Lab Answer Key: Module 8: Using Built-In Functions

Lab: Using Built-in Functions

Exercise 1: Writing Queries That Use Conversion Functions

Task 1: Prepare the Lab Environment

1. Ensure that the **20761C-MIA-DC** and **20761C-MIA-SQL** virtual machines are both running, and then log on to **20761C-MIA-SQL** as **ADVENTUREWORKS\Student** with the password **Pa55w.rd**.
2. In the **D:\Labfiles\Lab08\Starter** folder, right-click **Setup.cmd**, and then click **Run as administrator**.
3. In the **User Account Control** dialog box, click **Yes**.
4. At the command prompt, type **y**, and then press Enter.
5. Wait for the script to finish, and press Enter.

Task 2: Write a SELECT Statement that Uses the CAST or CONVERT Function

1. Start SQL Server Management Studio and connect to the **MIA-SQL** database engine using Windows authentication.
2. On the **File** menu, point to **Open**, and then click **Project/Solution**.
3. In the **Open Project** dialog box, navigate to the **D:\Labfiles\Lab08\Starter\Project** folder, and then double-click **Project.ssmssln**.
4. In Solution Explorer, expand the **Queries** folder, and then double-click **51 - Lab Exercise 1.sql**.
5. In the query pane, highlight the statement **USE TSQL;**, and then click **Execute**.
6. In the query pane, type the following query after the **Task 1** description:

```
SELECT N'The unit price for the ' + productname + N' is ' + CAST(unitprice AS
NVARCHAR(10)) + N' $.' AS productdesc
FROM Production.Products;
```

   This query uses the CAST function rather than the CONVERT function. It is better to use the CAST function because it is an ANSI SQL standard. You should use the CONVERT function only when you need to apply a specific style during a conversion.

7. Highlight the written query, and click **Execute**.

Task 3: Write a SELECT Statement to Filter Rows Based on Specific Date Information

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

```
SELECT orderid, orderdate, shippeddate, COALESCE(shipregion, 'No region') AS shipregion
FROM Sales.Orders
WHERE
orderdate >= CONVERT(DATETIME, '4/1/2007', 101)
AND orderdate <= CONVERT(DATETIME, '11/30/2007', 101)
AND shippeddate > DATEADD(DAY, 30, orderdate);
```

2. Highlight the written query and click **Execute**.
3. Note that you could also write a solution using the PARSE function. The query would look like this:

```
SELECT orderid, orderdate, shippeddate, COALESCE(shipregion, 'No region') AS shipregion
FROM Sales.Orders
WHERE
orderdate >= PARSE('4/1/2007' AS DATETIME USING 'en-US')
AND orderdate <= PARSE('11/30/2007' AS DATETIME USING 'en-US')
AND shippeddate > DATEADD(DAY, 30, orderdate);
```

Task 4: Write a SELECT Statement to Convert the Phone Number Information to an Integer Value

1. In the query pane, type the following query after the **Task 3** description:

```
SELECT
CONVERT(INT, REPLACE(REPLACE(REPLACE(REPLACE(phone, N'-', N''), N'(', ''), N')', ''), '
', '')) AS phonenoasint
FROM Sales.Customers;
```

   This query is trying to use the CONVERT function to convert phone numbers that include characters, such as hyphens and parentheses, into an integer value.

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

   Observe the error message:

```
Conversion failed when converting the nvarchar value '67.89.01.23' to data type int.
```

Because you want to retrieve rows without conversion errors and have a NULL for those that produce a conversion error, you can use the TRY_CONVERT function.

3. Modify the query to use the TRY_CONVERT function. The query should look like this:

```
SELECT
TRY_CONVERT(INT, REPLACE(REPLACE(REPLACE(REPLACE(phone, N'-', N''), N'(', ''), N')',
''), ' ', '')) AS phonenoasint
FROM Sales.Customers;
```

4. Highlight the written query, and click **Execute**. Observe the result. The rows that could not be converted have a NULL.

Result: After this exercise, you should be able to use conversion functions.

Exercise 2: Writing Queries That Use Logical Functions

Task 1: Write a SELECT Statement to Mark Specific Customers Based on Their Country and Contact Title

1. In Solution Explorer, double-click **61 - Lab Exercise 2.sql**.
2. In the query pane, highlight the statement **USE TSQL;**, and then click **Execute**.
3. In the query pane, type the following query after the **Task 1** description:

```
SELECT
IIF(country = N'Mexico' AND contacttitle = N'Owner', N'Target group', N'Other') AS
segmentgroup, custid, contactname
FROM Sales.Customers;
```

The IIF function was new in SQL Server 2012. It was added mainly to support migrations from Microsoft Access to SQL Server. You can use the CASE expression to achieve the same result.

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

Task 2: Modify the T-SQL Statement to Mark Different Customers

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

```
SELECT
IIF(contacttitle = N'Owner' OR region IS NOT NULL, N'Target group', N'Other') AS
```

```
segmentgroup,  custid,  contactname
FROM Sales.Customers;
```

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

Task 3: Create Four Groups of Customers

1. In the query pane, type the following query after the **Task 3** description:

```
SELECT CHOOSE(custid % 4 + 1, N'Group One', N'Group Two', N'Group Three', N'Group
Four') AS segmentgroup, custid, contactname
FROM Sales.Customers;
```

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

Result: After this exercise, you should know how to use the logical functions.

Exercise 3: Writing Queries That Test for Nullability

Task 1: Write a SELECT Statement to Retrieve the Customer Fax Information

1. In Solution Explorer, double-click the query **71 - Lab Exercise 3.sql**.
2. In the query pane, highlight the statement **USE TSQL;**, and click **Execute**.
3. In the query pane, type the following query after the **Task 1** description:

```
SELECT contactname, COALESCE(fax, N'No information') AS faxinformation
FROM Sales.Customers;
```

This query uses the COALESCE function to retrieve customers' fax information.

4. Highlight the written query, and click **Execute**.
5. In the query pane, type the following query after the previous query:

```
SELECT contactname, ISNULL(fax, N'No information') AS faxinformation
FROM Sales.Customers;
```

This query uses the ISNULL function. What is the difference between the ISNULL and COALESCE functions? COALESCE is a standard ANSI SQL function and ISNULL is not. So you should use the COALESCE function.

6. Highlight the written query, and click **Execute**.

Task 2: Write a Filter for a Variable That Could Be a Null

1. Highlight the query provided under the **Task 2** description, and click **Execute**.
2. Modify the query so that it looks like this:

```
DECLARE @region AS NVARCHAR(30) = NULL;

SELECT
custid, region
FROM Sales.Customers
WHERE region = @region OR (region IS NULL AND @region IS NULL);
```

3. Highlight the modified query, and click **Execute**.

Task 3: Write a SELECT Statement to Return All the Customers That Do Not Have a Two-Character Abbreviation for the Region

1. In the query pane, type the following query after the **Task 3** description:

```
SELECT custid, contactname, city, region
FROM Sales.Customers
WHERE
region IS NULL
OR LEN(region) <> 2;
```

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

Result: After this exercise, you should have an understanding of how to test for nullability.