```
--*  BUSIT 103                    Assignment   #6                    DUE DATE :  Consult course calendar
/*
Name: Christopher Singleton
Class: BUSIT103 - Online
Instructor: Art Lovestedt
Date: 11/01/2014
*/
--You are to develop SQL statements for each task listed. You should type your SQL statements under
--each task. You are required to use INNER JOINs to solve each problem. Even if you know another method
--that will produce the result, this module is practice in INNER JOINs.

/*     Submit your .sql file named with your last name, first name and assignment # (e.g.,
GriggsDebiAssignment6.sql).
       Submit your file to the instructor using through the course site.  */

--NOTE: We are now using a different database.
USE AdventureWorks2012;

/*  Reminder: You are required to use the INNER JOIN syntax to solve each problem. INNER JOIN is ANSI syntax.
       It is generally considered more readable, especially when joining many tables. Even if you have prior
       experience with joins, you will still use the INNER JOIN syntax and refrain from using any OUTER or
       FULL joins in Modules 6 and 7 covering INNER JOINs. */

--1.a. (4) List any products that have product reviews.  Show product name, product ID, and comments.
--            Sort alphabetically on the product name. Don't over complicate this. A correctly written
--            INNER JOIN will return only those products that have product reviews; i.e., matching values in
--            the linking field. Hint:  Use the Production.Product and Production.ProductReview tables.

SELECT p.[Name] AS ProductName
       , p.[ProductID]
          , pr.[Comments] AS ProductReviewComments
FROM [Production].[Product] AS p
    INNER JOIN [Production].[ProductReview] AS pr
           ON p.ProductID = pr.ProductID
                ORDER BY [Name] ASC

--1.b. (2) Copy/paste 1.a. to 1.b. then modify 1.b. to show only records in which the word 'heavy' is
--            found in the Comments field. Show product ID, product name, and comments. Sort on ProductID.

SELECT p.[Name] AS ProductName
       , p.[ProductID]
          , pr.[Comments] AS ProductReviewComments
FROM [Production].[Product] AS p
    INNER JOIN [Production].[ProductReview] AS pr
           ON p.ProductID = pr.ProductID
                WHERE [Comments] LIKE '%heavy%'
                  ORDER BY p.ProductID

--2.a. (5) List product models with products. Show the product model ID, model name, product ID,
--            product name, standard cost, and class. Round all money values to exactly two decimal places.
--            Be sure to give any derived fields an alias. Order by standard cost from highest to lowest.
--            Hint: You know you need to use the Product table. Now look for a related table that contains
--            the information about the product model and inner join it to Product on the linking field.

SELECT pm.[ProductModelID]
     , pm.[Name] AS ProductModelName
        , p.[ProductID]
        , p.[Name] AS ProductName
        , CAST([StandardCost] AS DECIMAL(6,2)) AS [StandardCost_Rounded]
        , p.[Class] AS ProductClass
  FROM [Production].[Product] AS p
    INNER JOIN [Production].[ProductModel] AS pm
         ON p.ProductModelID = pm.ProductModelID
           ORDER BY [StandardCost] DESC
```

```sql
--2.b. (2) Copy/paste 2.a. to 2.b. then modify 2.b. to list only products with a value in the
--            class field. Do this using NULL appropriately in the WHERE clause. Hint: Remember
--            that nothing is ever equal (on not equal) to NULL; it either is or it isn't NULL.

SELECT pm.[ProductModelID] AS ProductModelID
     , pm.[Name] AS ProductModelName
        , p.[ProductID]
        , p.[Name] AS ProductName
        , CAST([StandardCost] AS DECIMAL(6,2)) AS [StandardCost_Rounded]
        , p.[Class] AS ProductClass
  FROM [Production].[Product] AS p
    INNER JOIN [Production].[ProductModel] AS pm
         ON p.ProductModelID = pm.ProductModelID
           WHERE [Class] IS NOT NULL
           ORDER BY [StandardCost] DESC

--2.c. (2) Copy/paste 2.b. to 2.c. then modify 2.c. to list only products that contain a value in
--            the class field AND contain 'fork' or 'front' in the product model name. Be sure that NULL
--            does not appear in the Class field by using parentheses appropriately.
/*SELECT [ProductModelID], [Name]
FROM [Production].[ProductModel]
WHERE [Name] LIKE '%Fork%' OR [Name] LIKE '%Front%'*/

SELECT pm.[ProductModelID] AS ProductModelID
     , pm.[Name] AS ProductModelName
        , [ProductID]
        , p.[Name] AS ProductName
        , CAST([StandardCost] AS DECIMAL(6,2)) AS [StandardCost_Rounded]
        , p.[Class] AS ProductClass
  FROM [Production].[Product] AS p
    INNER JOIN [Production].[ProductModel] AS pm
         ON p.ProductModelID = pm.ProductModelID
           WHERE (pm.[Name] LIKE '%Fork%' OR pm.[Name] LIKE '%Front%') AND p.[Class] IS NOT NULL
                ORDER BY [StandardCost] DESC


--3. a.       (6) List Product categories, their subcategories and their products.  Show the category name,
--            subcategory name, product ID, and product name, in this order. Sort in alphabetical order on
---           category name, subcategory name, and product name, in this order. Give each Name field a
--            descriptive alias. For example, the Name field in the Product table will have the alias
ProductName.
--            Hint:  To understand the relationships, create a database diagram with the following tables:
--            Production.ProductCategory
--            Production.ProductSubCategory
--            Production.Product

SELECT pc.[Name] AS ProductCategoryName
     , ps.[Name] AS ProductSubCategoryName
        , p.[ProductID]
        , p.[Name] AS ProductName
FROM [Production].[Product] AS p
     INNER JOIN [Production].[ProductSubcategory] AS ps
          ON p.ProductSubcategoryID = ps.ProductSubcategoryID
        INNER JOIN [Production].[ProductCategory] AS pc
          ON ps.ProductCategoryID = pc.ProductCategoryID
ORDER BY pc.[Name]
       , ps.[Name]
          , p.[Name] ASC
```

```
--3. b.       (3) Copy/paste 3.a. to 3.b. then modify 3.b. to list only Products in product category 1.
--            Show the category name, subcategory name, product ID, and product name, in this order. Sort in
--            alphabetical order on category name, subcategory name, and product name.
--            Hint: Add product category id field to SELECT clause, make sure your results are correct, then
--            remove or comment out the field.  Something to consider: Look at the data in the ProductName
field.
--            Could we find bikes by searching for 'bike' in the ProductName field?

SELECT pc.[Name] AS ProductCategoryName
     , ps.[Name] AS ProductSubCategoryName
        , p.[ProductID]
        , p.[Name] AS ProductName
FROM [Production].[Product] AS p
     INNER JOIN [Production].[ProductSubcategory] AS ps
          ON p.ProductSubcategoryID = ps.ProductSubcategoryID
        INNER JOIN [Production].[ProductCategory] AS pc
          ON ps.ProductCategoryID = pc.ProductCategoryID
         WHERE pc.[ProductCategoryID] = 1
           ORDER BY pc.[Name]
                  , ps.[Name]
                     , p.[Name] ASC

-- Products are actual Bikes.

--3. c.       (2) Copy/paste 3.b. to 3.c. then modify 3.c. to list Products in product category 3. Make no
other changes
--            to the statement. Consider what kinds of products are in category 3.

SELECT pc.[Name] AS ProductCategoryName
     , ps.[Name] AS ProductSubCategoryName
        , p.[ProductID]
        , p.[Name] AS ProductName
FROM [Production].[Product] AS p
     INNER JOIN [Production].[ProductSubcategory] AS ps
          ON p.ProductSubcategoryID = ps.ProductSubcategoryID
        INNER JOIN [Production].[ProductCategory] AS pc
          ON ps.ProductCategoryID = pc.ProductCategoryID
         WHERE pc.[ProductCategoryID] = 3
           ORDER BY pc.[Name]
                  , ps.[Name]
                     , p.[Name] ASC

--Clothing Products are in ProductCategoryName for ProductCategoryID 3.
```

```sql
--4.a. (5) List Product models, the categories, the subcategories, and the products.  Show the model name,
--            category name, subcategory name, product ID, and product name in this order. Give each Name
field a
--            descriptive alias. For example, the Name field in the ProductModel table will have the alias
ModelName.
--            Sort in alphabetical order by model name.
--            Hint:  To understand the relationships, create a database diagram with the following tables:
--            Production.ProductCategory
--            Production.ProductSubCategory
--            Production.Product
--            Production.ProductModel
--            Choose a path from one table to the next and follow it in a logical order to create the inner
joins

SELECT pm.[Name] AS ModelName
     , pc.[Name] AS CategoryName
     , ps.[Name] AS SubCategoryName
        , p.[ProductID]
        , p.[Name] AS ProductName
FROM [Production].[Product] AS p
     INNER JOIN [Production].[ProductSubcategory] AS ps
          ON p.ProductSubcategoryID = ps.ProductSubcategoryID
        INNER JOIN [Production].[ProductCategory] AS pc
          ON ps.ProductCategoryID = pc.ProductCategoryID
        INNER JOIN [Production].[ProductModel] AS pm
          ON p.ProductModelID = pm.ProductModelID
           ORDER BY pm.[Name] ASC

--4. b.       (3) Copy/paste 4.a. to 4.b. then modify 4.b. to list those products in model ID 23 and
--            contain black in the product name. Modify the sort to order only on Product ID. Hint: Add the
--            product model id field to the select clause to check your results and then remove or comment it
out.

SELECT pm.[Name] AS ProductModelName
     , pc.[Name] AS ProductCategoryName
     , ps.[Name] AS ProductSubCategoryName
        , p.[ProductID]
        , p.[Name] AS ProductName
  -- , pm.[ProductModelID] AS ProductModelID
FROM [Production].[Product] AS p
     INNER JOIN [Production].[ProductSubcategory] AS ps
          ON p.ProductSubcategoryID = ps.ProductSubcategoryID
        INNER JOIN [Production].[ProductCategory] AS pc
          ON ps.ProductCategoryID = pc.ProductCategoryID
        INNER JOIN [Production].[ProductModel] AS pm
          ON p.ProductModelID = pm.ProductModelID
          WHERE (pm.[ProductModelID] = 23) AND p.[Name] LIKE '%Black%'
           ORDER BY p.[ProductID] ASC
```

```sql
--5.   a. (4) List products ordered by customer id 19670.  Show product name, product number, order quantity,
--            and sales order id.  Sort on product name and sales order id.  If you add customer id to check
your results,
--            be sure to remove or comment it out. Hint:  First create a database diagram with the following
tables:
--            Production.Product
--            Sales.SalesOrderHeader
--            Sales.SalesOrderDetail

SELECT p.[Name] AS ProductName
     , p.[ProductNumber]
        , od.[OrderQty]
        , od.[SalesOrderID]
  -- , oh.[CustomerID]
FROM [Production].[Product] AS p
    INNER JOIN [Sales].[SalesOrderDetail] AS od ON p.[ProductID] = od.[ProductID]
    INNER JOIN [Sales].[SalesOrderHeader] AS oh ON od.[SalesOrderID] = oh.[SalesOrderID]
WHERE oh.[CustomerID] = 19670
ORDER BY p.Name, od.SalesOrderID


--5. b. (4) List the orders and the shipping method for customer id 19670.  We are only concerned with orders
--            that have a shipping method, so don't screen for NULL. Show product name, product number, order
quantity,
--            sales order id, and the name of the shipping method. Sort on product name and sales order id.
--            HINT:  You will need to join an additional table. Add it to your database diagram first.
/* NOTES: TABLES NEEDED:
     p.[Production].[Product]
       od.[Sales].[SalesOrderDetail]
       oh.[Sales].[SalesOrderHeader]
    sm.[Purchasing].[ShipMethod]
*/

SELECT p.[Name] AS ProductName
     , p.[ProductNumber]
        , od.[OrderQty]
        , od.[SalesOrderID]
        , sm.[Name] AS ShippingMethod
  -- , oh.[CustomerID]
FROM [Production].[Product] AS p
    INNER JOIN [Sales].[SalesOrderDetail] AS od
          ON p.[ProductID] = od.[ProductID]
    INNER JOIN [Sales].[SalesOrderHeader] AS oh
          ON od.[SalesOrderID] = oh.[SalesOrderID]
       INNER JOIN [Purchasing].[ShipMethod] AS sm
          ON sm.[ShipMethodID] = oh.[ShipMethodID]
WHERE oh.[CustomerID] = 19670
ORDER BY p.[Name], od.[SalesOrderID]
```

```sql
--6.   (6) List all sales for clothing that were ordered during 2007.  Show sales order id, product ID,
--            product name, order quantity, and line total for each line item sale. Make certain you are
--            retrieving only clothing. There are multiple ways to find clothing. Show the results
--            by sales order id and product name. Hint: Refer to the diagram you created in #5.
--      [Production].[Product]
--      [Sales].[SalesOrderDetail]
--      [Sales].[SalesOrderHeader]
--      [Production].[ProductCategory]
--      [Production].[ProductSubcategory]

SELECT oh.[SalesOrderID]
      , p.[ProductID]
         , p.[Name] AS ProductName
         , od.[OrderQty]
      , CAST(od.[LineTotal] AS DECIMAL(6,2)) AS [LineTotal_Rounded] --Decimal two places. (Look's better
rounded)
--    , od.[LineTotal] AS OrderDetailLineTotal
--    , oh.[OrderDate] AS OrderHeaderOrderDate
--    , CAST(oh.[OrderDate] AS DATE) AS [OrderDate]   --Only in 2007 OrderDate Without Time. (Looks better
without time)
FROM [Production].[Product] AS p
      INNER JOIN [Sales].[SalesOrderDetail] AS od
            ON p.[ProductID] = od.[ProductID]
      INNER JOIN [Sales].[SalesOrderHeader] AS oh
            ON od.[SalesOrderID] = oh.[SalesOrderID]
         INNER JOIN [Production].[ProductSubcategory] AS ps
            ON p.ProductSubcategoryID = ps.ProductSubcategoryID
         INNER JOIN [Production].[ProductCategory] AS pc
            ON ps.ProductCategoryID = pc.ProductCategoryID
                 WHERE pc.[ProductCategoryID] = 3  AND  YEAR(oh.[OrderDate]) = 2007

/*NOTES: Product.ProductCategoryID Number 3 is Clothing.
--          SELECT *
--          FROM [Production].[ProductCategory]
*/
```

```
--7.   (2) In your own words, write a business question for AdventureWorks that you will try to answer with a
SQL query.
--            Then try to develop the SQL to answer the question using at least one INNER JOIN.
--            You may find that the AdventureWorks database structure is highly normalized and therefore,
difficult to work
--            with.  As a result, you may not run into difficulties when developing your SQL.  For this task
that is fine.
--            Just show your question and as much SQL as you were able to figure out.

/*  How many of each type of bicycle were sold in December of the year 2007 and
on what dates throughout the month were sales?
What was the total cost of sales of each bicycle name for December 2007? Round all dollar values to two
decimal places
and list the total cost of shipping, Tax Amount, Sales Sub Total for each type of bike. Sort by TotalDue */
--  TABLES:     [Production].[Product]
--              [Production].[ProductCategory]
--              [Production].[ProductSubcategory]
--              [Sales].[SalesOrderDetail]
--              [Sales].[SalesOrderHeader]


SELECT p.[Name] AS ProductName
      ,CAST(oh.[OrderDate] AS DATE) AS OrderDate                --CAST Date (Do Away with Time)
      ,ps.[ProductSubcategoryID]
         ,pc.[Name] AS ProductCategoryName
         ,od.[OrderQty] AS TotalOrderQuantity
         ,CAST(oh.[Freight] AS DECIMAL(6,2)) AS FreightCost        --CAST to two Decimal Places.
         ,CAST(oh.[TaxAmt] AS DECIMAL(6,2)) AS TaxAmount           --CAST to two Decimal Places.
         ,CAST(oh.[SubTotal] AS DECIMAL(7,2)) AS SalesSubTotal     --CAST to two Decimal Places.
         ,CAST(oh.[TotalDue] AS DECIMAL(8,2)) AS SalesTotalDue     --CAST to two Decimal Places.
FROM [Production].[Product] AS p                                  --Proction.Product Table Alias
      INNER JOIN [Sales].[SalesOrderDetail] AS od                 --INNER JOIN Sales.SalesOrderDetail Table.
            ON p.[ProductID] = od.[ProductID]                        --Common Key in both tables.
      INNER JOIN [Sales].[SalesOrderHeader] AS oh                 --INNER JOIN Sales.SalesOrderHeader Table.
            ON od.[SalesOrderID] = oh.[SalesOrderID]                 --Common Key in both tables.
         INNER JOIN [Production].[ProductSubCategory] AS ps         --INNER JOIN Production.ProductSubCategory
Table.
            ON p.ProductSubcategoryID = ps.ProductSubCategoryID      --Common Key in both tables.
         INNER JOIN [Production].[ProductCategory] AS pc            --INNER JOIN Production.ProductCategory
Table.
            ON ps.ProductCategoryID = pc.ProductCategoryID           --Common Key in both tables.
WHERE ps.[ProductSubcategoryID] IN (1, 2, 3)                    --Use the IN Keyword to filter all Bike
Categories.
      AND oh.[OrderDate] BETWEEN '2007-12-01' AND '2007-12-31'  --December 2007 (One Whole Month Only)
ORDER BY CAST(oh.[TotalDue] AS DECIMAL(8,2)) DESC                --Sort by SalesTotalDue highest to lowest.
```

```sql
/* Next: Experiment using the Aggregate function by using the GROUP BY Keyword.  (Final Outcome)*/

SELECT p.[Name] AS ProductName
      ,CAST(oh.[OrderDate] AS DATE) AS OrderDate                --CAST Date (Do Away with Time Function)
      ,ps.[ProductSubcategoryID]
         ,pc.[Name] AS ProductCategoryName
         ,SUM(od.[OrderQty]) AS TotalOrderQuantity                 --Aggregate Funcion, SUM (Combine)
OrderQty.
         ,CAST(oh.[Freight] AS DECIMAL(6,2)) AS FreightCost        --CAST to two Decimal Places.
         ,CAST(oh.[TaxAmt] AS DECIMAL(6,2)) AS TaxAmount           --CAST to two Decimal Places.
   -- ,CAST(SUM(od.[LineTotal]) AS DECIMAL(8,2)) AS LineTotal    -- (This is for checking purposes)
         ,CAST(oh.[SubTotal] AS DECIMAL(7,2)) AS SalesSubTotal     --CAST to two Decimal Places.
         ,CAST(oh.[TotalDue] AS DECIMAL(8,2)) AS SalesTotalDue     --CAST to two Decimal Places.
FROM [Production].[Product] AS p
      INNER JOIN [Sales].[SalesOrderDetail] AS od               --INNER JOIN Sales.SalesOrderDetail Table.
            ON p.[ProductID] = od.[ProductID]                      --Common Key in both tables.
      INNER JOIN [Sales].[SalesOrderHeader] AS oh               --INNER JOIN Sales.SalesOrderHeader Table.
            ON od.[SalesOrderID] = oh.[SalesOrderID]               --Common Key in both tables.
         INNER JOIN [Production].[ProductSubCategory] AS ps        --INNER JOIN Production.ProductSubCategory
Table.
            ON p.ProductSubcategoryID = ps.ProductSubCategoryID    --Common Key in both tables.
         INNER JOIN [Production].[ProductCategory] AS pc           --INNER JOIN Production.ProductCategory
Table.
            ON ps.ProductCategoryID = pc.ProductCategoryID         --Common Key in both tables.
WHERE (ps.[ProductSubcategoryID] IN (1, 2, 3))                  --Use the IN Keyword to filter Bike
Categories.
      AND oh.[OrderDate]  BETWEEN '2007-12-01' AND '2007-12-31'  --December 2007 (One Whole Month Only)
GROUP BY p.[Name]                                               --Aggregate function grouping elements.
      , oh.[OrderDate]
         , ps.[ProductSubcategoryID]
         , pc.[Name], oh.[Freight]
         , oh.[TaxAmt]
         , oh.[SubTotal]
         , oh.[TotalDue]
ORDER BY CAST(oh.[TotalDue] AS DECIMAL(8,2)) DESC                --Sort by SalesTotalDue highest to lowest.
```