Module 1: Introduction to Databases

Lab: Exploring and Querying SQL Server Databases

Exercise 1: Exploring an OLTP Schema and a Data Warehouse Schema

Task 1: Explore an OLTP Database Schema

1. On the taskbar, click **Microsoft SQL Server Management Studio**.
2. In the **Connect to Server** dialog box, in the **Server type** box, click **Database Engine**.
3. In the **Server name** box, type **MIA-SQL**.
4. In the **Authentication** box, click **Windows Authentication**, and then click **Connect**.
5. In Object Explorer, expand **Databases**, expand **AdventureWorks2016**, right-click **Database Diagrams**, and then click **New Database Diagram**.
6. If the **Microsoft SQL Server Management Studio** dialog box appears asking if you wish to create support objects for database diagramming, click **Yes**.
7. In the **Add Table** dialog box, press and hold down the CTRL key, click **Address (Person)**, click **Customer (Sales)**, click **SalesOrderDetail (Sales)**, click **SalesOrderHeader (Sales)**, click **SalesPerson (Sales)**, click **SalesTerritory (Sales)**, click **ShipMethod (Purchasing)**, click **Add**, and then click **Close**.
8. Review the tables and note the following points:
    1. The **SalesOrderHeader (Sales)** table contains the **SalesOrderID**, which is the primary key column.
    2. The **SalesOrderDetail (Sales)** table also contains a **SalesOrderID** column.
9. In the **SalesOrderDetail (Sales)** table, right-click the **SalesOrderID** column, and then click **Properties**.
10. In the **Properties** window, click the **Description** field, and then click the ellipsis button (…).
11. In the **Description Property** dialog box, note that the column is a primary key column, and that there is a foreign key that references the **SalesOrderID** column in the **SalesOrderHeader** column. Click **Cancel**.
12. Click the line between the **Customer (Sales)** table and the **SalesOrderHeader (Sales)** table. This line represents a foreign key relationship.
13. In the **Properties** window, click **Description**, and then click the ellipsis button (…).
14. In the **Description Property** dialog box, note that the foreign key references the **CustomerID** column in the **Customer (Sales)** table. Click **Cancel**.

15. On the **File** menu, click **Save Diagram_0**, in the **Choose Name** dialog box, type **AdventureWorks Diagram**, and then click **OK**.
16. Close the database diagram window, leaving SQL Server Management Studio open for the next task.

Task 2: Explore a Data Warehouse Schema

1. In Object Explorer, expand **Databases**, expand **AdventureWorksDW2016**, right-click **Database Diagrams**, and then click **New Database Diagram**.
2. If a dialog box appears asking if you wish to create support objects for database diagramming, click **Yes**.
3. In the **Add Table** dialog box, press and hold down the CTRL key, click **DimCustomer**, **DimDate**, **DimProduct**, **DimProductCategory**, **DimProductSubcategory**, **FactInternetSales**, click **Add**, and then click **Close**.
4. Note that the **FactInternetSales** table has foreign key relationships with the **DimCustomer**, **DimProduct**, and **DimDate** tables. Examine these relationships in the same way as you did in the previous task, noting the columns involved in the relationships.
5. On the **File** menu, click **Save Diagram_0**, in the **Choose Name** dialog box, type **AdventureWorks Data Warehouse Diagram**, and then click **OK**.
6. Close the database diagram window, leaving SQL Server Management Studio open for the next exercise.

Exercise 2: Querying a Database by Using Transact-SQL

Task 1: Write Transact-SQL SELECT Statements

1. In SQL Server Management Studio, in Object Explorer, right-click **AdventureWorks2016**, and then click **New Query**.
2. In the query window, type the following Transact-SQL statement, and then click **Execute**:

```
SELECT *
FROM Sales.SalesOrderHeader;
GO
```

3. Review the results in the results pane, and in the lower right corner, note the number of rows that the query returned.
4. In the query window, under the existing Transact-SQL statement, type the following Transact-SQL statement:

```
SELECT SalesOrderID, OrderDate, SalesPersonID
FROM Sales.SalesOrderHeader;
GO
```

5. Select the statement that you just typed, and then click **Execute**.
6. Review the results in the results pane, and in the lower right corner, verify that the number of rows returned is the same as for the previous query.

Task 2: Write Transact-SQL SELECT Statements with a WHERE Clause

1. In the query window, under the existing Transact-SQL statements, type the following Transact-SQL statement:

```
SELECT SalesOrderID, OrderDate, SalesPersonID
FROM Sales.SalesOrderHeader
WHERE SalesPersonID = 279;
GO
```

2. Select the statement that you just typed, and then click **Execute**.
3. Review the results in the results pane, and in the lower right corner, note the number of rows that the query returned.
4. In the query window, under the existing Transact-SQL statements, type the following Transact-SQL statement:

```
SELECT SalesOrderID, OrderDate, SalesPersonID
FROM Sales.SalesOrderHeader
WHERE SalesPersonID = 279 OR SalesPersonID = 282;
GO
```

5. Select the statement that you just typed, and then click **Execute**.
6. Review the results in the results pane, and in the lower right corner, verify that the number of rows is greater than that for the previous query.
7. In the query window, under the existing Transact-SQL statements, type the following Transact-SQL statement:

```
SELECT SalesOrderID, OrderDate
FROM Sales.SalesOrderHeader
WHERE SalesOrderID BETWEEN 57000 AND 58000;
GO
```

8. Select the statement that you just typed, and then click **Execute**.
9. Review the results in the results pane, and in the lower right corner, note the number of rows that the query returned.
10. In the query window, under the existing Transact-SQL statements, type the following Transact-SQL statement:

```
SELECT SalesOrderID, OrderDate
FROM Sales.SalesOrderHeader
WHERE SalesPersonID = 279 AND Year(OrderDate) = 2014;
GO
```

11. Select the statement that you just typed, and then click **Execute**.

12. Review the results in the results pane, and in the lower right corner, note the number of rows that the query returned.
13. Close SQL Server Management without saving changes.

Lab Review

Question Why did the number of rows returned by the queries that you wrote in the lab vary?

Answer The number of rows varied because the WHERE clause in the queries acts as a filter that limits the rows that the query returns. The WHERE clause in each statement uses different conditions, therefore different numbers of rows are returned.