N'Testing Connection...'

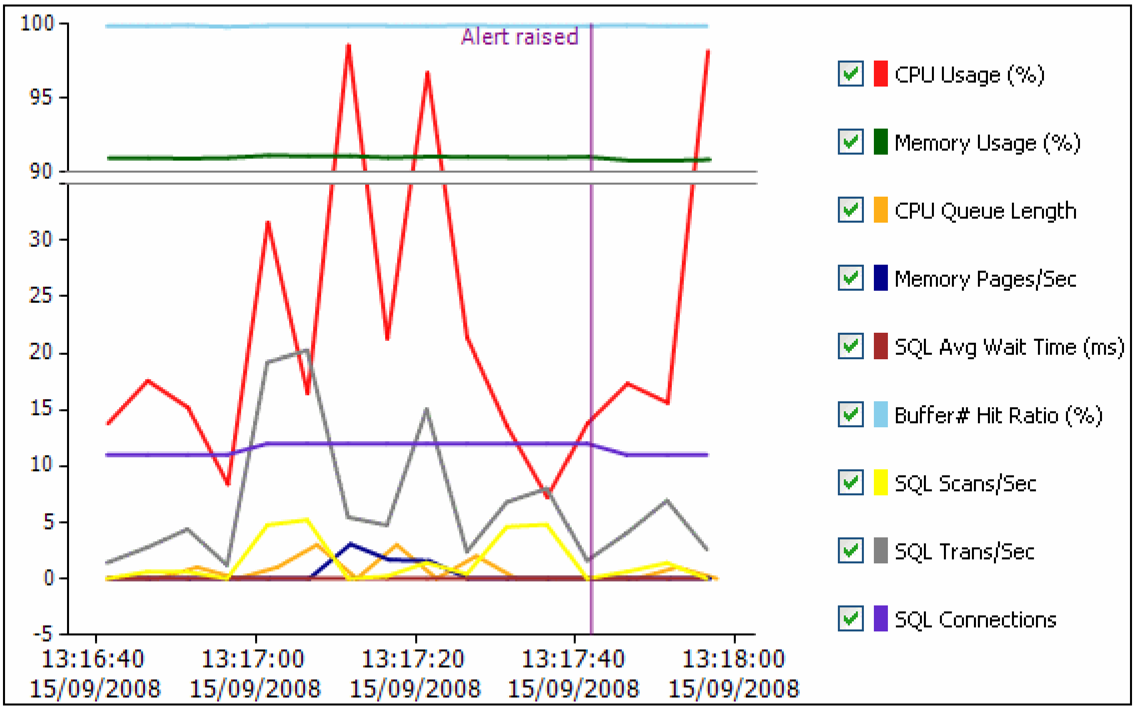
Performance snapshot information is available for all alerts except the following:

- Computer login failure, SQL Server login failure
- Computer unreachable, SQL Server unreachable
- SQL Agent not running

Working with the performance graph

The graph shows the values of performance counters leading up to the time the alert was raised.

For **CPU utilization unusual** alerts, only CPU and memory counters are available.



The relative position of the **Alert raised** line along the time axis varies, depending when the alert was raised relative to the last time SQL Response collected performance data from the SQL Server. SQL Response polls the SQL Server every few seconds for performance counter data.

Note: For a short period after first installing or restarting the Alert Repository service, some counters may not be available.

Selecting performance counters

Select the performance counters to display on the graph using the check boxes. The scale automatically adjusts based on your selection. The following counters are available to display:

CPU Usage: Percentage of processor resources being utilised

Memory Usage: Percentage of physical memory being used

CPU Queue Length: Number of threads waiting for processor time

Memory Pages/Sec: Number of memory pages read or written to disk

SQL Avg Wait Time: Amount of time in milliseconds that a user is waiting for a lock.

This is an average value compiled over the monitoring period

Buffer Hit Ratio: Percentage of pages in the buffer compared to the disk

SQL Scans/Sec Number of unrestricted full table or index scans per second

SQL Trans/Sec: Number of database transactions executed per second

SQL Connections: Number of active user connections

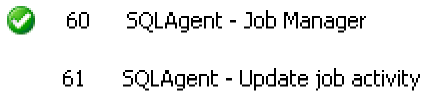
For more information about these performance counters, see:

- SQL Server Central: Basic Perfomance Monitoring Counters http://www.sqlservercentral.com/articles/Administering/performancemonitoringbasiccounters/1348/#cputime
- SQL Server Performance: Tips for Using Performance Monitor CPU Counters (http://www.sql-server-performance.com/tips/performance_monitor_cpu_counter_p1.aspx)

Viewing SQL process information and SQL Profiler trace data

To view the Transact-SQL statements for a SQL process, select the process in the list next to the graph. Transact-SQL statements are captured by a SQL Profiler trace that runs continuously on SQL Servers that you specify. If trace has not been enabled, Transact-SQL statements are not available.

Processes marked with a green check mark next to them have trace data to view; the statements are listed below the performance graph.

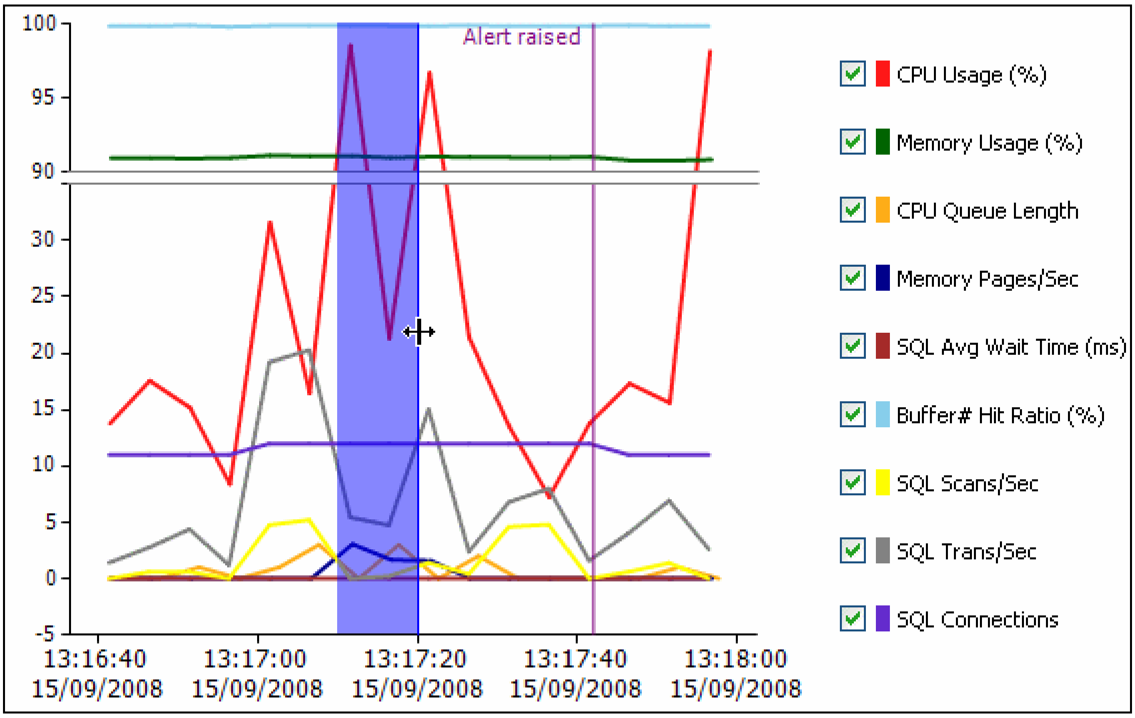


✓	60	SQLAgent - Job Manager	Trace data available
	61	SQLAgent - Update job activity	No trace data available

If a process does not have trace data, it indicates that the connection was idle at the time of the alert.

Examining SQL statements from a defined time period

When you select a process in the list, you can then click and drag to select a region on the performance graph:



When you select a region, SQL statements for the selected process are filtered to show only those that executed in the defined time period.

To clear the selected region, click once on the graph.

If you do not clear the region, the SQL statements for any process you select will continue to be filtered by the time period on the graph.

Turning on SQL Profiler trace

To turn on tracing for a SQL Server, right-click on the SQL Server in the **Servers to show** area of the Filter pane, and select **Server Properties**. In the Server Properties dialog box, select **Enable collection of trace data**. Future alerts raised on this SQL Server will show captured Transact-SQL statements.

See Collecting SQL Profiler trace data (page 38)for more information.

Clearing alerts and adding comments

Clearing a raised alert may be useful when you want to:

- categorise the alert as not currently a problem
- remove the alert from your checklist of issues requiring further investigation
- indicate that the problem raised by the alert is resolved
- temporarily hide the alert until the next time it is raised
- add a note to the alert to record the cause of the problem, or to describe any actions taken to resolve it

Cleared alerts can still be viewed; all the alert details (including performance snapshot data) are available for all cleared alerts.

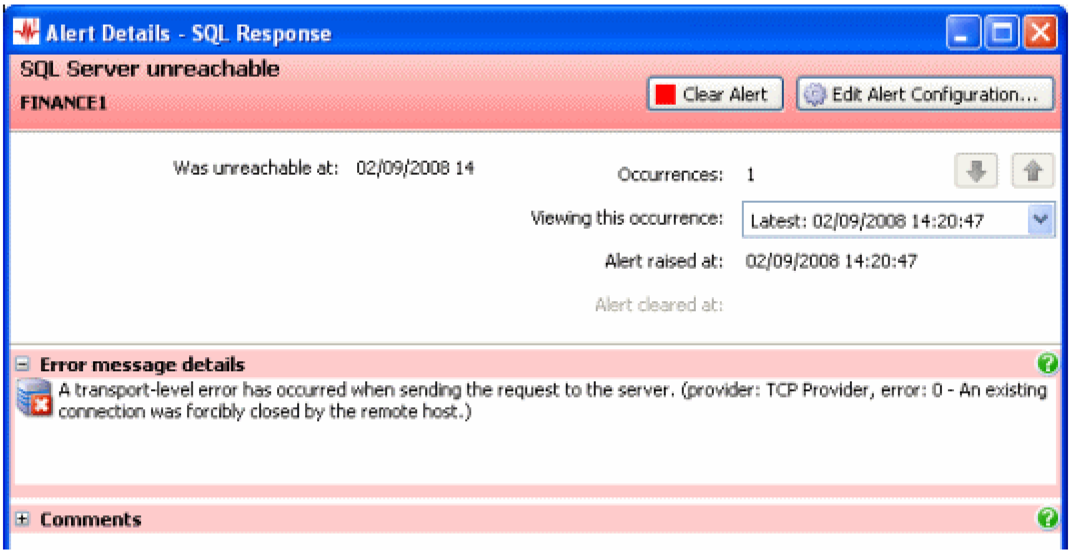
To clear an alert

To clear an alert, click **Clear Alert** in the alert details pane.

You can also right-click the alert in the main list, and select **Clear Alert**.

Difference between active and cleared alerts

Active alerts are alerts that are not currently at cleared status. The image below shows an alert raised at high level:



SQL Server unreachable
FINANCE1

Was unreachable at: 02/09/2008 14

Occurrences: 1

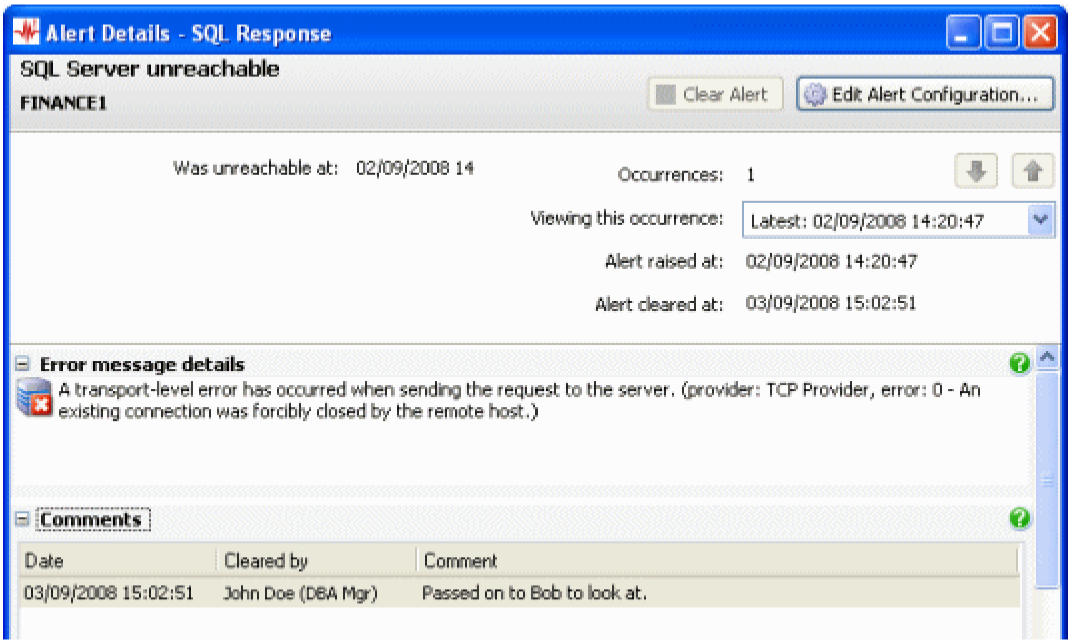
Viewing this occurrence: Latest: 02/09/2008 14:20:47

Alert raised at: 02/09/2008 14:20:47

Alert cleared at:

A transport-level error has occurred when sending the request to the server. (provider: TCP Provider, error: 0 - An existing connection was forcibly closed by the remote host.)

The same alert after being cleared:



SQL Server unreachable

FINANCE1

Was unreachable at: 02/09/2008 14

Occurrences: 1

Viewing this occurrence: Latest: 02/09/2008 14:20:47

Alert raised at: 02/09/2008 14:20:47

Alert cleared at: 03/09/2008 15:02:51

Error message details

A transport-level error has occurred when sending the request to the server. (provider: TCP Provider, error: 0 - An existing connection was forcibly closed by the remote host.)

Comments

Date	Cleared by	Comment
03/09/2008 15:02:51	John Doe (DBA Mgr)	Passed on to Bob to look at.

- All the alert details can still be viewed
- A comment has been added, displayed in the **Comments** section
- An **Alert cleared at** timestamp has been added; if the alert is raised again, this timestamp is removed

Clear or disable an alert?

A cleared alert will be raised again when the same problem re-occurs. For example, if you clear a **Job failure** alert for a particular job, and that job fails again an hour later, the alert is raised again, and the number of occurrences of the alert is increased by one.

Disable the alert when you do not want to be informed again about this particular job failing:

1. In the alert details pane for the selected alert, click **Edit Alert Configuration**.
2. Select **Disabled**.
3. Click **Apply Changes**.

By default, the alert is only disabled for the specific job; you can disable it for all jobs on the SQL Server, or for all Job failures on a number of SQL Servers. Disabling an alert stops it being raised in the future; it does not remove any historical alerts from SQL Response.

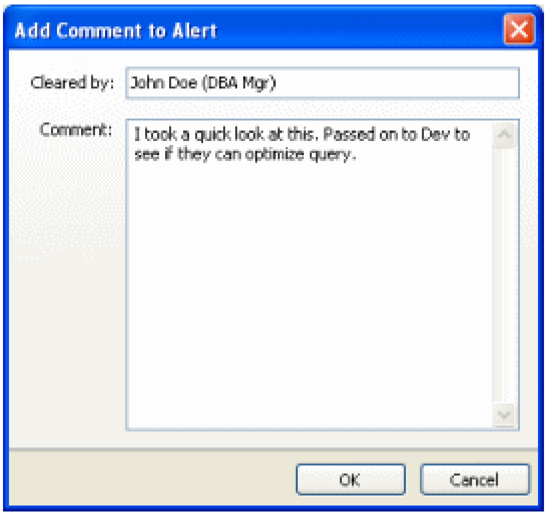
If you receive a large number of alerts for an issue, you may want to fine-tune the settings of the alert. Click **Edit Alert Configuration** to adjust the alert thresholds.

See Configuring alerts and recommendations (page 66).

Adding a comment when clearing an alert

To add a comment to an alert, you first need to clear the alert.

When you click **Clear Alert**, the Add Comment to Alert dialog box is displayed, in which you can enter the comment text.



Cleared by: John Doe (DBA Mgr)

Comment: I took a quick look at this. Passed on to Dev to see if they can optimize query.

Your user name is automatically entered in the **Cleared by** box; you can edit this name, if required.

- To add a comment, type the comment text and click **OK**.

  The comment is added to the Comments section of the alert details, and includes the time the alert was cleared and the user who cleared it. Each new comment is added to the top of the list.

- To add an entry to the Comments section indicating the time the alert was cleared and the user who cleared it, but without any comment text, click **OK** without typing anything in the **Comment** box.

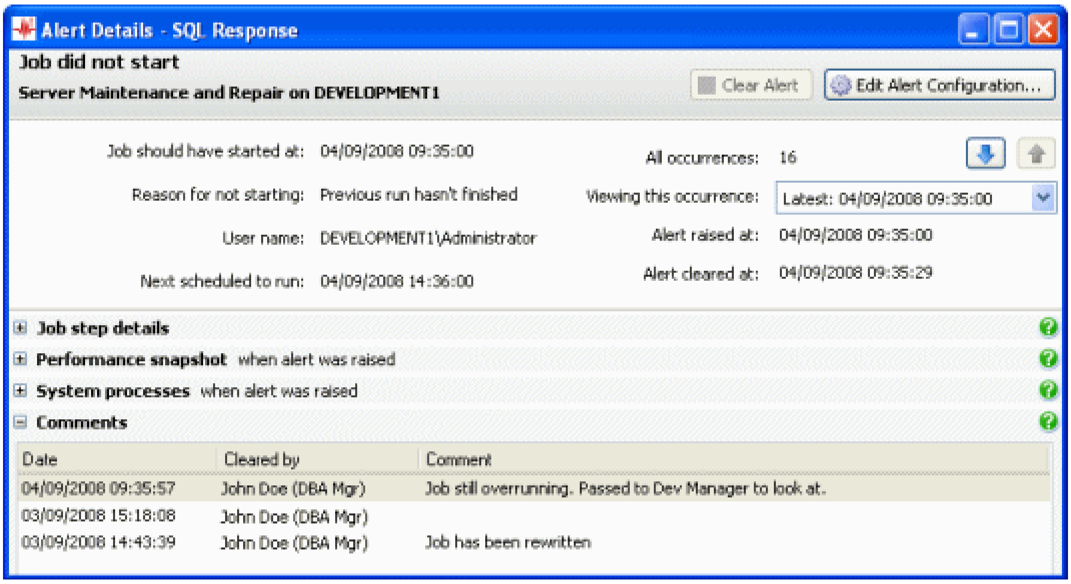
  Note: Adding blank entries may not be allowed (the **OK** button is unavailable until some text is typed). You can change this behavior in the Options dialog box.

- To clear the alert without adding a comment, click **Skip**. No entry will be added to the Comments section of the alert details.

  Note: Skipping alerts may not be allowed (the **Skip** button is unavailable). You can change this behavior in the Options dialog box (see the next section, Turning off comments).

Comments are not specific to an individual occurrence of an alert. All comments added to the alert are always displayed in the Comments section of the alert details, for any selected occurrence.

In the example below, the **Job did not start** alert has been raised a total of 16 times for the Server Maintenance and Repair job on the DEVELOPMENT1 server. Three comments have been made, indicating the alert has been cleared three times. Note that the second time the alert was cleared, a blank comment was added.


Alert Details - SQL Response
Clear Alert
Edit Alert Configuration...
Job step details
Performance snapshot when alert was raised
System processes when alert was raised
Comments

Job did not start

Server Maintenance and Repair on DEVELOPMENT1

Job should have started at: 04/09/2008 09:35:00

Reason for not starting: Previous run hasn't finished

User name: DEVELOPMENT1\Administrator

Next scheduled to run: 04/09/2008 14:36:00

All occurrences: 16

Viewing this occurrence: Latest: 04/09/2008 09:35:00

Alert raised at: 04/09/2008 09:35:00

Alert cleared at: 04/09/2008 09:35:29

Date	Cleared by	Comment
04/09/2008 09:35:57	John Doe (DBA Mgr)	Job still overrunning. Passed to Dev Manager to look at.
03/09/2008 15:18:08	John Doe (DBA Mgr)	
03/09/2008 14:43:39	John Doe (DBA Mgr)	Job has been rewritten

There is no correlation between any specific comment and a particular occurrence of an alert. Clearing an alert changes its overall status, rather than clearing a selected occurrence only.

Changing comment options

When you first run SQL Response, adding comments when clearing an alert is enabled by default, and comment text is required before the alert can be cleared. To change options for adding comments:

From the **Alert Repository** menu, select **Alert Repository Options**, then:

- To disallow comments, clear **Enable comments when clearing an alert**.
- To require comment text in the **Add Comment to Alert** dialog box before the alert can be cleared, select **Require comments**.

  When this option is cleared, you can **Skip** the adding of a comment.

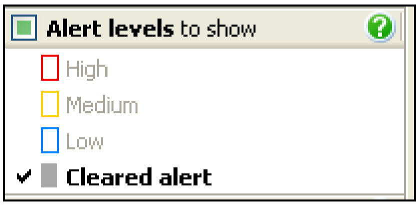
Viewing cleared alerts

Viewing cleared alerts allows you to review information about alerts, for example the number of times a particular problem occurred within the last month, and information about actions taken to resolve the problem.

To view cleared alerts:

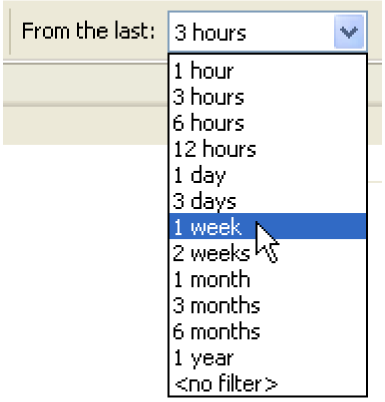
1. In the Filter pane, clear the check box next to **Alert levels to show** to hide alerts at all levels.

2. Select *Cleared alert*.



   Only cleared alerts will be displayed in the main list.

3. Select any other filters you want, for example to view cleared alerts on selected SQL Servers only, or to view specific types of alert.
4. Select a timeframe in the **From the last** drop-down list. For example, select *1 week* to view all cleared alerts from the last seven days:



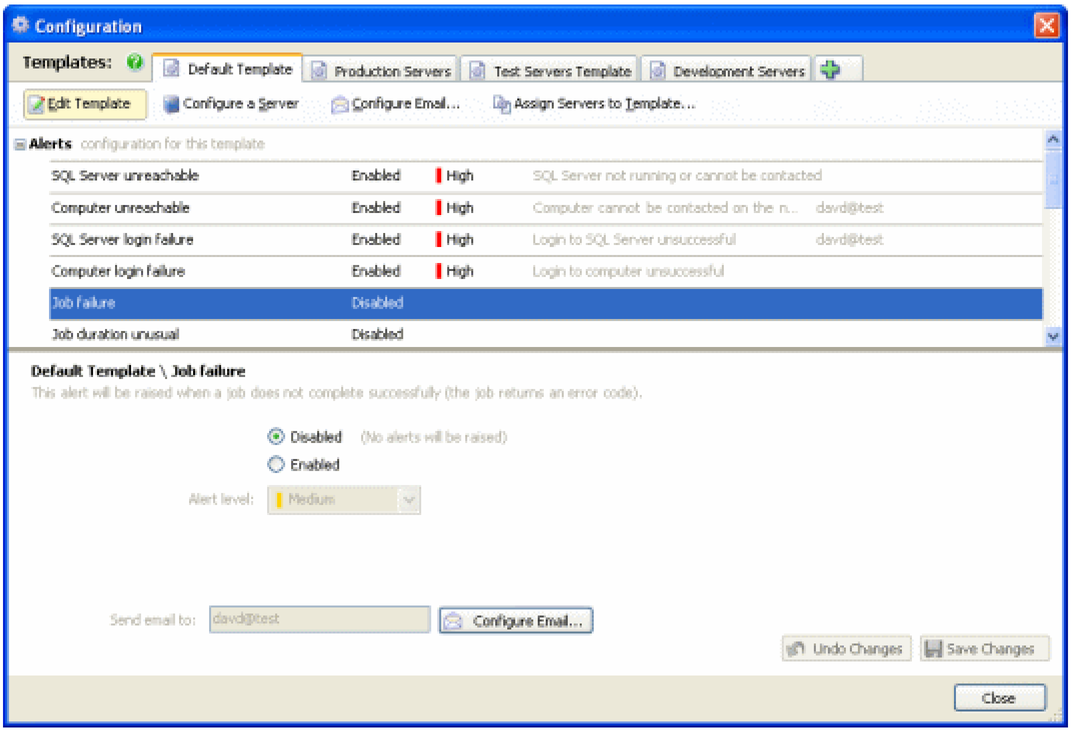
5. Select the **Group by server** check box in the main toolbar to collapse the list into SQL Server groups, if required.

Select an alert, then review the details for any occurrence of the cleared alert. See Viewing alert details (page 46).

Working with configuration templates

A configuration template contains the settings for each type of alert and recommendation. Each monitored SQL Server inherits the settings for all alerts and recommendations from the template it uses.

To manage templates, click **Configuration** in the main toolbar to display the **Configuration** dialog box.



Alerts configuration for this template

SQL Server unreachable	Enabled	High	SQL Server not running or cannot be contacted	
Computer unreachable	Enabled	High	Computer cannot be contacted on the n...	davd@test
SQL Server login failure	Enabled	High	Login to SQL Server unsuccessful	davd@test
Computer login failure	Enabled	High	Login to computer unsuccessful	
Job failure	Disabled			
Job duration unusual	Disabled			

Default Template \ Job failure

This alert will be raised when a job does not complete successfully (the job returns an error code).

● Disabled (No alerts will be raised)

○ Enabled

Alert level: Medium

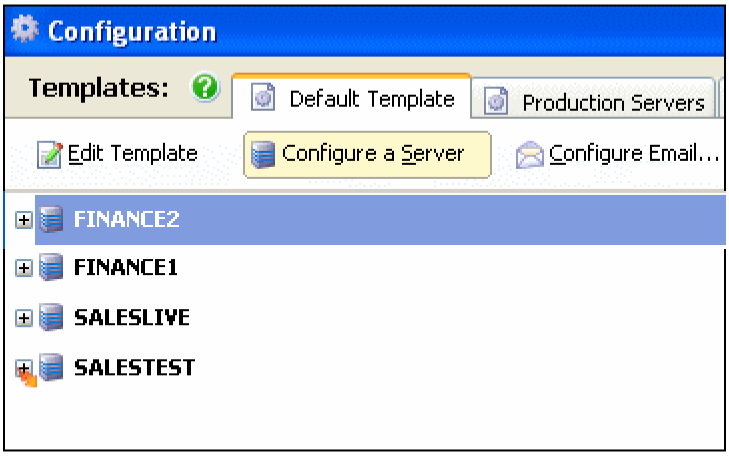
Send email to: davd@test

- The upper pane lists all the types of alert and recommendation available in SQL Response, and the current settings for each for the currently selected template.
- The lower pane enables you to edit the settings for each type of alert or recommendation, either at the template level (on the **Edit Template** page) or for a single SQL Server (on the **Configure a Server** page).

SQL Response has a **Default Template** available when you first run it. This provides suitable settings for monitoring a typical SQL Server, but you can create your own templates for groups of SQL Servers that require the same configuration for monitoring, for example *Production servers* or *Test servers*.

Listing SQL Servers that use a template

To see the list of SQL Servers using a template, click on the tab to select the template, then click **Configure a Server**.



Editing a template

Editing a template changes the configuration of all SQL Servers that use the template. When you have already applied an override at the SQL Server level, then editing the template will not change any overridden options. See Configuring alerts and recommendations (page 66).

To edit a template:

1. Click on the tab to select the template, then click **Edit Template**.
2. Expand **Alerts** or **Recommendations** to list all the types of alert or recommendation available in SQL Response.
3. Select the type of alert or recommendation you want to edit, then change the settings in the lower pane.
4. Click **Save Changes**. The change will be applied to all SQL Servers that use the template.

Note: A change to the configuration of an alert will only apply the next time the alert is raised. Currently raised alerts are not updated.

Configuring an individual SQL Server

Changing the configuration of a SQL Server overrides the template setting for that individual SQL Server.

To edit the settings on an individual SQL Server:

1. Click **Configure a Server**.
2. Expand the SQL Server and then expand **Alerts** or **Recommendations**.
3. Select the type of alert or recommendation you want to edit, then change the configuration for that alert in the lower pane.

4. Click **Apply Changes**. The change will be applied only to this SQL Server.

   An override icon indicates that the setting on this SQL Server is now different from the template setting, and will not be updated when the template is edited.

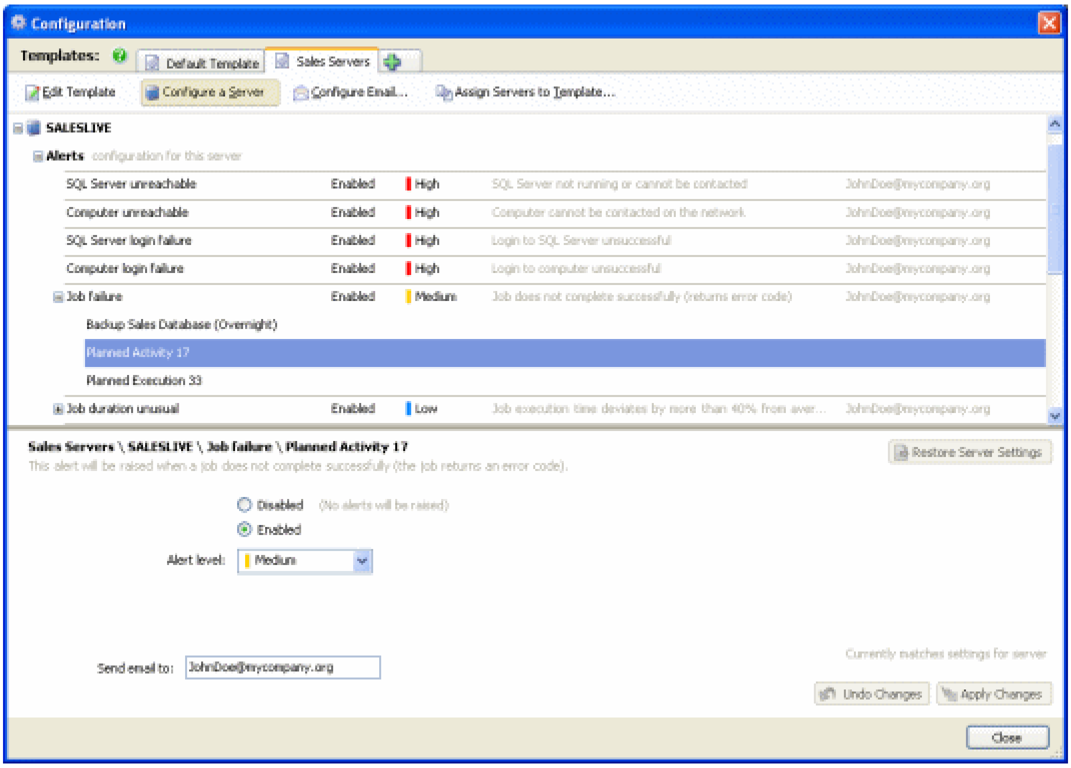
Configuring an individual job, disk drive or database

When you configure an alert that can apply to different instances of a problem, you can fine-tune the settings for each item individually. You can edit the alert settings for each individual:

- job (for **Job failure**, **Job did not start**, **Job duration unusual** alerts)
- disk drive (for **Low disk space** alerts)
- database (for **Database state change** alerts)

For example, the Job failure alert may be configured on a SQL Server to be raised at Medium level, but for a single specific job, you can disable it, or raise it at a different level. (Note: individual jobs, databases, and disks are not available if SQL Response cannot connect to the SQL Server.)

To change the settings for a specific job, expand the **Job failure** alert to list all the jobs on the SQL Server then select the job:



SALESLIVE

Alerts configuration for this server

SQL Server unreachable	Enabled	High	SQL Server not running or cannot be contacted	JohnDoe@mycompany.org
Computer unreachable	Enabled	High	Computer cannot be contacted on the network	JohnDoe@mycompany.org
SQL Server login failure	Enabled	High	Login to SQL Server unsuccessful	JohnDoe@mycompany.org
Computer login failure	Enabled	High	Login to computer unsuccessful	JohnDoe@mycompany.org
Job failure	Enabled	Medium	Job does not complete successfully (returns error code)	JohnDoe@mycompany.org
Backup Sales Database (Overnight)				
Planned Activity 17				
Planned Execution 33				
Job duration unusual	Enabled	Low	Job execution time deviates by more than 40% from aver...	JohnDoe@mycompany.org

Sales Servers \ SALESLIVE \ Job failure \ Planned Activity 17

This alert will be raised when a job does not complete successfully (the job returns an error code).

○ Disabled (No alerts will be raised)

● Enabled

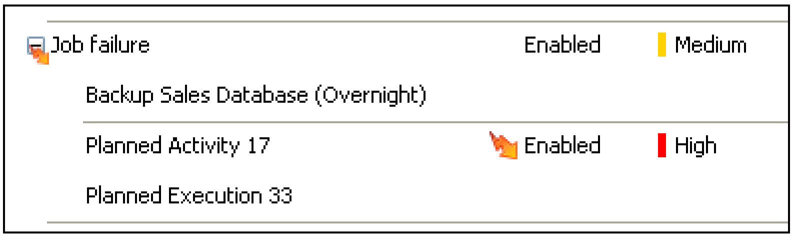
Alert level: Medium

Send email to: JohnDoe@mycompany.org

Currently matches settings for server

© Red Gate Software Ltd

When you click **Apply Changes**, the settings for that job will override the settings on the SQL Server, and the override icon will be added to the job:



Job failure	Enabled	Medium
Backup Sales Database (Overnight)		
Planned Activity 17	Enabled	High
Planned Execution 33		

Jobs, databases, or disks that are listed without an override icon inherit their settings from the alert at the SQL Server level (that is, from the alert type at the top of the list).

Note: An alert for a specific SQL Server can override the template setting, and can itself also have overrides at the job, database, or disk level.

Creating a new template

To create a new template:

1. Click the Create new template tab.

   The **Create New Template** dialog box is displayed.
2. Type the name for the new template in the **Template name** box.
3. Select whether to create the template with the same initial configuration as an existing template, or with the default Red Gate settings (these are the same configuration settings as those in the Default Template when first installed).
4. Click **Create**. The template is added as an additional tab on the right of the existing tabs, and selected by default.

You can rename a template by right-clicking on the template tab and selecting **Rename Template**. Template names must be unique.

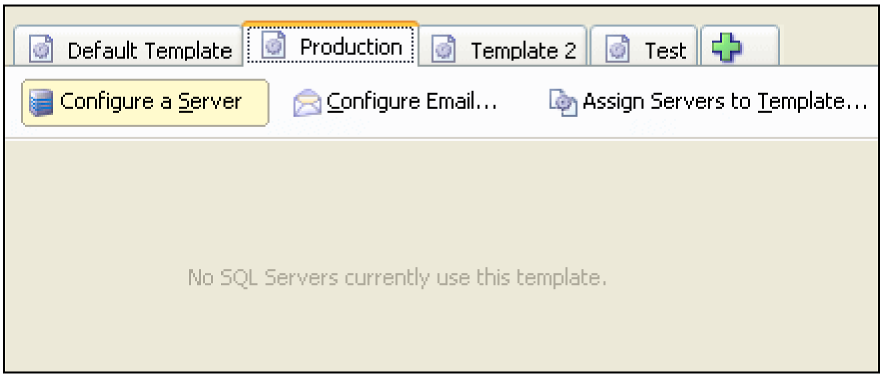
Assigning SQL Servers to a template

When you create a new template, it is created empty (no SQL Servers use it). Once you have created one or more new templates, you should assign SQL Servers to those templates.

To specify which template a SQL Server should use:

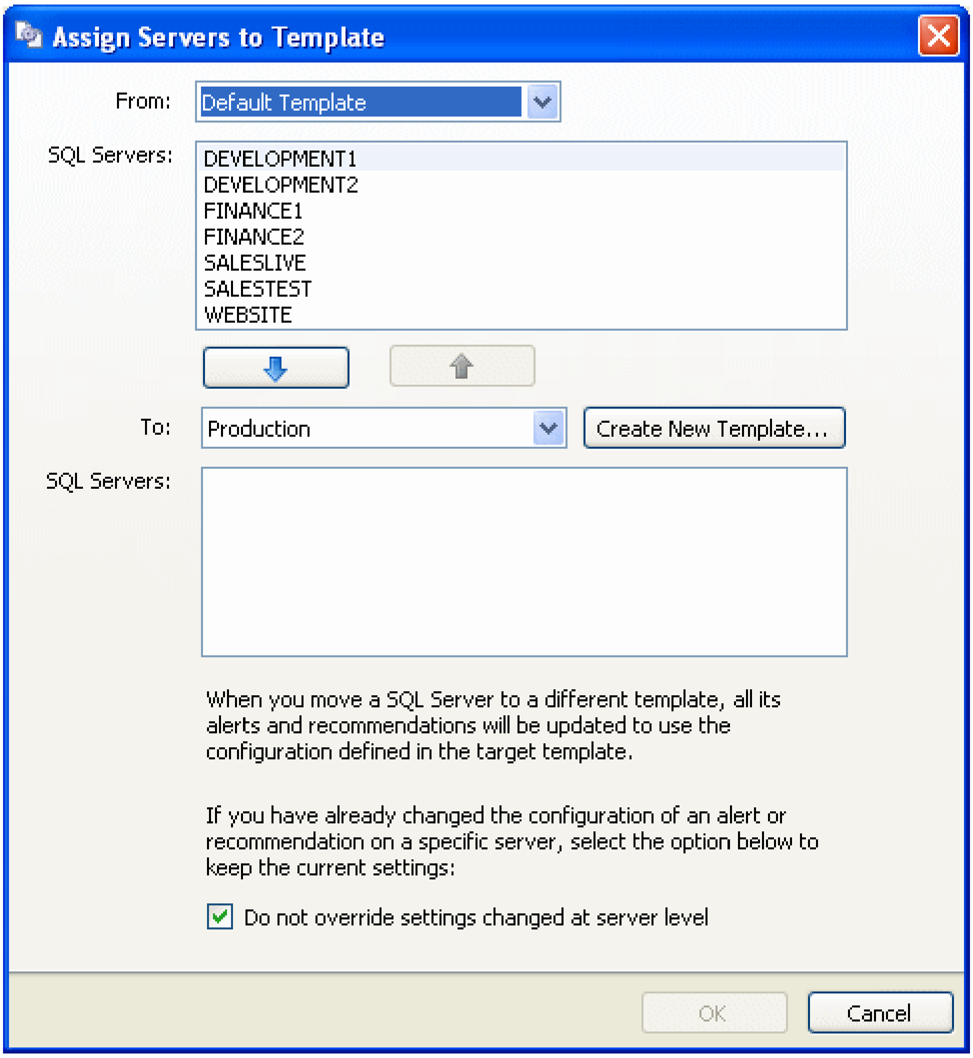
1. Click the tab for the template you want to assign one or more SQL Servers to.

To view the SQL Servers currently using the selected template, click **Configure a Server**.



No SQL Servers currently use this template.

2. Click **Assign Servers to Template**.
3. The **Assign Servers to Template** dialog box is displayed.



From: Default Template

SQL Servers:
DEVELOPMENT1
DEVELOPMENT2
FINANCE1
FINANCE2
SALESLIVE
SALESTEST
WEBSITE

To: Production

SQL Servers:

When you move a SQL Server to a different template, all its alerts and recommendations will be updated to use the configuration defined in the target template.

If you have already changed the configuration of an alert or recommendation on a specific server, select the option below to keep the current settings:

☑ Do not override settings changed at server level

4. The **From** list always selects the Default Template when the dialog box is first displayed, and lists below it the SQL Servers currently using the Default Template. If

the SQL Server you want to assign does not currently use the Default Template, select a different template from the **From** list.

5. The **To** list defaults to the template that you selected before clicking **Assign Servers to Template** (the target template). If this is not the correct template, select a different template from the **To** list.

   Note: the **SQL Servers** box below the **To** list is always empty when the dialog box is first displayed; it does not list any SQL Servers that already use the target template. If you select a different template from the **To** list, all the assigned SQL Servers will be moved.

6. Click ⬇ to assign a SQL Server to the target template. It will be displayed below the **To** list.
7. Select **Do not override settings changed at server level** if you want to keep overrides on any of the SQL Servers you are assigning to a different template. An override is created when you have edited the setting for an individual SQL Server, so that it differs from the template settings. Overrides are indicated by ↘ in the **Configuration** dialog box.
8. When you have assigned the SQL Servers you want, click **OK**.

Deleting a template

To delete a template, right-click on the template tab and select **Delete Template**.

You can only delete a template which is not used by any SQL Servers. To reassign SQL Servers to a different template, click **Assign Servers to Template**.

Configuring individual alerts and recommendations

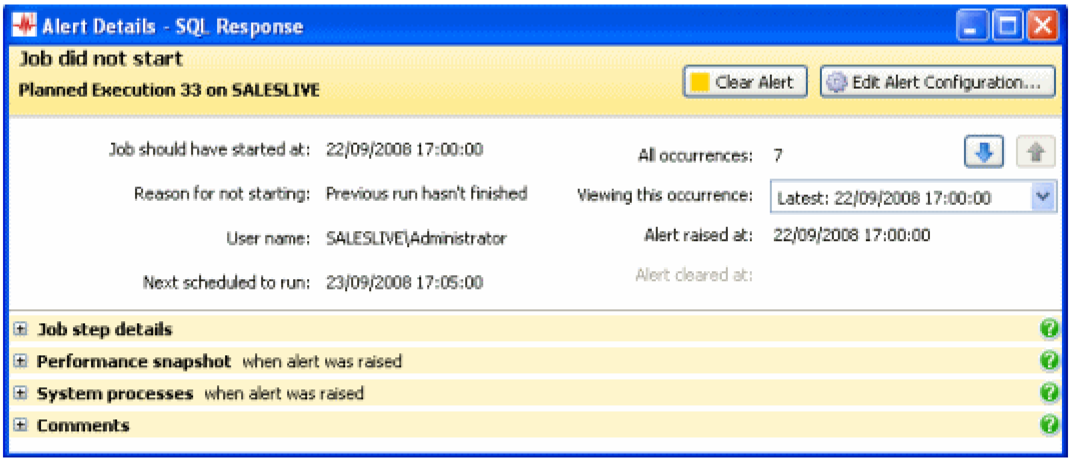
When you view a raised alert, you may decide that the particular alert raised for that job, database, disk drive or SQL Server is not behaving as you want. You can:

- disable it altogether, so it is never raised again
- change the level at which it is raised in future
- change the thresholds that trigger the alert to be raised
- set up an email recipient for the alert

To configure a raised alert, select the alert in the main list and click **Edit Alert Configuration**.

Changing the configuration of a raised alert

In this example, a **Job did not start** alert has been raised for the Planned Execution 33 job on the SALESLIVE server at medium level. Suppose you want to raise this type of alert as a high level alert in future.



Job did not start

Planned Execution 33 on SALESLIVE

Job should have started at: 22/09/2008 17:00:00

Reason for not starting: Previous run hasn't finished

User name: SALESLIVE\Administrator

Next scheduled to run: 23/09/2008 17:05:00

All occurrences: 7

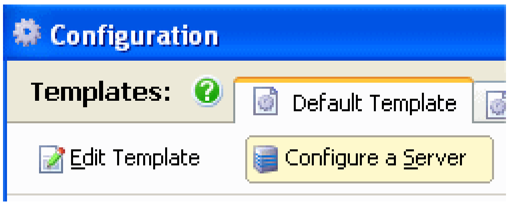
Viewing this occurrence: Latest: 22/09/2008 17:00:00

Alert raised at: 22/09/2008 17:00:00

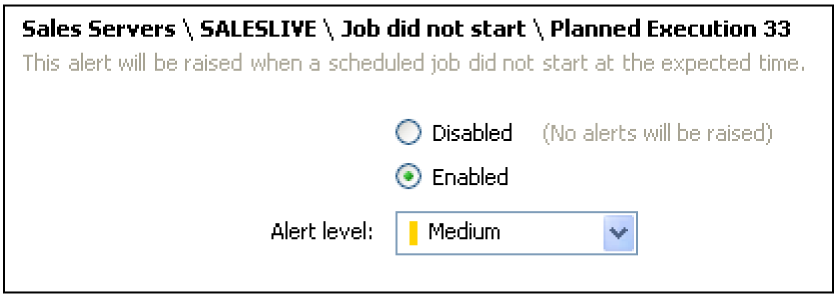
Alert cleared at:

Alert Details - SQL Response
Clear Alert
Edit Alert Configuration...
Job step details
Performance snapshot when alert was raised
System processes when alert was raised
Comments

Clicking **Edit Alert Configuration** displays the **Configuration** dialog box at the **Configure a Server** page for the SALESLIVE server:



In the lower pane you can see the current settings for this alert, and the scope of the changes you are making.



Sales Servers \ SALESLIVE \ Job did not start \ Planned Execution 33

This alert will be raised when a scheduled job did not start at the expected time.

○ Disabled (No alerts will be raised)

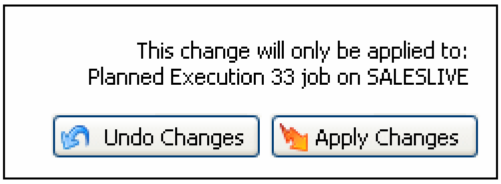
● Enabled

Alert level: Medium

When you edit a raised alert, by default you change the settings at the most specific level, as indicated by the breadcrumbs (the path shown in bold text) above the alert options. These breadcrumbs are in the format:

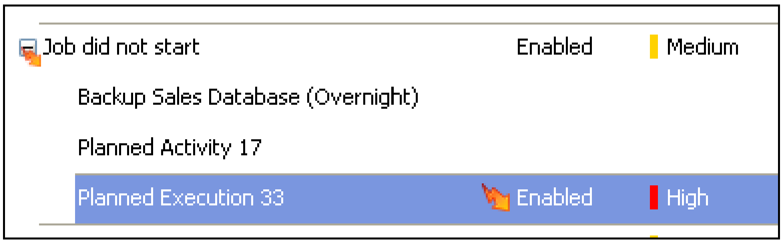
Template \ SQL Server \ Alert type \ Job

When you edit the alert settings, a message confirming the scope of the changes is displayed in the bottom right of the pane:



This change will only be applied to:
Planned Execution 33 job on SALESLIVE

This message confirms that you are editing the settings for a single job only on the SQL Server. When you change the alert settings for an individual job, an override icon indicates that the settings for the specific job (changed to high level) are now different from the settings on the SQL Server (medium level):



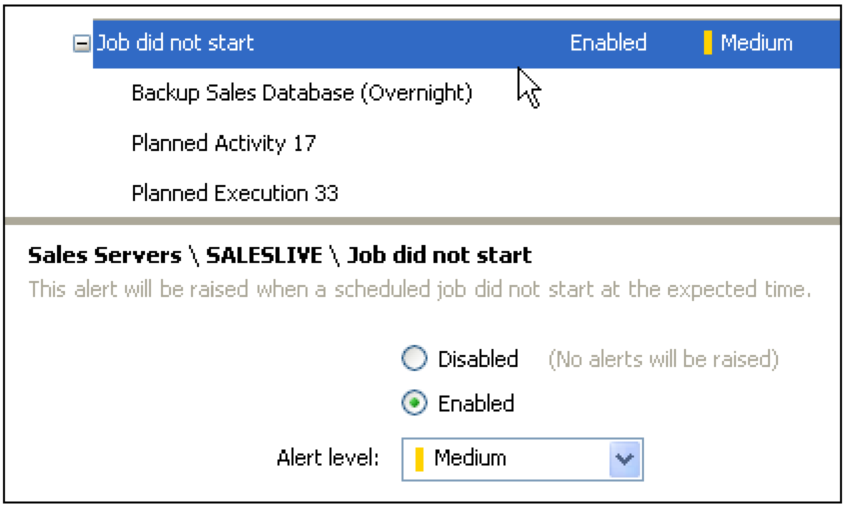
Job did not start	Enabled	Medium
Backup Sales Database (Overnight)		
Planned Activity 17		
Planned Execution 33	Enabled	High

Note that not all alert types can be expanded to show more detail. For example, Long running query alerts or Blocked SQL process alerts cannot be expanded to list individual alerts or queries.

The level of the existing alert will not be altered; when the alert is next raised, its level will be high.

Configuring all jobs on a SQL Server

If you want to change the settings for all jobs on the SQL Server, click the **Job did not start** alert type above the list of individual jobs in the upper pane of the **Configuration** dialog box:



⊟ Job did not start Enabled ▮ Medium

Backup Sales Database (Overnight)

Planned Activity 17

Planned Execution 33

Sales Servers \ SALESLIVE \ Job did not start

This alert will be raised when a scheduled job did not start at the expected time.

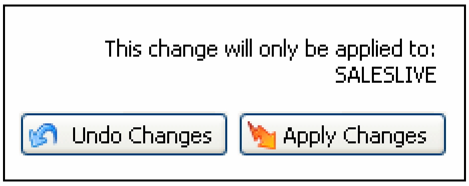
○ Disabled (No alerts will be raised)

● Enabled

Alert level: ▮ Medium

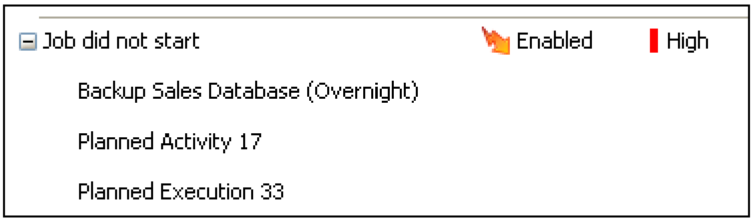
The breadcrumbs in the lower pane now indicate that you are editing the settings for the **Job did not start** alert type on the SALESLIVE server only, but a specific job is not listed. The changes will be applied to all jobs on this SQL Server.

When you edit the alert settings, a message confirming the scope of the changes is displayed:



This change will only be applied to:
SALESLIVE

When you change the alert settings on a SQL Server, an override icon indicates that the setting for the SQL Server (in this case, high level) is now different from the setting in the template (medium level):



⊟ Job did not start Enabled ▮ High

Backup Sales Database (Overnight)

Planned Activity 17

Planned Execution 33

Editing the email recipient

To edit the email recipient setting for an alert, you first need to enter your Mail Server settings. If you have not yet set up your Mail Server in SQL Response, the **Send email to** box is unavailable. You can apply an email recipient to a single alert, or to all alerts in a template.

Emails are not sent for recommendations.

See Setting up email (page 70).

Configuring recommendations

Recommendations are configured in the same way as alerts:

- Each type of recommendation can be configured for a specific database on a SQL Server, or for all databases on the SQL Server, or at the template level
- Most recommendations do not have configurable thresholds
- Recommendations do not have different levels
- Recommendations do not send emails

Configuring several SQL Servers

If you want to edit the settings of an alert or recommendation for several SQL Servers, then you need to edit the template (page 60). Editing the template applies the configuration to all SQL Servers that use the template.

Setting up email

SQL Response can send an email to one or more email recipients whenever an alert is raised.

An email is sent when an alert is first raised, and whenever an alert changes from cleared to active status. If you do not clear an alert, no further emails will be sent for that alert, regardless of any additional occurrences of the alert.

No emails are sent for recommendations.

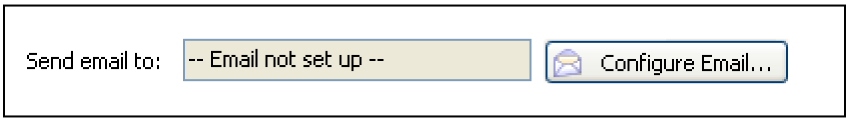
Setting up email overview

To set up email recipients for alerts, you need to:

1. Enter your mail server settings.
2. Apply an email recipient to a template; this changes the settings on all alerts for all SQL Servers using the template.
3. If required, change the email recipient for an alert type in a template.
4. If required, change the email recipient for an alert type on a specific SQL Server.

Entering your mail server settings

Before you can set up an email recipient for an individual alert or for a template, you need to enter the details of your mail server. The **Send email to** box in the **Configuration** dialog box is unavailable for an alert until you set up the mail server.

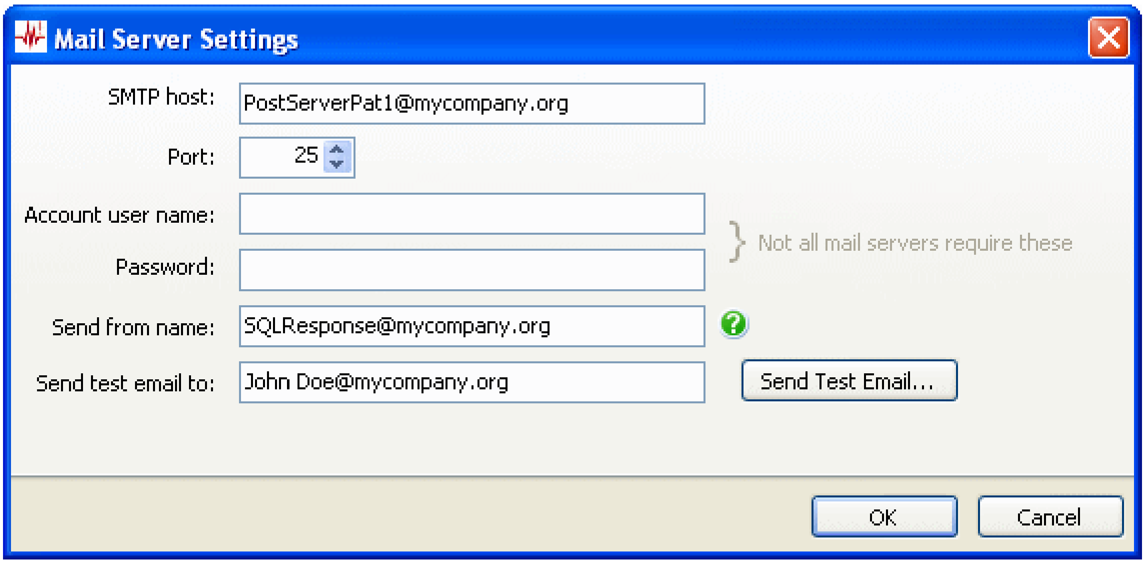


Send email to: -- Email not set up --

1. In the main toolbar, click ⚙ **Configuration**.
2. Click **Configure Email**.

   Note: You can also click **Configure Email** next to the **Send email to** box if you are editing the settings of an alert in the **Configuration** dialog box.

3. Click **Mail Server Settings**. The Mail Server Settings dialog box is displayed:



SMTP host: PostServerPat1@mycompany.org

Port: 25

Account user name:

Password:

} Not all mail servers require these

Send from name: SQLResponse@mycompany.org

Send test email to: John Doe@mycompany.org

4. Enter the details of your SMTP host, port number and, if required, the account user name and password.
5. In the **Send from name** box, enter the name that you want to be displayed in the From field of emails sent by SQL Response. The name must be a valid email format, including the @ symbol.
6. Click **OK**.

Note: When your mail server settings have been entered, the **Configure Email** button is no longer displayed next to the email address for an alert type.

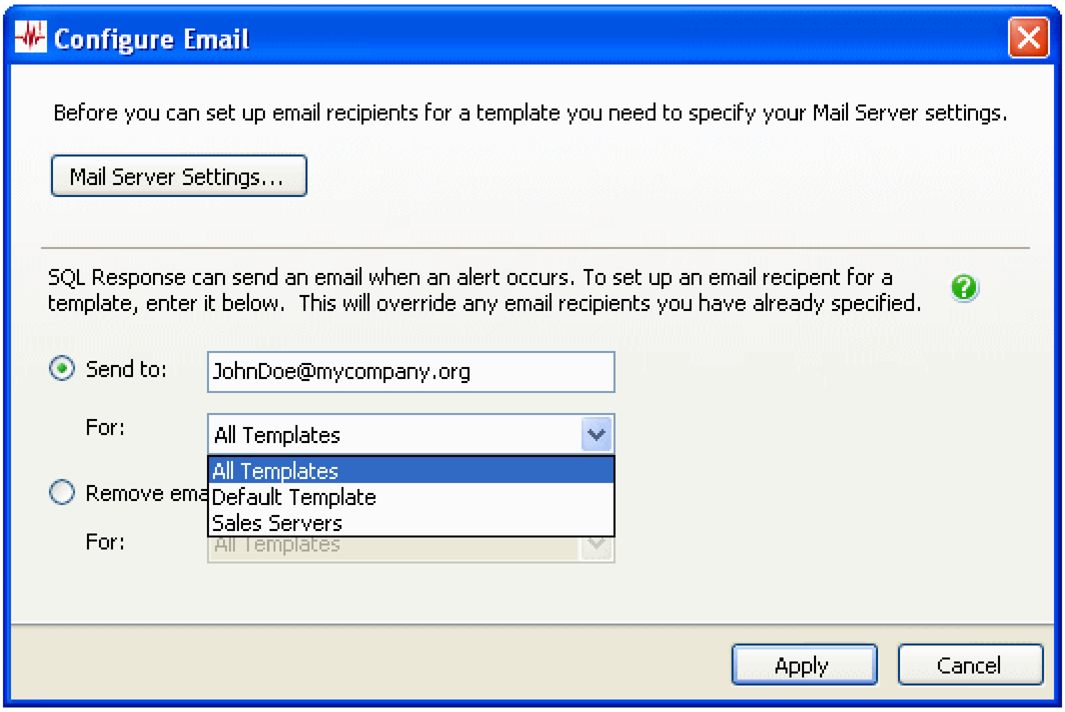
Applying an email recipient to a template

An email recipient applied to a template will be used for all alerts on each SQL Server that uses the template. This is a quick way to apply the same email address to a group of SQL Servers at once.

1. In the main toolbar, click **Configuration**.
2. Click **Configure Email**.
3. In the **Configure Email** dialog box, enter the email recipient address in the **Send to** box.

   To send an email to multiple recipients, separate the email addresses with a comma. To send emails to larger groups of people, use a mailing list.

4. Select the template to apply the recipient to. You can select **All Templates** to use the email address for all alerts.



Configure Email

Before you can set up email recipients for a template you need to specify your Mail Server settings.

SQL Response can send an email when an alert occurs. To set up an email recipent for a template, enter it below. This will override any email recipients you have already specified.

● Send to: JohnDoe@mycompany.org

For: All Templates

○ Remove ema

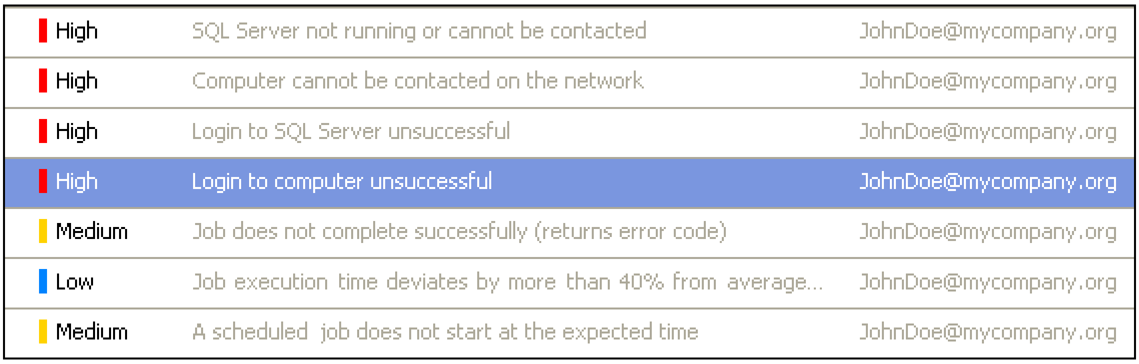
For: All Templates

5. Click **Apply**.

Applying a new email recipient for a template will override any existing email recipients for all alerts in that template, including email recipients you have edited individually in the template or changed at the SQL Server level.

Changing the email recipient for an alert type in a template

When you have applied an email recipient to a template, you can see the email recipient for each alert type as part of the alert configuration. The column on the right in the upper pane lists the email address for each type of alert. If you have already applied an email recipient to the template, these email addresses will all be the same:

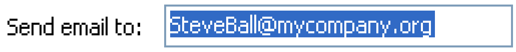


🟥 High	SQL Server not running or cannot be contacted	JohnDoe@mycompany.org
🟥 High	Computer cannot be contacted on the network	JohnDoe@mycompany.org
🟥 High	Login to SQL Server unsuccessful	JohnDoe@mycompany.org
🟥 High	Login to computer unsuccessful	JohnDoe@mycompany.org
🟨 Medium	Job does not complete successfully (returns error code)	JohnDoe@mycompany.org
🟦 Low	Job execution time deviates by more than 40% from average...	JohnDoe@mycompany.org
🟨 Medium	A scheduled job does not start at the expected time	JohnDoe@mycompany.org

To override the email recipient for any individual alert type in the template:

1. Select the alert type in the upper pane.

2. In the **Send email to** box in the lower pane, delete the existing text, and type the new email address.



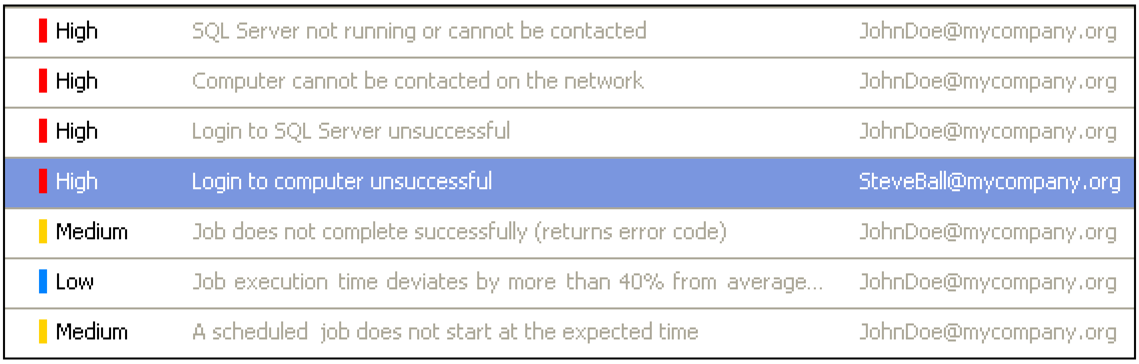
Send email to: SteveBall@mycompany.org

To disable email alerts on this alert type, delete the email address to leave the **Send email to** box blank.

3. Click **Save Changes**.

Note: If you select a different alert type without first saving, any changes will be lost; you need to save one alert type at a time.

The email recipient is changed for the selected alert type:



🟥 High	SQL Server not running or cannot be contacted	JohnDoe@mycompany.org
🟥 High	Computer cannot be contacted on the network	JohnDoe@mycompany.org
🟥 High	Login to SQL Server unsuccessful	JohnDoe@mycompany.org
🟥 High	Login to computer unsuccessful	SteveBall@mycompany.org
🟨 Medium	Job does not complete successfully (returns error code)	JohnDoe@mycompany.org
🟦 Low	Job execution time deviates by more than 40% from average...	JohnDoe@mycompany.org
🟨 Medium	A scheduled job does not start at the expected time	JohnDoe@mycompany.org

Any **Computer login failure** alerts for a SQL Server using this template will now be sent to the specified email address. If you subsequently apply a different email address to the template, this email address will be overwritten.

Changing the email recipient for an alert type on a specific SQL Server

You can also change the email recipient for an alert type on a specific SQL Server, or for a single job, database, or disk on a SQL Server.

To change the email recipient for an alert that has been raised (active or cleared):

1. Click **Edit Alert Configuration** in the alert details pane to display the settings for the alert.
2. In the **Send email to** box, type the new email address.
3. Click **Apply Changes**.

By default, the email recipient will be changed only for the specifically selected alert on the selected SQL Server. For example, if you are editing the configuration of a job alert (such as **Job failure** or **Job did not start**), the email recipient will be used only for the particular job.

If you change the email recipient for a raised, active alert (that is, an alert not currently at cleared status), an email will not be sent until the alert is cleared and then re-raised.

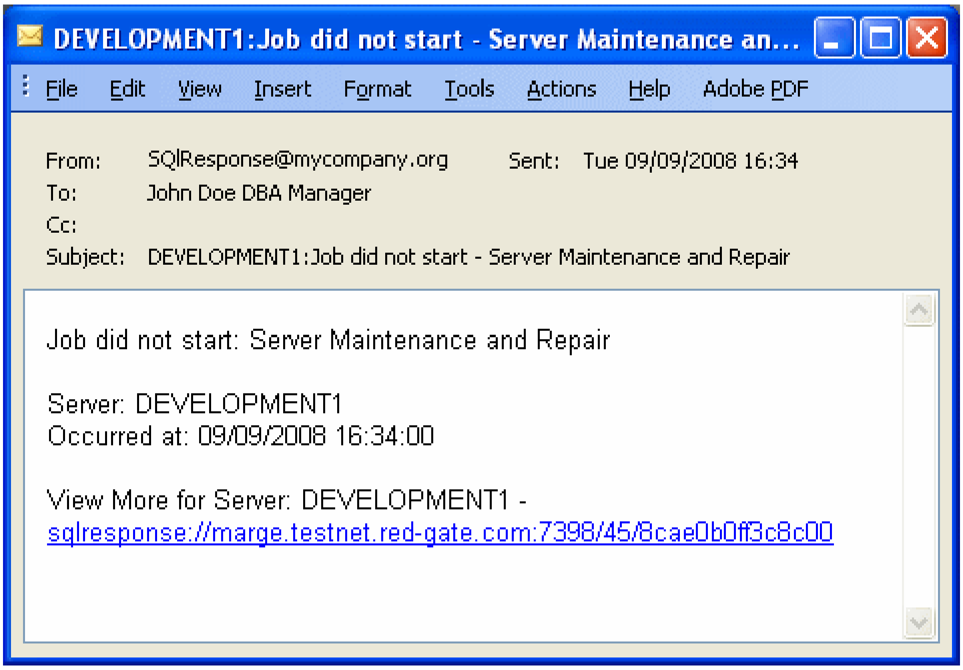
To change the email recipient for an alert that has not been raised:

1. In the main toolbar, click **Configuration**.
2. In the Configuration dialog box, select the tab for the template used by the SQL Server whose email settings you want to edit.
3. Click **Configure a Server**.
4. Expand the list of alerts to find the alert type you want to change the email recipient for, then select the alert type.
5. To change the email recipient only for a particular job, database, or disk, expand the alert type further and select the specific entry beneath the alert type.
6. In the **Send email to** box, type the new email address.
7. Click **Apply Changes**.

If you subsequently apply a different email recipient to the template, the email for an alert type or a specific job, database, or disk, will be overwritten.

Using an email hyperlink to view an alert

The emails that SQL Response sends contain basic details about the alert, and a hyperlink to open the alert in SQL Response:



From: SQlResponse@mycompany.org Sent: Tue 09/09/2008 16:34
To: John Doe DBA Manager
Cc:
Subject: DEVELOPMENT1:Job did not start - Server Maintenance and Repair

Job did not start: Server Maintenance and Repair

Server: DEVELOPMENT1
Occurred at: 09/09/2008 16:34:00

View More for Server: DEVELOPMENT1 -
sqlresponse://marge.testnet.red-gate.com:7398/45/8cae0b0ff3c8c00

When you click the hyperlink:

- SQL Response will be started, if it is not already running
- The alert will be opened in a new window

If the alert has been raised by a different Alert Repository to the one that you are currently connected to, then you will be prompted to connect to the correct Alert Repository to view the alert.

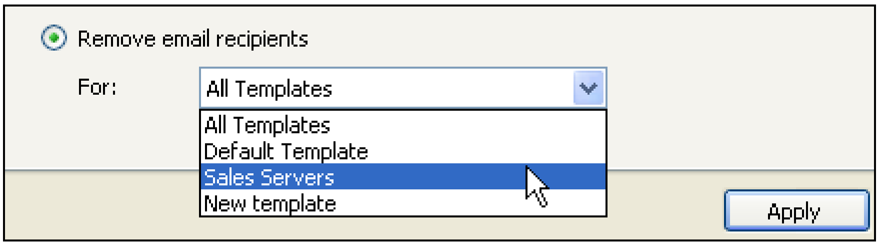
Removing email recipients

To remove an email recipient for a specific alert:

1. In the **Configuration** dialog box, select the alert type.
2. Delete the email address, leaving the **Send email to** box empty.
3. Click **Apply Changes**.

To remove email recipients for all SQL Servers in a template:

1. In the **Configuration** dialog box, click **Configure Email.**
2. In the **Configure Email** dialog box, select **Remove email recipients**.
3. Select the template to remove the recipients from. You can select **All Templates** to remove all email addresses for all alerts.



● Remove email recipients

For: All Templates

4. Click **Apply**.

About fragmented indexes

The **Fragmented index** recommendation is raised when SQL Response determines that the percentage of logical fragmentation of any one index in a monitored database, based on running the `DBCC SHOWCONTIG` command, exceeds a specified percentage.

Causes of fragmentation

In the normal operation of any production SQL Server, as INSERTs, UPDATEs, and DELETEs are made to tables (and their indexes), rows become fragmented (scattered throughout the database, in addition to leaving empty spaces). Fragmentation requires SQL Server to perform extra work when accessing indexes, which in turn can lead to slower performance. Because of this, it is a best practice to remove this fragmentation on a regular basis.

If you have created an Index Rebuild or Reorganize job, and are still seeing this recommendation, it is possible the jobs are failing. If this is the case, a **Job failure** or **Job did not start** alert for that job will also be raised by SQL Response.

Resolving a fragmented index

There are two ways to resolve index fragmentation:

- Use the `ALTER INDEX` command with the `REBUILD` option
- Use the `ALTER INDEX` command with the `REORGANIZE` option

One or the other of these commands should be executed as part of a regular index defragmentation plan, either executed as a SQL Server Agent job or via a Maintenance Plan.

If an index is heavily fragmented (>30%), then the `REBUILD` option should be used because it physically rebuilds all of the indexes. However, this is a resource-intensive process that can affect your server's performance and temporarily block users from accessing their data. If you have the Enterprise version of SQL Server, the REBUILD option has an ONLINE option that reduces a lot of these problems.

If an index is not heavily fragmented (between 5% and 30%), then you can use the `REORGANIZE` option instead. This option does not do as thorough a job as the `REBUILD` option does, but is it much more lightweight and has minimal impact on an active production database.

Ideally, you should only REBUILD or REORGANIZE indexes that are fragmented. Not all indexes become fragmented (for example, because they are read-only or little used), and defragmenting them uses unnecessary resources. Unfortunately, it is not all that easy to write Transact-SQL code that goes through every index, determines its level of fragmentation, and then performs the appropriate type of defragmentation only on those indexes that need it. Because of this, many DBAs run either REBUILD or REORGANIZE on all of a database's indexes at the same time.

Links to more information

Microsoft SQL Server 2000 Index Defragmentation Best Practices
(http://www.microsoft.com/technet/prodtechnol/sql/2000/maintain/ss2kidbp.mspx)

SQL Server Performance.com : Understanding SQL Server's DBCC SHOWCONTIG
(http://www.sql-server-performance.com/articles/dba/dt_dbcc_showcontig_p1.aspx)

SQL Server Performance.com : Tips for rebuilding indexes
(http://www.sql-server-performance.com/tips/rebuilding_indexes_p1.aspx)

MS SQL City : Reducing SQL Server Index Fragmentation
(http://www.mssqlcity.com/Articles/Adm/index_fragmentation.htm)

About full backups

The **Full backup overdue** recommendation is raised when:

- There are no entries in the `[msdb].[dbo].[backupfile]` system table for a full backup of a monitored database, or
- The last full backup of a monitored database in the `[msdb].[dbo].[backupfile]` system table is older than a time you specify.

Best practice for full backups

As a best practise, system and production databases should have a full backup performed regularly. Ideally, full backups should be performed daily, but if a database is too large to be backed up during the available backup window, then some combination of full backups and differential backups should be made as an alternative option. If you do use a combination full backup and differential backup strategy, you will need to set the last full backup configuration option to match your full backup schedule, otherwise the **Full backup ovrdue** recommendation will continue to be raised. For example, if you make a full backup once a week, and differential backups daily, then you should set the last full backup time to seven days.

Backups can be implemented using Transact-SQL code inside of scheduled SQL Server Agent Jobs, or they can be made using Maintenance Plans.

If your backup window does not allow you to perform a daily full backup, consider using a third-party tool to backup your databases. Through compression, using SQL Backup to back up your databases can reduce backup time up to 90%, depending on the type of data stored in the database. In many cases, the use of SQL Backup will allow you to perform full backups in almost any backup window, negating the need to for a combination full backup and differential backup strategy.

Links to more information

Creating backup jobs in SQL Server 2005
(http://www.sql-server-performance.com/articles/dba/creating_backup_jobs_p1.aspx)

SQL Server backup checklist
(http://www.mssqltips.com/tip.asp?tip=1284)

Information about SQL Backup from Red Gate
(http://www.red-gate.com/products/SQL_Backup/index.htm)

Backing up your database

The **No backup exists for database** recommendation is raised when there are no entries in the `[msdb].[dbo].[backupfile]` system table for any backup of a monitored database.

Backups let you restore data after a failure. With good backups, you can recover from many failures, such as:

- Media failure
- User errors, for example, dropping a table by mistake
- Hardware failures, for example, a damaged disk drive or permanent loss of a server
- Natural disasters

Additionally, backups of a database are useful for routine administrative purposes, such as copying a database from one server to another, setting up database mirroring, and archiving

Links to more information

MSDN: Backing up and restoring databases in SQL Server
(http://msdn.microsoft.com/en-us/library/ms187048.aspx)

Backup overview SQL Server 2005
(http://msdn.microsoft.com/en-us/library/ms175477.aspx)

Introduction to SQL Server 2005 Database Backups
(http://www.sqlteam.com/article/introduction-to-sql-server-database-backups)

MSDN: How to backup a database using SSMS
(http://msdn.microsoft.com/en-us/library/ms187510.aspx)

Information about SQL Backup from Red Gate
http://www.red-gate.com/products/SQL_Backup/index.htm)

About integrity checks

The **Integrity check overdue** recommendation is raised when:

- No DBCC CHECKDB integrity checks are found in the current SQL Server error log for a monitored database, or
- The last DBCC CHECKDB integrity check for a monitored database found in the current SQL Server error log is older than a time you specify.

Running an integrity check

The DBCC CHECKDB command can be run as a SQL Server Agent job as part of a Transact-SQL script, or it can be included as part of a Database Maintenance Plan. Running the command as part of a Transact-SQL script provides you with more options.

While data corruption errors in SQL Server databases are not common, they occur and you must be prepared for them. As a best practice, the DBCC CHECKDB command should be run for every production and system database as often as a full backup for the database is made. For example, if you perform a full database backup daily, then you should run the DBCC CHECKDB command daily. If you perform a full database backup weekly, then you should run the DBCC CHECKDB command at least weekly.

DBCC CHECKDB should be run before performing a full backup, not after. The reason for this is because if there are any problems with the database, you want to know about them before backing up the database, as there is little value backing up a database that has corruption.

Reviewing the results of an integrity check

After the DBCC CHECKDB command has executed, you must check the results to see if any problems have been detected. Some simple problems detected by DBCC CHECKDB, such as corrupted non-clustered indexes, can be easily fixed, while many other corruption problems detected by DBCC CHECKDB cannot be fixed, and the only solution is to restore your backups. This is another good reason to have a comprehensive backup strategy in place.

Note: Running DBCC CHECKDB can be a resource intensive task. Because of this, it should only be run on a production server when it is less busy than usual.

Variations of DBCC CHECKDB

While there are many different variations of the DBCC CHECKDB command, the most useful variation is:

```
DBCC CHECKDB ('DATABASE_NAME') WITH NO_INFOMSGS, ALL_ERRORMSGS;
```

This variation of the command returns no data if there are no problems. If data is returned, a problem of some sort has been detected and needs to be immediately identified and corrected.

If you find that the above variation of the `DBCC CHECKDB` command takes more resources (and time) to execute than you can spare, then use the following option:

```
DBCC CHECKDB ('DATABASE_NAME') WITH NO_INFOMSGS, ALL_ERRORMSGS,
PHYSICAL_ONLY;
```

The addition of the `PHYSICAL_ONLY` option limits integrity checking to the physical structure of pages and record headers, the physical structure of B-trees, and the allocation consistency of the database. It uses minimal overhead, but will catch obvious data corruption problems, such as torn pages, checksum failures, and other hardware-related data integrity issues

Full syntax

```
DBCC CHECKDB

[

    [ ( 'database_name' | database_id | 0

        [ , NOINDEX

        | , { REPAIR_ALLOW_DATA_LOSS | REPAIR_FAST | REPAIR_REBUILD } ]

        ) ]

    [ WITH

        {

            [ ALL_ERRORMSGS ]

            [ , NO_INFOMSGS ]

            [ , TABLOCK ]

            [ , ESTIMATEONLY ]

            [ , { PHYSICAL_ONLY | DATA_PURITY } ]

        }

    ]

]
```

Further information

SQL Server 2005/2008 MS SQL Server Developer Center Integrity Check topic (http://msdn.microsoft.com/en-us/library/ms176064.aspx)

SQL Server 2000 MS SQL Server Developer Center Integrity Check topic (http://msdn.microsoft.com/en-us/library/aa258278(SQL.80).aspx)

About log backups

The **Log backup overdue** recommendation is raised when:

- There are no entries in the `[msdb].[dbo].[backupfile]` system table for a transaction log backup of a monitored database, or
- The last transaction log backup of a monitored database in the `[msdb].[dbo].[backupfile]` system table is older than a time you specify.

This recommendation is only raised for databases set to the Full Recovery or Bulk-Logged Recovery models. It will not be raised for the `master` database.

What is the transaction log?

A database's transaction log (the LDF file) is a critical component of SQL Server and is used to record all DML changes (INSERT, UPDATE, DELETE) and DDL (database structure) changes immediately after they occur, even before these changes are written permanently to the database (the MDF file).

The transaction log is used to help ensure database consistency should a server failure occur, allowing incomplete transactions to be rolled back, or completed transactions to be rolled-forward, when the database comes back online and completes the recovery process. Transaction log activity occurs automatically and cannot be turned off.

Recovery models

- If a database is set to the **Simple Recovery model,** the transaction log continues to work, but it cannot be backed up. This means that should a database need to be restored, then it must be restored from a full backup (or a combination of a full and differential backup), and that any changes to the database that were made to the database after the last full or differential backup are lost. This option should not be used for production databases.
- If a database is set to the **Full** or the **Bulk-Logged Recovery models**, then the transaction log can be backed up, and should be backed up. As a best practice, all production databases should include a backup strategy that includes a regular full backup (daily if possible), differential backups (only if necessary), and transaction log backups.

Why run a log backup?

The main reason transaction log backups should be made is that they can be made more frequently than full or differential backups, which means that more of your data is backed up, and should you ever have to restore your backups, you reduce the amount of data that could potentially be lost.

For example, one backup strategy could be to perform a full database log backup daily, and a transaction log backup hourly. This way, should the database need to be restored, in the worst case, no more than an hour's worth of data would be lost.

Another very important reason for performing transaction log backups is that they are required in order to prevent transaction logs from growing so large that you run out of disk space. When a database is set to the Full Recovery or Bulk-Logged Recoery model, the transaction log will continue to grow until it is backed up, manually truncated, or the disk runs out of space. Obviously, backing up the transaction log is the ideal solution, and is considered the best practice.

Another thing to keep in mind is that when a transaction log is backed up and the older data is truncated from the log file, the physical size of the log file does not decrease. When a transaction log is truncated manually, or by taking a transaction log backup, the inactive data is marked so that it can be overwritten as needed, but no physical free space is created. If you need to shrink an oversized transaction log, you should manually shrink it using `DBCC SHRINKFILE`.

How often to run log backups

How often you should perform transaction log backups depends on how much data you are willing to potentially lose, while taking into account the physical impact of taking the transaction log backups. While transaction log backups rarely impact production systems, it is possible that they can. Some people take transaction log backups every 4 hours, some every hour, and some every 15 minutes.

Impact of running a log backup

In most cases, transaction log backups are quick and take few server resources, but there are exceptions. If your transaction log backup takes more time and resources than you prefer, and they are negatively impacting a production system, consider using SQL Backup to backup your transaction logs. By using compression, the amount of time it takes to execute a transaction log backup can be reduced by up to 90%.

Full syntax

```
BACKUP LOG { database_name | @database_name_var }
{
  TO < backup_device > [ ,...n ]
  [ WITH
    [ BLOCKSIZE = { blocksize | @blocksize_variable } ]
    [ [ , ] DESCRIPTION = { 'text' | @text_variable } ]
    [ [ ,] EXPIREDATE = { date | @date_var }
    | RETAINDAYS = { days | @days_var } ]
    [ [ , ] PASSWORD = { password | @password_variable } ]
    [ [ , ] FORMAT | NOFORMAT ]
    [ [ , ] { INIT | NOINIT } ]
    [ [ , ] MEDIADESCRIPTION = { 'text' | @text_variable } ]
    [ [ , ] MEDIANAME = { media_name | @media_name_variable } ]
    [ [ , ] MEDIAPASSWORD = { mediapassword | @mediapassword_variable } ]
    [ [ , ] NAME = { backup_set_name | @backup_set_name_var } ]
    [ [ , ] NO_TRUNCATE ]
    [ [ , ] { NORECOVERY | STANDBY = undo_file_name } ]
    [ [ , ] { NOREWIND | REWIND } ]
    [ [ , ] { NOSKIP | SKIP } ]
    [ [ , ] { NOUNLOAD | UNLOAD } ]
    [ [ , ] RESTART ]
    [ [ , ] STATS [ = percentage ] ]
  ]
}

< backup_device > ::=
 {
    { logical_backup_device_name | @logical_backup_device_name_var }
    |
    { DISK | TAPE } =
      { 'physical_backup_device_name' | @physical_backup_device_name_var }
 }

< file_or_filegroup > ::=
 {
   FILE = { logical_file_name | @logical_file_name_var }
   |
   FILEGROUP = { logical_filegroup_name | @logical_filegroup_name_var }
 }
```

More help on this issue

SQL Server 2000 : How to create a transaction log backup
(http://msdn.microsoft.com/en-us/library/aa176744(SQL.80).aspx)

SQL Server 2005: How to create a transaction log backup
(http://msdn.microsoft.com/en-us/library/ms191284(SQL.90).aspx)

Information about SQL Backup from Red Gate
(http://www.red-gate.com/products/SQL_Backup/index.htm)

About page verification

The **Page verification is off** recommendation is raised when:

- The database option PAGE_VERIFY is set to NONE for a monitored database in SQL Server 2005 or SQL Server 2008, or
- The database option TORN_PAGE_DETECTION in SQL Server 7.0 or SQL Server 2000 is set to FALSE.

Overview of page corruption

Between the time SQL Server writes a page to disk, then later reads the same page, it is possible that the data stored in the page may get corrupted due to circumstances outside the control of SQL Server. It is a rare occurrence, but when it does occur, it can be devastating to a database, often requiring that the entire database be restored. While SQL Server cannot prevent corruption outside of its control, it does at least have the ability to identify corrupt data. While this won't prevent or fix the problem, it can make the DBA aware of the problem so that it can be addressed as quickly as possible.

Identifying corrupt pages in SQL Server 2000

In SQL Server 2000, the only option available to identify corrupt data pages is to set the database option TORN_PAGE_DETECTION to TRUE. In SQL Server 2005 and SQL Server 2008, Microsoft changed how data corruption was detected. A new database setting, PAGE_VERIFY replaced the older torn page detection option, which allows the DBA to choose from two different types of page verification: TORN_PAGE_DETECTION and CHECKSUM. As a best practise, some form of page verification should be turned on for all production and system databases. While the act of page verification incurs some overhead, the overhead is minimal, and the benefit it provides (helping to identify corrupt data) is more than worth the extra overhead.

Identifying corrupt pages in SQL Server 2005 and 2008

In SQL Server 2005 and SQL Server 2008, you can choose from one of three PAGE_VERIFY options:

- NONE
- CHECKSUM
- TORN_PAGE_DETECTION

If this option is set to NONE, no page verification is done and there is no way SQL Server can detect corrupt data in a page, other than the case where SQL Server is unable to read the page from disk and returns a disk read error.

Setting this option to NONE is not a recommended best practise; hence SQL Response raises the **Page verification is off** recommendation of for all monitored databases that don't use some form of page verification.

Checksum or torn page detection?

Of the two available PAGE_VERIFY options that check for corrupt data, which is best: CHECKSUM or TORN_PAGE_DETECTION?

CHECKSUM offers better overall corruption checking than TORN_PAGE_DETECTION, but it also incurs slightly more overhead. So the trade-off you face is between the effectiveness of the page verification option and the amount of overhead. In either case, the overhead is very slight, and in almost all cases, the CHECKSUM option is recommended. That is why it is the default setting for new databases that are created in SQL Server.

The TORN_PAGE_DETECTION option might be considered if your SQL Server has a significant disk I/O bottleneck, and you have tried every other available disk I/O turning option, and this is the only option you have left to help alleviate the problem.

By default, when a new database is created in SQL Server 2005 or SQL Server 2008, the CHECKSUM database option is turned on.

More help on this issue

MSDN SQL Server Storage Engine blog article on Checksum in SQL Server 2005 (http://blogs.msdn.com/sqlserverstorageengine/archive/2006/06/29/Enabling-CHECKSUM-in-SQL2005.aspx)

About free space in data files

The **Data file has excessive free space** recommendation is raised when:

- The percentage of free space in any data file of a monitored database exceeds a specified value (default threshold: 75%), and
- The data file is above a specified size (default threshold: 100MB)

You can adjust both these thresholds.

Having excessive free space in a database file is rarely a problem, and in most cases is a good thing. Because of this, you may want to disable this recommendation.

Managing space in your data files

Generally speaking, as a best practice, whenever a new database (MDF file) is created, the amount of space that you think will be needed to store the data for the upcoming year (perhaps longer) should be pre-allocated. This serves several purposes:

- It forces you to plan for the future. It is better to know what will happen with regard to data growth instead of being surprised. In addition, it allows you to consider if, and how, you want to implement the following options: selecting ideal file locations, using multiple files, using filegroups, and using partitions.
- Pre-allocating space helps to ensure that when the physical MDF file is created, it is created on contiguous clusters, reducing or eliminating physical file fragmentation.
- Pre-allocating space reduces or eliminates the need for Autogrowth to grow the file as new data is added. Autogrowth can contribute to unplanned resource usage, and can lead to physical file fragmentation. Autogrowth should only be used as a safety valve to allow a database to grow should you accidently run out of space. It should not be used to manage your MDF file growth.

Tips on using this recommendation

One way that you might use the **Data file has excessive free space** recommendation is when your SQL Server has limited space, and you need to maximize disk space utilization by minimizing unused space found in MDF files. A better solution would be to add additional disk space.

Links to more information

ASPFAQ.com: How do I reclaim space in SQL Server? (http://sqlserver2000.databases.aspfaq.com/how-do-i-reclaim-space-in-sql-server.html)

SQL Server Developer Center: Shrinking a database (SQL Server 2005) (http://msdn.microsoft.com/en-us/library/ms189080(SQL.90).aspx)

About free space in log files

The **Transaction log has excessive free space** recommendation is raised when:

- The percentage of free space in the transaction log file of a monitored database exceeds a specified value (default threshold: 75%), and
- The log file is above a specified size (default threshold: 100MB)

You can adjust both these thresholds.

Having excessive free space in a database's transaction log file is rarely a problem, and in most cases is a good thing. Because of this, you may want to disable this recommendation.

Managing the size of the transaction log

Generally speaking, as a best practice, whenever a new database (MDF file) and its associated transaction log file (LDF file) are created, the transaction log file should be pre-allocated to the size you think it will need to be when the system goes into live service. This size will depend on the number of transactions you expect, and how often you backup the transaction log. Pre-allocating transaction log disk space serves several purposes:

- It forces you to plan for the future. It is better to know what will happen with regard to transaction log growth instead of being surprised. In addition, it allows you to consider if, and how, you want to locate the physical transaction log files, along with determining the ideal number of physical files that should be used for each transaction log.
- Pre-allocating space helps to ensure that when the physical LDF file is created, that it is created on contiguous clusters, reducing or eliminating physical file fragmentation.
- Pre-allocating space reduces or eliminates the need for Autogrowth to grow the file as data is added. Autogrowth can contribute to unplanned resource usage, and can lead to physical file fragmentation. Autogrowth should only be used as a safety valve to allow a transaction log file to grow should you accidently run out of space. It should not be used to manage your LDF file growth.

Tips on using this recommendation

One reason you might use the **Log file has excessive free space** recommendation is if your SQL Server has limited space, and you need to maximize disk space utilization by minimizing unused space found in LDF files. A better solution would be to add additional disk space.

In addition, if your server experiences an unexpected transaction that causes the transaction log file to grow excessively, and this event is not expected to occur again, then this recommendation can be useful to point this out once the transaction log has been backed up and truncated. At this point, you could then manually shrink the transaction log file back to its normal size. On the other hand, if such events occur occasionally, then the transaction log file should not be shrunk, but left at its current size, ready for these occasional bursts of activity.

Links to more information

SQL Server 2000:Shrinking the transaction log (http://msdn.microsoft.com/en-us/library/aa174524(SQL.80).aspx)

MS: How to stop the transaction log of a SQL Server database from growing unexpectedly (http://support.microsoft.com/kb/873235 (http://support.microsoft.com/kb/873235)

MS: How to shrink the transaction log file in SQL Server 2005 (http://support.microsoft.com/kb/907511 (http://support.microsoft.com/kb/907511)

Managing the size of the Alert Repository data files on disk

Every time an alert is raised, one or more alert files are created containing details about the alert. These alert files are stored on the same computer that runs the Alert Repository service.

When SQL Response has been running for several days or weeks, the number of alert files stored in the data store folder could grow to be quite large. This takes up a lot of disk space, and may adversely affect the performance of SQL Response.

To keep the disk space used at a manageable level, you can choose to automatically delete alert files older than a specified date:

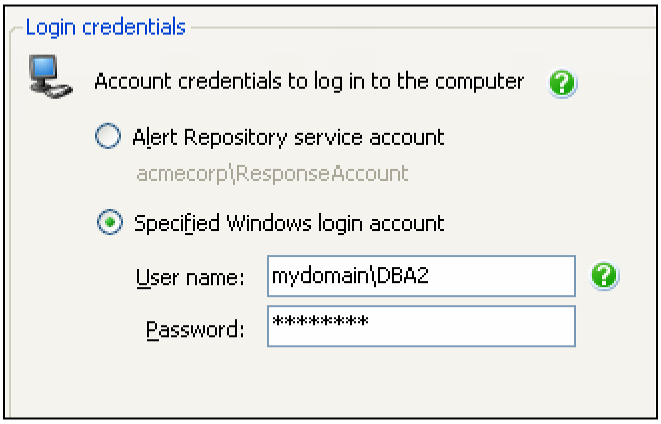
1. From the **Alert Repository** menu in the SQL Response client, select **Alert Repository Options**.
2. Select **Automatically delete old Alert Repository files** and specify when to delete old files.

SQL Response checks every few minutes to remove any files that are older than the time you specify.

Changing the Windows login credentials for a specific computer

By default, SQL Response uses the Alert Repository service account to connect to all monitored SQL Servers. This is the account specified during installation of the Alert Repository.

When the operating system on the computer running the Alert Repository service is supported, SQL Response allows you to supply a **Specified Windows login account** for a particular computer in the **Login credentials** area of the Server Properties dialog box:



Login credentials

Account credentials to log in to the computer

○ Alert Repository service account

acmecorp\ResponseAccount

● Specified Windows login account

User name: mydomain\DBA2

Password: *********

To display the Server Properties dialog box, right-click on the instance in the **Servers to show** area, and select **Server Properties**.

Supported operating systems on the computer running the Alert Repository

Only the following operating systems are supported for specifying a different Windows account:

- Windows XP 32-bit SP3 +
- Windows XP 64-bit SP2 +
- Windows Server 2003 SP2 +
- Windows Server 2008 (all versions)
- Windows Vista (all versions)
- Windows 7 (currently still at BETA)

When the operating system on the computer you are using to run the Alert Repository is **not** supported, SQL Response does not allow you to specify an alternative Windows account to use for logging in to a particular computer. If this is the case, then only the Alert Repository service account can be used.

Note: the list of supported operating systems only applies when changing the account details for the Windows login to a specific computer. If you do not need to do this, you can leave your SQL Response setup as it is.

Credentials not valid for this server?

If the credentials are not valid to connect to this server, you will need to do one of the following:

- Change the Alert Repository service account or give this account permissions to connect to this computer
- Upgrade the operating system on the computer running the Alert Repository to one of the supported systems listed above, then specify a different account for connecting to this computer.

  When it recognizes a supported operating system on the computer running the Alert Repository service, the SQL Response interface includes the option to specify a different account for each computer (as shown above).

Changing the Alert Repository service account

To change the account details for the Alert Repository service, edit the service properties on the computer where the Alert Repository was installed.

The Alert Repository service must be run under an administrator account.

For more information see Changing the Alert Repository service account (http://www.red-gate.com/supportcenter/Content.aspx?p=SQL%20Response&amp;c=SQL_Response\articles\SR_Network_issues.htm#o6238).

Reducing deadlocks

Deadlocking occurs when two or more user processes have locks on separate objects and each process is trying to acquire a lock on the object that the other process has locked. When this happens, SQL Server resolves the deadlock by automatically aborting one process, the "victim" process, allowing the other processes to continue.

The aborted transaction is rolled back and an error message is sent to the user of the aborted process. Generally, the transaction that requires the least amount of overhead to roll back is the transaction that is aborted.

Deadlocks can cause a strain on a SQL Server's resources, especially CPU utilization.

Dealing with deadlocks

Most well-designed applications will resubmit the aborted transaction after receiving a deadlock message, which is then likely to run successfully. This process can affect performance, however. If the application has not been written to trap deadlock errors and automatically resubmit deadlocked transactions, users may receive deadlock error messages on their computer.

Tips on avoiding deadlocks

- Ensure the database design is properly normalized.
- Develop applications to access server objects in the same order each time.
- Do not allow any user input during transactions.
- Avoid cursors.
- Keep transactions as short as possible.
    - Reduce the number of round trips between your application and SQL Server by using stored procedures or by keeping transactions within a single batch.
    - Reduce the number of reads. If you do need to read the same data more than once, cache it by storing it in a variable or an array, and then re-reading it from there.

- Reduce lock time. Develop applications that obtain locks at the latest possible time, and release them at the earliest possible time.
- If appropriate, reduce lock escalation by using `ROWLOCK` or `PAGLOCK`.
- If the data being locked is not modified very frequently, consider using `NOLOCK` to prevent locking.
- If appropriate, use the lowest possible isolation level for the user connection running the transaction.
- Consider using bound connections.

Changing default deadlock behavior

When a deadlock occurs, by default, SQL Server chooses a deadlock "victim" by identifying which of the two processes will use the least resources to roll back, and then returns error message 1205. You can change the default behaviour using the following command:

```
SET DEADLOCK_PRIORITY { LOW NORMAL @deadlock_var }
```

- `LOW` tells SQL Server that the current session should be the preferred deadlock victim, not the session that incurs the least rollback resources. The standard deadlock error message 1205 is returned.
- `NORMAL` tells SQL Server to use the default deadlock method.
- `@deadlock_var` is a character variable specifying which deadlock method you want to use. Specify "3" for low, or "6" for normal.

This command is set at runtime for a specified user connection.

Links to more information

SQL Server 2008 Books Online: Detecting and Ending Deadlocks
http://msdn.microsoft.com/en-us/library/ms178104.aspx
(http://msdn.microsoft.com/en-us/library/ms178104.aspx)

Reducing blocks

Blocking occurs when one connection to SQL Server locks one or more records, and a second connection to SQL Server requires a conflicting lock type on the record or records locked by the first connection. This causes the second connection to wait until the first connection releases its locks.

By default, a connection will wait indefinitely for the blocking lock to end.

Possible causes of excessive blocking include:

- long-running queries
- canceling queries without rollback
- changing large numbers of records in a single transaction
- lack of appropriate indexes

Links to more information

http://www.sql-server-performance.com/tips/blocking_p1.aspx (http://www.sql-server-performance.com/tips/blocking_p1.aspx)
(SQL Server Performance: How to Minimize SQL Server Blocking)

http://www.mssqltips.com/tip.asp?tip=1359
(http://www.mssqltips.com/tip.asp?tip=1359)
(Locking and Blocking Scripts in SQL Server 2000 vs SQL Server 2005)

http://technet.microsoft.com/en-us/magazine/cc434694.aspx
(http://technet.microsoft.com/en-us/magazine/cc434694.aspx)
(TechNet: Minimize Blocking in SQL Server)