LaunchCode T-SQL Workshop Class 2 Labs

Module 4

INFO for Lab 1

Expressions

```
--Concatenate strings
SELECT column1 + column2 AS NewColumn
FROM theTable;

--Math

SELECT 1 + 2, column1 * column2
FROM theTable;

--Cast an integer to a string
SELECT CAST(ID as varchar(10)) + Name AS CustID
FROM theTable;

SELECT CONVERT(varchar(10), ID)  + Name AS CustID
FROM theTable;

--See CAST and Convert article for styles
SELECT CONVERT(varchar(10),OrderDate,101) AS FormattedDate
FROM theTable;

--Replace NULL with an empty string

SELECT FirstName + ' ' + ISNULL(Middlename, '') + ' ' + LastName
FROM Names;
```

INFO for Lab 2

String functions

- RTRIM, LTRIM -- remove spaces
- LEFT, RIGHT -- return a number of characters
- LEN, DATALENGTH -- return the length
- CHARINDEX -- find a string
- SUBSTRING -- return part of a string
- REVERSE -- returns the string backwards
- UPPER, LOWER -- returns all upper or lower case
- REPLACE -- replace part of a string
- CONCATENATE – build a string and it takes care of NULL and conversions

INFO for Lab 3

Date functions

- GETDATE, SYSDATETIME -- returns the server date

- DATEADD -- adds a time period to a date
- DATEDIFF -- subtract a time period from a date
- DATENAME, DATEPART -- returns part of a date
- DAY, MONTH, YEAR -- returns part of a date
- CONVERT and FORMAT – formatting dates
- MM = months, mm = minutes (when using the Format function)

Lab 1: 10 minutes

1. Write a query using the Production.Product table displaying the Product ID, Color, and Name columns. Add a column called Description formatted as “Name: Color”. If the color is missing, substitute “N/A”.
2. Write a query using the Sales.SpecialOffer table. Display MinQty and MaxQty columns and the difference between them along with the SpecialOfferID and Description columns.

Lab 2: 15 minutes

1. Write a query displaying the BusinessEntityID from the Person.Person table. Also return the first and last names all in uppercase.
2. Return the first initial and last name from all the names in the Person.Person table.
3. Return the LastNames from Person.Person in spelled reverse order.

Lab 3: 15 minutes

1. Write a query that calculates the number of days between the date an order was placed and the date that it was shipped using the Sales.SalesOrdeHeader table. Include the SalesOrderID, OrderDate, and ShipDate columns. (Note that the answer is 7 for all the rows).
2. Write a query that displays only the date, not the time, for the OrderDate, DueDate, and ShipDate in the Sales.SalesOrderHeader table. Try to use a different function for each column.
3. Write a query that adds six months to each OrderDate in the Sales.SalesOrderHeader table. Include the SalesOrderID and OrderDate columns.

Module 5

INFO for Lab 1

Inner Join

Returns the columns from two tables where there is a match on a key. This can be one or more columns.

```
--These tables join on the same column name
SELECT SOH.SalesOrderID, SOH.OrderDate,
       SOD.ProductID, SOD.OrderQty, SOD.LineTotal
FROM Sales.SalesOrderHeader AS SOH
INNER JOIN Sales.SalesOrderDetail AS SOD
ON SOD.SalesOrderID = SOH.SalesOrderID;

--These table join on a different column name
SELECT Cust.CustomerID, Cust.StoreID,
       Pers.FirstName, Pers.LastName
FROM Sales.Customer AS Cust
INNER JOIN Person.Person AS Pers
       ON Cust.PersonID = Pers.BusinessEntityID;

--You can continue to bring in tables
SELECT SOH.SalesOrderID, SOH.OrderDate,
       SOD.ProductID, Prod.Name, SOD.OrderQty, SOD.LineTotal
FROM Sales.SalesOrderHeader AS SOH
INNER JOIN Sales.SalesOrderDetail AS SOD
ON SOD.SalesOrderID = SOH.SalesOrderID
```

```
INNER JOIN Production.Product AS Prod
ON Prod.ProductID = SOD.ProductID;
```

INFO for LAB 2

LEFT Outer Join

Returns the rows from the first table (LEFT) and include columns from the second table (RIGHT) even if there is not a match. The table on the right will return NULL for rows that don't match.

```
--return customers even if they don't have sales
--this is filtered so you can easily see the NULL results
SELECT c.CustomerID, s.SalesOrderID, s.OrderDate
FROM Sales.Customer AS c
LEFT OUTER JOIN Sales.SalesOrderHeader AS s ON c.CustomerID = s.CustomerID
WHERE c.CustomerID IN (11028,11029,1,2,3,4);
```

Add more tables to LEFT join

Continue LEFT once you start down that path

```
SELECT C.CustomerID, SOH.SalesOrderID, SOD.SalesOrderDetailID,
    SOD.ProductID, T.Name
FROM Sales.Customer AS C
LEFT OUTER JOIN Sales.SalesOrderHeader AS SOH ON C.CustomerID = SOH.CustomerID
LEFT OUTER JOIN Sales.SalesOrderDetail AS SOD ON SOH.SalesOrderID = SOD.SalesOrderID
LEFT OUTER JOIN Sales.SalesTerritory AS T ON C.TerritoryID = T.TerritoryID
WHERE C.CustomerID IN (11028,11029,1,2,3,4);
```

RIGHT Outer Join (reference, not needed for lab)

Returns rows from the RIGHT table and include columns from the LEFT table even if there is not a match. I recommend LEFT JOIN.

```
SELECT c.CustomerID, s.SalesOrderID, s.OrderDate
FROM  Sales.SalesOrderHeader AS s
RIGHT OUTER JOIN Sales.Customer AS c ON c.CustomerID = s.CustomerID
WHERE c.CustomerID IN (11028,11029,1,2,3,4);
```

FULL Outer Join (reference, not needed for lab)

This is rarely used. Return all the rows from both sides even if there is not a match. There are no tables in AdventureWorks that work, so run this code to create a table.

```
DROP TABLE IF EXISTS Production.ProductColor;
CREATE table Production.ProductColor
    (Color nvarchar(15) NOT NULL PRIMARY KEY);
GO
--Insert most of the existing colors
INSERT INTO Production.ProductColor
SELECT DISTINCT Color
FROM Production.Product
WHERE Color IS NOT NULL and Color <> 'Silver';
--Insert some additional colors
INSERT INTO Production.ProductColor
VALUES ('Green'),('Orange'),('Purple');

--Here is the query:
SELECT c.Color AS "Color from list", p.Color, p.ProductID
FROM Production.Product AS p
FULL OUTER JOIN Production.ProductColor AS c ON p.Color = c.Color
ORDER BY p.ProductID;
```

INFO for Lab 3

LEFT outer join with NULL filter on RIGHT table

```
--Find rows in Customer that do not exist in SalesOrderHeader
SELECT c.CustomerID, s.SalesOrderID, s.OrderDate
FROM Sales.Customer AS c
LEFT OUTER JOIN Sales.SalesOrderHeader AS s ON c.CustomerID = s.CustomerID
WHERE s.SalesOrderID IS NULL;
```

Lab 1: 15 minutes

1. The HumanResources.Employee table does not contain the employee name. Join that table to the Person.Person table on the BusinessEntityID column. Display the BusinessEntityID, first and last names, job title, and birth date.
2. Join the Production.Product table to the Production.ProductSubCategory table. Then join to the Production.ProductCategory. Return the ProductID. Return the Name of the product, subcategory and category. Be sure to use aliases. You'll use ProductSubCategoryID and ProductCategoryID as keys in this example.

Lab 2: 15 minutes

1. Write a query that displays all the products along with the SalesOrderID even if an order has never been placed for that product. Join to the Sales.SalesOrderDetail table using the ProductID column. Include the ProductID, name, SalesOrderID, and OrderQty.
2. Write a query that returns all the rows from the Sales.SalesOrderHeader table joined to the Sales.SalesPerson table (SalesPersonID to BusinessEntityID) even if no orders match. Include the SalesPersonID, SalesYTD and SalesOrderID columns in the results.

Lab 3: 15 minutes

1. Change the query written in Lab 2 Question 1 so that only products that have not been ordered show up in the results of the query.

Change the query in Lab 2 Question 2 so that the orders with no salesperson are returned.

Solutions

Module 4

Lab 1

```
--1
SELECT ProductID, Color, Name,
       Name + ': ' + ISNULL(Color,'N/A') AS Description
FROM Production.Product;

--Another option
SELECT ProductID, Color, Name,
       CONCAT(Name, ': ', COALESCE(Color,'N/A')) AS Description
FROM Production.Product;

--2
SELECT SpecialOfferID, Description, MinQty, MaxQty, MaxQty - MinQty AS Range
FROM Sales.SpecialOffer;
```

Lab 2

```
--1
SELECT BusinessEntityID, UPPER(FirstName) AS FirstName, UPPER(LastName) AS LastName
FROM Person.Person;

--2
SELECT LEFT(FirstName,1) AS FirstInitial, LastName
FROM Person.Person;

--3
SELECT REVERSE(LastName) AS ReverseName
FROM Person.Person;
```

Lab 3

```
--1
SELECT SalesOrderID, OrderDate, ShipDate,
       DATEDIFF(DAY,OrderDate,ShipDate) AS NoOfDays
FROM Sales.SalesOrderHeader;

--2
--Shows all the ways to do this
SELECT CAST(OrderDate AS DATE) AS OrderDate, CONVERT(VARCHAR(10),DueDate,20) AS DueDate,
       FORMAT(ShipDate,'yyyy-MM-dd') AS ShipDate
FROM sales.SalesOrderHeader;

--3
SELECT SalesOrderID, OrderDate, DATEADD(MONTH, 6, OrderDate) AS inSixMonths
FROM Sales.SalesOrderHeader;
```

Module 5

Lab 1

```
--1
SELECT Emp.BusinessEntityID, P.FirstName, P.LastName,
       Emp.JobTitle, Emp.BirthDate
FROM HumanResources.Employee AS EMP
INNER JOIN Person.Person AS P ON  P.BusinessEntityID = EMP.BusinessEntityID;

--2
SELECT Prod.ProductID, Prod.Name AS ProductName,
       sub.Name AS SubCategoryName, Prod.Name AS CategoryName
FROM Production.Product AS Prod
```

```
INNER JOIN Production.ProductSubcategory AS Sub ON Sub.ProductSubcategoryID = Prod.ProductSubcategoryID
INNER JOIN Production.ProductCategory AS Cat ON Cat.ProductCategoryID = Sub.ProductCategoryID;
```

Lab 2

```
--1
SELECT Prod.ProductID, Prod.Name, SOD.SalesOrderID, SOD.OrderQty
FROM Production.Product AS Prod
LEFT OUTER JOIN Sales.SalesOrderDetail AS SOD
       ON SOD.ProductID = Prod.ProductID;

--2
SELECT SOH.SalesPersonID, SP.SalesYTD, SOH.SalesOrderID
FROM Sales.SalesOrderHeader AS SOH
LEFT OUTER JOIN Sales.SalesPerson AS SP
       ON SOH.SalesPersonID = SP.BusinessEntityID;
```

Lab 3

```
--1
SELECT Prod.ProductID, Prod.Name, SOD.SalesOrderID, SOD.OrderQty
FROM Production.Product AS Prod
LEFT OUTER JOIN Sales.SalesOrderDetail AS SOD
       ON SOD.ProductID = Prod.ProductID
WHERE SOD.SalesOrderID IS NULL;

--2
SELECT SOH.SalesPersonID, SP.SalesYTD, SOH.SalesOrderID
FROM Sales.SalesOrderHeader AS SOH
LEFT OUTER JOIN Sales.SalesPerson AS SP
       ON SOH.SalesPersonID = SP.BusinessEntityID
WHERE SP.BusinessEntityID IS NULL;
```