

Create and manage SQL Agent jobs

1. Introduction

<https://learn.microsoft.com/en-us/training/modules/schedule-tasks-using-sql-server-agent/1-introduction>

Introduction

Completed

Database systems require regular maintenance, including tasks like backups and updating statistics. Maintenance may also involve scheduled jobs that run against a database. Common examples include extracting, transforming, and loading data from a transaction processing system into a data warehouse. In SQL Server and Azure SQL Managed Instance, the SQL Server Agent service allows you to schedule these maintenance tasks.

Azure offers built-in resource monitoring, and you can also use the platform's options for handling and responding to events, enhancing your ability to maintain and manage your database systems effectively.

Most SQL Server Agent features are supported in Azure SQL Managed Instance, but there are some minor T-SQL differences between SQL Server and Azure SQL Managed Instance that might affect job scripts.

Learning objectives

At the end of this module, you'll understand:

- What maintenance activities you should perform on your databases
- How to configure notifications and alerts on SQL Server Agent jobs and SQL Server
- How to configure notifications alerts based on performance monitor values

2. Create a SQL Server maintenance plan

Create a SQL Server maintenance plan

Completed

Typical activities you can schedule for regular SQL Server maintenance include:

- Database and transaction log backups
- Database consistency checks
- Index maintenance
- Statistics updates

It's crucial to understand the importance of backups, as well as index and statistics maintenance, for all your databases. Database consistency checks, also known as **CHECKDB** (using the command **DBCC CHECKDB**), are equally important because they are the only way to check an entire database for corruption. Depending on the size of your databases and your uptime requirements, you might perform all these activities nightly. However, in production systems, maintenance operations are often spread out over the week, as both index maintenance and consistency checks are very I/O intensive and typically done during weekend hours.

Many DBAs stagger backups of large databases, performing one full backup a week and using differential and transaction log backups to manage recovery to a specific point in time. SQL Server offers a built-in way to manage all these tasks using Maintenance Plans. Maintenance Plans create a workflow of tasks to support your databases and are created as Integration Services packages, allowing you to schedule your maintenance activities. Additionally, many DBAs use open-source scripts for database maintenance to gain more flexibility and control over maintenance activities.

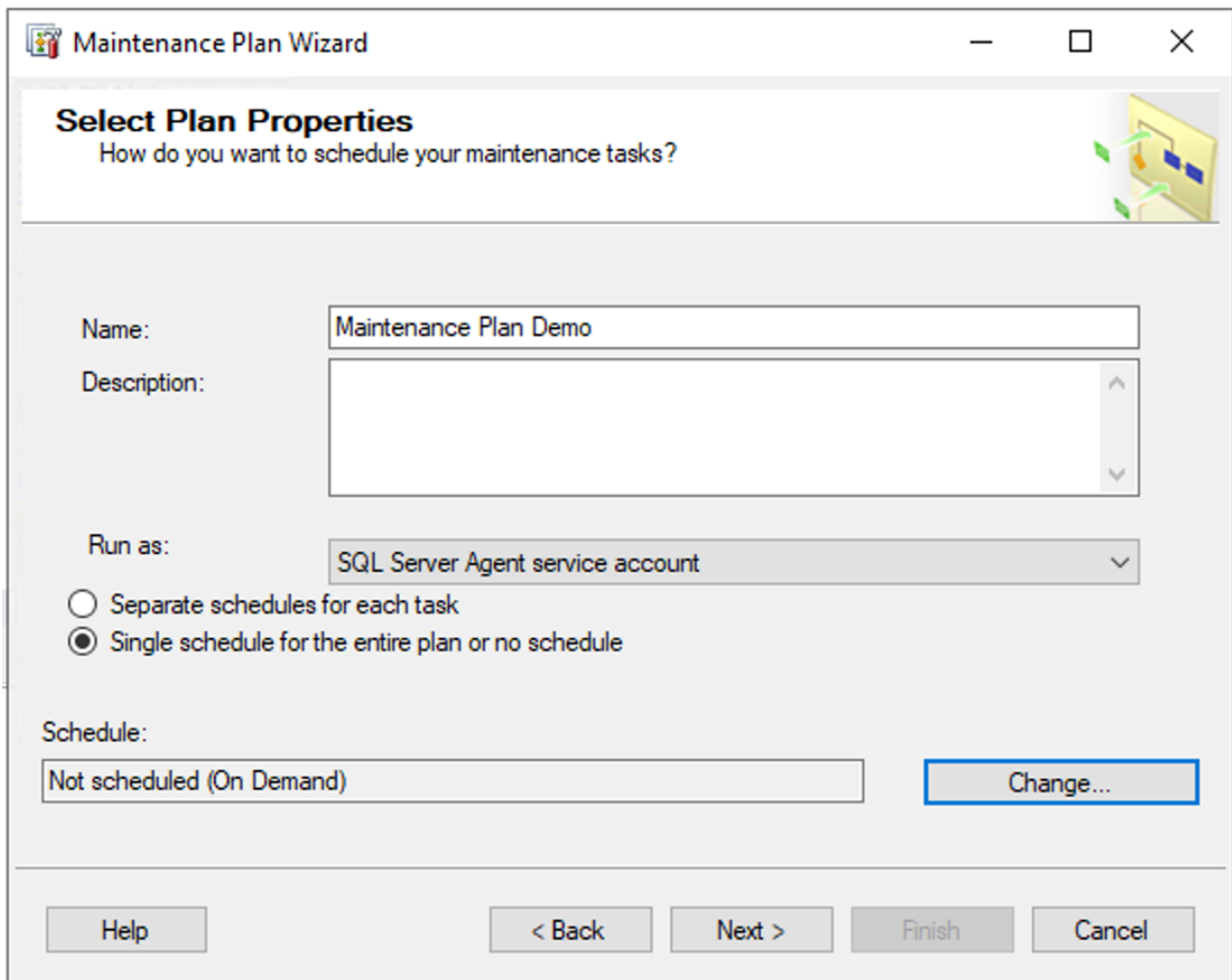
Best practices for maintenance plans

Maintenance plans not only help you perform database maintenance but also offer options to prune data from the **msdb** database, which serves as the data store for the SQL Server Agent. Also, maintenance plans allow you to specify the removal of older database backups from disk. Removing old backup files reduces the size of your backup volume and helps manage the size of the **msdb** database.

Ensure that your backup retention period is longer than your consistency check window. For example, if you run a consistency check weekly, you should retain sufficient backup history to recover from potential corruption detected during consistency checks. Note that the backup operation does not detect corruption in a database, so it is possible to have corruption within a backup file. Maintenance plan activities are scheduled as SQL Server Agent jobs for execution.

Create a maintenance plan

You can create a maintenance plan using SQL Server Management Studio, as shown below. In the example, multiple maintenance tasks are combined into one maintenance plan. However, the best practice is to create a separate maintenance plan for each type of task, and possibly even for specific databases on your server. For instance, you might create one maintenance plan to back up system databases and another to back up user databases. Additionally, you could have a separate maintenance plan for handling the backup of a particularly large user database. The image below and the following examples demonstrate how to create a maintenance plan using the Maintenance Plan Wizard.



The screenshot shows the 'Maintenance Plan Wizard' window with the 'Select Plan Properties' step. The title bar reads 'Maintenance Plan Wizard'. The main heading is 'Select Plan Properties' with the subtitle 'How do you want to schedule your maintenance tasks?'. The form contains the following fields and options:

- Name:** A text box containing 'Maintenance Plan Demo'.
- Description:** A large empty text box.
- Run as:** A dropdown menu showing 'SQL Server Agent service account'.
- Scheduling options:** Two radio buttons: 'Separate schedules for each task' (unselected) and 'Single schedule for the entire plan or no schedule' (selected).
- Schedule:** A dropdown menu showing 'Not scheduled (On Demand)'.
- Buttons:** 'Help', '< Back', 'Next >', 'Finish', and 'Cancel'.
- Change... button:** A button next to the 'Schedule' dropdown, highlighted with a blue border.

The image shows the first screen of the Maintenance Plan Wizard in SQL Server Management Studio (SSMS). You need to specify a name for your maintenance plan and a run-as account. Most maintenance tasks will run as the SQL Server Agent service account, but for security purposes, some tasks may need to run as a different account. For example, if you need to back up to a file share accessible only by a specific account, you would use a proxy user, which is a component of the SQL Server Agent.

What is a proxy account?

A proxy account is an account with stored credentials that the SQL Server Agent can use to execute specific job steps as a designated user. The login information for this user is stored as a credential in the SQL Server instance. Proxy accounts are typically used when specific job steps require very granular security rights.

Suppose you have a SQL Server Agent job that needs to back up a database to a network file share. If the SQL Server Agent service account does not have access to the file share, you can create a proxy account with the necessary permissions. This proxy account can then be used to run the backup step, ensuring it has the required access rights.

Job schedules

Job schedules are part of the job system in the `msdb` system database. SQL Server Agent jobs and schedules have a many-to-many relationship, meaning each job can have multiple schedules, and each schedule can be assigned to multiple jobs. However, the Maintenance Plan Wizard does not allow the creation of independent schedules. Instead, it creates a specific schedule for each maintenance plan.

The following example shows the schedule for a weekly execution, but you also have the option to create a schedule with hourly or daily recurrence.

New Job Schedule

Name:

Schedule type:
☒ Enabled

One-time occurrence

Date:
Time:

Frequency

Occurs:

Recurs every:
 week(s) on

☐ Monday
☐ Wednesday
☐ Friday
☐ Saturday
☒ Sunday
☐ Tuesday
☐ Thursday

Daily frequency

☒ Occurs once at:

☐ Occurs every:
 hour(s)

Starting at:

Ending at:

Duration

Start date:

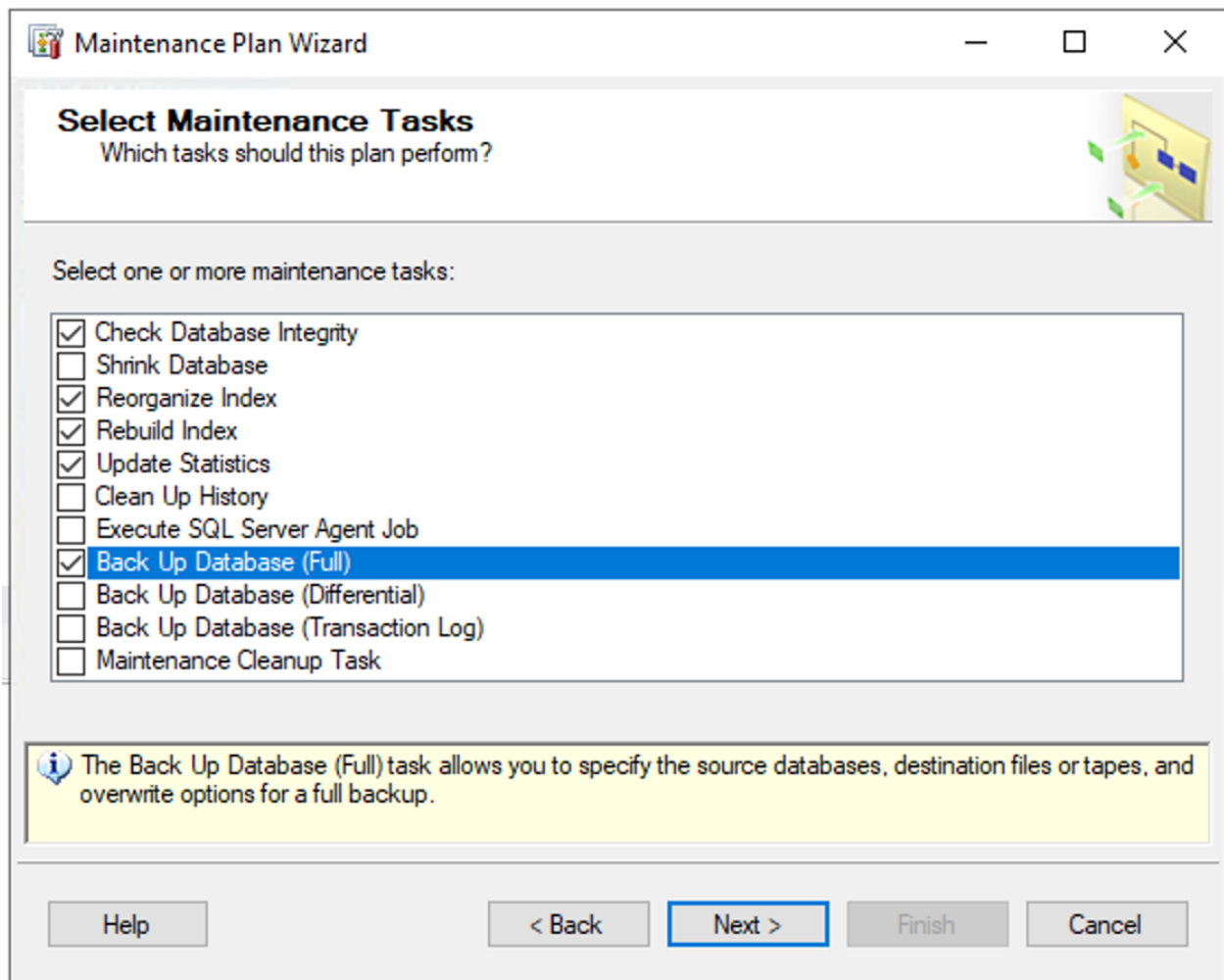
☐ End date:

☒ No end date:

Summary

Description:

The next step is to select the maintenance tasks to add to the plan. The following example shows the operations available to be performed by your maintenance plan.



Check database integrity - This task runs the `DBCC CHECKDB` command to validate the logical and physical consistency of each database page. You should perform this task regularly and align it with your backup retention window. Ensure you complete a consistency check before discarding any prior backups to prevent carrying over corruption.

Shrink database - This task reduces the size of a database or transaction log file by moving data into free space on pages. Once enough space is freed, it can be returned to the file system. It is recommended not to include this action in regular maintenance as it causes severe index fragmentation, harming database performance. The operation is also very I/O and CPU intensive, which can significantly impact system performance.

Reorganize/Rebuild index - This task checks the level of fragmentation in a database's indexes and either rebuilds or reorganizes the index based on the user-defined fragmentation level. Rebuilding an index also updates its statistics.

Update statistics - This task updates the column and index statistics used by SQL Server to build query execution plans. Accurate statistics are crucial for the query optimizer to make the best decisions. You can choose which tables and indexes to scan and the percentage or number of rows to scan. The default sampling rate is usually sufficient, but you may need more detailed statistics for specific tables.

Cleanup history - This task deletes the history of backup and restore operations from the `msdb` database, as well as the history of SQL Server Agent jobs. It helps manage the size of the `msdb` database.

Execute SQL Server Agent job - This task runs a user-defined SQL Server Agent job.

Backup Database (Full/Differential/Log) - This task backs up databases on a SQL Server instance. A full backup captures the entire database and serves as the starting point for a restore. Differential backups capture the pages that have changed since the last full backup, providing an incremental restore point. Transaction log backups capture the active pages in your transaction log, allowing you to define your recovery point objective. Note that transaction log backups cannot be performed on databases in SIMPLE recovery mode.

For example, If you take a full backup on Sunday and a differential backup each weeknight, to restore your database to noon on Thursday, you would restore Sunday's full backup, Wednesday's differential backup, and the transaction log backups from Wednesday's differential to Thursday at noon.

Maintenance Cleanup Tasks - This task removes old files related to maintenance plans, including text reports and backup files. It only removes backups in the specified folders, so any subfolders must be explicitly listed or they will be skipped.

Each task can be scoped to user databases, system databases, or a custom selection of databases, and each has specific configuration options.

Check Database Integrity



Check Database integrity on Local server connection

Databases: All databases

Include indexes

Physical only



Reorganize Index



Reorganize index on Local server connection

Databases: All databases

Object: Tables and views

Compact large objects



Rebuild Index



Rebuild index on Local server connection

Databases: All databases

Object: Tables and views

Original amount of free space



Update Statistics



Update Statistics on Local server connection

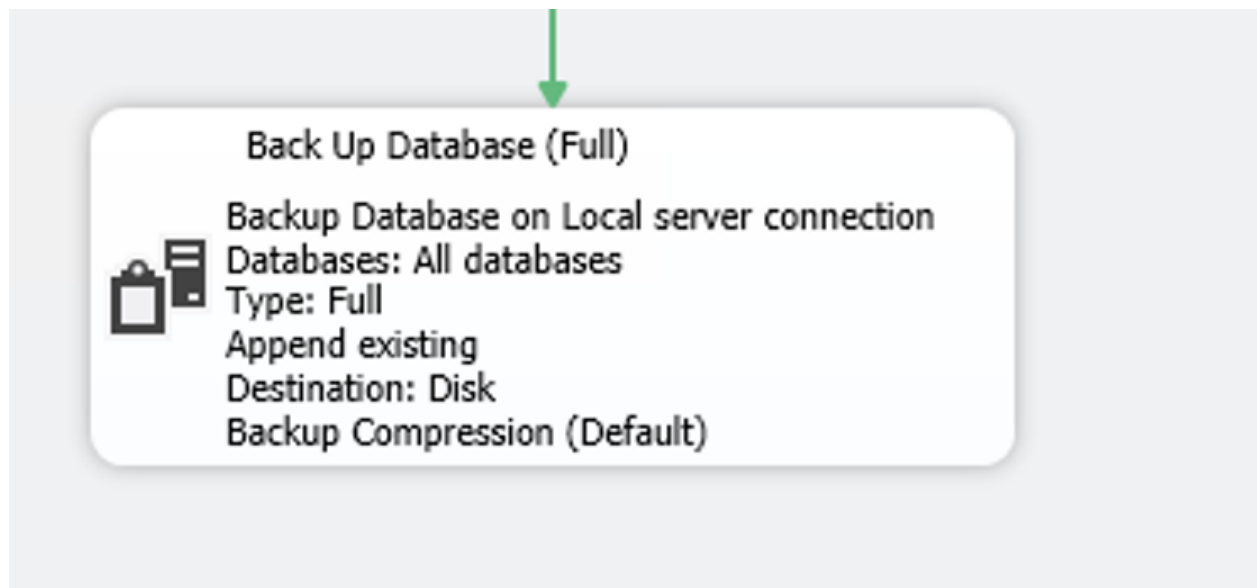
Databases: All databases

Object: Tables and views

All existing statistics

Scan type: Full scan





Upon creation, the plan will appear as a job in the SQL Server Agent. If you added a schedule either during the creation process or after, that job will be executed and the maintenance tasks will be performed.

Multiserver environment

In a [multiserver environment](#), SQL Server Agent allows you to designate one server as a primary server that can execute jobs on other servers, known as target servers. The primary server stores the main source of the jobs and distributes them to the target servers. Target servers periodically connect to the primary server to update their job schedules. This setup enables you to define a job once and deploy it across your enterprise. For example, you can configure database maintenance tasks on the primary server and push them out to a group of target servers, ensuring consistent deployment.

3. Describe task status notifications

<https://learn.microsoft.com/en-us/training/modules/schedule-tasks-using-sql-server-agent/3-describe-task-status-notifications>

Describe task status notifications

Completed

A crucial part of automation is providing notifications for job failures or specific system errors. SQL Server Agent facilitates this through a set of objects, with alerting commonly done via email using SQL Server's Database Mail functionality. The key objects in this workflow are:

- **Operators:** Aliases for individuals or groups who receive notifications.

- **Notifications:** Inform an operator about the completion, success, or failure of a job.
- **Alerts:** Assigned to an operator for either a notification or a defined error condition.

Operators

Operators act as aliases for users or groups configured to receive notifications of job completions or alerts from the error log. An operator is defined by a name and contact information, typically mapping to an email group. Using email groups provides redundancy, ensuring notifications aren't missed if someone is unavailable. It also simplifies updates when employees leave the organization. To send emails to an operator, you need to enable the SQL Server Agent's email profile, as shown below:

The screenshot shows the 'SQL Server Agent Properties - VM1' dialog box. The left pane has 'Select a page' with options: General, Advanced, Alert System, Job System, Connection, and History. The 'Connection' page is selected, showing 'Server: .', 'Connection: VM1\dantoni', and a 'View connection properties' link. The 'Progress' section shows a 'Ready' status with a circular progress indicator. The main pane has tabs for 'Script' and 'Help'. The 'Mail session' section includes a checked 'Enable mail profile' checkbox, a 'Mail system' dropdown set to 'Database Mail', a 'Mail profile' dropdown, a 'Test...' button, and a checked 'Save copies of the sent messages in the Sent Items folder' checkbox. The 'Pager e-mails' section has an 'Address formatting for pager e-mails:' label and fields for 'Prefix:', 'Pager:', and 'Suffix:'. Below these are checkboxes for 'To line:', 'Cc line:', and 'Subject:'. A text area for 'To:' and 'Cc:' is also present. A checked 'Include body of e-mail in notification message' checkbox is below. The 'Fail-safe operator' section has an unchecked 'Enable fail-safe operator' checkbox, an 'Operator:' dropdown, and 'Notify using:' checkboxes for 'Email' and 'Pager'. The 'Token replacement' section has an unchecked 'Replace tokens for all job responses to alerts' checkbox. 'OK' and 'Cancel' buttons are at the bottom right.

Notifications

Notifications are part of each SQL Server Agent job. You can choose to send a notification on job completion, failure, or success. Most DBAs notify only on failure to avoid an influx of notifications for

successful jobs. Notifications depend on an existing operator to send the alert.

Job Properties - Maintenance Plan Demo.Subplan_1

Select a page

- General
- Steps
- Schedules
- Alerts
- Notifications
- Targets

Script Help

Actions to perform when the job completes:

☒ E-mail:	DBA Team	When the job fails
☐ Page:		When the job fails
☒ Write to the Windows Application event log:		When the job fails
☐ Automatically delete job:		When the job succeeds

Connection

Server:

Connection:

VM1\dantoni

[View connection properties](#)

Progress

Ready

OK Cancel

Alerts

SQL Server Agent alerts enable proactive monitoring of your SQL Server. The agent reads the SQL Server error log and notifies an operator when it finds an error number for which an alert is defined. Besides monitoring the error log, you can set up alerts for SQL Server performance conditions and Windows Management Instrumentation (WMI) events. You can specify alerts for one or more events. A common practice is to raise alerts for all SQL Server errors of level 16 and higher and add alerts for specific critical storage errors or Availability Group failovers. Another example is to alert on performance conditions like high CPU utilization or low Page Life Expectancy.

DBAs may also want to be notified of certain server conditions, such as CPU utilization over 90% for five minutes or low Page Life Expectancy. This is done by creating performance condition alerts based on Windows Performance Monitor (perfmon) metrics tracked within the SQL Server database engine. You can access the alert configuration screen by right-clicking **SQL Server Agent** (if it is running) and choosing **New | Alert**.

New Alert

Select a page

- General
- Response
- Options

Connection

Server: .

Connection: DCAC\joey

[View connection properties](#)

Progress

Ready

Script **Help**

Name: ☒ Enable

Type:

Performance condition alert definition

Object:

Counter:

Instance:

Alert if counter Value:

OK **Cancel**

You have options for responding to performance conditions: notify an operator via email, which is the most common approach, or execute another SQL Server Agent job to resolve the issue. Executing another job is useful for well-known conditions that can be handled without manual intervention. For example, you could create an alert for SQL Server storage error conditions (errors 823, 824, 825) and execute a job to perform a database consistency check. Notifications for these alerts use the same SQL Server Agent subsystem.

4. Exercise: Create a CPU status alert for a SQL Server

<https://learn.microsoft.com/en-us/training/modules/schedule-tasks-using-sql-server-agent/4-exercise-create-cpu-status-alert>

Exercise: Create a CPU status alert for a SQL Server

Completed

In this exercise, you'll learn how to create a CPU status alert for a SQL Server on Azure.

Note

To complete this exercise, you'll need an [Azure subscription](#).

Launch the exercise and follow the instructions.

[Launch Exercise](#)

5. Knowledge check

<https://learn.microsoft.com/en-us/training/modules/schedule-tasks-using-sql-server-agent/5-knowledge-check>

Knowledge check

Completed

6. Summary

<https://learn.microsoft.com/en-us/training/modules/schedule-tasks-using-sql-server-agent/6-summary>

Summary

Completed

The SQL Server Agent provides a robust mechanism for managing and maintaining your SQL Server and Azure SQL managed instance deployments. In addition to providing job scheduling, it provides alerting and some monitoring for your databases. Maintenance plans can be used to provide backups, index and statistics maintenance, and log management activity. Note that the SQL Server Agent is only available on SQL Server and Azure SQL Managed Instance, but not Azure SQL Database.

Now that you've reviewed this module, you should be able to:

- Describe what maintenance activities you should perform on your databases
- Describe how to configure notifications and alerts on SQL Server Agent jobs and SQL Server
- Describe how to configure notifications alerts based on performance monitor values