# Data Translation Challenge - Technical Report

**Erin Ballar**

## About the AdventureWorksDW2017 Data Warehouse

The Adventure Works data warehouse is one of Microsoft's sample databases which contains data about a fictitious sporting gear company's products, consumers, sales, and more.

You may notice that many of these table names have a 'Fact' or 'Dim' in front of it. This is due to the data warehouse's dimensional design. These indicators help database managers get an idea of what kind of data is within the table. Tables that start with 'Fact' consist of numeric values and measurements, while tables that start with 'Dim' (dimensional) contain data that gives you more contextual information.

A few recurrent tables you will see being used in nearly all the queries in this technical report include:

**DimCustomers:** Information on the company's customers such as their name, address, income, and household.

**FactInternetSales**: Order details on all orders customers made directly through their website.

**FactReseller Sales:** Order details on all sales the company made to resellers.

**DimProduct:** Information about the different products the company manufactures, including product's subcategory, size, production line, cost, and description.

**Problem 1:**

Provide a detailed list of Internet sales with the following columns for the financial analyst team to review:

Category, Model, CustomerKey, Region, IncomeGroup, CalendarYear, FiscalYear, Month, OrderNumber, Quantity, and Amount.

The Income group column should categorize the people based on "Low" being less than 40,000, "High" being greater than 60,000, and the rest will be "Moderate".<

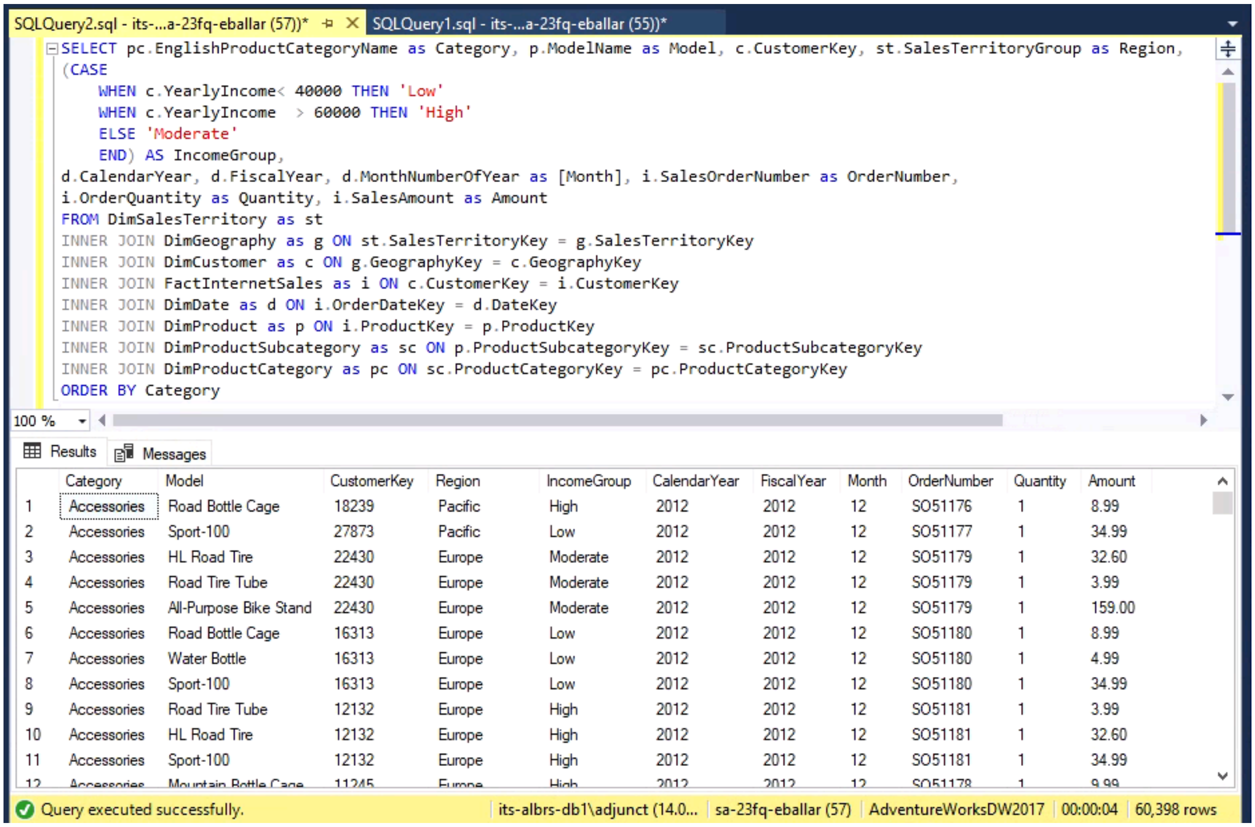
**Approach and Conclusions:**

Due to the dimensional design of the data warehouse, I had to use many joins in order to achieve the correct results. For example, we needed to grab the category of the product that was ordered. However, because the FactProduct table only includes the product's subcategory, I had to add an additional layer of joins connecting the DimProductSubcategory table to the DimProductCategory table. The same approach was used to get the sales territory group with the DimSalesTerritory and DimGeography tables. We were also required to create a new column called IncomeGroup that categorizes customers by their income. To achieve this, I used a simple CASE function. The results from this query posed more as an exploration query, but with more filtering I've discovered that most of the sales come from North America or customers with moderate incomes. Accessory products are their most purchased items as well.

**Query:**

```
SELECT pc.EnglishProductCategoryName as Category, p.ModelName as
Model, c.CustomerKey, st.SalesTerritoryGroup as Region,
(CASE
     WHEN c.YearlyIncome< 40000 THEN 'Low'
     WHEN c.YearlyIncome  > 60000 THEN 'High'
     ELSE 'Moderate'
     END) AS IncomeGroup,
d.CalendarYear, d.FiscalYear, d.MonthNumberOfYear as [Month],
i.SalesOrderNumber as OrderNumber,
i.OrderQuantity as Quantity, i.SalesAmount as Amount
FROM DimSalesTerritory as st
INNER JOIN DimGeography as g ON st.SalesTerritoryKey =
g.SalesTerritoryKey
INNER JOIN DimCustomer as c ON g.GeographyKey = c.GeographyKey
INNER JOIN FactInternetSales as i ON c.CustomerKey = i.CustomerKey
INNER JOIN DimDate as d ON i.OrderDateKey = d.DateKey
INNER JOIN DimProduct as p ON i.ProductKey = p.ProductKey
INNER JOIN DimProductSubcategory as sc ON p.ProductSubcategoryKey =
sc.ProductSubcategoryKey
INNER JOIN DimProductCategory as pc ON sc.ProductCategoryKey =
pc.ProductCategoryKey
ORDER BY Category
```

**Results:**



SQLQuery2.sql - its-...a-23fq-eballar (57))*
SQLQuery1.sql - its-...a-23fq-eballar (55))*
100 %
Results
Messages

```sql
SELECT pc.EnglishProductCategoryName as Category, p.ModelName as Model, c.CustomerKey, st.SalesTerritoryGroup as Region,
(CASE
    WHEN c.YearlyIncome< 40000 THEN 'Low'
    WHEN c.YearlyIncome  > 60000 THEN 'High'
    ELSE 'Moderate'
    END) AS IncomeGroup,
d.CalendarYear, d.FiscalYear, d.MonthNumberOfYear as [Month], i.SalesOrderNumber as OrderNumber,
i.OrderQuantity as Quantity, i.SalesAmount as Amount
FROM DimSalesTerritory as st
INNER JOIN DimGeography as g ON st.SalesTerritoryKey = g.SalesTerritoryKey
INNER JOIN DimCustomer as c ON g.GeographyKey = c.GeographyKey
INNER JOIN FactInternetSales as i ON c.CustomerKey = i.CustomerKey
INNER JOIN DimDate as d ON i.OrderDateKey = d.DateKey
INNER JOIN DimProduct as p ON i.ProductKey = p.ProductKey
INNER JOIN DimProductSubcategory as sc ON p.ProductSubcategoryKey = sc.ProductSubcategoryKey
INNER JOIN DimProductCategory as pc ON sc.ProductCategoryKey = pc.ProductCategoryKey
ORDER BY Category
```

| | Category | Model | CustomerKey | Region | IncomeGroup | CalendarYear | FiscalYear | Month | OrderNumber | Quantity | Amount |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Accessories | Road Bottle Cage | 18239 | Pacific | High | 2012 | 2012 | 12 | SO51176 | 1 | 8.99 |
| 2 | Accessories | Sport-100 | 27873 | Pacific | Low | 2012 | 2012 | 12 | SO51177 | 1 | 34.99 |
| 3 | Accessories | HL Road Tire | 22430 | Europe | Moderate | 2012 | 2012 | 12 | SO51179 | 1 | 32.60 |
| 4 | Accessories | Road Tire Tube | 22430 | Europe | Moderate | 2012 | 2012 | 12 | SO51179 | 1 | 3.99 |
| 5 | Accessories | All-Purpose Bike Stand | 22430 | Europe | Moderate | 2012 | 2012 | 12 | SO51179 | 1 | 159.00 |
| 6 | Accessories | Road Bottle Cage | 16313 | Europe | Low | 2012 | 2012 | 12 | SO51180 | 1 | 8.99 |
| 7 | Accessories | Water Bottle | 16313 | Europe | Low | 2012 | 2012 | 12 | SO51180 | 1 | 4.99 |
| 8 | Accessories | Sport-100 | 16313 | Europe | Low | 2012 | 2012 | 12 | SO51180 | 1 | 34.99 |
| 9 | Accessories | Road Tire Tube | 12132 | Europe | High | 2012 | 2012 | 12 | SO51181 | 1 | 3.99 |
| 10 | Accessories | HL Road Tire | 12132 | Europe | High | 2012 | 2012 | 12 | SO51181 | 1 | 32.60 |
| 11 | Accessories | Sport-100 | 12132 | Europe | High | 2012 | 2012 | 12 | SO51181 | 1 | 34.99 |
| 12 | Accessories | Mountain Bottle Cage | 11245 | Europe | High | 2012 | 2012 | 12 | SO51178 | 1 | 9.99 |

Query executed successfully. | its-albrs-db1\adjunct (14.0... | sa-23fq-eballar (57) | AdventureWorksDW2017 | 00:00:04 | 60,398 rows

**Problem 2:**

Provide a similar analysis for Reseller sales with the following columns (Category, Model, CalendarYear, FiscalYear, Month, OrderNumber, Quantity, Amount).

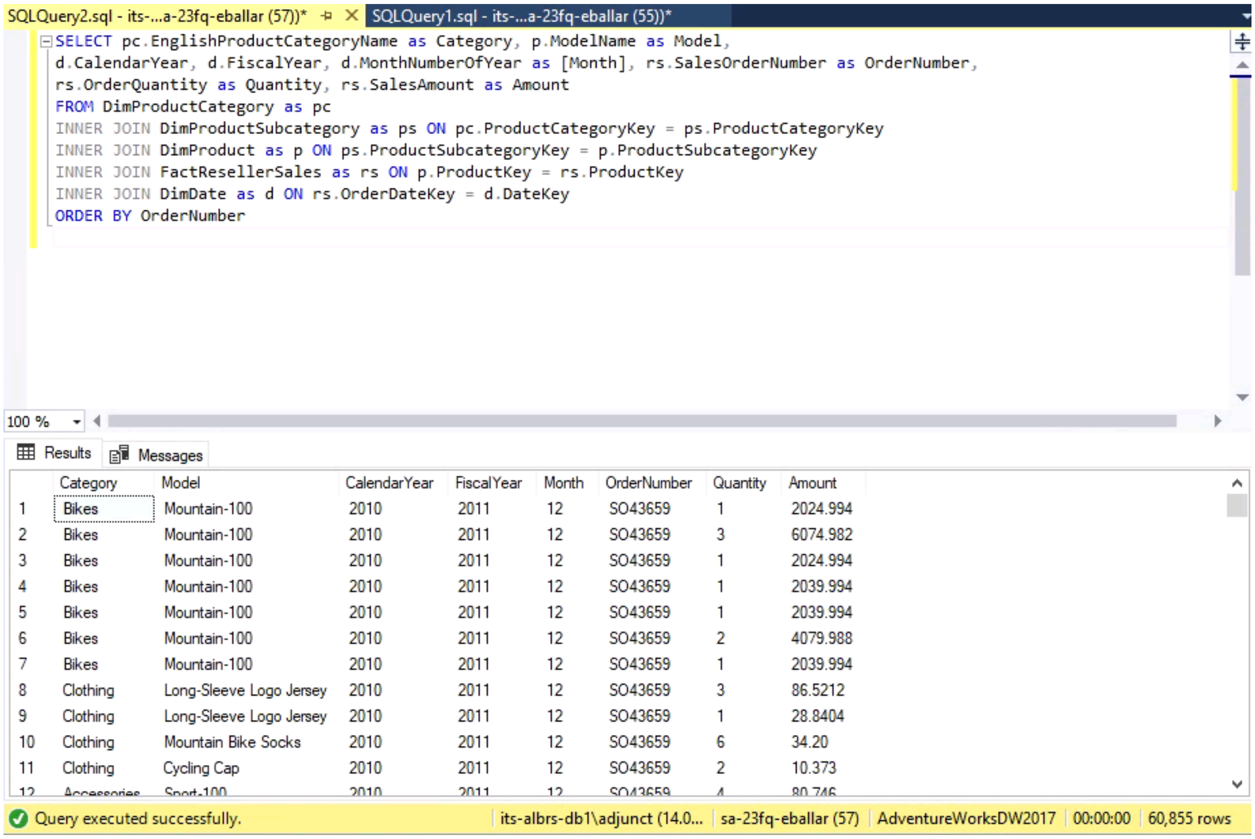
**Approach and Conclusions:**

By using a similar approach to problem one, problem two's query was straightforward. The key difference between the two queries is that we had to use the FactResellerSales table instead of the FactInternetSales. This query also did not require a CASE function for income categorization. From exploring the data from the query, I've found that bikes are the most dominant product type that is purchased by resellers.

**Query:**

```
SELECT pc.EnglishProductCategoryName as Category, p.ModelName as
Model,
d.CalendarYear, d.FiscalYear, d.MonthNumberOfYear as [Month],
rs.SalesOrderNumber as OrderNumber,
rs.OrderQuantity as Quantity, rs.SalesAmount as Amount
FROM DimProductCategory as pc
INNER JOIN DimProductSubcategory as ps ON pc.ProductCategoryKey =
ps.ProductCategoryKey
INNER JOIN DimProduct as p ON ps.ProductSubcategoryKey =
p.ProductSubcategoryKey
INNER JOIN FactResellerSales as rs ON p.ProductKey = rs.ProductKey
INNER JOIN DimDate as d ON rs.OrderDateKey = d.DateKey
ORDER BY OrderNumber
```

## Results:



SQLQuery2.sql - its-...a-23fq-eballar (57))*
SQLQuery1.sql - its-...a-23fq-eballar (55))*

```sql
SELECT pc.EnglishProductCategoryName as Category, p.ModelName as Model,
d.CalendarYear, d.FiscalYear, d.MonthNumberOfYear as [Month], rs.SalesOrderNumber as OrderNumber,
rs.OrderQuantity as Quantity, rs.SalesAmount as Amount
FROM DimProductCategory as pc
INNER JOIN DimProductSubcategory as ps ON pc.ProductCategoryKey = ps.ProductCategoryKey
INNER JOIN DimProduct as p ON ps.ProductSubcategoryKey = p.ProductSubcategoryKey
INNER JOIN FactResellerSales as rs ON p.ProductKey = rs.ProductKey
INNER JOIN DimDate as d ON rs.OrderDateKey = d.DateKey
ORDER BY OrderNumber
```

| | Category | Model | CalendarYear | FiscalYear | Month | OrderNumber | Quantity | Amount |
|---|---|---|---|---|---|---|---|---|
| 1 | Bikes | Mountain-100 | 2010 | 2011 | 12 | SO43659 | 1 | 2024.994 |
| 2 | Bikes | Mountain-100 | 2010 | 2011 | 12 | SO43659 | 3 | 6074.982 |
| 3 | Bikes | Mountain-100 | 2010 | 2011 | 12 | SO43659 | 1 | 2024.994 |
| 4 | Bikes | Mountain-100 | 2010 | 2011 | 12 | SO43659 | 1 | 2039.994 |
| 5 | Bikes | Mountain-100 | 2010 | 2011 | 12 | SO43659 | 1 | 2039.994 |
| 6 | Bikes | Mountain-100 | 2010 | 2011 | 12 | SO43659 | 2 | 4079.988 |
| 7 | Bikes | Mountain-100 | 2010 | 2011 | 12 | SO43659 | 1 | 2039.994 |
| 8 | Clothing | Long-Sleeve Logo Jersey | 2010 | 2011 | 12 | SO43659 | 3 | 86.5212 |
| 9 | Clothing | Long-Sleeve Logo Jersey | 2010 | 2011 | 12 | SO43659 | 1 | 28.8404 |
| 10 | Clothing | Mountain Bike Socks | 2010 | 2011 | 12 | SO43659 | 6 | 34.20 |
| 11 | Clothing | Cycling Cap | 2010 | 2011 | 12 | SO43659 | 2 | 10.373 |
| 12 | Accessories | Sport-100 | 2010 | 2011 | 12 | SO43659 | 4 | 80.746 |

Query executed successfully. | its-albrs-db1\adjunct (14.0... | sa-23fq-eballar (57) | AdventureWorksDW2017 | 00:00:00 | 60,855 rows

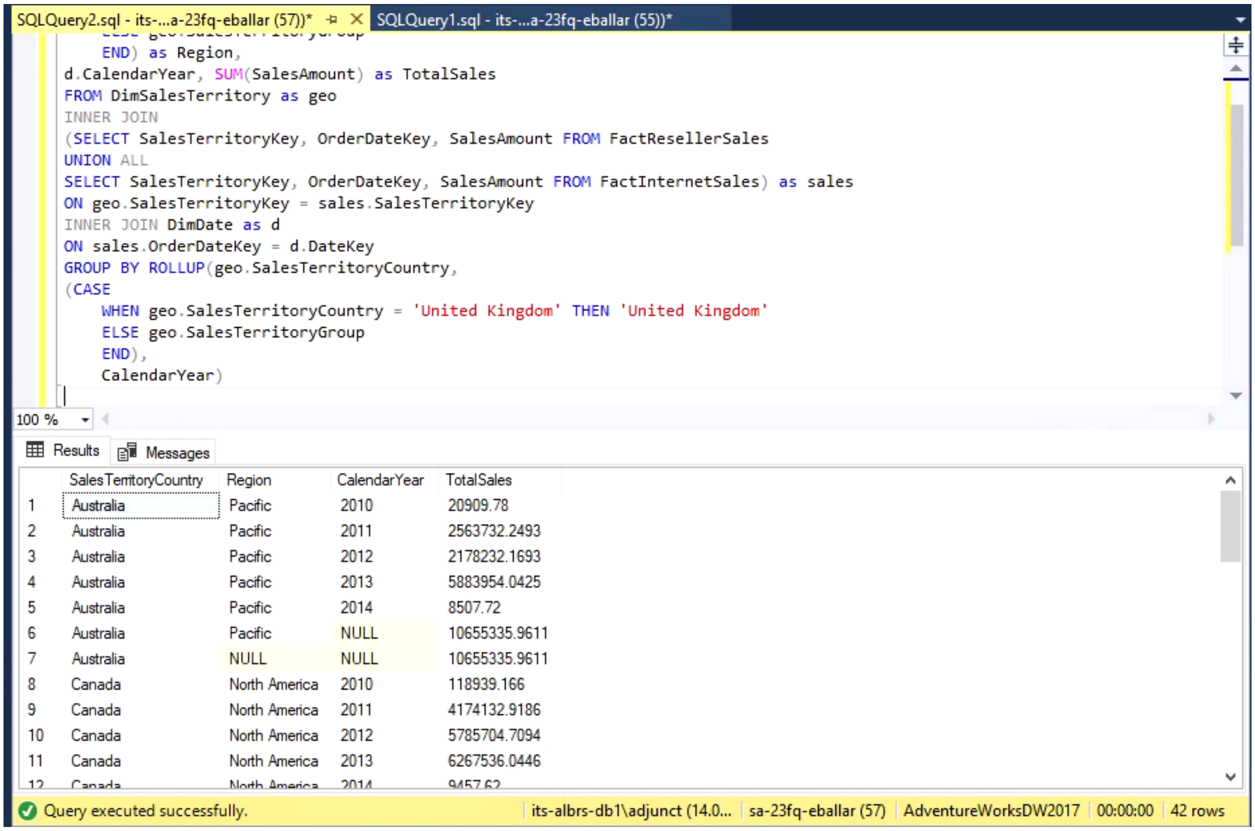
**Problem 3:**
Show the total sales (overall) by year rolled up by the Territory group and country. A special request from management is that the United Kingdom is no longer part of the European Union (EU) and they would like to see the UK's totals as a separate Territory group. You cannot modify the data, so you will need to address this request in your query.

**Approach and Conclusions:**
In addition to the joins, I had to "add in" the FactResellerSales table into the FactInternetsales table. I technically could have done this by also using an INSERT INTO function, but because it is a shared database, I opted for using the UNION ALL function. I verified it was combined correctly by making note of how many rows were in each table and ensuring the final query returned the sum of all the rows in the two tables. I also used a CASE function to give the United Kingdom its own sales region. The problem asked for the total sales to be rolled up so I did accordingly. However, I do think using CUBE would also provide some more valuable information since it returns you all the different possible combinations such as yearly sales by the country and region. This query solidified that North America generates the most sales (both internet and resellers) of all regions. This is possibly influenced by the abundant public natural spaces the region has.

**Query:**

```
SELECT geo.SalesTerritoryCountry,
(CASE
      WHEN geo.SalesTerritoryCountry = 'United Kingdom' THEN 'United
Kingdom'
      ELSE geo.SalesTerritoryGroup
      END) as Region,
d.CalendarYear, SUM(SalesAmount) as TotalSales
FROM DimSalesTerritory as geo
INNER JOIN
(SELECT SalesTerritoryKey, OrderDateKey, SalesAmount FROM
FactResellerSales
UNION ALL
SELECT SalesTerritoryKey, OrderDateKey, SalesAmount FROM
FactInternetSales) as sales
ON geo.SalesTerritoryKey = sales.SalesTerritoryKey
INNER JOIN DimDate as d
ON sales.OrderDateKey = d.DateKey
GROUP BY ROLLUP(geo.SalesTerritoryCountry,
(CASE
      WHEN geo.SalesTerritoryCountry = 'United Kingdom' THEN 'United
Kingdom'
      ELSE geo.SalesTerritoryGroup
      END),
      CalendarYear)
```

## Results:


SQLQuery2.sql - its-...a-23fq-eballar (57))*
SQLQuery1.sql - its-...a-23fq-eballar (55))*
100 %
Results
Messages

```sql
    ELSE geo.SalesTerritoryGroup
    END) as Region,
d.CalendarYear, SUM(SalesAmount) as TotalSales
FROM DimSalesTerritory as geo
INNER JOIN
(SELECT SalesTerritoryKey, OrderDateKey, SalesAmount FROM FactResellerSales
UNION ALL
SELECT SalesTerritoryKey, OrderDateKey, SalesAmount FROM FactInternetSales) as sales
ON geo.SalesTerritoryKey = sales.SalesTerritoryKey
INNER JOIN DimDate as d
ON sales.OrderDateKey = d.DateKey
GROUP BY ROLLUP(geo.SalesTerritoryCountry,
(CASE
    WHEN geo.SalesTerritoryCountry = 'United Kingdom' THEN 'United Kingdom'
    ELSE geo.SalesTerritoryGroup
    END),
    CalendarYear)
```

|  | SalesTerritoryCountry | Region | CalendarYear | TotalSales |
|---|---|---|---|---|
| 1 | Australia | Pacific | 2010 | 20909.78 |
| 2 | Australia | Pacific | 2011 | 2563732.2493 |
| 3 | Australia | Pacific | 2012 | 2178232.1693 |
| 4 | Australia | Pacific | 2013 | 5883954.0425 |
| 5 | Australia | Pacific | 2014 | 8507.72 |
| 6 | Australia | Pacific | NULL | 10655335.9611 |
| 7 | Australia | NULL | NULL | 10655335.9611 |
| 8 | Canada | North America | 2010 | 118939.166 |
| 9 | Canada | North America | 2011 | 4174132.9186 |
| 10 | Canada | North America | 2012 | 5785704.7094 |
| 11 | Canada | North America | 2013 | 6267536.0446 |
| 12 | Canada | North America | 2014 | 9457.62 |

Query executed successfully. | its-albrs-db1\adjunct (14.0... | sa-23fq-eballar (57) | AdventureWorksDW2017 | 00:00:00 | 42 rows

**Problem 4 :**

Provide an analysis of sales performance by Promotion. It would be interesting to see how different types of promotions drive sales (quantity and revenue), especially by product category or region. The comparison between Internet and Reseller sales is probably interesting as well. (Hint: don't attempt to do everything, but show some good analysis related to Promotion.)

**Approach and Conclusions:**

I initially noticed that the FactInternetSales table did not have any orders that included a discount amount despite having a linked promotion key, so my primary focus was the FactResellerSales table. In my queries, you will notice that I filter out orders with promotion key 1, which indicates that there were no discounts applied. By using the CUBE function I was able to compare the total sales of all the different promotion types by the category.  I also included a second query to take a look at the net revenue we gain from using promotions as well as the revenue "lost" because of it. Using this approach, I found that the promotions that generated the most sales for the company were the 2-5% volume discount (promotion keys 2 and 3) and the 20% off promotion for the new Touring-1000 (promotion key 14).  In addition, the bikes product category had the most demand when a discount promotion was offered. Results from the second query are not surprising. We still earn more net revenue from sales directly from the website without promotions. However, the overall amount of sales generated from reseller orders generated more gross revenue.

**Query One: Looks at Total Discounted Sales by category and promokey, in comparison to the NonDiscountedTotals, (only looks at sales with promos). Also totals up which orders belonged to what promo and category. insights: shows us which promos attracted more orders, and what categories performed the best**

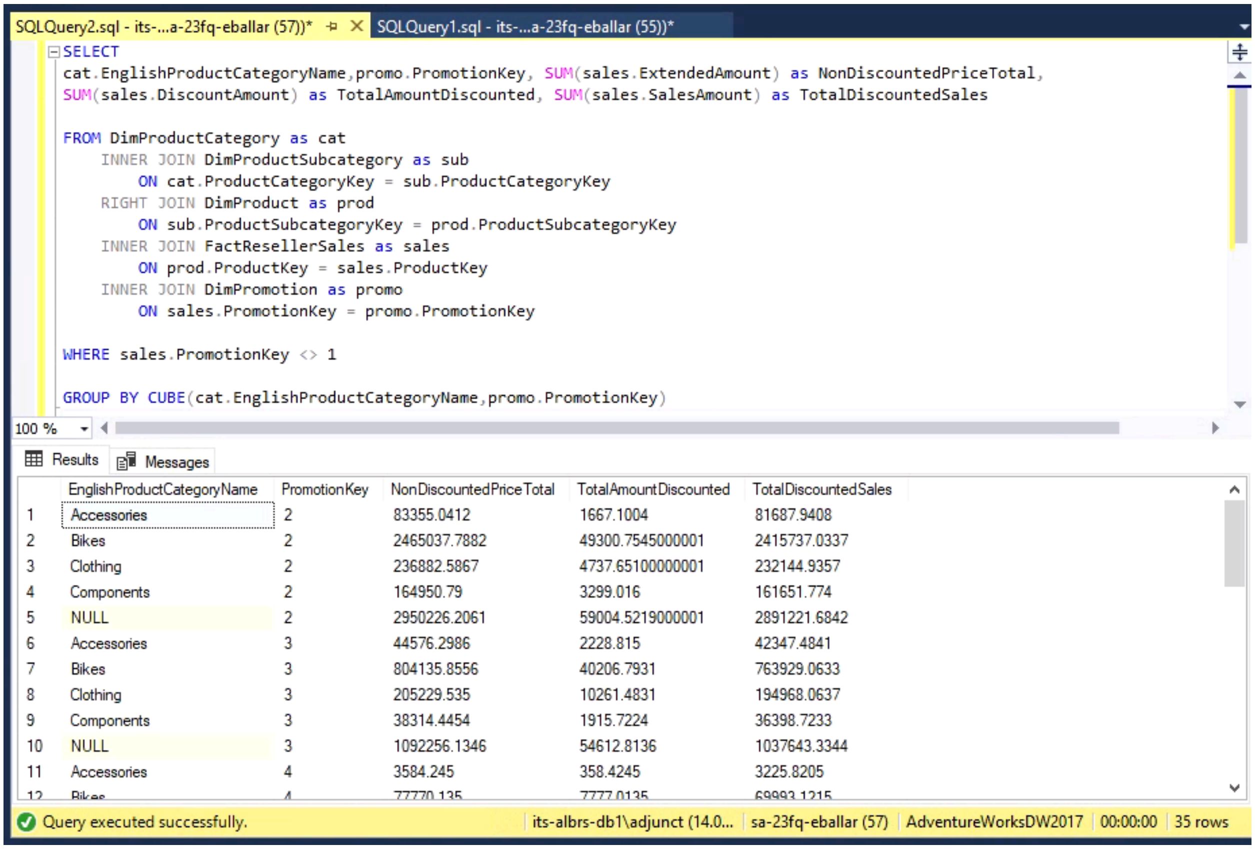
```
SELECT
cat.EnglishProductCategoryName,promo.PromotionKey,
SUM(sales.ExtendedAmount) as NonDiscountedPriceTotal,
SUM(sales.DiscountAmount) as TotalAmountDiscounted,
SUM(sales.SalesAmount) as TotalDiscountedSales

FROM DimProductCategory as cat
     INNER JOIN DimProductSubcategory as sub
          ON cat.ProductCategoryKey = sub.ProductCategoryKey
     RIGHT JOIN DimProduct as prod
          ON sub.ProductSubcategoryKey = prod.ProductSubcategoryKey
     INNER JOIN FactResellerSales as sales
          ON prod.ProductKey = sales.ProductKey
     INNER JOIN DimPromotion as promo
          ON sales.PromotionKey = promo.PromotionKey

WHERE sales.PromotionKey <> 1

GROUP BY CUBE(cat.EnglishProductCategoryName,promo.PromotionKey)
```

## Results:


SQLQuery2.sql - its-...a-23fq-eballar (57))*
SQLQuery1.sql - its-...a-23fq-eballar (55))*
100 %
Results
Messages

```sql
SELECT
cat.EnglishProductCategoryName,promo.PromotionKey, SUM(sales.ExtendedAmount) as NonDiscountedPriceTotal,
SUM(sales.DiscountAmount) as TotalAmountDiscounted, SUM(sales.SalesAmount) as TotalDiscountedSales

FROM DimProductCategory as cat
    INNER JOIN DimProductSubcategory as sub
        ON cat.ProductCategoryKey = sub.ProductCategoryKey
    RIGHT JOIN DimProduct as prod
        ON sub.ProductSubcategoryKey = prod.ProductSubcategoryKey
    INNER JOIN FactResellerSales as sales
        ON prod.ProductKey = sales.ProductKey
    INNER JOIN DimPromotion as promo
        ON sales.PromotionKey = promo.PromotionKey

WHERE sales.PromotionKey <> 1

GROUP BY CUBE(cat.EnglishProductCategoryName,promo.PromotionKey)
```

| | EnglishProductCategoryName | PromotionKey | NonDiscountedPriceTotal | TotalAmountDiscounted | TotalDiscountedSales |
|---|---|---|---|---|---|
| 1 | Accessories | 2 | 83355.0412 | 1667.1004 | 81687.9408 |
| 2 | Bikes | 2 | 2465037.7882 | 49300.7545000001 | 2415737.0337 |
| 3 | Clothing | 2 | 236882.5867 | 4737.65100000001 | 232144.9357 |
| 4 | Components | 2 | 164950.79 | 3299.016 | 161651.774 |
| 5 | NULL | 2 | 2950226.2061 | 59004.5219000001 | 2891221.6842 |
| 6 | Accessories | 3 | 44576.2986 | 2228.815 | 42347.4841 |
| 7 | Bikes | 3 | 804135.8556 | 40206.7931 | 763929.0633 |
| 8 | Clothing | 3 | 205229.535 | 10261.4831 | 194968.0637 |
| 9 | Components | 3 | 38314.4454 | 1915.7224 | 36398.7233 |
| 10 | NULL | 3 | 1092256.1346 | 54612.8136 | 1037643.3344 |
| 11 | Accessories | 4 | 3584.245 | 358.4245 | 3225.8205 |
| 12 | Bikes | 4 | 77770.135 | 7777.0135 | 69993.1215 |

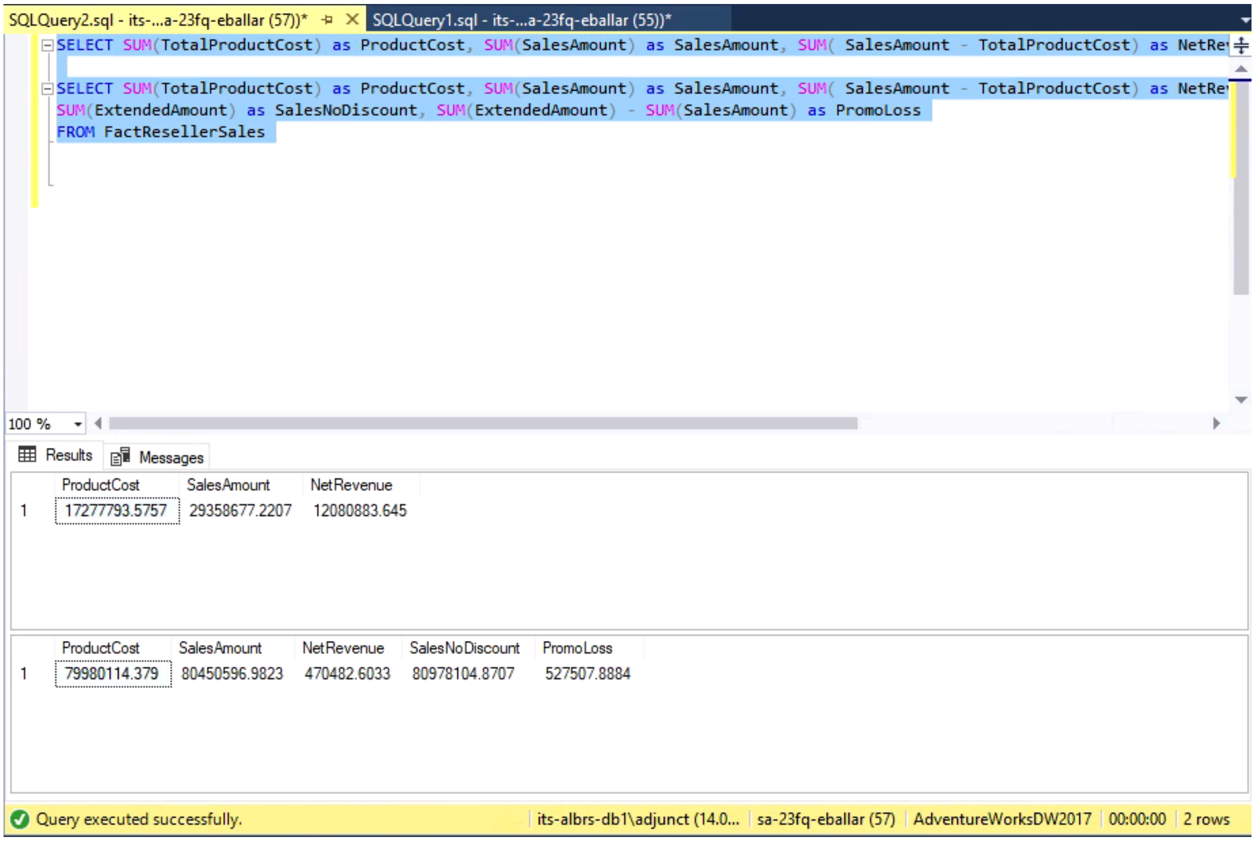
Query executed successfully. | its-albrs-db1\adjunct (14.0... | sa-23fq-eballar (57) | AdventureWorksDW2017 | 00:00:00 | 35 rows

## Query Two: data exploration, net revenues and sales totals

```
SELECT SUM(TotalProductCost) as ProductCost, SUM(SalesAmount) as
SalesAmount, SUM( SalesAmount - TotalProductCost) as NetRevenue FROM
FactInternetSales

SELECT SUM(TotalProductCost) as ProductCost, SUM(SalesAmount) as
SalesAmount, SUM( SalesAmount - TotalProductCost) as NetRevenue,
SUM(ExtendedAmount) as SalesNoDiscount, SUM(ExtendedAmount) -
SUM(SalesAmount) as PromoLoss
FROM FactResellerSales
```

## Results:



SQLQuery2.sql - its-...a-23fq-eballar (57))*
SQLQuery1.sql - its-...a-23fq-eballar (55))*

```sql
SELECT SUM(TotalProductCost) as ProductCost, SUM(SalesAmount) as SalesAmount, SUM( SalesAmount - TotalProductCost) as NetRe

SELECT SUM(TotalProductCost) as ProductCost, SUM(SalesAmount) as SalesAmount, SUM( SalesAmount - TotalProductCost) as NetRe
SUM(ExtendedAmount) as SalesNoDiscount, SUM(ExtendedAmount) - SUM(SalesAmount) as PromoLoss
FROM FactResellerSales
```

|  | ProductCost | SalesAmount | NetRevenue |
|---|---|---|---|
| 1 | 17277793.5757 | 29358677.2207 | 12080883.645 |

|  | ProductCost | SalesAmount | NetRevenue | SalesNoDiscount | PromoLoss |
|---|---|---|---|---|---|
| 1 | 79980114.379 | 80450596.9823 | 470482.6033 | 80978104.8707 | 527507.8884 |

Query executed successfully. | its-albrs-db1\adjunct (14.0... | sa-23fq-eballar (57) | AdventureWorksDW2017 | 00:00:00 | 2 rows

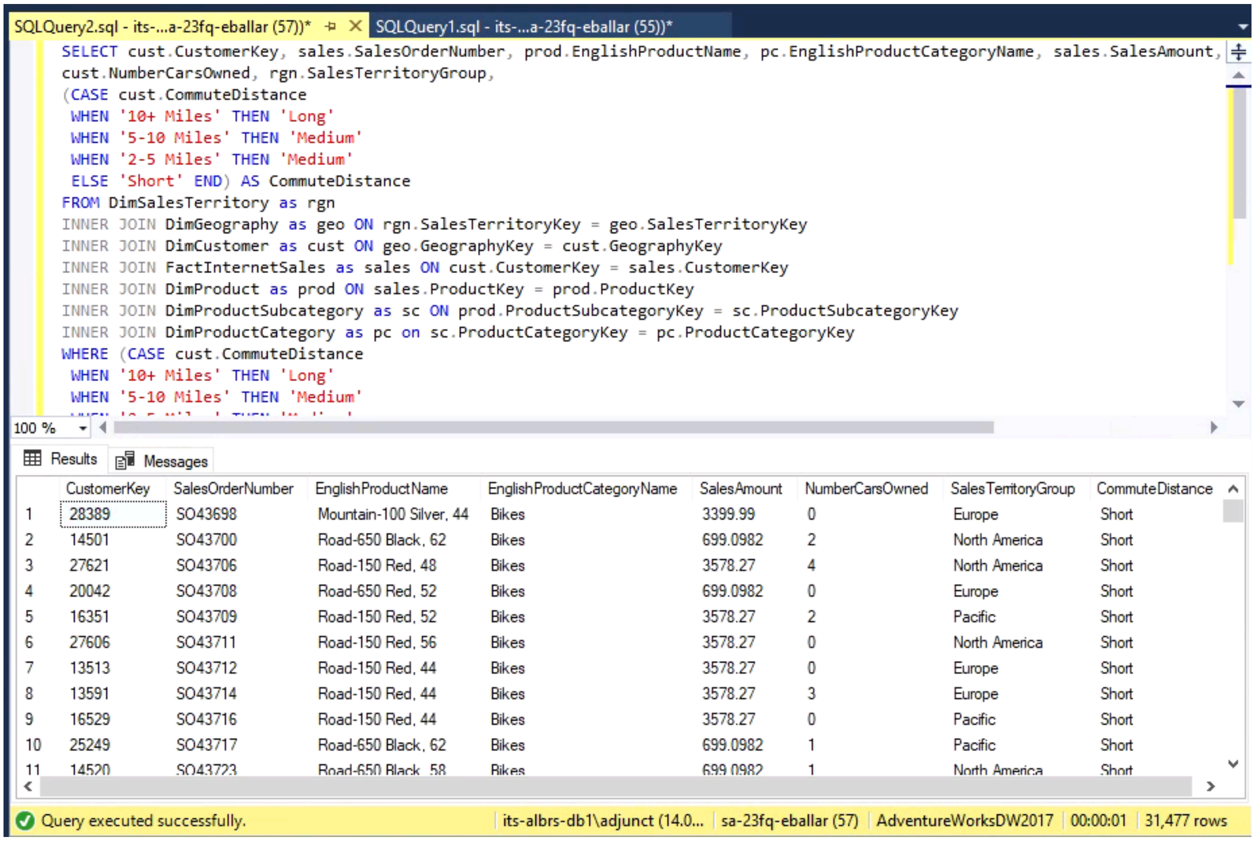
**Problem 5:**
Customers are always a big discussion topic with management and the sales team. The Customer table has a wealth of data categories that could be joined with Internet sales and all the extra data that brings along. Take this opportunity to experiment with the data and see what insights can be found.

**Approach Conclusions:**
For this problem, I wanted to explore customer's commute distances, and how it affects what they buy and their purchase frequency. I also wanted to see if there was a potential correlation between region, commute distance, and sales. To do this, I used a CASE function to categorize all the customers commuting distance into three levels: short(1-2 miles), medium(2-10 miles), and long(10+ miles). After creating a base query to use as a virtual table, I then did several other queries to explore the data further. I discovered so many insights with these queries. Firstly, short distance commuters generate the most sales for the company at a total of 31,477 internet sales. With this knowledge, we can assume many of our short commuters use our gear as their primary form of transportation. Knowing this, we can emphasize commuter friendly gear in our marketing campaigns for short commuters, and then promote our gear as primarily for fitness and adventure for our medium and long distance commuters. Short distance commuters also purchase mostly accessories (probably for maintenance), suggesting we should push more promotions on bikes and clothing rather than accessories. The last query was more or less a given, but I wanted to look at the total sales by income group and number of cars. As a result, the low and moderate income group produced the most sales, and as the number of cars went up, so did the sales.

**Base Query**(I would switch around the commute distance filter to get a general idea of the data)**:**
```
SELECT cust.CustomerKey, sales.SalesOrderNumber,
prod.EnglishProductName, pc.EnglishProductCategoryName,
sales.SalesAmount,
cust.NumberCarsOwned, rgn.SalesTerritoryGroup,
(CASE cust.CommuteDistance
 WHEN '10+ Miles' THEN 'Long'
 WHEN '5-10 Miles' THEN 'Medium'
 WHEN '2-5 Miles' THEN 'Medium'
 ELSE 'Short' END) AS CommuteDistance
FROM DimSalesTerritory as rgn
INNER JOIN DimGeography as geo ON rgn.SalesTerritoryKey =
geo.SalesTerritoryKey
INNER JOIN DimCustomer as cust ON geo.GeographyKey =
cust.GeographyKey
INNER JOIN FactInternetSales as sales ON cust.CustomerKey =
sales.CustomerKey
```

```
INNER JOIN DimProduct as prod ON sales.ProductKey = prod.ProductKey
INNER JOIN DimProductSubcategory as sc ON prod.ProductSubcategoryKey
=  sc.ProductSubcategoryKey
INNER JOIN DimProductCategory as pc on sc.ProductCategoryKey =
pc.ProductCategoryKey
WHERE (CASE cust.CommuteDistance
 WHEN '10+ Miles' THEN 'Long'
 WHEN '5-10 Miles' THEN 'Medium'
 WHEN '2-5 Miles' THEN 'Medium'
 ELSE 'Short' END) = 'Short'
```

**Results:**


SQLQuery2.sql - its-...a-23fq-eballar (57))*
SQLQuery1.sql - its-...a-23fq-eballar (55))*
100 %
Results
Messages

```
SELECT cust.CustomerKey, sales.SalesOrderNumber, prod.EnglishProductName, pc.EnglishProductCategoryName, sales.SalesAmount,
cust.NumberCarsOwned, rgn.SalesTerritoryGroup,
(CASE cust.CommuteDistance
 WHEN '10+ Miles' THEN 'Long'
 WHEN '5-10 Miles' THEN 'Medium'
 WHEN '2-5 Miles' THEN 'Medium'
 ELSE 'Short' END) AS CommuteDistance
FROM DimSalesTerritory as rgn
INNER JOIN DimGeography as geo ON rgn.SalesTerritoryKey = geo.SalesTerritoryKey
INNER JOIN DimCustomer as cust ON geo.GeographyKey = cust.GeographyKey
INNER JOIN FactInternetSales as sales ON cust.CustomerKey = sales.CustomerKey
INNER JOIN DimProduct as prod ON sales.ProductKey = prod.ProductKey
INNER JOIN DimProductSubcategory as sc ON prod.ProductSubcategoryKey = sc.ProductSubcategoryKey
INNER JOIN DimProductCategory as pc on sc.ProductCategoryKey = pc.ProductCategoryKey
WHERE (CASE cust.CommuteDistance
 WHEN '10+ Miles' THEN 'Long'
 WHEN '5-10 Miles' THEN 'Medium'
```

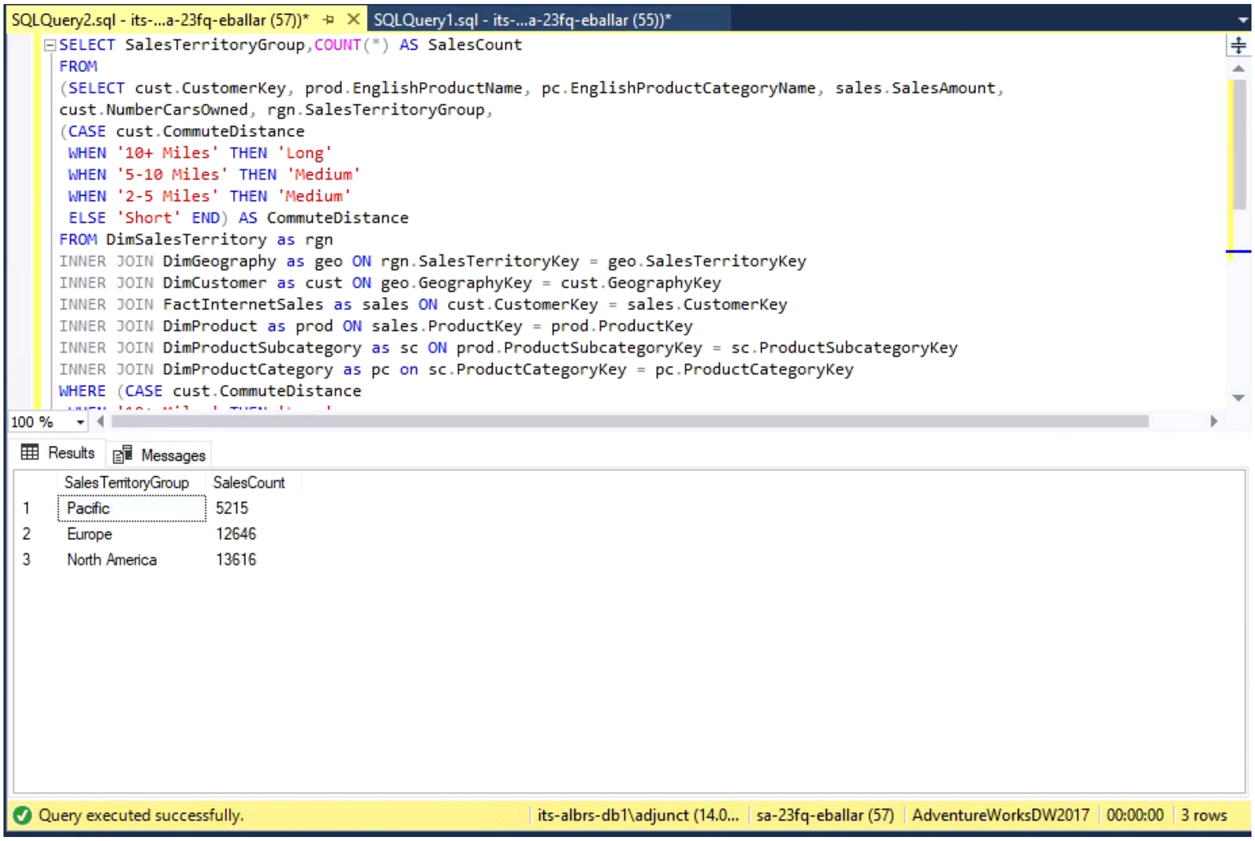
| | CustomerKey | SalesOrderNumber | EnglishProductName | EnglishProductCategoryName | SalesAmount | NumberCarsOwned | SalesTerritoryGroup | CommuteDistance |
|---|---|---|---|---|---|---|---|---|
| 1 | 28389 | SO43698 | Mountain-100 Silver, 44 | Bikes | 3399.99 | 0 | Europe | Short |
| 2 | 14501 | SO43700 | Road-650 Black, 62 | Bikes | 699.0982 | 2 | North America | Short |
| 3 | 27621 | SO43706 | Road-150 Red, 48 | Bikes | 3578.27 | 4 | North America | Short |
| 4 | 20042 | SO43708 | Road-650 Red, 52 | Bikes | 699.0982 | 0 | Europe | Short |
| 5 | 16351 | SO43709 | Road-150 Red, 52 | Bikes | 3578.27 | 2 | Pacific | Short |
| 6 | 27606 | SO43711 | Road-150 Red, 56 | Bikes | 3578.27 | 0 | North America | Short |
| 7 | 13513 | SO43712 | Road-150 Red, 44 | Bikes | 3578.27 | 0 | Europe | Short |
| 8 | 13591 | SO43714 | Road-150 Red, 44 | Bikes | 3578.27 | 3 | Europe | Short |
| 9 | 16529 | SO43716 | Road-150 Red, 44 | Bikes | 3578.27 | 0 | Pacific | Short |
| 10 | 25249 | SO43717 | Road-650 Black, 62 | Bikes | 699.0982 | 1 | Pacific | Short |
| 11 | 14520 | SO43723 | Road-650 Black, 58 | Bikes | 699.0982 | 1 | North America | Short |

Query executed successfully. | its-albrs-db1\adjunct (14.0... | sa-23fq-eballar (57) | AdventureWorksDW2017 | 00:00:01 | 31,477 rows

**How many sales for short distance commuters (by region)**

```
SELECT SalesTerritoryGroup,COUNT(*) AS SalesCount
FROM
(SELECT cust.CustomerKey, prod.EnglishProductName,
pc.EnglishProductCategoryName, sales.SalesAmount,
cust.NumberCarsOwned, rgn.SalesTerritoryGroup,
(CASE cust.CommuteDistance
 WHEN '10+ Miles' THEN 'Long'
 WHEN '5-10 Miles' THEN 'Medium'
 WHEN '2-5 Miles' THEN 'Medium'
 ELSE 'Short' END) AS CommuteDistance
FROM DimSalesTerritory as rgn
INNER JOIN DimGeography as geo ON rgn.SalesTerritoryKey =
geo.SalesTerritoryKey
INNER JOIN DimCustomer as cust ON geo.GeographyKey =
cust.GeographyKey
INNER JOIN FactInternetSales as sales ON cust.CustomerKey =
sales.CustomerKey
INNER JOIN DimProduct as prod ON sales.ProductKey = prod.ProductKey
INNER JOIN DimProductSubcategory as sc ON prod.ProductSubcategoryKey
= sc.ProductSubcategoryKey
INNER JOIN DimProductCategory as pc on sc.ProductCategoryKey =
pc.ProductCategoryKey
WHERE (CASE cust.CommuteDistance
 WHEN '10+ Miles' THEN 'Long'
 WHEN '5-10 Miles' THEN 'Medium'
 WHEN '2-5 Miles' THEN 'Medium'
 ELSE 'Short' END) = 'Short') AS one
 GROUP BY SalesTerritoryGroup
 ORDER BY SalesCount
```

## Results:


SQLQuery2.sql - its-...a-23fq-eballar (57))*
SQLQuery1.sql - its-...a-23fq-eballar (55))*
100 %
Results
Messages

```
SELECT SalesTerritoryGroup,COUNT(*) AS SalesCount
FROM
(SELECT cust.CustomerKey, prod.EnglishProductName, pc.EnglishProductCategoryName, sales.SalesAmount,
cust.NumberCarsOwned, rgn.SalesTerritoryGroup,
(CASE cust.CommuteDistance
 WHEN '10+ Miles' THEN 'Long'
 WHEN '5-10 Miles' THEN 'Medium'
 WHEN '2-5 Miles' THEN 'Medium'
 ELSE 'Short' END) AS CommuteDistance
FROM DimSalesTerritory as rgn
INNER JOIN DimGeography as geo ON rgn.SalesTerritoryKey = geo.SalesTerritoryKey
INNER JOIN DimCustomer as cust ON geo.GeographyKey = cust.GeographyKey
INNER JOIN FactInternetSales as sales ON cust.CustomerKey = sales.CustomerKey
INNER JOIN DimProduct as prod ON sales.ProductKey = prod.ProductKey
INNER JOIN DimProductSubcategory as sc ON prod.ProductSubcategoryKey = sc.ProductSubcategoryKey
INNER JOIN DimProductCategory as pc on sc.ProductCategoryKey = pc.ProductCategoryKey
WHERE (CASE cust.CommuteDistance
```

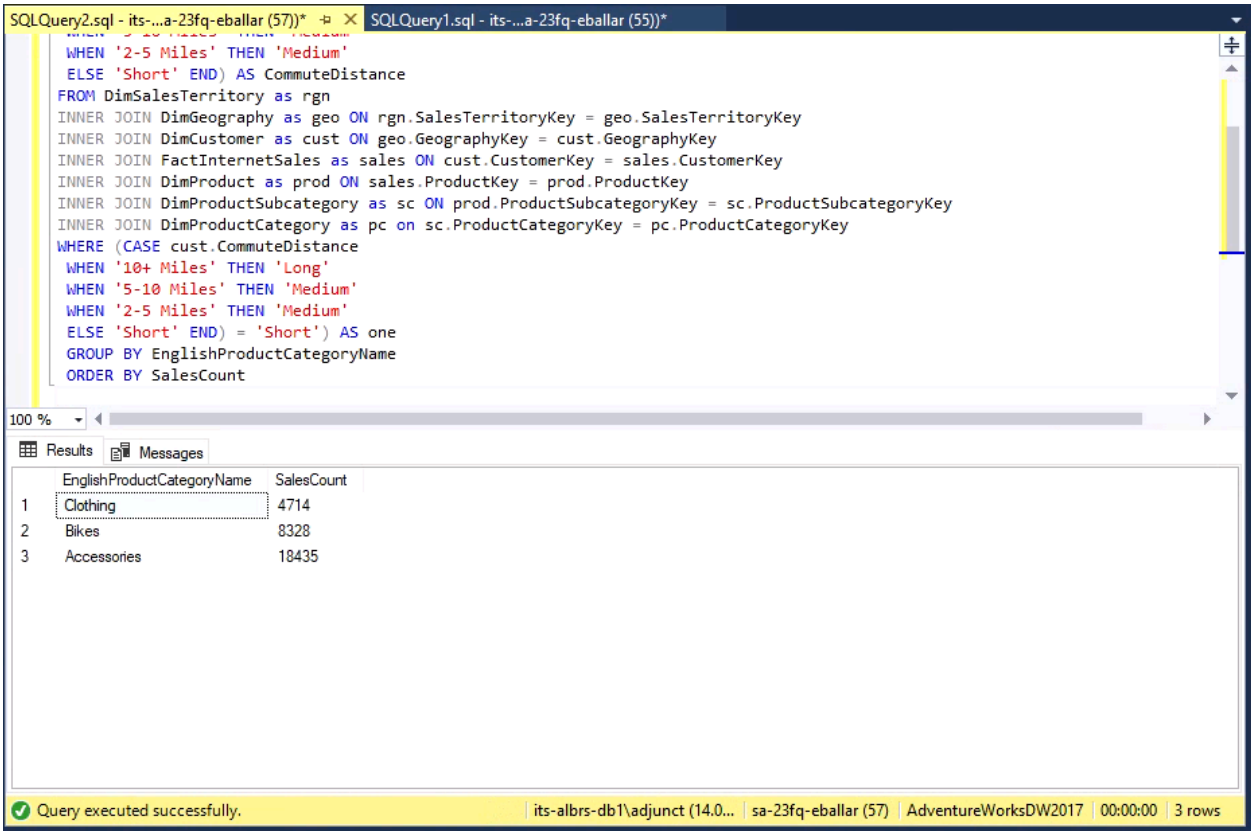
| | SalesTerritoryGroup | SalesCount |
|---|---|---|
| 1 | Pacific | 5215 |
| 2 | Europe | 12646 |
| 3 | North America | 13616 |

Query executed successfully. | its-albrs-db1\adjunct (14.0... | sa-23fq-eballar (57) | AdventureWorksDW2017 | 00:00:00 | 3 rows

## How many sales by short distance commuters by category

```
SELECT EnglishProductCategoryName,COUNT(*) AS SalesCount
FROM
(SELECT cust.CustomerKey, prod.EnglishProductName,
pc.EnglishProductCategoryName, sales.SalesAmount,
cust.NumberCarsOwned, rgn.SalesTerritoryGroup,
(CASE cust.CommuteDistance
 WHEN '10+ Miles' THEN 'Long'
 WHEN '5-10 Miles' THEN 'Medium'
 WHEN '2-5 Miles' THEN 'Medium'
 ELSE 'Short' END) AS CommuteDistance
FROM DimSalesTerritory as rgn
INNER JOIN DimGeography as geo ON rgn.SalesTerritoryKey =
geo.SalesTerritoryKey
INNER JOIN DimCustomer as cust ON geo.GeographyKey =
cust.GeographyKey
INNER JOIN FactInternetSales as sales ON cust.CustomerKey =
sales.CustomerKey
INNER JOIN DimProduct as prod ON sales.ProductKey = prod.ProductKey
INNER JOIN DimProductSubcategory as sc ON prod.ProductSubcategoryKey
= sc.ProductSubcategoryKey
INNER JOIN DimProductCategory as pc on sc.ProductCategoryKey =
pc.ProductCategoryKey
WHERE (CASE cust.CommuteDistance
 WHEN '10+ Miles' THEN 'Long'
 WHEN '5-10 Miles' THEN 'Medium'
 WHEN '2-5 Miles' THEN 'Medium'
 ELSE 'Short' END) = 'Short') AS one
 GROUP BY EnglishProductCategoryName
 ORDER BY SalesCount
```

## Results:



SQLQuery2.sql - its-...a-23fq-eballar (57))*
SQLQuery1.sql - its-...a-23fq-eballar (55))*
100 %
Results
Messages

```
 WHEN '2-5 Miles' THEN 'Medium'
 ELSE 'Short' END) AS CommuteDistance
FROM DimSalesTerritory as rgn
INNER JOIN DimGeography as geo ON rgn.SalesTerritoryKey = geo.SalesTerritoryKey
INNER JOIN DimCustomer as cust ON geo.GeographyKey = cust.GeographyKey
INNER JOIN FactInternetSales as sales ON cust.CustomerKey = sales.CustomerKey
INNER JOIN DimProduct as prod ON sales.ProductKey = prod.ProductKey
INNER JOIN DimProductSubcategory as sc ON prod.ProductSubcategoryKey = sc.ProductSubcategoryKey
INNER JOIN DimProductCategory as pc on sc.ProductCategoryKey = pc.ProductCategoryKey
WHERE (CASE cust.CommuteDistance
 WHEN '10+ Miles' THEN 'Long'
 WHEN '5-10 Miles' THEN 'Medium'
 WHEN '2-5 Miles' THEN 'Medium'
 ELSE 'Short' END) = 'Short') AS one
 GROUP BY EnglishProductCategoryName
 ORDER BY SalesCount
```

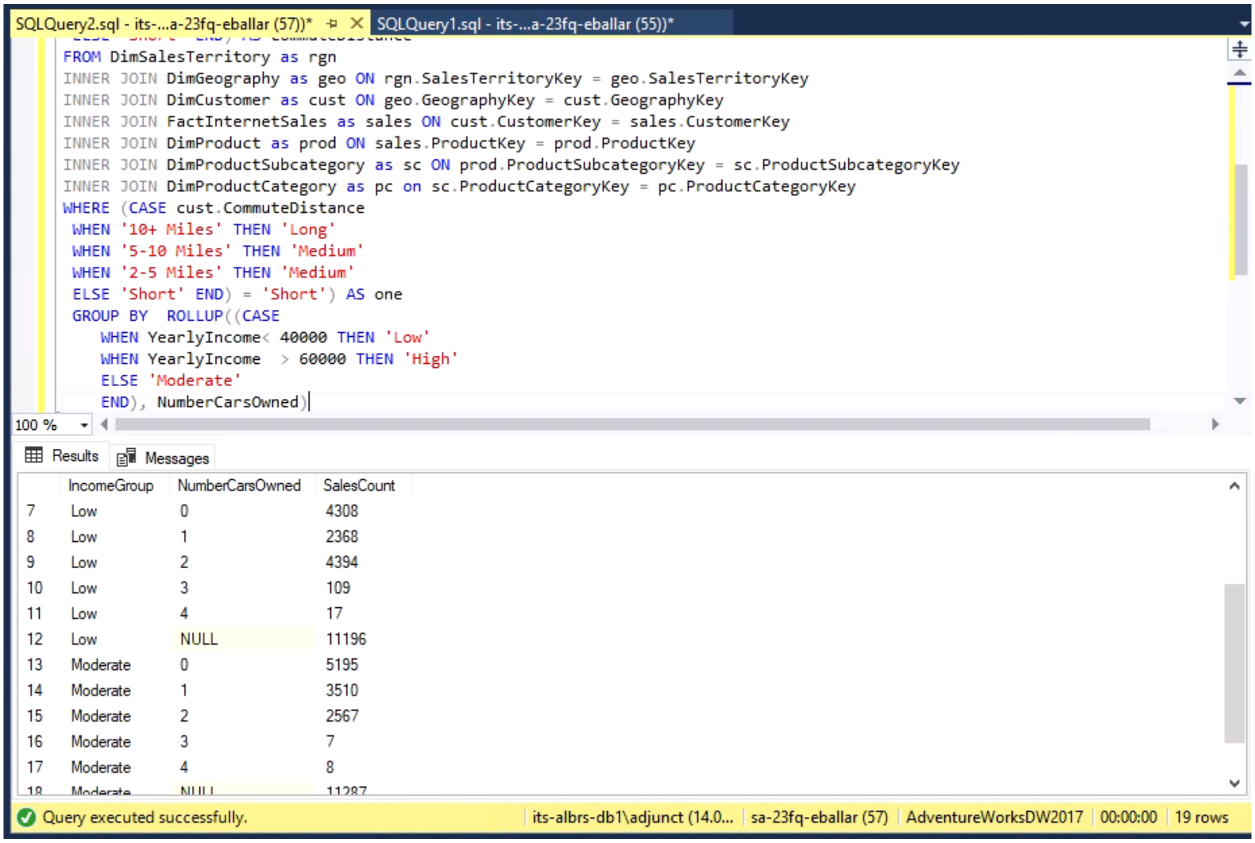
|  | EnglishProductCategoryName | SalesCount |
|---|---|---|
| 1 | Clothing | 4714 |
| 2 | Bikes | 8328 |
| 3 | Accessories | 18435 |

Query executed successfully. | its-albrs-db1\adjunct (14.0... | sa-23fq-eballar (57) | AdventureWorksDW2017 | 00:00:00 | 3 rows

**Number of purchases by income group and number of cars**

```
SELECT (CASE
     WHEN YearlyIncome< 40000 THEN 'Low'
     WHEN YearlyIncome  > 60000 THEN 'High'
     ELSE 'Moderate'
     END) AS IncomeGroup, NumberCarsOwned,COUNT(*) AS SalesCount
FROM
(SELECT cust.CustomerKey, prod.EnglishProductName,
pc.EnglishProductCategoryName, sales.SalesAmount,
cust.NumberCarsOwned, cust.YearlyIncome,rgn.SalesTerritoryGroup,
(CASE cust.CommuteDistance
 WHEN '10+ Miles' THEN 'Long'
 WHEN '5-10 Miles' THEN 'Medium'
 WHEN '2-5 Miles' THEN 'Medium'
 ELSE 'Short' END) AS CommuteDistance
FROM DimSalesTerritory as rgn
INNER JOIN DimGeography as geo ON rgn.SalesTerritoryKey =
geo.SalesTerritoryKey
INNER JOIN DimCustomer as cust ON geo.GeographyKey =
cust.GeographyKey
INNER JOIN FactInternetSales as sales ON cust.CustomerKey =
sales.CustomerKey
INNER JOIN DimProduct as prod ON sales.ProductKey = prod.ProductKey
INNER JOIN DimProductSubcategory as sc ON prod.ProductSubcategoryKey
= sc.ProductSubcategoryKey
INNER JOIN DimProductCategory as pc on sc.ProductCategoryKey =
pc.ProductCategoryKey
WHERE (CASE cust.CommuteDistance
 WHEN '10+ Miles' THEN 'Long'
 WHEN '5-10 Miles' THEN 'Medium'
 WHEN '2-5 Miles' THEN 'Medium'
 ELSE 'Short' END) = 'Short') AS one
 GROUP BY  ROLLUP((CASE
     WHEN YearlyIncome< 40000 THEN 'Low'
     WHEN YearlyIncome  > 60000 THEN 'High'
     ELSE 'Moderate'
     END), NumberCarsOwned)
```

## Results:



SQLQuery2.sql - its-...a-23fq-eballar (57))*
SQLQuery1.sql - its-...a-23fq-eballar (55))*
100 %
Results
Messages

```sql
 ELSE 'Short' END) AS CommuteDistance
FROM DimSalesTerritory as rgn
INNER JOIN DimGeography as geo ON rgn.SalesTerritoryKey = geo.SalesTerritoryKey
INNER JOIN DimCustomer as cust ON geo.GeographyKey = cust.GeographyKey
INNER JOIN FactInternetSales as sales ON cust.CustomerKey = sales.CustomerKey
INNER JOIN DimProduct as prod ON sales.ProductKey = prod.ProductKey
INNER JOIN DimProductSubcategory as sc ON prod.ProductSubcategoryKey = sc.ProductSubcategoryKey
INNER JOIN DimProductCategory as pc on sc.ProductCategoryKey = pc.ProductCategoryKey
WHERE (CASE cust.CommuteDistance
 WHEN '10+ Miles' THEN 'Long'
 WHEN '5-10 Miles' THEN 'Medium'
 WHEN '2-5 Miles' THEN 'Medium'
 ELSE 'Short' END) = 'Short') AS one
 GROUP BY  ROLLUP((CASE
    WHEN YearlyIncome< 40000 THEN 'Low'
    WHEN YearlyIncome  > 60000 THEN 'High'
    ELSE 'Moderate'
    END), NumberCarsOwned)
```

| | IncomeGroup | NumberCarsOwned | SalesCount |
|---|---|---|---|
| 7 | Low | 0 | 4308 |
| 8 | Low | 1 | 2368 |
| 9 | Low | 2 | 4394 |
| 10 | Low | 3 | 109 |
| 11 | Low | 4 | 17 |
| 12 | Low | NULL | 11196 |
| 13 | Moderate | 0 | 5195 |
| 14 | Moderate | 1 | 3510 |
| 15 | Moderate | 2 | 2567 |
| 16 | Moderate | 3 | 7 |
| 17 | Moderate | 4 | 8 |
| 18 | Moderate | NULL | 11287 |

Query executed successfully. | its-albrs-db1\adjunct (14.0... | sa-23fq-eballar (57) | AdventureWorksDW2017 | 00:00:00 | 19 rows