LaunchCode T-SQL Workshop Day 2 Labs

Module 7: Subqueries and Common Table Expressions (CTE)

INFO for Lab 1

IN Subquery

Used in the WHERE clause. Must return one column.

```
--Returns customers who have orders
SELECT CustomerID, AccountNumber
FROM Sales.Customer
WHERE CustomerID IN (SELECT CustomerID FROM Sales.SalesOrderHeader);

--Returns customers who don't have orders
SELECT CustomerID, AccountNumber
FROM Sales.Customer
WHERE CustomerID NOT IN (SELECT CustomerID FROM Sales.SalesOrderHeader);
```

INFO for Lab 2

Correlated subquery

A subquery used in the SELECT list to include an aggregate or other single value. The subquery can see the main query.

```
SELECT CustomerID, SalesOrderID, OrderDate, TotalDue,
       (SELECT SUM(TotalDue)
       FROM Sales.SalesOrderHeader
       WHERE CustomerID = SOD.CustomerID) AS TotalCustSales,
       (SELECT SUM(TotalDue)
       FROM Sales.SalesOrderHeader) AS TotalSales
FROM Sales.SalesOrderHeader AS SOD
ORDER BY CustomerID;
```

INFO for Lab 3

Derived table (reference, not needed for lab)

A subquery used in place of a table in FROM. It can't see the main query. These are often nested.

```
SELECT SOH.CustomerID, SOH.SalesOrderID, SOH.OrderDate, SOH.TotalDue, TS.TotalSales
FROM Sales.SalesOrderHeader AS SOH
INNER JOIN (SELECT SUM(TotalDue) AS TotalSales, CustomerID
            FROM Sales.SalesOrderHeader
                    GROUP BY CustomerID) AS TS ON SOH.CustomerID = TS.CustomerID
ORDER BY SOH.CustomerID;
```

Common Table Expressions

Like a derived table, but is defined at the top of the query. Nesting not allowed, but you can base one CTE on a previous one.

```
WITH TS AS
      (SELECT SUM(TotalDue) AS TotalSales, CustomerID
```

```
        FROM Sales.SalesOrderHeader
        GROUP BY CustomerID)
SELECT SOH.CustomerID, SOH.SalesOrderID, SOH.OrderDate, SOH.TotalDue, TS.TotalSales
FROM Sales.SalesOrderHeader AS SOH
INNER JOIN TS ON SOH.CustomerID = TS.CustomerID
ORDER BY SOH.CustomerID;


WITH MonthTotal AS (
        SELECT SUM(TotalDue) AS MonthTotal, YEAR(OrderDate) AS OrderYear,
                MONTH(OrderDate) AS OrderMonth
        FROM Sales.SalesOrderHeader
        GROUP BY YEAR(OrderDate), MONTH(OrderDate)
        ),
YearTotal AS (
        SELECT SUM(MonthTotal) AS YearTotal, OrderYear
        FROM MonthTotal
        GROUP BY OrderYear
        ),
Sales AS (
        SELECT SalesOrderID, OrderDate, TotalDue,
                YEAR(OrderDate) AS OrderYear, MONTH(OrderDate) AS OrderMonth
        FROM Sales.SalesOrderHeader)
SELECT Sales.SalesOrderID, Sales.OrderDate, Sales.TotalDue,
        YearTotal.YearTotal, MonthTotal.MonthTotal
FROM Sales
INNER JOIN YearTotal ON YearTotal.OrderYear = Sales.OrderYear
INNER JOIN MonthTotal ON MonthTotal.OrderMonth = Sales.OrderMonth
        AND MonthTotal.OrderMonth = Sales.OrderMonth
ORDER BY MonthTotal.OrderYear, MonthTotal.OrderMonth;
```

Lab 1: 10 minutes

1. Using a subquery with the Sales.SalesOrderDetail table, display the product names and product ID numbers from the Production.Product table that have been ordered.

2. Change the query written in question 1 to display the products that have not been ordered.

Lab 2: 15 minutes

1. Write a query returning the list of products from Production.Product. Include the ProductID, Name, and Color. Using a correlated subquery, include the sum of OrderQty for each product from the Sales.SalesOrderDetail table. Filter the query to only return rows that have a FinishedGoodsFlag of 1.
2. Change the query you wrote in Lab 2 Question 1 so that the Average of OrderQty for each product is also included.

Lab 3: 15 minutes

1. Change the query you wrote in Lab 2 Question 1 to use a Common Table Expression (CTE) instead of the correlated subquery.
2. Change the query so that it also includes the Average calculation.

Module 8: UNION and related operators

Use the UNION operator to combine the results of two queries. The queries must have the same number of columns and compatible data types.

Info for Lab 1

UNION

```
--Union eliminates duplicates
SELECT BusinessEntityID AS ID
FROM HumanResources.Employee
UNION
SELECT BusinessEntityID
FROM Person.Person
UNION
SELECT SalesOrderID
FROM Sales.SalesOrderHeader
ORDER BY ID;
```

UNION ALL

```
--Union ALL includes duplicates
SELECT BusinessEntityID AS ID
FROM HumanResources.Employee
UNION ALL
SELECT BusinessEntityID
FROM Person.Person
UNION ALL
SELECT SalesOrderID
FROM Sales.SalesOrderHeader
ORDER BY ID;
```

EXCEPT

```
--Rows in first query but not in second
SELECT BusinessEntityID AS ID
FROM HumanResources.Employee
EXCEPT
SELECT BusinessEntityID
FROM Person.Person;
```

INTERSECT

```
--Rows in both queries
SELECT BusinessEntityID AS ID
FROM HumanResources.Employee
INTERSECT
SELECT BusinessEntityID
FROM Person.Person;
```

Lab 1

1. The Person.Person table contains names for several tables. Write the following queries and then use UNION ALL operator to return one result set. Each query should return the ID, role (a literal string), first name and last name.

Table	Key joining to BusinessEntityID	Role
HumanResources.Employee	BusinessEntityID	Employee
HumanResources.JobCandidate	BusinessEntityID	Job Candidate
Sales.Customer	PersonID	Customer
Sales.SalesPerson	BusinessEntityID	Salesperson

Solutions

Module 7

Lab 1

```
--1
SELECT Prod.ProductID, Prod.Name
FROM Production.Product AS Prod
WHERE Prod.ProductID IN (
       SELECT ProductID
       FROM Sales.SalesOrderDetail);

--2
SELECT Prod.ProductID, Prod.Name
FROM Production.Product AS Prod
WHERE Prod.ProductID NOT IN (
       SELECT ProductID
       FROM Sales.SalesOrderDetail);
```

Lab 2

```
--1
SELECT ProductID, Name, Color,
       (SELECT SUM(OrderQty) FROM
       Sales.SalesOrderDetail
       WHERE ProductID = Prod.ProductID) AS TotalQty
FROM Production.Product AS Prod
WHERE Prod.FinishedGoodsFlag =1;

--2
SELECT ProductID, Name, Color,
       (SELECT SUM(OrderQty) FROM
       Sales.SalesOrderDetail
       WHERE ProductID = Prod.ProductID) AS TotalQty,
       (SELECT AVG(OrderQty) FROM
       Sales.SalesOrderDetail
       WHERE ProductID = Prod.ProductID) AS AvgQty
FROM Production.Product AS Prod
WHERE Prod.FinishedGoodsFlag =1;
```

Lab 3

```
--1
WITH Sales AS (
       SELECT ProductID, SUM(OrderQty) AS TotalQty
       GROUP BY ProductID)
SELECT Prod.ProductID, Name, Color, TotalQty
FROM Production.Product AS Prod
INNER JOIN Sales ON Sales.ProductID = Prod.ProductID;


--2
WITH Sales AS (
       SELECT ProductID, SUM(OrderQty) AS TotalQty,
              AVG(OrderQty) AS AvgQty
       GROUP BY ProductID)
```

```
SELECT Prod.ProductID, Name, Color, TotalQty, AvgQty
FROM Production.Product AS Prod
INNER JOIN Sales ON Sales.ProductID = Prod.ProductID;
```

Module 8

Lab 1

```
SELECT Emp.BusinessEntityID, Pers.FirstName, Pers.LastName, 'Employee' AS Role
FROM Person.Person AS Pers
INNER JOIN HumanResources.Employee AS Emp
       ON Emp.BusinessEntityID = Pers.BusinessEntityID
UNION ALL
SELECT JC.BusinessEntityID, Pers.FirstName, Pers.LastName, 'Job Candidate'
FROM Person.Person AS Pers
INNER JOIN HumanResources.JobCandidate AS JC
       ON JC.BusinessEntityID = Pers.BusinessEntityID
UNION ALL
SELECT Cust.PersonID, Pers.FirstName, Pers.LastName, 'Customer'
FROM Person.Person AS Pers
INNER JOIN Sales.Customer AS Cust
       ON Cust.PersonID = Pers.BusinessEntityID
UNION ALL
SELECT SP.BusinessEntityID, Pers.FirstName, Pers.LastName, 'Salesperson'
FROM Person.Person AS Pers
INNER JOIN Sales.SalesPerson AS SP
       ON SP.BusinessEntityID = Pers.BusinessEntityID;
```