```sql
--*  BUSIT 103                                Assignment   #7                                                 DUE DATE :  Consult
course calendar
/*
Name: Christopher Singleton
Class: BUSIT103 - Online
Instructor: Art Lovestedt
Date: 11/08/2014
*/

/*     You are to develop SQL statements for each task listed. You should type your SQL statements
       under each task. You are required to use the INNER JOIN syntax to solve each problem. INNER JOIN is
       ANSI syntax. 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 introducing 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.  */

/*     Ideas for consideration: Run the statement in stages: Write the SELECT and FROM clauses first
       and run the statement. Add the ORDER BY clause. Then add the WHERE clause; if it is a compound
       WHERE clause, add piece at a time. Lastly perform the CAST or CONVERT. When the statement is
       created in steps, it is easier to isolate the error. Check the number of records returned
       to see if it makes sense.*/

/*  When there are multiple versions of a field, such as EnglishCountryRegionName,
       SpanishCountryRegionName, FrenchCountryRegionName, use the English version of the field.*/

--     Do not remove the USE statement

USE AdventureWorksDW2012;

--1.a. (4) List the names and locations of AdventureWorks customers who are male.
--            Show customer key, first name, last name, state name, and country name. Order the list
--            by country name, state name, last name, and first name in alphabetical order.
--            Check your results. Did you get 9,351 records?
-- [dbo].[DimCustomer]
--[dbo].[DimGeography]

SELECT dc.[CustomerKey]
     , dc.[FirstName]
        , dc.[LastName]
        , dg.[StateProvinceName]
        , dg.[EnglishCountryRegionName]
FROM [dbo].[DimCustomer] AS dc
     INNER JOIN [DimGeography] AS dg ON dg.[GeographyKey] = dc.[GeographyKey]
WHERE dc.[Gender] = 'M'
ORDER BY dg.[EnglishCountryRegionName]
           ,dg.[StateProvinceName]
              ,dc.[LastName]
              ,dc.[FirstName] ASC

--1.b. (2) Copy/paste the statement from 1.a to 1.b. Modify the WHERE clause in 1.b to show only
--            those AdventureWorks customers who are female and from the US City of Birmingham.
--            Show customer key, first name, last name, and city name.
--            Change the sort order to list by last name, then first name in alphabetical order.

SELECT dc.[CustomerKey], dc.[FirstName], dc.[LastName], dg.[City]
FROM [dbo].[DimCustomer] AS dc
     INNER JOIN [DimGeography] AS dg ON dg.[GeographyKey] = dc.[GeographyKey]
WHERE dc.[Gender] = 'F' AND City = 'Birmingham'
ORDER BY dc.[LastName], dc.[FirstName] ASC --(ASC is default, but put it in anyways.)
```

```sql
--1.c. (2) Copy/paste statement from 1.b to 1.c. Modify the WHERE clause in 1.c to list only
--            AdventureWorks customers from the US city of Seattle who are female and have 2 or more cars.
--            Show customer key, first name, last name, and total number of cars.
--            Order the list by number of cars in descending order, then by last name and first name
--            in alphabetical order.

SELECT dc.[CustomerKey], dc.[FirstName], dc.[LastName], dc.[NumberCarsOwned]
FROM [dbo].[DimCustomer] AS dc
     INNER JOIN [DimGeography] AS dg ON dg.[GeographyKey] = dc.[GeographyKey]
WHERE dg.[City] = 'Seattle' AND dc.[Gender] = 'F' AND dc.[NumberCarsOwned] >= 2
ORDER BY dc.[NumberCarsOwned] DESC, dc.[LastName], dc.[FirstName] --Default is ASC

--2.a. (2) Explore the data warehouse using ONLY the DimProduct table. No joins required.
--            Show the English product name, product key, product alternate key, standard cost, list price,
--            and status. Sort on English product name. Notice that some of the products appear to be
duplicates.
--            The name and the alternate key remain the same but the product is added again with a new product
--            key to track the history of changes to the product attributes. For example, look at AWC Logo
Cap.
--            Notice the history of changes to StandardCost and ListPrice and to the value in the Status
field.

SELECT [EnglishProductName]
     , [ProductKey]
        , [ProductAlternateKey]
        , [StandardCost]
        , [ListPrice]
        , [Status]
FROM [dbo].[DimProduct]
ORDER BY [EnglishProductName]

--2.b. Write two SELECT statements (no joins required) using the DimProduct table and write down
--            the row count returned when you run the statement in the place below where you see "List row
count..."
--            1. Show the product key, English product name, and product alternate key for each product only
once.
--            Sort on English product name.
--            2. Show the English product name and product alternate key for each product only once. Sort on
English product
--            name. Recall terms like “only once”, “one time”, and "unique" all indicate the need for the
DISTINCT keyword.
--            (1) List row count for the results set for 1.
--            (2) List row count for the results set for 2.

--(1)
SELECT DISTINCT [ProductKey], [EnglishProductName], [ProductAlternateKey]
FROM [dbo].[DimProduct]
ORDER BY [EnglishProductName]
--"List row count = 606 rows"

--(2)
SELECT DISTINCT [EnglishProductName], [ProductAlternateKey]
FROM [dbo].[DimProduct]
ORDER BY [EnglishProductName]

--"List row count = 504 rows"

--(Notes: For my reference: the reason why (2) has less rows than (1) is becasue One Product can have multiple
historical records in DimProduct
--      with different Productkey (unique). Since (1) extracts the ProductKey, there will be more
rows/records.)
```

```
--2.c. (4) Join tables to the product table to also show the category and subcategory name for each product.
--            Show the English category name, English subcategory name, English product name, and product
alternate key
--            only once. Sort the results by the English category name, English subcategory name,
--            and English product name. The record count will decrease to 295. Some products in the product
--            table are inventory and not for sale. They don't have a value in the ProductSubcategory field
and
--            are removed from the results set by the INNER JOIN. We will explore this more in OUTER JOINs.

SELECT DISTINCT pc.[EnglishProductCategoryName]
               ,ps.[EnglishProductSubcategoryName]
                        ,p.[EnglishProductName]
                        ,p.[ProductAlternateKey]
FROM [dbo].[DimProduct] AS p
     INNER JOIN [dbo].[DimProductSubcategory] AS ps ON p.[ProductSubcategoryKey] = ps.[ProductSubcategoryKey]
     INNER JOIN [dbo].[DimProductCategory] AS pc ON ps.[ProductCategoryKey] = pc.[ProductCategoryKey]
ORDER BY pc.[EnglishProductCategoryName]
           ,ps.[EnglishProductSubcategoryName]
              ,p.[EnglishProductName]
--"row count 295"


--3.a. (5) List the English name for products purchased over the Internet by customers who indicate education
--            as high school or partial high school. Show Product key and English Product Name and English
Education.
--            Order the list by English Product name. Show a product only once even if it has been purchased
several times.
--            We are not interested in the customer names because we are looking only at the broad
demographics of buyers.

SELECT DISTINCT p.[ProductKey]
              , p.[EnglishProductName]
                       , c.[EnglishEducation]
FROM [dbo].[DimProduct] AS p
     INNER JOIN [dbo].[FactInternetSales] AS fis ON p.[ProductKey] = fis.[ProductKey]
     INNER JOIN  [dbo].[DimCustomer] AS c ON fis.[CustomerKey] = c.[CustomerKey]
WHERE c.[EnglishEducation] IN ('High School', 'Partial High School')
ORDER BY p.[EnglishProductName]

--3.b. (3) List the English name for products purchased over the Internet by customers who indicate
--            high school or partial high school, or partial college. Show Product key and English Product
Name
--            and English Education. Order the list by English Product name and then by English Education.
--            Show a product only once even if it has been purchased several times.

SELECT DISTINCT p.[ProductKey]
               ,p.[EnglishProductName]
                        ,c.[EnglishEducation]
FROM [dbo].[DimProduct] AS p
     INNER JOIN [dbo].[FactInternetSales] AS fis ON p.[ProductKey] = fis.[ProductKey]
     INNER JOIN  [dbo].[DimCustomer] AS c ON fis.[CustomerKey] = c.[CustomerKey]
WHERE c.[EnglishEducation] IN ('High School', 'Partial High School', 'Partial College')
ORDER BY p.[EnglishProductName]
           ,c.[EnglishEducation]
```

```sql
--4.   (5) List the English name for products purchased over the Internet by customers who work in clerical,
--            manual, or skilled manual occupations. Show Product key and English Product Name and English
Occupation.
--            Add a meaningful sort. Show a product only once even if it has been purchased several times.

SELECT DISTINCT p.[ProductKey]
               ,p.[EnglishProductName]
                        ,c.[EnglishOccupation]
FROM [dbo].[DimProduct] AS p
     INNER JOIN [dbo].[FactInternetSales] AS fis ON p.[ProductKey] = fis.[ProductKey]
     INNER JOIN  [dbo].[DimCustomer] AS c ON fis.[CustomerKey] = c.[CustomerKey]
WHERE c.[EnglishOccupation] IN ('Clerical', 'Manual', 'Skilled Manual')
ORDER BY p.[EnglishProductName], c.[EnglishOccupation]


--     Question 5 contains exploratory questions. You may wish to read all three questions before beginning.
--     Seeing the purpose of the questions may help understand the requests.

--5.a. (6) List customers who have purchased clothing over the Internet.  Show customer first name,
--            last name, and English product category. If a customer has purchased clothing items more than
once,
--            show only one row for that customer. This means that the customer should not appear twice.
--            Order the list by last name, then first name. Did you return 6,839 records in your results set?

SELECT DISTINCT c.[FirstName]
               ,c.[LastName]
                        ,pc.[ProductCategoryKey]
FROM [dbo].[DimProduct] AS p
     INNER JOIN [dbo].[FactInternetSales] AS fis ON p.[ProductKey] = fis.[ProductKey]
     INNER JOIN  [dbo].[DimCustomer] AS c ON fis.[CustomerKey] = c.[CustomerKey]
     INNER JOIN [dbo].[DimProductSubcategory] AS ps ON p.[ProductSubcategoryKey] = ps.[ProductSubcategoryKey]
     INNER JOIN [dbo].[DimProductCategory] AS pc ON ps.[ProductCategoryKey] = pc.[ProductCategoryKey]
WHERE pc.[ProductCategoryKey] = 3
ORDER BY c.[LastName], c.[FirstName]
--6,839 records processed

--5.b. (2) Copy/paste 5.a to 5.b and modify 5.b.  Show customer key, first name, last name, and English
--            product category. If a customer has purchased clothing more than once, show only one row for
that customer.
--            This means that the customer should not appear twice. Order the list by last name, then first
name.

SELECT DISTINCT c.[CustomerKey]
              , c.[FirstName]
              , c.[LastName]
                , pc.[EnglishProductCategoryName]
FROM [dbo].[DimProduct] AS p
     INNER JOIN [dbo].[FactInternetSales] AS fis ON p.[ProductKey] = fis.[ProductKey]
     INNER JOIN  [dbo].[DimCustomer] AS c ON fis.[CustomerKey] = c.[CustomerKey]
     INNER JOIN [dbo].[DimProductSubcategory] AS ps ON p.[ProductSubcategoryKey] = ps.[ProductSubcategoryKey]
     INNER JOIN [dbo].[DimProductCategory] AS pc ON ps.[ProductCategoryKey] = pc.[ProductCategoryKey]
WHERE pc.[ProductCategoryKey] = 3
ORDER BY c.[LastName], c.[FirstName]
--6,852 records processed
```

```sql
--5.c. BONUS +2 Why is there a difference between the number of records received in 5a and 5b?
--            To be eligible for the bonus your answer MUST show as a COMMENT and it must be detailed
--            enough to show that you understand why customers are "missing" from 5a. Be brief.
--            This is actually a simple answer.

--This will count any of the CustomerKey showing More than 1 CustomerKey (Becuse of Multiple Sales Line
Items).
SELECT DISTINCT c.[FirstName]
     , c.[LastName]
     , pc.[EnglishProductCategoryName]
        ,COUNT(c.[CustomerKey]) AS CustomerKeyCount
FROM [dbo].[DimProduct] AS p
  INNER JOIN [dbo].[FactInternetSales] AS fis ON p.[ProductKey] = fis.[ProductKey]
  INNER JOIN  [dbo].[DimCustomer] AS c ON fis.[CustomerKey] = c.[CustomerKey]
  INNER JOIN [dbo].[DimProductSubcategory] AS ps ON p.[ProductSubcategoryKey] = ps.[ProductSubcategoryKey]
  INNER JOIN [dbo].[DimProductCategory] AS pc ON ps.[ProductCategoryKey] = pc.[ProductCategoryKey]
   WHERE pc.[ProductCategoryKey] = 3
     GROUP BY c.[FirstName]
            , c.[LastName]
           , pc.[EnglishProductCategoryName]
       HAVING COUNT(c.[CustomerKey]) > 1
         ORDER BY c.[FirstName], c.[LastName]
--Answer:
--The reason that b has more records is because that the FactInternetSales table has multiple sales line items
--for each customerkey.
--Checking Customers...
/*
SELECT * FROM DimCustomer c
     INNER JOIN [dbo].[FactInternetSales] f ON f.CustomerKey = c.CustomerKey
WHERE FirstName = 'Edward' AND LastName = 'Alexander'

SELECT * FROM DimCustomer c
     INNER JOIN [dbo].[FactInternetSales] f ON f.CustomerKey = c.CustomerKey
WHERE FirstName = 'Kyle' AND LastName = 'Adams'

SELECT * FROM [dbo].[FactInternetSales]
WHERE CustomerKey = 18812
*/

--6,852 records processed in 5b.

 --Because some Customers have two Customerkey's. Even though DISTINCT is in the clause,
 --showing CustomerKey makes the customer that has more than one Sales Line Item show more than once.
(CustomerKey only shows once)
 -- Alexandria Bryant, Abigail Simmons, Jordan Adams

--6.   (6) List all Internet sales for accessories that occurred during 2008 (Order Date in 2008).
--            Show Order date, product key, product name, and sales amount for each line item sale.
--            Show the date as mm/dd/yyyy as DateOfOrder. Use CONVERT and look up the style code.
--            Show the list in oldest to newest order by date and alphabetically by product name.
--            21,067 Rows

SELECT CONVERT(VARCHAR (10), fis.[OrderDate], 101) AS DateOfOrder
      ,p.[ProductKey]
         ,p.[EnglishProductName]
         ,fis.[SalesAmount]
FROM [dbo].[DimProduct] AS p
     INNER JOIN [dbo].[FactInternetSales] AS fis ON p.[ProductKey] = fis.[ProductKey]
     INNER JOIN [dbo].[DimProductSubcategory] AS ps ON p.[ProductSubcategoryKey] = ps.[ProductSubcategoryKey]
     INNER JOIN [dbo].[DimProductCategory] AS pc ON ps.[ProductCategoryKey] = pc.[ProductCategoryKey]
WHERE pc.[ProductCategoryKey] = 4 AND YEAR(fis.[OrderDate]) = 2008
ORDER BY fis.[OrderDate], p.[EnglishProductName]
```

```
--7.   (5) List all Internet sales of Accessories to customers in Paris, France during 2008.
--            Show product key, product name, order date as mm/dd/yyyy, SalesAmount, and City for each line
item sale.
--            Show the list in order alphabetically by product name. If you add fields to verify the results
--            be sure to remove or comment out the fields before submitting the homework.

SELECT  p.[ProductKey]
          ,p.[EnglishProductName]
          ,CONVERT(VARCHAR (10), fis.[OrderDate], 101)  AS DateOfOrder
          ,fis.[SalesAmount]
          ,dg.[City]
FROM [dbo].[DimProduct] AS p
     INNER JOIN [dbo].[FactInternetSales] AS fis ON p.[ProductKey] = fis.[ProductKey]
     INNER JOIN  [dbo].[DimCustomer] AS c ON fis.[CustomerKey] = c.[CustomerKey]
     INNER JOIN [dbo].[DimGeography] AS dg ON c.[GeographyKey] = dg.[GeographyKey]
     INNER JOIN [dbo].[DimProductSubcategory] AS ps ON p.[ProductSubcategoryKey] = ps.[ProductSubcategoryKey]
     INNER JOIN [dbo].[DimProductCategory] AS pc ON ps.[ProductCategoryKey] = pc.[ProductCategoryKey]
WHERE pc.[ProductCategoryKey] = 4
      AND dg.[City] = 'Paris'
         AND EnglishCountryRegionName = 'France'
         AND YEAR(fis.[OrderDate]) = 2008
ORDER BY p.[EnglishProductName]

--8.   (2) In your own words, write a business question that you can answer by querying the data warehouse.
--            Then write the SQL query using an INNER JOIN that will provide the information that you are
seeking.
--            Try it. You get credit for writing a question and trying to answer it.

--Question:
--I want to know top 5 customers with highest sales amount on the internet in December of 2007,
--what procucts that they purchased, how much they paid for the items and provide the gender, address, city,
phone number
--and country where they reside and if they are married. Sort by amount purchased.

SELECT TOP 5
       c.[CustomerKey]
      ,c.[FirstName]
         ,ISNULL(c.[MiddleName],'')
         ,c.[LastName]
         ,c.[Phone]
         ,c.[MaritalStatus]
         ,c.[Gender]
         ,dg.[City]
         ,dg.[StateProvinceName]
         ,dg.[EnglishCountryRegionName]
         ,SUM(fis.[SalesAmount]) AS SalesAmount
FROM [dbo].[DimProduct] AS p
        INNER JOIN [dbo].[FactInternetSales] AS fis ON p.[ProductKey] = fis.[ProductKey]
     INNER JOIN  [dbo].[DimCustomer] AS c ON fis.[CustomerKey] = c.[CustomerKey]
     INNER JOIN [dbo].[DimGeography] AS dg ON c.[GeographyKey] = dg.[GeographyKey]
WHERE MONTH(fis.[OrderDate]) = 12 AND YEAR(fis.[OrderDate]) = 2007
GROUP BY c.CustomerKey,c.[FirstName]
           ,ISNULL(c.[MiddleName],'')
           ,c.[LastName]
           ,c.[Phone]
           ,c.[MaritalStatus]
           ,c.[Gender]
           ,dg.[City]
           ,dg.[StateProvinceName]
           ,dg.[EnglishCountryRegionName]
ORDER BY fis.SalesAmount DESC
```

```sql
--Checking for cutomer 12650 (one of the top 5)
--Total Sales Amount: 7235.36

--Checking:
/*
SELECT CustomerKey, SUM(SalesAmount) FROM [dbo].[FactInternetSales]
WHERE CustomerKey = 12650 AND (YEAR(OrderDate) = 2007 AND MONTH(OrderDate) = 12)
GROUP BY CustomerKey
*/
/*
Nice job on this Chris! A few quick comments: - 1b and 1c should also filter for EnglishCountryRegionName =
'United States'; I realize there isn't another "Seattle", but there are cities named "Birmingham" in other
countries. - Excellent use of "IN" in the WHERE clauses - 5c; +2 - it is possible that two different people
have the same names as well (versus the same person being in the system twice).
Art Lovestedt, Nov 18 at 10:02pm
*/
```