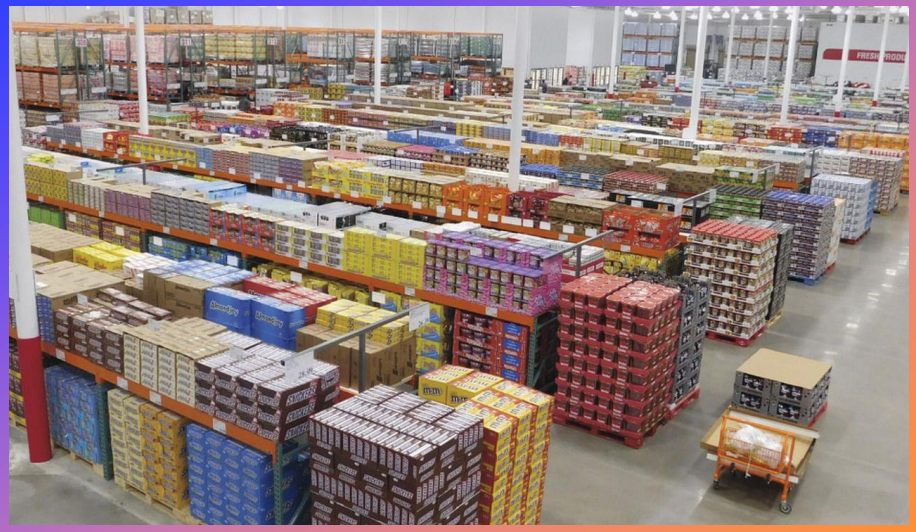
ETL & Analytics of Superstore dataset in Dimensional Data Warehouse using

Sushant Khot

MET CS 779 – Term Project

02/22/2022



Agenda

- Self-Introduction
- Project Introduction
- Dataset Attributes
- Goals and Business Questions
- Normalization ERD
- Staging
- ETL using Python
- Dimensional Datawarehouse Modelling ERD
- Demo
- Analytics & Conclusion

Self-Introduction

- Currently a project manager / enterprise content manager for a technology services
- This is my last course, and I will graduate in a few weeks
- I selected this course so I could learn and understand more about managing data
- I selected this project because I am Interested to learn how to migrate data between data stores

Project Introduction

- **Superstore Dataset:** It is a time series data of a Superstore transaction of Orders, Products, Customers etc. It is a retail dataset of a United States superstore for 4 Years.
- Any Business performing transactions on daily basis would like to analyze their data to understand their Customer behaviour, Popular products, Sales and many such entities to make better informed decisions to develop and grow their business.

- **Normalization of database:** I have demonstrated the Normalization technique on this dataset based on the skills acquired from this course. Although I have not uploaded any data in the normalized tables of the database, I have created the DDL to created the database structure and have designed a Normalized database ERD.

- **ETL into SQL SERVER using Python:** One of the key goals of the project is to perform ETL on this data using Python and loading the data into SQL SERVER. I have extracted a .csv format data of the superstore. This data needed cleansing, format changes and correction of inconsistent data which I have performed completely using Python.

- **Dimensional Datawarehouse:** I also wanted to explore the topic of Dimensional Datawarehouse and implement a solution to answer some basic Business questions for the Superstore. The FACT tables and Dimensional Tables will help us answer some Key Business Questions.
- I have used Python for ETL, SQL SERVER as my Database to query for analytics and Tableau for visualizations

Dataset Attributes

- **Order Data:** Order ID, Order Date, Ship Date, Ship Mode
- **Customer Data:** Customer ID, Customer Name, Segment
- **Location:** Country, City, State, Postal Code, Region
- **Product Data:** Product ID, Category, Sub-Category, Product Name
- **$ Sales:** Sales

Sample Data:

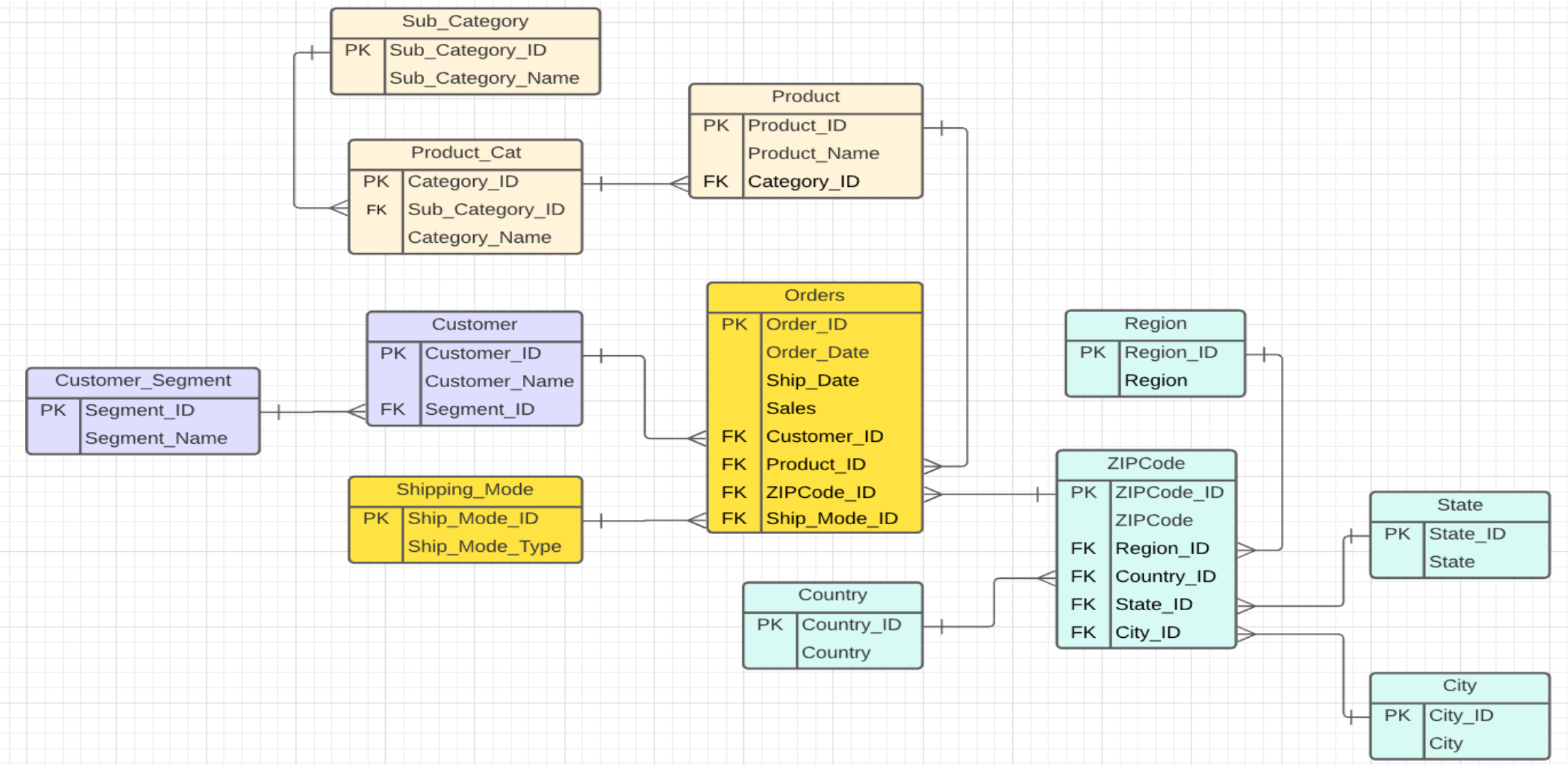
Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	State	Postal Code	Region	Product ID	Category	Sub-Category
1	CA-2017-152156	08-11-2017	11-11-2017	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	42420	South	FUR-BO-10001798	Furniture	Bookcases
2	CA-2017-152156	08-11-2017	11-11-2017	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	42420	South	FUR-CH-10000454	Furniture	Chairs
3	CA-2017-138688	12-06-2017	16/06/2017	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	California	90036	West	OFF-LA-10000240	Office Supplies	Labels

Product Name	Sales
Bush Somerset Collection Bookcase	261.96
Hon Deluxe Fabric Upholstered Stacking Chairs, Rounded Back	731.94
Self-Adhesive Address Labels for Typewriters by Universal	14.62

Goals / Business Questions

1. What are the TOP 3 most Popular Products in terms of Sales and at which Location?
2. Which Product Sub-category is the most popular?
3. Which Location has the maximum Average Sales and in which Month-Year?
4. Which are the TOP 5 Cities with Highest number of Orders placed?
5. Which Customer has the most Orders? What is their Segment?
6. Which Shipping Mode is most requested by the Customers?
7. What is the Maximum delay between Order and Ship Date? How many cities have this maximum delay?
8. Which Year had the most Sales?

Normalization ERD - Corrected



Staging in SQL Server

```
7   -- Create the Superstore Database
8   --CREATE DATABASE Superstore
9   --go
10
11  use Superstore;
12
13  -- drop table Staging_Table
14
15  -- STAGING Table for storing all our data in one Table to further distribute to different FACT and Dimension Tables
16  CREATE TABLE Staging_Table(
17  Row_ID int IDENTITY(1,1),
18  Order_ID varchar(20),
19  Order_dt date,
20  Ship_dt date,
21  Ship_Mode varchar(20),
22  Customer_ID varchar(20),
23  Customer_Name varchar(100),
24  Segment varchar(20),
25  Country varchar(32),
26  City varchar(32),
27  State varchar(32),
28  Zip_Code varchar(10),
29  Region varchar(10),
30  Product_ID varchar(32),
31  Category varchar(32),
32  Subcategory varchar(32),
33  Product_Name varchar(255),
34  Sales numeric(8,2),
35  CONSTRAINT Superstore_RowId_PK PRIMARY KEY (Row_ID));
36
```

ETL USING PYTHON

- EXTRACT

```python
# -*- coding: utf-8 -*-
"""
Sushant Khot
Class: MET CS 779 - Advanced Database Management
Date: 02/17/2022
MET CS 779 Term Project:
Topic: ETL and Analytics of a SuperStore Dataset into Dimensional Data Warehouse using Python
"""

# Import Libraries
import os
import pyodbc
import pandas as pd
from datetime import datetime
import math


# Code to load the dataset using Relative Path
here = os.path.abspath(__file__)  # Relative Path code
input_dir = os.path.abspath(os.path.join(here, os.pardir))
superstore_dataset = os.path.join(input_dir, 'SuperStore_dataset.csv')
# superstore_dataset = "C:\\Users\\sushk\\Downloads\\BU\\MET CS 779\\Term Project\\KhotSushant_


try:
    ss_df = pd.read_csv(superstore_dataset)

except Exception as e:
    print(e)
    print('failed to read Super Store data into Data Frame')


# COnnection to SQL Server
conn = pyodbc.connect('Driver={SQL Server};'
                      'Server=ARNAVDESKTOP;'
                    #  'Server=SUSHANTSURFACE3;'
                      'Database=Superstore;'
                      'Trusted_Connection=yes;')


# Create a cursor for SQL code execution
cursor = conn.cursor()

```

Transform and Load

```
# We can exclude the 1st "Row ID" column as it is set as an Identity column in the Staging Table
ss_df = ss_df.iloc[:, 1:]

# Check if any columns in the dataframe have blank values
print(ss_df.isnull().any())

# We can see that the Postal Code / ZIPCode column has blank values.
# We will handle this while inserting data in the Staging Table.

# We see that the date values have "-" and "/" as separators randomly. The date format in the csv is dd-mm-yyyy OR dd/mm/y
# We will make the format consitent by replacing "/" with "-"
ss_df['Order Date'] = ss_df['Order Date'].str.replace('/','-')
ss_df['Ship Date'] = ss_df['Ship Date'].str.replace('/','-')

# Truncate the Staging Table in case we re-run this script multiple times on the same csv to avoid duplication of records.
cursor.execute('TRUNCATE TABLE dbo.Staging_Table')


# Insert DataFrame records one by one into the Staging table.
for i,row in ss_df.iterrows():

    # Store the row values in a Python List
    val_list = list(row)

    # Convert the string values to Date
    val_list[1] = datetime.strptime(val_list[1], '%d-%m-%Y')
    val_list[2] = datetime.strptime(val_list[2], '%d-%m-%Y')

    # Check if ZIPCode value (val_list[10]) is blank then add a default ZIPCode = 99999
    if math.isnan(val_list[10]):
        val_list[10] = 99999


    # Create the INSERT statement and execute it via the cursor connection
    # sql = "INSERT INTO dbo.Staging_ss ('" +cols + "') VALUES(?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)"
    sql = "INSERT INTO dbo.Staging_Table VALUES(?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)"
    cursor.execute(sql, val_list)
```

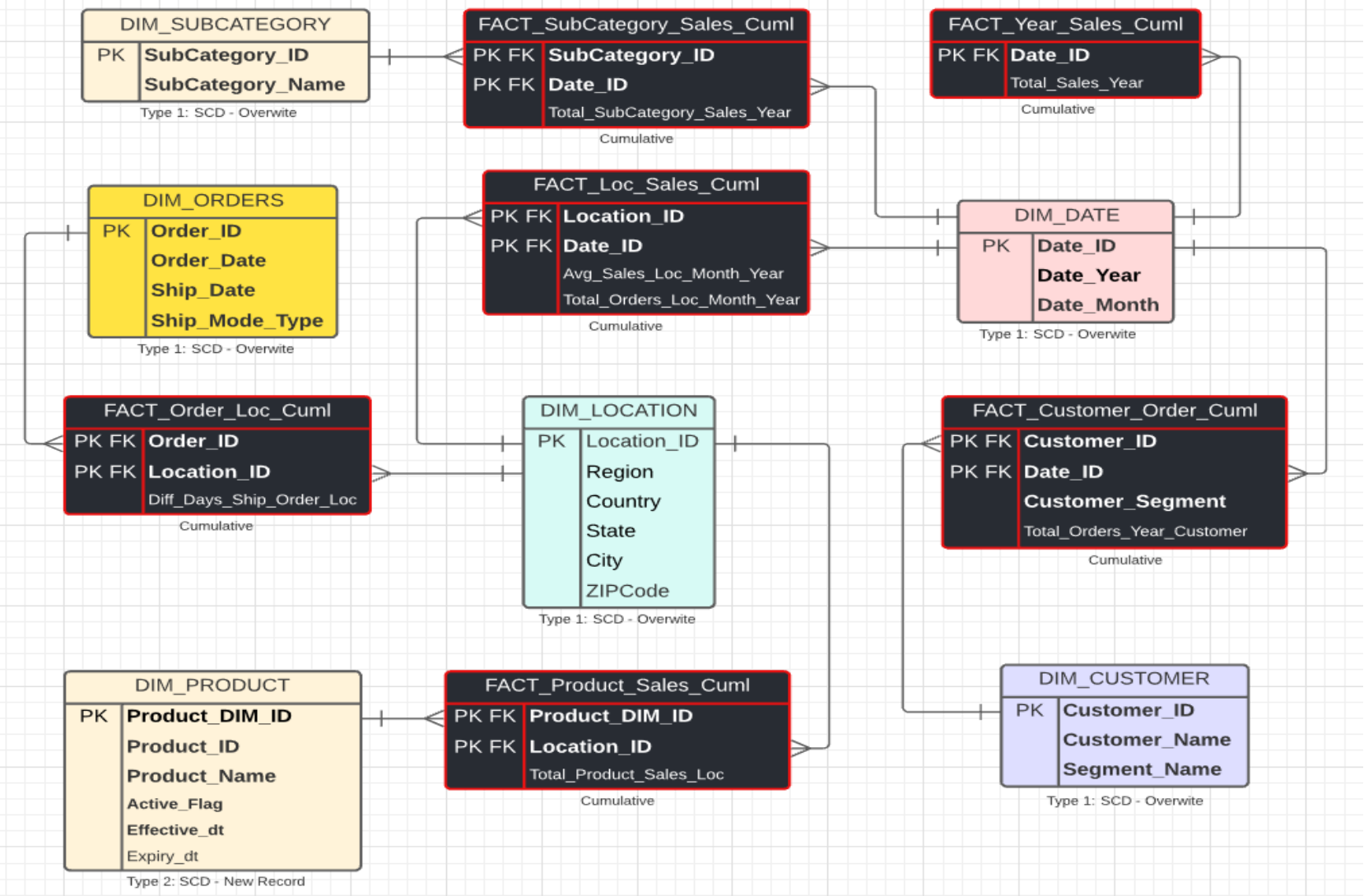
```
# The below code will take care of inconsitent Product Names that I found in the dataset.
# This will replace all inconsitent Product names with one of the values from the list of assigned Product Names.
# For e.g. I saw that there were 32 Product IDs which had more than 1 Product Name assigned.
cursor.execute('SELECT Product_ID FROM Staging_Table GROUP BY Product_ID HAVING COUNT(DISTINCT(Product_Name)) > 1')
prod_ID = cursor.fetchall()

for prdID in prod_ID:
    cursor.execute("SELECT TOP 1 Product_Name FROM Staging_Table WHERE Product_ID = '" + str(prdID[0]) + "'")
    prod_Name = cursor.fetchone()
    update_sql = "UPDATE Staging_Table SET Product_Name = ? WHERE Product_ID = ?"
    cursor.execute(update_sql, [str(prod_Name[0]), str(prdID[0])])


# Commit and close the connection
conn.commit()
cursor.close()
conn.close()
```

Additional Cleansing and Load

Dimensional Datawarehouse Modelling ERD



MERGE data into Dimension and FACT tables

```sql
/*
==================================
MERGE Data into the Dimension Tables
==================================
*/

-- PRODUCT Dimension FROM Staging
MERGE INTO DIM_PRODUCT AS tgt
USING (SELECT DISTINCT(Product_ID), Product_Name FROM  Staging_Table) AS src
ON src.Product_ID = tgt.Product_ID
WHEN NOT MATCHED BY TARGET THEN
    INSERT (Product_ID, Product_Name)
    VALUES (src.Product_ID, src.Product_Name)
WHEN MATCHED THEN UPDATE SET
    tgt.Product_ID = src.Product_ID,
    tgt.Product_Name = src.Product_Name;

--DELETE FROM DIM_PRODUCT
-- SELECT * FROM DIM_PRODUCT


-- CUSTOMER Dimension FROM Staging
MERGE INTO DIM_CUSTOMER AS tgt
USING (SELECT DISTINCT(Customer_ID), Customer_Name, Segment FROM  Staging_Table) AS src
ON src.Customer_ID = tgt.Customer_ID
WHEN NOT MATCHED BY TARGET THEN
    INSERT (Customer_ID, Customer_Name, Segment_Name)
    VALUES (src.Customer_ID, src.Customer_Name, src.Segment)
WHEN MATCHED THEN UPDATE SET
    tgt.Customer_ID = src.Customer_ID,
    tgt.Customer_Name = src.Customer_Name,
    tgt.Segment_Name = src.Segment;
```

```sql
/*
==============================
MERGE Data into the FACT Tables
==============================
*/

-- Product_Sales FACT Table
MERGE INTO FACT_Product_Sales_Cuml AS tgt
USING (SELECT DP.PRODUCT_DIM_ID, DL.Location_ID, SUM(ST.Sales) AS Total_Product_Sales_Loc
       FROM Staging_Table ST
       JOIN DIM_PRODUCT DP
       ON ST.Product_ID = DP.Product_ID
       JOIN DIM_LOCATION DL
       ON ST.Zip_Code = DL.ZIPCode
       GROUP BY PRODUCT_DIM_ID, DL.Location_ID) AS src
ON (src.PRODUCT_DIM_ID = tgt.PRODUCT_DIM_ID AND
    src.Location_ID = tgt.Location_ID)
WHEN NOT MATCHED BY TARGET THEN
    INSERT (PRODUCT_DIM_ID, Location_ID, Total_Product_Sales_Loc)
    VALUES (src.PRODUCT_DIM_ID, src.Location_ID, src.Total_Product_Sales_Loc)
WHEN MATCHED THEN UPDATE SET
    tgt.PRODUCT_DIM_ID = src.PRODUCT_DIM_ID,
    tgt.Location_ID = src.Location_ID,
    tgt.Total_Product_Sales_Loc = src.Total_Product_Sales_Loc;

--DELETE FROM FACT_Product_Sales_Cuml
-- Select * from FACT_Product_Sales_Cuml
```

DEMO

ANALYTICS


Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Series 1
Series 2
Series 3

Answering our Business Questions

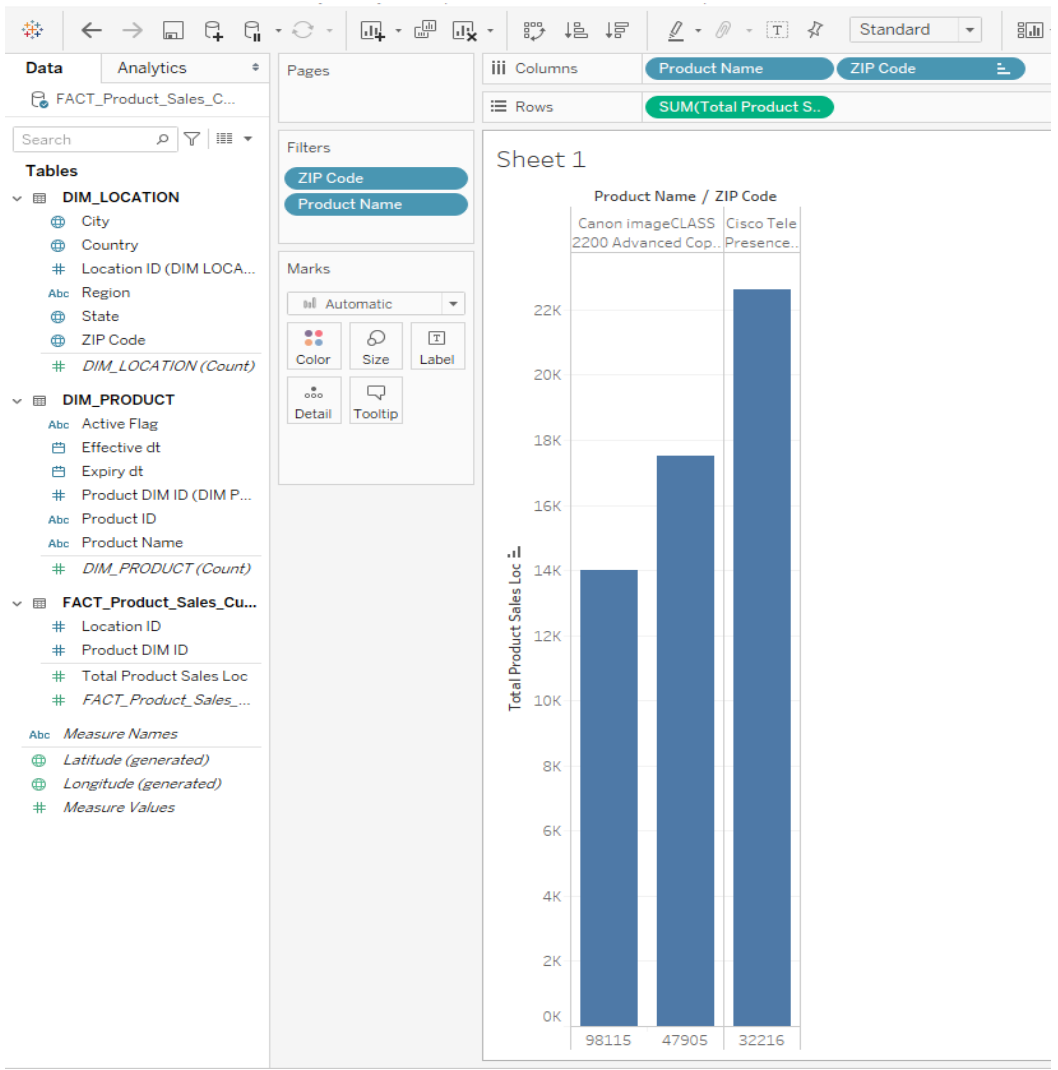
1. What are the TOP 3 most Popular Products in terms of Sales and at which Location?

```sql
-- 1. What are the TOP 3 most Popular Products in terms of Sales and at which Location?
SELECT TOP 3 DP.Product_ID,
             DP.Product_Name,
             DL.City,
             DL.State,
             DL.ZIPCode,
             FP.Total_Product_Sales_Loc AS Total_Sales_$
FROM FACT_Product_Sales_Cuml FP
JOIN DIM_PRODUCT DP
ON FP.Product_DIM_ID = DP.Product_DIM_ID
JOIN DIM_LOCATION DL
ON FP.Location_ID = DL.Location_ID
ORDER BY FP.Total_Product_Sales_Loc desc
```

121 %

Results | Messages

	Product_ID	Product_Name	City	State	ZIPCode	Total_Sales_$
1	TEC-MA-10002412	Cisco TelePresence System EX90 Videoconferencing ...	Jacksonville	Florida	32216	22638.48
2	TEC-CO-10004722	Canon imageCLASS 2200 Advanced Copier	Lafayette	Indiana	47905	17499.95
3	TEC-CO-10004722	Canon imageCLASS 2200 Advanced Copier	Seattle	Washington	98115	13999.96



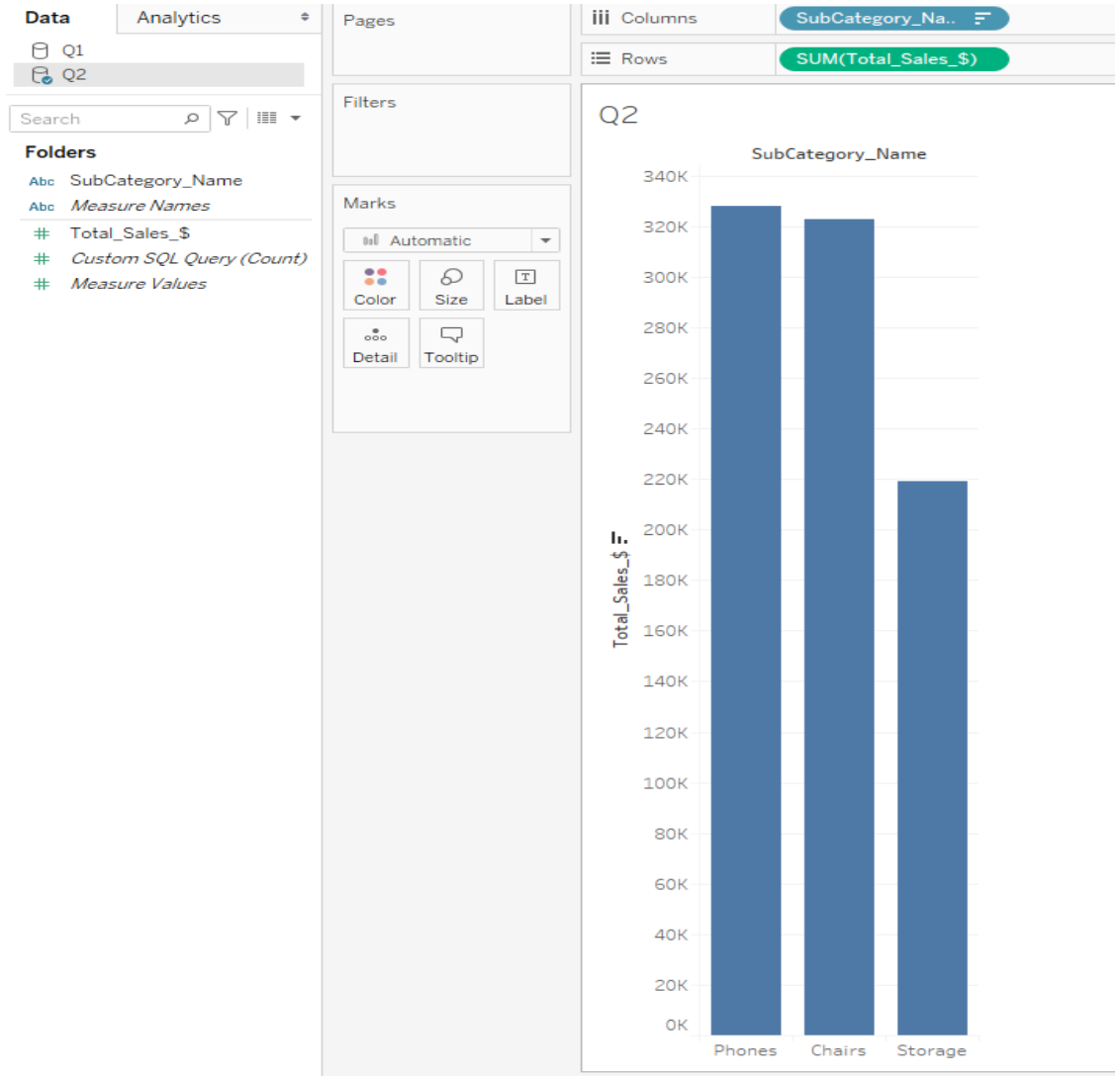
2. Which Product Sub-category is the most popular?

```sql
-- 2. Which Product Sub-category is the most popular?
SELECT TOP 3 DS.SubCategory_Name,
       SUM(FS.Total_SubCategory_Sales_Year) AS Total_Sales_$
FROM FACT_Subcategory_Sales_Cuml FS
JOIN DIM_SUBCATEGORY DS
ON FS.SubCategory_ID = DS.SubCategory_ID
GROUP BY DS.SubCategory_Name
ORDER BY SUM(FS.Total_SubCategory_Sales_Year) desc
```

121 %

Results | Messages

	SubCategory_Name	Total_Sales_$
1	Phones	327782.49
2	Chairs	322822.71
3	Storage	219343.37



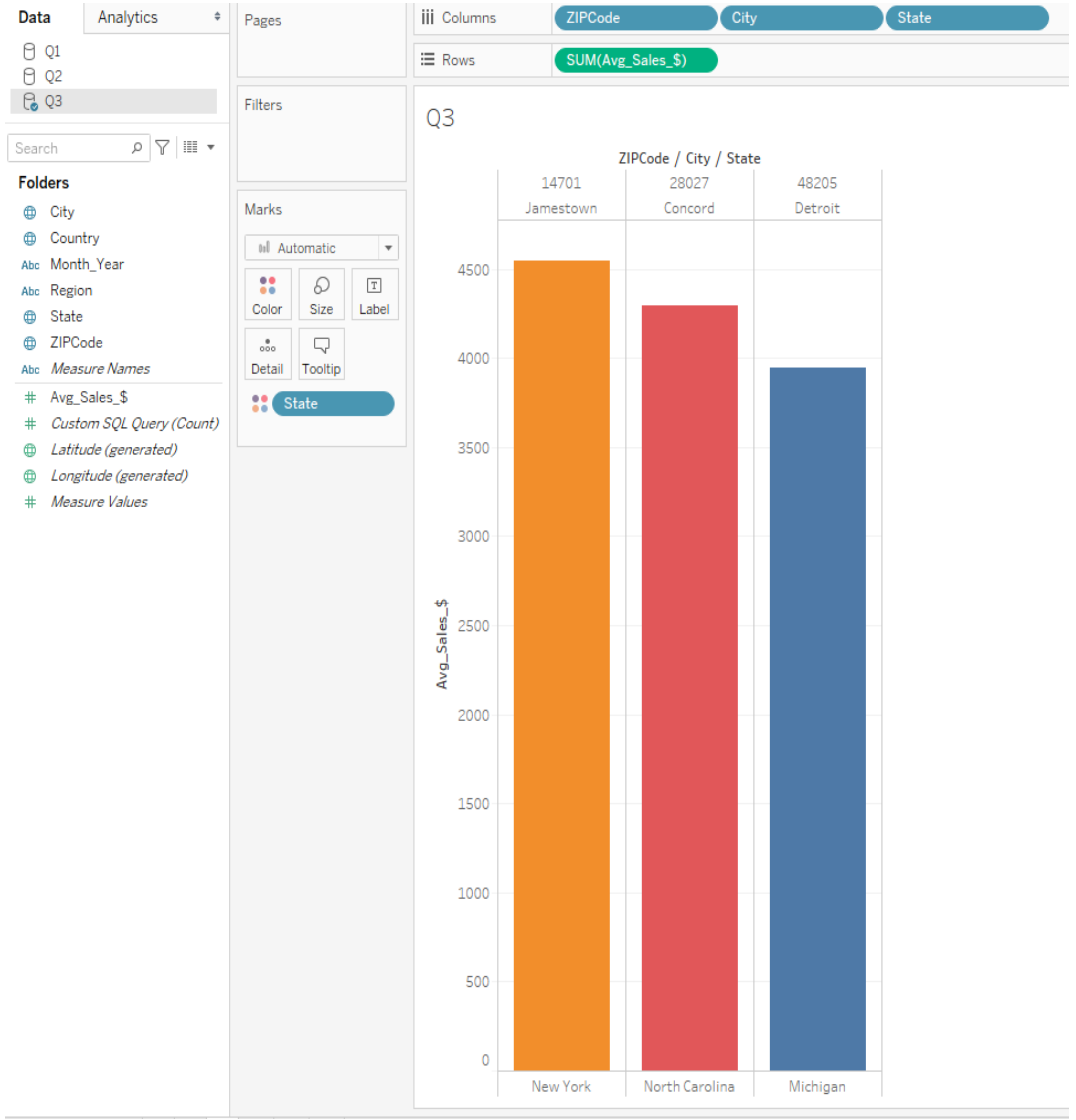
3. Which Location has the maximum Average Sales and in which Month-Year?

```sql
-- 3. Which Location has the maximum Average Sales and in which Month-Year?
SELECT TOP 3 DL.City,
       DL.State,
       DL.Region,
       DL.Country,
       DL.ZIPCode,
       FL.Date_ID AS Month_Year,
       FL.Avg_Sales_Loc_Month_Year AS Avg_Sales_$
FROM FACT_Loc_Sales_Cuml FL
JOIN DIM_LOCATION DL
ON FL.Location_ID = DL.Location_ID
ORDER BY FL.Avg_Sales_Loc_Month_Year desc
```

121 %

Results | Messages

	City	State	Region	Country	ZIPCode	Month_Year	Avg_Sales_$
1	Jamestown	New York	East	United States	14701	11-2015	4548.81
2	Concord	North Carolina	South	United States	28027	1-2016	4297.64
3	Detroit	Michigan	Central	United States	48205	12-2017	3947.35



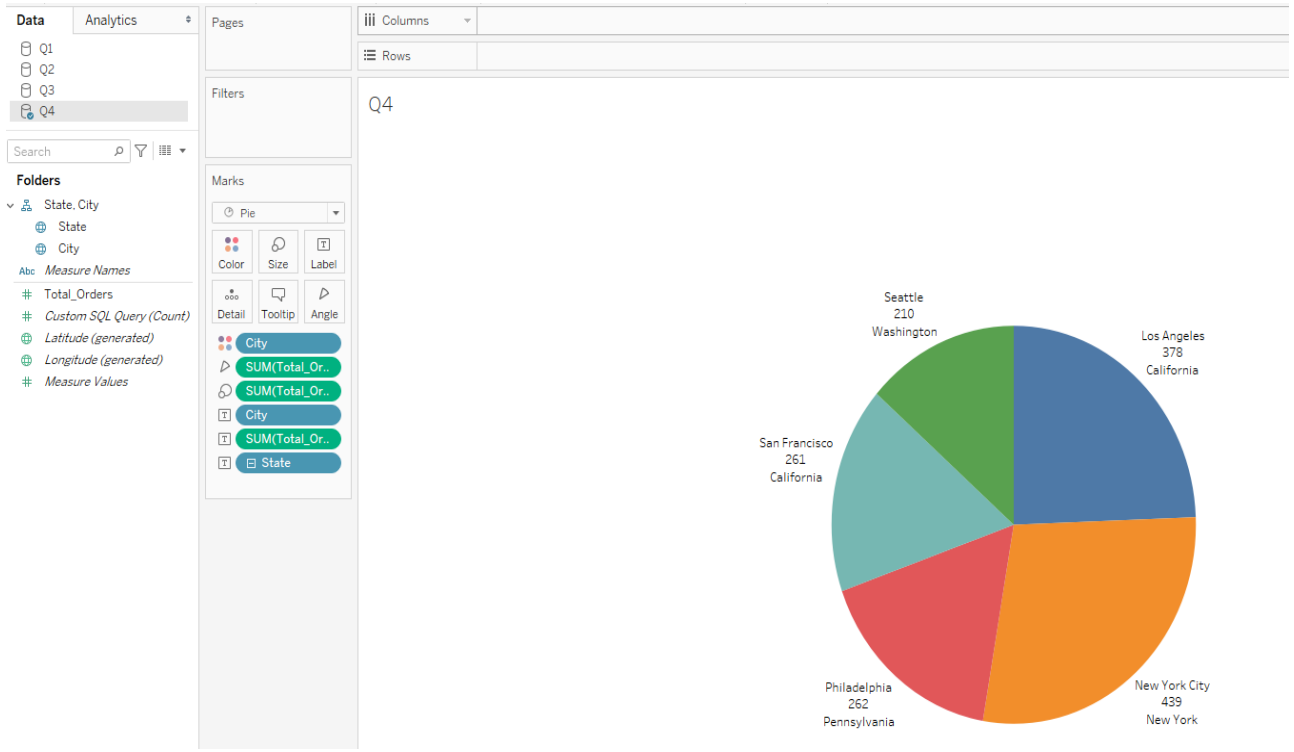
4. Which are the TOP 5 Cities with Highest number of Orders placed?

```sql
-- 4. Which are theh TOP 5 Cities with Highest number of Orders placed
SELECT TOP 5 DL.City,
       DL.State,
       SUM(FL.Total_Orders_Loc_Month_Year) AS Total_Orders
FROM FACT_Loc_Sales_Cuml FL
JOIN DIM_LOCATION DL
ON FL.Location_ID = DL.Location_ID
GROUP BY DL.City,
         DL.State
ORDER BY SUM(FL.Total_Orders_Loc_Month_Year) desc
```

121 %

Results | Messages

	City	State	Total_Orders
1	New York City	New York	439
2	Los Angeles	California	378
3	Philadelphia	Pennsylvania	262
4	San Francisco	California	261
5	Seattle	Washington	210



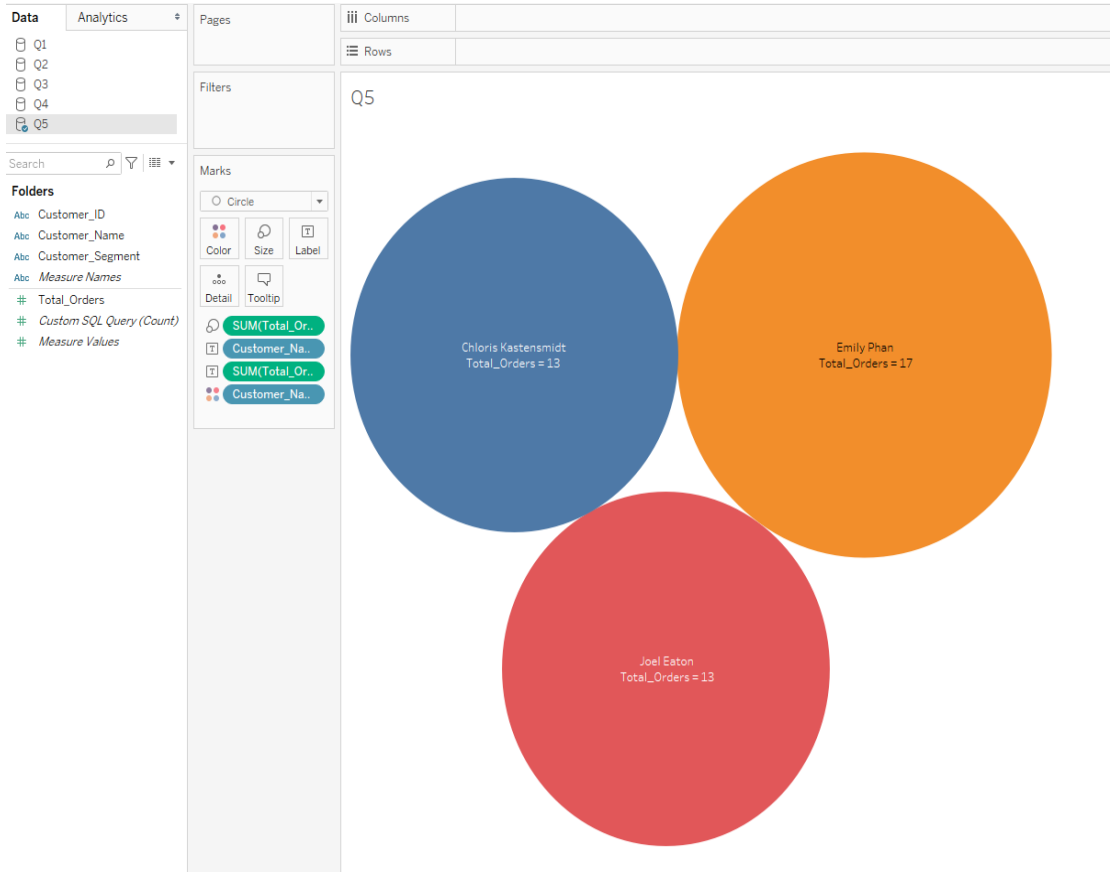
5. Which Customer has the most Orders? What is their Segment?

```sql
-- 5. Which Customer has the most Orders? What is their Segment?
SELECT TOP 3 FC.Customer_ID,
       DC.Customer_Name,
       FC.Customer_Segment,
       SUM(FC.Total_Orders_Year_Customer) AS Total_Orders
FROM FACT_Customer_Order_Cuml FC
JOIN DIM_CUSTOMER DC
ON FC.Customer_ID = DC.Customer_ID
GROUP BY FC.Customer_ID,
       DC.Customer_Name,
       FC.Customer_Segment
ORDER BY SUM(FC.Total_Orders_Year_Customer) desc
```

121 %

Results | Messages

	Customer_ID	Customer_Name	Customer_Segment	Total_Orders
1	EP-13915	Emily Phan	Consumer	17
2	CK-12205	Chloris Kastensmidt	Consumer	13
3	JE-15745	Joel Eaton	Consumer	13



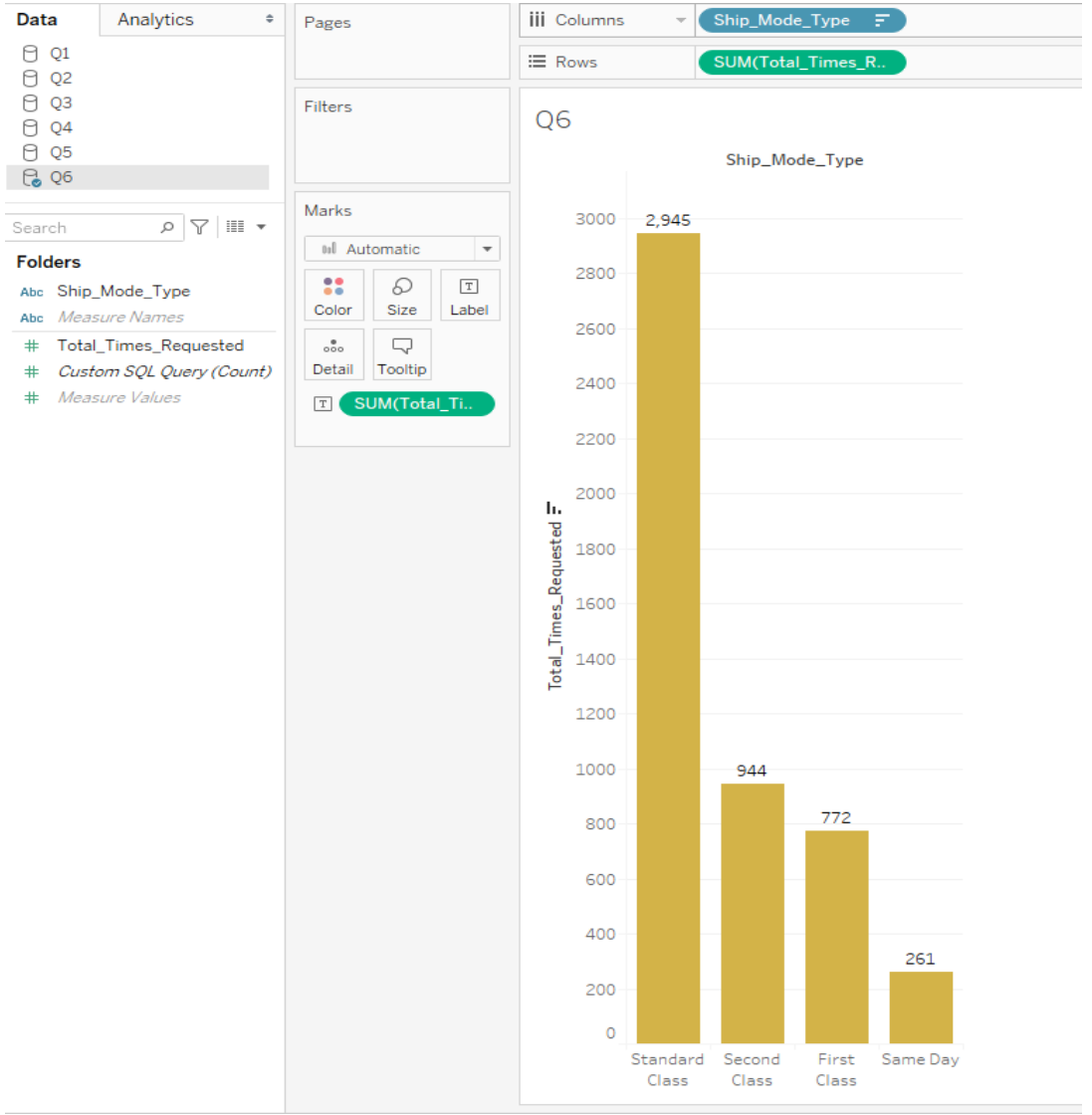
6. Which Shipping Mode is most requested by the Customers?

```sql
-- 6. Which Shipping Mode is most requested by the Customers?
SELECT Ship_Mode_Type,
       COUNT(DISTINCT(Order_ID)) AS Total_Times_Requested
FROM DIM_ORDERS
GROUP BY Ship_Mode_Type
ORDER BY COUNT(DISTINCT(Order_ID)) desc
```

121 %

Results | Messages

	Ship_Mode_Type	Total_Times_Requested
1	Standard Class	2945
2	Second Class	944
3	First Class	772
4	Same Day	261



7. What is the Maximum delay between Order and Ship Date? How many cities have this maximum delay?

8. Which Year had the most Sales?

```sql
-- 7. What is the Maximum delay between Order and Ship Date? How many cities have this maxmimum delay?
SELECT COUNT(DISTINCT(DL.City)) AS Num_of_Cities,
       MAX(FO.Diff_Days_Ship_Order_Loc) AS Maximum_Delay_in_Days
FROM FACT_Order_Loc_Cuml FO
JOIN DIM_LOCATION DL
ON FO.Location_ID = DL.Location_ID
ORDER BY MAX(FO.Diff_Days_Ship_Order_Loc) desc
```

121 %

Results | Messages

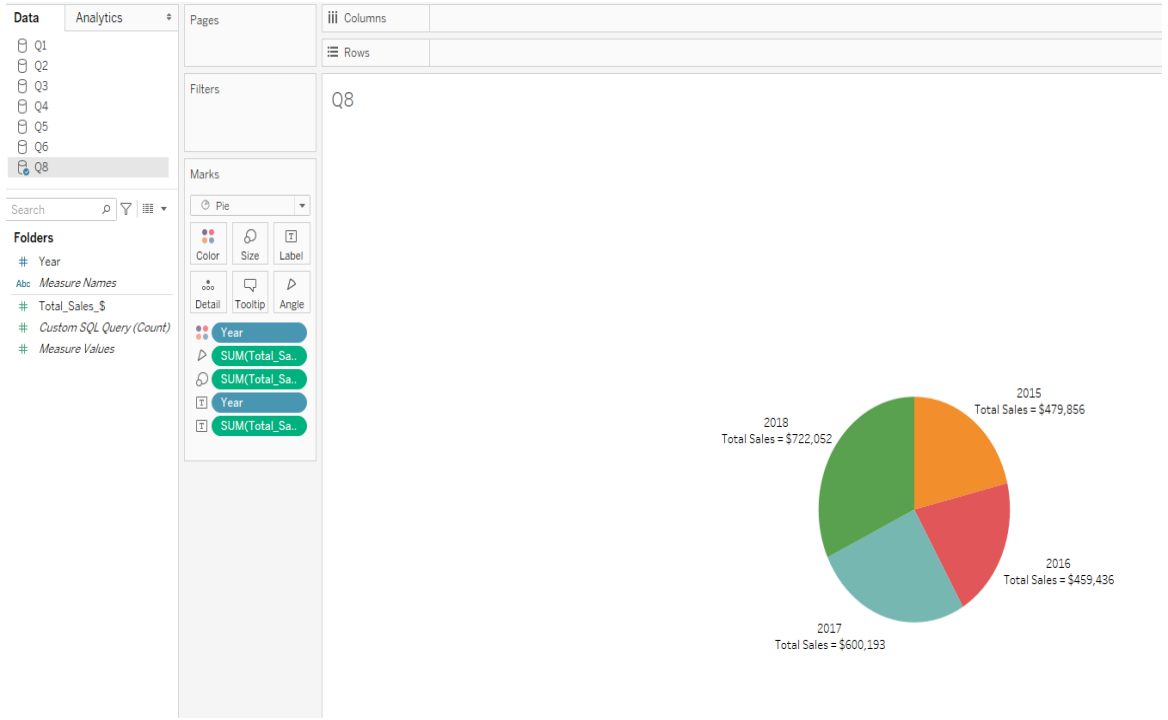
	Num_of_Cities	Maximum_Delay_in_Days
1	528	7

```sql
-- 8. Which Year had the most Sales?
SELECT Date_ID AS Year,
       Total_Sales_Year AS Total_Sales_$
FROM FACT_Year_Sales_Cuml
ORDER BY Total_Sales_Year desc
```

121 %

Results | Messages

	Year	Total_Sales_$
1	2018	722051.84
2	2017	600192.64
3	2015	479856.19
4	2016	459435.89



CONCLUSION

– NEW SLIDE

Business Decisions based on our Analysis – New Slide

Thus, we conclude our ETL and Datawarehouse project by finding the answers for our Business questions. These will help the Business in the following manner:

- We will be able to easily find which are our Top Products that the Customers buy at different Locations. this will help us maintain appropriate pricing and stock of these products based on this Customer behavior.
- We will know the Subcategory of Products that are most popular amongst our customers, and we can concentrate on introducing more Products of these sub-categories to grow our business.
- We will also find out which specific Store Location is doing the most business and then we can make informed decisions on Inventory for this location, Scalability of the store and probably some discounts and perks for our customers at these specific locations.
- The same is applicable to the top cities who have the most Orders.
- Our analysis on Customers with the highest number of Orders can be used to introduce specific offers for these repeating customers and we can further analyze their transaction behavior, bundle offers based on their buying needs and their Segments etc.
- We also found out the most request Shipping Mode by our Customers. We can introduce offers like free shipping for our lowest mode of Shipping and try to introduce a subscription model for customers who opt for the most requested Shipping Mode.
- We can also look at the reason behind the delays between Order Dates and Ship dates for the specific cities. There could be issues with supply of materials, transportation or the overall supply chain might have to be optimized.
- Finally, we can analyze our data on Sales data for each Business year to find out what we did good on our most profitable year and what went wrong on our lowest profitable Year.

THANK YOU

Sushant Khot

sushantk@bu.edu

BU ID: U73866118


Questions?