Lab Answer Key: Module 10: Auditing Data Access and Encrypting Data

Lab: Auditing Data Access and Encrypting Data

Exercise 1: Implementing Auditing

Task 1: Prepare the Lab Environment

1. Ensure that the 20462C-MIA-DC and 20462C-MIA-SQL virtual machines are both running, and then log on to 20462C-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In the D:\Labfiles\Lab10\Starter folder, right-click **Setup.cmd** and then click **Run as administrator**.
3. Click **Yes** when prompted to confirm that you want to run the command file, and then wait for the script to finish.

Task 2: Create an Audit

1. Start SQL Server Management Studio, and connect to the **MIA-SQL** database engine instance using Windows authentication.
2. In SQL Server Management Studio, open the **Audit.sql** script file in the D:\Labfiles\Lab10\Solution folder.
3. Select the code under the comment **Create an audit**, and click **Execute**. This creates an audit that logs events to files in D:\Labfiles\Lab10\Starter\Audits.
4. In Object Explorer, expand **Security**, and expand **Audits** (if **Audits** is not expandable, refresh it and try again).
5. Double-click the **AW_Audit** audit you created and view its properties. Then click **Cancel**.

Task 3: Create a Server Audit Specification

1. In SQL Server Management Studio, in the **Audit.sql** script, select the code under the comment **Create a server audit specification** and click **Execute**. This creates an audit specification for the **AW_Audit** audit that logs failed and successful login attempts.
2. In Object Explorer, refresh the **Server Audit Specifications** folder and expand it. Then double-click **AW_ServerAuditSpec,** view its properties, and click **Cancel**.

Task 4: Create a Database Audit Specification

1. In SQL Server Management Studio, in the **Audit.sql** script, select the code under the comment **Create a database audit specification** and click **Execute**. This creates an audit specification for the **AW_Audit** audit that logs user-defined audit events in the **InternetSales** database as well as specific actions by the **customers_reader** database role and the **sales_admin** application role in the **Customers** schema.
2. In Object Explorer, expand **Databases**, expand **InternetSales**, expand **Security**, and expand **Database Audit Specifications**.
3. Double-click **AW_DatabaseAuditSpec** and view its properties. Then click **Cancel**.

Task 5: Implement a Custom Action

1. In the **Audit.sql** script, select the code under the comment **Create a trigger for a user-defined action** and click **Execute**. This creates a trigger that writes a custom audit event when the **EmailAddress** column in the **Customers.Customer** table is updated.

2. Select the code under the comment **Grant permission on sp_audit_write** and click **Execute**. This grants EXECUTE permission on the **sp_audit_write** stored procedure to the **public** role in the **master** database.

Task 6: View Audited Events

1. Open a command prompt and enter the following command to run sqlcmd as **ADVENTUREWORKS\VictoriaGray**. This user is a member of the **ADVENTUREWORKS\Sales_Europe** global group, which in turn is a member of the **ADVENTUREWORKS\InternetSales_Users** domain local group.

```
runas /user:adventureworks\victoriagray /noprofile sqlcmd
```

2. When prompted for a password, enter **Pa$$w0rd**.

3. In the SQLCMD window, enter the following commands to query the **Customers.Customer** table:

```
SELECT FirstName, LastName, EmailAddress FROM
Customers.Customer;
GO
```

4. In the SQLCMD window, enter the following commands to activate the **sales_admin** application role and update the **Customers.Customer** table:

```
EXEC sp_setapprole 'sales_admin', 'Pa$$w0rd';
UPDATE Customers.Customer SET EmailAddress =
'user1@contoso.com' WHERE CustomerID = 1699;
GO
```

5. Close the SQLCMD and command prompt windows.

6. In the D:\Labfiles\Lab10\Starter\Audits folder, verify that an audit file has been created.

7. In SQL Server Management Studio, in the **Audit.sql** script, select the code under the comment **View audited events** and click **Execute**. This queries the files in the audit folder and displays the audited events (events logged for the Student user and the service account for SQL Server have been excluded to simplify the results).

8. Note that all events are logged with the server principal name **ADVENTUREWORKS\VictoriaGray** despite the fact that this user accesses SQL Server through membership of a Windows group and does not have an individual login. This identity is audited even when executing statements in the security context of an application role.

9. Keep SQL Server Management Studio open for the next exercise.

Result: After this exercise, you should have created an audit, a server audit specification, and a database specification.

Exercise 2: Implementing Transparent Database Encryption

Task 1: Create Encryption Keys

1. In SQL Server Management Studio, open the **TDE.sql** script file in the D:\Labfiles \Lab10\Solution folder.

2. Select the code under the comment **Create DMK** and click **Execute**. This creates a database master key in the **master** database.

3. Select the code under the comment **Create server certificate** and click **Execute**. This

creates a certificate.

4. Select the code under the comment **Back up the certificate** and click **Execute**. This backs up the certificate and its private key.

5. Select the code under the comment **Create DEK** and click **Execute**. This creates a database encryption key in the **HumanResources** database

Task 2: Enable Database Encryption

1. In the TDE.sql script, select the code under the comment **Enable encryption** and click **Execute**. This enables encryption for the **HumanResources** database.

2. Select the code under the comment **Verify encryption status** and click **Execute.** This retrieves database encryption status from the **sys.databases** table in the **master** database.

3. Review the query results, and verify that the **is_encypted** value for **HumanResources** is **1**.

Task 3: Move the Database

1. In Object Explorer, in the **Databases** folder, right-click **HumanResources**, point to **Tasks**, and click **Detach**. Then, in the **Detach Database** dialog box, click **OK.**

2. In Object Explorer, in the **Connect** drop-down list, click **Database Engine**. Then connect to **MIA-SQL\SQL2** using Windows authentication.

3. In Object Explorer, under the **MIA-SQL\SQL2** instance, right-click **Databases** and click **Attach**.

4. In the **Attach Database** dialog box, click **Add**, select the **HumanResources.mdf** file in the M:\Data folder, and click **OK**. Then click **OK** on the error message that is displayed because the certificate with which the database encryption key is protected does not exist on the **MIA-SQL\SQL2** instance.

5. Open the **MoveDB.sql** script file in the D:\Labfiles\Lab10\Solution folder.

6. Select the code under the comment **Create a database master key** and click **Execute**. This creates a database **master** key in the master database on **MIA-SQL\SQL2**.

7. Select the code under the comment **Create certificate from backup** and click **Execute**. This creates a certificate in the **master** database on **MIA-SQL\SQL2** from the backup files you created previously.

8. Select the code under the comment **Attach database** and click **Execute**. This attaches the **HumanResources** database on **MIA-SQL\SQL2**.

9. Select the code under the comment **Test database** and click **Execute**. This queries the **Employees.Employee** table in the **HumanResources** database.

10. Review the query results. Then close SQL Server Management Studio without saving any files.

Result: After completing this exercise, you should have configured TDE and moved the encrypted HumanResource database to another instance of SQL Server.