Microsoft

OFFICIAL MICROSOFT LEARNING PRODUCT

29761A

Querying Data with Transact-SQL

Product Number: 29761A

Part Number (if applicable):

Released: 04/2016

Module 1

Introduction to Microsoft SQL Server 2016

Contents:

Lesson 1

The Basic Architecture of SQL Server

Contents:

Question and Answers

Put the following T-SQL commands in order by numbering each to create a script that will execute without errors:

	Steps
	CREATE TABLE HR.Employees (EmployeeID int PRIMARY KEY, LastName nvarchar(25), FirstName nvarchar(25));
	GO
	INSERT INTO HR.Employees (EmployeeID, LastName, FirstName) VALUES (121, N'O''Neill, N'Carlene');
	GO

Answer:

	Steps
1	CREATE TABLE HR.Employees (EmployeeID int PRIMARY KEY, LastName nvarchar(25), FirstName nvarchar(25));
2	GO
3	INSERT INTO HR.Employees (EmployeeID, LastName, FirstName) VALUES (121, N'O''Neill, N'Carlene');
2	GO

Lesson 2

SQL Server Editions and Versions

Contents:

Question and Answers

SQL Server Versions

Question: Which version of SQL Server are you currently working with? Have you worked with any earlier versions?

Answer: Answers will vary.

SQL Server Editions

Question: You have founded a new company with two friends. Your new application (app) uses a SQL Server database to store information. You are unsure whether your app will be successful but if it is, you will need both high performance and space for large volumes of data. However, you have not yet launched, so are unsure how many people will use your app. Which edition of SQL Server 2016 should you use for this system?

( ) Azure SQL Database

( ) Enterprise Edition

( ) Express Edition

( ) Business Intelligence Edition

( ) Any edition is appropriate for these requirements

Answer: Azure SQL Database allows you to start small, and scale up as required. As a startup, using Azure SQL Database also means you do not need to buy server hardware to run SQL Server. Because Azure SQL Database is Software as a Service (SaaS), you pay for what you use, without high upfront costs.

Lesson 3

Getting Started with SQL Server Management Studio

Contents:

Question and Answers

Connecting to SQL Server

Question: In your organization, which authentication method do you use to log on to SQL Server?

Answer: Either Windows Authentication or SQL Server Authentication.

Introducing Microsoft SQL Server 2016

Question: A colleague has asked you to run some test queries against the company's scheduling database. Administrators have provided you with the name of the server where the database is hosted, and the name of the database. Permissions to run the necessary queries have been granted to your Active Directory account. You are logged on to a client computer with this Active Directory account and have started SQL Server Management Studio. What other information do you need to connect to the database?

( ) Your Active Directory account username

( ) Your Active Directory account password

( ) The name of the login created for you in the SQL Server instance

( ) The name of the instance that hosts the database

( ) The name of the user created for you in the SQL Server database

Answer: The name of the instance that hosts the database.

You do not need to provide your Active Directory credentials because these will be sent to SQL Server automatically when you connect. This account has been associated with a login and user in SQL Server by database administrators when they granted access to your account. However, unless the database is on the default instance, you must specify which instance to connect to.

Answer: Module Review and Takeaways

Review Question(s)

Question: Can an SQL Server database be stored across multiple instances?

Answer: No. A database is completely contained within a single instance.

Question: If no T-SQL code is selected in a query window, which code lines will be run when you click the Execute button?

Answer: All statements in the script will be executed.

Question: What does an SQL Server Management Studio solution contain?

Answer: Projects.

Module 2

Introduction to T-SQL Querying

Contents:

Lesson 1

Introducing T-SQL

Contents:

Question and Answers

Question: From the following T-SQL elements, select the one that does not contain an expression:

( ) SELECT FirstName, LastName, SkillName AS Skill, GetDate() – DOB AS Age

( ) WHERE HumanResources.Department.ModifiedDate > (SYSDATETIME() – 31)

( ) JOIN HumanResources.Skills ON Employees.ID = Skills.EmployeeID

( ) WHERE Skill.Level + Skill.Confidence > 10

Answer: An expression is a combination of identifiers, symbols, and operators that return a single result. The only element that does not contain a symbol and an operator is Option 3.

Demonstration: T-SQL Language Elements

Demonstration Steps

Use T-SQL Language Elements

1. Ensure that the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines are both running, and then log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Open File Explorer, browse to **D:\Demofiles\Mod02**, right-click **Setup.cmd**, and then click **Run as administrator**.
3. In the **User Account Control** dialog box, click **Yes**.
4. When the script has finished, press any key.
5. Start SQL Server Management Studio and connect to the **MIA-SQL** database engine instance using Windows® Authentication.
6. On the **File** menu, point to **Open**, and then click **Project/Solution**.
7. In the **Open Project** dialog box, browse to the **D:\Demofiles\Mod02\Demo** folder, and then double-click **Demo.ssmssln**.
8. On the **View** menu, click **Solution Explorer**.
9. In Solution Explorer, expand **Queries**, and then double-click the **11 – Demonstration A.sql** script file.
10. Select the code under **Step 1**, and then click **Execute**.
11. Select the code under **Step 2**, and then click **Execute**.
12. Select the code under **Step 3**, and then click **Execute**.
13. Select the code under **Step 4**, and then click **Execute**.
14. Select the code under **Step 5**, and then click **Execute**.
15. Select the code under **Step 6**, and then click **Execute**.
16. Select the code under **Step 7**, and then click **Execute**.
17. Select the code under **Step 8**, and then click **Execute**.
18. Select the code under the comment **Cleanup task if needed**, and then click **Execute**.
19. Close SQL Server Management Studio.

Lesson 2

Understanding Sets

Contents:

Question and Answers

Place each employee into the appropriate set. Indicate your answer by writing the set number to the right of each item.

Items	
1	Carolos Lamy Works in: London Skills: JavaScript XML
2	Naiyana Kunakorn Works in: Washington DC Skills: JavaScript SQL Server Administration T-SQL XML
3	Zachary Parsons Works in: Seattle Skills: Active Directory Administration SharePoint Administration SQL Server Administration
4	Patrick Lorenzen Works in: London Skills: SharePoint Administration SQL Server Administration
5	Frederic Towle Works in: Tokyo Skills: Active Directory Administration T-SQL
6	Nickolas McLaughlin Works in: Seattle Skills: C# JavaScript SQL Server Administration
7	Jeanie Sheppard Works in: Buenos Aires Skills: C# JavaScript T-SQL

Category 1		Category 2		Category 3
Employees in London		Employees who know T-SQL		Employees in Seattle who know SQL Server Administration

Answer:

Category 1		Category 2		Category 3
Employees in London		Employees who know T-SQL		Employees in Seattle who know SQL Server Administration
Carolos Lamy Works in: London Skills: JavaScript XML Patrick Lorenzen Works in: London Skills: SharePoint Administration SQL Server Administration		Naiyana Kunakorn Works in: Washington DC Skills: JavaScript SQL Server Administration T-SQL XML Frederic Towle Works in: Tokyo Skills: Active Directory Administration T-SQL Jeanie Sheppard Works in: Buenos Aires Skills: C# JavaScript T-SQL		Zachary Parsons Works in: Seattle Skills: Active Directory Administration SharePoint Administration SQL Server Administration Nickolas McLaughlin Works in: Seattle Skills: C# JavaScript SQL Server Administration

Resources

Set Theory Applied to SQL Server Queries

Additional Reading: For more information on set theory and logical query processing, and its application to SQL queries, see Chapter 1 of Itzik Ben-Gan's *T-SQL Querying* (Microsoft Press, 2015).

Lesson 3

Understanding Predicate Logic

Contents:

Question and Answers

Question: From the following T-SQL elements, select the one that can include a predicate:

( ) WHERE clauses

( ) JOIN conditions

( ) HAVING clauses

( ) WHILE statements

( ) All of the above

Answer: WHERE clauses use predicates to determine which rows to return. JOIN conditions use predicates to determine which rows from different tables to join into a single result row. HAVING clauses use predicates to determine which groups to return. WHILE statements use predicates to determine when to stop executing T-SQL statements in a batch.

Lesson 4

Understanding the Logical Order of Operations in SELECT Statements

Contents:

Question and Answers

Put the following T-SQL elements in order by numbering each to indicate the order that SQL Server will process them in when they appear in a single SELECT statement.

	Steps
	FROM
	WHERE
	GROUP BY
	HAVING
	SELECT
	ORDER BY

Answer:

	Steps
1	FROM
2	WHERE
3	GROUP BY
4	HAVING
5	SELECT
6	ORDER BY

Demonstration: Logical Query Processing

Demonstration Steps

View Query Output That Illustrates Logical Processing Order

1. Ensure that the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines are both running, and then log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Open SQL Server Management Studio.
3. In the **Connect to Server** dialog, in the **Server name** field, enter the server you created during preparation. For example, **29761Aa-azure.database.windows.net**.
4. In the **Authentication** list, select **SQL Server Authentication**.
5. In the **Login** box, type **Student**.
6. In the **Password** box, type **Pa$$w0rd**, and then click **Connect**.
7. If the **New Firewall Rule** dialog box appears, click **Sign In**, enter your Azure credentials, and then click **Sign in**. In the **New Firewall Rule** dialog box, ensure **Add my client IP** is selected and then click **OK**.
8. If the **Microsoft SQL Server Management Studio** dialog box appears, click **OK**.
9. On the **File** menu, point to **Open**, and then click **Project/Solution**.

10. In the **Open Project** dialog box, browse to the **D:\Demofiles\Mod02\Demo** folder, and then double-click **Demo.ssmssln**.
11. On the **View** menu, click **Solution Explorer**.
12. In Solution Explorer, double-click the **21 – Demonstration B.sql** script file.
13. In the **Available Databases** list, click **AdventureWorksLT**. If AdventureWorksLT is not in the list, right-click in the query pane, point to **Connection**, and then click **Change Connection**. Repeat steps 3 to 6 to connect to the server you created during preparation. Repeat this step to connect to **AdventureWorksLT**.
14. Select the code under the comment **Step 1**, and then click **Execute**.
15. Select the code under **Step 2**, and then click **Execute**.
16. Select the code under **Step 3**, and then click **Execute**.
17. Select the code under **Step 4**, and then click **Execute**.
18. Select the code under **Step 5**, and then click **Execute**.
19. Select the code under **Step 6**, and then click **Execute**.
20. Select the code under **Step 7**, and then click **Execute**.
21. Select the code under **Step 8**, and then click **Execute**.
22. Close SQL Server Management Studio, without saving any changes.

Module Review and Takeaways

Question: Which category of T-SQL statements concerns querying and modifying data?

Answer: DML.

Question: What are some examples of aggregate functions supported by T-SQL?

Answer: SUM, MIN, COUNT, COUNTBIG, MAX, AVG.

Question: Which SELECT statement element will be processed before a WHERE clause?

Answer: FROM.

Module 3

Writing SELECT Queries

Contents:

Lesson 1

Writing Simple SELECT Statements

Contents:

Question and Answers

Question: You have a table named Sales with the following columns: Country, NumberOfReps, TotalSales.

You want to find out the average amount of sales a sales representative makes in each country. What SELECT query could you use?

Answer: SELECT Country, (TotalSales / NumberOfReps) AS AverageSalesPerRep

FROM Sales;

Demonstration: Writing Simple SELECT Statements

Demonstration Steps

Use Simple SELECT Queries

1. Ensure that the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines are both running, and then log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Run D:\Demofiles\Mod03\Setup.cmd as an administrator. In the **User Account Control** dialog box, click **Yes**.
3. At the command prompt, type **y**, and press Enter. When the script has completed, press any key.
4. Start SQL Server Management Studio and connect to the **Azure SQL** database engine instance using SQL Server authentication.
5. If the **Microsoft SQL Server Management Studio** dialog box appears, click **OK**.
6. Open the **Demo.ssmssln** solution in the D:\Demofiles\Mod03\Demo folder.
7. If the Solution Explorer pane is not visible, on the **View** menu, click **Solution Explorer**.
8. Expand **Queries**, and open the **Demonstration A.sql** script file. You may need to enter your password to connect to the **Azure SQL** database engine.
9. In the **Available Databases** list, click **AdventureWorksLT**.
10. Select the code under the comment **Step 2**, and then click **Execute**.
11. Select the code under the comment **Step 3**, and then click **Execute**.
12. Select the code under the comment **Step 4**, and then click **Execute**.
13. Select the code under the comment **Step 5**, and then click **Execute**.
14. Select the code under the comment **Step 6**, and then click **Execute**.
15. Select the code under the comment **Step 7**, and then click **Execute**.
16. Keep SQL Server Management Studio open for the next demonstration.

Lesson 2

Eliminating Duplicates with DISTINCT

Contents:

Question and Answers

Question: You have company departments in five countries. You have the following query for the Human Resources database:

```
SELECT DeptName, Country
FROM HumanResources.Departments
```

This returns:

```
DeptName  Country
--------- --------
Sales     UK
Sales     USA
Sales     France
Sales     Japan
Marketing USA
Marketing Japan
Research  USA
```

You add a DISTINCT keyword to the SELECT query. How many rows are returned?

Answer: 7

Demonstration: Eliminating Duplicates with DISTINCT

Demonstration Steps

Eliminate Duplicate Rows

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**, and run D:\Demofiles\Mod03\Setup.cmd as an administrator.
2. If SQL Server Management Studio is not already open, start it and connect to the **Azure SQL** database engine instance using SQL Server authentication, and then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod03\Demo folder.
3. In Solution Explorer, open the **Demonstration B.sql** script file. You may need to enter your password to connect to the **Azure SQL** database engine.
4. In the **Available Databases** list, click **AdventureWorksLT**.
5. Select the code under the comment **Step 2**, and then click **Execute**.
6. Select the code under the comment **Step 3**, and then click **Execute**.
7. Select the code under the comment **Step 4**, and then click **Execute**.
8. Keep SQL Server Management Studio open for the next demonstration.

Lesson 3

Using Column and Table Aliases

Contents:

Question and Answers

Question: You have the following query:

SELECT FirstName LastName

FROM HumanResources.Employees;

You are surprised to find that the query returns the following:

LastName

Fred

Rosalind

Anil

Linda

What error have you made in the SELECT query?

Answer: You have omitted a comma between FirstName and LastName.

Use Aliases to Refer to Columns

Question: Which of the following statements use correct column aliases?

SELECT Name AS ProductName FROM Production.Product

SELECT Name = ProductName FROM Production.Product

SELECT ProductName == Name FROM Production.Product

SELECT ProductName = Name FROM Production.Product

SELECT Name AS Product Name FROM Production.Product

Answer: Statements 1 and 4 are correct.

Demonstration: Using Column and Table Aliases

Demonstration Steps

Use Column and Table Aliases

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**, and run D:\Demofiles\Mod03\Setup.cmd as an administrator.
2. If SQL Server Management Studio is not already open, start it and connect to the **Azure SQL** database engine instance using SQL Server authentication, and then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod03\Demo folder.
3. In Solution Explorer, open the **Demonstration C.sql** script file. You may need to enter your password to connect to the **Azure SQL** database engine.
4. In the **Available Databases** list, click **AdventureWorksLT**.
5. Select the code under the comment **Step 2**, and then click **Execute**.
6. Select the code under the comment **Step 3**, and then click **Execute**.
7. Select the code under the comment **Step 4**, and then click **Execute**.

8. Select the code under the comment **Step 5**, and then click **Execute**.
9. Keep SQL Server Management Studio open for the next demonstration.

Lesson 4

Writing Simple CASE Expressions

Contents:

Question and Answers

Question: You have the following SELECT query:

```
SELECT FirstName, LastName, Sex
FROM HumanResources.Employees;
```

This returns:

```
FirstName  LastName  Sex
----------  ---------  ----
Maya       Steele     1
Adam       Brookes    0
Naomi      Sharp      1
Pedro      Fielder    0
Zachary    Parsons    0
```

How could you make these results clearer?

Answer: Use the following query:

```
SELECT FirstName, LastName, Gender =
  CASE Sex
    WHEN 1 THEN 'Female'
    WHEN 0 THEN 'Male'
    ELSE 'Unspecified'
  END
FROM HumanResources.Employees;
```

Demonstration: Simple CASE Expressions

Demonstration Steps

Use a Simple CASE Expression

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**, and run D:\Demofiles\Mod03\Setup.cmd as an administrator.
2. If SQL Server Management Studio is not already open, start it and connect to the **Azure SQL** database engine instance using SQL Server authentication, and then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod03\Demo folder.
3. In Solution Explorer, open the **Demonstration D.sql** script file. You may need to enter your password to connect to the **Azure SQL** database engine.
4. In the **Available Databases** list, click **AdventureWorksLT**.
5. Select the code under the comment **Step 2**, and then click **Execute**.
6. Select the code under the comment **Step 3**, and then click **Execute**.
7. Close SQL Server Management Studio, without saving changes.

Module Review and Takeaways

Best Practice

Terminate all T-SQL statements with a semicolon. This will make your code more readable, avoid certain parsing errors, and protect your code against changes in future versions of SQL Server.

Consider standardizing your code on the AS keyword for labeling column and table aliases. This will make it easier to read and avoids accidental aliases.

Review Question(s)

Question: Why is the use of SELECT * not a recommended practice?

Answer: There are two answers:

1) * asks for all columns, which is typically too much.

2) Query exposed to changes in underlying table structure and could return unexpected results.

Real-world Issues and Scenarios

You can create a column alias without using the AS keyword, something you are likely to see in code samples online, or written by developers you work with. While the T-SQL engine will parse this without issue, there is a problem when a comma is omitted between column names—the first column will take the name of the second column as its alias. Not only will the column have a misleading name, but you will also have one column too few in your result set. Always use the AS keyword to avoid this problem.

Module 4

Querying Multiple Tables

Contents:

Lesson 1

Understanding Joins

Contents:

Question and Answers

Question: You have the following T-SQL query:

```
SELECT o.ID AS OrderID, o.CustomerName, p.ProductName, p.ModelNumber,
FROM Sales.Orders AS o
JOIN Sales.Products AS p
ON o.ProductID = p.ID;
```

Which of the following types of join will the query perform?

( ) A cross join

( ) An inner join

( ) An outer left join

( ) An outer right join

Answer: The JOIN operator defaults to an INNER join, unless it is qualified with CROSS or OUTER.

Demonstration: Understanding Joins

Demonstration Steps

Use Joins

1. Ensure that the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines are both running, and then log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Run D:\Demofiles\Mod04\Setup.cmd as an administrator. In the **User Account Control** dialog box, click **Yes**. When prompted, type **y**, and then press Enter. When the script has completed, press any key.
3. Start SQL Server Management Studio and connect to the **MIA-SQL** database engine instance using Windows authentication.
4. Open the **Demo.ssmssln** solution in the D:\Demofiles\Mod04\Demo folder.
5. If Solution Explorer is not visible, on the **View** menu, click **Solution Explorer**.
6. In Solution Explorer, expand **Queries**, and then double-click the **11 – Demonstration A.sql** script file.
7. Select the code under the comment **Step 1**, and then click **Execute**.
8. Select the code under the comment **Step 2**, and then click **Execute**.
9. Select the code under the comment **Step 3**, and then click **Execute**.
10. Select the code under the comment **Step 4**, and then click **Execute**.
11. Select the code under the comment **Step 5**, and then click **Execute**.
12. Keep SQL Server Management Studio open for the next demonstration.

Lesson 2

Querying with Inner Joins

Contents:

Question and Answers

Question: You have the following T-SQL query:

```
SELECT HumanResources.Employees.ID, HumanResources.Employers.ID AS
CompanyID,
   HumanResources.Employees.Name, HumanResources.Employers.Name AS
CompanyName
FROM HumanResources.Employees
JOIN HumanResources.Employers
ON HumanResources.Employees.EmployerID = HumanResources.Employers.ID;
```

How can you improve the readability of this query?

Answer: Use AS to create more readable aliases for the Employees and Employers tables.

Demonstration: Querying with Inner Joins

Demonstration Steps

Use Inner Joins

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**, and run D:\Demofiles\Mod04\Setup.cmd as an administrator.
2. If SQL Server Management Studio is not already open, start it and connect to the **MIA-SQL** database engine instance using Windows authentication, and then open the Demo.ssmssln solution in the D:\Demofiles\Mod04\Demo folder.
3. In Solution Explorer, open the **21 – Demonstration B.sql** script file.
4. Select the code under the comment **Step 1**, and then click **Execute**.
5. Select the code under the comment **Step 2**, and then click **Execute**.
6. Select the code under the comment **Step 3**, and then click **Execute**.
7. Select the code under the comment **Step 4**, and then click **Execute**.
8. Select the code under the comment **Step 5**, and then click **Execute**.
9. Select the code under the comment **Step 6**, and then click **Execute**.
10. Keep SQL Server Management Studio open for the next demonstration.

Lesson 3

Querying with Outer Joins

Contents:

Question and Answers

Question: You have a table named PoolCars and a table named Bookings in your ResourcesScheduling database. You want to return all the pool cars for which there are zero bookings. Which of the following queries should you use?

( ) SELECT pc.ID, pc.Make, pc.Model, pc.LicensePlate
FROM ResourcesScheduling.PoolCars AS pc, ResourcesScheduling.Bookings AS b
WHERE pc.ID = b.CarID;

( ) SELECT pc.ID, pc.Make, pc.Model, pc.LicensePlate
FROM ResourcesScheduling.PoolCars AS pc
RIGHT OUTER JOIN ResourcesScheduling.Bookings AS b
ON pc.ID = b.CarID;

( ) SELECT pc.ID, pc.Make, pc.Model, pc.LicensePlate
FROM ResourcesScheduling.PoolCars AS pc
JOIN ResourcesScheduling.Bookings AS b
ON pc.ID = b.CarID;

( ) SELECT pc.ID, pc.Make, pc.Model, pc.LicensePlate
FROM ResourcesScheduling.PoolCars AS pc
LEFT OUTER JOIN ResourcesScheduling.Bookings AS b
ON pc.ID = b.CarID
WHERE b.BookingID IS NULL;

> **Answer:** If a pool car has no bookings, there will be no rows in the Bookings table for that matching Car ID. When you perform a LEFT OUTER JOIN for Bookings to PoolCars, your results include all cars and all bookings for those cars. Because it is a LEFT JOIN, all cars are included even if they have no bookings. Those records, in the result set, have null values for the BookingID. By testing for these null values in the WHERE clause, you can filter out all cars with bookings and leave only those cars with no bookings.

Demonstration: Querying with Outer Joins

Demonstration Steps

Use Outer Joins

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**, and run D:\Demofiles\Mod04\Setup.cmd as an administrator.
2. If SQL Server Management Studio is not already open, start it and connect to the **MIA-SQL** database engine instance using Windows authentication, and then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod04\Demo folder.
3. In Solution Explorer, open the **31 – Demonstration C.sql** script file.
4. Select the code under the comment **Step 1**, and then click **Execute**.
5. Select the code under the comment **Step 2**, and then click **Execute**.
6. Select the code under the comment **Step 3**, and then click **Execute**.
7. Select the code under the comment **Step 4**, and then click **Execute**.
8. Select the code under the comment **Step 5**, and then click **Execute**.

9. Select the code under the comment **Step 6**, and then click **Execute**.
10. Select the code under the comment **Step 7**, and then click **Execute**.
11. Keep SQL Server Management Studio open for the next demonstration.

Lesson 4

Querying with Cross Joins and Self Joins

Contents:

Question and Answers

Question: You have two tables named FirstNames and LastNames. You want to generate a set of fictitious full names from this data. There are 150 entries in the FirstNames table and 250 entries in the LastNames table. You use the following query:

```
SELECT (f.Name + ' ' + l.Name) AS FullName
FROM FirstNames AS f
CROSS JOIN LastNames AS l
```

How many fictitious full names will be returned by this query?

Answer: 37,500 names

Demonstration: Querying with Cross Joins and Self Joins

Demonstration Steps

Use Self Joins and Cross Joins

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**, and run D:\Demofiles\Mod04\Setup.cmd as an administrator.
2. If SQL Server Management Studio is not already open, start it and connect to the **MIA-SQL** database engine instance using Windows authentication, and then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod04\Demo folder.
3. In Solution Explorer, open the **41 – Demonstration D.sql** script file.
4. Select the code under the comment **Step 1**, and then click **Execute**.
5. Select the code under the comment **Step 2**, and then click **Execute**.
6. Select the code under the comment **Step 3**, and then click **Execute**.
7. Select the code under the comment **Step 4**, and then click **Execute**.
8. Close SQL Server Management Studio without saving any files.

Module Review and Takeaways

Best Practice

Table aliases should always be defined when joining tables.

Joins should be expressed using SQL-92 syntax, with JOIN and ON keywords.

Review Question(s)

Question: How does an inner join differ from an outer join?

> **Answer:** An inner join filters out rows which do not satisfy the predicate in the ON clause. An outer join includes all rows from both tables and includes NULLs for attributes where no match is found.

Question: Which join types include a logical Cartesian product?

> **Answer:** CROSS, INNER and OUTER

Question: Can a table be joined to itself?

> **Answer:** Yes, as a self join. An alias to at least one table is required in the FROM clause.

Module 5

Sorting and Filtering Data

Contents:

Lesson 1

Sorting Data

Contents:

Question and Answers

Question: If you declare an alias for a column in the SELECT clause, you cannot use that alias in the WHERE clause—but you can use it in the ORDER BY clause. Why is this?

> **Answer:** This is because of the order in which clauses of a SELECT query are processed. The WHERE clause is processed before the SELECT column list; therefore, column aliases are not available for sorting. The ORDER BY clause is processed last, so column aliases are available and can be used without errors.

Demonstration: Sorting Data

Demonstration Steps

1. Ensure that the 29761A-MIA-DC, and 29761A-MIA-SQL virtual machines are running, and then log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Start SQL Server Management Studio and connect to your Azure instance of the **AdventureWorksLT** database engine instance using SQL Server authentication.
3. If the **Microsoft SQL Server Management Studio** dialog box appears, click **OK**.
4. Open the **Demo.ssmssln** solution in the D:\Demofiles\Mod05\Demo folder.
5. If the Solution Explorer pane is not visible, on the **View** menu, click **Solution Explorer**.
6. Expand **Queries**, and double-click the **11 – Demonstration A.sql** script file.
7. In the **Available Databases** list, click **ADVENTUREWORKSLT**.
8. Select the code under the comment **Step 1**, and then click **Execute**.
9. Select the code under **Step 2**, and then click **Execute**.
10. Select the code under the comment **Step 3** and then click **Execute**.
11. Select the code under the comment **Step 4**, and then click **Execute**.
12. Select the code under the comment **Step 5**, and then click **Execute**.
13. Select the code under the comment **Step 6**, and then click **Execute**.
14. Keep SQL Server Management Studio open for the next demonstration.

Lesson 2

Filtering Data with Predicates

Contents:

Question and Answers

Question: You have a table named Employees that includes a column named StartDate. You want to find who started in any year other than 2014. What query would you use?

Answer: Multiple answers are possible. For example:

```
SELECT FirstName, LastName
FROM Employees
WHERE Employees.StartDate < '20140101' OR Employees.StartDate >= '20150101';
```

Demonstration: Filtering Data with Predicates

Demonstration Steps

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the 29761A-MIA-DC, and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. If SQL Server Management Studio is not already open, start it and connect to your Azure instance of the **AdventureWorksLT** database engine instance using SQL Server authentication, and then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod05\Demo folder.
3. In Solution Explorer, open the **21 – Demonstration B.sql** script file.
4. In the **Available Databases** list, click **ADVENTUREWORKSLT**.
5. Select the code under the comment **Step 1**, and then click **Execute**.
6. Select the code under the comment **Step 2**, and then click **Execute**.
7. Select the code under the comment **Step 3**, and then click **Execute**.
8. Select the code under the comment **Step 4**, and then click **Execute**.
9. Select the code under the comment **Step 5**, and then click **Execute**.
10. Select the code under the comment **Step 6**, and then click **Execute**.
11. Select the code under the comment **Step 7**, and then click **Execute**.
12. Select the code under the comment **Step 8**, and then click **Execute**.
13. Select the code under the comment **Step 9**, and then click **Execute**.
14. Select the code under the comment **Step 10**, and then click **Execute**.
15. Select the code under the comment **Step 11**, and then click **Execute**.
16. Keep SQL Server Management Studio open for the next demonstration.

Lesson 3

Filtering Data with TOP and OFFSET-FETCH

Contents:

Question and Answers

Question: You have a table named Products in your Sales database. You are creating a paged display of products in an application that shows 20 products on each page, ordered by name. Which of the following queries would return the third page of products?

( ) SELECT ProductID, ProductName, ProductNumber
FROM Sales.Products
ORDER BY ProductName ASC
OFFSET 60 ROWS FETCH NEXT 20 ROWS ONLY

( ) SELECT ProductID, ProductName, ProductNumber
FROM Sales.Products
ORDER BY ProductName ASC
OFFSET 40 ROWS FETCH NEXT 20 ROWS ONLY;

( ) SELECT TOP (20) ProductID, ProductName, ProductNumber
FROM Sales.Products
ORDER BY ProductName ASC

( ) SELECT TOP (20) WITH TIES ProductID, ProductName, ProductNumber
FROM Sales.Products
ORDER BY ProductName ASC

Answer: Option 1 returns the fourth page of 20 products. Options 3 and 4 return the first page of 20 products, assuming there are no duplicate product names.

Demonstration: Filtering Data with TOP and OFFSET-FETCH

Demonstration Steps

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the 29761A-MIA-DC, and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd.**
2. If SQL Server Management Studio is not already open, start it and connect to your Azure instance of the **AdventureWorksLT** database engine instance using SQL Server authentication, and then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod05\Demo folder.
3. In Solution Explorer, open the **31 – Demonstration C.sql** script file.
4. In the **Available Databases** list, ensure **ADVENTUREWORKSLT** is selected.
5. Select the code under the comment **Step 1**, and then click **Execute**.
6. Select the code under the comment **Step 2**, and then click **Execute**.
7. Select the code under the comment **Step 3**, and then click **Execute**.
8. Select the code under the comment **Step 4**, and then click **Execute**.
9. Select the code under the comment **Step 5**, and then click **Execute**.
10. Select the code under the comment **Step 6**, and then click **Execute**.
11. Select the code under the comment **Step 7**, and then click **Execute**.
12. Select the code under the comment **Step 8**, and then click **Execute**.
13. Keep SQL Server Management Studio open for the next demonstration.

Lesson 4

Working with Unknown Values

Contents:

Question and Answers

Question: You have the following query:

SELECT e.Name, e.Age

FROM HumanResources.Employees AS e

WHERE YEAR(e.Age) < 1990;

Several employees have asked for their age to be removed from the Human Resources database, and this requested action has been applied to the database. Will the above query return these employees?

Answer: No.

Demonstration: Working with NULL

Demonstration Steps

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the 29761A-MIA-DC, and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. If SQL Server Management Studio is not already open, start it and connect to your Azure instance of the **AdventureWorksLT** database engine instance using SQL Server authentication, and then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod05\Demo folder.
3. In Solution Explorer, open the **41 – Demonstration D.sql** script file.
4. In the **Available Databases** list, ensure **ADVENTUREWORKSLT** is selected.
5. Select the code under the comment **Step 2**, and then click **Execute**.
6. Select the code under the comment **Step 3**, and then click **Execute**.
7. Select the code under the comment **Step 4**, and then click **Execute**.
8. Select the code under the comment **Step 5**, and then click **Execute**.
9. Select the code under the comment **Step 6**, and then click **Execute**.
10. Select the code under the comment **Step 7**, and then click **Execute**.
11. Close SQL Server Management Studio.

Module Review and Takeaways

Question: Does the physical order of rows in an SQL Server table guarantee any sort order in queries using the table?

Answer: No.

Question: You have the following query:

```
SELECT p.PartNumber, p.ProductName, o.Quantity
FROM Sales.Products AS p
LEFT OUTER JOIN Sales.OrderItems AS o
ON p.ID = o.ProductID
ORDER BY o.Quantity ASC
```

You have one new product that has yet to receive any orders. Will this product appear at the top or the bottom of the results?

Answer: At the top of the results.

Lab Review Questions and Answers

Lab: Sorting and Filtering Data

Question and Answers

Lab Review

Question: What is the difference between filtering using the TOP option, and filtering using the WHERE clause?

Answer: The TOP option can only be used to filter results based on the columns specified in the ORDER BY clause; and then it can only be used to return a count of rows (or a percentage of rows) from the top of that result set. There is no support for more complex filters that cannot be expressed in terms of sorting. TOP has no facility for filtering NULL values—NULL values in the columns included in the ORDER BY clause are always returned first.

Filters in the WHERE clause have no such limitations; you can specify complex filters and filters that handle NULL.

Module 6

Working with SQL Server 2016 Data Types

Contents:

Lesson 1

Introducing SQL Server 2016 Data Types

Contents:

Question and Answers

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

Items	
1	tinyint
2	float
3	binary
4	int
5	real
6	varbinary
7	bigint
8	decimal
9	money
10	bit

Category 1		Category 2		Category 3
Exact Numeric Data Types		Approximate Numeric Data Types		Binary Data Types

Answer:

Category 1		Category 2		Category 3
Exact Numeric Data Types		Approximate Numeric Data Types		Binary Data Types
tinyint int bigint decimal money bit		float real		binary varbinary

Resources

Other Data Types

Additional Reading: See course 20472-2: *Developing Microsoft® SQL Server® Databases* for additional information on the XML data type.

Additional Reading: See course 20472-2: *Developing Microsoft® SQL Server® Databases* for additional information on the **hierarchyid** data type.

Demonstration: SQL Server Data Types

Demonstration Steps

1. Ensure that the 29761A-MIA-DC, and 29761A-MIA-SQL virtual machines are running, and then log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Start SQL Server Management Studio and connect to your Azure instance of the **AdventureWorksLT** database engine instance using SQL Server authentication.
3. If the **New Firewall Rule** dialog box appears, click **Sign In**, enter your Azure credentials, and then click **Sign in**. In the **New Firewall Rule** dialog box, ensure **Add my client IP** is selected, and then click **OK**.
4. If the **Microsoft SQL Server Management Studio** dialog box appears, click **OK**.
5. Open the **Demo.ssmssln** solution in the **D:\Demofiles\Mod06\Demo** folder.
6. If the Solution Explorer pane is not visible, on the **View** menu, click **Solution Explorer**.
7. Expand **Queries**, and double-click the **11 – Demonstration A.sql** script file.
8. In the **Available Databases** list, click **AdventureWorksLT**.
9. Select the code under the comment **Step 2**, and then click **Execute**.
10. Select the code under the comment **Step 3**, and then click **Execute**.

11. Select the code under the comment **Step 4**, and then click **Execute**.
12. Keep SQL Server Management Studio open for the next demonstration.

Lesson 2

Working with Character Data

Contents:

Question and Answers

Question: You have the following query:

SELECT FirstName

FROM HumanResources.Employees

WHERE FirstName LIKE N'[^MA]%'

Will the query return an employee with the first name 'Matthew'?

Answer: No.

Demonstration: Working with Character Data

Demonstration Steps

Manipulate Character Data

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the MSL-TMG1, 29761A-MIA-DC, and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. If SQL Server Management Studio is not already open, start it and connect to your Azure instance of the **AdventureWorksLT** database engine instance using SQL Server authentication, and then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod06\Demo folder.
3. In Solution Explorer, open the **21 – Demonstration B.sql** script file.
4. In the **Available Databases** list, click **AdventureWorksLT**.
5. Select the code under the comment **Step 2**, and then click **Execute**.
6. Select the code under the comment **Step 3**, and then click **Execute**.
7. Select the code under the comment **NOTE: this will return no results**, and then click **Execute**.
8. Select the code under the comment **Step 4**, and then click **Execute**.
9. Select the code under the comment **Step 5**, and then click **Execute**.
10. Select the code under the comment **Step 6**, and then click **Execute**.
11. Select the code under the comment **end FORMAT example**, and then click **Execute**.
12. Select the code under the comment **Step 7**, and then click **Execute**.
13. Keep SQL Server Management Studio open for the next demonstration.

Lesson 3

Working with Date and Time Data

Contents:

Question and Answers

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

Items	
1	datetime
2	datetime2
3	DATEFROMPARTS
4	smalldatetime
5	date
6	EOMONTH
7	time
8	datetimeoffset

Category 1		Category 2		Category 3
Present in all versions of SQL Server		Only present in SQL Server 2008 and later		Only present in SQL Server 2012 and later

Answer:

Category 1		Category 2		Category 3
Present in all versions of SQL Server		Only present in SQL Server 2008 and later		Only present in SQL Server 2012 and later
datetime smalldatetime		datetime2 date time datetimeoffset		DATEFROMPARTS EOMONTH

Demonstration: Working with Date and Time Data

Demonstration Steps

Query Data and Time Values

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the MSL-TMG1, 29761A-MIA-DC, and 29761A-MIA-SQL virtual machines, then log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. If SQL Server Management Studio is not already open, start it and connect to your Azure instance of the **AdventureWorksLT** database engine instance using SQL Server authentication, and then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod06\Demo folder.
3. In Solution Explorer, open the **31 – Demonstration C.sql** script file.
4. In the **Available Databases** list, click **AdventureWorksLT**.
5. Select the code under the comment **Step 2**, and then click **Execute**.
6. Select the code under the comment **Step 3**, and then click **Execute**.
7. Select the code under the comment **Step 4**, and then click **Execute**.
8. Select the code under the comment **Step 5**, and then click **Execute**.
9. Select the code under the comment **Step 6**, and then click **Execute**.
10. Select the code under the comment **Step 7**, and then click **Execute**.
11. Close SQL Server Management Studio without saving any files.

Module Review and Takeaways

Question: Will SQL Server be able to successfully and implicitly convert an **int** data type to a **varchar**?

Answer: No, **int** has a higher type precedence than **varchar**.

Question: What data type is suitable for storing Boolean flag information, such as TRUE or FALSE?

Answer: The **bit** data type.

Question: What logical operators are useful for retrieving ranges of date and time values?

Answer: >=, <

Module 7

Using DML to Modify Data

Contents:

Lesson 1

Adding Data to Tables

Contents:

Question and Answers

Using SELECT INTO

Question: You want to populate three columns of an existing table with data from another table in the same database. Which of the following types of query should you use?

( ) INSERT INTO <TableName> (<Columns,…>) VALUES (<Column Value> …)

( ) INSERT INTO <DestinationTableName> SELECT <Columns> FROM <SourceTableName>

( ) INSERT INTO <DestinationTableName> EXECUTE usp_SomeStorerdProcedure

( ) SELECT <Columns,…> INTO DestinationTableName FROM SourceTableName

( ) SELECT <Columns,…> INTO SourceTableName FROM DestinationTableNAme

Answer:

(√) INSERT INTO <TableName> (<Columns,…>) VALUES (<Column Value> …)

( ) INSERT INTO <DestinationTableName> SELECT <Columns> FROM <SourceTableName>

( ) INSERT INTO <DestinationTableName> EXECUTE usp_SomeStorerdProcedure

( ) SELECT <Columns,…> INTO DestinationTableName FROM SourceTableName

( ) SELECT <Columns,…> INTO SourceTableName FROM DestinationTableNAme

Demonstration: Adding Data to Tables

Demonstration Steps

INSERT Data into a Table

1. Start the **29761AD-MIA-DC** and **29761A-MIA-SQL** virtual machines, log on to **29761A-MIA-SQL** as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**,
2. Run **D:\Demofiles\Mod07\Setup.cmd** as an administrator. Click **Yes** when prompted.
3. In the Command Prompt window press **y**, and then press Enter.
4. When the script has finished, press Enter.
5. Open **SQL Server Management Studio**, and connect to the **MIA-SQL** database engine instance using Windows authentication.
6. In the **D:\Demofiles\Mod07\Demo** folder, open the **Demo.ssmssln** solution.
7. In Solution Explorer, expand **Queries**, and double-click **11 - Demonstration A.sql**.
8. Highlight the code **USE TSQL GO**, and click **Execute**.
9. First we will populate a table with some data from a stored procedure. Highlight the code under the comment that begins **-- First try the INSERT by stored procedure**

```
INSERT INTO Production.Products
   (       productID
   ,       productname
   ,       supplierid
   ,       categoryid
   ,       unitprice)
EXEC Production.AddNewProducts;
```

Click **Execute** you will receive a message saying the procedure is not there.

10. Highlight the code below the comment **--Create a backup of the Products with a chosen ID**, and click **Execute**.

```
DROP TABLE IF EXISTS NewProducts
GO
SELECT * INTO NewProducts
FROM PRODUCTION.PRODUCTS WHERE ProductID >= 70
```

We are creating a new Table for NewProducts where the Product ID >= 70.

11. We are also going to create a NewOrderDetails table that will contain rows for those products that have been transferred into NewProducts. To do this highlight the code under the comment **-- Create a backup of the Order Details for the chosen productID,** up to the point shown in the code section for the next step below, and click **Excute.**

```
DROP TABLE IF EXISTS NewOrderDetails
GO
SELECT * INTO NewOrderDetails
FROM SALES.OrderDetails WHERE ProductID >= 70
-- Delete the copied data from the original tables
DELETE FROM SALES.OrderDetails
OUTPUT DELETED.*
WHERE ProductID >= 70
DELETE FROM Production.Products
OUTPUT DELETED.*
WHERE ProductID >= 70
-- check that they have been transferred safely
SELECT * FROM NewProducts
SELECT * FROM NewOrderDetails
SELECT * FROM SALES.OrderDetails
WHERE productid >= 70
SELECT * FROM Production.Products
WHERE productid >= 70
```

12. Highlight the code below the **Now we can put back the rows from the NewTables, using the INSERT statement**, and click **Execute**.

```
DROP PROCEDURE IF EXISTS Production.AddNewProducts
GO
CREATE PROCEDURE Production.AddNewProducts
AS
BEGIN
SELECT Productid, productname, SupplierID, CategoryID, Unitprice FROM NewProducts
END
```

When you click Execute. SQL Server creates the stored procedure that we were previously missing when we tried to run it at the beginning of the demo.

13. Now we need to populate the original products table with the data within the secondary table as if it was new rows that we are adding. Highlight the code below the comment **– Having created it, we can run it to feed the missing rows into the Products table**

```
INSERT INTO Production.Products (productid, productname, supplierid, categoryid,
unitprice)
EXEC Production.AddNewProducts;
SELECT * FROM Production.Products
WHERE productid >= 70
```

And click **Execute**, to transfer the rows and see that they have been transferred.

14. For the other table we will use the SELECT INSERT statement. Highlight the code below the comment **-- The OrderDetails will be put back using INSERT .. SELECT** and click **Execute**.

```
INSERT Sales.OrderDetails (orderid, productid, unitprice, qty, discount)
OUTPUT INSERTED.*
SELECT * FROM NewOrderDetails
```

15. Having seen various ways to add data to a new or existing table, we can clean up the database by dropping the objects used in this demo. Highlight the rest of the code below **-- Clean up the database** and click **Execute**

```
DROP TABLE NewProducts
GO
DROP TABLE NewOrderDetails
GO
DROP PROCEDURE Production.AddNewProducts
```

Lesson 2

Modifying and Removing Data

Contents:

Question and Answers

Question: A user cannot delete records in the Cars table by using a DELETE statement. His query was intended to remove all pool cars that have been sold. The query used was:

DELETE

FROM Scheduling.Cars

WHERE Cars.DateSold <> NULL

What mistake did the user make?

Answer: An expression comparison operator '<>' was used instead of the IS NOT NULL clause.

Demonstration: Manipulating Data Using the UPDATE and DELETE Statements and MERGING Data Using Conditional DML

Demonstration Steps

Update and Delete Data in a Table

1. Start the **29761AD-MIA-DC** and **29761A-MIA-SQL** virtual machines, log on to **29761A-MIA-SQL** as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**,
2. Run **D:\Demofiles\Mod07\Setup.cmd** as an administrator.
3. In the Command Prompt window press **y**, and then press **Enter**.
4. When the script has finished, press **Enter**.
5. If SQL Server Management Studio is not already open, start it and connect to the **MIA-SQL** database engine instance using Windows authentication,
6. Open the **Demo.ssmssln** solution in the D:\Demofiles\Mod07\Demo folder.
7. In Solution Explorer, open the **21 – Demonstration B.sql** script file.
8. Highlight the code **USE AdventureWorks GO**, and click **Execute**.
9. Select the code under **USE AdventureWorks GO**, and then click **Execute**.
10. Select the code under the comment **Remove the copied rows from the store table**, and then click **Execute**.
11. Select the code under the comment **Show that they have been removed**, and then click **Execute**.
12. Select the code under the comment **Use the Merge statement to put them back**, and then click **Execute**.
13. Select the code under the comment **SELECT * FROM Sales.Store where 1 = 0 -- used to extract column names for all columns, without cost of data access**, and then click **Execute**.
14. Select the code under the comment **Use the Merge statement to Change the names back**, and then click **Execute**.
15. Select the code under the comment **Ensure that the environment has been restored to the state it was in before the changes were made**, and then click **Execute**.
16. Select the code under the comment **Clean up the database**, and then click **Execute**.
17. Close SQL Server Management Studio without saving any files.

Lesson 3

Generating Automatic Column Values

Contents:

Question and Answers

Using IDENTITY

Question: You are using an IDENTITY column to store the sequence in which orders were placed in a given year. It is a new year and you want to start the count again from 1. Which of the following statements should you use?

( ) OrderSequence int IDENTITY(1,1) NOT NULL

( ) SET IDENTITY INSERT

( ) SCOPE_IDENTITY()

( ) DBCC CHECKIDENT

( ) CREATE SEQUENCE

Answer:

(√) OrderSequence int IDENTITY(1,1) NOT NULL

( ) SET IDENTITY INSERT

( ) SCOPE_IDENTITY()

( ) DBCC CHECKIDENT

( ) CREATE SEQUENCE

Common Issues and Troubleshooting Tips

Common Issue	Troubleshooting Tip
You are part way through the exercises and want to start again from the beginning. You run the set-up script within the solution and receive lots of error messages. This may occur if you have tried to execute the set-up script without running the clean-up script to remove any changes you may have made during the lab.	Run the clean-up script before running the set-up script.

Lab Review Questions and Answers

Lab: Using DML to Modify Data

Question and Answers

Lab Review

Question: What attributes of the source columns are transferred to a table created with a SELECT INTO query?

Answer: Name, data type, and whether the column is NULL enabled.

Question: The presence of which constraint prevents TRUNCATE TABLE from executing successfully?

Answer: A foreign key reference to the table.

Module 8

Using Built-In Functions

Contents:

Lesson 1

Writing Queries with Built-In Functions

Contents:

Question and Answers

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

Items	
1	GETDATE()
2	SUM()
3	OPENDATASOURCE()
4	DATEADD()
5	MIN()
6	OPENQUERY()
7	UPPER()
8	MAX()
9	OPENROWSET()
10	YEAR()
11	COUNT()
12	OPENXML()
13	ABS()
14	AVG()
15	DB_NAME()

Category 1		Category 2		Category 3
Scalar Functions		Aggregate Functions		Rowset Functions

Answer:

Category 1		Category 2		Category 3
Scalar Functions		Aggregate Functions		Rowset Functions
GETDATE() DATEADD() UPPER() YEAR() ABS() DB_NAME()		SUM() MIN() MAX() COUNT() AVG()		OPENDATASOURCE() OPENQUERY() OPENROWSET() OPENXML()

Demonstration: Writing Queries Using Built-in Functions

Demonstration Steps

Use Built-in Scalar Functions

1. Ensure that the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines are both running, and then log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Run **D:\Demofiles\Mod08\Setup.cmd** as an administrator. Click **Yes** when prompted.
3. When prompted press **y**, and then press Enter.
4. When the script has finished press Enter.
5. Start SQL Server Management Studio and connect to the **MIA-SQL** database engine instance using Windows authentication.
6. Open the **Demo.ssmssln** solution in the D:\Demofiles\Mod08\Demo folder.
7. If the Solution Explorer pane is not visible, on the **View** menu, click **Solution Explorer**.
8. Expand Queries, and double-click the **11 – Demonstration A.sql** script file.
9. Select the code under the comment **Step 1**, and then click **Execute**.
10. Select the code under the comment **Step 2**, and then click **Execute**.
11. Select the code under the comment **Step 3**, and then click **Execute**.
12. Select the code under the comment **Step 4**, and then click **Execute**.
13. Keep SQL Server Management Studio open for the next demonstration.

Lesson 2

Using Conversion Functions

Contents:

Question and Answers

Question: You are writing a query against a Human Resources database. You want to ensure that the Employee.StartDate values are displayed in standard British form. What function should you use?

Answer: PARSE() or TRY_PARSE.

Demonstration: Using Conversion Functions

Demonstration Steps

Use Functions to Convert Data

1. If you have not completed the previous demonstration or need to start over, perform these two steps before proceeding to step 2; otherwise go to step 2:
   a. Start the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**, and run D:\Demofiles\Mod08\Setup.cmd as an administrator.
   b. If SQL Server Management Studio is not already open, start it and connect to the **MIA-SQL** database engine instance using Windows authentication, and then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod08\Demo folder.
2. In Solution Explorer, open the **21 – Demonstration B.sql** script file.
3. Select the code under the comment **Step 1**, and then click **Execute**.
4. Select the code under the comment **Step 2**, and then click **Execute**.
5. Select the code under the comment **Step 3**, and then click **Execute**.
6. Select the code under the comment **Step 4**, and then click **Execute**.
7. Select the code under the comment **THIS WILL FAIL at converting datetime2 to int**, and then click **Execute**.
8. Select the code under the comment **Step 5**,and then click **Execute**.
9. Select the code under the comment **Step 6**, and then click **Execute**.
10. Select the code under the comment **Step 7**, and then click **Execute**.
11. Select the code under the comment **Step 8**, and then click **Execute**.
12. Select the code under the comment **This will succeed**, and then click **Execute**.
13. Keep SQL Server Management Studio open for the next demonstration.

Lesson 3

Using Logical Functions

Contents:

Question and Answers

Question:

You have the following query:

SELECT e.FirstName, e.LastName, e.FirstAider

FROM Employees AS e

The FirstAider column contains ones and zeros. How can you change the query to make the results more readable?

Answer:

Use an IIF function:

SELECT e.FirstName, e.LastName,

IIF(e.FirstAider = 1, 'Can administer First Aid','No First Aid Skills')

FROM Employees AS e

This will output descriptive text instead of ones and zeros.

Demonstration: Using Logical Functions

Demonstration Steps

Using Logical Functions

1. If you have not completed the previous demonstration or need to start over, perform these two steps before proceeding to step 2; otherwise go to step 2:
   a. Start the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**, and run D:\Demofiles\Mod08\Setup.cmd as an administrator.
   b. If SQL Server Management Studio is not already open, start it and connect to the **MIA-SQL** database engine instance using Windows authentication, and then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod08\Demo folder.
2. In Solution Explorer, open the **31 – Demonstration C.sql** script file.
3. Select the code under the comment **Step 1**. and then click **Execute**.
4. Select the code under the comment **Step 2**, and then click **Execute**.
5. Select the code under the comment **Step 3**, and then click **Execute**.
6. Select the code under the comment **Step 4**, and then click **Execute**.
7. Select the code under the comment **Step 5**, and then click **Execute**.
8. Keep SQL Server Management Studio open for the next demonstration.

Lesson 4

Using Functions to Work with NULL

Contents:

Question and Answers

Question: You are writing a query against the Employees table in the Human Resources database. The CurrentStatus column can contain the string values "New", "Retired", and "Under Caution". Many employees have this column set to NULL when those statuses do not apply to them. For confidentiality, you want to ensure that the employees currently under caution are displayed like those employees with no applicable status. What function should you use?

( ) ISNULL()

( ) COALESCE()

( ) NULLIF()

( ) TRY_PARSE()

( ) PARSE()

Answer: You can use NULLIF(Employees.CurrentStatus, 'Under Caution') to display the employees currently under caution; for example, employees with no applicable status.

Demonstration: Using Functions to Work with NULL

Demonstration Steps

Use Functions to Work with NULL

1. If you have not completed the previous demonstration or need to start over, perform these two steps before proceeding to step 2; otherwise go to step 2:
   a. Start the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**, and run D:\Demofiles\Mod08\Setup.cmd as an administrator.
   b. If SQL Server Management Studio is not already open, start it and connect to the **MIA-SQL** database engine instance using Windows authentication, and then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod08\Demo folder.
2. In Solution Explorer, open the **41 – Demonstration D.sql** script file.
3. Select the code under the comment **Step 1**, and then click **Execute**.
4. Select the code under the comment **Step 2**, and then click **Execute**.
5. Select the code under the comment **Step 3**, and then click **Execute**.
6. Select the code under the comment **First, set up sample data**, and then click **Execute**.
7. Select the code under the comment **Populate the sample data**, and then click **Execute**.
8. Select the code under the comment **Show the sample data**, and then click **Execute**.
9. Select the code under the comment **Use NULLIF to show which employees have actual values different from their goals**, and then click **Execute**.
10. Select the code under the comment **Step 5**, and then click **Execute**.
11. Close SQL Server Management Studio without saving any files.

Module Review and Takeaways

Best Practice

When possible, use standards-based functions, such as CAST or COALESCE, rather than SQL Server-specific functions like NULLIF or CONVERT.

Consider the impact of functions in a WHERE clause on query performance.

Review Question(s)

Question: Which function should you use to convert from an int to a nchar(8)?

Answer: CAST()

Question: Which function will return a NULL, rather than an error message, if it cannot convert a string to a date?

Answer: TRY_CONVERT().

Question: What is the name for a function that returns a single value?

Answer: A scalar function.

Module 9

Grouping and Aggregating Data

Contents:

Lesson 1

Using Aggregate Functions

Contents:

Question and Answers

Question: You have the following query:

SELECT COUNT(*) AS RecordCount

FROM Sales.Products;

There are 250 records in the Products table. How many rows will be returned by this query?

Answer: One.

Demonstration: Using Aggregate Functions

Demonstration Steps

Use Built-in Aggregate Functions

1. Ensure that the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines are both running, and then log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Run D:\Demofiles\Mod09\Setup.cmd as an administrator. Click **Yes** when prompted.
3. Start SQL Server Management Studio and connect to the MIA-SQL database instance using windows authentication.
4. On the **File** menu, point to **Open**, and then click **File**.
5. Browse to **D:\Demofiles\Mod09\Demo** folder, click **Demo.ssmssln**, and then click **Open**.
6. If the Solution Explorer pane is not visible, on the **View** menu, click **Solution Explorer**.
7. Expand the **Queries** folder, double-click the **11 – Demonstration A.sql** script file.
8. Select the code under the comment **Step 1: Using built-in Aggregate functions**, and then click **Execute**.
9. Select the code under the comment **Step 2: Using built-in Aggregate functions**, and then click **Execute**.
10. Select the code under the comment **Select and execute the following query to show This will succeed and return the AVG/MIN/MAX of all rows**, and then click **Execute**.
11. Select the first instance, under the comment **Select and execute the following query to show the use of aggregates with non-numeric data types**, and then click **Execute**.
12. Select the second instance, under the comment **Select and execute the following query to show the use of aggregates with non-numeric data types**, and then click **Execute**.
13. Select the code under the comment **Select and execute the following query to show the use of DISTINCT with aggregate functions**, and then click **Execute**.
14. Select the code under the comment **Select and execute the following query to show the impact of NULL on aggregate functions First, show the existence of NULLs in Sales.Orders**, and then click **Execute**.
15. Select the code under the comment **Then show that MIN, MAX and COUNT ignore NULL, COUNT(*) doesn't. Show the messages tab in the SSMS results pane for Warning: Null value is eliminated by an aggregate or other SET operation**, and then click **Execute**.
16. Select the code under the comment **Create an example table**, and then click **Execute**.
17. Select the code under the comment **Populate it**, and then click **Execute**.
18. Select the code under the comment **View the contents. Note the NULL**, and then click **Execute**.

19. Select the code under the comment **Execute this query to compare the behavior of AVG to an arithmetic average (SUM/COUNT)**, and then click **Execute**.
20. Select the code under the comment **Clean up the created table**, and then click **Execute**.
21. Select the code under the comment **Execute this query to demonstrate replacement of NULL before aggregating**, and then click **Execute**.
22. Select the code under the comment **Populate test table**, and then click **Execute**.
23. Select the code under the comment **Show table contents**, and then click **Execute**.
24. Select the code under the comment **Show standard AVG versus replacement of NULL with zero**, and then click **Execute**.
25. Select the code under the comment **clean up** and then click **Execute**.
26. Keep SQL Server Management Studio open for the next demonstration.

Lesson 2

Using the GROUP BY Clause

Contents:

Question and Answers

Question: You are writing the following T-SQL query to find out how many employees work in each department in your organization:

SELECT d.DepartmentID, d.DepartmentName, COUNT(e.EmployeeID) AS EmployeeCount

FROM HumanResources.Departments AS d

INNER JOIN HumanResources.Employees AS e

ON d.DepartmentID = e.DepartmentID

GROUP BY

Which columns should be included in the GROUP BY clause?

( ) All Columns

( ) EmployeeCount

( ) DepartmentID, DepartmentName

( ) DepartmentID

Answer: When using an aggregate function in the SELECT clause, all columns not included in an aggregate function must be included in the GROUP BY clause, otherwise an error will occur.

Demonstration: Using GROUP BY

Demonstration Steps

Use the GROUP BY Clause

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**, and run D:\Demofiles\Mod09\Setup.cmd as an administrator.
2. If SQL Server Management Studio is not already open, start it and connect to the **MIA-SQL** database engine instance using Windows authentication, and then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod09\Demo folder.
3. In Solution Explorer, open the **21 – Demonstration B.sql** script file.
4. Select the code under the comment **Step 1: Using GROUP BY**, and then click **Execute**.
5. Select the code under the comment **Step 2: Using GROUP BY**, and then click **Execute**.
6. Select the code under the comment **Select this query and execute it to show customer orders per customer and per year**, and then click **Execute**.
7. Select the code under the comment **Step 3: Workflow of grouping**, and then click **Execute**.
8. Select the code under the comment **Step 4: Using Aggregates with GROUP BY**, and then click **Execute**.
9. Select the code under the comment **Show an aggregate on a column not in GROUP BY list**, and then click **Execute**.
10. Keep SQL Server Management Studio open for the next demonstration.

Lesson 3

Filtering Groups with HAVING

Contents:

Question and Answers

Question: You are writing a query to count the number of orders placed for each product. You have the following query:

SELECT p.ProductName, COUNT(*) AS OrderCount

FROM Sales.Products AS p

JOIN Sales.OrderItems AS o

ON p.ProductID = o.ProductID

GROUP BY p.ProductName;

You want to change the query to return only products that cost more than $10. Should you add a HAVING clause or a WHERE clause?

Answer: Add a WHERE clause such as WHERE p.Price > 10.

Demonstration: Filtering Groups with HAVING

Demonstration Steps

Filter Grouped Data Using the HAVING Clause

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**, and run D:\Demofiles\Mod09\Setup.cmd as an administrator.
2. If SQL Server Management Studio is not already open, start it and connect to the **MIA-SQL** database engine instance using Windows authentication, and then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod09\Demo folder.
3. In Solution Explorer, open the **31 – Demonstration C.sql** script file.
4. Select the code under the comment **Step 1: Filtering Groups with HAVING Change to Adventureworks database**, and then click **Execute**.
5. Select the code under the comment **Step 2: Using the HAVING clause**, and then click **Execute**.
6. Select the code under the comment **This query uses a HAVING clause to filter out customers with fewer than 10 orders**, and then click **Execute**.
7. Select the code under the comment **Review the logical order of operations**, and then click **Execute**.
8. Select the code under the comment **Select and execute the following queries to show difference between WHERE filter and HAVING filter**, and then click **Execute**.
9. Select the code under the comment **This query uses a HAVING clause to filter groups with an average quantity > 20**, and then click **Execute**.
10. Select the code under the comment **Select and execute the following query to show All customers and how many orders they have placed 89 rows**, and then click **Execute**.
11. Select the code under the comment **Use HAVING to filter only customers who have placed more than 20 orders**, and then click **Execute**.
12. Select the code under the comment **Select and execute the following query to show All products and in how many orders they appear**, and then click **Execute**.

13. Select the code under the comment **Use HAVING to filter only products which have been ordered less than 20 times**, and then click **Execute**.
14. Close SQL Server Management Studio without saving any files.

Module Review and Takeaways

Question: What is the difference between the COUNT function and the COUNT_BIG function?

Answer: COUNT returns an int; COUNT_BIG returns a big_int.

Question: Can a GROUP BY clause include more than one column?

Answer: Yes, separated by commas.

Question: In a query, can a WHERE clause and a HAVING clause filter on the same column?

Answer: Yes.

Module 10

Using Subqueries

Contents:

Lesson 1

Writing Self-Contained Subqueries

Contents:

Question and Answers

Question: You are troubleshooting a query. The outer query contains an inner query in its WHERE clause. The first inner query also contains a second inner query in its WHERE clause. Both inner queries are self-contained. The complete query returns an error. How should you approach this task?

> **Answer:** Break the complete query into its constituent subqueries. Start with the inner most subquery and test to see if it returns the information you expect. Next, test the first inner query. Finally, test the complete query.

Demonstration: Writing Self-Contained Subqueries

Demonstration Steps

Write a Nested Subquery

1. Ensure that the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines are both running, and then log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. In the **D:\Demofiles\Mod10** folder, right-click **Setup.cmd** and click **Run as administrator**.
3. Click **Yes** when prompted to confirm you want to run the command file, and wait for the script to finish.
4. When prompted press any key.
5. Start SQL Server Management Studio and connect to the **MIA-SQL** database engine instance using Windows® authentication.
6. On the **File** menu, point to **Open**, click **File**, browse to the **D:\Demofiles\Mod10\Demo** folder, and then double-click **Demo.ssmssln**.
7. If the Solution Explorer pane is not visible, on the **View** menu, click **Solution Explorer**.
8. Expand the **Queries** folder, and double-click the **11 – Demonstration A.sql** script file.
9. Select the code under the comment **Step 1: Open a new query window to the TSQL database**, and then click **Execute**.
10. Select the code under the comment **Step 2: Scalar subqueres**, and then click **Execute**.
11. Select the code under the comment **Select this query and execute it to find details in Sales.OrderDetails for most recent order**, and then click **Execute**.
12. Select the code under the comment **THIS WILL FAIL, since subquery returns more than 1 value**, and then click **Execute**.
13. Select the code under the comment **Step 3: Multi-valued subqueries**, and then click **Execute**.
14. Select the code under the comment **Same result expressed as a join**, and then click **Execute**.
15. Keep SQL Server Management Studio open for the next demonstration.

Lesson 2

Writing Correlated Subqueries

Contents:

Question and Answers

Working with Correlated Subqueries

Question: Why can't a correlated subquery be executed separately from the outer query?

Answer: The subquery depends on input from the outer query for its values.

Question: Which of the following statements about correlated subqueries is correct?

( ) To troubleshoot a correlated subquery, execute the inner query first on its own, before placing it into the outer query.

( ) In a correlated subquery, the inner query is run only once, regardless of the number of rows the outer query returns.

( ) In a correlated subquery, the inner query uses data returned by the outer query.

( ) In a correlated subquery, the inner query is executed first, the outer query second.

Answer: You cannot troubleshoot a correlated subquery by executing the inner query separately, because the inner query requires data from the outer query. In a correlated subquery, multiple values may be passed into the inner query, which would require the inner query to run multiple times. The outer query must be executed first, so that data is available to pass to the inner query.

Demonstration: Writing Correlated Subqueries

Demonstration Steps

Write a Correlated Subquery

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**, and run D:\Demofiles\Mod10\Setup.cmd as an administrator.
2. If SQL Server Management Studio is not already open, start it and connect to the **MIA-SQL** database engine instance using Windows authentication, and then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod10\Demo folder.
3. In Solution Explorer, open the **21 – Demonstration B.sql** script file.
4. Select the code under the comment **Step 1: Open a new query window to the TSQL database**, and then click **Execute**.
5. Select the code under the comment **Step 2: Correlated subqueries**, and then click **Execute**.
6. Select the code under the comment **Select and execute the following query to show the use of a correlated subquery that uses the empid from Sales.Orders to retrieve orders placed by an employee on the latest order date for each employee**, and then click **Execute**.
7. Select the code under the comment **Select and execute the following query to show the use of a correlated subquery that uses the custid from Sales.Custorders to retrieve orders placed by a customer with the highest quantity for each customer**, and then click **Execute**.
8. Keep SQL Server Management Studio open for the next demonstration.

Lesson 3

Using the EXISTS Predicate with Subqueries

Contents:

Question and Answers

Question: The Human Resources database has recently been extended to record the skills possessed by employees. Employees have added their skills to the database by using a web-based user interface. You want to find employees who have not yet added their skills. You have the following query:

```
SELECT e.EmployeeID, e.FirstName
FROM HumanResources.Employees AS e
WHERE NOT EXISTS (
      SELECT s.EmployeeID, s.SkillName, s.SkillCategory
      FROM HumanResources.Skills AS s
      WHERE e.EmployeeID = s.EmployeeID);
```

How can you improve the query?

Answer: In the inner query SELECT clause, replace the list of columns with "*".

Demonstration: Writing Subqueries Using EXISTS

Demonstration Steps

Write Queries Using EXISTS and NOT EXISTS

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the 29761A-MIA-DC and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**, and run D:\Demofiles\Mod10\Setup.cmd as an administrator.
2. If SQL Server Management Studio is not already open, start it and connect to the **MIA-SQL** database engine instance using Windows authentication, then open the **Demo.ssmssln** solution in the D:\Demofiles\Mod10\Demo folder.
3. In Solution Explorer, open the **31 – Demonstration C.sql** script file.
4. Select the code under the comment **Step 1: Open a new query window to the TSQL database**, and then click **Execute**.
5. Select the code under the comment **Step 2: Using EXISTS**, and then click **Execute**.
6. Select the code under the comment **Step 3: Using NOT EXISTS**, and then click **Execute**.
7. Select the code under the comment **Step 4: Compare COUNT(*)>0 to EXISTS**, and then click **Execute**.
8. Select the code under the comment **Use EXISTS**, and then click **Execute**.
9. Close SQL Server Management Studio without saving any files.

Module Review and Takeaways

Question: Can a correlated subquery return a multi-valued set?

Answer: Yes.

Question: What type of subquery may be rewritten as a JOIN?

Answer: Correlated subqueries.

Question: Which columns should appear in the SELECT list of a subquery following the EXISTS predicate?

Answer: Only a * needs to be specified. No actual columns will be retrieved.

Module 11

Using Set Operators

Contents:

Lesson 1

Writing Queries with the UNION Operator

Contents:

Question and Answers

Question: The results from a UNION query can contain duplicate rows.

( ) True

( ) False

Answer:

( ) True

(√) False

Question: When combining the output of two sets, UNION and UNION ALL queries cannot include rows with NULL values, because NULL values cannot be compared.

( ) True

( ) False

Answer:

( ) True

(√) False

Demonstration: Using UNION and UNION ALL

Demonstration Steps

1. Ensure that the MSL-TMG1, 29761A-MIA-DC, and 29761A-MIA-SQL virtual machines are running, and then log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. Start SQL Server Management Studio and connect to your Azure instance of the **AdventureWorksLT** database engine instance using SQL Server authentication.
3. If the **Microsoft SQL Server Management Studio** dialog box appears, click **OK**.
4. Open the **Demo.ssmssln** solution in the **D:\Demofiles\Mod11\Demo** folder.
5. If the Solution Explorer pane is not visible, on the **View** menu, click **Solution Explorer**.
6. Expand **Queries**, and double-click the **11 – Demonstration A.sql** script file.
7. In the **Available Databases** list, click **AdventureWorksLT**.
8. Select the code under the comment **Step 2**, and then click **Execute**.
9. Select the code under the comment **Step 3**, and then click **Execute**.
10. Keep SQL Server Management Studio open for the next demonstration.

Lesson 2

Using EXCEPT and INTERSECT

Contents:

Question and Answers

Question: You have a table of employees and a table of customers, both of which contain a column holding the name of the country where the customer or employee is located. You want to know which countries have at least one customer and at least one employee. Which set operator should you use?

( ) UNION ALL

( ) UNION

( ) EXCEPT

( ) INTERSECT

( ) None of the above

Answer: You should use INTERSECT.

UNION lists all results that appear in either set, which would list all countries with at least one customer or at least one employee. UNION ALL would include duplicates found in the sets.

EXCEPT lists results that appear in only the first set, which would list countries with at least one customer but no employee; or a list of countries with at least one employee but no customer, depending on the order in which the tables appear in your query.

INTERSECT lists results that appear in both sets—which is the list of countries you want to find.

Demonstration: Using EXCEPT and INTERSECT

Demonstration Steps

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the MSL-TMG1, 29761A-MIA-DC, and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. If SQL Server Management Studio is not already open, start it and connect to your Azure instance of the **AdventureWorksLT** database engine instance using SQL Server authentication, and then open the **Demo.ssmssln** solution in the **D:\Demofiles\Mod11\Demo** folder.
3. In Solution Explorer, open the **21 – Demonstration B.sql** script file.
4. In the **Available Databases** list, click **AdventureWorksLT**.
5. Select the code under the comment **Step 2**, and then click **Execute**.
6. Select the code under the comment **Step 3**, and then click **Execute**.
7. Select the code under the comment **Step 4**, and then click **Execute**.
8. Select the code under the comment **Step 5**, and then click **Execute**.
9. Select the code under the comment **Step 6**, and then click **Execute**.
10. Keep SQL Server Management Studio open for the next demonstration.

Lesson 3

Using APPLY

Contents:

Question and Answers

Question: What is the difference between CROSS APPLY and CROSS JOIN?

> **Answer:** CROSS JOIN returns all the possible combinations of the left and right table sources; CROSS APPLY returns only the values from the left table source where a value is found in the right table source.

Demonstration: Using CROSS APPLY and OUTER APPLY

Demonstration Steps

1. Ensure that you have completed the previous demonstration in this module. Alternatively, start the, 29761A-MIA-DC, and 29761A-MIA-SQL virtual machines, log on to 29761A-MIA-SQL as **ADVENTUREWORKS\Student** with the password **Pa$$w0rd**.
2. If SQL Server Management Studio is not already open, start it and connect to your Azure instance of the **AdventureWorksLT** database engine instance using SQL Server authentication, and then open the **Demo.ssmssln** solution in the **D:\Demofiles\Mod11\Demo** folder.
3. In Solution Explorer, open the **31 – Demonstration C.sql** script file.
4. In the **Available Databases** list, click **AdventureWorksLT**.
5. Select the code under the comment **Step 2**, and then click **Execute**.
6. Select the code under the comment **Step 3**, and then click **Execute**.
7. Select the code under the comment **Step 4**, and then click **Execute**.
8. Select the code under the comment **Step 5**, and then click **Execute**.
9. Select the code under the comment **Test with CROSS APPLY**, and then click **Execute**.
10. Select the code under the comment **Step 6**, and then click **Execute**.
11. Select the code under the comment **Step 7**, and then click **Execute**.
12. Select the code under the comment **Use OUTER APPLY to include customers with no orders**, and then click **Execute**.
13. Close SQL Server Management Studio, without saving any changes.

Module Review and Takeaways

Question: Which set operator would you use to combine sets if you knew there were no duplicates and wanted the best possible performance?

Answer: UNION ALL

Question: Which form of the APPLY operator will not return rows from the left table if the result of the right table expression was empty?

Answer: CROSS APPLY

Question: Which form of the APPLY operator can be used to rewrite LEFT OUTER JOIN queries?

Answer: OUTER APPLY