Lab Manual, T2765, Installing and provisioning SQL Server

Contents

Lab 2: Installing SQL Server

Scenario

You have been tasked with creating a new instance of SQL Server that will be used by the IT department as a test server for new applications.

Objectives

After completing this lab, you will be able to:

- Assess resources available for a SQL Server installation.
- Install SQL Server.
- Perform post-installation checks.
- Automate a SQL Server installation.

Virtual machine: **20765C-MIA-SQL**

User name: **ADVENTUREWORKS\Student**

Password: **Pa55w.rd**

Exercise 1: Preparing to Install SQL Server

Scenario

You are preparing to install SQL Server for the IT department in Adventure Works Cycles. Before installing, you want to find out if the server hardware provisioned for the instance is ready.

The main tasks for this exercise are as follows:

1. Prepare the Lab Environment

2. View Hardware and Software Requirements

3. Run the System Configuration Checker

▶ Task 1: Prepare the Lab Environment

1. Ensure that the MT17B-WS2016-NAT, 20765C-MIA-DC, and 20765C-MIA-SQL virtual machines are running, and then log on to 20765C-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa55w.rd**.

2. Run **Setup.cmd** in the **D:\Labfiles\Lab02\Starter** folder as Administrator.

▶ Task 2: View Hardware and Software Requirements

1. Run **setup.exe** in the **X:** folder to run the SQL Server installation program.

2. In the SQL Server Installation Center, on the **Planning** page, view the **Hardware and Software Requirements**.

▶ Task 3: Run the System Configuration Checker

1. In the SQL Server Installation Center, on the **Tools** page, use the **System Configuration Checker** to assess the computer's readiness for a SQL Server installation.

2. Keep the SQL Server Installation Center window open. You will use it again in a later exercise.

Results: After this exercise, you should have run the SQL Server setup program and used the tools in the SQL Server Installation Center to assess the computer's readiness for SQL Server installation.

Exercise 2: Installing SQL Server

Scenario

The required configuration details for the new SQL Server instance you must install are described in the following table:

Item	Configuration value
Instance name	SQLTEST
Features	Database Engine only (excluding Replication, Full-Text, and DQS)
User database directory	M:\SQLTEST\Data
User database log directory	L:\SQLTEST\Logs
Service accounts	ADVENTUREWORKS\ServiceAcct / Pa55w.rd for all services
Startup	Both SQL Server and SQL Server Agent should start manually
Server collation	SQL_Latin1_General_CP1_CI_AS
Authentication mode	Mixed
SA Password	Pa55w.rd
Administrative user	ADVENTUREWORKS\Student
Filestream support	Disabled

The main tasks for this exercise are as follows:

1. Install the SQL Server Instance

▶ Task 1: Install the SQL Server Instance

- Install the required instance of SQL Server:
  - On the **Product Key** page, select **Evaluation** edition, which does not require a product key.
  - On the **Feature Selection** page, select only the features that are required.
  - On the **Server Configuration** page, configure the service account name and password, the startup type for the SQL Server Agent and SQL Server Database Engine services, and verify the collation.
  - On the **Database Engine Configuration** page, configure the authentication mode and the SA password; add the current user (Student) to the SQL Server administrators list, specify the required data directories, and verify that Filestream is not enabled.

Results: After this exercise, you should have installed an instance of SQL Server.

Exercise 3: Perform Post-Installation Checks

Scenario

In this exercise, you will start the SQL Server service for the new instance and connect to it using SSMS to make sure that the instance works.

The main tasks for this exercise are as follows:

1. Start the SQL Server Service
2. Configure Network Protocols and Aliases
3. Verify Connectivity to SQL Server

▶ Task 1: Start the SQL Server Service

1. Start SQL Server 2017 Configuration Manager, and view the properties of the **SQL Server (SQLTEST)** service.
2. Verify that the service is configured to log on as **ADVENTUREWORKS\ServiceAcct**, and then start the service.

▶ Task 2: Configure Network Protocols and Aliases

1. In SQL Server Configuration Manager, view the SQL Server network protocols configuration for the **SQLTEST** instance and verify that the **TCP/IP** protocol is enabled.
2. View the SQL Server Native Client 32-bit client protocols and verify that the **TCP/IP** protocol is enabled. Create an alias named **Test** that uses TCP/IP to connect to the **MIA-SQL\SQLTEST** instance from 32-bit clients.
3. View the SQL Server Native Client protocols and verify that the **TCP/IP** protocol is enabled. Create an alias named **Test** that uses TCP/IP to connect to the **MIA-SQL\SQLTEST** instance from 64-bit clients.

▶ Task 3: Verify Connectivity to SQL Server

1. Use sqlcmd to connect to the MIA-SQL\SQLTEST instance of SQL Server using a trusted connection, and run the following command to verify the instance name:

```
SELECT @@ServerName;
GO
```

2. Use SQL Server Management Studio to connect to the **Test** alias.
3. In SQL Server Management Studio, in Object Explorer:
   a. View the properties of the **Test** instance and verify that the value of the **Name** property is **MIA-SQL\SQLTEST**.
   b. Stop the **Test** service.
4. Close SQL Server Management Studio.

Results: After this exercise, you should have started the SQL Server service and connected using SSMS.

Exercise 4: Automating Installation

Scenario

The Adventure Works development team want to install SQL Server database engine instances on several development servers.

The configuration of these instances should match the configuration of the SQLTEST instance you have just installed, with the following differences:

- The instance name should be SQLDEV.
- TCP/IP should be disabled.
- User database files and log files should all be placed in C:\devdb.
- The development servers have two CPU cores; you should configure **tempdb** accordingly.
- You must create a configuration file to standardize the installation process.

The main tasks for this exercise are as follows:

1. Review an Unattended Installation File
2. Update an Unattended Installation File

▶ Task 1: Review an Unattended Installation File

1. Open **D:\Labfiles\Lab02\Starter\ConfigurationFile.ini** using Notepad.
2. Review the content of the file, paying particular attention to the following properties:
    a. INSTANCEID
    b. INSTANCENAME
    c. ACTION
    d. FEATURES
    e. TCPENABLED
    f. SQLUSERDBDIR
    g. SQLUSERDBLOGDIR
    h. SQLTEMPDBFILECOUNT
3. Leave the file open in Notepad.

▶ Task 2: Update an Unattended Installation File

1. Return to the **ConfigurationFile.ini** file open in Notepad:
    a. Amend the file to reflect the changes needed for this task.
    b. The instance name should be SQLDEV.
    c. TCP/IP should be disabled.
    d. User database files and log files should all be placed in C:\devdb.
    e. The development servers have two CPU cores; **tempdb** should be configured accordingly.
2. Save the file and compare your changes to the solution shown in the file D:\Labfiles\Lab02\Solution\SolutionConfigurationFile.ini.

Results: After this exercise, you will have reviewed and edited an unattended installation configuration file.

Lab Answer Key 2: Installing SQL Server

Exercise 1: Preparing to Install SQL Server

▶ Task 1: Prepare the Lab Environment

1. Ensure that the MT17B-WS2016-NAT, 20765C-MIA-DC, and 20765C-MIA-SQL virtual machines are running, and then log on to 20765C-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa55w.rd**.
2. In the **D:\Labfiles\Lab02\Starter** 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 finish.

▶ Task 2: View Hardware and Software Requirements

1. In the **X:** folder, double-click **setup.exe**.
2. In the **User Account Control** dialog box, click **Yes**.
3. In the SQL Server Installation Center, on the **Planning** page, click **Hardware and Software Requirements**.
4. In Internet Explorer, note that the documentation provides detailed information about hardware and software requirements for SQL Server. Close Internet Explorer.

▶ Task 3: Run the System Configuration Checker

1. In the SQL Server Installation Center, on the **Tools** page, click **System Configuration Checker**, and wait for the tool to start.
2. When the tool has run, review the checks that were performed. (If the checks are not visible, click **Show details**.)
3. Click **OK** to close SQL Server Setup.
4. Keep the SQL Server Installation Center window open. You will use it again in a later exercise.

Results: After this exercise, you should have run the SQL Server setup program and used the tools in the SQL Server Installation Center to assess the computer's readiness for SQL Server installation.

Exercise 2: Installing SQL Server

▶ Task 1: Install the SQL Server Instance

1. In the SQL Server Installation Center window, on the **Installation** page, click **New SQL Server stand-alone installation or add features to an existing installation** and wait for SQL Server setup to start.
2. If the **Microsoft Updates** or **Product Updates** pages are displayed, clear any check boxes and click **Next**.

On the **Install Rules** page, note that the list of rules has been checked. If the list of checks is notshown, click **Show Details**. If a warning about Windows Firewall is displayed, you can continue.

3. On the **Install Rules** page, click **Next**.
4. On the **Installation Type** page, ensure that **Perform a new installation of SQL Server** is selected, and then click **Next**.
5. On the **Product Key** page, in the **Specify a free edition** box, select **Evaluation**, and then click **Next**.
6. On the **License Terms** page, note the Microsoft Software License Terms, select **I accept the license terms**, and then click **Next**.
7. On the **Feature Selection** page, select **Database Engine Services**, and then click **Next**.
8. On the **Instance Configuration** page, ensure that **Named instance** is selected, in the **Named instance** box, type **SQLTEST**, and then click **Next**.
9. On the **Server Configuration** page, on the **SQL Server Agent** and **SQL Server Database Engine** rows, enter the following values:
    - **Account Name**: ADVENTUREWORKS\ServiceAcct
    - **Password**: Pa55w.rd
    - **Startup Type**: Manual
10. On the **Collation** tab, ensure that **SQL_Latin1_General_CP1_CI_AS** is selected and click **Next**.
11. On the **Database Engine Configuration** page, on the **Server Configuration** tab, in the **Authentication Mode** section, select **Mixed Mode (SQL Server authentication and Windows authentication)**. Enter and confirm the password, **Pa55w.rd**.
12. Click **Add Current User**; this will add the user **ADVENTUREWORKS\Student (Student)** to the list of Administrators.
13. On the **Data Directories** tab, change the **User database directory** to **M:\SQLTEST\Data**.
14. Change the **User database log directory** to **L:\SQLTEST\Logs**.
15. On the **TempDB** tab, review the default values that have been selected for **tempdb** data files.
16. On the **FILESTREAM** tab, ensure that **Enable FILESTREAM for Transact-SQL access** is not selected, and then click **Next**.
17. On the **Ready to Install** page, review the summary, then click **Install** and wait for the installation to complete.
18. On the **Complete** page, click **Close**.
19. Close the SQL Server Installation Center window.

Results: After this exercise, you should have installed an instance of SQL Server.

Exercise 3: Perform Post-Installation Checks

▶ Task 1: Start the SQL Server Service

1. On the Start screen, type **SQL Server 2017 Configuration Manager**, and then click **SQL Server 2017 Configuration Manager**.
2. In the **User Account Control** dialog box, click **Yes**.
3. In the left-hand pane of the SQL Server Configuration Manager window, click **SQL Server Services**.
4. In the right-hand pane, double-click **SQL Server (SQLTEST)**.
5. In the **SQL Server (SQLTEST) Properties** dialog box, verify that the service is configured to log on as **ADVENTUREWORKS\ServiceAcct**, and then click **Start**. When the service has started, click **OK**.

▶ Task 2: Configure Network Protocols and Aliases

1. In SQL Server Configuration Manager, expand **SQL Server Network Configuration**, click **Protocols for SQLTEST**, and verify that the **TCP/IP** protocol is **Enabled** for this instance of SQL Server.
2. In SQL Server Configuration Manager, expand **SQL Native Client 11.0 Configuration (32bit)**, click **Client Protocols**, and verify that the **TCP/IP** protocol is **Enabled** for 32-bit client applications.
3. Click **Aliases**, and note that there are currently no aliases defined for 32-bit clients.
4. Right-click **Aliases** and click **New Alias**.
5. In the Alias - New window, in the **Alias Name** text box, type **Test**.
6. In the **Protocol** drop-down list box, click **TCP/IP**.
7. In the **Server** text box, type **MIA-SQL\SQLTEST** and click **OK**.
8. In SQL Server Configuration Manager, expand **SQL Native Client 11.0 Configuration**, click **Client Protocols**, and verify that the **TCP/IP** protocol is enabled for 64-bit client applications.
9. Click **Aliases**, and note that there are currently no aliases defined for 64-bit clients.
10. Right-click **Aliases** and click **New Alias**.
11. In the Alias - New window, in the **Alias Name** text box, type **Test**.
12. In the **Protocol** drop-down list box, click **TCP/IP**.
13. In the **Server** text box, type **MIA-SQL\SQLTEST** and click **OK**.
14. Close SQL Server Configuration Manager.

▶ Task 3: Verify Connectivity to SQL Server

1. Click the **Start** button, type **CMD** and press ENTER.
2. At the command prompt, enter the following command to connect to the MIA-SQL\SQLTEST instance of SQL Server:

```
sqlcmd -S MIA-SQL\SQLTEST -E
```

3. At the sqlcmd prompt, enter the following command to display the SQL Server instance name, and then press ENTER:

```
SELECT @@ServerName;
GO
```

4. Close the command prompt window.

Start SQL Server Management Studio, and when prompted, connect to the database engine named **Test** using Windows Authentication.

5. In Object Explorer, right-click **Test**, and then click **Properties**.
6. Verify that the value of the **Name** property is **MIA-SQL\SQLTEST** and click **Cancel**.
7. In Object Explorer, right-click **Test** and click **Stop**.
8. In the **User Account Control** dialog box, click **Yes**.
9. When prompted to confirm that you want to stop the MSSQL$SQLTEST service, click **Yes**.
10. When the service has stopped, close SQL Server Management Studio.

Results: After this exercise, you should have started the SQL Server service and connected using SSMS.

Exercise 4: Automating Installation

▶ Task 1: Review an Unattended Installation File

1. On the taskbar, click the **File Explorer** shortcut.
2. In File Explorer, browse to **D:\Labfiles\Lab02\Starter**.
3. Double-click **ConfigurationFile.ini**. The file will open in Notepad.
4. Review the content in conjunction with the *Install SQL Server From the Command Prompt* topic in the SQL Server online documentation. In particular, note the values of the following properties:
   a. INSTANCEID
   b. INSTANCENAME
   c. ACTION
   d. FEATURES
   e. TCPENABLED
   f. SQLUSERDBDIR
   g. SQLUSERDBLOGDIR
   h. SQLTEMPDBFILECOUNT
5. Leave the file open in Notepad.

▶ Task 2: Update an Unattended Installation File

1. Return to the **ConfigurationFile.ini** file open in Notepad.
2. Locate the **INSTANCENAME** parameter in the file. Edit so that its value is **SQLDEV**. The line should look like this:

```
INSTANCENAME="SQLDEV"
```

3. Locate the **INSTANCEID** parameter in the file. Edit so that its value is **SQLDEV**. The line should look like this:

```
INSTANCEID="SQLDEV"
```

Locate the **TCPENABLED** parameter in the file. Edit so that its value is **0**. The line should look like this:

```
TCPENABLED="0"
```

4. Locate the **SQLUSERDBDIR** parameter in the file. Edit so that its value is **C:\devdb**. The line should look like this:

```
SQLUSERDBDIR="C:\devdb"
```

5. Locate the **SQLUSERDBLOGDIR** parameter in the file. Edit so that its value is **C:\devdb**. The line should look like this:

```
SQLUSERDBLOGDIR="C:\devdb"
```

6. Locate the **SQLTEMPDBFILECOUNT** parameter in the file. Edit so that its value is **2**. The line should look like this:

```
SQLTEMPDBFILECOUNT="2"
```

7. Save the file and then close Notepad.

Results: After this exercise, you will have reviewed and edited an unattended installation configuration file.

Lab 3: Upgrading SQL Server

Scenario

You are a database administrator for Adventure Works. As part of an upgrade of company databases from SQL Server 2014 to SQL Server 2017, you need to complete a two-server side-by-side upgrade of a database called TSQL. The upgrade is using backup and restore to move the databases.

You have been given a full database backup and a transaction log backup taken from the SQL Server 2014 instance. You must restore the database to upgrade it to SQL Server 2017, and create any missing logins.

Objectives

After completing this lab, you will be able to:

- Upgrade a database to SQL Server 2017 by using backup and restore.
- Create a database login with SSMS and via the CREATE LOGIN command.
- Repair an orphaned database user.
- Change database compatibility level.

Virtual machine: **20765C-MIA-SQL**

User name: **ADVENTUREWORKS\Student**

Password: **Pa55w.rd**

Exercise 1: Create the Application Logins

Scenario

The TSQL database has two database users:

- appuser1, linked to a login appuser.
- reportuser1, linked to a login reportuser.

You should create the logins before the database is restored.

The SID and password hash for reportuser are available.

The main tasks for this exercise are as follows:

1. Prepare the Lab Environment

2. Create the appuser Login

3. Create the reportuser Login Using CREATE USER

▶ Task 1: Prepare the Lab Environment

1. Ensure that the MT17B-WS2016-NAT, 20765C-MIA-DC, and 20765C-MIA-SQL virtual machines are both running, and then log on to 20765C-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa55w.rd**.

2. Run **Setup.cmd** in the **D:\Labfiles\Lab03\Starter** folder as Administrator.

Task 2: Create the appuser Login

1. Open SSMS and connect to the **MIA-SQL** database engine.
2. Using the SSMS UI, create a login with the following credentials:
   - Login: **appuser**
   - Password: **Pa55w.rd1**

▶ Task 3: Create the reportuser Login Using CREATE USER

1. Open the project file **D:\Labfiles\Lab03\Starter\Project\Project.ssmssln** and the Transact-SQL **Lab Exercise 01 - create login.sql**. Ensure that you are connected to the master database.
2. Write a CREATE LOGIN statement to create a login with the name **reportuser** and the SID and password hash values specified in the Transact-SQL script.

Results: After this exercise, you should be able to create a login using SSMS and the CREATE USER command.

Exercise 2: Restore the Backups of the TSQL Database

Scenario

As the logins have been created, you can restore the database backups. You have been given a full backup of the TSQL database and a transaction log backup—both of these must be restored. When the restore is complete, you should update the database statistics.

The main tasks for this exercise are as follows:

1. Restore the Full Backup
2. Restore the Transaction Log Backup
3. Update Statistics

▶ Task 1: Restore the Full Backup and a Transaction Log Backup

- Restore the backup **D:\Labfiles\Lab03\Starter\TSQL1.bak** and **D:\Labfiles\Lab03\Starter\TSQL1_trn1.trn** to the **MIA-SQL** instance.

▶ Task 2: Update Statistics

- Update the database statistics for the database **TSQL** with the **sp_updatestats** system stored procedure.

Results: At the end of this exercise, you should be able to:

Prepare for a migration by creating application logins.

Restore a database backup taken from one SQL Server instance and restore it to another.

Detect and repair orphaned users.

Exercise 3: Orphaned Users and Database Compatibility Level

Scenario

Now that the database has been restored, you should check for orphaned users. After this is complete, you have been asked to raise the database to the SQL Server 2017 database compatibility level.

The main tasks for this exercise are as follows:

1. Detect Orphaned Users
2. Repair Orphaned Users
3. Change Database Compatibility Level

▶ Task 1: Detect Orphaned Users

- Run a check for orphaned users in the newly restored TSQL database. Hint: use **sp_change_users_login** with the @Action parameter set to 'Report'.

  The report should return one result - appuser1.

▶ Task 2: Repair Orphaned Users

- Repair the orphaned user you found in the first task by linking it to the appuser login.

▶ Task 3: Change Database Compatibility Level

1. Raise the database compatibility level to SQL Server 2017 (compatibility level = 140).
2. Close SQL Server Management Studio, without saving any changes.

Results: After this lab, you should be able to:

Identify orphaned database users.

Repair orphaned database users.

Update the database compatibility level.

Lab Answer Key 3: Upgrading SQL Server

Exercise 1: Create the Application Logins

▶ Task 1: Prepare the Lab Environment

1. Ensure that the MT17B-WS2016-NAT, 20765C-MIA-DC, and 20765C-MIA-SQL virtual machines are both running, and then log on to 20765C-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa55w.rd**.
2. In the **D:\Labfiles\Lab03\Starter** folder, right-click **Setup.cmd**, and then click **Run as administrator**.
3. In the **User Account Control** dialog box, click **Yes**, wait for the script to finish, and then press any key.

▶ Task 2: Create the appuser Login

1. Start SQL Server Management Studio and connect to the **MIA-SQL** database engine using Windows authentication.
2. In Object Explorer, under the **MIA-SQL** server node, expand **Security**.
3. Right-click **Logins**, then click **New Login**.
4. In the Login - New window, in the **Login name** box, type **appuser**.
5. Click **SQL Server authentication**, then in the **Password** and **Confirm password** boxes, type **Pa55w.rd1**.
6. Clear **User must change password at next login**, then click **OK**.

▶ Task 3: Create the reportuser Login Using CREATE USER

1. In SQL Server Management Studio, on the **File** menu, point to **Open**, and click **Project/Solution**.
2. In the **Open Project** dialog box, browse to **D:\Labfiles\Lab03\Starter\Project**, and then double-click **Project.ssmssln**.
3. In the **View** menu, click **Solution Explorer**.
4. In Solution Explorer, expand **Queries**, and then double-click the query **Lab Exercise 01 - create login.sql**.
5. In the query window, highlight the statement **USE master**, and click **Execute**.
6. In the query pane, after the **Task 2** description, type the following query:

```
CREATE LOGIN [reportuser]
WITH PASSWORD=<password_hash_value> HASHED,
          SID=<sid_value> ;
```

7. From the task description, copy and paste the password hash value (starting **0x02…**) over **<password_hash_value>**.
8. From the task description, copy and paste the SID value (starting **0x44…**) over **<sid_value>**.
9. Highlight the query and click **Execute**.

Results: After this exercise, you should be able to create a login using SSMS and the CREATE USER command.

Exercise 2: Restore the Backups of the TSQL Database

▶ Task 1: Restore the Full Backup

1. In SQL Server Management Studio, in Object Explorer, right-click **Databases**, and then click **Restore Database**.
2. On the **General** page, in the **Source** section, click **Device**, and then click the ellipsis (**...**) button.
3. In the **Select backup devices** dialog box, click **Add**.
4. In the **Locate Backup File - MIA-SQL** dialog box, browse to the **D:\Labfiles\Lab03\Starter** folder, click **TSQL1.bak**, and then click **OK**.
5. In the **Locate Backup File - MIA-SQL** dialog box, browse to the **D:\Labfiles\Lab03\Starter** folder, click **TSQL1_trn1.trn**, and then click **OK**.
6. You should now have added two backup files: TSQL1.bak and TSQL1_trn1.trn.
7. In the **Select backup devices** dialog box, click **OK**.
8. On the **Files** page, select **Relocate all files to folder**, in the **Data file folder** box, type **D:\Labfiles\Lab03**, and then in the **Log file folder** box, type **D:\Labfiles\Lab03**.

▶ Task 2: Update Statistics

1. In SQL Server Management Studio, in Solution Explorer, double-click the query **Lab Exercise 02 - restore database.sql**.
2. In the query window, after the **Task 3** heading, type the following:

```
EXEC TSQL.sys.sp_updatestats;
```

3. Highlight the text you have typed and click **Execute**.

Results: At the end of this exercise, you should be able to:

Prepare for a migration by creating application logins.

Restore a database backup taken from one SQL Server instance and restore it to another.

Detect and repair orphaned users.

Exercise 3: Orphaned Users and Database Compatibility Level

▶ Task 1: Detect Orphaned Users

1. In SQL Server Management Studio, in Solution Explorer, double-click the query **Lab Exercise 03 - orphaned users.sql**.
2. In the query window, highlight the statement **USE TSQL;** and then click **Execute**.
3. After the **Task 1** description, type the following:

```
EXEC sp_change_users_login @Action = 'Report'
```

4. Highlight the query and click **Execute**.

▶ Task 2: Repair Orphaned Users

1. In the query pane, after the **Task 2** description, type the following query:

```
EXEC sp_change_users_login @Action = 'Update_One', @UserNamePattern = 'appuser1',
@LoginName = 'appuser';
```

2. Highlight the query and click **Execute**.

▶ Task 3: Change Database Compatibility Level

1. In SQL Server Management Studio, in Object Explorer, expand the **Databases**, right-click **TSQL**, and click **Properties**.
2. In the Databases Properties - TSQL window, on the **Options** page, change the value of the **Compatibility level** box to **SQL Server 2017 (140)**, and then click **OK**.
3. Close SQL Server Management Studio, without saving any changes.

Results: After this lab, you should be able to:

Identify orphaned database users.

Repair orphaned database users.

Update the database compatibility level.

Lab 4: Managing Database Storage

Scenario

As a database administrator at Adventure Works Cycles, you are responsible for managing system and user databases on the MIA-SQL instance of SQL Server. There are several new applications that require databases, which you must create and configure.

Objectives

After completing this lab, you will be able to:

- Configure **tempdb**.
- Create databases.
- Attach a database.

Virtual machine: **20765C-MIA-SQL**

User name: **ADVENTUREWORKS\Student**

Password: **Pa55w.rd**

Exercise 1: Configuring tempdb Storage

Scenario

The application development team has notified you that some of the new applications will make extensive use of temporary objects. To support this requirement while minimizing I/O contention, you have decided to move the **tempdb** database files to a dedicated storage volume and increase the size of the data and log files.

The main tasks for this exercise are as follows:

1. Prepare the Lab Environment

2. Configure tempdb Files

▶ Task 1: Prepare the Lab Environment

1. Ensure that the MT17B-WS2016-NAT, 20765C-MIA-DC, and 20765C-MIA-SQL virtual machines are both running, and then log on to 20765C-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa55w.rd**.

2. Run **Setup.cmd** in the **D:\Labfiles\Lab04\Starter** folder as **Administrator**.

▶ Task 2: Configure tempdb Files

1. Use SQL Server Management Studio to view the properties of the **tempdb** system database on the **MIA-SQL** database engine instance, and note the current location and size of the tempdb database files.

2. Alter **tempdb** so that the database files match the below specification. If the T: drive doesn't exist, then use a drive and folder that ***does*** exist.
    - **Tempdev**:
        - **Initial Size**: 10 MB
        - **File growth**: 5 MB
        - **Maximum size**: Unlimited
        - **File Name**: T:\tempdb.mdf
    - **Templog**:
        - **Initial Size**: 5 MB
        - **File growth**: 1 MB
        - **Maximum size**: Unlimited
        - **File Name**: T:\templog.ldf
3. Restart the **SQL Server** service and verify that the changes have taken effect.

Results: After this exercise, you should have inspected and configured the **tempdb** database.

Exercise 2: Creating Databases

Scenario

The following two applications have been developed and both require databases:

- The Human Resources application is a simple solution for managing employee data. It is not expected to be used heavily or to grow substantially.
- The Internet Sales application is a new e-commerce website and must support a heavy workload that will capture a large volume of sales order data.

The main tasks for this exercise are as follows:

1. Create the HumanResources Database
2. Create the InternetSales Database
3. View Data File Information

▶ Task 1: Create the HumanResources Database

- Create a new database named **HumanResources** with the following database files: If the M: and L: drives doesn't exist, then create folder(s) under a drive that does exist and use that folder.

Logical Name	Filegroup	Initial Size	Growth	Path
HumanResources	PRIMARY	50 MB	5 MB / Unlimited	M:\Data\HumanResources.mdf
HumanResources_log		5 MB	1 MB / Unlimited	L:\Logs\HumanResources.ldf

Task 2: Create the InternetSales Database

- Create a new database named **InternetSales** with the following database files:

Logical Name	Filegroup	Initial Size	Growth	Path
InternetSales	PRIMARY	50 MB	1MB / Unlimited	M:\Data\InternetSales.mdf
InternetSales_data1	SalesData	100MB	10MB / Unlimited	N:\Data\InternetSales_data1.ndf
InternetSales_data2	SalesData	100MB	10MB / Unlimited	N:\Data\InternetSales_data2.ndf
InternetSales_log		2Mb	10% / Unlimited	L:\Logs\InternetSales.ldf

▶ Task 3: View Data File Information

1. In SQL Server Management Studio, open the **ViewFileInfo.sql** script file in the **D:\Labfiles\Lab04\Starter** folder.
2. Execute the code under the comment **View page usage** and note the **UsedPages** and **TotalPages** values for the **SalesData** filegroup.
3. Execute the code under the comments **Create a table on the SalesData** filegroup and **Insert 10,000 rows**.
4. Execute the code under the comment **View page usage again** and verify that the data in the table is spread across the files in the filegroup.

Results: After this exercise, you should have created a new **HumanResources** database and an **InternetSales** database that includes multiple filegroups.

Exercise 3: Attaching a Database

Scenario

Business analysts at Adventure Works Cycles have developed a data warehouse that must be hosted on **MIA-SQL**. The analysts have supplied you with the database files so that you can attach the database.

The database has multiple filegroups, including a filegroup for archive data, which should be configured as read-only.

The main tasks for this exercise are as follows:

1. Attach the AWDataWarehouse Database
2. Configure Filegroups

▶ Task 1: Attach the AWDataWarehouse Database

1. Using File Explorer, move **AWDataWarehouse.ldf** from the **D:\Labfiles\Lab04\Starter** folder to the **L:\Logs** folder, and then move the following files from the **D:\Labfiles\Lab04\Starter** folder to the **M:\Data** folder (or to a drive and folder that does exist):
   - AWDataWarehouse.mdf
   - AWDataWarehouse_archive.ndf
   - AWDataWarehouse_current.ndf
2. Attach the **AWDataWarehouse** database by selecting the **AWDataWarehouse.mdf** file and ensuring that the other files are found automatically.

▶ Task 2: Configure Filegroups

1. View the properties of the **AWDataWarehouse** database and note the filegroups it contains.
2. Set the **Archive** filegroup to read-only.
3. View the properties of the **dbo.FactInternetSales** table and verify that it is stored in the **Current** filegroup.
4. View the properties of the **dbo.FactInternetSalesArchive** table and verify that it is stored in the **Archive** filegroup.
5. Edit the **dbo.FactInternetSales** table and modify a record to verify that the table is updateable.
6. Edit the **dbo.FactInternetSalesArchive** table and attempt to modify a record to verify that the table is read-only.
7. Close SQL Server Management Studio without saving any changes.

Results: After this exercise, you should have attached the **AWDataWarehouse** database to MIA-SQL.

Lab Answer Key 4: Managing Database Storage

Exercise 1: Configuring tempdb Storage

▶ Task 1: Prepare the Lab Environment

1. Ensure that the MT17B-WS2016-NAT, 20765C-MIA-DC, and 20765C-MIA-SQL virtual machines are both running, and then log on to 20765C-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa55w.rd**.
2. Run **Setup.cmd** in the **D:\Labfiles\Lab04\Starter** folder as **Administrator**.
3. In the **User Account Control** dialog box, click **Yes**.
4. Wait for the script to complete.

▶ Task 2: Configure tempdb Files

1. Start SQL Server Management Studio and connect to the **MIA-SQL** database engine using Windows authentication.
2. In Object Explorer, expand **Databases**, expand **System Databases**, right-click **tempdb**, and click **Properties**.
3. On the **Files** page, view the current file settings, and then click **Cancel**.
4. On the toolbar, click **New Query**.
5. Enter the below statements and click **Execute**. Note that if the T: drive doesn't exist, then use a drive and folder that **does** exist.

```
USE master;
GO
ALTER DATABASE tempdb
MODIFY FILE (NAME = tempdev, SIZE = 10MB, FILEGROWTH = 5MB, FILENAME =
'T:\tempdb.mdf');
ALTER DATABASE tempdb
MODIFY FILE (NAME = templog, SIZE=5MB, FILEGROWTH = 1MB, FILENAME =
'T:\templog.ldf');
GO
```

6. In Object Explorer, right-click **MIA-SQL** and click **Restart**.
7. In the **User Account Control** dialog box, click **Yes**.
8. When prompted to allow changes, to restart the service, and to stop any dependent services, click **Yes**.
9. View the contents of **T:** (or whatever you specified above) and note that the **tempdb.mdf** and **templog.ldf** files have been moved tothis location.
10. In SQL Server Management Studio, in Object Explorer, right-click **tempdb**, and click **Properties**.
11. On the **Files** page, verify that the file settings have been modified, and then click **Cancel**.
12. Save the script file as **Configure TempDB.sql** in the **D:\Labfiles\Lab04\Starter** folder.
13. Keep SQL Server Management Studio open for the next exercise.

Results: After this exercise, you should have inspected and configured the **tempdb** database.

Exercise 2: Creating Databases

▶ Task 1: Create the HumanResources Database

1. In SQL Server Management Studio, click **New Query**.
2. Enter the following statements and click **Execute**: If the M: drive doesn't exist, then create a folder under a drive that does exist and use that folder.

```
CREATE DATABASE HumanResources
 ON  PRIMARY
(NAME = 'HumanResources', FILENAME = 'M:\Data\HumanResources.mdf', SIZE = 50MB,
FILEGROWTH = 5MB)
 LOG ON
(NAME = 'HumanResources_log', FILENAME = 'L:\Logs\HumanResources.ldf', SIZE = 5MB,
FILEGROWTH = 1MB);
GO
```

3. In Object Explorer, right-click the **Databases** folder, and then click **Refresh** to confirm that the **HumanResources** database has been created.
4. Save the script file as **Create HumanResources.sql** in the **D:\Labfiles\Lab04\Starter** folder.
5. Keep SQL Server Management Studio open for the next exercise.

▶ Task 2: Create the InternetSales Database

1. In SQL Server Management Studio, click **New Query**.
2. Enter the following statements and click **Execute**. If the M: drive doesn't exist, then create a folder under a drive that does exist and use that folder.

```
CREATE DATABASE InternetSales
 ON  PRIMARY
(NAME = 'InternetSales', FILENAME = 'M:\Data\InternetSales.mdf', SIZE = 5MB,
FILEGROWTH = 1MB),
 FILEGROUP SalesData
(NAME = 'InternetSales_data1', FILENAME = 'M:\Data\InternetSales_data1.ndf', SIZE =
100MB, FILEGROWTH = 10MB ),
(NAME = 'InternetSales_data2', FILENAME = 'N:\Data\InternetSales_data2.ndf', SIZE =
100MB, FILEGROWTH = 10MB )
 LOG ON
(NAME = 'InternetSales_log', FILENAME = 'L:\Logs\InternetSales.ldf', SIZE = 2MB,
FILEGROWTH = 10%);
GO
```

3. Under the existing code, enter the following statements, select the statements you have just added, and then click **Execute**:

```
ALTER DATABASE InternetSales
MODIFY FILEGROUP SalesData DEFAULT;
```

4. Save the script file as Create **InternetSales.sql** in the **D:\Labfiles\Lab04\Starter** folder.
5. Keep SQL Server Management Studio open for the next exercise.

▶ Task 3: View Data File Information

1. In SQL Server Management Studio, open the **ViewFileInfo.sql** script file in the **D:\Labfiles\Lab04\Starter** folder.
2. Select the code under the comment **View page usage** and click **Execute**. This query retrieves data about the files in the **InternetSales** database.
3. Note the **UsedPages** and **TotalPages** values for the **SalesData** filegroup.

Select the code under the comment **Create a table on the SalesData filegroup** and click **Execute**.

4. Select the code under the comment **Insert 10,000 rows** and click **Execute**.
5. Select the code under the comment **View page usage again** and click **Execute**.
6. Note the **UsedPages** value for the **SalesData** filegroup, and verify that the data in the table is spread across the files in the filegroup.
7. Keep SQL Server Management Studio open for the next exercise.

Results: After this exercise, you should have created a new **HumanResources** database and an **InternetSales** database that includes multiple filegroups.

Exercise 3: Attaching a Database

▶ Task 1: Attach the AWDataWarehouse Database

1. Using File Explorer, move **AWDataWarehouse.ldf** from the **D:\Labfiles\Lab04\Starter** folder to the **L:\Logs** folder.
2. Using File Explorer, move the following files from the **D:\Labfiles\Lab04\Starter** folder to the **M:\Data** folder:
   - AWDataWarehouse.mdf
   - AWDataWarehouse_archive.ndf
   - AWDataWarehouse_current.ndf
3. In SQL Server Management Studio, in Object Explorer, right-click **Databases** and click **Attach**.
4. In the **Attach Databases** dialog box, click **Add**.
5. In the **Locate Database Files - MIA-SQL** dialog box, in the **M:\Data** folder, select the **AWDataWarehouse.mdf** database file, and click **OK**.
6. In the **Attach Databases** dialog box, after you have added the **master** databases file, note that all of the database files are listed, and then click **OK**.
7. In Object Explorer, under **Databases**, verify that **AWDataWarehouse** is now listed.

▶ Task 2: Configure Filegroups

1. In Object Explorer, right-click the **AWDataWarehouse** database and click **Properties**.
2. On the **Filegroups** page, view the filegroups used by the database.
3. Select the **Read-Only** check box for the **Archive** filegroup and click **OK**.
4. In Object Explorer, expand **AWDataWarehouse**, expand **Tables**, right-click the **dbo.FactInternetSales** table and click **Properties**.
5. On the **Storage** page, verify that the **dbo.FactInternetSales** table is stored in the **Current** filegroup. Then click **Cancel**.
6. Right-click the **dbo.FactInternetSalesArchive** table and click **Properties**.
7. On the **Storage** page, verify that the **dbo.FactInternetSalesArchive** table is stored in the **Archive** filegroup, and then click **Cancel**.

In Object Explorer, right-click the **dbo.FactInternetSales** table and click **Edit Top 200 Rows**.

8. In the results output, change the **SalesAmount** value for the first record to **2500**, press Enter to update the record, and then close the **dbo.FactInternetSales** results.
9. In Object Explorer, right-click the **dbo.FactInternetSalesArchive** table and click **Edit Top 200 Rows**.
10. In the results output, change the **SalesAmount** value for the first record to **3500** and press Enter to update the record.
11. View the error message that is displayed, click **OK**, and then press Esc to cancel the update and close the **dbo.FactInternetSalesArchive** results.
12. Close SQL Server Management Studio without saving any changes.

Results: After this exercise, you should have attached the **AWDataWarehouse** database to MIA-SQL.

Lab 5: Performing Ongoing Database Maintenance

Scenario

You are a database administrator at Adventure Works Cycles, with responsibility for databases on the **MIA-SQL** Server instance. You must perform the ongoing maintenance of the database on this instance—this includes ensuring database integrity and managing index fragmentation.

Objectives

After completing this lab, you will be able to:

- Use DBCC CHECKDB.
- Defragment indexes.
- Create and run database maintenance plans.

Virtual machine: **20765C-MIA-SQL**

User name: **ADVENTUREWORKS\Student**

Password: **Pa55w.rd**

Exercise 1: Use DBCC CHECK to Verify Data Integrity

Scenario

Junior colleagues ask you to check a query they are writing before they commit the transaction. Before you can point out an error in the script, a failure occurs on the disk storing the log file and the server stops. After restarting the server, you will review the state of the data, replace the log with a copy from a mirror server, and run DBBC CHECKDB to repair the database.

The main tasks for this exercise are as follows:

1. Prepare the Lab Environment

2. Manage Database Integrity

▶ Task 1: Prepare the Lab Environment

1. Ensure that the 20765C-MIA-DC and 20765C-MIA-SQL virtual machines are both running, and then log on to 20765C-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa55w.rd**.
2. Run **Setup.cmd** in the **D:\Labfiles\Lab05\Starter** folder as Administrator.

▶ Task 2: Manage Database Integrity

1. In SQL Server Management Studio, connect to the **MIA-SQL** database engine using Windows authentication.
2. Open **Ex1.sln** from the **D:\Labfiles\Lab05\Starter\Ex1** folder and review the Transact-SQL code in the **Ex1-DBCC.sql** file.
3. Execute the code up to and including the CHECKPOINT statement.
4. In a new query window using a separate database connection, shut down the database server without performing checkpoints in every database.
5. In Windows Explorer, rename the **CorruptDB_log.ldf** file to simulate the file being lost.
6. Restart SQL Server and SQL Server Agent.

In SQL Server Management Studio, try to access the **CorruptDB** database, and then check its status.

7. Use emergency mode to review the data in the database and note that the erroneous transaction was committed on disk.
8. Set the database offline, rename the log file back to **CorruptDB_log.ldf** to simulate accessing it from a mirror copy, and then set the database online again.
9. Put the database in single user mode and run DBCC CHECKDB with the REPAIR_ALLOW_DATA_LOSS option.
10. Put the database back into multiuser mode and check that the data is restored to its original state.
11. Close the solution without saving changes, but leave SQL Server Management Studio open for the next exercise.

Results: After this exercise, you should have used DBBC CHECKDB to repair a corrupt database.

Exercise 2: Rebuild Indexes

Scenario

You have identified fragmentation in the **Sales.SalesOrderHeader** table in the **AdventureWorks** database and you are sure that performance is decreasing as the amount of fragmentation increases. You will rebuild the indexes for the table and confirm that the fragmentation level decreases.

The main tasks for this exercise are as follows:

1. Monitor and Maintain Indexes

▶ Task 1: Monitor and Maintain Indexes

1. Using SQL Server Management Studio, query the **sys.dm_db_index_physical_stats** function to retrieve information about the index fragmentation for indexes on the **Sales.SalesOrderHeader** table in the **AdventureWorks** database.
2. Note the **avg_page_space_used_in_percent** and **avg_fragmentation_in_percent** values for each index level.
3. Run the query in the **Ex2-InsertRows.sql** file in the **D:\Labfiles\Lab05\Starter\Ex2** folder to simulate OLTP activity.
4. Query the **sys.dm_db_index_physical_stats** function again, noting that the values for the same columns have changed due to the activity.
5. Rebuild all the indexes on the **Sales.SalesOrderHeader** table.
6. Query the **sys.dm_db_index_physical_stats** function again, confirming that the index fragmentation has decreased.
7. Close the solution without saving changes, but leave SQL Server Management Studio open for the next exercise.

Results: After this exercise, you should have rebuilt indexes on the **Sales.SalesOrderHeader** table, resulting in better performance.

Exercise 3: Create Database Maintenance Plans

Scenario

You want to ensure that there is an early detection of any integrity issues in the **AdventureWorks** database and that the database is regularly backed up to minimize recovery times. To do this, you will create database maintenance plans to schedule core operations on a weekly, daily and hourly basis.

The main tasks for this exercise are as follows:

1. Create a Database Backup Maintenance Plan
2. Create a Database Integrity Check Maintenance Plan
3. Run a Maintenance Plan

▶ Task 1: Create a Database Backup Maintenance Plan

- Create a database maintenance plan for the **AdventureWorks** database. The maintenance plan should perform the following operations:
  - A full database backup on a weekly basis.
  - A differential backup on a daily basis.
  - Transaction log backups on an hourly basis.

▶ Task 2: Create a Database Integrity Check Maintenance Plan

- Create a database maintenance plan for the **AdventureWorks** database. The maintenance plan should perform the following operation:
  - Check the integrity of the database using a short-term lock rather than an internal database snapshot.

▶ Task 3: Run a Maintenance Plan

1. Run the maintenance plan you created in the previous task.
2. View the history of the maintenance plan in the Log File Viewer in SQL Server Management Studio.
3. Close SQL Server Management Studio without saving changes.

Results: After this exercise, you should have created the required database maintenance plans.

Lab Answer Key 5: Performing Ongoing DatabaseMaintenance

Exercise 1: Use DBCC CHECK to Verify Data Integrity

▶ Task 1: Prepare the Lab Environment

1. Ensure that the 20765C-MIA-DC and 20765C-MIA-SQL virtual machines are both running, and then log on to 20765C-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa55w.rd**.
2. In the **D:\Labfiles\Lab05\Starter** folder, right-click **Setup.cmd**, and then click **Run as administrator**.
3. In the **User Account Control** dialog box, click **Yes** to confirm that you want to run the command file, and wait for the script to finish.

▶ Task 2: Manage Database Integrity

1. Start SQL Server Management Studio, and connect to the **MIA-SQL** database engine using Windows authentication.
2. On the **File** menu, point to **Open**, and then click **Project/Solution**.
3. In the **Open Project** dialog box, navigate to the **D:\Labfiles\Lab05\Starter\Ex1** folder, click **Ex1.ssmssln**, and then click **Open**.
4. In Solution Explorer, expand **Queries**, and double-click **Ex1-DBCC.sql**.
5. In the query pane, review the code under the comment **-- Update the Discontinued column for the Chai product**, select the code, and then click **Execute**.
6. In the query pane, review the code under the comment **-- Commit the transaction**, select the code, and then click **Execute**.
7. In the query pane, review the code under the comment **-- Simulate an automatic checkpoint**, select the code, and then click **Execute**.
8. In Solution Explorer, double-click **Ex1-Shutdown.sql**.
9. In the query pane, review the code under the comment **-- Open up a different connection and run this**, select the code, and then click **Execute**.
10. Close the **Ex1-Shutdown.sql** query pane without saving changes.
11. In Windows Explorer, navigate to the **D:\Labfiles\Lab05\Starter\Data** folder, right-click **CorruptDB_log**, and click **Rename**.
12. Type **CorruptDBlog_renamed** and press ENTER. In the **File Access Denied** message box, click **Continue**. This simulates losing the transaction log file in a disk failure.
13. Click **Start**, type **SQL Server Configuration**, and then click **SQL Server 2017 Configuration Manager**.
14. In the **User Account Control** dialog box, click **Yes**.
15. In SQL Server Configuration Manager, under **SQL Server Configuration Manager (Local)**, click **SQL Server Services**.
16. In the right-hand pane, right-click **SQL Server (MSSQLSERVER)**, and then click **Start**.

Right-click **SQL Server Agent (MSSQLSERVER)**, and then click **Start**.

17. Close SQL Server Configuration Manager.
18. In SQL Server Management Studio, in the **Ex1-DBCC.sql** query pane, review the code under the comment **-- Try to access the CorruptDB database**, select the code, and then click **Execute**. Note that the query causes an error.
19. Repeat Step 19, and note that the query causes an error.
20. In the query pane, review the code under the comment **-- Check the status of the database**, select the code, and then click **Execute**.
21. In the query pane, review the code under the comment **-- Confirm that the database is not online**, select the code, and then click **Execute**.
22. In Object Explorer, right-click **MIA-SQL**, click **Refresh**, and then expand **Databases**. Note that **CorruptDB** shows as **Recovery Pending**.
23. In the query pane, review the code under the comment **-- Use Emergency mode to review the data in the database**, select the code, and then click **Execute**.
24. In the query pane, review the code under the comment **-- Review the state of the discontinued products data**, select the code, and then click **Execute**. Note that the query shows zero products in stock, because the erroneous transaction was committed on disk and the transaction log file has been lost.
25. In the query pane, review the code under the comment **-- Set CorruptDB offline**, select the code, and then click **Execute**.
26. In Windows Explorer, navigate to the **D:\Labfiles\Lab05\Starter\Data** folder, right-click **CorruptDB_log_renamed**, and click **Rename**.
27. Type **CorruptDB_log** and press ENTER. In the **File Access Denied** message box, click **Continue**. This simulates replacing the lost transaction log file with a mirror copy.
28. In SQL Server Management Studio, in the query pane, review the code under the comment **-- After replacing the transaction log file, set CorruptDB back online**, select the code, and then click **Execute**.
29. In the query pane, review the code under the comment **-- Set the database in single user mode and use DBCC CHECKDB to repair the database**, select the code, and then click **Execute**.
30. In the query pane, review the code under the comment **-- Switch the database back into multi-user mode**, select the code, and then click **Execute**.
31. In the query pane, review the code under the comment **-- Check the data has returned to the pre-failure state**, select the code, and then click **Execute**. Note that the **Discontinued** column now shows that 77 products are in stock—the state that the data was in before the erroneous transaction was run.
32. Close the solution without saving changes, but leave SQL Server Management Studio open for the next exercise.

Results: After this exercise, you should have used DBBC CHECKDB to repair a corrupt database.

Exercise 2: Rebuild Indexes

▶ Task 1: Monitor and Maintain Indexes

1. In SQL Server Management Studio, on the **File** menu, point to **Open**, and then click **Project/Solution**.
2. In the **Open Project** dialog box, navigate to the **D:\Labfiles\Lab05\Starter\Ex2** folder, click **Ex2.ssmssln**, and then click **Open**.
3. In Solution Explorer, expand **Queries**, and then double-click **Ex2-Indexes.sql**.
4. In the query pane, review the code under the comment **-- View the statistics for index fragmentation on the Sales tables in the AdventureWorks database**, select the code, and then click **Execute**.
5. In the query pane, review the code under the comment **-- View the statistics for index fragmentation on the SalesOrderHeader table**, select the code, and then click **Execute**.
6. Note the results in the **Fragmentation Percent** column.
7. In the query pane, review the code under the comment **-- Insert an additional 10000 rows**, select the code, and then click **Execute**. Wait for the query to complete.
8. In the query pane, review the code under the comment **-- View the statistics for index fragmentation following the data insertion**, select the code, and then click **Execute**.
9. Note the results in the **Fragmentation Percent** column.
10. In the query pane, review the code under the comment **-- Rebuild the indexes**, select the code, and then click **Execute**.
11. In the query pane, review the code under the comment **-- View the statistics for index fragmentation following the index rebuild**, select the code, and then click **Execute**.
12. Note the results in the **Fragmentation Percent** column.
13. Close the solution without saving changes, but leave SQL Server Management Studio open for the next exercise.

Results: After this exercise, you should have rebuilt indexes on the **Sales.SalesOrderHeader** table, resulting in better performance.

Exercise 3: Create Database Maintenance Plans

▶ Task 1: Create a Database Backup Maintenance Plan

1. In SQL Server Management Studio, in Object Explorer, expand **Management**, right-click **Maintenance Plans**, and then click **Maintenance Plan Wizard**.
2. In the SQL Server Maintenance Plan Wizard, click **Next**.
3. On the **Select Plan Properties** page, in the **Name** box, type **Maintenance Plan for Backup of AdventureWorks Database**, select **Separate schedules for each task**, and then click **Next**.
4. On the **Select Maintenance Tasks** page, select the following tasks, and then click **Next**:
   - Back Up Database (Full)
   - Back Up Database (Differential)
   - Back Up Database (Transaction Log)
5. On the **Select Maintenance Task Order** page, click **Next**.
6. On the **Define Back Up Database (Full) Task** page, in the **Databases(s)** list, click **AdventureWorks**, and then click **OK** to close the drop-down list box.
7. Review the options on the **Destination** and **Options** tabs to see further changes possible.
8. On the **General** tab, in the **Schedule** section, click **Change**. Review the default schedule, and then click **OK**.
9. On the **Define Back Up Database (Full) Task** page, click **Next**.
10. On the **Define Back Up Database (Differential) Task** page, in the **Database(s)** list, select **AdventureWorks**, and then click **OK** to close the drop-down list box.
11. In the **Schedule** section, click **Change**.
12. In the **New Job Schedule** dialog box, in the **Frequency** section, in the **Occurs** drop-down list box, click **Daily**, and then click **OK**.
13. On the **Define Back Up Database (Differential) Task** page, click **Next**.
14. On the **Define Back Up Database (Transaction Log) Task** page, in the **Database(s)** list, select **AdventureWorks**, and then click **OK** to close the drop-down list box.
15. In the **Schedule** section, click **Change**.
16. In the **New Job Schedule** dialog box, in the **Frequency** section, in the **Occurs** drop-down list box, click **Daily**, in the **Daily frequency** section, select **Occurs every**, and then click **OK**.
17. On the **Define Back Up Database (Transaction Log) Task** page, click **Next**.
18. On the **Select Report Options** page, accept the default options, and then click **Next**.
19. On the **Complete the Wizard** page, click **Finish**. Wait for the operation to complete, and then click **Close**.

▶ Task 2: Create a Database Integrity Check Maintenance Plan

1. In Object Explorer, right-click **Maintenance Plans**, and then click **Maintenance Plan Wizard**.
2. In the **SQL Server Maintenance Plan Wizard**, click **Next**.
3. On the **Select Plan Properties** page, in the **Name** box, type **Maintenance Plan for Checking Integrity of the AdventureWorks Database**, and then click **Next**.
4. On the **Select Maintenance Tasks** page, select **Check Database Integrity** and click **Next**.

On the **Select Maintenance Task Order** page, click **Next**.

5. On the **Define Database Check Integrity Task** page, in the **Database(s)** list, click **AdventureWorks**, and then click **OK** to close the drop-down list box.
6. On the **Define Database Check Integrity Task** page, select the **Tablock** check box to minimize resource usage and maximize performance of the operations, and then click **Next**.
7. On the **Select Report Options** page, click **Next**.
8. On the **Complete the Wizard** page, click **Finish** to create the Maintenance Plan. Wait for the operation to complete, and then click **Close**.

▶ Task 3: Run a Maintenance Plan

1. In Object Explorer, expand **Maintenance Plans**, right-click **Maintenance Plan for Checking Integrity of the AdventureWorks Database**, and then click **Execute**.
2. In the **Execute Maintenance Plan** dialog box, wait until the maintenance plan succeeds, and then click **Close**.
3. Right-click **Maintenance Plan for Checking Integrity of the AdventureWorks Database**, and then click **View History**.
4. In the **Log File Viewer - MIA-SQL** dialog box, expand the **Date** value for the **Daily Maintenance** plan to see the individual task.
5. Review the data in the **Log file summary** section, and then click **Close**.
6. Close SQL Server Management Studio without saving changes.

Results: After this exercise, you should have created the required database maintenance plans.