OFFICIAL MICROSOFT LEARNING PRODUCT

20764A

Administering a SQL Database Infrastructure

Product Number: 20764A

Part Number (if applicable):

Released: 05/2016

Module 1

SQL Server Security

Contents:

Lesson 1

Authenticating Connections to SQL Server

Contents:

Question and Answers

Question: Which one of these statements is incorrect?

( ) There are two levels at which you configure Azure firewall rules: Server and Database.

( ) You can only reset passwords using the ALTER LOGIN Transact-SQL statement.

( ) The trusted server application model is commonly used in large-scale enterprise applications, websites, and Internet services.

( ) In a Windows-based environment, administrators can enable policies for Windows users that enforce password complexity and expiration. SQL Server can enforce similar restrictions for SQL Server logins.

( ) To connect to SQL Server, the principal must supply information that matches the credential data held by SQL Server.

Answer:

( ) There are two levels at which you configure Azure firewall rules: Server and Database.

(√) You can only reset passwords using the ALTER LOGIN Transact-SQL statement.

( ) The trusted server application model is commonly used in large-scale enterprise applications, websites, and Internet services.

( ) In a Windows-based environment, administrators can enable policies for Windows users that enforce password complexity and expiration. SQL Server can enforce similar restrictions for SQL Server logins.

( ) To connect to SQL Server, the principal must supply information that matches the credential data held by SQL Server.

Question: True or false? To allow connection to the Azure Database, your local firewall rules must allow TCP Port **1344**.

( ) True

( ) False

Answer:

( ) True

(√) False

Resources

Overview of SQL Server Security

Best Practice: When planning a security solution, consider the following best practices:

- Provide each principal with only the permissions they actually need.
- Use securable inheritance to minimize the number of implicit permissions that must be set to enable the required level of access.
- Use principal containers, such as groups or roles, to create a layer of abstraction between principals and permissions to access securables. You can then use membership of these groups to control access to resources via the permissions you have defined. Changes in personnel should not require changes to permissions.

SQL Server Authentication

Best Practice: If possible, Windows authentication should be used.

Azure SQL Database Firewall

Best Practice: Microsoft recommends using database-level firewall rules. You should consider using server rules if you have many databases with the same access requirements.

Demonstration: Authenticating Logins

Demonstration Steps

Set the Authentication Mode

1. Ensure that the 20764A-MIA-DC and 20764A-MIA-SQL virtual machines are running, and log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In the D:\DemoFiles\Mod01 folder, run **Setup.cmd** as an administrator.
3. Click **Yes** when prompted to confirm that you want to run the command file, and then wait for the script to finish.
4. Start SQL Server Management Studio, and connect to the **MIA-SQL** database engine using Windows authentication.
5. In Object Explorer, right-click the **MIA-SQL** instance, and click **Properties**.
6. In the **Server Properties - MIA-SQL** dialog box, on the **Security** page, verify that **SQL Server and Windows Authentication mode** is selected, and then click **Cancel**.

Create Logins

1. In Object Explorer, expand **Security**, and expand **Logins** to view the logins that are currently defined on this server instance.
2. Right-click **Logins**, and click **New Login**.
3. In the **Login - New** dialog box, next to the **Login** name box, click **Search**.

4. In the **Select User or Group** dialog box, click **Object Types**.
5. In the **Object Types** dialog box, ensure only **Users** and **Groups** are selected, and then click **OK**.
6. In the **Select User or Group** dialog box, click **Locations**.
7. In the **Locations** dialog box, expand **Entire Directory**, select **adventureworks.msft** and click **OK**.
8. In the **Select User**, **Service Account, or Group** dialog box, click **Advanced**. Then click **Find Now**. This produces a list of all users and groups in the Active Directory domain.
9. In the list of domain objects, select **HumanResources_Users** (this is a domain local group that contains multiple global groups, each of which in turn contains users), then click **OK**.
10. In the **Select User, Service Account, or Group** dialog box, ensure that **HumanResources_Users** is listed, and click **OK**.
11. In the **Login - New** dialog box, in the **Default database** drop-down list, select **AdventureWorks**, click **OK**.
12. In Object Explorer, verify that the **ADVENTUREWORKS\HumanResources_Users** login is added to the **Logins** folder.
13. Right-click **Logins** and click **New Login**.
14. In the **Login - New** dialog box, in the **Login name** box, type **Payroll_Application**, and click **SQL Server authentication**.
15. Enter and confirm the password **Pa$$w0rd**, and then clear the **Enforce password expiration** check box (which automatically clears the **User must change password at next login** check box).
16. In the **Default database** drop-down list, select **AdventureWorks**, and then click **OK**.
17. In Object Explorer, verify that the **Payroll_Application** login is added to the **Logins** folder.
18. Open the **CreateLogins.sql** script file in the D:\DemoFiles\Mod01 folder and review the code it contains, which creates a Windows login for the **ADVENTUREWORKS\AnthonyFrizzell** user and the **ADVENTUREWORKS\Database_Managers** local group, and a SQL Server login named **Web_Application**.
19. Click **Execute**. Then, when the script has completed successfully, refresh the **Logins** folder in Object Explorer and verify that the logins have been created.
20. Keep SQL Server Management Studio open for the next demo.

Lesson 2

Authorizing Logins to Connect to Databases

Contents:

Question and Answers

Question: True or false? You can view security tokens using **sys.login_token** and **sys.user_token** system views.

( ) True

( ) False

Answer:

(√) True

( ) False

Demonstration: Authorizing Logins and User Tokens

Demonstration Steps

1. In SQL Server Management Studio, open the **Security Tokens Demo.sql** script file in the D:\DemoFiles\Mod01 folder.

   Note: In some cases, to view the results in SQL Server Management Studio, you might have to right-click the containing node in Object Explorer and select **Refresh** to update the objects.
2. Select the code under **Step A**, and then click **Execute**.
3. Select the code under **Step B**, and then click **Execute**.
4. In Object Explorer, under **MIA-SQL** under **Security**, expand **Server Roles** to view the new server roles.
5. Select the code under **Step C**, and then click **Execute**.
6. In Object Explorer, under **MIA-SQL** under **Security**, under **Logins**, view the new login.
7. Select the code under **Step D**, and then click **Execute**.
8. Select the code under **Step E**, and then click **Execute**.
9. In Object Explorer, under **MIA-SQL** expand **Databases**, expand **AdventureWorks**, expand **Security**, expand **Roles**, and then expand **Database Roles** to view the new database roles.
10. Select the code under **Step F**, and then click **Execute**.
11. In Object Explorer, under **MIA-SQL** under **Databases**, under **AdventureWorks**, under **Security**, expand **Users** to view the new user.
12. Select the code under **Step G**, and then click **Execute**.
13. Select the code under **Step H**, and then click **Execute** to view the current user and login tokens. The code is executed in the security context of the login created for this demonstration. Notice that tokens relating to the database user, both the **MyExtDatabaseRole** and **MyDatabaseRole** database roles, and the **public** database role, are linked to the user.
14. Select the code under **Step I**, and then click **Execute** to remove all changes.
15. Keep SQL Server Management Studio open for the next demo.

Lesson 3
Authorization Across Servers

Contents:

Question and Answers

Question: True or false? The following code will relink any orphaned user, including the **dbo**, for the moved and linked database **My_Database** (assume that **MyDBUser** can be any orphaned user or **dbo**):

```
USE <My_Database>

GO

sp_change_users_login @Action='update_one', @UserNamePattern='<MyDBUser>',

  @LoginName='<MyLogin>';

GO
```

( ) True

( ) False

Answer:

( ) True

(√) False

Demonstration: Working with Mismatched Security IDs

Demonstration Steps

1. In SQL Server Management Studio, open the **MismatchedIDs.sql** script file in the D:\DemoFiles\Mod01 folder.
2. Select the code under **Step A**, and then click **Execute** to run the orphaned users report in the **TSQL** database. Two users should be returned. Note the SID for the **appuser1** user.
3. Select the code under **Step B**, and then click **Execute** to demonstrate that an **appuser** login exists, but has a different SID to that referenced by the **appuser1** user.
4. Select the code under **Step C**, and then click **Execute** to repair the **appuser1** user by linking it to the **appuser** login.
5. Select the code under **Step D**, and then click **Execute** to demonstrate that **appuser1** is no longer an orphaned user. Note the SID for the **reportuser1** user.
6. Select the code under **Step E**, and then click **Execute** to create the **reportuser** login with a defined SID value. The SID matches the SID returned in the orphaned users report for **reportuser1**.
7. Select the code under **Step F**, and then click **Execute** to demonstrate that no orphaned users remain.
8. Keep SQL Server Management Studio open.

Lesson 4

Partially Contained Databases

Contents:

Question and Answers

Question: True or False? The following code can be used to add **SalesMan1** to the **master** database:

```
USE master
GO

CREATE USER SalesMan1 WITH PASSWORD = 'Pa$$w0rd'

GO
```

( ) True

( ) False

Answer:

( ) True

(√) False

Question: Which option does not apply to partially contained databases?

( ) You can use the ALTER statement to convert a non-contained database to a partially contained database.

( ) SQL Server 2016 does not support FULL containment.

( ) Numbered and temporary procedures are supported in partially contained databases.

( ) Replication is not operational in partially contained databases.

( ) CDC and CT are not supported in partially contained databases.

Answer:

( ) You can use the ALTER statement to convert a non-contained database to a partially contained database.

( ) SQL Server 2016 does not support FULL containment.

(√) Numbered and temporary procedures are supported in partially contained databases.

( ) Replication is not operational in partially contained databases.

( ) CDC and CT are not supported in partially contained databases.

Demonstration: Creating a Partially Contained Database

Demonstration Steps

View the Containment Value

1. In SQL Server Management Studio, open **ContainedDatabase.sql** in the D:\Demofiles\Mod01 folder.

   Note: In some cases, to view the results in SQL Server Management Studio, you might have to right-click the containing node in Object Explorer and select **Refresh** to update the objects.

2. Select the code under Step A, and then click Execute.
   Note that the value returned is '1' as containment should be enabled.

3. Select the code under Step B, and then click Execute.
   Note that the value_in_use is ‘0’ (containment is disabled). To confirm this in Object Explorer, right-click MIA-SQL, click Properties, click Advanced and noting the Enable Contained Databases attribute is False, and then click Cancel.
4. Select the code under Step C, and then click Execute.
   Note that the value_in_use is set back to ‘1’. To confirm this in Object Explorer, right-click MIA-SQL, click Properties, click Advanced and noting the Enable Contained Databases attribute is True, and then click Cancel.
5. Select the code under **Step D**, and then click **Execute**.
6. In Object Explorer, under **MIA-SQL**, right-click on **Databases**, and then click **Refresh** to view a list of databases, including the new partially contained database.
7. Select the code under Step E, and then click Execute.
   The new users will be returned from the SELECT statement.
8. In Object Explorer, under **MIA-SQL**, under **Databases**, expand **PClientData**, expand **Views**, and then expand **System Views**, right-click **sys.database_principals**, and then click **Select Top 1000 Rows**, confirm the new users have been created in the list of contained users.
9. In the **ContainedDatabase.sql** tab, select the code under **Step F**, and then click **Execute**.
10. Close SQL Server Management Studio, without saving any changes.

Module Review and Takeaways

Review Question(s)

Question: What happens when a login does not have access to its default database and is used to open a connection to a SQL Server instance?

( ) The login can connect to SQL Server, but an error message is reported.

( ) The login cannot connect to SQL Server.

( ) The login is automatically disabled.

( ) The login is automatically granted access to its default database.

Answer:

( ) The login can connect to SQL Server, but an error message is reported.

(√) The login cannot connect to SQL Server.

( ) The login is automatically disabled.

( ) The login is automatically granted access to its default database.

Lab Review Questions and Answers

Lab: Authenticating Users

Question and Answers

Lab Review

Question: In the last exercise of the lab, why was the **ADVENTUREWORKS\WebApplicationSvc** user not reported as an orphaned user?

> **Answer:** Because it is linked to a login, based on a Windows user account.

Module 2

Assigning Server and Database Roles

Contents:

Lesson 1

Working with Server Roles

Contents:

Question and Answers

Question: Which fixed server role should be regarded as equivalent to **sysadmin** because of its ability to assign server principals to server roles?

( ) serveradmin

( ) securityadmin

( ) processadmin

( ) setupadmin

( ) bulkadmin

Answer:

( ) serveradmin

(√) securityadmin

( ) processadmin

( ) setupadmin

( ) bulkadmin

Resources

Public Server Role

Best Practice: Controlling access to securable server objects by granting additional permissions to **public** is not recommended, because doing so might inadvertently grant more access to every login than is intended.
For the same reason, you should regularly review the permissions granted to the **public** role.

Working with User-Defined Server Roles

Best Practice: It is not recommended that you make user-defined server roles members of fixed server roles. Doing so will give control over membership of the fixed server role to members of the user-defined server role, which may lead to unintended escalation of privileges.

Demonstration: Assigning Fixed and User-Defined Server Roles

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Run **Setup.cmd** in the D:\Demofiles\Mod02 folder as Administrator.
3. Start SQL Server Management Studio and connect to the **MIA-SQL** database engine instance using Windows authentication.
4. On the **File** menu, point to **Open**, and then click **Project/Solution**.
5. In the **Open Project** dialog box, navigate to **D:\Demofiles\Mod02\Project**, click **Project.ssmssln**, and then click **Open**.

6. In Solution Explorer, double-click **Demo 1 – server roles.sql**.
7. Execute the code under the heading for **Step 1** to show the permission hierarchy.
8. Execute the code under the heading for **Step 2** to create two logins.
9. Execute the code under the heading for **Step 3** to show that a new login is a member of **public**.
10. Execute the code under the heading for **Step 4** to demonstrate the permissions of a new login.
11. Execute the code under the heading for **Step 5** to add a login to the **diskadmin** role.
12. Execute the code under the heading for **Step 6** to verify the membership created in the previous step.
13. Execute the code under the heading for **Step 7** to create a user-defined server role.
14. Execute the code under the heading for **Step 8** to grant permissions to the new role.
15. Execute the code under the heading for **Step 9** to make a login a member of the new role.
16. Execute the code under the heading for **Step 10** to verify the membership created in the previous step.
17. Execute the code under the heading for **Step 11** to show the permissions of the login.
18. Execute the code under the heading for **Step 12** to remove the logins and role.
19. On the **File** menu, click **Close**.
20. Keep SQL Server Management Studio open for the next demonstration.

Lesson 2

Working with Fixed Database Roles

Contents:

Question and Answers

Question: Which of the following statements is incorrect (select one)?

( ) dbo and db_owner are terms that refer to the same thing.

( ) dbo is a database user that is an alias for the login that owns the database.

( ) db_owner is a fixed database role with full administrative permissions over a database.

( ) The db_owner role can have more than one member.

( ) By default, dbo is a member of the db_owner role.

Answer:

(√) dbo and db_owner are terms that refer to the same thing.

( ) dbo is a database user that is an alias for the login that owns the database.

( ) db_owner is a fixed database role with full administrative permissions over a database.

( ) The db_owner role can have more than one member.

( ) By default, dbo is a member of the db_owner role.

Demonstration: Managing Database Roles and Users

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In SQL Server Management Studio, in Object Explorer, on the toolbar, click **Connect**, and then click **Database Engine**.
3. In the **Connect to Server** dialog box, in the **Server name** box, type the name of your Azure instance running the **AdventureWorksLT** database, for example, **<servername>.database.windows.net**.
4. In the **Authentication** list, click **SQL Server Authentication**, in the **Login** box, type **Student**, in the **Password** box, type **Pa$$w0rd**, and then click **Connect**.
5. In Solution Explorer, double-click **Demo 2 – database roles.sql**.
6. On the **Query** menu, point to **Connection**, and then click **Change Connection**.
7. In the **Connect to Database Engine** dialog box, in the **Server name** list, click **<servername>.database.windows.net**, in the **Password** box, type **Pa$$w0rd**, and then click **Connect**.
8. Execute the code under the heading **Step 1** to create a new login.
9. On the toolbar, in the **Available Databases** list, click **AdventureWorksLT**.
10. Execute the code under the heading **Step 2** to create a new user.
11. Execute the code under the heading for **Step 3** to demonstrate the database permissions hierarchy.
12. Execute the code under the heading for **Step 4** to demonstrate that the user is a member of the **public** database role by default.
13. Execute the code under the heading for **Step 5** to add the user to two fixed database roles.
14. Execute the code under the heading for **Step 6** to verify the user’s role memberships.

15. On the toolbar, in the **Available Databases** list, click **master**.
16. Execute the code under **Step 7** to remove the demonstration objects.
17. On the **File** menu, click **Close**.
18. Keep SQL Server Management Studio open for the next demonstration.

Lesson 3

User-Defined Database Roles

Contents:

Question and Answers

Put the following steps, which describe the sequence of actions when an application uses an application role, in order by numbering each to indicate the correct order.

	Steps
	An administrator makes the user a member of the database public role.
	The user starts an application, which connects to the database using the user's credentials.
	The application activates an application role, using the password held in the application's configuration file.
	The user works in the application with the security context of the application role.
	The user closes the application.

Answer:

	Steps
1	An administrator makes the user a member of the database public role.
2	The user starts an application, which connects to the database using the user's credentials.
3	The application activates an application role, using the password held in the application's configuration file.
4	The user works in the application with the security context of the application role.
5	The user closes the application.

Demonstration: User-Defined Database Roles

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In SQL Server Management Studio, in Solution Explorer, double-click **Demo 3 – user database roles.sql**.
3. Execute the code under the heading **Step 1** to create a user-defined database role.
4. Execute the code under the heading **Step 2** to grant SELECT permissions on the **HumanResources** schema to the role.
5. Execute the code under the heading **Step 3** to verify the role permissions.
6. Execute the code under the heading **Step 4** to add two users to the role.
7. Execute the code under the heading **Step 5** to verify the role's membership.
8. Execute the code under the heading **Step 6** to remove the demonstration role.

9. On the **File** menu, click **Close**.
10. Keep SQL Server Management Studio open for the next demonstration.

Demonstration: Application Roles

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In SQL Server Management Studio, in Solution Explorer, double-click **Demo 4 – application roles.sql**.
3. Execute the code under the heading **Step 1** to create an application role.
4. Execute the code under the heading **Step 2** to grant permissions to the role.
5. Execute the code under the heading **Step 3** to demonstrate behavior before the application role is activated.
6. Execute the code under the heading **Step 4** to activate the application role.
7. Execute the code under the heading **Step 5** to demonstrate that the application role permissions apply.
8. Execute the code under the heading **Step 6** to show how the user’s identity is represented.
9. Execute the code under the heading **Step 7** to show that cross-database access is limited.
10. Execute the code under the heading **Step 8** to exit the application role.
11. Execute the code under the heading **Step 9** to show how the user’s identity is represented.
12. Execute the code under the heading **Step 10** to remove the application role.
13. On the **File** menu, click **Close**.
14. Close SSMS without saving any changes.

Module Review and Takeaways

Best Practice

When implementing role-based security in SQL Server, consider the following best practices:

- Use Windows group logins linked to roles to simplify ongoing management where possible.
- Aim to grant the minimum number of explicit permissions possible to meet the security requirements, and use membership of roles and inheritance to ensure the correct effective permissions.
- Ensure every database user has only the permission they actually require.

Review Question(s)

Question: True or False? When **sp_setapprole** is called with the @encrypt = 'odbc' parameter, the application role password is encrypted with strong encryption.

( ) True

( ) False

Answer:

( ) True

(√) False

Question: Which permission is required to view the contents of the query plan cache for an on-premises SQL Server instance?

( ) SHOWPLAN (database level)

( ) SHOWPLAN (server level)

( ) VIEW SERVER STATE (server level)

( ) VIEW SERVER STATE (database level)

Answer:

( ) SHOWPLAN (database level)

( ) SHOWPLAN (server level)

(√) VIEW SERVER STATE (server level)

( ) VIEW SERVER STATE (database level)

Lab Review Questions and Answers

Lab: Assigning Server and Database Roles

Question and Answers

Lab Review

Question:

Your organization wants to track data access by individual Windows users. Does this mean you cannot base logins on Windows groups?

Answer: No.

Module 3

Authorizing Users to Access Resources

Contents:

Lesson 1

Authorizing User Access to Objects

Contents:

Question and Answers

Question: What is the REFERENCES permission used for?

> **Answer:** The REFERENCES permission is used before a foreign key relationship can specify the object as a target if no other permissions exist on the referenced object.

Demonstration: Authorizing User Access to Objects

Demonstration Steps

1. Ensure that the MSL-TMG1, MIA-DC, and MIA-SQL virtual machines are running, and log on to MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Run **Setup.cmd** in the D:\Demofiles\Mod03 folder as Administrator.
3. On the taskbar, click the SQL Server Management Studio shortcut.
4. In the **Connect to Server** dialog box, click **Connect**.
5. On the **File** menu, point to **Open**, and then click **File**.
6. In the **Open File** dialog box, go to **D:\Demofiles\Mod03**, click **AuthorizingUserAccessToObjects.sql**, and then click **Open**.
7. Execute the code under the heading for **Step 1** to create a user for the demonstration.
8. Execute the code under the heading for **Step 2** to query the list of server principals. Note Mod03Login at the end of the list.
9. Execute the code under the heading for **Step 3** to query the list of database principals. Again, note Mod03Login in the list.
10. Execute the code under the heading for **Step 4** to grant SELECT permissions on the **Product** table to **Mod03Login**.
11. Execute the code under the heading for **Step 5** to change the execution context.
12. Execute the code under the heading for **Step 6** to test the permissions. Note that you can select from the **Product** table that you were granted permissions on, but not from the **ProductInventory** table.
13. Execute the code under the heading for **Step 7** to revert the execution context.
14. Execute the code under the heading for **Step 8** to grant SELECT permissions on specific columns in the **ProductInventory** table to **Mod03Login**.
15. Execute the code under the heading for **Step 9** to change the execution context.
16. Execute the code under the heading for **Step 10** to test the permissions. Note that the first query to select the two specific columns executes, but you cannot select all the columns from the **ProductInventory** table.
17. Execute the code under the heading for **Step 11** to revert the execution context.
18. On the **File** menu, click **Close**.
19. Leave SQL Server Management Studio open for the next demonstration.

Lesson 2

Authorizing Users to Execute Code

Contents:

Question and Answers

Question: What permission enables a user to change the definition of a stored procedure?

( ) CHANGE

( ) CHANGE DEFINITION

( ) ALTER

( ) ALTER DEFINITION

Answer:

( ) CHANGE

( ) CHANGE DEFINITION

(√) ALTER

( ) ALTER DEFINITION

Demonstration: Authorizing Users to Execute Code

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In SQL Server Management Studio, on the **File** menu, point to **Open**, and then click **File**.
3. In the **Open File** dialog box, go to **D:\Demofiles\Mod03**, click **AuthorizingUsersToExecuteCode.sql**, and then click **Open**.
4. Execute the code under the heading for **Step 1** to change database context.
5. Execute the code under the heading for **Step 2** to change execution context.
6. Execute the code under the heading for **Step 3** to try to execute the **uspGetManagerEmployees** stored procedure. Note that permission is denied.
7. Execute the code under the heading for **Step 4** to revert the execution context.
8. Execute the code under the heading for **Step 5** to grant EXECUTE permissions for the stored procedure.
9. Execute the code under the heading for **Step 6** to change execution context.
10. Execute the code under the heading for **Step 7** to try to execute the **uspGetManagerEmployees** stored procedure again. Note that this time the code executes.
11. Execute the code under the heading for **Step 8** to try to execute the **ufnGetStock** function. Note that permission is denied.
12. Execute the code under the heading for **Step 9** to revert the execution context.
13. Execute the code under the heading for **Step 10** to grant EXECUTE permissions on the function.
14. Execute the code under the heading for **Step 11** to change the execution context and test the new permission. Note that the function now works as expected.
15. On the **File** menu, click **Close**.
16. Leave SQL Server Management Studio open for the next demonstration.

Lesson 3

Configuring Permissions at the Schema Level

Contents:

Question and Answers

Question: True or false? If a user does not have a default schema, SQL Server assumes the **guest** schema.

( ) True

( ) False

Answer:

( ) True

(√) False

Demonstration: Configuring Permissions at the Schema Level

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In SQL Server Management Studio, on the **File** menu, point to **Open**, and then click **File**.
3. In the **Open File** dialog box, go to **D:\Demofiles\Mod03**, click **ConfiguringPermissionsAtSchemaLevel.sql**, and then click **Open**.
4. Execute the code under the heading for **Step 1** to change database context.
5. Execute the code under the heading for **Step 2** to revoke permission on the **uspGetManagerEmployees** stored procedure.
6. Execute the code under the heading for **Step 3** to confirm that permission was revoked.
7. Execute the code under the heading for **Step 4** to grant EXECUTE permissions on the **dbo** schema.
8. Execute the code under the heading for **Step 5** to try to confirm that permission is now granted on the schema and the stored procedure in it.
9. Execute the code under the heading for **Step 6** to deny permission to execute the stored procedure.
10. Execute the code under the heading for **Step 7** to confirm that permission is denied.
11. Execute the code under the heading for **Step 8** to change database context.
12. Execute the code under the heading for **Step 9** to create a new function.
13. Execute the code under the heading for **Step 10** to explore which permissions imply the ability to select from a schema.
14. Execute the code under the heading for **Step 11** to explore which permissions imply the ability to view the definition of an object.
15. Execute the code under the heading for **Step 12** to explore which permissions imply the ability to select from an object.
16. Execute the code under the heading for **Step 13** to drop the user and the login.
17. On the **File** menu, click **Close**.
18. Close SQL Server Management Studio.

Module Review and Takeaways

Best Practice

Assigning permissions at schema level can simplify your security architecture.

Review Question(s)

Question: How does SQL Server differ from ANSI SQL regarding permissions?

Answer: ANSI SQL does not include a DENY statement.

Lab Review Questions and Answers

Lab: Authorizing Users to Access Resources

Question and Answers

Lab Review

Question: Your organization needs to track data access by individual Windows users. Does this mean you cannot base logins on Windows groups?

Answer: No.

Module 4

Protecting Data with Encryption and Auditing

Contents:

Lesson 1

Options for Auditing Data Access in SQL Server

Contents:

Question and Answers

Categorize each audit method into the appropriate category. Indicate your answer by writing the category number to the right of each item.

Items	
1	Triggers
2	SQL Trace
3	Temporal Tables
4	SQL Server Profiler

Category 1		Category 2
Records Changes to Data or Database Objects		Records DML Statements or DDL Statements

Answer:

Category 1		Category 2
Records Changes to Data or Database Objects		Records DML Statements or DDL Statements
Triggers Temporal Tables		SQL Trace SQL Server Profiler

Demonstration: Auditing with Temporal Tables

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Start SQL Server Management Studio and connect to your Azure instance running the **AdventureWorksLT** database, using SQL Server authentication.
3. Open the **Demo.ssmssln** solution in the D:\Demofiles\Mod04\Demo folder. If the Solution Explorer pane is not visible, on the **View** menu, click **Solution Explorer**.
4. Open the **Demo 01 - temporal table audit.sql** query. Follow the instructions contained within the comments of the script file.
5. Execute the code under the heading for **Step 2** to create a system-versioned temporary table.
6. Execute the code under the heading for **Step 3** to insert some example data.
7. Execute the code under the heading for **Step 4** to update a row.
8. Execute the code under the heading for **Step 5** to examine the current and history tables that make up the temporal table.
9. Execute the code under the heading for **Step 6** to demonstrate the behavior of the FOR SYSTEM TIME ALL sub-clause.
10. Execute the code under the heading for **Step 7** to demonstrate the behavior of the FOR SYSTEM TIME AS OF sub-clause.
11. Execute the code under the heading for **Step 8** to demonstrate that the history table cannot be edited. Both commands will generate an error.
12. Execute the code under the heading for **Step 9** to demonstrate that a user with permission to update the table directly can insert misleading information.
13. Execute the code under the heading for **Step 10** to examine the temporal table again after the update.
14. When you have finished the demonstration execute the code under the heading for **Step 11** to remove the demonstration objects.
15. Close SQL Server Management Studio, without saving any changes.

Lesson 2

Implementing SQL Server Audit

Contents:

Question and Answers

Question: Which of the following is **not** a component of the SQL Server Audit architecture?

( ) Target

( ) SQL Trace

( ) Server Audit

( ) Database Audit Specification

( ) Server Audit Specification

Answer:

( ) Target

(√) SQL Trace

( ) Server Audit

( ) Database Audit Specification

( ) Server Audit Specification

Resources

Introduction to Extended Events

Additional Reading: Since Extended Events is both the basis for SQL Server Audit and the replacement for SQL Trace, you could opt to write your own auditing system based on Extended Events. See Module 12 of this course *Tracing Access to SQL Server with Extended Events* for more information on working with Extended Events.

Demonstration: Using SQL Server Audit

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Run **Setup.cmd** in the D:\Demofiles\Mod04 folder as Administrator.
3. In the **User Account Control** dialog box, click **Yes**, and then wait for the script to complete.
4. Start SQL Server Management Studio and connect to **MIA-SQL** using Windows authentication.
5. Open the **Demo.ssmssln** solution in the D:\Demofiles\Mod04\Demo folder. If the Solution Explorer pane is not visible, on the **View** menu, click **Solution Explorer**.
6. Open the **Demo 02 - audit.sql**. Follow the instructions contained within the comments of the script file.
7. Execute the code under the heading for **Step 1** to create a new audit.
8. Execute the code under the heading for **Step 2** to enable the new audit.
9. Execute the code under the heading for **Step 3** to add a server audit specification to the new audit.
10. Execute the code under the heading for **Step 4** to add a database audit specification to the new audit.

11. Execute the code under the heading for **Step 5** to alter the database audit specification by adding an additional action group.
12. Execute the code under the heading for **Step 6** to examine the audit metadata.
13. Execute the code under the heading for **Step 7** to examine the server audit specification metadata.
14. Execute the code under the heading for **Step 8** to examine the database audit specification metadata.
15. Execute the code under the heading for **Step 9** to remove the audit and specifications created for this demonstration.
16. Leave SQL Server Management Studio open for the next demonstration.

Demonstration: Using Custom Audit Events

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. If it is not already started, start SQL Server Management Studio and connect to **MIA-SQL** using Windows authentication.
3. If it is not already open, open the **Demo.ssmssln** solution in the D:\Demofiles\Mod04\Demo folder. If the Solution Explorer pane is not visible, on the **View** menu, click **Solution Explorer**.
4. Open the **Demo 03 - custom audit.sql**. Follow the instructions contained within the comments of the script file.
5. Execute the code under the heading for **Step 1** to create a new audit.
6. Execute the code under the heading for **Step 2** to create a server audit specification including the USER_DEFINED_AUDIT_GROUP action group.
7. Execute the code under the heading for **Step 3** to call **sp_audit_write** directly.
8. Execute the code under the heading for **Step 4** to demonstrate how the custom event appears in the audit log file.
9. Execute the code under the heading for **Step 5** to create a stored procedure which uses **sp_audit_write**. The stored procedure will log a custom audit event if the discount applied to a row in the **Sales.OrderDetails** table is greater than 30 percent.
10. Execute the code under the heading for **Step 6** to call the new stored procedure twice. The second call should cause a custom audit event to be logged because the discount applied is 45 percent.
11. Execute the code under the heading for **Step 7** to examine how the custom event was recorded in the audit log file.
12. Execute the code under the heading for **Step 8** to drop the demonstration audit objects.
13. Leave SQL Server Management Studio open for the next demonstration.

Lesson 3

Managing SQL Server Audit

Contents:

Question and Answers

Question: Which of the following is *not* a target for SQL Server Audit?

( ) File

( ) Windows Application Log

( ) Windows Security Log

( ) Ring Buffer

Answer:

( ) File

( ) Windows Application Log

( ) Windows Security Log

(√) Ring Buffer

Demonstration: Viewing the Output of SQL Server Audit

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. If it is not already started, start SQL Server Management Studio and connect to **MIA-SQL** using Windows authentication.
3. If it is not already open, open the **Demo.ssmssln** solution in the D:\Demofiles\Mod04\Demo folder. If the Solution Explorer pane is not visible, on the **View** menu, click **Solution Explorer**.
4. Open the **Demo 04 - audit output.sql**.
5. Execute the code under the heading for **Step 1** to create an audit with a file target.
6. Execute the code under the heading for **Step 2** to create an audit with a Windows application log target.
7. Execute the code under the heading for **Step 3** to add a specification to each audit which collects SELECT statements run against the **salesapp1.Sales** schema.
8. Execute the code under the heading for **Step 4** to execute a select statement which will be audited.
9. Execute the code under the heading for **Step 5** to examine the contents of the audit file target. Point out the most useful fields.
10. Right-click the **Start** button, and then click **Event Viewer**.
11. In Event Viewer, expand the **Windows Logs** node, then click **Application**. The audit entry will be the most recent one in the Application pane with a Source value of MSSQLSERVER. Demonstrate the entry, then close Event Viewer.
12. Execute the code under the heading for **Step 7** to remove the demonstration audit objects.
13. Leave SQL Server Management Studio open for the next demonstration.

Lesson 4

Protecting Data with Encryption

Contents:

Question and Answers

Categorize each item by the corresponding SQL Server feature. Indicate your answer by writing the category number to the right of each item.

Items	
1	Data encrypted at rest
2	Data encrypted at rest and in transit
3	Data values obfuscated
4	Encryption and decryption carried out by SQL Server
5	Encryption and decryption take place at client application
6	Data stored in plain text
7	Acts at database level
8	Acts at column level

Category 1		Category 2		Category 3
TDE		Always Encrypted		Dynamic Data Masking

Answer:

Category 1		Category 2		Category 3
TDE		Always Encrypted		Dynamic Data Masking
Data encrypted at rest Encryption and decryption carried out by SQL Server Acts at database level		Data encrypted at rest and in transit Encryption and decryption take place at client application Acts at column level		Data values obfuscated Data stored in plain text

Resources

Transparent Data Encryption

Additional Reading: TDE is available in Azure SQL Database; for more information, see the topic *Encryption with Azure SQL Database* later in this lesson.

Demonstration: Using Dynamic Data Masking

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. If it is not already started, start SQL Server Management Studio and connect to **MIA-SQL** using Windows authentication.
3. If it is not already open, open the **Demo.ssmssln** solution in the D:\Demofiles\Mod04\Demo folder. If the Solution Explorer pane is not visible, on the **View** menu, click **Solution Explorer**.
4. Open the **Demo 05 - masking.sql**. Follow the instructions contained within the comments of the script file.
5. Execute the code under the heading for **Step 1** to create a new table with data masked data, grant permission to a test user, and insert test data.
6. Execute the code under the heading for **Step 2** to demonstrate that an administrator can see unmasked data.
7. Execute the code under the heading for **Step 3** to demonstrate that a user with only SELECT permission sees the masked data. Spend some time comparing the masked output to the table definitions.
8. Execute the code under the heading for **Step 4** to add a mask to the **home_phone_number** column.
9. Execute the code under the heading for **Step 5** to demonstrate the new mask.
10. Execute the code under the heading for **Step 6** to remove the mask from the **salary** column.
11. Execute the code under the heading for **Step 7** to demonstrate that the mask on salary is no longer in place.
12. Execute the code under the heading for **Step 8** to grant the UNMASK permission to the test user. Note that it is a database-level permission.
13. Execute the code under the heading for **Step 9** to demonstrate that the effect of the UNMASK permission.
14. Execute the code under the heading for **Step 10** to drop the demonstration table.
15. Close SQL Server Management Studio, without saving any changes.

Module Review and Takeaways

Best Practice

When planning to implement auditing, consider the following best practices:

- Choose the option to shut down SQL Server on audit failure. There is usually no point in setting up auditing, and then having situations where events can occur but are not audited. This is particularly important in high-security environments.
- Make sure that file audits are placed on drives with large amounts of free disk space and ensure that the available disk space is monitored on a regular basis.

Best Practice

When planning to implement database encryption, consider the following best practices:

- Use a complex password to protect the database master key for the master database.
- Ensure you back up certificates and private keys used to implement TDE, and store the backup files in a secure location.
- If you need to implement data encryption on multiple servers in a large organization, consider using an EKM solution to manage encryption keys.
- If you are planning to use Always Encrypted, plan how you will store column master keys to make them accessible to applications that need to encrypt and decrypt data.

Review Question(s)

Question: You may wish to audit actions by a DBA. How would you know if the DBA stopped the audit while performing covert actions?

Answer: Changes to the audit status are logged.

Lab Review Questions and Answers

Lab: Using Auditing and Encryption

Question and Answers

Lab Review

Question: Which type of Always Encrypted encryption will consistently encrypt the same plain text value to the same cypher text (assuming the same encryption key is used)?

( ) Deterministic encryption

( ) Randomized encryption

( ) Neither of the above

Answer:

(√) Deterministic encryption

( ) Randomized encryption

( ) Neither of the above

Module 5

Recovery Models and Backup Strategies

Contents:

Lesson 1

Understanding Backup Strategies

Contents:

Question and Answers

Question: What are the advantages of using SQL Server Backup with Azure Blob storage?

Answer:

- Limitless storage capacity.
- No need to purchase or manage hardware.
- Offsite backup without the need for transportation of tapes.

Demonstration: Back up an On-premise Database to Microsoft Azure

Demonstration Steps

Create a Storage Account

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Run **Setup.cmd** in the D:\Demofiles\Mod05 folder as Administrator.
3. On the taskbar, click **Internet Explorer**, and go to **portal.azure.com**.
4. Log into your Azure account, in the left blade click **Browse**, and then click **Storage accounts (classic)**.
5. On the **Storage accounts (classic)** blade, click **Add**.
6. On the **Storage account** blade, enter the following details:
    a. **Name**: 20764+yourinitials+ TodaysDate, for example 20764jhd20160421
    b. **Subscription**: Azure pass
    c. **Resource Group**: 20764A
    d. **Location**: Your nearest location
7. Click **Create**. The storage account will be created. Minimize Internet Explorer.

Create a Storage Container and SAS Token

1. On the taskbar, right-click the **Windows PowerShell** icon, and then click **Windows PowerShell ISE**.
2. On the **File** menu, click **Open**.
3. In the **Open** dialog box, go to D:\Demofiles\Mod05, click **ContainerSAS.ps1**, and then click **Open**.
4. Amend the **$accountName** to the Microsoft account that is associated with your Azure pass.
5. Amend the **$storageAccountName** to the name of the Storage Account you created in the previous task.
6. On the toolbar, click **Run Script**, and then in the message box, click **OK**.
7. In the **Sign in to your account** dialog box, enter your Microsoft Azure account user name and password, and then click **Sign in**.
8. In the **Confirm** dialog box, click **Yes** to delete the account (don't worry—this doesn't actually delete the account).

Note: Leave the window open—you will need the information displayed in the next task.

Back up a Database with Azure Blob Storage

1. Start SQL Server Management Studio, and connect to the **MIA-SQL** instance.
2. On the **File** menu, point to **Open**, and then click **File**.
3. In the **Open File** dialog box, go to **D:\Demofiles\Mod05**, click **AzureBackup.sql**, and then click **Open**.
4. Change the two instances of **https://xxxx.blob.core.windows.net/aw2016** entries to the name of your Cloud Blob Container URL in the PowerShell window.
5. Amend the **sv=2015-04-05&sr=c&si=policy1&sig=sfRAah2c1LjNZAfH1YiJQH%2FPA1sBTjOdPvk8z9849al%3D** entry to your Shared Access Signature in the PowerShell window.
6. In SQL Server Management Studio, highlight the statement under the comment **Create credential**, and then click **Execute**.
7. Highlight the statements underneath the comment **Backup the database**, click **Execute**, and wait for the backup process to complete successfully.
8. In Internet Explorer, on the **Storage accounts (classic)** blade, click **Refresh**, and then click the name of your storage account.
9. On your account blade, under **Services**, click **Blobs**, and then on the **Blob service** blade, click **aw2016**.
10. Verify that the **logtest.bak** backup file has been created.
11. Close Internet Explorer and close Windows Powershell.
12. Leave SSMS open for the next demonstration.

Lesson 2

SQL Server Transaction Logs

Contents:

Question and Answers

Question: What are the unique features of transaction log restores?

> **Answer:** Point-in-time recovery and the ability to restore up to the point of failure if only data files are corrupt.

Demonstration: Logs and Full Recovery

Demonstration Steps

1. Ensure that the 20764A-MIA-DC and 20764A-MIA-SQL virtual machines are running, and log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In SQL Server Management Studio, in Object Explorer, expand **Databases**, right-click **LogTest**, and then click **Properties**.
3. In the **Database Properties - LogTest** dialog box, on the Options page, verify that the **Recovery model** is set to **Full**, and then click **Cancel**.
4. On the **File** menu, point to **Open**, and then click **File**.
5. In the **Open File** dialog box, go to **D:\Demofiles\Mod05\Demo**, click **LogComparisonTest.sql**, and then click **Open**.
6. Select the code under the comment **Perform a full database backup**, and then click **Execute**.
7. Select the code under the comment **View log file space**, and then click **Execute**. Note the log size and space used in the **LogTest** log.
8. Select the code under the comment **Insert data**, and then click **Execute** to insert 10000 rows.
9. Select the code under the comment **View log file space**, and then click **Execute**. Note the log size and space used in the **LogTest** log file has increased.
10. Select the code under the comment **Issue checkpoint**, and then click **Execute** to force SQL Server to perform a checkpoint and flush the modified pages to disk.
11. Select the code under the comment **View log file space**, and then click **Execute**. Note the space used in the **LogTest** log file has not decreased.
12. Select the code under the comment **Check log status**, and then click **Execute**. Note that SQL Server is awaiting a log backup before the log file can be truncated.
13. Select the code under the comment **Perform a log backup**, and then click **Execute**.
14. Select the code under the comment **Verify log file truncation**, and then click **Execute**. Note the space used in the **LogTest** log file has decreased because the log has been truncated.
15. Close SSMS without saving any changes.

Lesson 3

Planning Backup Strategies

Contents:

Question and Answers

Question: What kind of database might be a good candidate for a full backup strategy?

Answer: A small database that can be backed up quickly.

Module Review and Takeaways

Best Practice

Plan your backup strategy carefully.

Plan the backup strategy in conjunction with the business needs.

Choose the appropriate database recovery model.

Plan your transaction log size, based on the transaction log backup frequency.

Consider using differential, partial, and filegroup backups to speed recovery.

Review Question(s)

Question: When might a full database backup strategy be adequate?

> **Answer:** If it is sufficient, in case of a disaster, to restore the database to the points of full database backup only.

Module 6

Backing Up SQL Server Databases

Contents:

Lesson 1

Backing Up Databases and Transaction Logs

Contents:

Demonstration: Performing Backups

Demonstration Steps

Perform a Full Database Backup

1. Ensure that you have started the 20764A-MIA-DC and 20764A-MIA-SQL virtual machines, log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**, and in the D:\Demofiles\Mod06 folder, run **Setup.cmd** as Administrator. Wait for the script to finish and then press enter.
2. Start SQL Server Management Studio, and connect to the **MIA-SQL** database engine using Windows authentication.
3. In Object Explorer, under **Databases**, right-click **AdventureWorks**, point to **Tasks**, and click **Back Up**.
4. In the **Backup Up Database – AdventureWorks** dialog box, ensure that **Backup type** is set to **Full**, and in the **Destination** section, select each existing file path and click **Remove**. Then click **Add** and in the **Select Backup Destination** dialog box, enter the file name **D:\Demofiles\Mod06\Demo\AW.bak** and click **OK**.
5. In the **Back Up Database – AdventureWorks** dialog box, on the Media Options page, note that the default option is to append to an existing media set. In this case, there is no existing media set so a new one will be created, and there are no existing backup sets to overwrite.
6. In the **Backup Up Database – AdventureWorks** dialog box, on the Backup Options page, note the default backup name and expiration settings.
7. In the **Back Up Database – AdventureWorks** dialog box, in the **Script** drop-down list, click **Script Action to a New Query Window**. Then click **OK**.
8. When the backup has completed successfully, click **OK**.
9. In the query pane, view the Transact-SQL BACKUP statement that was used to back up the database.
10. View the D:\Demofiles\Mod06\Demo folder and note the size of the **AW.bak** file.

Perform a Differential Backup

1. In SQL Server Management Studio, open the **UpdatePrices.sql** script file from the D:\Demofiles\Mod06\Demo folder and click **Execute**. This script updates the **Production.Product** table in the **AdventureWorks** database.
2. In Object Explorer, under **Databases**, right-click **AdventureWorks**, point to **Tasks**, and click **Back Up**.
3. In the **Backup Up Database – AdventureWorks** dialog box, in the **Backup type** list, select **Differential**. Then in the **Destination** section, ensure that **D:\Demofiles\Mod06\Demo\AW.bak** is the only backup device listed.
4. In the **Backup Up Database – AdventureWorks** dialog box, on the Media Options page, verify that the option to append to the existing media set is selected.
5. In the **Backup Up Database – AdventureWorks** dialog box, on the Backup Options page, change the Name to **AdventureWorks-Diff Database Backup**.
6. In the **Backup Up Database – AdventureWorks** dialog box, in the **Script** drop-down list, click **Script Action to a New Query Window**. Then click **OK**.
7. When the backup has completed successfully, click **OK**.

8. In the query pane, view the Transact-SQL BACKUP statement that was used to back up the database. Note that it includes the WITH DIFFERENTIAL option.
9. View the D:\Demofiles\Mod06\Demo folder and note that size of the **AW.bak** file has increased, but not much—the second backup only includes the extents containing pages that were modified since the full backup.

Perform a Transaction Log Backup

1. In SQL Server Management Studio, switch to the **UpdatePrices.sql** script you opened previously and click **Execute** to update the **Production.Product** table in the **AdventureWorks** database again.
2. In Object Explorer, under **Databases**, right-click **AdventureWorks,** point to **Tasks**, and click **Back Up**.
3. In the **Back Up Database – AdventureWorks** dialog box, in the **Backup type** list, select **Transaction Log.** Then in the **Destination** section, ensure that **D:\Demofiles\Mod06\Demo\AW.bak** is the only backup device listed.
4. In the **Backup Up Database – AdventureWorks** dialog box, on the Media Options page, verify that the option to append to the existing media set is selected. Also verify that the option to truncate the transaction log is selected.
5. In the **Backup Up Database – AdventureWorks** dialog box, on the Backup Options page, change the **Name** to **AdventureWorks-Transaction Log Backup**.
6. In the **Backup Up Database – AdventureWorks** dialog box, in the **Script** drop-down list, click **Script Action to a New Query Window**. Then click **OK**.
7. When the backup has completed successfully, click **OK**.
8. In the query pane, view the Transact-SQL BACKUP statement that was used to back up the database. Note that this time the statement is BACKUP LOG.
9. View the D:\Demofiles\Mod06 folder and note that size of the **AW.bak** file has increased, but not much—the third backup only includes the transaction log entries for data modifications since the full backup.
10. Keep SQL Server Management Studio open for the next demonstration.

Lesson 2

Managing Database Backups

Contents:

Demonstration: Verifying Backups

Demonstration Steps

View the Backup and Restore Events Report

1. Ensure that you have performed the previous demonstration in this module.
2. In SQL Server Management Studio, in Object Explorer, under **Databases**, right-click **AdventureWorks**, point to **Reports**, point to **Standard Reports**, and click **Backup and Restore Events**.
3. In the **Backup and Restore Events [AdventureWorks]** report, expand **Successful Backup Operations** and view the backup operations that have been performed for this database.
4. In the **Device Type** column, expand each of the **Disk (temporary)** entries to view details of the backup media set files.

Query Backup History Tables

1. In SQL Server Management Studio, open the **VerifyingBackups.sql** script file in the D:\Demofiles\Mod06\Demo folder.
2. Highlight the code under the comment **View backup history**, and click **Execute**.
3. View the query results, which show the backups that have been performed for the **AdventureWorks** database.
4. Note line 21 restricts retrieval period currently set to 30 days. This can be modified by changing the number 30 or by removing this line.

Verify Backup Media

1. Highlight the code under the comment **Use RESTORE HEADERONLY**, and click **Execute**.
2. View the query results, which show the backups in the **AW.bak** backup device.
3. Highlight the code under the comment **Use RESTORE FILELISTONLY**, and click **Execute**.
4. View the query results, which show the database files contained in the backups.
5. Highlight the code under the comment **Use RESTORE VERIFYONLY**, and click **Execute**.
6. View the message that is returned, which should indicate that the backup is valid.

Lesson 3

Advanced Database Options

Contents:

Demonstration: Using Backup Compression

Demonstration Steps

Use Backup Compression

1. Ensure that you have performed the previous demonstration in this module.
2. In SQL Server Management Studio, in Object Explorer, under **Databases**, right-click **AdventureWorks**, point to **Tasks**, and click **Back Up**.
3. In the **Backup Up Database – AdventureWorks** dialog box, ensure that **Backup type** is set to **Full**, and in the **Destination** section, select the existing file path and click **Remove**. Then click **Add** and in the **Select Backup Destination** dialog box, enter the file name **D:\Demofiles\Mod06\AW_Comp.bak** and click **OK**.
4. In the **Backup Up Database – AdventureWorks** dialog box, on the Media Options page, note that the default option is to append to an existing media set. In this case, there is no existing media set so a new one will be created, and there are no existing backup sets to overwrite.
5. In the **Backup Up Database – AdventureWorks** dialog box, on the Backup Options page, change the **Name** to **AdventureWorks-Compressed Backup** and in the **Set backup compression** list, select **Compress backup**.
6. In the **Backup Up Database – AdventureWorks** dialog box, in the **Script** drop-down list, click **Script Action to a New Query Window**. Then click **OK**.
7. When the backup has completed successfully, click **OK**.
8. In the query pane, view the Transact-SQL BACKUP statement that was used to back up the database, noting that the COMPRESSION option was specified.
9. View the D:\Demofiles\Mod06 folder and note the size of the **AW_Comp.bak** file. This should be significantly smaller than the **AW.bak** file was after the full database backup in the previous demonstration.
10. Keep SQL Server Management Studio open for the next demonstration.

Demonstration: Using Backup Encryption

Demonstration Steps

Create a Database Master Key

1. Ensure that you have performed the previous demonstration in this module.
2. In SQL Server Management Studio, open the **EncyptionKeys.sql** script file in the D:\Demofiles\Mod06\Setupfiles folder.
3. Start File Explorer and create the D:\Backups folder.
4. Select the code under the comment **Create a database master key** and click **Execute**.
5. Select the code under the comment **Back up the database master key** and click **Execute**.

Create a Certificate

1. Select the code under the comment Create a certificate and click **Execute**.
2. Select the code under the comment **Back up the certificate and its private key** and click **Execute**.

Encrypt a Database Backup

1. In Object Explorer, under **Databases**, right-click **AdventureWorks**, point to **Tasks**, and click **Back Up**.
2. In the **Backup Up Database – AdventureWorks** dialog box, ensure that **Backup type** is set to **Full**, and in the **Destination** section, select the existing file path and click **Remove**. Then click **Add** and in the **Select Backup Destination** dialog box, enter the file name **D:\Backups\AW_Encrypt.bak** and click **OK**.
3. In the **Backup Up Database – AdventureWorks** dialog box, on the Media Options page, select **Back up to a new media set, and erase all existing backup sets**. Then enter the new media set name **Encrypted Backup**.
4. In the **Backup Up Database – AdventureWorks** dialog box, on the Backup Options page, change the **Name** to **AdventureWorks-Encrypted Backup** and in the **Set backup compression** list, select **Compress backup**.
5. In the **Encryption** section, select **Encrypt backup**. Then ensure that the **AES 256** algorithm is selected, and select the **AdventureWorks** certificate you created previously.
6. In the **Backup Up Database – AdventureWorks** dialog box, in the **Script** drop-down list, click **Script Action to a New Query Window**. Then click **OK**.
7. When the backup has completed successfully, click **OK**.
8. In the query pane, view the Transact-SQL BACKUP statement that was used to back up the database, noting that the ENCRYPTION option was specified.

Module Review and Takeaways

Best Practice

- Plan your backup strategy carefully.
- Plan the backup strategy in conjunction with the business needs.
- Choose the appropriate database recovery model.
- Plan your transaction log size based on the transaction log backup frequency.
- Consider using differential backups to speed recovery.
- Consider compressing backups to reduce storage requirements and backup time.

Review Question(s)

Question: What are the unique features of transaction log restores?

Answer: Point-in-time recovery and the ability to restore up to the point of failure if only data files are corrupt.

Question: When might a full database backup strategy be adequate?

Answer: If it is sufficient, in case of a disaster, to restore the database to the points of full database backup only.

Module 7

Restoring SQL Server 2016 Databases

Contents:

Lesson 2
Restoring Databases

Contents:

Demonstration: Restoring Databases

Demonstration Steps

Create a tail-log backup

1. Ensure that the 20764A-MIA-DC and 20764A-MIA-SQL virtual machines are running, and log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In the D:\Demofiles\Mod07 folder, run **Setup.cmd** as Administrator. In the **User Account Control** dialog box click **Yes**, and then wait until the script finishes.
3. Start SQL Server Management Studio and connect to the **MIA-SQL** database engine using Windows authentication.
4. Click **New Query** and execute the following Transact-SQL code to perform a tail-log backup:

   ```
   BACKUP LOG AdventureWorks TO DISK = 'D:\Demofiles\Mod07\BackupDemo.bak'

   WITH NO_TRUNCATE;
   ```

5. Click **Execute**, and view the resulting message to verify that the backup is successful.

Restore a Database

1. In Object Explorer, expand **Databases**, right-click **AdventureWorks**, point to **Tasks**, point to **Restore**, and click **Database**.
2. In the **Restore Database - AdventureWorks** dialog box, note that the restore operation will restore both the full backup and the transaction log that you recently backed up, and then click **OK**.
3. In the **Microsoft SQL Server Management Studio** dialog box, note that the restore operation was successful, and then click **OK**.
4. In Object Explorer, verify that the **AdventureWorks** database is ready to use.
5. Close SQL Server Management Studio without saving any files.

Lesson 3
Advanced Restore Scenarios

Contents:

Demonstration: Restoring an Encrypted Backup

Demonstration Steps

Restore an Encrypted Backup

1. Ensure that the 20764A-MIA-DC and 20764A-MIA-SQL virtual machines are running, and log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. If you did not complete the previous demonstration, in the D:\Demofiles\Mod07 folder, run **Setup.cmd** as Administrator.
3. Start SQL Server Management Studio and connect to the **MIA-SQL\SQL2** database engine using Windows authentication.
4. In Object Explorer, under the **MIA-SQL\SQL2** instance, expand **Databases** and view the existing databases on this instance.
5. Open the **Restore Encrypted Backup.sql** script file in the D:\Demofiles\Mod07 folder.
6. Select the code under the comment **Try to restore an encrypted backup** and click **Execute**. Note that this fails because the required certificate is not present.
7. Select the code under the comment **Create a database master key for master** and click **Execute**. This creates a database master key for the master database on **MIA-SQL\SQL2**.
8. Select the code under the comment **Import the backed up certificate** and click **Execute**. This creates a certificate from public and private key backups that were taken from the **MIA-SQL** instance.
9. Select the code under the comment **Restore the encrypted database** and click **Execute**. Note that this time the restore operation succeeds.
10. In Object Explorer, refresh the **Databases** folder and verify that the **AdventureWorks** database has been restored.
11. Close SQL Server Management Studio.

Lesson 4
Point-in-Time Recovery

Contents:

Demonstration: Performing a Point-in-Time Recovery

Demonstration Steps

Perform a Point-In-Time Recovery

1. Ensure that the 20764A-MIA-DC and 20764A-MIA-SQL virtual machines are running, and log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Start SQL Server Management Studio and connect to the **MIA-SQL** database engine using Windows authentication.
3. In SQL Server Management Studio, open the **Point-in-Time Restore.sql** script file in the D:\Demofiles\Mod07 folder.
4. Select and execute the code under the comment **Create a database and back it up**. This creates a database with a single table, and performs a full backup.
5. Select and execute the code under the comment **Enter some data**. This inserts a record into the **Customers** table.
6. Select and execute the code under the comment **Get the current time**. This displays the current date and time. Make a note of the current time.
7. Wait until a minute has passed, and then select and execute the code under the comment **Get the current time** again to verify that it is now at least a minute since you noted the time.
8. Select and execute the code under the comment **Enter some more data**. This inserts a second record into the **Customers** table.
9. Select and execute the code under the comment **Backup the transaction log**. This performs a transaction log backup of the database.
10. Close the query window.
11. In Object Explorer, expand **Databases** and verify that **BackupDemo** is listed (if not, right-click the **Databases** folder and click **Refresh**). Then right-click the **BackupDemo** database, point to **Tasks**, point to **Restore**, and click **Database**.
12. In the **Restore Database - BackupDemo** dialog box, click **Timeline**.
13. In the **Backup Timeline: BackupDemo** dialog box, select **Specific date and time** and set the **Time** value to the time you noted earlier (after the first row of data was inserted). Then click **OK**.
14. In the **Restore Database - BackupDemo** dialog box, click **OK**. When notified that the database has been restored successfully, click **OK**.
15. In Object Explorer, expand the **BackupDemo** database and its **Tables** folder. Then right-click **dbo.Customers** and click **Select Top 1000 Rows**. When the results are displayed, verify that the database was restored to the point in time after the first row of data was inserted, but before the second row was inserted.
16. Close SQL Server Management Studio without saving any files.

Module Review and Takeaways

Best Practice

- Don’t forget to back up the tail of the log before starting a restore sequence.
- If available, use differential restore to reduce the time taken by the restore process.
- Use file level restore to speed up restores when not all database files are corrupt.
- Perform regular database backups of **master**, **msdb** and **model** system databases.
- Create a disaster recovery plan for your SQL Server and make sure you regularly test restoring databases.

Review Question(s)

Question: What are the three phases of the restore process?

Answer: The three phases of a restore include data copy, redo and undo. The data copy phase involves copying all data, log and index pages from the backup media. The redo phase applies the transactions to the data copied from the backup to be rolled forward to the recovery point. The undo phase rolls back any uncommitted transactions and makes the database available to users. After the rollback phase, subsequent backups cannot be restored.

Module 8

Automating SQL Server Management

Contents:

Lesson 1

Automating SQL Server Management

Contents:

Question and Answers

What are the four core objects types provided by SQL Server Agent?

Items	
1	Jobs
2	Maintenance Plans
3	Schedules
4	Backup Tasks
5	Alerts
6	Logs
7	Operators
8	SCOM Reporting

Category 1		Category 2
Provided by SQL Server Agent		Not provided by SQL Server Agent

Answer:

Category 1		Category 2
Provided by SQL Server Agent		Not provided by SQL Server Agent
Jobs Schedules Alerts Operators		Maintenance Plans Backup Tasks Logs SCOM Reporting

Demonstration: SQL Server Agent

Demonstration Steps

Create a Job

1. Ensure that the 20764A-MIA-DC and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **AdventureWorks\Student** with the password **Pa$$w0rd**.
2. In Windows Explorer, navigate to **D:\Demofiles\Mod08**, right-click **Setup.cmd**, and then click **Run as administrator**.
3. In the **User Account Control** dialog box, click **Yes**.
4. On the taskbar, click **Microsoft SQL Server Management Studio**.
5. In the **Connect to Server** dialog box, in **Server name** box, type **MIA-SQL** and then click **Connect**.
6. In Object Explorer, expand **SQL Server Agent**, and then expand **Jobs**.
7. Note that there are existing jobs on the server.
8. Right-click the **Jobs** folder, click **New Job**.
9. In the **New Job** dialog box, in the **Name** box, type **Manual AdventureWorks Backup**, and in the **Category** list, select **Database Maintenance**.
10. On the Steps page, click **New**.
11. In the **New Job Step** dialog box, in the **Step name** box, type **Backup AdventureWorks**, and in the **Database** list, select **AdventureWorks**.
12. In the **Command** box type the following, and then click **OK**:

```
BACKUP DATABASE [AdventureWorks] TO  DISK =
N'D:\Demofiles\Mod08\Backups\AdventureWorksAgentBackup.bak' WITH NOFORMAT, INIT,
NAME = N'AdventureWorks-Full Database Backup', COMPRESSION,  STATS = 10
GO
```

13. In the **New Job** dialog box, click **OK**.

14. In Object Explorer, under **Jobs**, note that the new job, **Manual AdventureWorks Backup** is displayed.

Run and Review the Output from a Job

1. In Object Explorer, under **Jobs**, right-click **Manual AdventureWorks Backup**, and then click **Start Job at Step**.
2. In the **Start Jobs – MIA-SQL** dialog box, note that the status of each step when it finishes changes to **Success**, and then click **Close**.
3. In Object Explorer, under **SQL Server Agent**, double-click **Job Activity Monitor**.
4. In the **Job Activity Monitor – MIA-SQL** window, review the information available for SQL Server Agent jobs.
5. In the **Agent Job Activity** table, review the information in the **Manual AdventureWorks Backup** row, and then click **Close**.
6. In Windows Explorer, navigate to **D:\Demofiles\Mod08\Backups**. Verify that the **AdventureWorksAgentBackup.bak** file has been created.
7. Leave SSMS open for the next demonstration.

Lesson 2

Working with SQL Server Agent

Contents:

Question and Answers

Put the following SQL Server Agent steps in the order required to create a job, by numbering each to indicate the correct order.

	Steps
	Create a job.
	Categorize the job.
	Create one or more steps.
	Create one or more schedules.
	Create or select an operator for notifications.
	Schedule the job for execution.

Answer:

	Steps
1	Create a job.
2	Categorize the job.
3	Create one or more steps.
4	Create one or more schedules.
5	Create or select an operator for notifications.
6	Schedule the job for execution.

Demonstration: Scripting Jobs

Demonstration Steps

Script a Task to a Job

1. In SSMS, in Object Explorer, expand **Databases**, right-click **AdventureWorks**, point to **Tasks**, and then click **Back Up**.
2. In the **Back Up Database - AdventureWorks** dialog box, in the **Destination** section, click the existing backup destination, and then click **Remove**.
3. Click **Add** and in the **Select Backup Destination** dialog box, in the **File name** box, type **D:\Demofiles\Mod08\Backups\AdventureWorksScript.bak**, and then click **OK**.
4. In the **Back Up Database - AdventureWorks** dialog box, on the toolbar, in the **Script** drop-down list, select **Script Action to Job**.
5. In the **New Job** dialog box, on the General page, note the default name for the job (Back Up Database - AdventureWorks). On the Steps page, note that the job includes one **Transact-SQL step** named **1**.
6. On the Schedules page, click **New**.
7. In the **New Job Schedule** dialog box, in the **Name** box, type **Week Days**.

8. In the **Frequency** section, select **Monday**, **Tuesday**, **Wednesday**, **Thursday**, and **Friday**, and then click **OK**.
9. In the **New Job** dialog box, click **OK**.
10. In the **Back Up Database - AdventureWorks** dialog box, click **Cancel** so that the job is saved, but not yet run.
11. Verify that the job appears in the **Jobs** folder in **Object Explorer**.

Generate Scripts for Existing Jobs

1. In Object Explorer, under **Jobs**, right-click **Check AdventureWorks DB**, point to **Script Job as**, point to **CREATE To**, and then click **New Query Editor Window**. This generates the Transact-SQL code necessary to create the job.
2. In Object Explorer, right-click **Back Up Database - AdventureWorks**, point to **Script Job as**, point to **CREATE To**, and then click **Clipboard**. Place the insertion point at the end of the Transact-SQL code in the query editor window, and then on the **Edit** menu, click **Paste**.
3. Save the Transact-SQL script as **Create Jobs.sql** in the D:\Demofiles\Mod08 folder.
4. This technique is useful to generate script creation jobs so they can be recreated if they are accidentally deleted or are required on a different server.
5. Keep SQL Server Management Studio open for the next demonstration.

Lesson 3

Managing SQL Server Agent Jobs

Contents:

Question and Answers

What are four steps that can be undertaken to troubleshoot failed jobs?

Items	
1	Check SQL Server Agent Status
2	Start and Stop SQL Server
3	Review Job History
4	Check Free Disk Space
5	Check Job Execution
6	Check Activity Monitor
7	Check Access to Dependencies

Category 1		Category 2
Troubleshooting step		Other database activity

Answer:

Category 1		Category 2
Troubleshooting step		Other database activity
Check SQL Server Agent Status Review Job History Check Job Execution Check Access to Dependencies		Start and Stop SQL Server Check Free Disk Space Check Activity Monitor

Demonstration: Viewing Job History and Resolving Failed Jobs

Demonstration Steps

Run Jobs

1. In SQL Server Management Studio, in Object Explorer, right-click **Back Up Database – AdventureWorks**, and then click **Start Job at Step**.
2. When the job has completed successfully, click **Close**.
3. In Object Explorer, right-click **Check AdventureWorks DB**, and then click **Start Job at Step**.
4. Note that this job does not start automatically because it has more than one job step.
5. In the **Start Job on 'MIA-SQL'** dialog box, in the **Start execution at step** table, click **1**, and then click **Start**. Note that the job fails, and then click **Close**.

Troubleshoot a Failed Job

1. In Object Explorer, right-click **Back Up Database – AdventureWorks**, and then click **View History**.
2. In the **Log File Viewer - MIA-SQL** dialog box, expand the date for the most recent instance of the job, note that all steps succeeded, and then click **Close**.
3. In Object Explorer, right-click **Check AdventureWorks DB**, and then click **View History**.
4. In the **Log File Viewer - MIA-SQL** dialog box, expand the date for the most recent failed instance of the job, and note that the step **3** failed.
5. Select the step that failed, and in the pane at the bottom of the dialog box, view the message that was returned. You may have to scroll to the bottom. Then click **Close**.
6. In Object Explorer, double-click **Check AdventureWorks DB**.
7. In the **Job Properties - Check AdventureWorks DB** dialog box, on the Steps page, in the **Job step list** table, select step **3**, and then click **Edit**.
8. In the **Job Step Properties - Check DB** dialog box, click **Parse**.
9. Note the same error message that was shown in the Job History dialog, and then click **OK**.
10. Modify the text in the **Command** box as follows, and then click **OK**:

    ```
    DBCC CHECKDB ('AdventureWorks');
    ```

11. In the **Job Properties - Check AdventureWorks DB** dialog box, click **OK**.
12. In Object Explorer, right-click **Check AdventureWorks DB**, and then click **Start Job at Step**.
13. In the **Start Job on 'MIA-SQL'** dialog box, in the **Start execution at step** table, click **1**, and then click **Start**.
14. When the steps complete successfully, click **Close**.
15. In **Object Explorer**, double-click **Job Activity Monitor**.
16. In the Job Activity Monitor – MIA-SQL dialog box, in the **Agent Job Activity** table, note the status of the **Check AdventureWorks DB** job.
17. In the D:\Demofiles\Mod08\AdventureWorks folder, view the text files generated by the job.
18. In the D:\Demofiles\Mod08\AdventureWorks folder, verify that a backup file was created by the **Back Up Database - AdventureWorks** job.
19. Keep SQL Server Management Studio open for the next demonstration.

Lesson 4

Multi-Server Management

Contents:

Question and Answers

Question: Which of the following statements is false?

( ) A master server can have multiple target servers.

( ) A high-load master server won't have any adverse performance impact by having a number of target servers.

( ) Each target server can only connect to a single master server.

( ) Changing the name of a target server requires it to be registered with the master server.

> **Answer:**
>
> ( ) A master server can have multiple target servers.
>
> (√) A high-load master server won't have any adverse performance impact by having a number of target servers.
>
> ( ) Each target server can only connect to a single master server.
>
> ( ) Changing the name of a target server requires it to be registered with the master server.

Demonstration: Configuring Master and Target Servers

Demonstration Steps

Use the Master Server Wizard

1. In SQL Server Management Studio, in Object Explorer, under **MIA-SQL**, right-click **SQL Server Agent**, point to **Multi Server Administration**, and then click **Make this a Master**.
2. In the **Master Server Wizard – MIA-SQL** dialog box, on the Welcome to the Master Server Wizard page, click **Next**.
3. On the Master Server Operator page, in the **E-mail address** box, type **student@adventureworks.com**, and then click **Next**.
4. On the Target Servers page, expand **Database Engine**, expand **Local Server Groups**.
5. Click **mia-sql\sql2**, click the **>**, and then click **Next**.
6. In the **Checking Server Compatibility** dialog box, click **Close**.
7. On the Master Server Login Credentials page, click **Next**.
8. On the Complete the Wizard page, click **Finish**.
9. When configuration is complete, click **Close**.

Use Transact-SQL to Register a Target Server

1. In Object Explorer, on the toolbar, click **Connect**, and then click **Database Engine**.
2. In the **Connect to Server** dialog box, in the **Server name** list, select **MIA-SQL\SQL3**, and then click **Connect**.
3. In Object Explorer, under **MIA-SQL\SQL3**, expand **Databases**, expand **System Databases**, right-click **msdb**, and then click **New Query**.
4. In the new query window, type the following command, and then click **Execute**:

```
EXEC dbo.sp_msx_enlist N'MIA-SQL';
```

5. In Object Explorer, under **MIA-SQL**, right-click **SQL Server Agent (MSX)**, point to **Multi Server Administration**, and then click **Manage Target Servers**.
6. In the **Target Server Status – MIA-SQL** dialog box, note that both **MIA-SQL\SQL2** and **MIA-SQL\SQL3** are listed as target servers.
7. In the **Target Server Status – MIA-SQL** dialog box, click **Close**.
8. Close SSMS without saving any changes.

Module Review and Takeaways

Best Practice

When using a large number of target servers, avoid defining your master server on a production server because the target server traffic may impact performance on the production server.

Module 9

Configuring Security for SQL Server Agent

Contents:

Lesson 1

Understanding SQL Server Agent Security

Contents:

Question and Answers

Put the following fixed roles in order from least privileged to most privileged by numbering each to indicate the correct order.

	Steps
	SQLAgentUserRole
	SQLAgentReaderRole
	SQLAgentOperatorRole
	sysadmin

Answer:

	Steps
1	SQLAgentUserRole
2	SQLAgentReaderRole
3	SQLAgentOperatorRole
4	sysadmin

Demonstration: Assigning a Security Context to Job Steps

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764-MIA-DC, and 20764-MIA-SQL virtual machines are running, and then log on to 20764-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In the D:\Demofiles\Mod09 folder, right-click **Setup.cmd**, and then click **Run as Administrator**.
3. If a User Account Control window appears, click **Yes**.
4. Start Microsoft SQL Server Management Studio from the taskbar, and then connect to the **MIA-SQL** Database Engine instance by using Windows Authentication.
5. In the file menu, click Open and then click Project/Solution. Open the **Demo.ssmssln** solution in **D:\Demofiles\Mod09\Demo** folder. If the Solution Explorer pane is not visible, on the **View** menu, click **Solution Explorer**.
6. Double Click the **Demo 1 - security context.sql** script file.
7. In Object Explorer, expand **SQL Server Agent**, expand **Jobs**, right-click **Record Execution Identity**, and then click **Start Job at Step**. In the **Start Jobs – MIA-SQL** window, make sure that the job ran successfully, and then click **Close**.

   The job triggers the **dbo.RecordIdentity** stored procedure in the **AdminDB** database. The procedure logs the identity of whoever ran the procedure to **dbo.IdentityLog**.
8. In Object Explorer, right-click **Record Execution Identity**, and then click **View History**. In the **Log File Viewer – MIA-SQL** window, expand the visible job execution by clicking the plus sign on the row in the right pane, and then scroll the window to the right so that the **Message** column is visible.

Notice that the first row shows that the job was invoked by **ADVENTUREWORKS\Student**, but the job step row shows that it was executed as **ADVENTUREWORKS\ServiceAcct**. When a **sysadmin** user owns a job, the job steps are executed in the context of the SQL Server Agent service account by default.

9. Close the **Log File Viewer – MIA-SQL** window.
10. In the Demo 1 - security context.sql query pane, execute the query under the comment that begins **Task 1** to view the contents of the **AdminDB.dbo.IdentityLog** table. Notice that the identity of the service account was recorded.
11. Change the job owner. In Object Explorer, right-click **Record Execution Identity**, and then click **Properties**. On the **General** page, clear the **Owner** box, type **ITSupportLogin**, and then click **OK**.
12. In Object Explorer, right-click **Record Execution Identity**, and then click **Start Job at Step**. In the **Start Jobs – MIA-SQL** window, notice that the job fails, and then click **Close**.
13. To view the job history to find the reason for the failure, right-click **Record Execution Identity**, and then click **View History**. In the **Log File Viewer – MIA-SQL** window, expand the failed job by clicking the plus sign next to the error symbol, scroll the window to the right and until the **Message** column is visible, and then note the reason for the failure. The failure reason should show as follows:

    ```
    Executed as user: ITSupportLogin. The EXECUTE permission was denied on the object
    'RecordIdentity', database 'AdminDB', schema 'dbo'. [SQLSTATE 42000] (Error 229).
    The step failed.
    ```

14. Close the **Log File Viewer – MIA-SQL** window.

    The job was run as the **ITSupportLogin** login, which maps to the **ITSupport** user in the **AdminDB** database; that user has no permissions to execute the stored procedure, so the job step failed.

15. In the Demo 1 - security context.sql query pane, execute the query under the comment that begins **Task 2** to grant the permission that is necessary to enable the job to run.
16. In Object Explorer, right-click **Record Execution Identity**, and then click **Start Job at Step**. In the **Start Jobs – MIA-SQL** window, notice that the job succeeds, and then click **Close**.
17. In the Demo 1 - security context.sql query pane, execute the query under the comment that begins **Task 1** to view the contents of the **AdminDB.dbo.IdentityLog** table.
18. Keep SQL Server Management Studio open for the next demonstration.

Lesson 2

Configuring Credentials

Contents:

Question and Answers

Question: What happens to a credential when the password of the Windows user that the credential references is changed?

( ) The credential is automatically deleted.

( ) The credential is disabled.

( ) Attempts to use the credential fail until the password is updated.

( ) The credential continues to operate normally.

Answer:

( ) The credential is automatically deleted.

( ) The credential is disabled.

(√) Attempts to use the credential fail until the password is updated.

( ) The credential continues to operate normally.

Demonstration: Configuring Credentials

Demonstration Steps

1. On 20764-MIA-SQL, in SQL Server Management Studio, in Object Explorer, under **MIA-SQL** and **SQL Server Agent**, right-click **Jobs**, and then click **New Job**.
2. In the **New Job** window, on the **General** page, type **Copy Export File** in the **Name** box. On the **Steps** page, click **New**.
3. In the **New Job Step** window, on the **General** page, type **Copy File** in the **Step Name** box. In the **Type** box, select **Operating System (CmdExec)**. In the **Command** window, type the following:

```
copy d:\demofiles\Mod09\ExportData\export.txt
d:\demofiles\Mod09\ImportData\import.txt /Y
```

   Notice that the job is configured to run in the security context of the SQL Server Agent service account.

4. On the **Advanced** page, in the **On success action** box, select **Quit the job reporting success**, and then click **OK**. Click **OK** in the **New Job** window to create the job.
5. In Object Explorer, under **Jobs**, right-click **Copy Export File**, and then click **Start Job at Step**. In the **Start Jobs – MIA-SQL** dialog box, notice that the job fails, and then click **Close**.
6. To view the job history to find the reason for the failure, right-click **Copy Export File**, and then click **View History**. In the **Log File Viewer – MIA-SQL** window, expand the failed job by clicking the plus sign next to the error symbol, scroll the window to the right and until the **Message** column is visible, and then note the reason for the failure. The failure reason should show as follows:

```
Executed as user: ADVENTUREWORKS\ServiceAcct. Access is denied.  Process Exit Code 1.
The step failed.
```

   Close the **Log File Viewer – MIA-SQL** window. The job step failed because the service account does not have permission to access the source and target folders in the file system. (One solution would be to grant access to the folders to the service account, but instead you will create a credential, and then link it to a proxy account in the next demonstration.)

7. In Solution Explorer, double-click the **Demo 2 - credential.sql** script file.
8. Execute the code under the comment that begins **Task 1** to create a credential called **FileOperation** that is linked to the **ADVENTUREWORKS\FileOps** domain user with the secret **Pa$$w0rd**.
9. Execute the code under the comment that begins **Task 2** to examine the contents of the **sys.credentials** catalog view. One row should be returned for the credential that you have just created.
10. Leave SQL Server Management Studio open for the next demonstration.

Lesson 3

Configuring Proxy Accounts

Contents:

Question and Answers

Question: Why are credentials stored in the **master** system database and proxy accounts stored in the **msdb** system database?

> **Answer:** Credentials are not solely used for security in SQL Server Agent.

Demonstration: Configuring Proxy Accounts

Demonstration Steps

1. Ensure that you have completed the previous demonstration in this module.
2. On 20764-MIA-SQL, in Solution Explorer, double-click the **Demo 3 - proxy.sql** script file.
3. Execute the code under the comment that begins **Task 1** to create a new proxy account that is linked to the **FileOperation** credential that you created in the last demonstration.
4. Execute the code under the comment that begins **Task 2** to examine the **dbo.sysproxies** catalog view.
5. Execute the code under the comment that begins **Task 3** to examine the contents of the **dbo.syssubsystems** system view. Note that the **CmdExec** subsystem has a **subsystem_id** of **3**.
6. Execute the code under the comment that begins **Task 4** to associate the **FileOp** proxy account with the **CmdExec** subsystem (**@subsystem_id = 3**).
7. In Object Explorer, under **Jobs**, right-click **Copy Export File**, and then click **Properties**.
8. In the **Job Properties – Copy Export File** window, on the **Steps** page, click **Edit**.
9. In the **Job Step Properties – Copy File** window, in the **Run as** box, select **FileOp**, and then click **OK**. Click **OK** in the **Job Properties – Copy Export File** window.
10. In File Explorer, browse to D:\Demofiles\Mod09\ImportData to demonstrate that the folder is empty.
11. In SQL Server Management Studio, in Object Explorer, under **Jobs**, right-click **Copy Export File**, and then click **Start Job at Step**. In the **Start Jobs – MIA-SQL** dialog box, notice that the job succeeds, and then click **Close**.
12. In File Explorer, demonstrate that the folder now contains a copy of the file from the ExportData folder. Close File Explorer.
13. Close SQL Server Management Studio without saving any changes.

Module Review and Takeaways

Best Practice

Use a Windows domain user as the SQL Server Agent service account.

Use an account that has least privileges.

Create proxy accounts that have least permissions assigned for job execution.

Review Question(s)

Question: As a general rule, why should proxy accounts not be assigned access to all of the job step subsystems?

> **Answer:** To adhere to the principle of least privilege.

Lab Review Questions and Answers

Lab: Configuring Security for SQL Server Agent

Question and Answers

Lab Review

Question: The SQL Server Integration Services package in this lab uses SQL Server Authentication to connect to the MIA-SQL instance to extract data. If the SQL Server Integration Services package were configured to use Windows authentication for its database connection, under what security context is the connection made when the **Generate Sales Log** job is executed by **ADVENTUREWORKS\Administrator**?

Assume that the exercise was successfully completed when you are selecting your answer.

( ) ADVENTUREWORKS\Administrator

( ) PromoteApp

( ) ADVENTUREWORKS\Student

( ) The SQL Server Agent service account

Answer:

( ) ADVENTUREWORKS\Administrator

( ) PromoteApp

(√) ADVENTUREWORKS\Student

( ) The SQL Server Agent service account

Module 10

Monitoring SQL Server with Alerts and Notifications

Contents:

Lesson 1

Monitoring SQL Server Errors

Contents:

Question and Answers

Question: If an error message is for information only, which of the following ranges will its severity fall into?

( ) 0 to 10

( ) 11 to 16

( ) 17 to 19

( ) 20 to 24

Answer:

(√) 0 to 10

( ) 11 to 16

( ) 17 to 19

( ) 20 to 24

Demonstration: Working with the Database Engine Error Log

Demonstration Steps

View the SQL Server Error Log

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In the D:\Demofiles\Mod10 folder, run **Setup.cmd** as Administrator.
3. Start SQL Server Management Studio and connect to the MIA-SQL Database Engine instance using Windows authentication.
4. In Object Explorer, under MIA-SQL, expand **Management**, and then expand **SQL Server Logs**.
5. Right-click **Current**, and then click **View SQL Server Log**.
6. Maximize the **Log File Viewer - MIA-SQL** window and view the log entries. Note that when you select a log entry, its details are shown in the lower pane.
7. In the Select logs pane, expand **SQL Server Agent** and select **Current**.
8. Scroll the main log entries pane to the right until you can see the **Log Type** column, and then scroll down to find an entry with the log type **SQL Server Agent**.
9. When you have finished viewing the log entries, click **Close**.
10. Minimize SQL Server Management Studio and view the contents of the C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\Log folder. If you are prompted to change your permissions to get access to the folder, click **Continue**. Note that the current SQL Server log is stored here in the file named ERRORLOG, and the current SQL Server Agent log is stored as SQLAGENT.1. The remaining log files contain log entries for other SQL Server components and services.

Cycle the Log File

1. In SSMS, click **New Query**.
2. In the query window, type the following Transact-SQL code, and then click **Execute**:

```
EXEC sys.sp_cycle_errorlog;
```

3. In Object Explorer, right-click the **Current** SQL Server log and click **View SQL Server Log**.
4. Note that the log has been reinitialized, and then click **Close**.
5. Close the query window without saving changes, but leave SSMS open for the next demonstration.

Lesson 2
Configuring Database Mail

Contents:

Question and Answers

Question: You are troubleshooting Database Mail. You want to see a list of the email messages that have been successfully sent and a list of email messages that could not be sent. Where can you find this information?

> **Answer:** Query the **dbo.sysmail_sentitems** and **dbo.sysmail_faileditems** views in the **msdb** database.

Demonstration: Configuring Database Mail

Demonstration Steps

Create a Database Mail Profile

1. In SSMS, in Object Explorer, under **MIA-SQL**, under **Management**, right-click **Database Mail**, and then click **Configure Database Mail**.
2. On the **Welcome to Database Mail Configuration Wizard** page, click **Next**.
3. On the **Select Configuration Task** page, click **Set up Database Mail by performing the following tasks:**, and click **Next**.
4. On the **New Profile** page, in the **Profile name** box, type **SQL Server Agent Profile**.
5. Click **Add**, and then, in the **Add Account to Profile 'SQL Server Agent Profile'** dialog box, click **New Account**.
6. In the **New Database Mail Account** dialog box, enter the following details, and then click **OK**:
   - **Account name**: AdventureWorks Administrator
   - **E-mail address**: administrator@adventureworks.msft
   - **Display name**: Administrator (AdventureWorks)
   - **Reply e-mail**: administrator@adventureworks.msft
   - **Server name**: mia-sql.adventureworks.msft
7. On the **New Profile** page, click **Next**.
8. On the **Manage Profile Security** page, select **Public** for the **SQL Server Agent Profile** profile, and set its **Default Profile** setting to **Yes**. Then click **Next**.
9. On the **Configure System Parameters** page, click **Next**.
10. On the **Complete the Wizard** page, click **Finish**, and when configuration is complete, click **Close**.

Send a Test Email Message

1. In Object Explorer, right-click **Database Mail** and click **Send Test E-Mail**.
2. In the **Send Test E-Mail from MIA-SQL** dialog box, ensure that the **SQL Server Agent Profile** database mail profile is selected, in the **To** text box, type **student@adventureworks.msft**, and then click **Send Test E-mail**.
3. Using Windows Explorer, view the contents of the C:\inetpub\mailroot\Drop folder, and verify that an email message has been created.
4. Double-click the message to view it in Outlook. When you have read the message, close it and minimize the **Drop** folder window.

5. In the **Database Mail Test E-Mail** dialog box (which might be behind SQL Server Management Studio), click **OK**.

Query Database Mail System Tables

1. In SSMS, on the **File** menu, point to **Open**, and then click **Project/Solution**.
2. In the **Open Project** dialog box, navigate to **D:\Demofiles\Mod10\Demo**, click **Demo.ssmssln**, and then click **Open**.
3. In Solution Explorer, double-click **Demo 2 – database mail.sql**, review the code, and then click **Execute**.
4. View the results. The first result set shows system events for Database Mail, and the second result set shows records of email messages that have been sent.
5. Keep the solution and SQL Server Management Studio open for the next demonstration.

Lesson 3

Operators, Alerts, and Notifications

Contents:

Question and Answers

Question: True or false? SQL Server Database Mail can only be used for sending alerts and notifications.

( ) True

( ) False

Answer:

( ) True

(√) False

Resources

Overview of SQL Server Alerts

Best Practice: It is considered good practice to configure notifications for all error messages with severity level 19 and above.

Demonstration: Configuring SQL Server Agent Operators

Demonstration Steps

Enable a Mail Profile for SQL Server Agent

1. Ensure that you have completed the previous demonstrations in this module.
2. In SSMS Object Explorer, under **MIA-SQL**, right-click **SQL Server Agent**, and then click **Properties**.
3. In the **SQL Server Agent Properties** dialog box, on the **Alert System** page, select **Enable mail profile**, and then in the **Mail profile** drop-down list, select **SQL Server Agent Profile**, and click **OK**.
4. Close Management Studio.
5. Start SQL Server Management Studio and connect to the MIA-SQL Database Engine instance using Windows authentication.
6. In Object Explorer, right-click **SQL Server Agent** and click **Restart**. In the **User Account Control** dialog, click **Yes**. When prompted to confirm, click **Yes**.

Create an Operator

1. In Solution Explorer, double-click **Demo 3 - operators.sql**.
2. Select the code under the comment that begins **Task 2**, and then click **Execute** to create a new operator called **Student**.

Configure a Job to Notify an Operator

1. In Object Explorer, expand **SQL Server Agent**, expand **Jobs** and view the existing jobs.
2. Right-click **Back Up Database – AdventureWorks**, and click **Properties**.
3. In the **Job Properties - Back Up Database - AdventureWorks** dialog box, on the **Notifications** tab, select **E-mail**, select **Student**, select **When the job completes**, and then click **OK**.
4. In Object Explorer, expand **Operators**, right-click **Student** and click **Properties**. On the **Notifications** page, click **Jobs**, note the job notifications that have been defined for this operator, and then click **Cancel**.

5. Under **Jobs**, right-click **Back Up Database – AdventureWorks**, and click **Start Job at Step**. When the job has completed, click **Close**.
6. Under **Operators**, right-click **Student**, and click **Properties**. On the **History** page, note the most recent notification by email attempt, and then click **Cancel**.
7. In Windows Explorer, in the C:\inetpub\mailroot\Drop folder, verify that a new email message has been created.
8. Double-click the most recent file in C:\inetpub\mailroot\Drop to view it in Outlook. Then, when you have read the message, close it, and minimize the **Drop** window.
9. Keep the solution and SQL Server Management Studio open for the next demonstration.

Demonstration: Configuring SQL Server Agent Alerts

Demonstration Steps

Create an Alert

1. In SSMS, in Object Explorer, under **SQL Server Agent**, right-click **Alerts**, and click **New Alert**.
2. In the **New Alert** dialog box, on the **General** page, in the Name box, type **Log Full Alert**. In the **Type** drop-down list, note that you can configure alerts on WMI events, performance monitor conditions, and SQL Server events. Select **SQL Server event alert**, click **Error number**, and then in the **Error number** box, type **9002** (which is the error number raised by SQL Server when a database transaction log is full).
3. In the **New Alert** dialog box, on the **Response** page, select **Notify operators**, and then select the **E-mail** check box for the **Student** operator.
4. In the **New Alert** dialog box, on the **Options** page, under **Include alert error text in**, select **E-mail**. Then click **OK**.

Test an Alert

1. In Solution Explorer, double-click **Demo 4 - alerts.sql**, and then click **Execute**. Wait while the script fills a table in the **TestAlertDB** database. When the log file for that database is full, error 9002 occurs.
2. In Object Explorer, expand **Alerts**, right-click **Log Full Alert**, and then click **Properties**.
3. On the **History** page, note the **Date of last alert** and **Date of last response** values, and then click **Cancel**.
4. Using Windows Explorer, in the C:\inetpub\mailroot\Drop folder, verify that a new email message has been created.
5. Double-click the most recent message to view it in Outlook. Then, when you have read the message, close it, and close the **Drop** window.
6. Leave the solution and SSMS open for the next demonstration.

Lesson 4

Alerts in Azure SQL Database

Contents:

Question and Answers

Question: Which of the following metrics **cannot** be used as the basis for an Azure SQL Database alert?

( ) DTU percentage

( ) SQL Server error number 9002

( ) Total database size

( ) CPU percentage

( ) Blocked by Firewall

Answer:

( ) DTU percentage

(√) SQL Server error number 9002

( ) Total database size

( ) CPU percentage

( ) Blocked by Firewall

Demonstration: Configuring Alerts in Azure SQL Database

Demonstration Steps

1. On 20764A-MIA-SQL, open Internet Explorer and browse to **https://portal.azure.com/**.
2. Sign in to the Azure portal with your Azure Pass or Microsoft Account credentials.
3. In the menu, click **SQL databases**, and then on the **SQL databases** blade, click **AdventureWorksLT**.
4. On the **Settings** blade, click **Alert rules**. On the **Alert rules** blade, click **Add alert**.
5. On the **Add an alert rule** blade, notice that the **Resource** box is automatically populated with the database name.
6. Configure the alert using the following values, and then click **OK**:
   - **Name**: AdventureWorksLT DTU usage alert
   - **Metric**: DTU percentage
   - **Condition**: greater than
   - **Threshold**: 1
   - **Period**: Over the last five minutes
   - **Email owners, contributors, and readers**: Select if you want to demonstrate the content of an Azure alert email message, and you have access to the mailbox for the email address to which the Azure subscription is registered.
7. In SSMS, in Solution Explorer, double-click **Demo 5 - azure.sql**.
8. On the **Query** menu, point to **Connection**, and then click **Change Connection**. Connect to the Azure SQL Database server hosting your copy of the **AdventureWorksLT** database. (You must use the credentials you configured when you configured the Azure SQL Database server for this connection. If you are unsure of the server name, it is shown on the **AdventureWorksLT** blade of the Azure portal.)

9. After the query window is connected to your Azure SQL Database server, on the toolbar, in the **Available Databases** list, click **AdventureWorksLT**.
10. On the toolbar, click **Execute**. The query executes a simple SELECT statement 200 times, which should raise DTU consumption on the database above 1 percent.
11. In the Azure portal, wait for the **Alert rules** blade to update so that the rule shows a **LAST ACTIVE** value other than **Never**, indicating that the alert was triggered. This could take several minutes and you might need to refresh the page.
12. On the **Alert rules** blade, double-click **AdventureWorksLT DTU Usage alert**. Notice that a line chart shows the alert metric over time, with the threshold marked as a dotted line.
13. Click **Delete**, then click **Yes** to remove the rule.
14. If you are planning to show the content of an alert email message, log into the mailbox for the Azure subscription owner and examine mail delivered from **Microsoft Azure Alerts**.
15. Close Internet Explorer, and then close SSMS without saving any changes.

Module Review and Takeaways

Best Practice

When using Database Mail, and planning notifications and alerts in SQL Server, consider the following best practices:

- Configure different Database Mail profiles for different usage scenarios.
- Provide limited access to the ability to send email messages from the Database Engine.
- Implement a retention policy for Database Mail log and mail auditing.
- Create a fail-safe operator.
- Define alerts for severe error messages.

Review Question(s)

Question: True or false? You want to designate a colleague in the IT team as an operator, but this colleague does not have a login in the SQL Server instance. You must define a login for your colleague before they can be configured as an operator.

( ) True

( ) False

> **Answer:**
>
> ( ) True
>
> (√) False

Question: You are planning to send notifications from SQL Server, and think it might be easier to use NET SEND notifications instead of email. Why should you not do this?

> **Answer:** Broadcast messages are generally disabled on most modern operating systems, so NET SEND messages might not appear. Notification by NET SEND is marked for deprecation, and will be removed in a future release of SQL Server.

Lab Review Questions and Answers

Lab: Monitoring SQL Server with Alerts and Notifications

Question and Answers

Lab Review

Question: Under what circumstances would email notifications be sent to the DBA Team operator you created?

> **Answer:** The DBA Team operator is the fail-safe operator. The fail-safe operator receives notifications and alerts if **msdb** is inaccessible, or no pager operators are on duty.

Module 11

Introduction to Managing SQL Server Using PowerShell

Contents:

Lesson 1
Getting Started with Windows PowerShell

Contents:

Question and Answers

Question: What is a PowerShell alias?

( ) A PowerShell variable.

( ) The Unix alternative to PowerShell cmdlets.

( ) A familiar shortcut for a PowerShell cmdlet.

( ) A way of getting more information about a cmdlet.

( ) The full version of a cmdlet.

Answer:

( ) A PowerShell variable.

( ) The Unix alternative to PowerShell cmdlets.

(√) A familiar shortcut for a PowerShell cmdlet.

( ) A way of getting more information about a cmdlet.

( ) The full version of a cmdlet.

Resources

PowerShell Providers

Additional Reading: To learn more about Windows PowerShell, see *Learn Windows PowerShell in a Month of Lunches* (Don Jones and Jeffery D. Hicks). It is published by Manning and available as an e-book.

Demonstration: Exploring SQL Server Management Objects

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Run **Setup.cmd** in the D:\Demofiles\Mod11 folder as Administrator.
3. If a User Account Control window appears, click **Yes**.
4. On the taskbar, right-click **Windows PowerShell**, and then click **Run as Administrator**. In the **User Account Control** dialog box, click **Yes**.
5. In the Powershell console, verify that the text in the title bar reads “Administrator: Windows PowerShell”.
6. Clear the screen by typing **cls**, and then press Enter.
7. Type **cd \**, and then press Enter.
8. To understand why commands prompts appear to be working in PowerShell, type **Get-Alias**, and then press Enter.
9. Type **Get-Alias c***, and then press Enter
10. Type **Update-Help**, and then press Enter. Wait for the help files to install.

11. Type **Get-Help**, and then press Enter.
12. Type **Get-Help Get-Help**, and then press Enter.
13. Type **Get-Help Get-Item**, and then press Enter.
14. For each of the following cmdlets, type the cmdlets and then press Enter to show the different sets of parameters:

    Get-Help Get-Item –Examples
    Get-Help Get-Item –Detailed
    Get-Help Get-Item –ShowWindow

15. Close the Get-Item Help window.
16. Type **Get-I**, and then press TAB. Press TAB repeatedly to show all the cmdlets that start with Get-I. Press SHIFT+TAB to step backwards. Note how the capitalization is automatically corrected.
17. Press ESC.
18. Type **Get-PSProvider**, and then press Enter.
19. Type **Get-PSDrive**, and then press Enter.
20. Type **Import-Module SQLPS**, and then press Enter.
21. Repeat steps 18 and 19 again and note the additions to the list.
22. Type **Get-ChildItem** or **dir**, and then press Enter.
23. Type **Set-Location SQL** or **cd SQL**, and then press Enter.
24. Type **Get-ChildItem**, and then press Enter.
25. Type **Set-Location MIA-SQL**, and then press Enter.
26. Type **Get-ChildItem**, and then press Enter.
27. Type **Set-Location Default**, and then press Enter.
28. Type **Get-ChildItem**, and then press Enter.
29. Review the list of objects and consider how they map to objects in SQL Server.
30. Type **Set-Location Databases**, and then press Enter.
31. Type **Get-ChildItem**, and then press Enter.
32. Type **Exit**, and then press Enter.

Lesson 2

Configure SQL Server Using PowerShell

Contents:

Question and Answers

Put the following steps in order by numbering each to indicate the correct order.

	Steps
	Import SQL PowerShell module.
	Use the SQL PowerShell provider to navigate to an SMO object.
	Assign an SMO object to a variable.
	Discover the object's properties using Get-Member.
	Amend a property.
	Apply the amendment using the Alter method.

Answer:

	Steps
1	Import SQL PowerShell module.
2	Use the SQL PowerShell provider to navigate to an SMO object.
3	Assign an SMO object to a variable.
4	Discover the object's properties using Get-Member.
5	Amend a property.
6	Apply the amendment using the Alter method.

Lesson 3

Administer and Maintain SQL Server with PowerShell

Contents:

Resources

Managing Users and Roles

Additional Reading: For more information about automating SQL tasks, see *SQL Server 2014 with PowerShell v5 Cookbook* (Donabel Santos, Packt Publishing).

Demonstration: PowerShell for Troubleshooting

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. On the taskbar, right-click **Windows PowerShell**, and then click **Run ISE as Administrator**. In the **User Account Control** dialog box, click **Yes**.
3. On the **File** menu, click **Open**.
4. In the **Open** dialog box, navigate to D:\Demofiles\Mod11, click **InspectSqlInstances.ps1**, and then click **Open**.
5. Select the code under the **#1#** comment, and then on the toolbar, click **Run Selection** to import the SQL module.
6. Select the code under the **#2#** comment, and then on the toolbar, click **Run Selection** to set the location.
7. Select the code under the **#3#** comment, and then on the toolbar, click **Run Selection** to display the instances of SQL Server.
8. Select the code under the **#4#** comment, and then on the toolbar, click **Run Selection** to display a formatted list of SQL Server instances.
9. Select the code under the **#5#** comment, and then on the toolbar, click **Run Selection** to display a list of databases in descending order of size.
10. Select the code under the **#6#** comment, and then on the toolbar, click **Run Selection** to display a tabular list of databases in descending order of size.
11. Select the code under the **#7#** comment, and then on the toolbar, click **Run Selection** to output the information to a text file.
12. Select the code under the **#8#** comment, and then on the toolbar, click **Run Selection** to output the information to an XML file.
13. Select the code under the **#9#** comment, and then on the toolbar, click **Run Selection** to output the information to an Excel file.
14. Close PowerShell ISE without saving any changes.

Lesson 4
Managing Azure SQL Databases Using PowerShell

Contents:

Question and Answers

Question: What advantages are there to using PowerShell with Microsoft Azure?

Answer: Answers will vary, depending on students' exposure to Microsoft Azure.

Demonstration: Creating an Azure SQL Database with PowerShell

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.

2. On the taskbar, right-click **Windows PowerShell**, and then click **Run as Administrator**. In the **User Account Control** dialog box, click **Yes**.

3. In the console, type **cd\**, and then press Enter.

4. Type **cls**, and then press Enter.

5. Link your Azure account to PowerShell by typing the following cmdlet, and then press Enter.

```
Add-AzureRmAccount
```

6. Type **Y** to enable data collection.

7. Wait for the Azure sign-on screen appear, type the user name and password you use to sign in to the Azure portal, and then click **Sign in**.

8. If you have already linked your Azure account to PowerShell on this VM, type the following cmdlet:

```
Login-AzureRmAccount
```

9. Wait for the Azure sign to appear, type the user name and password you use to sign in to the Azure portal, and then click **Sign in**.

10. Copy the subscription ID from the previous step, type the following cmdlet, and then press Enter. Substitute the subscription ID that you copied earlier for <yoursubscriptionid>.

```
Select-AzureRmSubscription –SubscriptionID <yoursubscriptionid>
```

11. To return a list of Azure data center locations, type the following cmdlet, and then press Enter.

```
(Get-AzureRmResourceProvider –ListAvailable | Where-Object {$ .ProviderNamespace –
eq'Microsoft.Sql'}).Locations
```

12. To create a resource group, type the following cmdlet, and then press Enter. Substitute a location from the list returned in the previous step for <location>:

```
New-AzureRmResourceGroup -Name "20764ATest" –Location "<location>"
```

13. To create a new server in the resource group you just created, type the following cmdlet, and then press Enter. Substitute the location used in the previous step for <location>. Substitute a unique server name for <your server name>. This must be unique throughout the whole Azure service, so cannot be specified here. A suggested format is sql2016ps-<your initials><one or more digits>. For example, sql2016ps-js123. Letters must be lowercase.

```
New-AzureRmSqlServer -ResourceGroupName "20764ATest" -ServerName "<your server
name>" -Location "<location>" -ServerVersion "12.0"
```

14. In the **Windows PowerShell credential request** dialog box, in the **User name** box, type **psUser**, in the **Password** box, type **Pa$$w0rd**, and then click **OK**. Wait for Azure to create the administrator for your server and for the console to display the information.

15. Type the following cmdlet, and then press Enter to create a variable to store your external IP address. Substitute your own relevant information for the <your external ip> parameter:

```
$currentIP = "<your external ip>"
```

Note: You can get your current external IP address from the Azure Portal (see the value returned by the "Add Client IP" button on the firewall for an existing server), or from third-party services such as www.whatismyip.com.

16. Type the following cmdlet, and then press Enter to create a firewall rule that permits you to connect to the server. Substitute your own relevant information for the <your server name> parameter:

```
New-AzureRmSqlServerFirewallRule –ResourceGroupName "20764ATest" –ServerName "<your
server name>" –FirewallRuleName "Firewall1" –StartIpAddress $currentIP –EndIpAddress
$currentIP
```

17. Type the following cmdlet, and then press Enter to create an Azure SQL Database on the server you have just created. Substitute the name of your server for <your server name>:

```
New-AzureRmSqlDatabase -ResourceGroupName "20764ATest" -ServerName "<your server
name>" -DatabaseName "testpsdb" -Edition Standard -RequestedServiceObjectiveName
"S1"
```

    This will take a few minutes to complete. Wait for the details of the new database to be returned—this indicates that the database has been created.

18. Close Windows PowerShell.

Module Review and Takeaways

Best Practice

Use aliases when working with the console window, but make scripts easy to read by using correctly capitalized cmdlet names.

Review Question(s)

Question: What tasks might benefit from automating with PowerShell for your SQL Server environment?

Answer: Answers will vary and may include: reporting issues from error logs, reporting issues from other Microsoft installed products that have an impact on SQL Server, and reporting on SQL Server instances.

Lab Review Questions and Answers

Lab: Using PowerShell to Manage SQL Server

Question and Answers

Getting Started with PowerShell

Question: True or false? PowerShell providers offer an alternative to using SQL Management Objects.

( ) True

( ) False

Answer:

( ) True

(√) False

Using PowerShell to Change SQL Server Settings

Question: What is an SMO object?

( ) A SQL PowerShell provider.

( ) An object with which part of SQL Server can be managed programmatically.

( ) A SQL Server 2016 feature that improves performance.

( ) Part of the Windows operating system.

( ) The top level object in the SQL Server hierarchy.

Answer:

( ) A SQL PowerShell provider.

(√) An object with which part of SQL Server can be managed programmatically.

( ) A SQL Server 2016 feature that improves performance.

( ) Part of the Windows operating system.

( ) The top level object in the SQL Server hierarchy.

Lab Review

Question: Can you name three ways of getting information about a cmdlet?

Answer: Tab completion, Get-Help, and Get-Command.

Question: When would you use a PowerShell provider?

Answer: To access a data store such as SMO objects, the file system, or the registry.

Module 12

Tracing Access to SQL Server with Extended Events

Contents:

Lesson 1

Extended Events Core Concepts

Contents:

Question and Answers

Question: Which system DMV provides the list of events configured in an active Extended Events session?

( ) sys.dm_xe_session _targets

( ) sys.dm_xe_session_events

( ) sys.dm_xe_sessions

( ) sys.dm_xe_session_event_actions

Answer:

( ) sys.dm_xe_session _targets

(√) sys.dm_xe_session_events

( ) sys.dm_xe_sessions

( ) sys.dm_xe_session_event_actions

Demonstration: Creating an Extended Events Session

Demonstration Steps

1. Start the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In the D:\Demofiles\Mod12 folder, run **Setup.cmd** as Administrator.
3. Click **Yes** in the User Account Control window and wait for the script to finish.
4. Start SQL Server Management Studio and connect to the **MIA-SQL** database engine instance using Windows authentication.
5. On the **File** menu, point to **Open**, and then click **Project/Solution**. In the **Open Project** dialog box, navigate to the **D:\Demofiles\Mod12** folder, click **Demo.ssmssln**, and then click **Open**.
6. In Solution Explorer, double-click **Demo 1 – create xe session.sql**.
7. Select code under the comment that begins **Step 1**, and then click **Execute** to create an Extended Events session.
8. Select code under the comment that begins **Step 2**, and then click **Execute** to verify that the session metadata is visible.
9. Select code under the comment that begins **Step 3**, and then click **Execute** to start the session and execute some queries.
10. Select code under the comment that begins **Step 4**, and then click **Execute** to query the session data.
11. Select code under the comment that begins **Step 5**, and then click **Execute** to refine the session data query.
12. In Object Explorer, under **MIA-SQL**, expand **Management**, expand **Extended Events**, and then expand **Sessions**.
13. Expand **SqlStatementCompleted**, and then double-click **package0.ring_buffer**.
14. In the **Data** column, click the XML value, and note that this is the same data that is returned by the query under the comment that begins Step 4 (note that additional statements will have been captured because you ran the code earlier).

15. In Object Explorer, right-click **SqlStatementCompleted** and then click **Watch Live Data**.
16. In the **Demo 1 – create xe sessions.sql** query pane, select the code under the comment that begins **-- Step 7**, and then click **Execute** to execute some SQL statements.
17. Return to the **MIA-SQL – SqlStatementCompleted: Live Data** pane. Wait for the events to be captured and displayed; this can take a few seconds. Other SQL statements from background processes might be captured by the session.
18. In the **Demo 1 – create xe sessions.sql** query pane, select the code under the comment that begins **-- Step 8**, and then click **Execute** to stop the session.
19. In Object Explorer, right-click **SqlStatementCompleted**, and then click **Properties**. Review the settings on the **General**, **Events**, **Data Storage** and **Advanced** pages, if necessary referring back to the session definition under the comment that begins **-- Step 1**.
20. In the **Session Properties** dialog box, click **Cancel**.
21. Select the code under the comment that begins **-- Step 10**, and then click **Execute** to drop the session.
22. Keep SQL Server Management Studio open for the next demonstration.

Lesson 2

Working with Extended Events

Contents:

Question and Answers

Categorize each Extended Events target type into the appropriate category. Indicate your answer by writing the category number to the right of each item.

Items	
1	Ring buffer target
2	Event file target
3	Histogram target
4	Event tracking for Windows target
5	Event pairing target
6	Event counter target

Category 1		Category 2
Written to Memory Buffers		Written to File on Disk

Answer:

Category 1		Category 2
Written to Memory Buffers		Written to File on Disk
Ring buffer target Histogram target Event pairing target Event counter target		Event file target Event tracking for Windows target

Demonstration: Tracking Session-Level Waits

Demonstration Steps

1. In SSMS, in Solution Explorer, double-click **Demo 2 - track waits by session.sql**.
2. In Object Explorer, expand **Management**, expand **Extended Events**, right-click **Sessions**, and then click **New Session**.
3. In the **New Session** dialog box, on the **General** page, in the **Session name** box, type **Waits by Session**.
4. On the **Events** page, in the **Event library** box, type **wait**, and then, in the list below, double-click **wait_info** to add it to the **Selected events** list.
5. Click **Configure** to display the **Event configuration options** list.
6. In the **Event configuration options** list, on the **Global Fields (Actions)** tab, select **session_id**.
7. On the **Filter (Predicate)** tab, click **Click here to add a clause**. In the **Field** list, click **sqlserver.session_id**, in the **Operator** list, click **>**, and then in the **Value** box, type **50**. This filter will exclude most system sessions from the session.
8. On the **Data Storage** page, click **Click here to add a target**. In the **Type** list, click **event_file**, in the **File name on server** box, type **D:\Demofiles\Mod12\waitbysession**, in the first **Maximum file size** box, type **5**, in the second **Maximum file size** box, click **MB**, and then click **OK**.
9. In Object Explorer, expand **Sessions**, right-click **Waits by Session**, and then click **Start Session**.
10. In Windows Explorer, in the D:\Demofiles\Mod12 folder, right-click **start_load_1.ps1**, and then click **Run with PowerShell**. If a message is displayed asking you to confirm a change in execution policy, type **Y**, and then press ENTER. Leave the workload to run for a minute or so before proceeding.
11. In SSMS, in the **Demo 2 - track waits by session.sql** pane, select the code under the comment that begins **Step 14**, click **Execute**, and then review the results.
12. Select the code under the comment that begins **-- Step 15**, and then click **Execute** to stop and drop the session, and to stop the workload.
13. In Windows Explorer, in the D:\Demofiles\Mod12 folder, note that one (or more) files with a name matching **waitbysession*.xel** have been created.
14. Close Windows Explorer, close SSMS without saving changes, and then in the Windows PowerShell window, press ENTER to close the window.

Module Review and Takeaways

Question: Which of the following sources does **not** contain detailed information about Extended Events event definitions?

( ) SQL Server Management Studio Extended Events GUI.

( ) The DMV sys.dm_xe_objects.

( ) The SQL Server 2016 Technical Documentation.

Answer:

( ) SQL Server Management Studio Extended Events GUI.

( ) The DMV sys.dm_xe_objects.

(√) The SQL Server 2016 Technical Documentation.

Lab Review Questions and Answers

Lab: Extended Events

Question and Answers

Lab Review

Question: If an Extended Events session has no targets defined, how would you view the data generated by the session?

Answer: Use the **Watch Live Data** feature in SSMS.

Module 13

Monitoring SQL Server

Contents:

Lesson 1
Monitoring Activity

Contents:

Question and Answers

Question: Why might you use the **sys.dm_os_performance_counters** system DMV, instead of Performance Monitor, to access SQL Server counters?

> **Answer:** Querying the **sys.dm_os_performance_counters** system DMV enables you to compare information from system performance counters and from internal performance tools such as DMOs.

Demonstration: Using DMOs to View Activity

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In the D:\Demofiles\Mod13 folder, right-click **Setup.cmd**, and then click **Run as administrator**.
3. In the **User Account Control** dialog box, click **Yes**, and then wait for the script to complete.
4. In the D:\Demofiles\Mod13 folder, run **Workload1.cmd**.
5. Start SQL Server Management Studio, and then connect to the **MIA-SQL** database engine instance by using Windows Authentication.
6. On the **file** menu point to **open**, click **Project/Solution**, browse to D:\Demofiles\Mod13\Demo folder, open the **Demo.ssmssln** solution.
7. In Solution Explorer, expand **Queries**, and then double-click **Demo 1 - DMO.sql**.
8. Execute the code under the heading that begins with **Task 2** to view currently executing requests. Approximately 50 rows should be returned, but most are system requests.
9. Execute the code under the heading that begins with **Task 3** to view user processes.
10. Execute the code under the heading that begins with **Task 4** to filter executing requests by user sessions to show only user activity.
11. Execute the code under the heading that begins with **Task 5** to retrieve details of the Transact-SQL batch that is associated with each request.
12. Execute the code under the heading that begins with **Task 6** to show details of the Transact-SQL statement that is currently executing within each batch. This statement is complex, but it is fundamentally a substring operation on the results of the previous step.
13. Execute the code under the heading that begins with **Task 7** to stop the workload script. (The workload is configured to stop when the **##stopload** global temporary table is created.)
14. Execute the code under the heading that begins with **Task 8** to examine the contents of the query plan cache.
15. Execute the code under the heading that begins with **Task 9** to identify the top 10 most expensive queries in the query plan cache based on average logical reads.
16. Execute the code under the heading that begins with **Task 10** to view I/O statistics for database files.
17. Execute the code under the heading that begins with **Task 11** to view wait statistics. The purpose of this query is to demonstrate the range of wait types that wait statistics are collected for.
18. Leave SQL Server Management Studio open for the next demonstration.

Demonstration: Using Activity Monitor in SQL Server Management Studio

Demonstration Steps

1. In SQL Server Management Studio, in Solution Explorer, double-click **Demo 2a - blocker.sql**.
2. Review the contents of the file and notice that it starts a transaction without committing it, and then click **Execute**.
3. In Solution Explorer, double-click **Demo 2b - blocked.sql**, and then click **Execute**. Notice that a result set is not returned and the query remains running; this query is blocked from accessing the **HR.Employees** table by the transaction that you opened in the previous step.
4. In Object Explorer, right-click **MIA-SQL**, and then click **Activity Monitor**.
5. In the MIA-SQL Activity Monitor pane, click **Processes** to expand the **Processes** section.
6. In the **Processes** section, in the **Database Name** column, in the column header, click the **Filter** button, and then click **InternetSales**.
7. Notice that one of the processes has a **Task State** of **SUSPENDED**; this is the query that is running in the Demo 2b - blocked.sql query pane.
8. Point to the column header for the **Blocked By** column to demonstrate that the column tooltip describes the DMO column that contains the information—the **sys.dm_os_waiting_tasks.blocking_session_id**.
9. In the **SUSPENDED** row, note the value in the **Blocked By** column. This is the session ID of the blocking session—the query in the Demo 2a - blocker.sql query pane.
10. In the **Processes** section, in the **Session ID** column, in the column header, click the **Filter** button, and then click the value of the session that you identified as the blocker in the previous step. Only one row should now be visible in the **Processes** section. Notice that the value in the **Head Blocker** column is **1**. This indicates that this session is the first in a blocking chain.
11. In the **Processes** section, right-click the row, and then click **Kill Process**. In the **&Kill Process** dialog box, click **Yes** to roll back the query in the Demo 2a - blocker.sql query pane.

Note: Note that processes should be killed only as a last resort.

12. Close the Activity Monitor pane.
13. In the **Demo 2b - blocked.sql** query pane, notice that the query has completed because the block has been removed.
14. Leave SQL Server Management Studio open for the next demonstration.

Demonstration: Using Performance Monitor

Demonstration Steps

1. Click **Start**, click **Administrative Tools**, and then double-click **Performance Monitor**.
2. In the Performance Monitor window, in the leftmost pane, expand **Data Collector Sets**, and then expand **System**.
3. Click **System Performance**, and then in the rightmost pane, double-click **Performance Counter**.
4. In the **Performance Counter Properties** dialog box, observe the different performance counters that this set collects, and then click **Cancel**.

5. Right-click **System Performance**, and then click **Start**. Note that the symbol for the **System Performance** collector set changes to reflect that it is running. The collector set will collect data for one minute before it stops automatically. Wait for the collector set to finish and the symbol to change back to its original form.
6. On the **Action** menu, click **Latest Report**. Notice that new nodes are added to the tree in the leftmost pane. In the rightmost pane, expand each section of the report to demonstrate the information that has been collected.
7. When you have finished reviewing the report, in the leftmost pane, click **Performance Monitor**. Notice that the rightmost pane changes to show a chart, with the **% Processor Time** counter preselected. In the rightmost pane, right-click anywhere in the pane, and then click **Add Counters**.
8. In the **Add Counters** dialog box, in the **Available counters** list, select the following counters, and then click **Add >>** to add them:
    - **SQLServer:Memory Manager: Total Server Memory (KB)**
    - **SQLServer:Memory Manager: Target Server Memory (KB)**
    - **SQLServer:Databases: Percent Log Used** (Instance **InternetSales**)
    - **MSSQL$SQL2:Memory Manager: Total Server Memory (KB)**
9. Click **OK**.
10. Review the changes to the Performance Monitor chart and then close Performance Monitor.
11. In Solution Explorer, double-click **Demo 3 - counters.sql**. Execute the code in the file to demonstrate that SQL Server Performance Monitor counters are accessible by using the **sys.dm_os_performance_counters** system DMV.
12. Leave SQL Server Management Studio open for the next demonstration.

Lesson 2

Capturing and Managing Performance Data

Contents:

Question and Answers

Question: True or false? You can use the SQL Server data collector for real-time monitoring.

( ) True

( ) False

Answer:

( ) True

(√) False

Demonstration: Configuring Data Collector

Demonstration Steps

Configure the Management Data Warehouse

1. In Object Explorer, under **MIA-SQL**, expand **Management**, right-click **Data Collection**, point to **Tasks**, and then click **Configure Management Data Warehouse**.
2. In the Configure Management Data Warehouse Wizard window, click **Next**.
3. On the Configure Management Data Warehouse Storage page, click **New**.
4. In the **New Database** dialog box, in the **Database name** box, type **MDW**, and then click **OK**.
5. On the Configure Management Data Warehouse Storage page, click **Next**.
6. On the Map Logins and Users page, review the available options, and then click **Next**.
7. On the Complete the Wizard page, click **Finish**.
8. Wait for the configuration process to complete, and then click **Close**.

Enroll the **MIA-SQL** instance for data collection

1. In Object Explorer, under **Management**, right-click **Data Collection**, point to **Tasks**, and then click **Configure Data Collection**.
2. In the Configure Data Collection Wizard window, click **Next**.
3. On the Setup Data Collection Sets page, next to the **Server name** box, click the **Ellipsis** (**…**) button.
4. In the **Connect to Server** dialog box, verify that the **Server name** box has the value **MIA-SQL**, and then click **Connect**.
5. On the Setup Data Collection Sets page, in the **Database name** box, select **MDW**. Under **Select data collector sets you want to enable**, select **System Data Collection Sets**, and then click **Next**.
6. On the Complete the Wizard page, click **Finish**.
7. Wait for the configuration process to complete, and then click **Close**.

Configure a data collection set

1. In Object Explorer, under **Management**, expand **Data Collection**, and then expand **System Data Collection Sets**. Observe that four collection sets are available but one of them is stopped.
2. Right-click **Disk Usage**, and then click **Properties**.
3. In the **Data Collection Set Properties** dialog box, on the General page, click **Pick**.

4. In the **Pick Schedule for Job** dialog box, click **CollectorSchedule_Every_5min** (the row with **ID** = **2**), and then click **OK**.
5. In the Data Collection Set Properties window, click **OK**.
6. Leave SQL Server Management Studio open for the next demonstration.
7. In the D:\Demofiles\Mod13 folder, start **Workload1.cmd** and allow it to run. This script will generate some activity that will appear in the data collection reports in the demonstrations in the next lesson.

Lesson 3

Analyzing Collected Performance Data

Contents:

Question and Answers

Question: Which system data collection set report would you use to get the history of memory usage?

( ) The Server Activity report

( ) The Query Statistics report

( ) The Disk Usage report

Answer:

(√) The Server Activity report

( ) The Query Statistics report

( ) The Disk Usage report

Demonstration: Viewing the Disk Usage Report

Demonstration Steps

Force a data collection

1. In Object Explorer, under **Management**, under **Data Collection**, under **System Data Collection Sets**, right-click **Disk Usage**, and then click **Collect and Upload Now**.
2. In the **Collect and upload Data Collection Set** dialog box, click **Close**.
3. In Object Explorer, right-click **Query Statistics**, and then click **Collect and Upload Now**.
4. In the **Collect and upload Data Collection Set** dialog box, click **Close**.
5. In Object Explorer, right-click **Server Activity**, and then click **Collect and Upload Now**.
6. In the **Collect and upload Data Collection Set** dialog box, click **Close**.

Access the Disk Usage report

1. In Object Explorer, under **MIA-SQL**, expand **Databases**, right-click **MDW**, point to **Reports**, point to **Management Data Warehouse**, and then click **Management Data Warehouse Overview**.
2. Review the hyperlinks under each report name that show the last data collection date and time. Under **Disk Usage**, click the date and time hyperlink.
3. In the **Disk Usage Collection Set** report, observe the data that is available in the report, and then click **InternetSales**.
4. In the **Disk usage for database: InternetSales** report, observe the available information, and in the report pane, in the upper-left corner, click the **Navigate Backward** button to return to the **Disk Usage Collection Set** report.
5. In the **Log Trend** column, click the trend line for the **MDW** database. Note that you must click the line itself; clicking elsewhere in the cell that contains the line will not work.
6. In the **Disk Usage Collection Set – Log: [MDW]** report, observe the information that is available in the report. Note that the chart in this report is based on percentage of free space in the data file.
7. Click the **Navigate Backward** button to return to the **Disk Usage Collection Set** report.
8. In the report pane, in the top left, click the **Navigate Backward** button to return to the **Management Data Warehouse Overview: MDW** report.
9. Leave SQL Server Management Studio open for the next demonstration.

Demonstration: Viewing the Server Activity Report

Demonstration Steps

1. In SQL Server Management Studio, in the Management Data Warehouse Overview pane, in the report pane, in the upper-left corner, click the **Refresh** button.
2. In the main report, under **Server Activity**, click the date and time hyperlink.
3. In the **Server Activity History** report, observe the timeline and the six charts. Explain that some of the charts are empty because no relevant activity has taken place.
4. Demonstrate the use of the timeline; under the timeline, click the **Zoom In** button twice. Note how the range of dark blue squares, each representing a data collection point, reduces each time that you click, and the range of data in the charts changes to match the selected time range.
5. Click the **Zoom Out** button twice.
6. In the **Memory Usage** chart, click the lower line to drill through.
7. In the **SQL Server Memory Usage** report, scroll down to view the **SQL Server Internal Memory Consumption By Type** chart. Expand the **Average Memory Use by Component** section of the report to view detailed memory usage information for each SQL Server component.
8. In the report pane, in the upper-left corner, click the **Navigate Backward** button to return to the **Server Activity History** report.
9. In the report pane, in the top left, click the **Navigate Backward** button five times to return to the **Management Data Warehouse Overview** report.
10. Leave SQL Server Management Studio open for the next demonstration.

Demonstration: Viewing the Query Statistics Report

Demonstration Steps

1. In SQL Server Management Studio, in the Management Data Warehouse Overview pane, in the report pane, in the upper-left corner, click the **Refresh** button.
2. In the main report, under **Query Statistics**, click the date and time hyperlink.
3. In the **Query Statistics History** report, observe that the report uses the same timeline control as the **Server Activity** report to navigate through the data.
4. Under the **Navigate through the historical snapshots of data using the time line below**, click the ⏭ icon.
5. Under the **Top Queries by Total CPU** chart, click **Duration**, click **Total I/O**, click **Physical Reads**, and then click **Logical Writes** to demonstrate the different views of the data.
6. Click **CPU** to return to the original view.
7. Under the report, in the **Query** column of the data table click the first row.
8. In the **Query Details** report, scroll down to view the various components of the report.
9. At the bottom of the report, in the **Top Query Plans By Average CPU Per Execution** table, in the **Plan #** column, click the first row.
10. In the **Query Plan Details** report, scroll down, and in the **Query Execution Statistics** section, click **View graphical query execution plan**.
11. Note that a new pane opens that contains the query plan.

12. Leave SQL Server Management Studio open for the next demonstration.

Lesson 4

SQL Server Utility

Contents:

Question and Answers

Question: You can only have one UCP on a network segment. True or false?

( ) True

( ) False

Answer:

( ) True

(√) False

Demonstration: Creating a Utility Control Point

Demonstration Steps

Stop Data Collection

Note: Note that you must stop active data collection sets before you configure a UCP. You can restart them after the UCP setup is complete.

1. In SQL Server Management Studio, in Object Explorer, under **System Data Collection Sets**, right-click **Disk Usage**, and then click **Stop Data Collection Set**.
2. In the **Stop Data Collection Sets – MIA-SQL** dialog box, click **Close**.
3. Repeat steps 1 and 2 for the Query Statistics and Server Activity data collection sets.

Create a UCP

1. On the **View** menu, click **Utility Explorer**.
2. In the Getting Started pane, click **Create a Utility Control Point (UCP)**.
3. In the **Create Utility Control Point** Wizard, on the Introduction page, click **Next**.
4. On the Specify the Instance of SQL Server page, click **Connect**.
5. In the **Connect to Server** dialog box, verify that the **Server name** box has the value **MIA-SQL**, and then click **Connect**.
6. On the Specify the Instance of SQL Server page, click **Next**.
7. On the Utility Collection Set Account page, click **Use the SQL Server Agent service account**, and then click **Next**.
8. On the SQL Server Instance Validation page, wait for the validation to complete, and then click **Next**. For the purposes of this demonstration, you can ignore the warning for WMI configuration.
9. On the Summary of UCP Creation page, click **Next**.
10. When the configuration process completes, click **Finish**.

Connect to a UCP

1. On the leftmost side of the screen, in the Utility Explorer pane, on the toolbar, click **Disconnect from Utility**.
2. Note that this action clears the Utility Explorer pane and the Utility Explorer Content pane.
3. In the Utility Explorer pane, on the toolbar, click **Connect to Utility**.

4. In the **Connect to Server** dialog box, verify that the **Server name** box has the value **MIA-SQL**, and then click **Connect**.
5. In the Utility Explorer pane, click **Managed Instances**. Note that the UCP instance has automatically been enrolled. Data collection takes some time to complete, so data will not immediately be available for the instance.
6. Close SQL Server Management Studio without saving any changes.

Module Review and Takeaways

Review Question(s)

Question: Which SQL Server activity monitoring tools are best suited to your organization’s needs?

Answer: Answers will vary.

Lab Review Questions and Answers

Lab: Monitoring SQL Server

Question and Answers

Lab Review

Question: What are the benefits of using a central data warehouse for SQL Server performance data, instead of local collection on each server?

> **Answer:** Performance data is easier to access and report on if it is held in a single central location.

Module 14

Troubleshooting SQL Server

Contents:

Lesson 1

A Troubleshooting Methodology for SQL Server

Contents:

Question and Answers

Number each of the following troubleshooting phases to indicate their correct order.

	Steps
	Investigation Phase
	Analysis Phase
	Implementation Phase
	Validation Phase
	Create Documentation

Answer:

	Steps
1	Investigation Phase
2	Analysis Phase
3	Implementation Phase
4	Validation Phase
5	Create Documentation

Lesson 2

Resolving Service-Related Issues

Contents:

Question and Answers

Question: Which log(s) will give you the most information when the SQL Server service will not start?

( ) The Windows system log.

( ) The Windows application log.

( ) The SQL Server error log.

( ) The SQL Server error log and the Windows system log.

( ) The SQL Server error log and the Windows application log.

Answer:

( ) The Windows system log.

( ) The Windows application log.

( ) The SQL Server error log.

(√) The SQL Server error log and the Windows system log.

( ) The SQL Server error log and the Windows application log.

Demonstration: Troubleshooting Service-Related Issues

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764-MIA-DC, and 20764-MIA-SQL virtual machines are running, and log on to 20764-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In the D:\Demofiles\Mod14 folder, run **Setup.cmd** as Administrator.
3. If a User Account Control window appears, click **Yes**.
4. Try to start the **MIA-SQL\SQL2** instance using SQL Server Configuration Manager. Click the **Start** button, then type **Configuration Manager**. Click **SQL Server 2016 CTP3.3 Configuration Manager**, then click **Yes** in the **User Account Control** dialog.
5. In the left-hand pane of the **Sql Server Configuration Manager** window, click **SQL Server Services**. In the list of services in the right-hand pane, note that the **SQL Server (SQL2)** service is not running.
6. Right-click **SQL Server (SQL2)** and click **Start**. Note that the service does not start successfully, and an error message is returned. In the **SQL Server Configuration Manager** dialog, click **OK**.
7. To check the Windows system log, click **Start**, then type **Event Viewer** and press **Enter**.
8. In the left-hand pane of the **Event Viewer** window, expand **Windows Logs** and click **System**. Click on the most recent message with a **Level** of **Error** and a **Source** of **Service Control Manager**. Note that the error message states that there is a service specific error, and provides only the following details:

```
SQL Server (SQL2)
%%17113
```

9. Close the **Event Viewer** window.

10. To check the SQL Server error log, start File Explorer and go to C:\Program Files\Microsoft SQL Server\MSSQL13.SQL2\MSSQL\Log. If a permission dialog is displayed, click **Continue**. Right-click **ERRORLOG**, then click **Open with**, then click **Notepad**. Notice the last three lines of the file include the error number displayed in the Windows system log (17113), and a detailed description of the problem (scroll to the right to read the full error message). The message indicates that the data file for the **master** database cannot be found.

11. In File Explorer, go to the folder location mentioned in the error message (C:\Program Files\Microsoft SQL Server\MSSQL13.SQL2\MSSQL\DATA). If a permission dialog is displayed, click **Continue**. Notice that the folder contains the file **master.AV0001**. The demonstration simulates the situation where the **master.mdf** file has been quarantined by an anti-virus application, which has renamed it **master.AV0001**.

    (In a real-world scenario, you would need to recover the file from the quarantine system, and prevent the anti-virus application from scanning this folder, before attempting to restart the service.)

12. For the purposes of this demonstration, rename the file **master.AV0001** to **master.mdf**; right-click **master.AV0001** then click **Rename**. Replace the **AV0001** file extension with **mdf**, then press Enter. Click **Yes** in the **Rename** dialog box. Close Windows Explorer.

13. In the right-hand pane of SQL Server Configuration Manager, right-click **SQL Server (SQL2)** and click **Start**. Note that the service starts successfully.

14. Close SQL Server Configuration Manager and Notepad.

Lesson 3
Resolving Connectivity and Login Issues

Contents:

Question and Answers

Question: You want to define an alias for a named instance of the Database Engine. The alias will be used by clients using both 32-bit and 64-bit native client drivers. In SQL Server Configuration Manager, where should you define the alias?

( ) Under SQL Native Client 11.0 Configuration.

( ) Under SQL Native Client 11.0 Configuration (32-bit).

( ) Under both Under SQL Native Client 11.0 Configuration (32-bit) and Under SQL Native Client 11.0 Configuration.

> **Answer:**
>
> ( ) Under SQL Native Client 11.0 Configuration.
>
> ( ) Under SQL Native Client 11.0 Configuration (32-bit).
>
> (√) Under both Under SQL Native Client 11.0 Configuration (32-bit) and Under SQL Native Client 11.0 Configuration.

Resources

Best Practice: Using logins based on Windows authentication removes the need for you, as a database administrator, to deal with most password and authentication-related issues.

Troubleshooting Connectivity Issues

Best Practice: When you suspect a network connectivity problem in a TCP/IP network, start by testing a connection from the client to the server using the server IP address—and, if necessary, the TCP port number. Should a connection by IP address fail, the issue is likely to be with the network—perhaps a routing or firewall problem. If a connection by IP address is successful but a connection by name fails, the issue is likely to be with name resolution—either DNS, the SQL Server Browser service, or SQL Server aliases.

Module Review and Takeaways

Question: How do you rate your troubleshooting skills? What could you do to improve them?

Answer: Answers will vary by individual.

Tools

Participating in online forums, where developers and administrators post questions about SQL Server issues, is a good way to practice your troubleshooting skills, and to get insight into methods used by other troubleshooters.

Lab Review Questions and Answers

Lab: Troubleshooting Common Issues

Question and Answers

Lab Review

Question: What tools might you use to monitor an intermittent or long-term issue?

Answer: SQL Server Data Collector, Extended Events, and SQL Trace are examples of tools you might use for long-term monitoring.

Module 15

Importing and Exporting Data

Contents:

Lesson 1

Transferring Data to and from SQL Server

Contents:

Question and Answers

Question: What is the effect of disabling the clustered index on a row store table?

( ) The index is ignored, but the table can be updated.

( ) The table becomes read-only.

( ) The table is completely inaccessible.

( ) The table is deleted.

Answer:

( ) The index is ignored, but the table can be updated.

( ) The table becomes read-only.

(√) The table is completely inaccessible.

( ) The table is deleted.

Resources

Disabling and Enabling Constraints

Best Practice: In general, you should not disable primary key or unique constraints during bulk load operations without very good reason. Both constraint types are critical to the integrity of your data; it is likely to be easier to prevent invalid data from being loaded—by leaving primary key and unique constraints in place—than it would be to correct it after a bulk load.

Demonstration: Disabling and Enabling Constraints

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Start SQL Server Management Studio and connect to your Azure instance running the AdventureWorksLT database, using SQL Server authentication.
3. Open the **Demo.ssmssln** solution in the D:\Demofiles\Mod15\Demo folder. If the Solution Explorer pane is not visible, on the **View** menu, click **Solution Explorer**.
4. Open the query file **Demo 01 - constraints.sql**.
5. Connect the query window to your copy of the **AdventureWorksLT** database.
6. Execute the code under the heading for **Step 2** to create two tables for this demonstration.
7. Execute the code under the heading for **Step 3** to show the current state of the check constraint.
8. Execute the code under the heading for **Step 4** to disable the check constraint.
9. Execute the code under the heading for **Step 5** to show that the check constraint is marked as disabled and untrusted.
10. Execute the code under the heading for **Step 6** to enable the check constraint with CHECK.
11. Execute the code under the heading for **Step 7** to show that the constraint is enabled and marked untrusted.

12. Execute the code under the heading for **Step 8** to enable the check constraint WITH CHECK CHECK.
13. Execute the code under the heading for **Step 9** to show that the constraint is enabled and trusted.
14. Execute the code under the heading for **Step 10** to disable a non-clustered primary key.
15. Execute the code under the heading for **Step 11** to show the state of the indexes on the table.
16. Execute the code under the heading for **Step 12** to demonstrate that data can still be inserted into the table.
17. Execute the code under the heading for **Step 13** to enable the index.
18. Execute the code under the heading for **Step 14** to show the state of the indexes on the table.
19. Execute the code under the heading for **Step 15** to disable a clustered primary key constraint. Note the warning messages generated by this command.
20. Execute the code under the heading for **Step 16** to show that all the indexes on the table are disabled.
21. Execute the code under the heading for **Step 17** to enable the clustered index.
22. Execute the code under the heading for **Step 18** to show that the non-clustered index remains disabled.
23. Execute the code under the heading for **Step 19** to enable the non-clustered index.
24. Execute the code under the heading for **Step 20** to enable the foreign key constraint which references the clustered primary key.
25. Execute the code under the heading for **Step 21** to drop the demonstration objects.
26. Leave SSMS open for the next demonstration.

Demonstration: Switching Partitions for Data Transfer

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Open the **Demo.ssmssln** solution in the D:\Demofiles\Mod15\Demo folder. If the Solution Explorer pane is not visible, on the **View** menu, click **Solution Explorer**.
3. Open the query file **Demo 02 - partition switch.sql**.
4. Connect the query window to your copy of the **AdventureWorksLT** database.
5. Execute the code under the heading for **Step 2** to create a partition function, partition scheme and partitioned table.
6. Execute the code under the heading for **Step 3** to create and add data to unpartitioned table which matches the schema of the partitioned table.
7. Execute the code under the heading for **Step 4** to switch partition one of the partitioned table with the unpartitioned table.
8. Execute the code under the heading for **Step 5** to demonstrate the effect of the switch.
9. Execute the code under the heading for **Step 6** to create three identical unpartitioned tables, and add data to **SalesLT.ShippingRate**.

10. Execute the code under the heading for **Step 7** to add data to **SalesLT.ShippingRateStaging**, representing a data load.
11. Execute the code under the heading for **Step 8** to switch partitions. Note the use of the third table so that one of the participants in a switch is always empty.
12. Execute the code under the heading for **Step 9** to demonstrate the effect of the switch.
13. Execute the code under the heading for **Step 10** to drop the demonstration objects.
14. Leave SSMS open for the next demonstration.

Lesson 2

Importing and Exporting Table Data

Contents:

Question and Answers

Question: True or false? The Data Import and Export Wizard can only be used to import and export data to Microsoft formats (such as Excel, Access, and SQL Server).

( ) True

( ) False

Answer:

( ) True

(√) False

Demonstration: Working with SSIS

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Run **Setup.cmd** in the D:\Demofiles\Mod15 folder as Administrator.
3. Open the **SSISProject.sln** solution in the D:\Demofiles\Mod15\SSISProject folder. If you see a dialog box asking how you wish to open the file, click **Visual Studio Version Selector**.
4. Give a brief demonstration of the different areas of an SSIS project in Visual Studio.
5. In Solution Explorer, double-click **Package.dtsx**.
6. On the **SSIS** menu, click **SSIS Toolbox**.
7. Click and drag **Data Flow Task** from the SSIS Toolbox pane to the Package.dtsx [Design] pane. Release the **Data Flow Task** anywhere within the **Control Flow** tab of the Package.dtsx [Design] pane.
8. Right-click the **Data Flow Task** on the **Control Flow** tab of the Package.dtsx [Design] pane. Click **Rename**, then type **Top Level Domain Name Import** and press Enter.
9. Double-click **Top Level Domain Name Import** to go to the **Data Flow** tab of the designer. Click and drag **Source Assistant** from the SSIS Toolbox pane to the **Data Flow** tab of the Package.dtsx [Design] pane.
10. In the Source Assistant – Add New Source window, select **Flat File** in the **Select source type** box, then double-click **New...** in the **Select connection managers** box.
11. In the Flat File Connection Manager Editor window, on the General page, in the **Connection Manager Name** box type **TLD File**. In the **File name** box type **D:\Demofiles\Mod15\Data\top_level_domains.txt**. in the **Header rows to skip** box, type **1**. Click to clear the **Column names in the first data row** box.
12. On the Columns page, examine the preview to ensure that two columns are shown.
13. On the Advanced page, change the **OutputColumnWidth** for **Column 0** to **100**. Click **OK**.
14. Right-click **Flat File Source** in the **Data Flow** tab and click **Rename**. Type **TLD File Source**, then press Enter.

15. Click and drag **Destination Assistant** from the SSIS Toolbox pane to the **Data Flow** tab. In the Destination Assistant – Add New Destination window, confirm that **Select destination type** is **SQL Server**, then in the **Select Connection managers** box, click **New…**, then click **OK**.
16. In the Connection Manager window, type **MIA-SQL** in the **Server name** box. In the **Select or enter a database name** box, select **salesapp1**, then click **Test Connection**. Click **OK** in the test result dialog, then click **OK**.
17. Right-click **OLE DB Destination** in the **Data Flow** tab and click **Rename**. Type **salesapp1 DB**, then press Enter.
18. Click **TLD File Source**, then click the leftmost (blue) arrow on the bottom of the **TLD File Source** object, then click the **salesapp1 DB** object.
19. Double-click the **salesapp1 DB** object. In the **OLE DB Destination Editor** box, on the Connection Manager page, in the **Name of the table or the view** box, click **[dbo].[TopLevelDomain]**. Point out the **Table Lock** and **Check constraints** check boxes and relate them back to the previous lesson.
20. On the Mappings page, in the first **Input Column** box, click **Column 0**. In the second **Input Column** box, click **Column 1**. Click **OK**.
21. On the **Debug** menu click **Start Debugging**. When the package has completed, on the **Debug** menu click **Stop Debugging**.
22. In SQL Server Management Studio, open the query file **Demo 03 - SSIS.sql**.
23. On the toolbar, click **Change Connection**, In the **Connect to Database Engine** dialog box, in **Server name** box, type **MIA-SQL**, and in the Authentication box select **Windows** Authentication, then click **Connect**.
24. Execute the query in the file to view the uploaded contents of the **dbo.TopLevelDomain** table.
25. Close Visual Studio without saving changes. Leave SSMS open for the next demonstration.

Demonstration: Using the SQL Server Import and Export Wizard

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In SQL Server Management Studio, in Object Explorer, click **connect**, and then click **Database Engine**.
3. In the Connect to Database Engine dialog box, in **Server name** box, type **MIA-SQL**, and then click **Connect**.
4. In SSMS Object Explorer, expand the **Databases** node under **MIA-SQL**. Right-click **salesapp1**, then point to **Tasks**, then click **Export Data**.
5. In the SQL Server Import and Export Wizard window, click **Next**. In the **Data Source** box, click **SQL Server Native Client 11.0**. Verify that the **Database** box has the value **salesapp1**, then click **Next**.
6. On the Choose a Destination page, in the **Destination** box, click **Flat File Destination**. Type **D:\Demofiles\Mod15\export.txt** in the **File name** box, then click **Next**. Verify that **Copy data from one or more tables or views** is selected, then click **Next**.
7. Select **[Production].[Categories]** in the **Source table or view** box. Click **Next**.

8. Verify that **Run immediately** is selected, then click **Finish**. Click **Finish** on the Complete the Wizard page to run the export. When the export completes, click **Close**.
9. Open **D:\Demofiles\Mod15\export.txt** to verify the result of the export.
10. Leave SSMS open for the next demonstration.

Lesson 3

Using bcp and BULK INSERT to Import Data

Contents:

Question and Answers

Question: True or false? By default, **bcp** and BULK INSERT ignore check constraints, foreign key constraints, and triggers when importing data.

( ) True

( ) False

Answer:

(√) True

( ) False

Demonstration: Working with bcp

Demonstration Steps

1. Open a command prompt, type the following command, and then press Enter to view the **bcp** syntax help:

```
bcp -?
```

2. At the command prompt, type the following command, and then press Enter to create a text format file:

```
bcp salesapp1.HR.Employees format nul -S MIA-SQL -T -w -t ^| -r \n -f
D:\Demofiles\Mod15\bcp\EmployeesFmt.txt
```

3. At the command prompt, type the following command, and then press Enter to create an XML format file:

```
bcp salesapp1.HR.Employees format nul -S MIA-SQL -T -w -t ^| -r \n -x -f
D:\Demofiles\Mod15\bcp\EmployeesFmt.xml
```

4. Open **D:\Demofiles\Mod15\bcp\EmployeesFmt.txt** and **D:\Demofiles\Mod15\bcp\EmployeesFmt.xml** using Notepad. Review and compare the contents of the files, then close Notepad.

5. At the command prompt, type the following command, and then press Enter to export data using the XML format file:

```
bcp salesapp1.HR.Employees out D:\Demofiles\Mod15\bcp\Employees.csv -S MIA-SQL -T -f
D:\Demofiles\Mod15\bcp\EmployeesFmt.xml
```

6. Close the command prompt.

7. Using Notepad, open the **D:\Demofiles\Mod15\bcp\Employees.csv** file and view the data that has been exported. Note that the commas in several of the data fields make this data unsuitable for export using a comma as a field delimiter. Close Notepad when you have finished reviewing.

Demonstration: Working with BULK INSERT

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Open the query file **Demo 06 - BULK INSERT.sql**. Execute the code under the heading for **Step 1** to demonstrate that the **Finance.dbo.Currency** table is empty.
3. Execute the code under the heading for **Step 2** to run a BULK INSERT statement to load **Finance.dbo.Currency** with data.
4. Execute the code under the heading for **Step 3** to verify that the table has been loaded with data.
5. Leave SSMS open for the next demonstration.

Demonstration: Working with OPENROWSET

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Open the query file **Demo 07 - OPENROWSET.sql**. Execute the code under the heading for **Step 1** to demonstrate that the **Finance.dbo.SalesTaxRate** table is empty.
3. Execute the code under the heading for **Step 2** to demonstrate a SELECT statement using the OPENROWSET BULK provider.
4. Execute the code under the heading for **Step 3** to demonstrate that the output of an OPENROWSET statement can be filtered with a WHERE clause.
5. Execute the code under the heading for **Step 4** to use an OPENROWSET statement to insert data into the **Finance.dbo.SalesTaxRate** table.
6. Execute the code under the heading for **Step 5** to demonstrate that the **Finance.dbo.SalesTaxRate** table now contains data.
7. Leave SSMS open for the next demonstration.

Lesson 4
Deploying and Upgrading Data-Tier Applications

Contents:

Question and Answers

Question: Which of the following is *not* an action you can carry out on a DACPAC?

( ) EXTRACT

( ) UPGRADE

( ) DEPLOY

( ) REGISTER

( ) EXPORT

Answer:

( ) EXTRACT

( ) UPGRADE

( ) DEPLOY

( ) REGISTER

(√) EXPORT

Resources

Performing In-Place Upgrades of Data-Tier Applications

Best Practice: You should take a full database backup before proceeding with an in-place upgrade of a DAC.

Demonstration: Working with Data-Tier Applications

Demonstration Steps

1. Ensure that the MSL-TMG1, 20764A-MIA-DC, and 20764A-MIA-SQL virtual machines are running, and then log on to 20764A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In SSMS Object Explorer, expand the **Databases** node under the **MIA-SQL** node. Right-click the node for the **Finance** database, then point to **Tasks**, then click **Extract Data-tier Application**.
3. In the **Extract Data-tier Application** window, click **Next**. In the **Save to DAC package file (include .dacpac extension with the file name)** box, type **D:\Demofiles\Mod15\dacpac\Finance.dacpac**. Click **Next**.
4. On the **Validation and Summary** page, click **Next**. The extract will begin. When the extraction process is complete, click **Finish**.
5. In File Explorer, navigate to **D:\Demofiles\Mod15\dacpac** to view the exported DACPAC file.
6. To import the DACPAC, in SSMS Object Explorer, right-click the **Databases** node under the **MIA-SQL** node, then click **Deploy Data-tier Application**.
7. In the **Deploy Data-tier Application** window, click **Next**.
8. On the **Select Package** page, in the **DAC package (file name with the .dacpac extension)** box type **D:\Demofiles\Mod15\dacpac\Finance.dacpac**, then click **Next**.

9. On the **Update Configuration** page, in the **Name (the name of the deployed DAC and database)** box type **FinanceDAC**, then click **Next**.
10. On the **Summary** page click **Next**. The deployment will run. When the deployment is complete, click **Finish**.
11. In Object Explorer, right-click the **Databases** node and then click Refresh. Verify that the **FinanceDAC** database exists. Expand the **FinanceDAC** node, then expand the **Tables** node. Right-click the **dbo.Currency** node, then click **Select Top 1000 Rows** to verify that the table has been created with no data.
12. Close SSMS.

Module Review and Takeaways

Best Practice

- Choose the right tool for bulk-imports.
- Use SSIS for complex transformations.
- Use **bcp** or BULK INSERT for fast imports and exports.
- Use OPENROWSET when data needs to be filtered before it is inserted.
- Try to achieve minimal logging to speed up data import.

Review Question(s)

Question: What other factors should you consider when importing or exporting data?

Answer: The answer will vary by circumstances.

Lab Review Questions and Answers

Lab: Importing and Exporting Data

Question and Answers

Lab Review

Question: What alternative methods to an SSIS package might you use to export the output of the **Sales.usp_prospect_list** stored procedure to a file?

> **Answer:** Several different options might be suitable alternatives.

Question: If the **HR.JobCandidate** table has included a column for a resume in Microsoft Word document format, which of the following commands could you use to import the document into a column in a table?

( ) The BULK provider in the OPENROWSET command with the SINGLE_BLOB option.

( ) The BULK provider in the OPENROWSET command with the SINGLE_CLOB option.

( ) The BULK provider in the OPENROWSET command with the SINGLE_NCLOB option.

( ) None of the above.

> **Answer:**
>
> (√) The BULK provider in the OPENROWSET command with the SINGLE_BLOB option.
>
> ( ) The BULK provider in the OPENROWSET command with the SINGLE_CLOB option.
>
> ( ) The BULK provider in the OPENROWSET command with the SINGLE_NCLOB option.
>
> ( ) None of the above.