SQL Server Agent

01/19/2017 • 8 minutes to read • Contributors

In this article

Benefits of SQL Server Agent
SQL Server Agent Components
Security for SQL Server Agent Administration
Related Tasks
See Also

THIS TOPIC APPLIES TO: SQL Server Azure SQL Database (Managed Instance only) Azure SQL Data Warehouse Parallel Data Warehouse

> **Important**
>
> On Azure SQL Database Managed Instance, most, but not all SQL Server Agent features are currently supported. See Azure SQL Database Managed Instance T-SQL differences from SQL Server for details.

SQL Server Agent is a Microsoft Windows service that executes scheduled administrative tasks, which are called *jobs* in SQL Server 2017.

In This Topic

- Benefits of SQL Server Agent
- Components of SQL Server Agent
- Security for SQL Server Agent Administration

Benefits of SQL Server Agent

SQL Server Agent uses SQL Server to store job information. Jobs contain one or more job steps. Each step contains its own task, for example, backing up a database.

SQL Server Agent can run a job on a schedule, in response to a specific event, or on demand. For example, if you want to back up all the company servers every weekday after hours, you can automate this task. Schedule the backup to run after 22:00 Monday through Friday; if the backup encounters a problem, SQL Server Agent can record the event and notify you.

> **Note**
>
> By default, the SQL Server Agent service is disabled when SQL Server 2017 is installed unless the user explicitly chooses to autostart the service.

SQL Server Agent Components

SQL Server Agent uses the following components to define the tasks to be performed, when to perform the tasks, and how to report the success or failure of the tasks.

Jobs

A *job* is a specified series of actions that SQL Server Agent performs. Use jobs to define an administrative task that can be run one or more times and monitored for success or failure. A job can run on one local server or on multiple remote servers.

> **Important**
>
> SQL Server Agent jobs that are running at the time of a failover event on a SQL Server failover cluster instance do not resume after failover to another failover cluster node. SQL Server Agent jobs that are running at the time a Hyper-V node is paused do not resume if the pause causes a failover to another node. Jobs that begin but fail to complete because of a failover event are logged as started, but do not show additional log entries for completion or failure. SQL Server Agent jobs in these scenarios appear to have never ended.

You can run jobs in several ways:

- According to one or more schedules.
- In response to one or more alerts.
- By executing the sp_start_job stored procedure.

Each action in a job is a *job step*. For example, a job step might consist of running a Transact-SQL statement, executing an SSIS package, or issuing a command to an Analysis Services server. Job steps are

managed as part of a job.

Each job step runs in a specific security context. For job steps that use Transact-SQL, use the EXECUTE AS statement to set the security context for the job step. For other types of job steps, use a proxy account to set the security context for the job step.

Schedules

A *schedule* specifies when a job runs. More than one job can run on the same schedule, and more than one schedule can apply to the same job. A schedule can define the following conditions for the time when a job runs:

- Whenever SQL Server Agent starts.
- Whenever CPU utilization of the computer is at a level you have defined as idle.
- One time, at a specific date and time.
- On a recurring schedule.

For more information, see Create and Attach Schedules to Jobs.

Alerts

An *alert* is an automatic response to a specific event. For example, an event can be a job that starts or system resources that reach a specific threshold. You define the conditions under which an alert occurs.

An alert can respond to one of the following conditions:

- SQL Server events
- SQL Server performance conditions
- Microsoft Windows Management Instrumentation (WMI) events on the computer where SQL Server Agent is running

An alert can perform the following actions:

- Notify one or more operators
- Run a job

For more information, see Alerts.

Operators

An *operator* defines contact information for an individual responsible for the maintenance of one or more instances of SQL Server. In some enterprises, operator responsibilities are assigned to one individual. In enterprises with multiple servers, many individuals can share operator responsibilities. An operator does not contain security information, and does not define a security principal.

SQL Server can notify operators of alerts through one or more of the following:

- E-mail
- Pager (through e-mail)
- **net send**

> **Note**
>
> To send notifications by using **net send**, the Windows Messenger service must be started on the computer where SQL Server Agent resides.

> **Important**
>
> The Pager and **net send** options will be removed from SQL Server Agent in a future version of SQL Server. Avoid using these features in new development work, and plan to modify applications that currently use these features.

To send notifications to operators by using e-mail or pagers, you must configure SQL Server Agent to use Database Mail. For more information, see Database Mail.

You can define an operator as the alias for a group of individuals. In this way, all members of that alias are notified at the same time. For more information, see Operators.

Security for SQL Server Agent Administration

SQL Server Agent uses the **SQLAgentUserRole**, **SQLAgentReaderRole**, and **SQLAgentOperatorRole** fixed database roles in the **msdb** database to control access to SQL Server Agent for users who are not members of the **sysadmin** fixed server role. In addition to these fixed database roles, subsystems and proxies help database administrators ensure that each job step runs with the minimum permissions required to perform its task.

Roles

Members of the **SQLAgentUserRole**, **SQLAgentReaderRole**, and **SQLAgentOperatorRole** fixed database roles in **msdb**, and members of the **sysadmin** fixed server role have access to SQL Server

Agent. A user that does not belong to any of these roles cannot use SQL Server Agent. For more information on the roles used by SQL Server Agent, see Implement SQL Server Agent Security.

Subsystems

A subsystem is a predefined object that represents functionality that is available to a job step. Each proxy has access to one or more subsystems. Subsystems provide security because they delimit access to the functionality that is available to a proxy. Each job step runs in the context of a proxy, except for Transact-SQL job steps. Transact-SQL job steps use the EXECUTE AS command to set the security context.

SQL Server defines the subsystems listed in the following table:

Subsystem name	Description
Microsoft ActiveX Script	Run an ActiveX scripting job step. **Warning** The ActiveX Scripting subsystem will be removed from SQL Server Agent in a future version of Microsoft SQL Server. Avoid using this feature in new development work, and plan to modify applications that currently use this feature.
Operating System (**CmdExec**)	Run an executable program.
PowerShell	Run a PowerShell scripting job step.
Replication Distributor	Run a job step that activates the replication Distribution Agent.
Replication Merge	Run a job step that activates the replication Merge Agent.
Replication Queue Reader	Run a job step that activates the replication Queue Reader Agent.
Replication Snapshot	Run a job step that activates the replication Snapshot Agent.
Replication Transaction Log Reader	Run a job step that activates the replication Log Reader Agent.
Analysis Services Command	Run an Analysis Services command.
Analysis Services Query	Run an Analysis Services query.
SSIS package execution	Run an SSIS package.

> **Note**
>
> Because Transact-SQL job steps do not use proxies, there is no SQL Server Agent subsystem for Transact-SQL job steps.

SQL Server Agent enforces subsystem restrictions even when the security principal for the proxy would normally have permission to run the task in the job step. For example, a proxy for a user that is a member of the sysadmin fixed server role cannot run an SSIS job step unless the proxy has access to the SSIS subsystem, even though the user can run SSIS packages.

Proxies

SQL Server Agent uses proxies to manage security contexts. A proxy can be used in more than one job step. Members of the **sysadmin** fixed server role can create proxies.

Each proxy corresponds to a security credential. Each proxy can be associated with a set of subsystems and a set of logins. The proxy can be used only for job steps that use a subsystem associated with the proxy. To create a job step that uses a specific proxy, the job owner must either use a login associated with that proxy or be a member of a role with unrestricted access to proxies. Members of the **sysadmin** fixed server role have unrestricted access to proxies. Members of **SQLAgentUserRole**, **SQLAgentReaderRole**, or **SQLAgentOperatorRole** can only use proxies to which they have been granted specific access. Each user that is a member of any of these SQL Server Agent fixed database roles must be granted access to specific proxies so that the user can create job steps that use those proxies.

Related Tasks

Use the following steps to configure SQL Server Agent to automate SQL Server administration:

1. Establish which administrative tasks or server events occur regularly and whether these tasks or events can be administered programmatically. A task is a good candidate for automation if it involves a predictable sequence of steps and occurs at a specific time or in response to a specific event.

2. Define a set of jobs, schedules, alerts, and operators by using SQL Server Management Studio, Transact-SQL scripts, or SQL Server Management Objects (SMO). For more information, see Create Jobs.

3. Run the SQL Server Agent jobs you have defined.

> **Note**
>
> For the default instance of SQL Server, the SQL Server service is named SQLSERVERAGENT. For named instances, the SQL Server Agent service is named SQLAgent$*instancename*.

If you are running multiple instances of SQL Server, you can use multiserver administration to automate tasks common across all instances. For more information, see Automated Administration Across an Enterprise.

Use the following tasks to get started with SQL Server Agent:

Description	Topic
Describes how to configure SQL Server Agent.	Configure SQL Server Agent
Describes how to start, stop, and pause the SQL Server Agent service.	Start, Stop, or Pause the SQL Server Agent Service
Describes considerations for specifying an account for the SQL Server Agent service.	Select an Account for the SQL Server Agent Service
Describes how to use the SQL Server Agent error log.	SQL Server Agent Error Log
Describes how to use performance objects.	Use Performance Objects
Describes the Maintenance Plan Wizard, which is a utility that you can use to help create jobs, alerts, and operators to automate administration of an instance of SQL Server.	Use the Maintenance Plan Wizard
Describes how to automate administrative tasks using SQL Server Agent.	Automated Administration Tasks (SQL Server Agent)

See Also

Surface Area Configuration

> **Note**
>
> The feedback system for this content will be changing soon. Old comments will not be carried over. If content within a comment thread is important to you, please save a copy. For more information on the upcoming change, we invite you to read our blog post.

0 Comments

Sign in

10 people listening