Lab 2-1

Connection values:

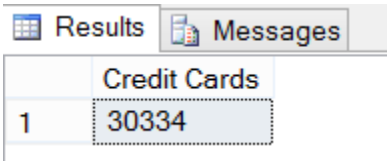
Server Type = Database Engine
Server Name = boyce.coe.neu.edu
Authentication = SQL Server Authentication
Login = INFO6210
Password = NEUHusky!

Note:
Two ways to specify comments in SQL commands:
Use -- for a line of comments
or use /* */ for a block of comments.

```sql
-- Get started with SQL

--Set the database context or open the database
USE AdventureWorks2008R2;
-- Or any version of AdventureWorks after it

--Get all columns about each product
--* means all columns from a table
--SELECT lists the columns that we want to retrieve
--FROM list where we are getting the data from

/*
In Production.Product, Production is the schema name,
Product is the table name. Schema is used for
organizing database objects or controlling access.
*/

SELECT *
FROM Production.Product;

--Get only specific columns about each product.
SELECT ProductID, Name, ProductNumber, ListPrice
FROM Production.Product;

--Get only specific columns about a specific product.
--WHERE does the filtering

SELECT ProductID, Name, ProductNumber, ListPrice
FROM Production.Product
WHERE Name = 'AWC Logo Cap'; /* Text literals are surrounded with
                                single quotes and appear in RED. */
```

```
/*
SSMS (SQL Server Management Studio) Query Editor
Window colors have special meanings:

Green: Comments
Black: All names and numeric literals
Blue: Key words
Red: Text literals
Pink: Function names
*/



/*
A table or column alias is an alternate name for a table or column.
An alias is used to give a table or column an abbreviated or a more
meaningful name. We'll learn more about aliases next.
*/


/*
Use an alias for:
(1) Creating a descriptive and meaningful column header
(2) Streamlining our SQL code
(3) Making several references to the same table in
    advanced SQL programming
*/
```

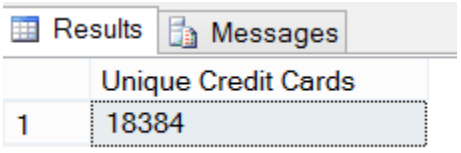
```sql
-- Set the database context
USE AdventureWorks2008R2;

--AGGREGATE FUNCTIONS

--Count number of credit cards in the Sales.SalesOrderHeader table.
--“Credit Cards” in the following example is an alias.

SELECT COUNT(CreditCardID) "Credit Cards"
FROM Sales.SalesOrderHeader;
```



Results | Messages

	Credit Cards
1	30334

```sql
/*
Count number of distinct values in the CreditCardID column of the
Sales.SalesOrderHeader.
The DISTINCT key word means unique.
*/

SELECT COUNT(DISTINCT CreditCardID) "Unique Credit Cards"
FROM Sales.SalesOrderHeader;
```



Results | Messages

	Unique Credit Cards
1	18384

```sql
/*
Demo many different aggregate functions and different ways
to specify column alias names.
*/

SELECT MIN(OrderQty) AS [Minimum Quantity], --Minimum Quantity is alias
       MAX(OrderQty) AS [Maximum Quantity],
       [Total] = SUM(OrderQty), -- Total is alias
       AVG(OrderQty) AverageQuantity, -- AverageQuantity is alias
       SUM(OrderQty)/COUNT(OrderQty), -- No column heading, no alias
       COUNT(*) as "Count" -- Alias using key word
FROM Sales.SalesOrderDetail
WHERE OrderQty BETWEEN 30 AND 50;
```

	Minimum Quantity	Maximum Quantity	Total	AverageQuantity	(No column name)	Count
1	30	44	1093	34	34	32

```
/* The COUNT function is commonly used together with GROUP BY. */


SELECT CustomerID, AccountNumber, COUNT(SalesOrderID) AS '# of Orders'
FROM AdventureWorks2008R2.Sales.SalesOrderHeader
GROUP BY CustomerID, AccountNumber
ORDER BY '# of Orders' DESC;
```

Results | Messages

	CustomerID	AccountNumber	# of Orders
1	11176	10-4030-011176	28
2	11091	10-4030-011091	28
3	11223	10-4030-011223	27
4	11276	10-4030-011276	27
5	11277	10-4030-011277	27
6	11331	10-4030-011331	27
7	11200	10-4030-011200	27
8	11262	10-4030-011262	27
9	11287	10-4030-011287	27
10	11300	10-4030-011300	27

```
/* When the COUNT is used in the SELECT clause, all other columns
   of the SELECT clause must be contained in the GROUP BY clause.
   If this syntax rule is not met, we'll get an error. */

SELECT CustomerID, AccountNumber, COUNT(SalesOrderID) AS '# of Orders'
FROM AdventureWorks2008R2.Sales.SalesOrderHeader
GROUP BY CustomerID
ORDER BY '# of Orders' DESC;
```

```
Msg 8120, Level 16, State 1, Line 5
Column 'AdventureWorks2008R2.Sales.SalesOrderHeader.AccountNumber' is invalid in the select
list because it is not contained in either an aggregate function or the GROUP BY clause.
```

/* All non-aggregated columns in the SELECT clause must be included in the GROUP BY clause. */

Useful Links

USE SQL Server Management Studio

http://msdn.microsoft.com/en-us/library/ms174173.aspx

Writing SQL Queries

http://technet.microsoft.com/en-us/library/bb264565(v=sql.90).aspx

SQL Aggregate Functions

http://msdn.microsoft.com/en-us/library/ms173454.aspx

Types of JOIN in SQL Server

http://www.codeproject.com/Tips/712941/Types-of-Join-in-SQL-Server

GROUP BY and HAVING
http://technet.microsoft.com/en-us/library/ms180199.aspx

Subquery Fundamentals

http://technet.microsoft.com/en-us/library/ms189575(v=sql.105).aspx