ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

DATA MANAGEMENT AND BUSINESS INTELLIGENCE

ASSIGNMENT 2

ACADEMIC YEAR 2020 – 2021 (PART TIME)

SUBJECT: **GLOBAL SUPERSTORE REPORT**

STYLIANOS VRETTEAS – P2822003
ANDRIANI KARPATHAKI – P2822020

BUSINESS ANALYTICS
Master of Science

Contents

1. INTRODUCTION

The scope of this report was to design and develop a data warehouse, build a data cube on top of it, develop some OLAP reports and visualize our results based on an existing business model referring to retail sales. This was accomplished by using SQL Server Database, SQL Server Analysis Services and Power BI of Microsoft as presented below in detail.

2. CASE STUDY

Our business study concerns a multinational Superstore specialized in retail with trading activities in 137 different countries. With a wide selection of products regarding three main categories (office supplies, furniture, technology), Superstore stays competitive by offering customers more than _____ items.

However, sales success is a result of providing the right product in the right place at the right time and to the appropriate customer, given that each environment generates different needs, unique for any individual. That is to say, right product selection powers energetic growth and a higher return on investment.

This can be accomplished by using historical data and market analysis in order to identify customer trends and changing preferences by combining data from different areas.

In addition, by studying sales data with a variety of related factors, businesses such as Superstore, can recognize emerging trends, anticipate them better and eventually create the agility to respond efficiently to market complexity.

That is exactly our subject of interest so we are going to use a data set containing information about products, sales, profits for four (consecutive) years etc. in order to identify key areas for improvement within this fictitious company.

3. DATA SOURSE

The Data Source we used so as to obtain our dataset was *kaggle*, one of the best free open Data Sources where it is possible to find and publish data sets as well as explore and build models in a web-based data-science environment. The dataset we obtained for the purpose of this assignment was *superstore_data*.

Data Sourse	https://www.kaggle.com/datasets
Dataset	https://www.kaggle.com/jr2ngb/superstore-data

3.1. Description of the Dataset

The initial dataset was a csv file containing 24 columns:

- Row ID primary key
- Order ID
- Order Date
- Ship Date
- Customer ID
- Customer Name
- Segment
- City
- State
- Country
- Postal Code
- Market
- Region
- Product ID
- Category
- Sub-Category
- Product Name
- Order Priority
- Sales
- Quantity
- Discount
- Profit
- Shipping Cost

Metrics

As seen above there is plenty of information available regarding sales, such as the product names, their categories and sub-categories, information about the customers including names and the place they live as well as information about the order and ship date. The last five (5) columns show the sales metrics.

4. ANALYSIS

The analysis of Superstore retail sales focused on the below steps:

1. Design and develop the data warehouse
2. Build the data cube
3. Develop some OLAP reports
4. Visualize our results

4.1. Design and develop the data warehouse

We connected to the SQL Server Database in order to create the database in which we would load the dataset and start the analysis. The steps we followed at this stage come as follows:

1. Importing the data into the database
2. Cleaning and Transforming the Data
3. Creating the dimensions
4. Creating the fact table
5. Creating the relationships between fact table and dimensions
6. Create the Schema

4.1.1. Importing the data into the Database

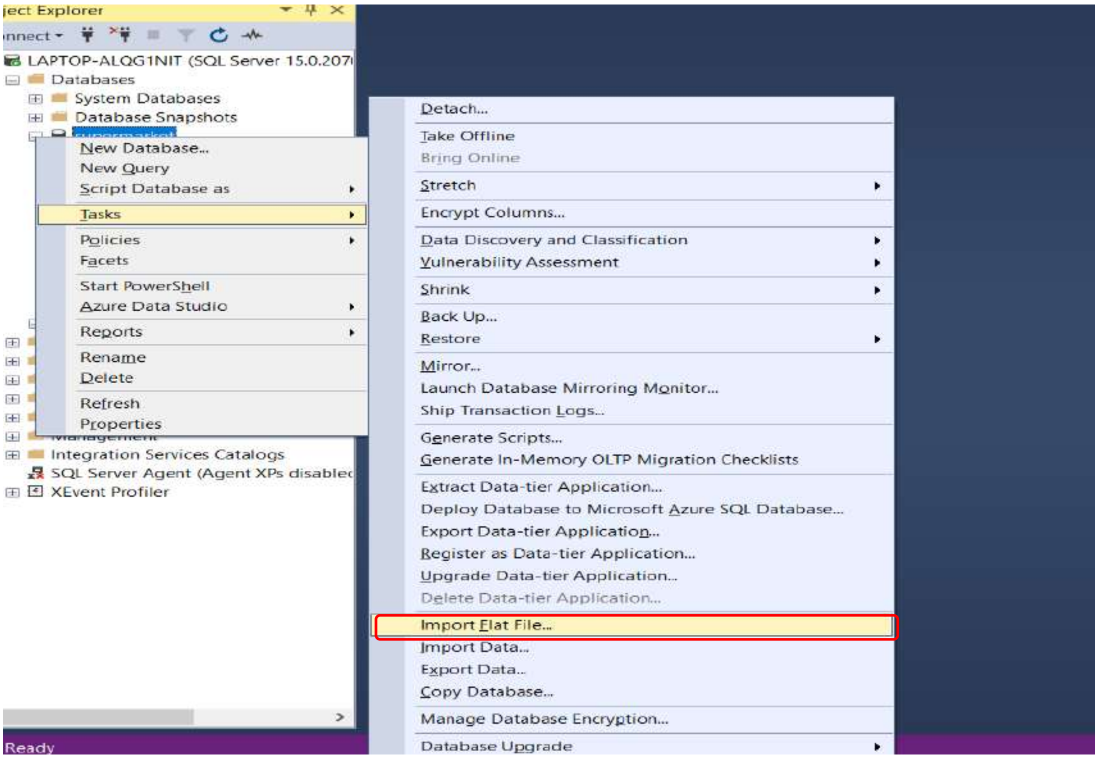
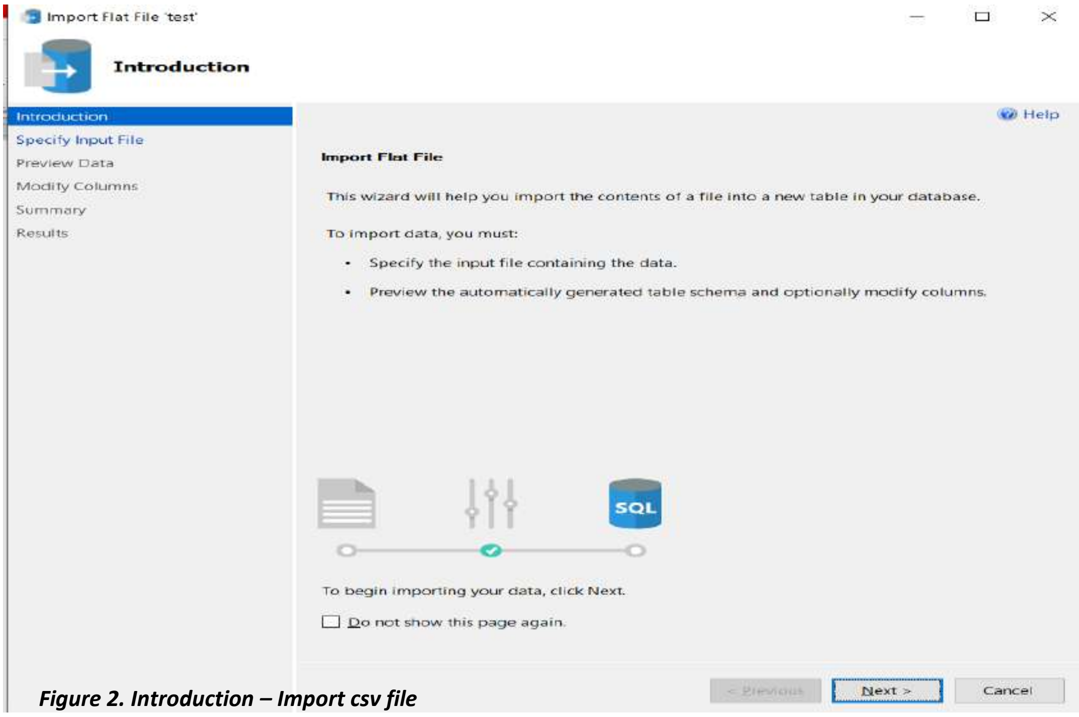


Figure 1. Importing csv file into the Database



Import Flat File

This wizard will help you import the contents of a file into a new table in your database.

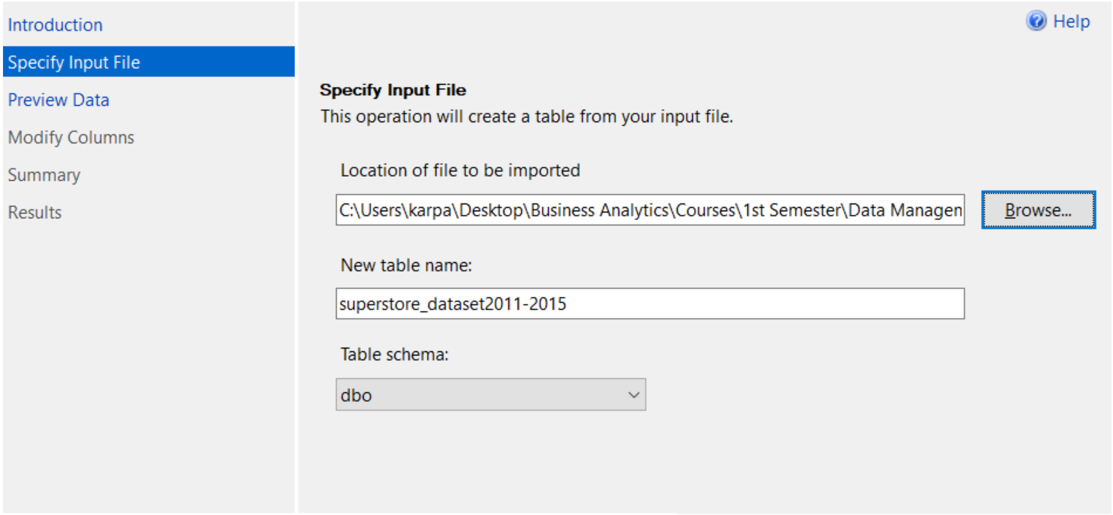
To import data, you must:

- Specify the input file containing the data.
- Preview the automatically generated table schema and optionally modify columns.

To begin importing your data, click Next.

□ Do not show this page again.

Figure 2. Introduction – Import csv file



Specify Input File

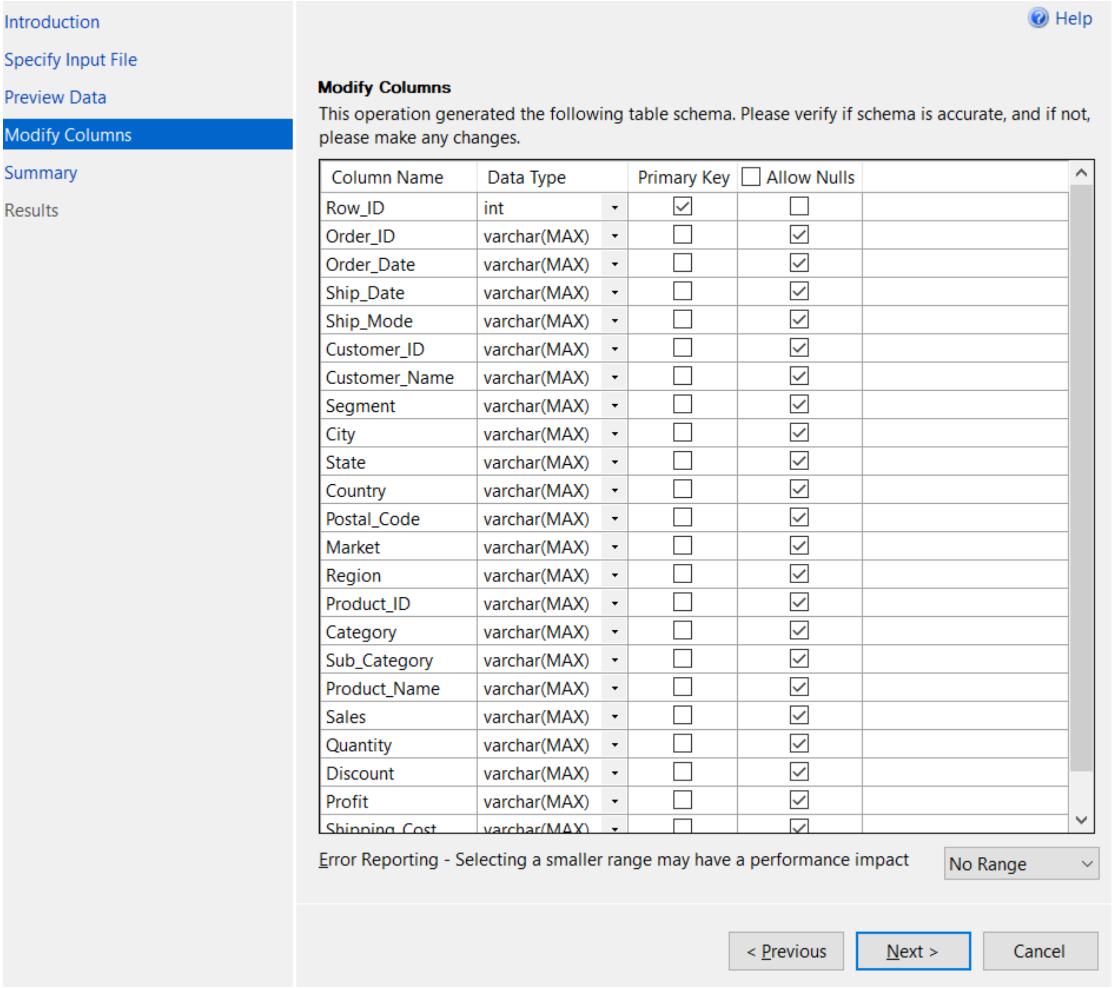
This operation will create a table from your input file.

Location of file to be imported: C:\Users\karpa\Desktop\Business Analytics\Courses\1st Semester\Data Managen

New table name: superstore_dataset2011-2015

Table schema: dbo

Figure 3. Specify Input File – Import csv file



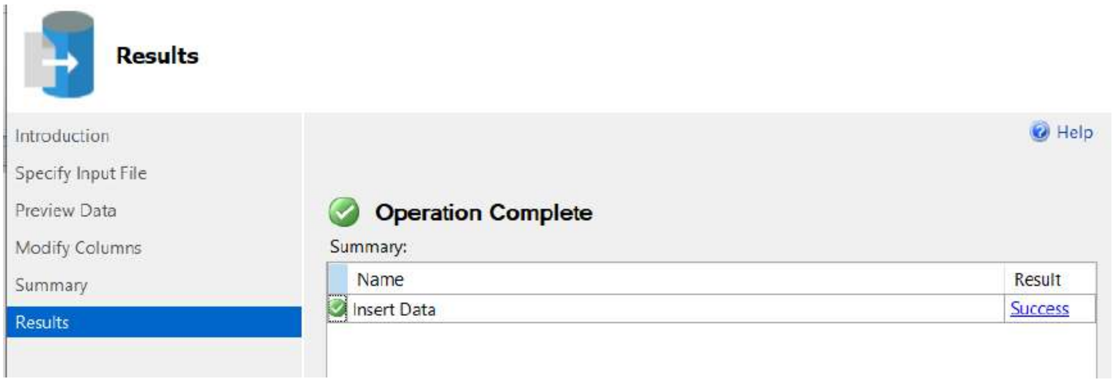
Modify Columns

This operation generated the following table schema. Please verify if schema is accurate, and if not, please make any changes.

Column Name	Data Type	Primary Key	□ Allow Nulls
Row_ID	int	☑	□
Order_ID	varchar(MAX)	□	☑
Order_Date	varchar(MAX)	□	☑
Ship_Date	varchar(MAX)	□	☑
Ship_Mode	varchar(MAX)	□	☑
Customer_ID	varchar(MAX)	□	☑
Customer_Name	varchar(MAX)	□	☑
Segment	varchar(MAX)	□	☑
City	varchar(MAX)	□	☑
State	varchar(MAX)	□	☑
Country	varchar(MAX)	□	☑
Postal_Code	varchar(MAX)	□	☑
Market	varchar(MAX)	□	☑
Region	varchar(MAX)	□	☑
Product_ID	varchar(MAX)	□	☑
Category	varchar(MAX)	□	☑
Sub_Category	varchar(MAX)	□	☑
Product_Name	varchar(MAX)	□	☑
Sales	varchar(MAX)	□	☑
Quantity	varchar(MAX)	□	☑
Discount	varchar(MAX)	□	☑
Profit	varchar(MAX)	□	☑
Shipping_Cost	varchar(MAX)	□	☑

Error Reporting - Selecting a smaller range may have a performance impact: No Range

Figure 4. Modify Columns – Import csv File



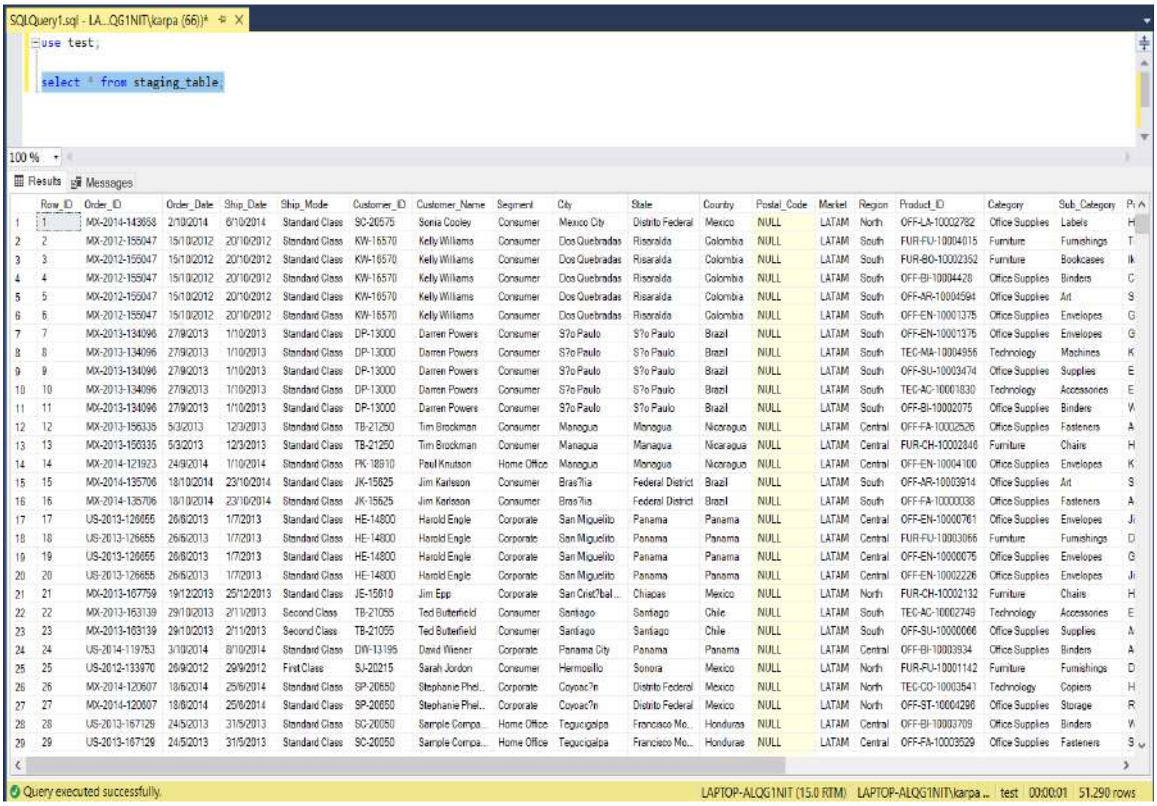
Operation Complete

Summary:

Name	Result
Insert Data	Success

Figure 5. Results – Import csv File

During this process we faced some problems while defining the type of some columns, more precisely the order and ship date column as well as metrics. In order to overcome this issue we assigned varchar(max) type to every column except for Row_ID, which we would fix at the creation of dimensions (Figure 4. Modify Columns – Import csv File).



```
use test;

select * from staging_table;
```

	Row_ID	Order_ID	Order_Date	Ship_Date	Ship_Mode	Customer_ID	Customer_Name	Segment	City	State	Country	Postal_Code	Market	Region	Product_ID	Category	Sub_Category	Pı
1	1	MX-2014-143658	2/10/2014	6/10/2014	Standard Class	SC-20575	Sonia Cooley	Consumer	Mexico City	Distrito Federal	Mexico	NULL	LATAM	North	OFF-LA-10002782	Office Supplies	Labels	H
2	2	MX-2012-155047	15/10/2012	20/10/2012	Standard Class	KW-16570	Kelly Williams	Consumer	Dos Quebradas	Risaralda	Colombia	NULL	LATAM	South	FUR-FU-10004015	Furniture	Furnishings	T
3	3	MX-2012-155047	15/10/2012	20/10/2012	Standard Class	KW-16570	Kelly Williams	Consumer	Dos Quebradas	Risaralda	Colombia	NULL	LATAM	South	FUR-BO-10002352	Furniture	Bookcases	Ik
4	4	MX-2012-155047	15/10/2012	20/10/2012	Standard Class	KW-16570	Kelly Williams	Consumer	Dos Quebradas	Risaralda	Colombia	NULL	LATAM	South	OFF-BI-10004428	Office Supplies	Binders	C
5	5	MX-2012-155047	15/10/2012	20/10/2012	Standard Class	KW-16570	Kelly Williams	Consumer	Dos Quebradas	Risaralda	Colombia	NULL	LATAM	South	OFF-AR-10004594	Office Supplies	Art	S
6	6	MX-2012-155047	15/10/2012	20/10/2012	Standard Class	KW-16570	Kelly Williams	Consumer	Dos Quebradas	Risaralda	Colombia	NULL	LATAM	South	OFF-EN-10001375	Office Supplies	Envelopes	G
7	7	MX-2013-134096	27/9/2013	1/10/2013	Standard Class	DP-13000	Darren Powers	Consumer	S?o Paulo	S?o Paulo	Brazil	NULL	LATAM	South	OFF-EN-10001375	Office Supplies	Envelopes	G
8	8	MX-2013-134096	27/9/2013	1/10/2013	Standard Class	DP-13000	Darren Powers	Consumer	S?o Paulo	S?o Paulo	Brazil	NULL	LATAM	South	TEC-MA-10004956	Technology	Machines	K
9	9	MX-2013-134096	27/9/2013	1/10/2013	Standard Class	DP-13000	Darren Powers	Consumer	S?o Paulo	S?o Paulo	Brazil	NULL	LATAM	South	OFF-SU-10003474	Office Supplies	Supplies	E
10	10	MX-2013-134096	27/9/2013	1/10/2013	Standard Class	DP-13000	Darren Powers	Consumer	S?o Paulo	S?o Paulo	Brazil	NULL	LATAM	South	TEC-AC-10001830	Technology	Accessories	E
11	11	MX-2013-134096	27/9/2013	1/10/2013	Standard Class	DP-13000	Darren Powers	Consumer	S?o Paulo	S?o Paulo	Brazil	NULL	LATAM	South	OFF-BI-10002075	Office Supplies	Binders	W
12	12	MX-2013-156335	5/3/2013	12/3/2013	Standard Class	TB-21250	Tim Brockman	Consumer	Managua	Managua	Nicaragua	NULL	LATAM	Central	OFF-FA-10002526	Office Supplies	Fasteners	A
13	13	MX-2013-156335	5/3/2013	12/3/2013	Standard Class	TB-21250	Tim Brockman	Consumer	Managua	Managua	Nicaragua	NULL	LATAM	Central	FUR-CH-10002846	Furniture	Chairs	H
14	14	MX-2014-121923	24/9/2014	1/10/2014	Standard Class	PK-18910	Paul Knutson	Home Office	Managua	Managua	Nicaragua	NULL	LATAM	Central	OFF-EN-10004100	Office Supplies	Envelopes	K
15	15	MX-2014-135706	18/10/2014	23/10/2014	Standard Class	JK-15625	Jim Karlsson	Consumer	Bras?lia	Federal District	Brazil	NULL	LATAM	South	OFF-AR-10003914	Office Supplies	Art	S
16	16	MX-2014-135706	18/10/2014	23/10/2014	Standard Class	JK-15625	Jim Karlsson	Consumer	Bras?lia	Federal District	Brazil	NULL	LATAM	South	OFF-FA-10000038	Office Supplies	Fasteners	A
17	17	US-2013-126655	26/6/2013	1/7/2013	Standard Class	HE-14800	Harold Engle	Corporate	San Miguelito	Panama	Panama	NULL	LATAM	Central	OFF-EN-10000761	Office Supplies	Envelopes	Ji
18	18	US-2013-126655	26/6/2013	1/7/2013	Standard Class	HE-14800	Harold Engle	Corporate	San Miguelito	Panama	Panama	NULL	LATAM	Central	FUR-FU-10003066	Furniture	Furnishings	D
19	19	US-2013-126655	26/6/2013	1/7/2013	Standard Class	HE-14800	Harold Engle	Corporate	San Miguelito	Panama	Panama	NULL	LATAM	Central	OFF-EN-10000075	Office Supplies	Envelopes	G
20	20	US-2013-126655	26/6/2013	1/7/2013	Standard Class	HE-14800	Harold Engle	Corporate	San Miguelito	Panama	Panama	NULL	LATAM	Central	OFF-EN-10002226	Office Supplies	Envelopes	Ji
21	21	MX-2013-167759	19/12/2013	25/12/2013	Standard Class	JE-15610	Jim Epp	Corporate	San Crist?bal ...	Chiapas	Mexico	NULL	LATAM	North	FUR-CH-10002132	Furniture	Chairs	H
22	22	MX-2013-163139	29/10/2013	2/11/2013	Second Class	TB-21055	Ted Butterfield	Consumer	Santiago	Santiago	Chile	NULL	LATAM	South	TEC-AC-10002749	Technology	Accessories	E
23	23	MX-2013-163139	29/10/2013	2/11/2013	Second Class	TB-21055	Ted Butterfield	Consumer	Santiago	Santiago	Chile	NULL	LATAM	South	OFF-SU-10000066	Office Supplies	Supplies	A
24	24	US-2014-119753	3/10/2014	8/10/2014	Standard Class	DW-13195	David Wiener	Corporate	Panama City	Panama	Panama	NULL	LATAM	Central	OFF-BI-10003934	Office Supplies	Binders	A
25	25	US-2012-133970	26/9/2012	29/9/2012	First Class	SJ-20215	Sarah Jordon	Consumer	Hermosillo	Sonora	Mexico	NULL	LATAM	North	FUR-FU-10001142	Furniture	Furnishings	D
26	26	MX-2014-120607	18/6/2014	25/6/2014	Standard Class	SP-20650	Stephanie Phel...	Corporate	Coyoac?n	Distrito Federal	Mexico	NULL	LATAM	North	TEC-CO-10003541	Technology	Copiers	H
27	27	MX-2014-120607	18/6/2014	25/6/2014	Standard Class	SP-20650	Stephanie Phel...	Corporate	Coyoac?n	Distrito Federal	Mexico	NULL	LATAM	North	OFF-ST-10004296	Office Supplies	Storage	R
28	28	US-2013-167129	24/5/2013	31/5/2013	Standard Class	SC-20050	Sample Compa...	Home Office	Tegucigalpa	Francisco Mo...	Honduras	NULL	LATAM	Central	OFF-BI-10003709	Office Supplies	Binders	W
29	29	US-2013-167129	24/5/2013	31/5/2013	Standard Class	SC-20050	Sample Compa...	Home Office	Tegucigalpa	Francisco Mo...	Honduras	NULL	LATAM	Central	OFF-FA-10003529	Office Supplies	Fasteners	S

Query executed successfully. LAPTOP-ALQG1NIT (15.0 RTM) LAPTOP-ALQG1NIT\karpa ... test 00:00:01 51.290 rows

Figure 6. Data illustration

4.1.2. Cleaning and Transforming the Data

At this stage we cleaned and transformed our data. At first we removed the Postal_Code column because many values were missing so we concluded that it would not contribute to our analysis. Later on we checked for NULL values for all columns and we extracted rows that were missing many or important values.

After deciding the data we would work on, we started transforming them. Our first attempt was to convert the metrics from varchar to float type. In order to do that we created a staging table that fulfill all the requirements needed for importing only clean data in the fact table.

Following, we conducted the same procedure as before in order to transform the order and ship date columns.

During this procedure, we faced an unpredictable event .The entries for the date did not have the same format as ddmmyyyy and had also different symbol as a separator for day, month and year. More precisely, some had the *slash* " / " and some other the *dash* " - " symbol, so we had to make a query in order to convert first the sting text in the same format and later on the string format to date.

4.1.3. Creating the dimensions

Data warehouses are constructed using dimensional data models containing fact and dimension tables. By theory, dimension tables are used to hold the descriptive criteria by which a user can organize the data and they consist of primary keys, values and attributes as well.

Indeed, a dimension table can be expressed by using one or more criteria according to the circumstances. In the current retail sales analysis we defined a snowflake schema in our database as it is best suited when dimension tables store a large number of rows with redundant data, and space is an issue.

Based on the above mentioned, the dimension tables were chosen to be as following:

- Dimension for Customers
- Dimension for Product
- Dimension for Order Date
- Dimension for Ship Date
- Dimension for Priority
- Dimension for City (City- State-Country)
- Dimension for Market-Region (Region-Market)
- Dimension for Ship Mode

Market- Region Dimension

Superstore dataset was initially built in a way to divide the Globe into *Markets* and *Regions*. Some markets are distributed to Regions. More specifically:

- **EU** is divided into South, Central and North Europe
- **EMEA** does not have a region since it is a combining Market
- **LATAM** is divided into Central, Caribbean, North and South
- **Canada** the same with EMEA
- **APAC** is divided into North Asia, Central Asia, Southeast Asia and Oceania.
- **US** is divided into Central , West, South and East.

For simplicity we concatenated the Market and Region labels in order to distinct the Region labels being identical for more than one Market.

MARKET ABBREVIATIONS	
EU	Europe
EMEA	Europe-Middle East-Asia
AFRICA	Africa
LATAM	Latin America
APAC	Asia Pacific
CANADA	Canada
US	US

Table 1. Market accronyms

Subsequently, we create the Region dimension table which is normalized into another related table, the Market table.

Customers Dimension

Superstore needs to have a dimension with all customer details, for example Name and Surname as well as their segment. *Customer name* shows the full name of the customer and *Customer segment* categorizes customers depending on their identity (Home Office, Corporate and Consumer). The last helps the company to understand what customers really want and It also explains how grouping customers into market segments is a good foundation for winning and keeping profitable customers.

Priority and Ship Mode Dimension.

Figure 11. Dimension Region

Priority dimension includes information about how urgent an order is and Ship Mode refers to which method of transportation is used for the delivery.

City Dimension

City dimension includes information regarding the city where the order was delivered. Countries are divided into states and states into cities as well. Concerning these we create the dimension table based on roll-up process which refers to the process of viewing data with decreasing detail.

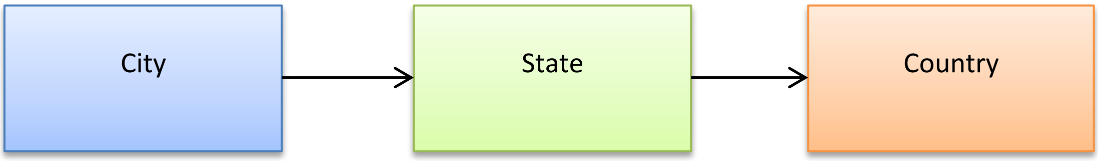


Table 2. City dimension flow

Product Dimension

The Superstore offers a vast amount of products. Each product has a unique name, belongs to a category and each category consists of subcategories. Given that, our approach was similar to the one for creating the dimension city.

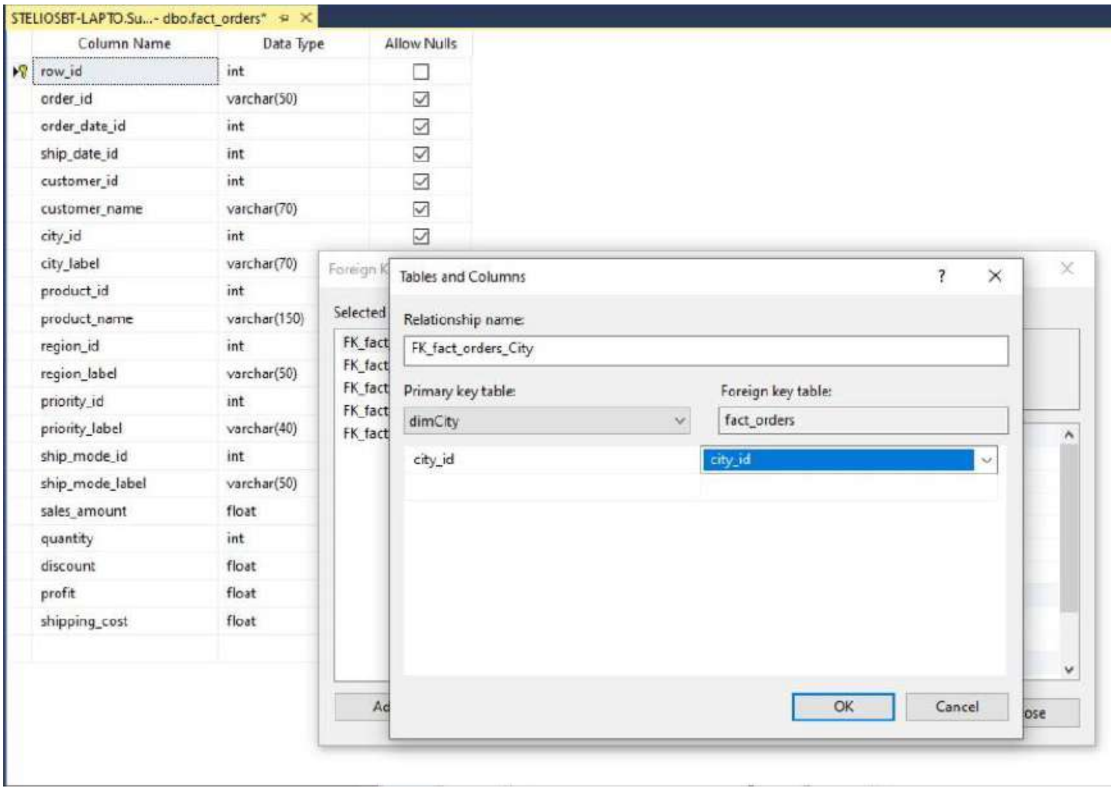
Date Dimension

This dimension includes information about time, in order to make further analysis and monitor the stores' activities by date, month, quarter and year. To be noted that we have two date dimensions regarding order date and ship date.

4.1.4. Creating the fact table

After creating the dimensions we continued with the creation of the fact table (Figure 12. Creating the fact table). Following that, we populated the fact table using the staging_table and dimension tables with the INSERT INTO and the UPDATE function. The relevant queries are in last section (Section 5).

The final step is to define the relationship between the fact table and the dimensions. More thoroughly, the ID columns of the fact table are determined as foreign keys of the dimension tables. Below are some examples of the creation of these relationships (Figure 7 and Figure 8).



STELIOSBT-LAPTO.Su...- dbo.fact_orders*

Column Name	Data Type	Allow Nulls
row_id	int	□
order_id	varchar(50)	☑
order_date_id	int	☑
ship_date_id	int	☑
customer_id	int	☑
customer_name	varchar(70)	☑
city_id	int	☑
city_label	varchar(70)	
product_id	int	
product_name	varchar(150)	
region_id	int	
region_label	varchar(50)	
priority_id	int	
priority_label	varchar(40)	
ship_mode_id	int	
ship_mode_label	varchar(50)	
sales_amount	float	
quantity	int	
discount	float	
profit	float	
shipping_cost	float	

Tables and Columns

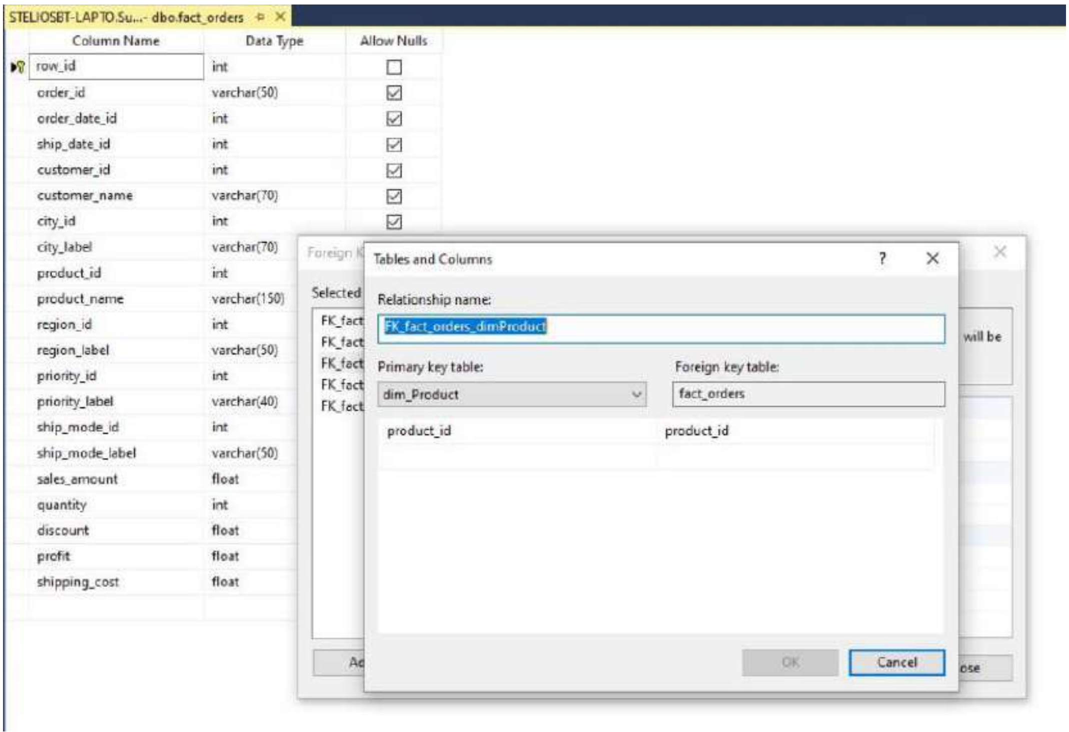
Relationship name: FK_fact_orders_City

Primary key table: dimCity

Foreign key table: fact_orders

Primary key table	Foreign key table
city_id	city_id

Figure 7 . Create relationship between the fact and City Dimension table



STELIOSBT-LAPTO.Su...- dbo.fact_orders

Column Name	Data Type	Allow Nulls
row_id	int	□
order_id	varchar(50)	☑
order_date_id	int	☑
ship_date_id	int	☑
customer_id	int	☑
customer_name	varchar(70)	☑
city_id	int	☑
city_label	varchar(70)	
product_id	int	
product_name	varchar(150)	
region_id	int	
region_label	varchar(50)	
priority_id	int	
priority_label	varchar(40)	
ship_mode_id	int	
ship_mode_label	varchar(50)	
sales_amount	float	
quantity	int	
discount	float	
profit	float	
shipping_cost	float	

Tables and Columns

Relationship name: FK_fact_orders_dimProduct

Primary key table: dim_Product

Foreign key table: fact_orders

Primary key table	Foreign key table
product_id	product_id

Figure 8. Create relationship between the fact and Product Dimension table

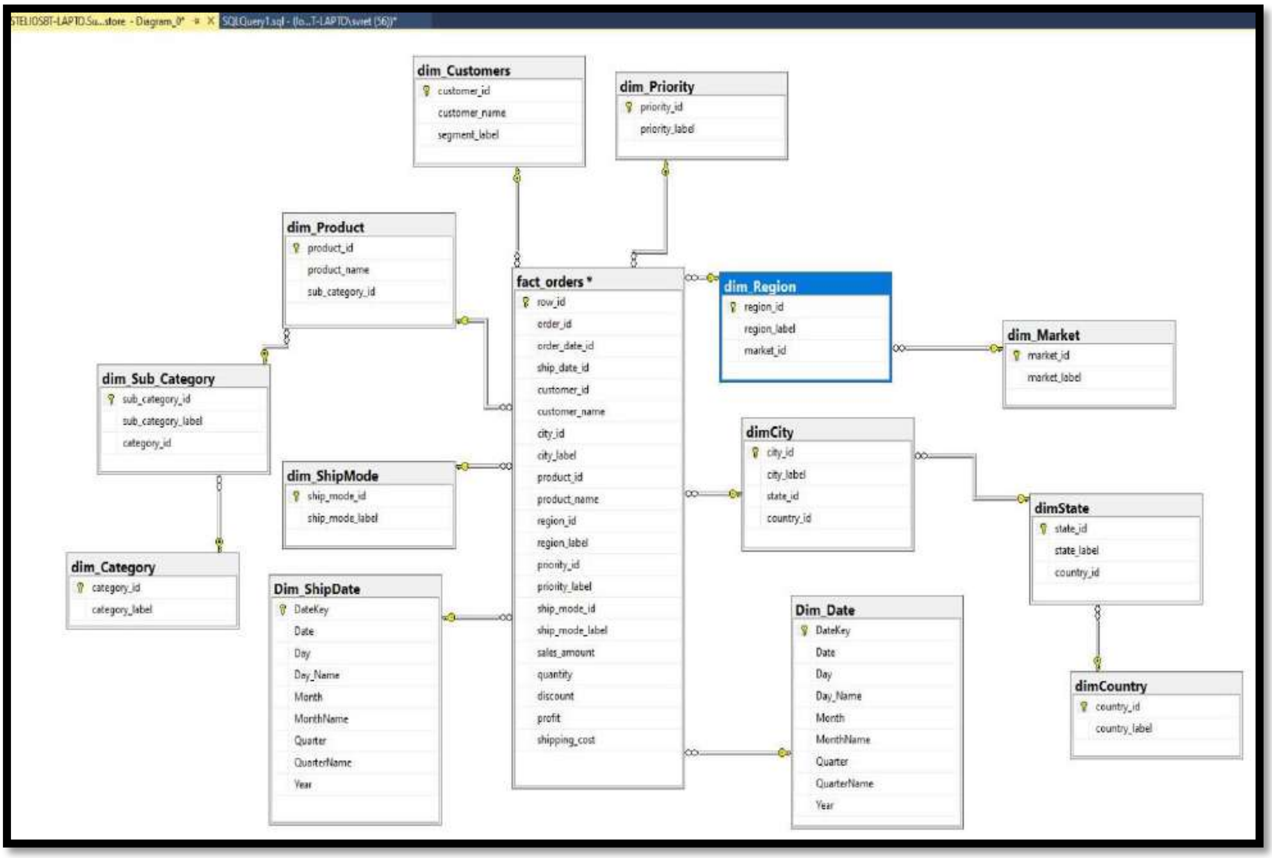
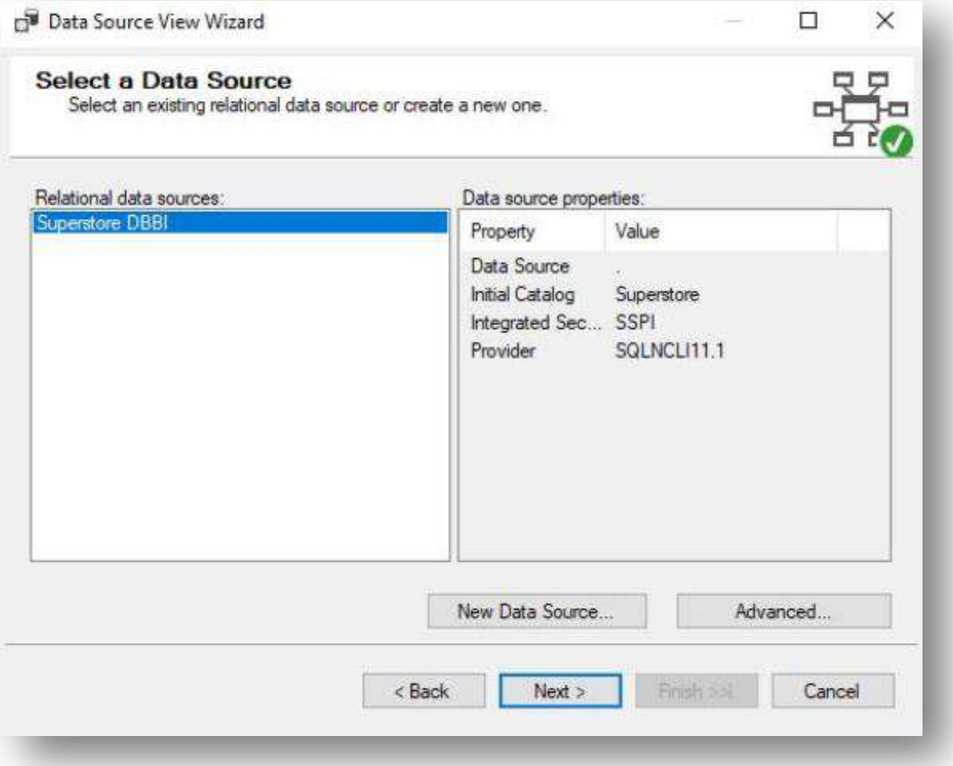


Figure 9. Database Schema

4.2. Build the data Cube

After completing the construction of the Data Warehouse, we designed the cube in Visual Studio using the Multidimensional Analysis Service and Data Mining Tool. Below are the steps followed:

- Creation of the data warehouse project
- Connect the data warehouse with the project
- Create the data cube (define measures of the fact table)
- Add Dimensions and connect them with the cube
- Define Hierarchies
- Deployment and Process



Select a Data Source

Select an existing relational data source or create a new one.

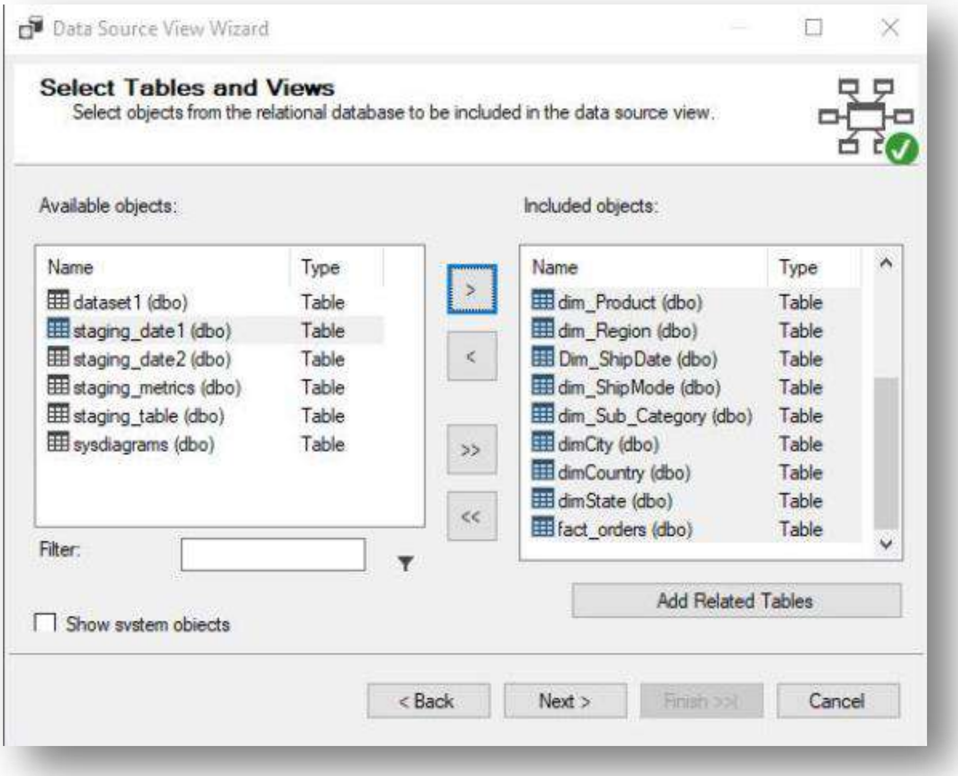
Relational data sources:

Superstore DBBI

Data source properties:

Property	Value
Data Source	.
Initial Catalog	Superstore
Integrated Sec...	SSPI
Provider	SQLNCLI11.1

Figure 10. Connecting the project with the Superstore DB



Select Tables and Views

Select objects from the relational database to be included in the data source view.

Available objects:

Name	Type
dataset1 (dbo)	Table
staging_date1 (dbo)	Table
staging_date2 (dbo)	Table
staging_metrics (dbo)	Table
staging_table (dbo)	Table
sysdiagrams (dbo)	Table

Filter:

□ Show system objects

Included objects:

Name	Type
dim_Product (dbo)	Table
dim_Region (dbo)	Table
Dim_ShipDate (dbo)	Table
dim_ShipMode (dbo)	Table
dim_Sub_Category (dbo)	Table
dimCity (dbo)	Table
dimCountry (dbo)	Table
dimState (dbo)	Table
fact_orders (dbo)	Table

Figure 11. Choosing dimensions for the Cube

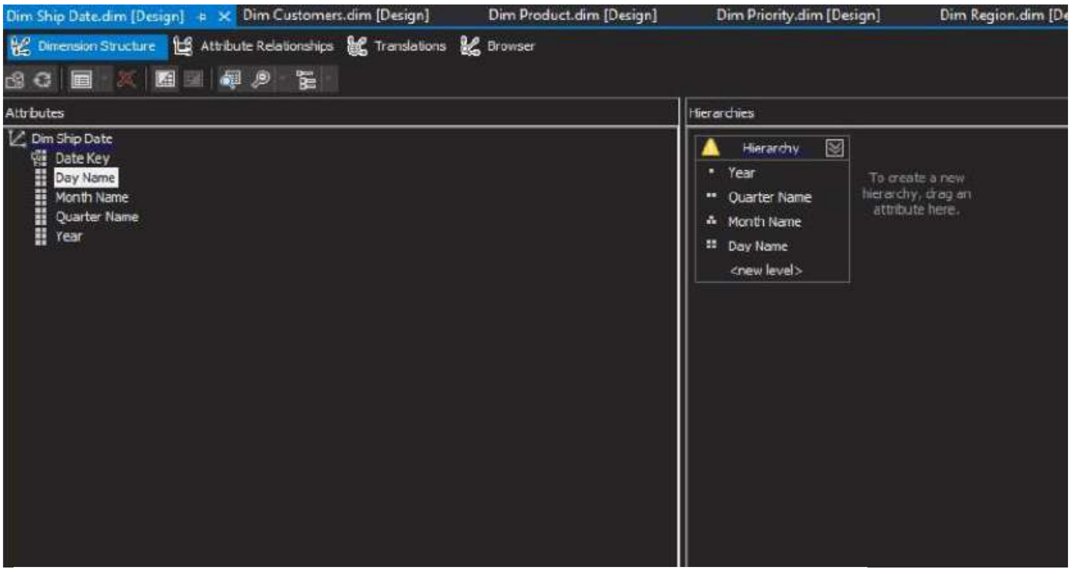


Figure 12. Creating Hierarchies

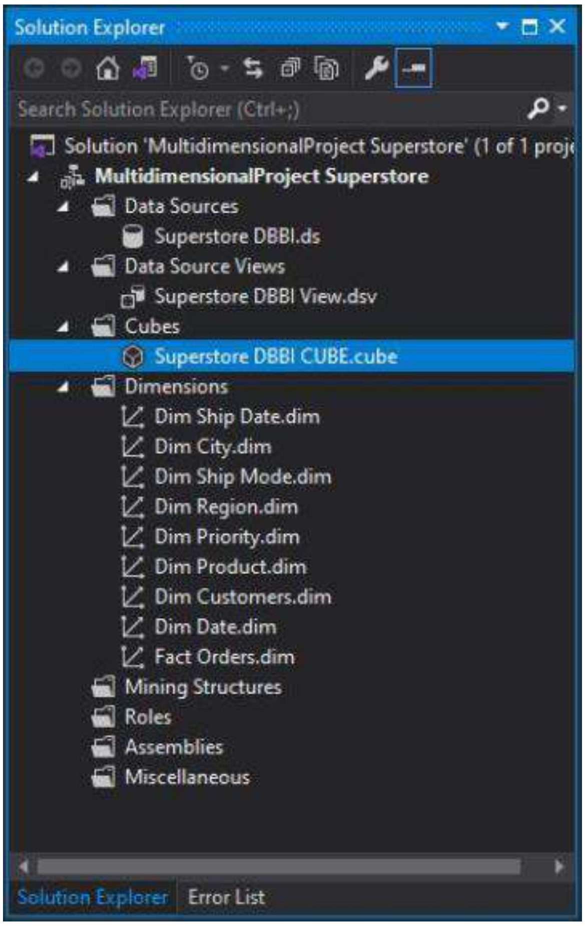


Figure 13. The Cube

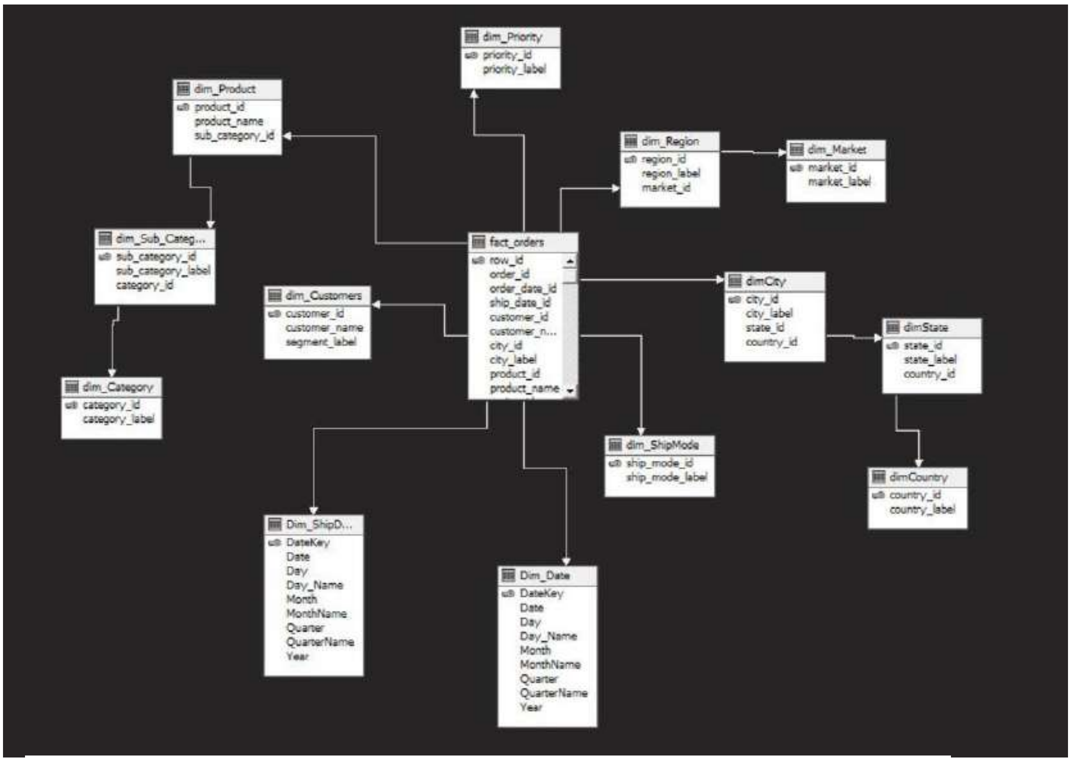


Figure 14. The schema

4.3. OLAP

OLAP operations stand for Online analytical processing server.
The basic OLAP operations are the following:

Drill down: In drill-down operation, the less detailed data is converted into highly detailed data. It can be done by:

- Moving down in the concept hierarchy
- Adding a new dimension

We did a drill down operation when moving down to the product category and found the top 5 sold and profitable phones respectively (Section 4.4).

Roll up: It is just opposite of the drill-down operation. It performs aggregation on the OLAP cube. It can be done by:

- Climbing up in the concept hierarchy
- Reducing the dimensions

When roll up is performed one or more dimensions are removed.

Slice: It selects a single dimension from the cube which results in a new sub-cube creation.

Dice: It selects a sub-cube from the OLAP cube by selecting two or more dimensions.

In our cube a sub-cube is selected by selecting following dimensions with criteria:

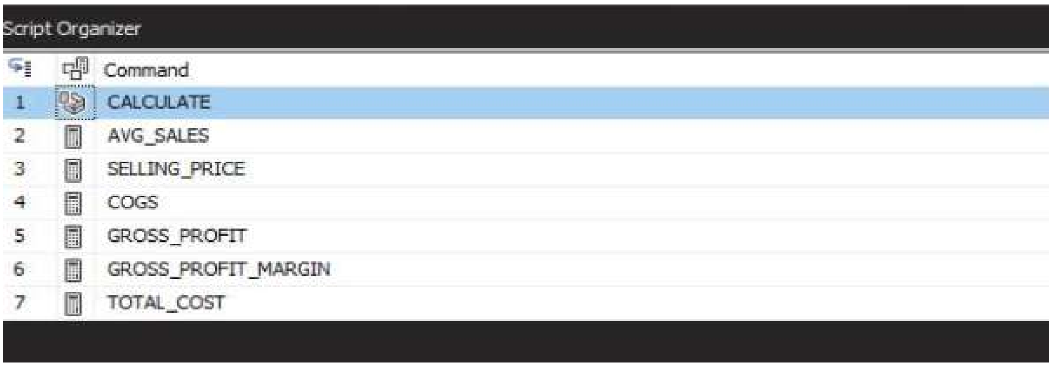
- Location = “France” or “China”
- Time = “Q1” or “Q2”
- Product = “Copiers” or “ Phones”

The above procedure can be described as “slice and dice “

4.3.1. Measures

New measurements have been created such as max for selling price. Some of them are:

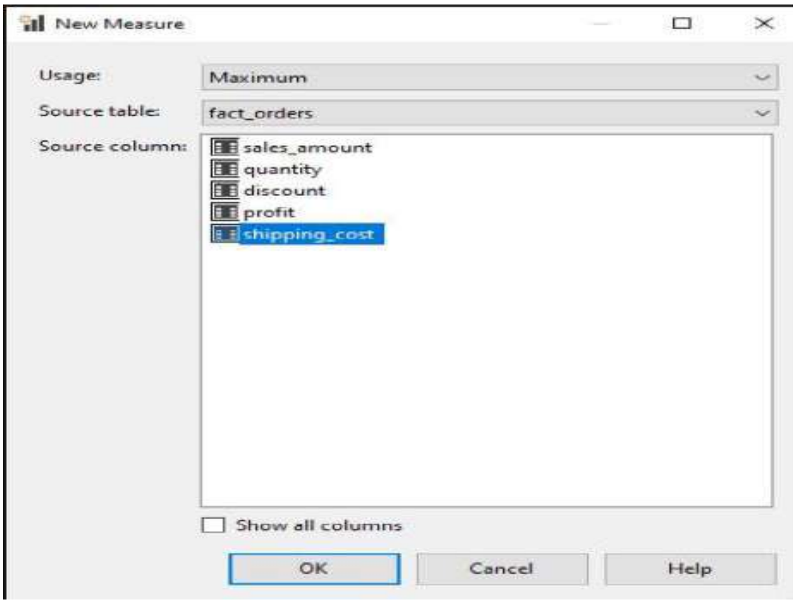
- Avg_Sales
- Selling_Price
- Cogs cost[i1]
- Gross_Profit_Margin
- Total Cost



Script Organizer

	Command
1	CALCULATE
2	AVG_SALES
3	SELLING_PRICE
4	COGS
5	GROSS_PROFIT
6	GROSS_PROFIT_MARGIN
7	TOTAL_COST

Figure 15. New measures



New Measure

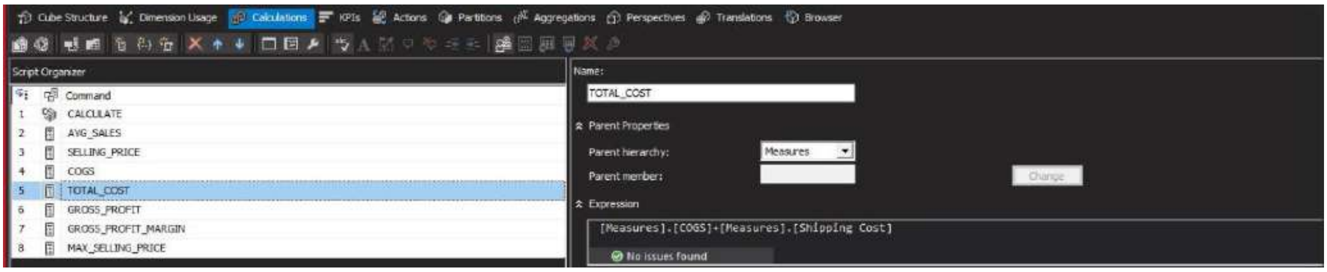
Usage: Maximum

Source table: fact_orders

Source column:
- sales_amount
- quantity
- discount
- profit
- shipping_cost

□ Show all columns

Figure 16. New measures



Script Organizer

	Command
1	CALCULATE
2	AVG_SALES
3	SELLING_PRICE
4	COGS
5	TOTAL_COST
6	GROSS_PROFIT
7	GROSS_PROFIT_MARGIN
8	MAX_SELLING_PRICE

Name: TOTAL_COST

Parent Properties

Parent hierarchy: Measures

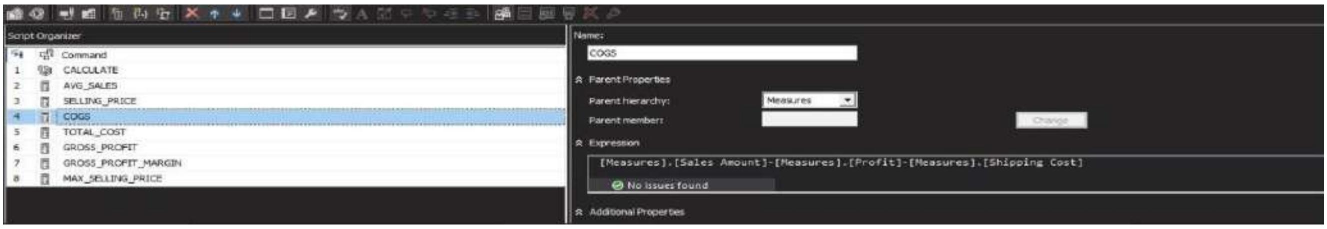
Parent member:

Expression

```
[Measures].[COGS]+[Measures].[Shipping Cost]
```

No issues found

Figure 17. New measures



Script Organizer

	Command
1	CALCULATE
2	AVG_SALES
3	SELLING_PRICE
4	COGS
5	TOTAL_COST
6	GROSS_PROFIT
7	GROSS_PROFIT_MARGIN
8	MAX_SELLING_PRICE

Name: COGS

Parent Properties

Parent hierarchy: Measures

Parent member:

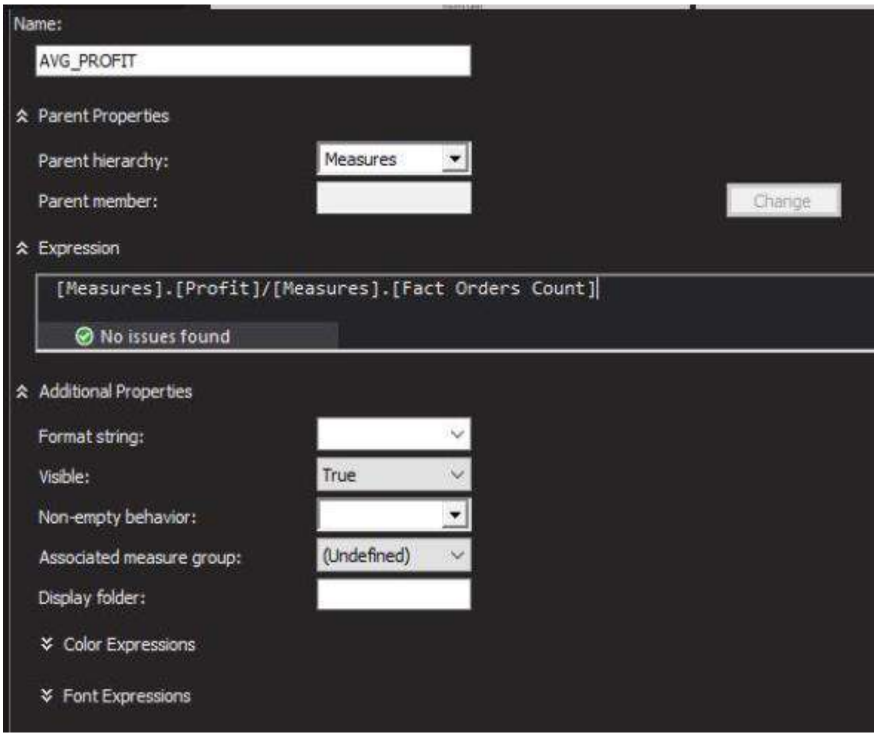
Expression

```
[Measures].[Sales Amount]-[Measures].[Profit]-[Measures].[Shipping Cost]
```

No issues found

Additional Properties

Figure 18. New measures



Name: AVG_PROFIT

Parent Properties

Parent hierarchy: Measures

Parent member:

Expression

```
[Measures].[Profit]/[Measures].[Fact Orders Count]
```

No issues found

Additional Properties

Format string:

Visible: True

Non-empty behavior:

Associated measure group: (Undefined)

Display folder:

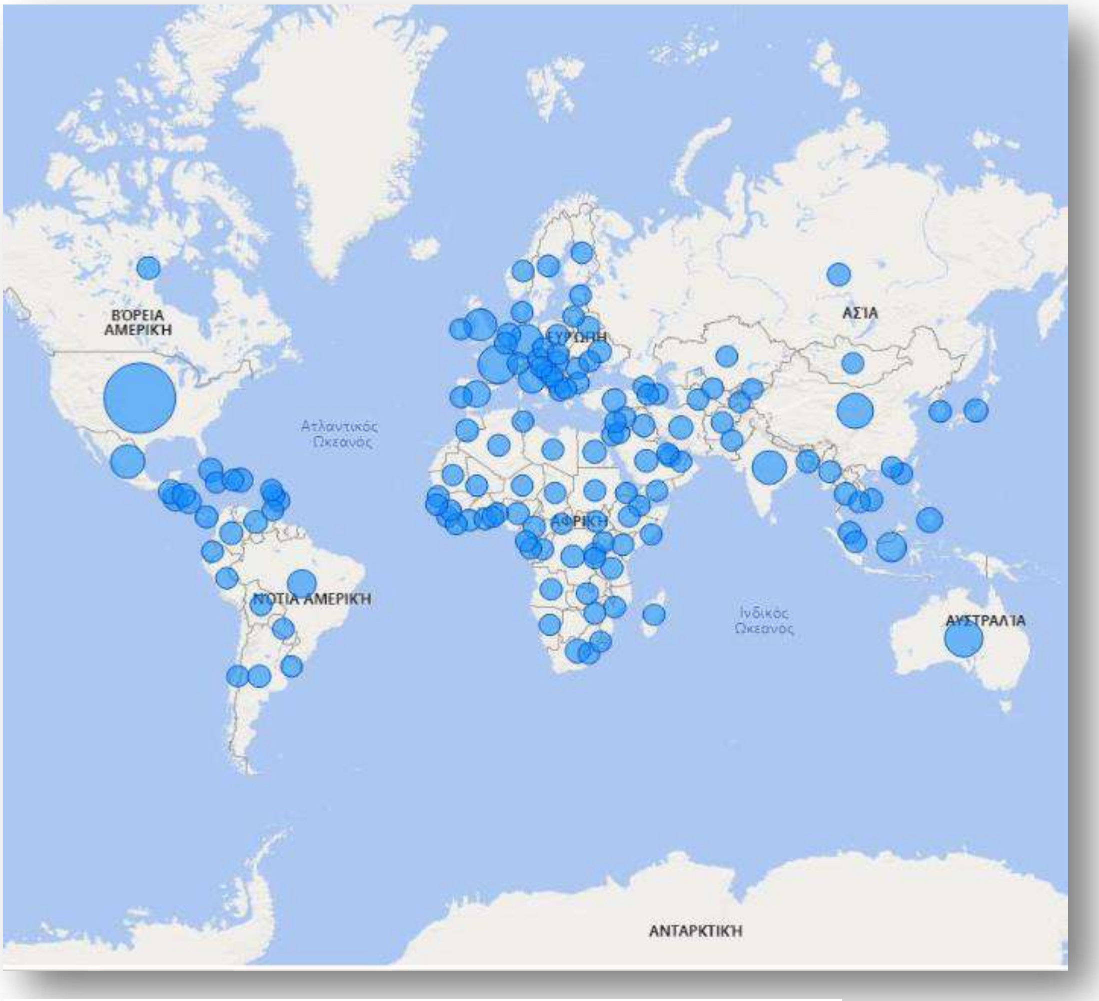
Color Expressions

Font Expressions

Figure 19. New measures

4.4. Visualizations

Below are some of the visuals created in Power BI:



Visual 1. Sales distributed for each country.

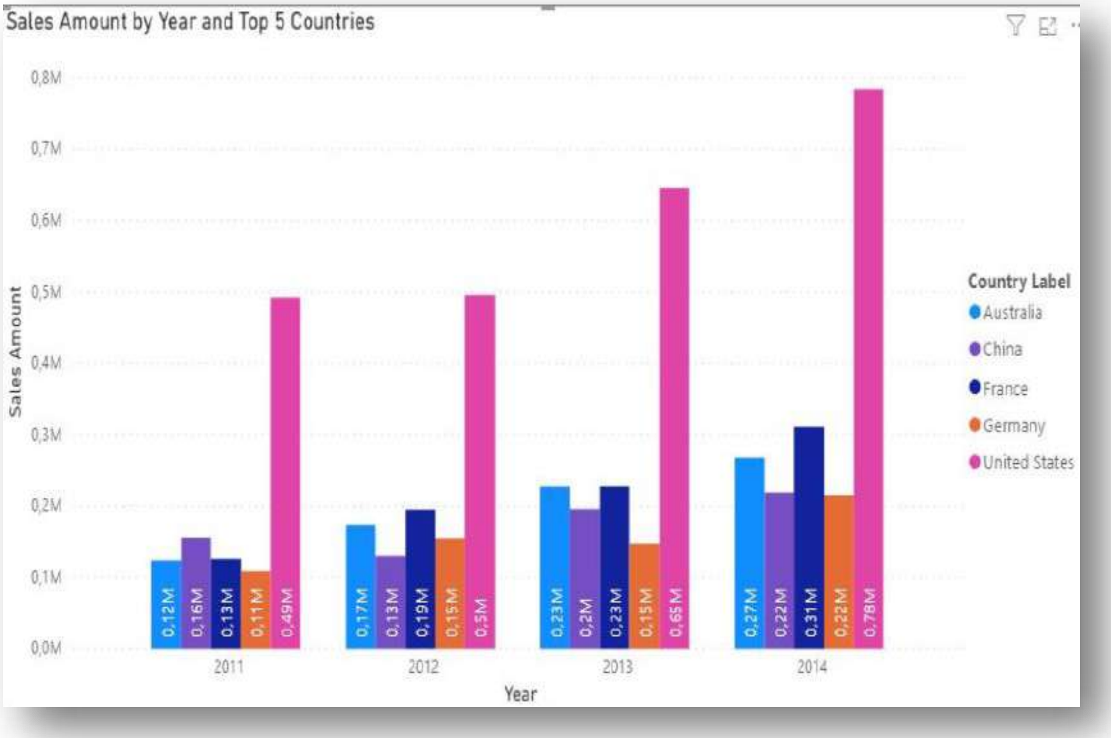
In the above graph (Visual 1) we see the sales distributed for each country.

Each bubble represents the portion of total global sales.

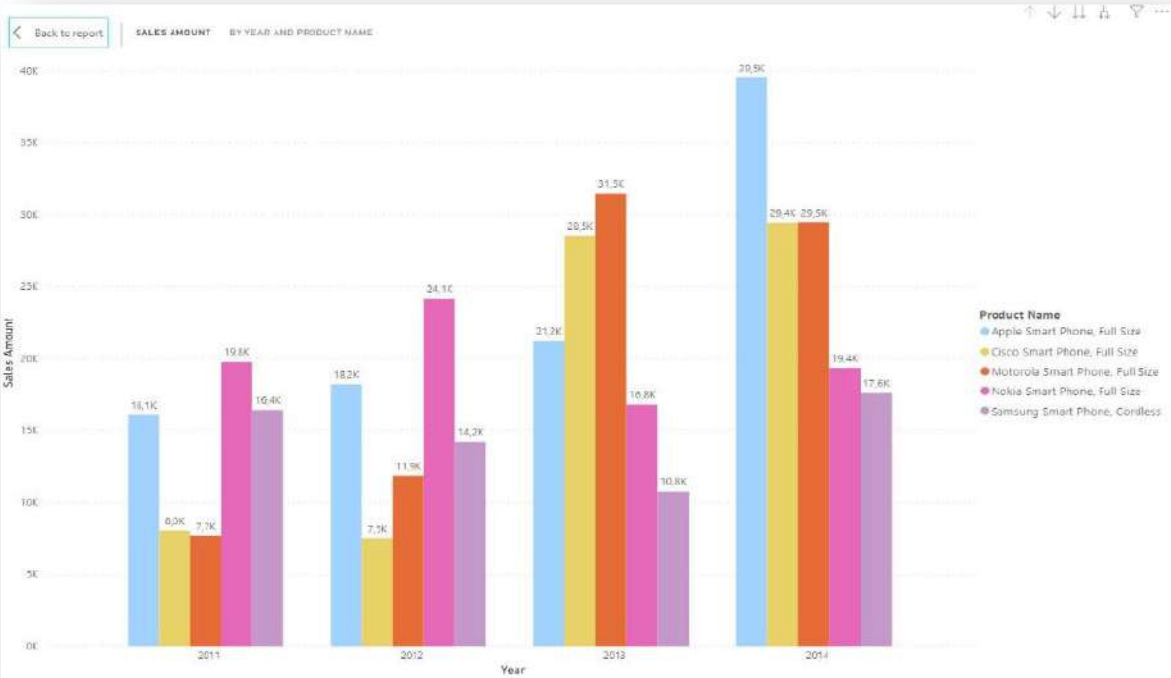
The **bigger** a bubble is the more this country contributes to sales.

Interpretation:

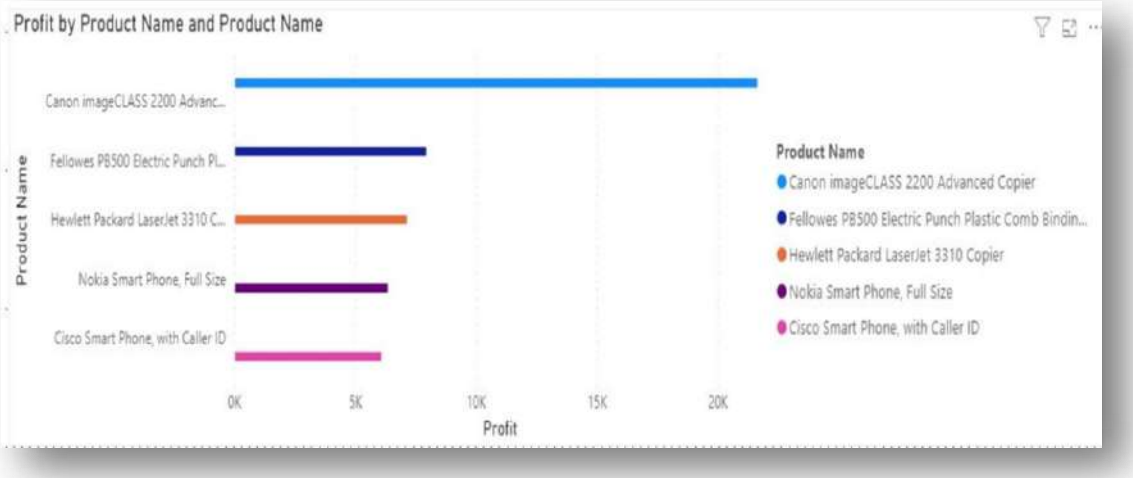
We see that US, China, France, Germany and Australia are the countries with the most impact.



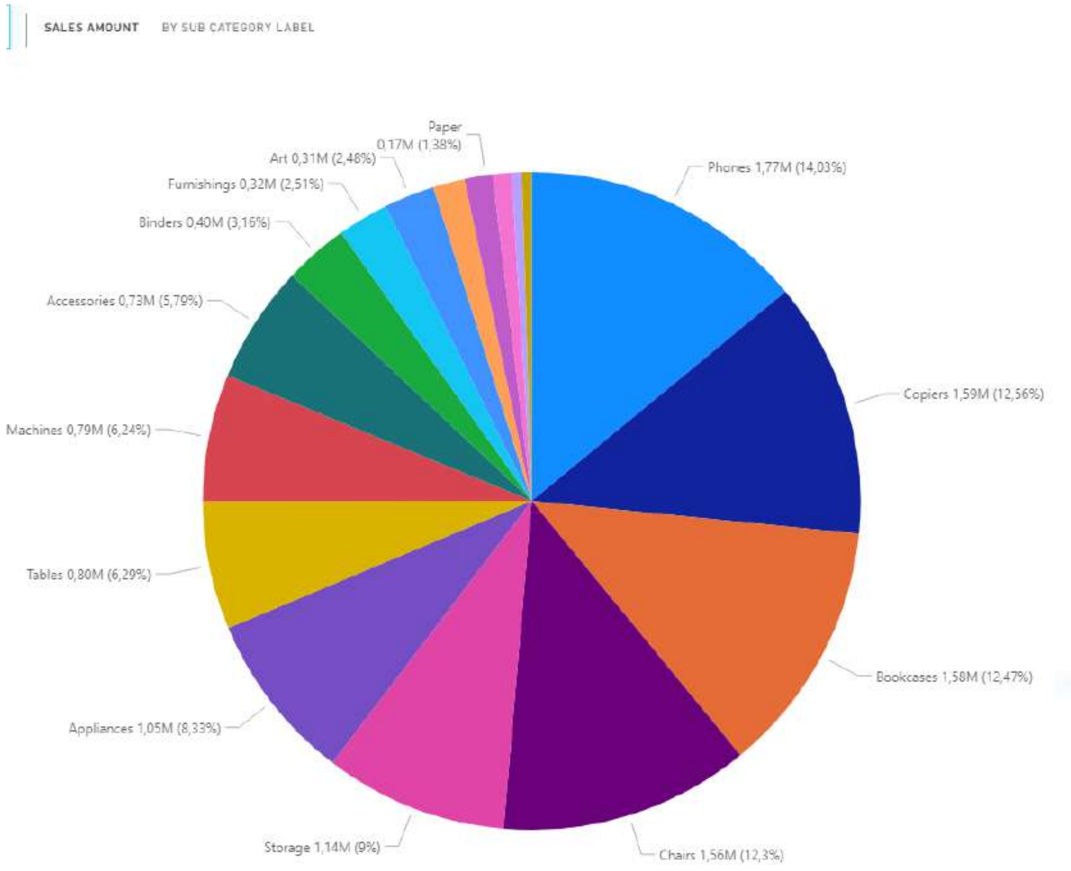
Visual 2. Sales amount by year for Top 5 Countries



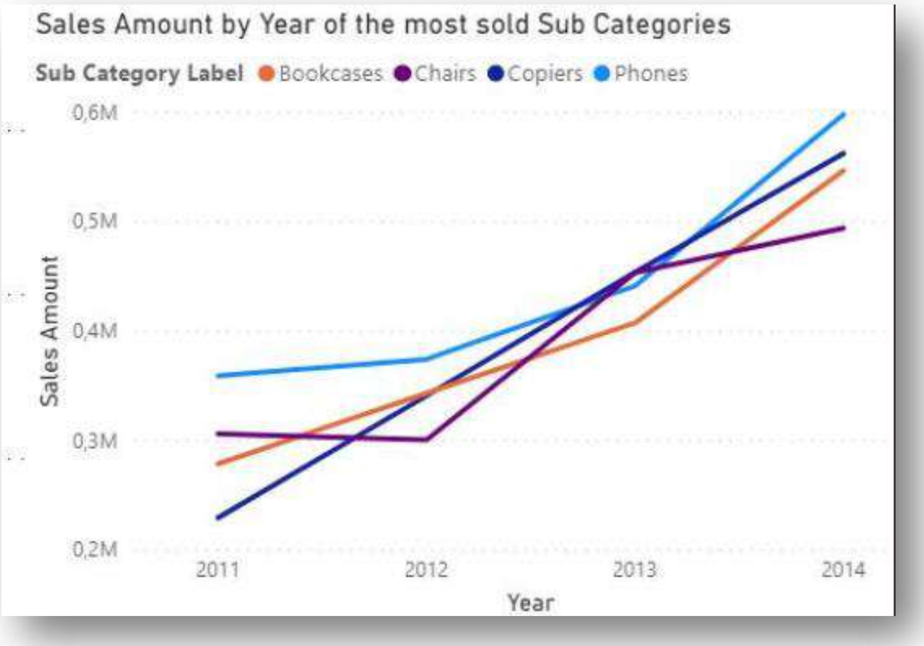
Visual 3. Sales amount by year for Top 5 Countries



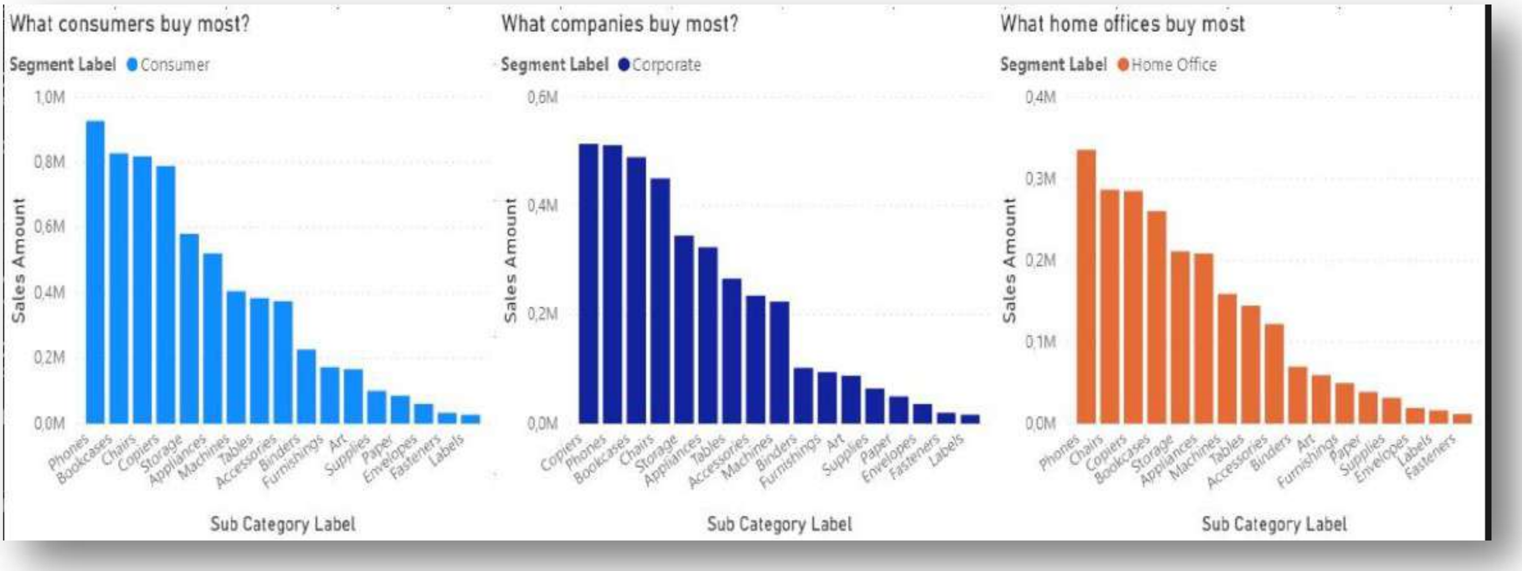
Visual 4. Sales amount by year for Top 5 Countries



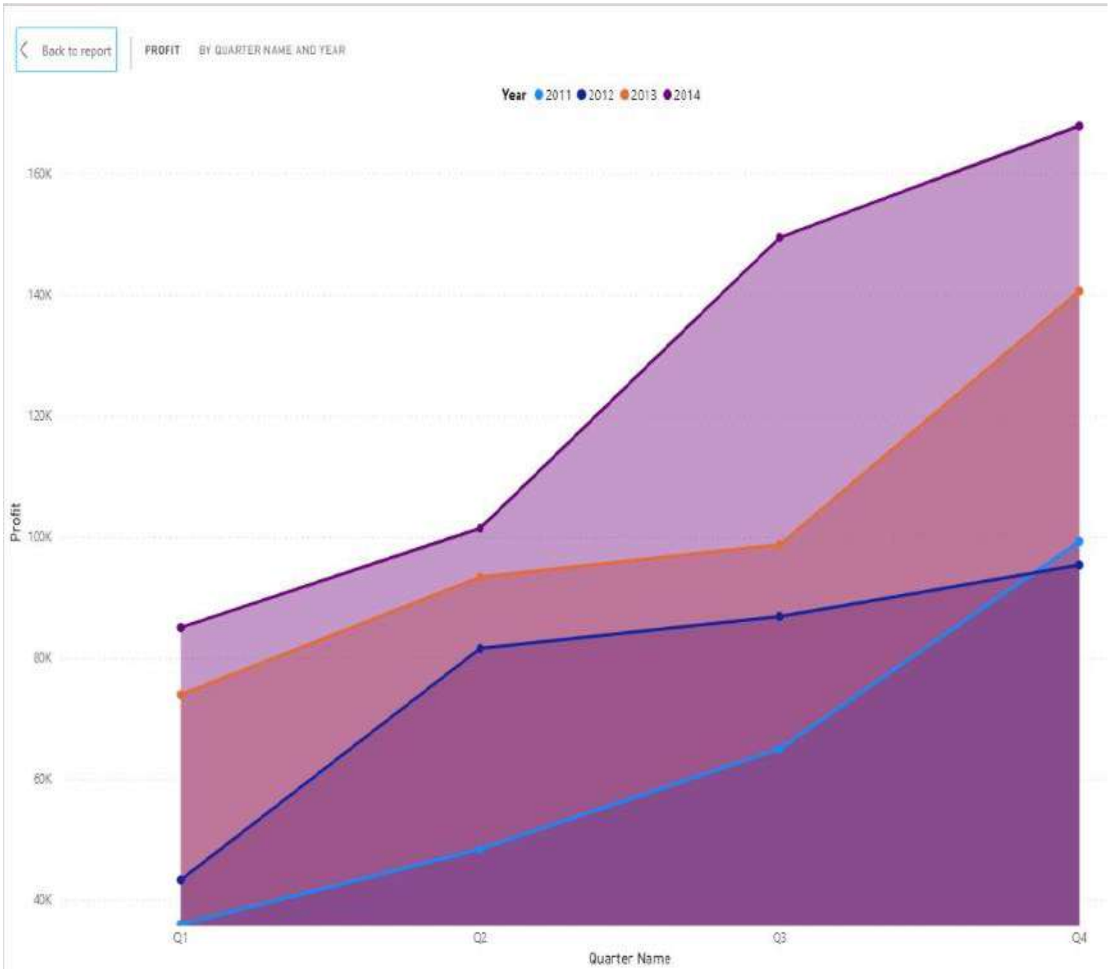
Visual 5. Sales amount by Sub-Category



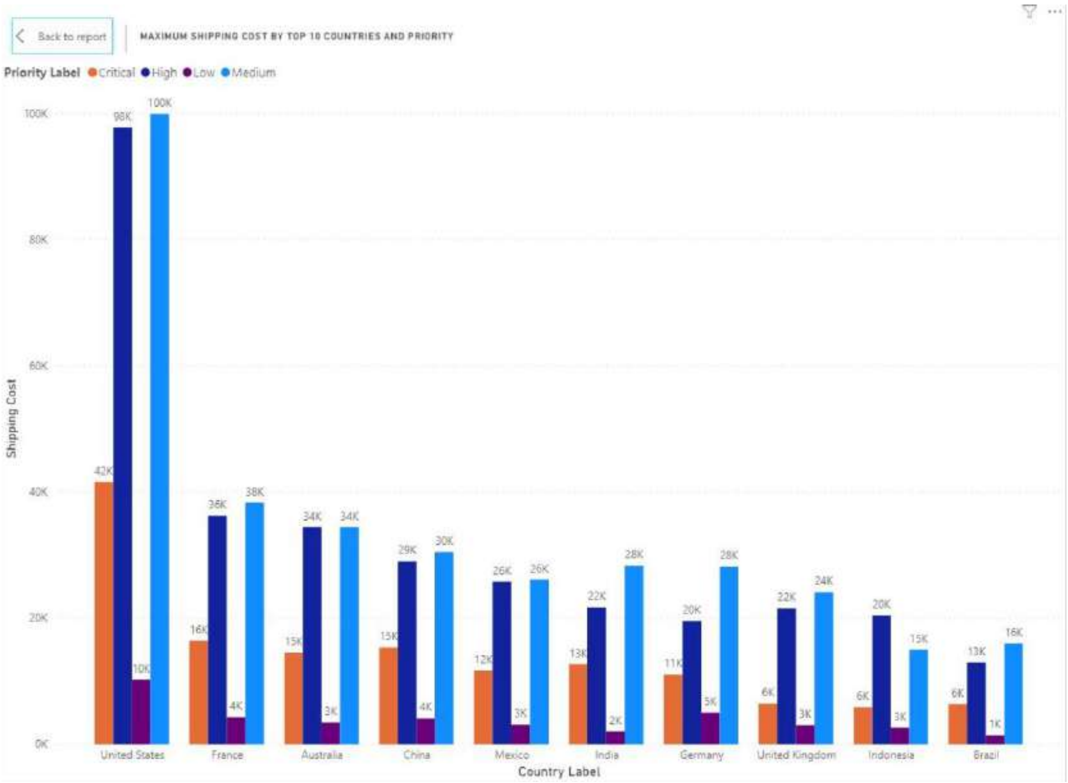
Visual 6. Sales amount by Sub-Category during time



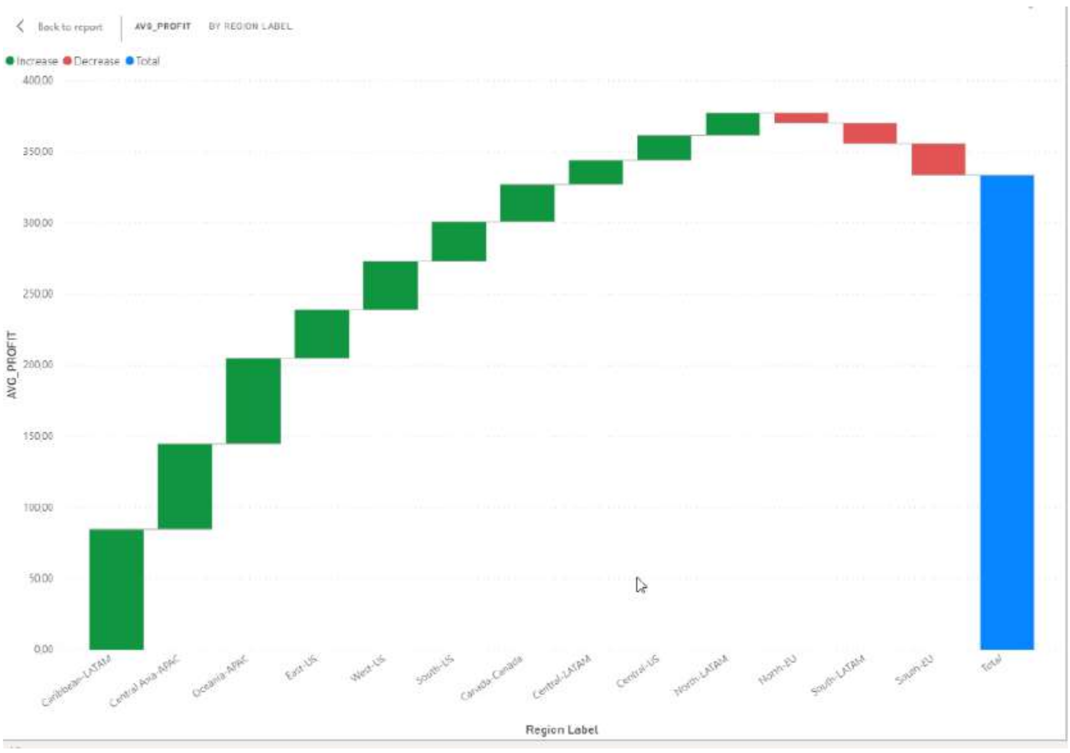
Visual 7. Sales by customer segments



Visual 8. Profit per year



Visual 9. Top 10 Countries with most Shipping Cost grouped by order priority



Visual 10. Average profit by region

Conclusions

1. Most sales come from the Apple Smart Phone in 2014 and has the most significant increase during the years.
2. The most profitable product is by far the Canon imageClass 2200 Advanced Copier.
3. Technology products have the greater demand, meaning more profit for the company.
4. US, China, France, Germany and Australia are the countries with the most impact.
5. Almost half of total sales come from
   - Phones (14,03%)
   - Copiers (12,56%)
   - Bookcases (12,47%)
   - Chairs (12,3%)
6. Consumers and home offices seem to buy mostly phones
7. Companies invest more on copiers and phones as well
8. Profit increases rapidly from 2011 to 2014.
9. High and Medium order priority seems to have more shipping cost for all 10 countries with most shipping cost.
10. Biggest average profit per order has the Caribbean-LATAM region and biggest negative South-EU.

5. Code

```
Use superstore;
alter table staging_table drop column Postal_Code
select * from staging_table where Row_ID is null
select * from staging_table where Order_Date is null
select * from staging_table where Ship_Date is null
select * from staging_table where Ship_Mode is null
select * from staging_table where Customer_ID is null
select * from staging_table where Customer_Name is null
select * from staging_table where Segment is null
select * from staging_table where City is null
select * from staging_table where State is null
select * from staging_table where Country is null
select * from staging_table where Market is null
select * from staging_table where Region is null
select * from staging_table where Product_ID is null
select * from staging_table where Product_Name is null
select * from staging_table where Category is null
select * from staging_table where Sub_Category is null
select * from staging_table where Sales is null
select * from staging_table where Quantity is null
select * from staging_table where Discount is null
select * from staging_table where Profit is null
select * from staging_table where Shipping_Cost is null
select * from staging_table where Order_Priority is null


create table staging_metrics (
row_id int not null identity(1,1) primary key,
sales_amount float,
quantity int,
discount float,
profit_amount float,
shipping_cost float
);






SELECT round(CAST(CAST (Sales AS NUMERIC(19,4)) AS float),2) as sales_amount,
                             CAST(CAST (Quantity AS NUMERIC(19,4)) AS int) as
quantity,
            round(CAST(CAST (Discount AS NUMERIC(19,4)) AS float),2) as
discount,
            round(CAST(CAST (Profit AS NUMERIC(19,4)) AS float),2)
as profit_amount,
            round(CAST(CAST (Shipping_Cost AS NUMERIC(19,4)) AS float),2)
as shipping_cost
from staging_table --- works

insert into staging_metrics(sales_amount, quantity,
discount, profit_amount, shipping_cost)
select t.sales_amount, t.quantity, t.discount, t.profit_amount, t.shipping_cost
from (
SELECT round(CAST(CAST (Sales AS NUMERIC(19,4)) AS float),2) as sales_amount,
                             CAST(CAST (Quantity AS NUMERIC(19,4)) AS int) as
quantity,
            round(CAST(CAST (Discount AS NUMERIC(19,4)) AS float),2) as
discount,
            round(CAST(CAST (Profit AS NUMERIC(19,4)) AS float),2)
as profit_amount,
            round(CAST(CAST (Shipping_Cost AS NUMERIC(19,4)) AS float),2)
as shipping_cost
```

```
from staging_table) t

select * from staging_metrics


select * from staging_metrics











create table staging_date1 (
date_id int not null identity(1,1) primary key,
order_date date,
);

insert into staging_date1(order_date)
select convert(date, t.order_date_final, 103) from (
SELECT REPLACE(Order_Date, '-', '/') as staging_date,

charindex('/',REPLACE(Order_Date, '-', '/')) as position_ist,--
 find the position of the first /

substring(REPLACE(Order_Date, '-', '/'),1,(charindex('/',REPLACE(Order_Date, '-
', '/')) ) ) as order_day, --
select the characters in field Order_Date from the beggining of the field until 
the first /

RIGHT('00'+convert(varchar,substring(REPLACE(Order_Date, '-
', '/'),1,(charindex('/',REPLACE(Order_Date, '-', '/')) -
1) )),2) as Order_day_dd_format,--convert the above field to dd format

SUBSTRING(REPLACE(Order_Date, '-', '/'), charindex('/',REPLACE(Order_Date, '-
', '/'))+1,len(REPLACE(Order_Date, '-', '/'))) as remaining_text,--
 select the text folowing the first /,

charindex('/', (SUBSTRING(REPLACE(Order_Date, '-
', '/'), charindex('/',REPLACE(Order_Date, '-
', '/'))+1,len(REPLACE(Order_Date, '-', '/')))) ) as position_2nd, --
 find the position of the first / of field remaining_text

SUBSTRING((SUBSTRING(REPLACE(Order_Date, '-
', '/'), charindex('/',REPLACE(Order_Date, '-
', '/'))+1,len(REPLACE(Order_Date, '-
', '/')))),1,(charindex('/', (SUBSTRING(REPLACE(Order_Date, '-
', '/'), charindex('/',REPLACE(Order_Date, '-
', '/'))+1,len(REPLACE(Order_Date, '-', '/')))) )-1)) AS Order_month,

RIGHT('00'+CONVERT(VARCHAR,SUBSTRING((SUBSTRING(REPLACE(Order_Date, '-
', '/'), charindex('/',REPLACE(Order_Date, '-
', '/'))+1,len(REPLACE(Order_Date, '-
', '/')))),1,(charindex('/', (SUBSTRING(REPLACE(Order_Date, '-
', '/'), charindex('/',REPLACE(Order_Date, '-
', '/'))+1,len(REPLACE(Order_Date, '-', '/')))) )-
1))),2) AS Order_month_mm_format,



substring(SUBSTRING(REPLACE(Order_Date, '-
', '/'), charindex('/',REPLACE(Order_Date, '-
', '/'))+1,len(REPLACE(Order_Date, '-
', '/'))),charindex('/', (SUBSTRING(REPLACE(Order_Date, '-
', '/'), charindex('/',REPLACE(Order_Date, '-
', '/'))+1,len(REPLACE(Order_Date, '-
', '/')))) )+1,LEN(SUBSTRING(REPLACE(Order_Date, '-
', '/'), charindex('/',REPLACE(Order_Date, '-
', '/'))+1,len(REPLACE(Order_Date, '-', '/'))))) as Order_year,


RIGHT('00'+convert(varchar,substring(REPLACE(Order_Date, '-
', '/'),1,(charindex('/',REPLACE(Order_Date, '-', '/')) -1) )),2)  + '/'+

RIGHT('00'+CONVERT(VARCHAR,SUBSTRING((SUBSTRING(REPLACE(Order_Date, '-
', '/'), charindex('/',REPLACE(Order_Date, '-
', '/'))+1,len(REPLACE(Order_Date, '-
', '/')))),1,(charindex('/', (SUBSTRING(REPLACE(Order_Date, '-
', '/'), charindex('/',REPLACE(Order_Date, '-
', '/'))+1,len(REPLACE(Order_Date, '-', '/')))) )-1))),2) + '/'+

substring(SUBSTRING(REPLACE(Order_Date, '-
', '/'), charindex('/',REPLACE(Order_Date, '-
', '/'))+1,len(REPLACE(Order_Date, '-
', '/'))),charindex('/', (SUBSTRING(REPLACE(Order_Date, '-
', '/'), charindex('/',REPLACE(Order_Date, '-
', '/'))+1,len(REPLACE(Order_Date, '-
', '/')))) )+1,LEN(SUBSTRING(REPLACE(Order_Date, '-
', '/'), charindex('/',REPLACE(Order_Date, '-
', '/'))+1,len(REPLACE(Order_Date, '-', '/'))))) as order_date_final

from staging_table) t

create table staging_date2 (
date_id int not null identity(1,1) primary key,
ship_date date
);

insert into staging_date2(ship_date)
select convert(date, t.Ship_Date_final, 103) from (
SELECT REPLACE(Ship_Date, '-', '/') as staging_date,

charindex('/',REPLACE(Ship_Date, '-', '/')) as position_ist,--
 find the position of the first /

substring(REPLACE(Ship_Date, '-', '/'),1,(charindex('/',REPLACE(Ship_Date, '-
', '/')) ) ) as order_day, --
select the characters in field Ship_Date from the beggining of the field until t
he first /

RIGHT('00'+convert(varchar,substring(REPLACE(Ship_Date, '-
', '/'),1,(charindex('/',REPLACE(Ship_Date, '-', '/')) -
1) )),2) as Order_day_dd_format,--convert the above field to dd format



SUBSTRING(REPLACE(Ship_Date, '-', '/'), charindex('/',REPLACE(Ship_Date, '-
', '/'))+1,len(REPLACE(Ship_Date, '-', '/'))) as remaining_text,--
 select the text folowing the first /,

charindex('/', (SUBSTRING(REPLACE(Ship_Date, '-
', '/'), charindex('/',REPLACE(Ship_Date, '-', '/'))+1,len(REPLACE(Ship_Date, '-
', '/')))) ) as position_2nd, --
 find the position of the first / of field remaining_text

SUBSTRING((SUBSTRING(REPLACE(Ship_Date, '-
', '/'), charindex('/',REPLACE(Ship_Date, '-', '/'))+1,len(REPLACE(Ship_Date, '-
', '/')))),1,(charindex('/', (SUBSTRING(REPLACE(Ship_Date, '-
', '/'), charindex('/',REPLACE(Ship_Date, '-', '/'))+1,len(REPLACE(Ship_Date, '-
', '/')))) )-1)) AS Order_month,

RIGHT('00'+CONVERT(VARCHAR,SUBSTRING((SUBSTRING(REPLACE(Ship_Date, '-
', '/'), charindex('/',REPLACE(Ship_Date, '-', '/'))+1,len(REPLACE(Ship_Date, '-
', '/')))),1,(charindex('/', (SUBSTRING(REPLACE(Ship_Date, '-
', '/'), charindex('/',REPLACE(Ship_Date, '-', '/'))+1,len(REPLACE(Ship_Date, '-
', '/')))) )-1))),2) AS Order_month_mm_format,



substring(SUBSTRING(REPLACE(Ship_Date, '-
', '/'), charindex('/',REPLACE(Ship_Date, '-', '/'))+1,len(REPLACE(Ship_Date, '-
', '/'))),charindex('/', (SUBSTRING(REPLACE(Ship_Date, '-
', '/'), charindex('/',REPLACE(Ship_Date, '-', '/'))+1,len(REPLACE(Ship_Date, '-
', '/')))) )+1,LEN(SUBSTRING(REPLACE(Ship_Date, '-
', '/'), charindex('/',REPLACE(Ship_Date, '-', '/'))+1,len(REPLACE(Ship_Date, '-
', '/'))))) as Order_year,


RIGHT('00'+convert(varchar,substring(REPLACE(Ship_Date, '-
', '/'),1,(charindex('/',REPLACE(Ship_Date, '-', '/')) -1) )),2)  + '/'+

RIGHT('00'+CONVERT(VARCHAR,SUBSTRING((SUBSTRING(REPLACE(Ship_Date, '-
', '/'), charindex('/',REPLACE(Ship_Date, '-', '/'))+1,len(REPLACE(Ship_Date, '-
', '/')))),1,(charindex('/', (SUBSTRING(REPLACE(Ship_Date, '-
', '/'), charindex('/',REPLACE(Ship_Date, '-', '/'))+1,len(REPLACE(Ship_Date, '-
', '/')))) )-1))),2) + '/'+

substring(SUBSTRING(REPLACE(Ship_Date, '-
', '/'), charindex('/',REPLACE(Ship_Date, '-', '/'))+1,len(REPLACE(Ship_Date, '-
', '/'))),charindex('/', (SUBSTRING(REPLACE(Ship_Date, '-
', '/'), charindex('/',REPLACE(Ship_Date, '-', '/'))+1,len(REPLACE(Ship_Date, '-
', '/')))) )+1,LEN(SUBSTRING(REPLACE(Ship_Date, '-
', '/'), charindex('/',REPLACE(Ship_Date, '-', '/'))+1,len(REPLACE(Ship_Date, '-
', '/'))))) as Ship_Date_final

from staging_table) t




select * from staging_date1
select * from staging_date2






create table dim_Order (
    row_id int not null identity(1,1) primary key,
    order_id varchar(50) not null
);

insert into dim_Order(order_id)
select t.order_id from (select order_id from staging_table) t

select * from dim_Order


create table dim_Priority (
    priority_id int not null identity(1,1) primary key,
    priority_label varchar(50) not null
);
```

```
insert into dim_Priority(priority_label)
select distinct Order_Priority from staging_table



create table dim_ShipMode (
    ship_mode_id int not null identity(1,1) primary key,
    ship_mode_label varchar(50) not null
);

insert into dim_ShipMode(ship_mode_label)
select distinct Ship_Mode from staging_table

create table dim_Category(
category_id int not null identity(1,1) primary key,
category_label varchar(50))

create table dim_Sub_Category(
sub_category_id int not null identity(1,1) primary key,
sub_category_label varchar(140) not null,
category_id int foreign key references dim_Category(category_id))

create table dim_Product(
product_id int not null identity(1,1) primary key,
product_name varchar(170),
sub_category_id int foreign key references dim_Sub_Category(sub_category_id))


insert into dim_Category(category_label) select distinct category from staging_t
able

insert into dim_Sub_Category (sub_category_label, category_id)
select *  from
(select distinct t1.Sub_Category, t2.category_id from staging_table t1 join dim_
Category t2 on t1.Category = t2.category_label) t

insert into dim_Product (product_name, sub_category_id)
select * from
(select distinct t1.Product_Name, t2.sub_category_id from staging_table t1 join
dim_Sub_Category t2 on t1.Sub_Category = t2.sub_category_label) t


use superstore

CREATE TABLE dbo.Dim_Date (
   DateKey INT NOT NULL PRIMARY KEY,
   [Date] DATE NOT NULL,
   [Day] TINYINT NOT NULL,
   [Day_Name] VARCHAR(20) NOT NULL,
   [Month] TINYINT NOT NULL,
   [MonthName] VARCHAR(15) NOT NULL,
   [Quarter] TINYINT NOT NULL,
   [QuarterName] VARCHAR(6) NOT NULL,
   [Year] INT NOT NULL,
)

insert into Dim_Date (DateKey, date, Day, Day_Name, Month, MonthName,
Quarter, QuarterName, Year)
select date_id, order_date,
day(order_date) day_id, datename(dw,order_date) date_name,
                   month(order_date) month_id, datename(MONTH,order_date) month_n
ame,
                   datename(QUARTER,order_date) quarter_id,
                  concat ('Q',datename(QUARTER,order_date)) quarter_name,
year(order_date) year
from staging_date1

select * from dim_Date


------------------ time dimension - ship_date --------------------------------

CREATE TABLE dbo.Dim_ShipDate (
   DateKey INT NOT NULL PRIMARY KEY,
   [Date] DATE NOT NULL,
   [Day] TINYINT NOT NULL,
   [Day_Name] VARCHAR(20) NOT NULL,
   [Month] TINYINT NOT NULL,
   [MonthName] VARCHAR(15) NOT NULL,
   [Quarter] TINYINT NOT NULL,
   [QuarterName] VARCHAR(6) NOT NULL,
   [Year] INT NOT NULL,
)


insert into Dim_ShipDate (DateKey, date, Day, Day_Name, Month, MonthName,
Quarter, QuarterName, Year)
select date_id, Ship_Date, day(Ship_Date) day_id, datename(dw,Ship_Date) date_na
me,
                  month(Ship_Date) month_id, datename(MONTH,Ship_Date) month_nam
e,
                  datename(QUARTER,Ship_Date) quarter_id,
                  concat ('Q',datename(QUARTER,Ship_Date)) quarter_name,
year(Ship_Date) year
from staging_date2

select * from Dim_Date
select * from Dim_ShipDate


create table dim_Customers (
    customer_id int not null identity(1,1) primary key,
    customer_name varchar(50) not null,
    segment_label varchar(50) not null
);



insert into dim_Customers(customer_name, segment_label ) (select distinct Custom
er_Name, Segment from staging_table);

select * from dim_Customers


create table dim_Country (
country_id int not null identity(1,1) primary key,
country_label varchar(150)
);

create table dim_State (
state_id int not null identity(1,1) primary key,
state_label varchar(150),
country_id int not null foreign key references dim_Country(country_id)
);

create table dim_City (
city_id int not null identity(1,1) primary key,
city_label varchar(150),
state_id int foreign key references dim_State (state_id),
country_id int foreign key references dim_Country(country_id)
);




insert into dim_Country(country_label) select distinct Country
from staging_table

insert into dim_State(state_label, country_id) select *
from (select distinct State, t2.country_id from staging_table t1
join dim_Country t2 on t1.Country = t2.country_label)t

insert into dim_City(city_label, state_id, country_id) select *
from (select distinct t1.City, t2.state_id, t3.country_id
from dataset1 t1 join dim_State t2 on t1.State = t2.state_label
join dim_Country  t3
on t2.country_id = t3.country_id) t

create table fact_orders (
         row_id int not null identity(1,1) primary key,
         order_id varchar(50),
         order_date_id int,
         ship_date_id int,
         customer_id int,
         customer_name varchar(70),
         city_id int,
         city_label varchar(70),
         product_id int,
         product_name varchar(150),
         region_id int,
         region_label varchar(50),
         priority_id int,
         priority_label varchar(40),
         ship_mode_id int,
         ship_mode_label varchar(50),
         sales_amount float,
         quantity int,
         discount float,
         profit float,
         shipping_cost float);



insert into fact_orders(sales_amount, quantity, discount,
profit, shipping_cost)

select sales_amount, quantity,
discount, profit_amount, shipping_cost from staging_metrics

select * from fact_orders


UPDATE
    fact_orders
SET
    fact_orders.order_id = Table_B.order_id
FROM
    fact_orders AS Table_A
    INNER JOIN dim_Order AS Table_B
        ON Table_A.row_id = Table_B.row_id

select * from fact_orders

UPDATE
    fact_orders
```

```
SET
    fact_orders.order_date_id = Table_B.DateKey
FROM
    fact_orders AS Table_A
    INNER JOIN Dim_Date AS Table_B
        ON Table_A.row_id = Table_B.DateKey
UPDATE
    fact_orders
SET
    fact_orders.ship_date_id = Table_B.DateKey
FROM
    fact_orders AS Table_A
    INNER JOIN Dim_ShipDate AS Table_B
        ON Table_A.row_id = Table_B.DateKey

select * from fact_orders

UPDATE
    fact_orders
SET
    fact_orders.customer_name = Table_B.Customer_Name
FROM
    fact_orders AS Table_A
    INNER JOIN staging_table AS Table_B
        ON Table_A.order_id = Table_B.Order_ID


UPDATE
    fact_orders
SET
    fact_orders.customer_id = Table_B.Customer_ID
FROM
    fact_orders AS Table_A
    INNER JOIN dim_Customers AS Table_B
        ON Table_A.customer_name = Table_B.customer_name


UPDATE
    fact_orders
SET
    fact_orders.city_label = Table_B.City
FROM
    fact_orders AS Table_A
    INNER JOIN dataset1 AS Table_B
        ON Table_A.order_id = Table_B.Order_ID


UPDATE
    fact_orders
SET
    fact_orders.city_id = Table_B.city_id
FROM
    fact_orders AS Table_A
    INNER JOIN dimCity AS Table_B
        ON Table_A.city_label = Table_B.city_label


UPDATE
    fact_orders
SET
    fact_orders.product_name = Table_B.Product_Name
FROM
    fact_orders AS Table_A
    INNER JOIN staging_table AS Table_B
        ON Table_A.order_id = Table_B.Order_ID
```

```
UPDATE
    fact_orders
SET
    fact_orders.product_id = Table_B.product_id
FROM
    fact_orders AS Table_A
    INNER JOIN dim_Product AS Table_B
        ON Table_A.product_name = Table_B.product_name

UPDATE
    fact_orders
SET
    fact_orders.region_label = tab.concatRegion
FROM (
         select Table_A.order_id, concat(Region,'-',Market) as concatRegion
         from fact_orders AS Table_A
    INNER JOIN staging_table Table_B ON Table_A.order_id = Table_B.Order_ID)
tab
         inner join fact_orders on tab.order_id = fact_orders.order_id



UPDATE
    fact_orders
SET
    fact_orders.region_id = Table_B.region_id
FROM
    fact_orders AS Table_A
    INNER JOIN dim_Region AS Table_B
        ON Table_A.region_label = Table_B.region_label


UPDATE
    fact_orders
SET
    fact_orders.priority_label = Table_B.Order_Priority
FROM
    fact_orders AS Table_A
    INNER JOIN staging_table AS Table_B
        ON Table_A.order_id = Table_B.Order_ID

UPDATE
    fact_orders
SET
    fact_orders.priority_id = Table_B.priority_id
FROM
    fact_orders AS Table_A
    INNER JOIN dim_Priority AS Table_B
        ON Table_A.priority_label = Table_B.priority_label

UPDATE
    fact_orders
SET
    fact_orders.ship_mode_label = Table_B.Ship_Mode
FROM
    fact_orders AS Table_A
    INNER JOIN staging_table AS Table_B
        ON Table_A.order_id = Table_B.Order_ID




s
UPDATE
```

```
    fact_orders
SET
    fact_orders.ship_mode_id = Table_B.ship_mode_id
FROM
    fact_orders AS Table_A
    INNER JOIN dim_ShipMode AS Table_B
        ON Table_A.ship_mode_label = Table_B.ship_mode_label
```

[1] COGS is the cost of goods sold - cost of sales - it is the direct