رواد مصر الرقمية

Data Analyst Specialist

CAI1_DAT1_G12d

DEPI Final Project

Store Sales Dataset Analysis

Group “D”

Names:

Rawan Haitham

Engy Talaat

Yara Osama

Toqa Kashef

Raneem Zaki

Supervised by:

Eng. Ahmed Samir

Week 1 (Task 1 Report):

Week 1: Build Data Model, Data Cleaning and Preprocessing

- **Tasks:**
  - ➔ Data Preprocessing: Build a data model and clean and preprocess the data.
  - ➔ Tools: SQL, Python (pandas, Matplotlib).
- **Deliverables:**
  - ➔ Cleaned dataset ready for analysis.
  - ➔ Data preprocessing notebook.

For the first week we needed to clean our data from the dataset given to us "Superstore Sales Data". For that we used Python Jupyter and utilized Pandas and Matplotlib to find out what we needed to do with our data. As our data didn't have much to do with it we started a simple data cleaning process and decided to focus more on our analysis questions and the story that our data can tell us later on in the visualization process.

The first step done was importing pandas, numby, matplotlib and seaborn and code was as follows:

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns
```

We then inserted our CSV file that had the data:

```
df=pd.read_csv('Superstore Sales Dataset.csv')
```

We then used the following code to find out how many distinct Order IDs we had which we found were **4922:**

```
df['Order ID'].nunique()
```

We then checked to see if there's any Null data using the following code:

```
df.isnull().sum()
```

The results showed only the postal code

```
Row ID            0
Order ID          0
Order Date        0
Ship Date         0
Ship Mode         0
Customer ID       0
Customer Name     0
Segment           0
Country           0
City              0
State             0
Postal Code      11
Region            0
Product ID        0
Category          0
Sub-Category      0
Product Name      0
Sales             0
dtype: int64
```

It was decided after looking at the data that certain columns were not needed for our analysis, the columns were (Row ID, Customer Name and Postal Code) which is why they were removed using the following code:

```
df_cleaned = df.drop(columns=['Row ID', 'Customer Name' , 'Postal Code'])
```

To get a summary on how are data currently looks like we used the following code:

```
df_cleaned.info()
```

The results were as follows, we found out the types of our columns were almost all of the Object type, and we had one Float type:

```
<class 'pandas.core.frame.DataFrame'>

RangeIndex: 9800 entries, 0 to 9799

Data columns (total 15 columns):

 #   Column        Non-Null Count  Dtype

---  ------        --------------  -----

 0   Order ID      9800 non-null   object

 1   Order Date    9800 non-null   object

 2   Ship Date     9800 non-null   object

 3   Ship Mode     9800 non-null   object

 4   Customer ID   9800 non-null   object

 5   Segment       9800 non-null   object

 6   Country       9800 non-null   object

 7   City          9800 non-null   object

 8   State         9800 non-null   object

 9   Region        9800 non-null   object

 10  Product ID    9800 non-null   object

 11  Category      9800 non-null   object

 12  Sub-Category  9800 non-null   object

 13  Product Name  9800 non-null   object

 14  Sales         9800 non-null   float64

dtypes: float64(1), object(14)
```

This is why we changed some column types such as the Ship Date and Order Date into the datetime type as well as, all the column types of object are changed to be String:

```
df_cleaned['Ship      Date']=pd.to_datetime(df_cleaned['Ship      Date'],
format='%d/%m/%Y')
```

```
df_cleaned['Order    Date']=pd.to_datetime(df_cleaned['Order    Date'],
format='%d/%m/%Y')



df['Order ID']= df['Order ID'].astype("string")

df['Ship Mode']= df['Ship Mode'].astype("string")

df['Customer ID']= df['Customer ID'].astype("string")

df['Segment']= df['Segment'].astype("string")

df['Country']= df['Country'].astype("string")

df['City']= df['City'].astype("string")

df['State']= df['State'].astype("string")

df['Region']= df['Region'].astype("string")

df['Product ID']= df['Product ID'].astype("string")

df['Category']= df['Category'].astype("string")

df['Sub-Category']= df['Sub-Category'].astype("string")

df['Product Name']= df['Product Name'].astype("string")

df['Category']= df['Category'].astype("string")
```

The results were as follows:

```
<class 'pandas.core.frame.DataFrame'>

RangeIndex: 8655 entries, 0 to 8654

Data columns (total 15 columns):

 <class 'pandas.core.frame.DataFrame'>

RangeIndex: 9800 entries, 0 to 9799

Data columns (total 15 columns):

 #   Column        Non-Null Count  Dtype

---  ------        --------------  -----

 0   Order ID      9800 non-null   string
```

```
 1   Order Date    9800 non-null   datetime64[ns]
 2   Ship Date     9800 non-null   datetime64[ns]
 3   Ship Mode     9800 non-null   string
 4   Customer ID   9800 non-null   string
 5   Segment       9800 non-null   string
 6   Country       9800 non-null   string
 7   City          9800 non-null   string
 8   State         9800 non-null   string
 9   Region        9800 non-null   string
 10  Product ID    9800 non-null   string
 11  Category      9800 non-null   string
 12  Sub-Category  9800 non-null   string
 13  Product Name  9800 non-null   string
 14  Sales         9800 non-null   float64
dtypes: datetime64[ns](2), float64(1), string(12)


memory usage: 1014.4 KB
```

We checked for any duplicates in the data but ended up finding none and the result was as follows: `np.int64(0)`

```
df.duplicated().sum()
```

Finally we wanted to check for any outliers using Matplotlib and we actually discovered there were outliers and we removed them.

Steps:

1. **Calculate Q1 (25th percentile) and Q3 (75th percentile)** for the "Sales" column.
2. **Compute the IQR (Q3 - Q1)**.
3. **Determine the lower and upper bounds** for acceptable data points as:
    - Lower bound = Q1 - 1.5 * IQR
    - Upper bound = Q3 + 1.5 * IQR
4. **Filter out the outliers** that fall outside this range.

5. **Remove the outliers** from the dataset.

After applying the IQR method, 1,145 outliers were detected and removed from the dataset. The cleaned dataset now contains 8,655 rows.

To ensure the quality of the data and prevent skewing the analysis, outliers in the "Sales" column were identified and removed. The Interquartile Range (IQR) method was used to detect outliers. This method identifies data points that fall significantly below the 25th percentile (Q1) or above the 75th percentile (Q3). The following steps were carried out:

1. The 25th percentile (Q1) and 75th percentile (Q3) of the "Sales" data were calculated.
2. The IQR was determined by subtracting Q1 from Q3.
3. Outliers were defined as any value below Q1 - 1.5 * IQR or above Q3 + 1.5 * IQR.
4. A total of 1,145 outliers were removed, reducing the dataset to 8,655 entries from the original 9,800.

This procedure ensures that extreme values do not disproportionately influence the analysis, leading to more reliable insights.

Visualization:

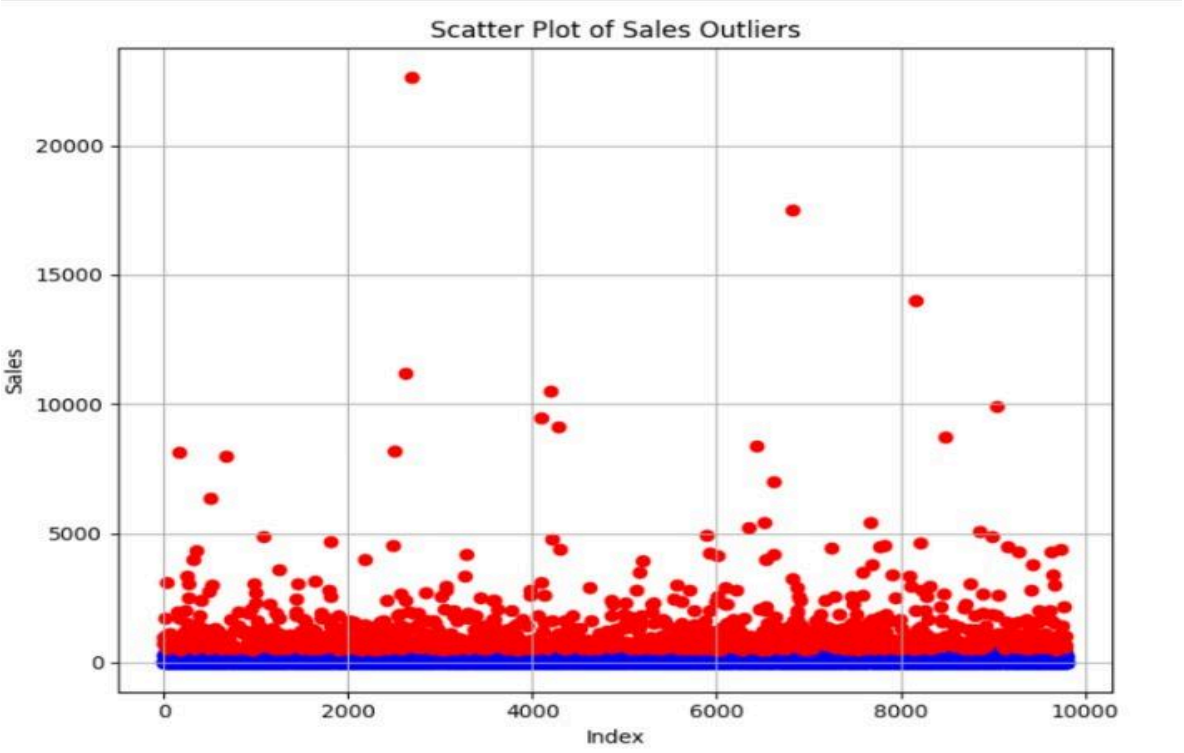
- A scatter plot was generated where:
  - **X-axis:** Represents the index of each data point (row number in the dataset).
  - **Y-axis:** Represents the sales value.
  - **Red points** indicate outliers, while **blue points** represent non-outlier data.

Key Observations:

- Outliers were identified as extreme sales values either far lower or far higher than the general sales data range.
- The majority of the data falls within the non-outlier range, with a few points (outliers) marked in red, indicating exceptionally high or low sales transactions.

```
Q1 = df['Sales'].quantile(0.25)

Q3 = df['Sales'].quantile(0.75)

IQR = Q3 - Q1



lower_bound = Q1 - 1.5 * IQR

upper_bound = Q3 + 1.5 * IQR
```

```
outliers = df[(df['Sales'] < lower_bound) | (df['Sales'] > upper_bound)]

non_outliers = df[(df['Sales'] >= lower_bound) & (df['Sales'] <= upper_bound)]



plt.figure(figsize=(14, 6))

plt.subplot(1, 2, 1)

plt.scatter(non_outliers.index, non_outliers['Sales'], color='blue', label='Non-Outliers')

plt.scatter(outliers.index, outliers['Sales'], color='red', label='Outliers')

plt.xlabel('Index')

plt.ylabel('Sales')

plt.title('Scatter Plot of Sales Outliers')

plt.grid()

plt.tight_layout()

plt.show()
```



Finally we used the following code to compare between the number of row before and after removing the outliers:

```
original_rows = df_cleaned.shape[0]

non_outliers = non_outliers.shape[0]

original_rows, non_outliers
```

The output:

(9800, 8655)

The result showed that there were about 1,145 rows that's not in the range.

Task 2:

Week 2: Analysis Questions Phase

- **Tasks:**

  ➔ Determine Data Analysis Questions: Determine all possible analysis questions that can be deducted from the given dataset and would be of interest to the organization's decision makers, e.g., what is the impact on products category and regions on sales performance?

  ➔ Tools: SQL, Python (pandas, Matplotlib).

- **Deliverables:**

  ➔ Set of analysis questions that can be answered via the dataset.

x

Key Areas for Analysis:

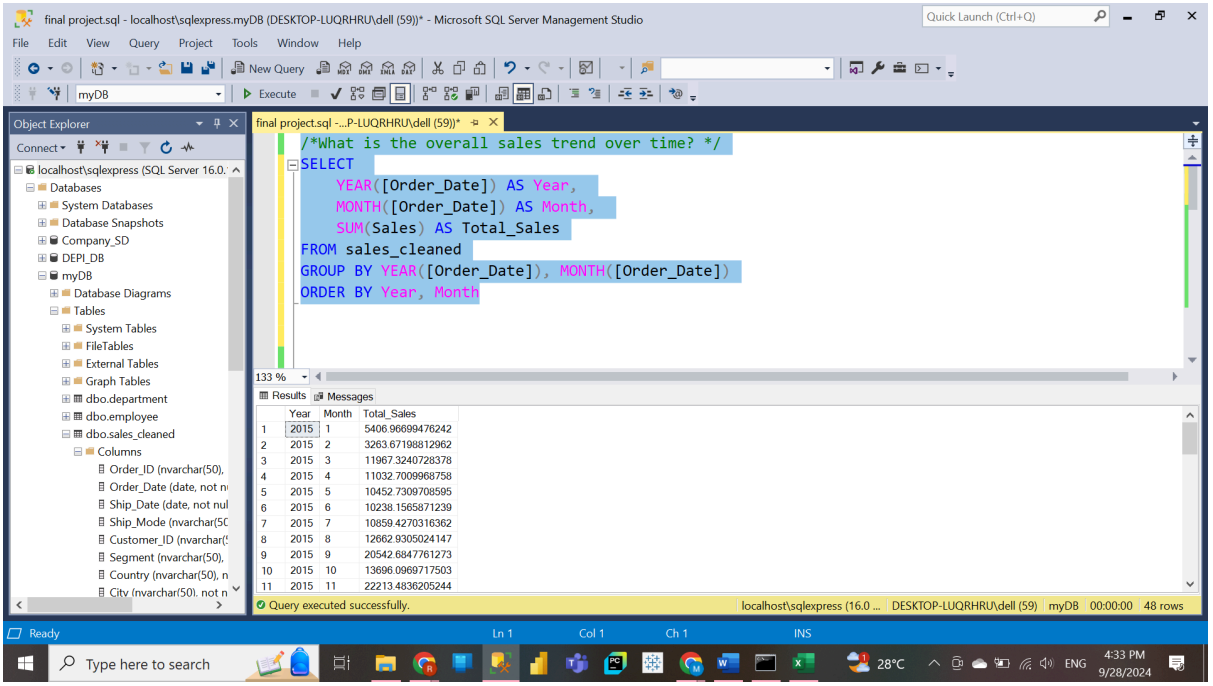
We can categorize potential analysis questions into various business areas such as **sales performance**, **customer behavior**, **regional analysis**, **product performance**, and **logistics**.

Analysis Questions:

1. Sales Performance Analysis:

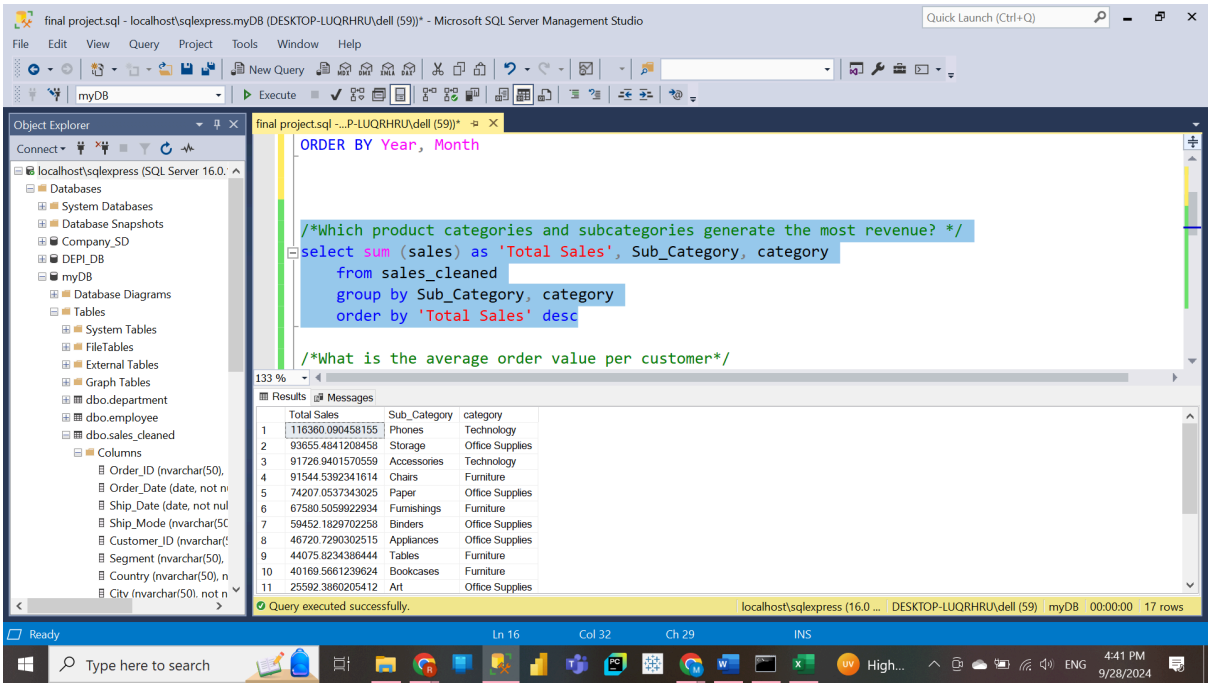
1. What is the overall sales trend over time?

To see the overall sales trend over time, we first needed to Convert the Order Date to a format that allows aggregation (month/year), then we grouped by time period using the following SQL code:



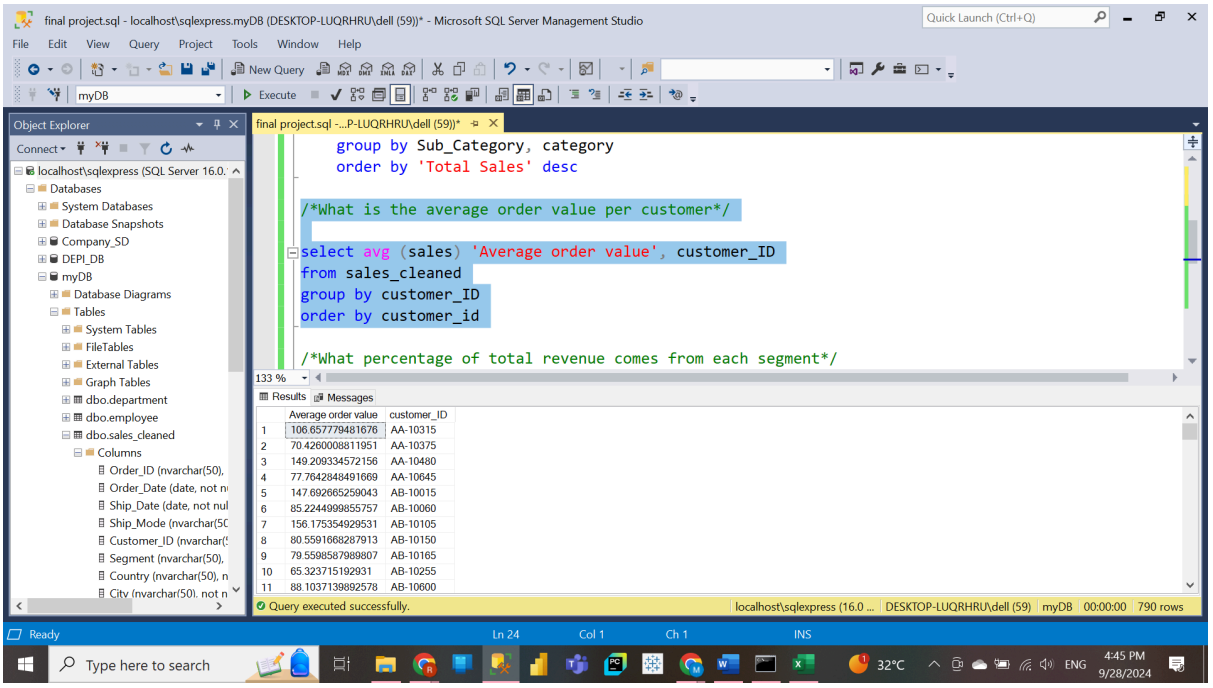
2. Which product categories and subcategories generate the most revenue?

To see which category and sub-category generated the most sales we used the following code:

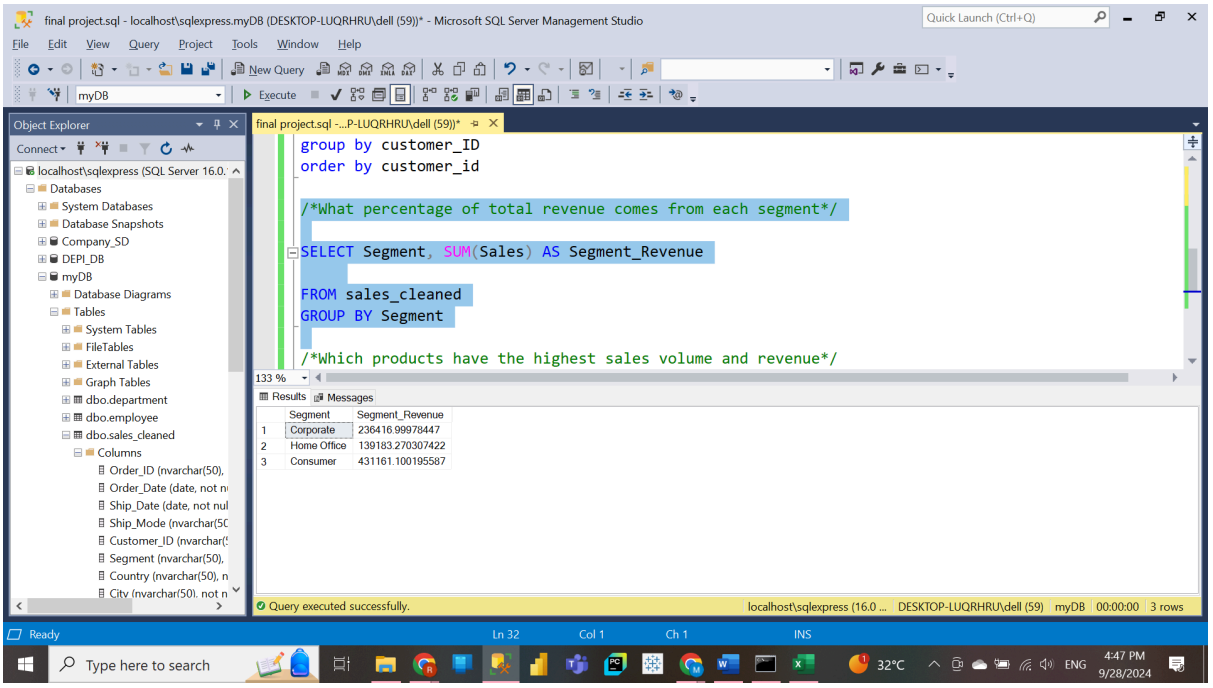


3. What is the average order value per customer?

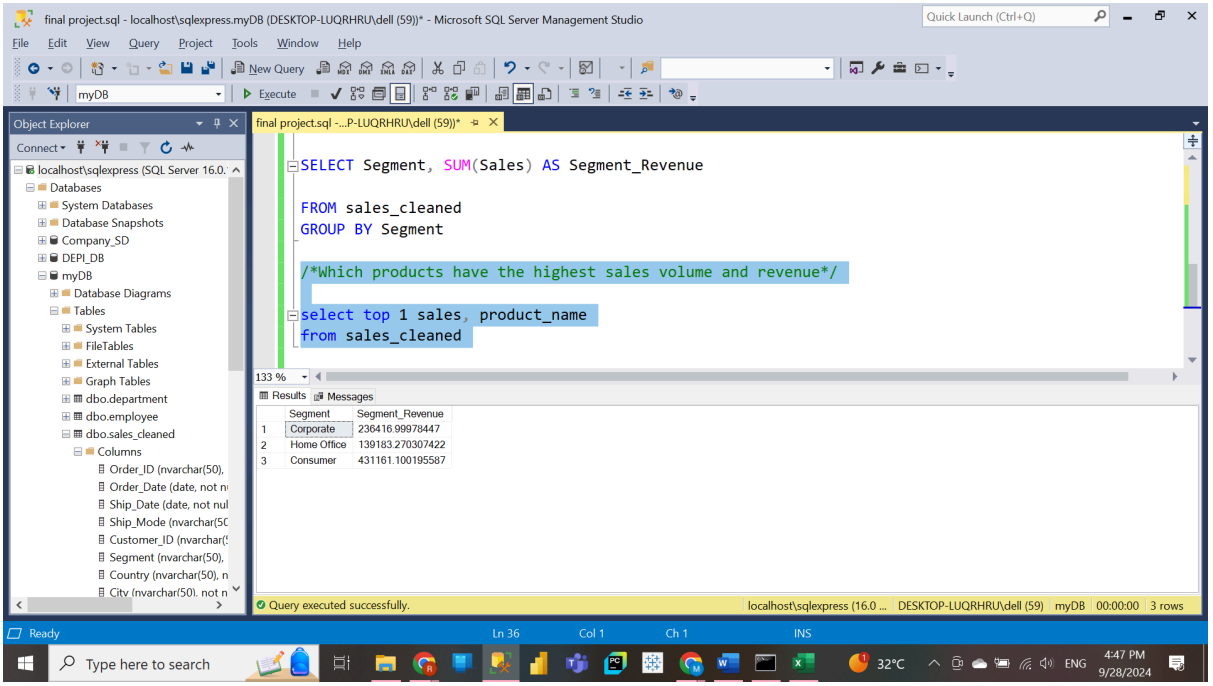
To see the averages sale value for each customer we used the following SQL code:



4. What percentage of total revenue comes from each segment (Consumer, Corporate, Home Office)?



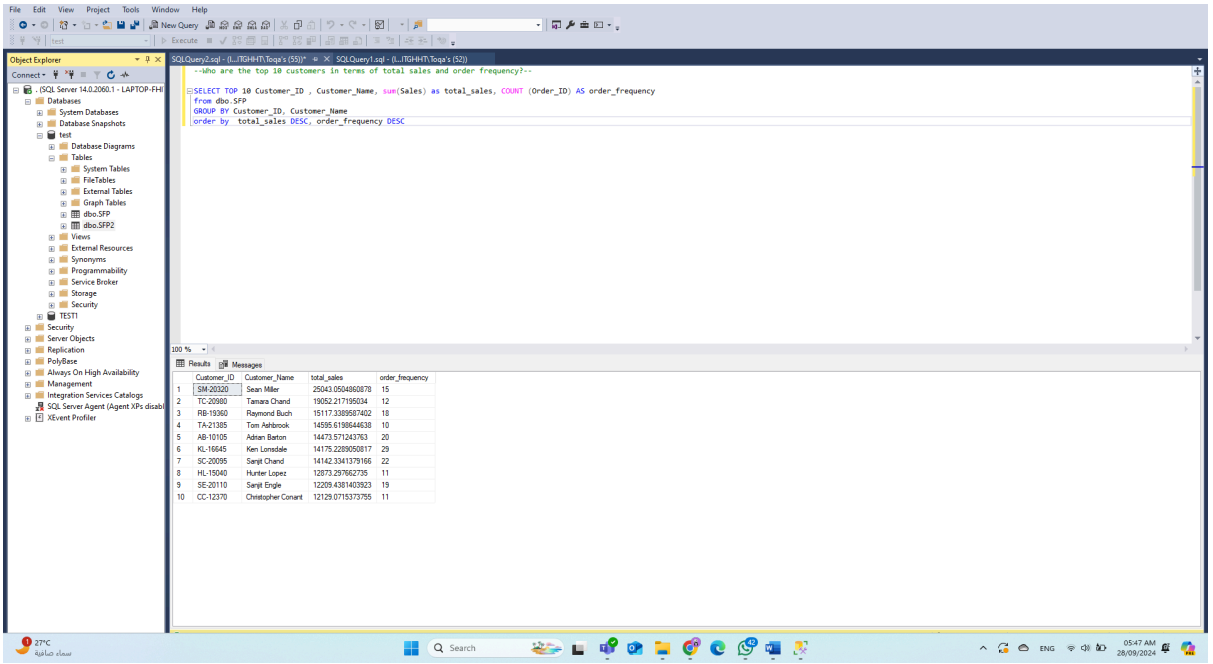
5. Which products have the highest sales volume and revenue?



2. Customer Behavior Analysis:

7. Who are the top 10 customers in terms of total sales and order frequency?

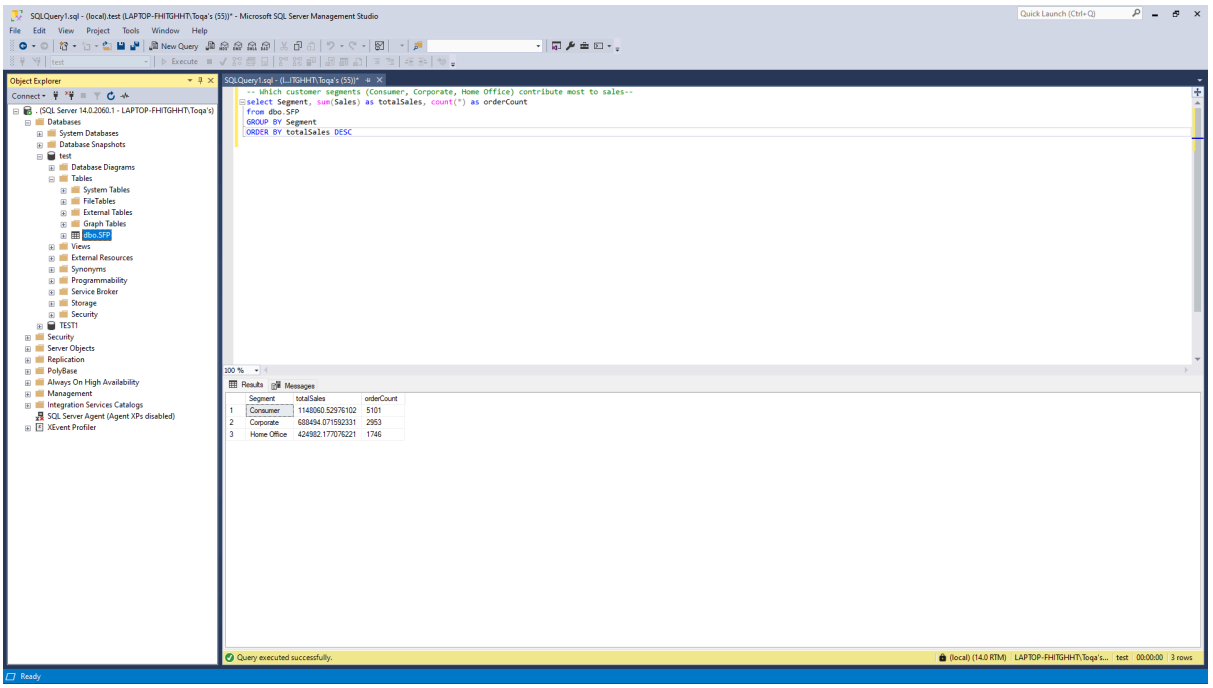
To see the top 10 customers based on their total sales and order frequency, the calculates their total sales by summing the Sales column and counts the number of orders placed was created on SQL to help us answer this question and it was as follows:



As we can see the customer with the highest total sales is SM-20320, Sean Miller, with a total sales of (25043.05). They also placed 15 orders.

8. **Which customer segments (Consumer, Corporate, Home Office) contribute most to sales?**

We will know the most customer segment contributed to sales by group the data by segment, calculate the total revenue and order count for each segment using SQL

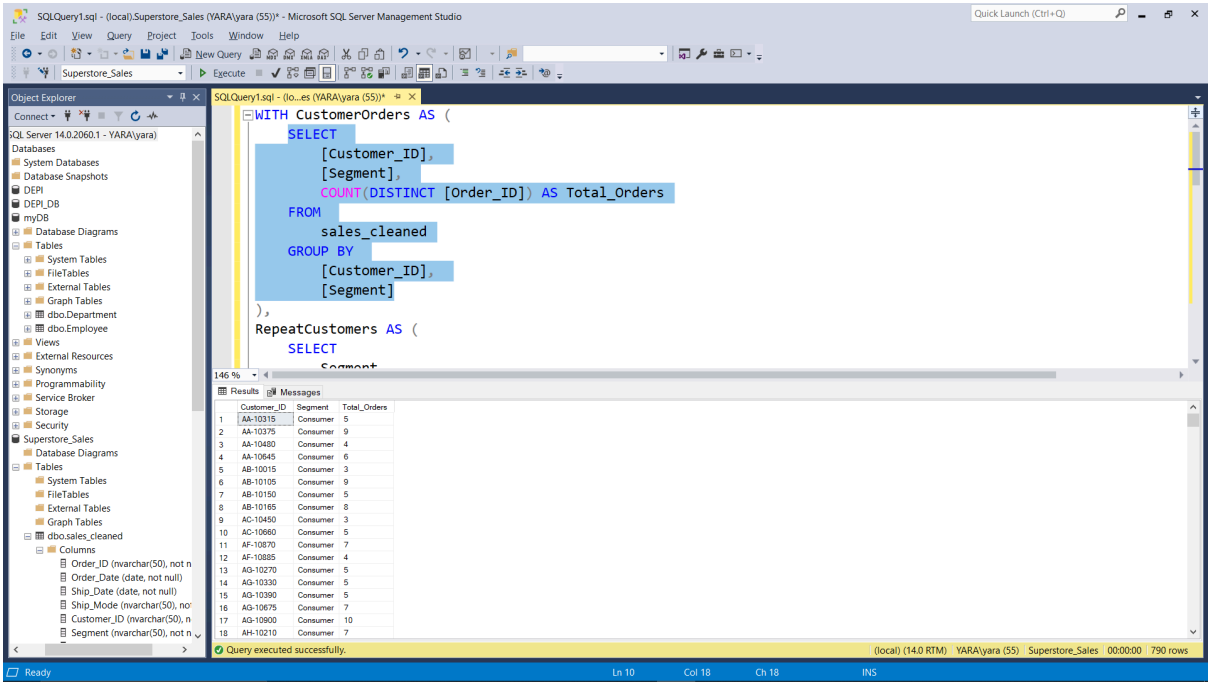


As we can see the customer segment that contributed most to sales is the consumer with total sales of (1148060.53) and order count of 5101

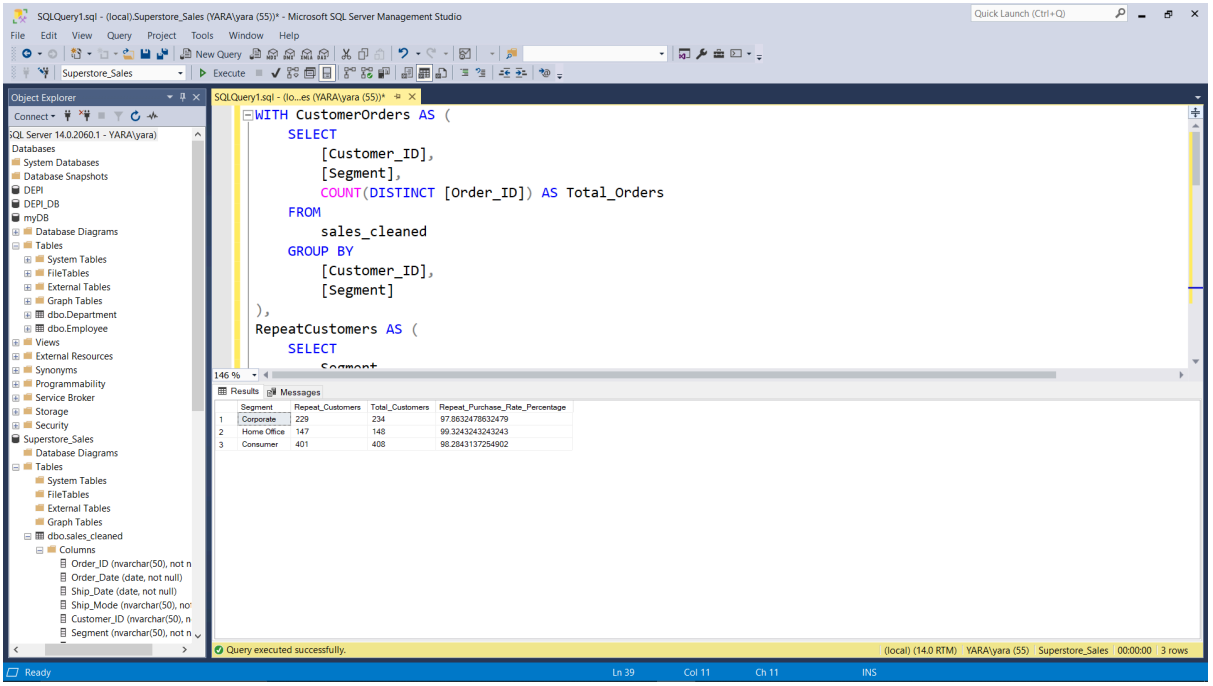
9. **What is the repeat purchase rate by customer segment?**

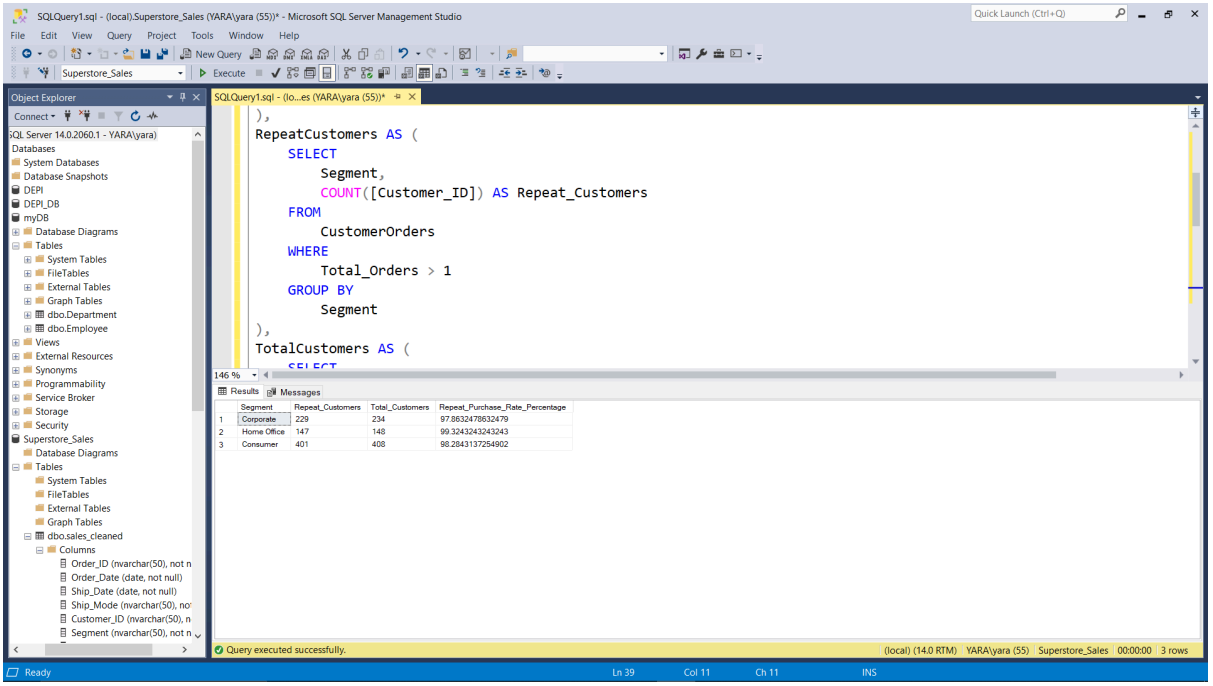
This question wanted to see how many customers made more than one order in each of the three segments (Corporate, Home Office and Consumer). So in order to write the code for this on SQL it needed to be done in multiple steps.

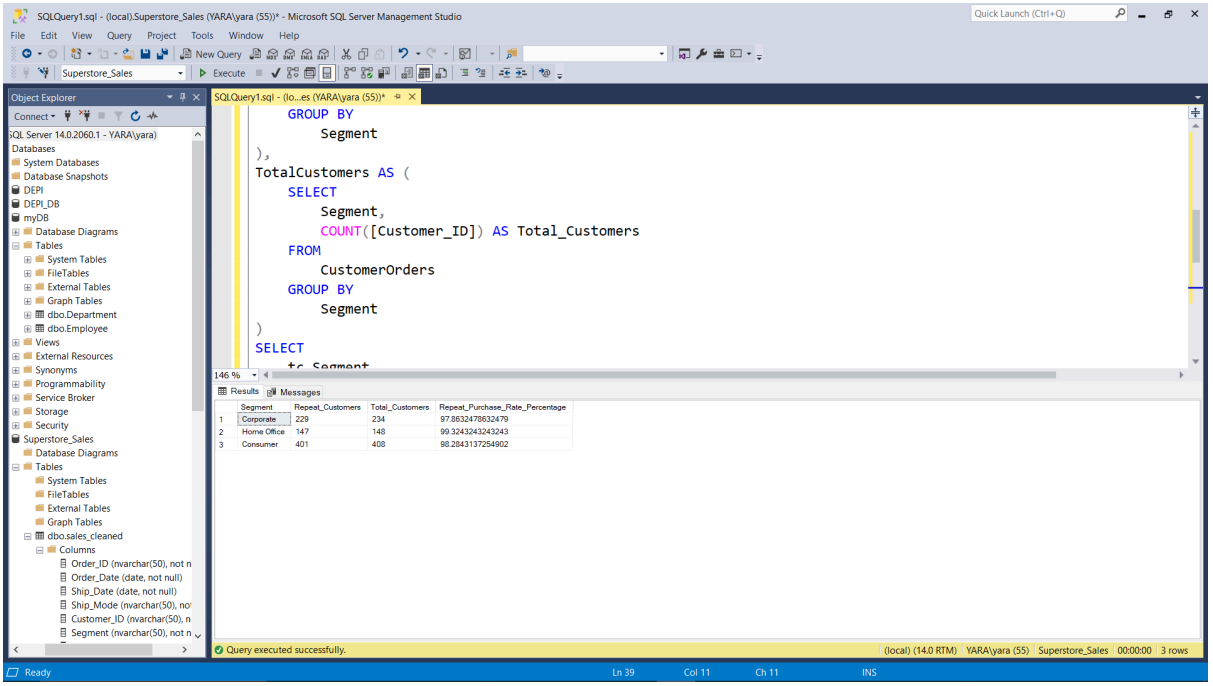
We had to use the “WITH” CTE to reach a final conclusion, starting with a normal SELECT function to find the total orders done by each distinct customer in all the segments and the code was as follows:

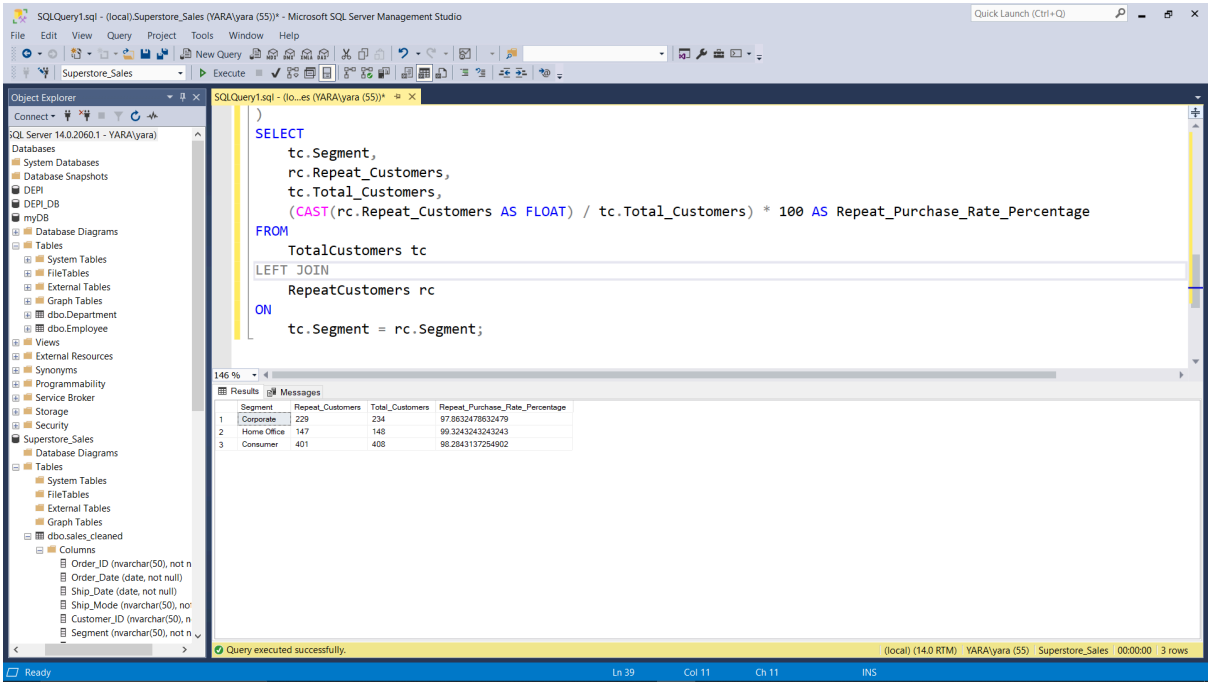


However we needed to find the total repeat customers in each segment not the individual customers which is why the following code was written that gets the total repeat customers and compares it with the total customers in each segment in addition to the percentage of repeated purchases in each segment.


SQLQuery1.sql - (local).Superstore_Sales (YARA\yara (55))* - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
File Edit View Query Project Tools Window Help
New Query
Superstore_Sales
Execute
Object Explorer
Connect
SQL Server 14.0.2060.1 - YARA\yara)
Databases
System Databases
Database Snapshots
DEPI
DEPI_DB
myDB
Database Diagrams
Tables
System Tables
FileTables
External Tables
Graph Tables
dbo.Department
dbo.Employee
Views
External Resources
Synonyms
Programmability
Service Broker
Storage
Security
Superstore_Sales
Database Diagrams
Tables
System Tables
FileTables
External Tables
Graph Tables
dbo.sales_cleaned
Columns
Order_ID (nvarchar(50), not n
Order_Date (date, not null)
Ship_Date (date, not null)
Ship_Mode (nvarchar(50), no
Customer_ID (nvarchar(50), n
Segment (nvarchar(50), not n
SQLQuery1.sql - (lo...es (YARA\yara (55))*
WITH CustomerOrders AS (
SELECT
[Customer_ID],
[Segment],
COUNT(DISTINCT [Order_ID]) AS Total_Orders
FROM
sales_cleaned
GROUP BY
[Customer_ID],
[Segment]
),
RepeatCustomers AS (
SELECT
146 %
Results
Messages
Segment Repeat_Customers Total_Customers Repeat_Purchase_Rate_Percentage
1 Corporate 229 234 97.8632478632479
2 Home Office 147 148 99.3243243243243
3 Consumer 401 408 98.2843137254902
Query executed successfully.
(local) (14.0 RTM) YARA\yara (55) Superstore_Sales 00:00:00 3 rows
Ready
Ln 39 Col 11 Ch 11 INS


SQLQuery1.sql - (local).Superstore_Sales (YARA\yara (55))* - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
File Edit View Query Project Tools Window Help
New Query
Superstore_Sales
Execute
Object Explorer
Connect
SQL Server 14.0.2060.1 - YARA\yara)
Databases
System Databases
Database Snapshots
DEPI
DEPI_DB
myDB
Database Diagrams
Tables
System Tables
FileTables
External Tables
Graph Tables
dbo.Department
dbo.Employee
Views
External Resources
Synonyms
Programmability
Service Broker
Storage
Security
Superstore_Sales
Database Diagrams
Tables
System Tables
FileTables
External Tables
Graph Tables
dbo.sales_cleaned
Columns
Order_ID (nvarchar(50), not n
Order_Date (date, not null)
Ship_Date (date, not null)
Ship_Mode (nvarchar(50), no
Customer_ID (nvarchar(50), n
Segment (nvarchar(50), not n
SQLQuery1.sql - (lo...es (YARA\yara (55))*
),
RepeatCustomers AS (
SELECT
Segment,
COUNT([Customer_ID]) AS Repeat_Customers
FROM
CustomerOrders
WHERE
Total_Orders > 1
GROUP BY
Segment
),
TotalCustomers AS (
146 %
Results
Messages
Segment Repeat_Customers Total_Customers Repeat_Purchase_Rate_Percentage
1 Corporate 229 234 97.8632478632479
2 Home Office 147 148 99.3243243243243
3 Consumer 401 408 98.2843137254902
Query executed successfully.
(local) (14.0 RTM) YARA\yara (55) Superstore_Sales 00:00:00 3 rows
Ready
Ln 39 Col 11 Ch 11 INS

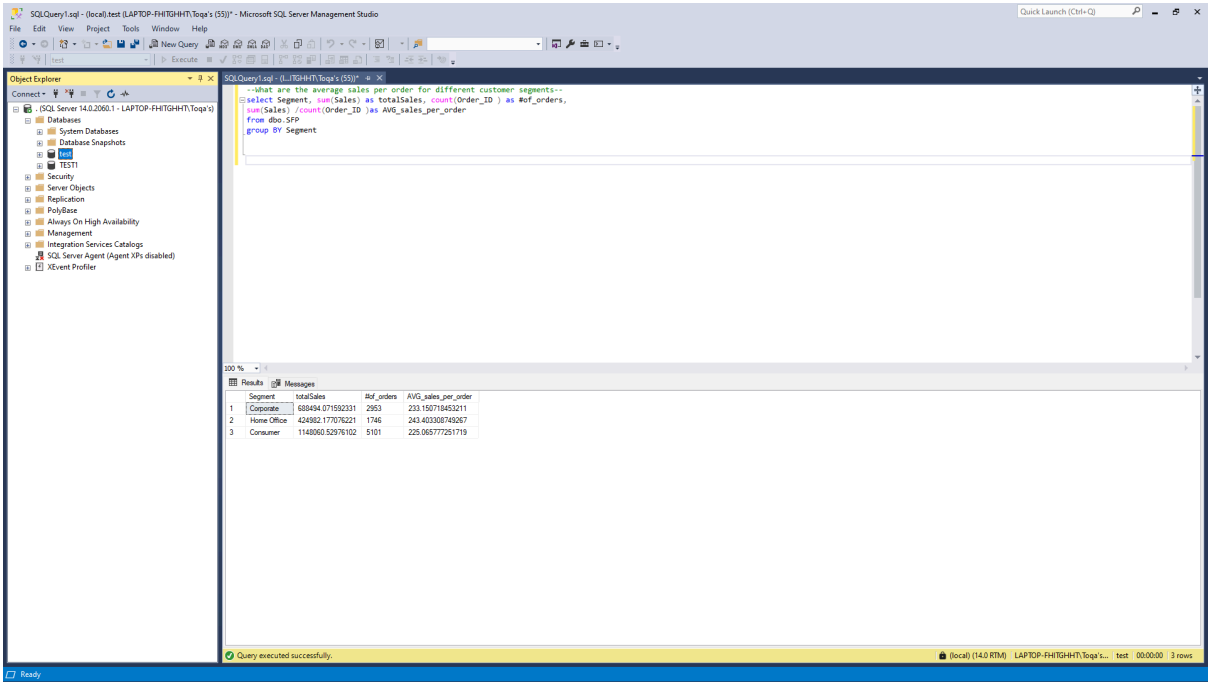




As we can see the home office segment had the highest percentage of repeat customers (99.3%), followed by the consumer segment (98.28%) and the corporate segment was in third place with (97.86%).

10. What are the average sales per order for different customer segments?

To get the average sales per order we must get first the total sales and number of orders by SQL:

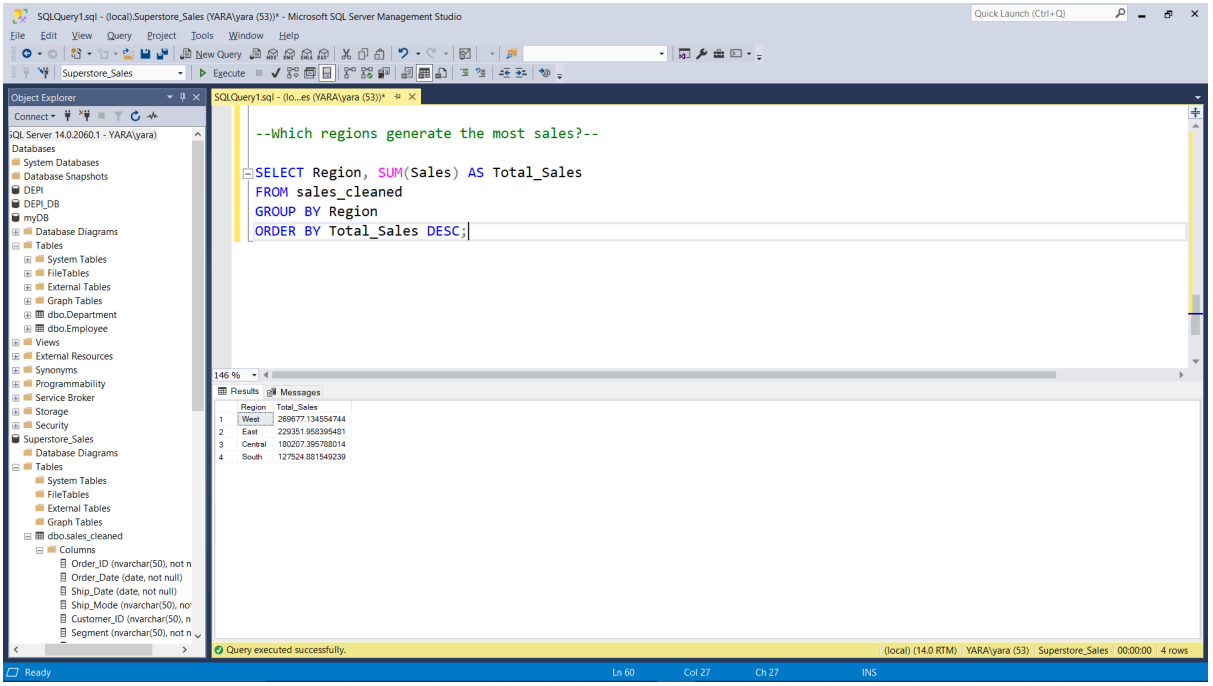


As we can see The "Consumer" segment generated the highest total sales, with a total of (1148060.52). The "Corporate" segment placed the most orders, with a total of 2953 orders. While the "Home Office" segment had the highest average sales per order, with an average of (243.4).

3. Regional Analysis:

11. Which regions generate the most sales?

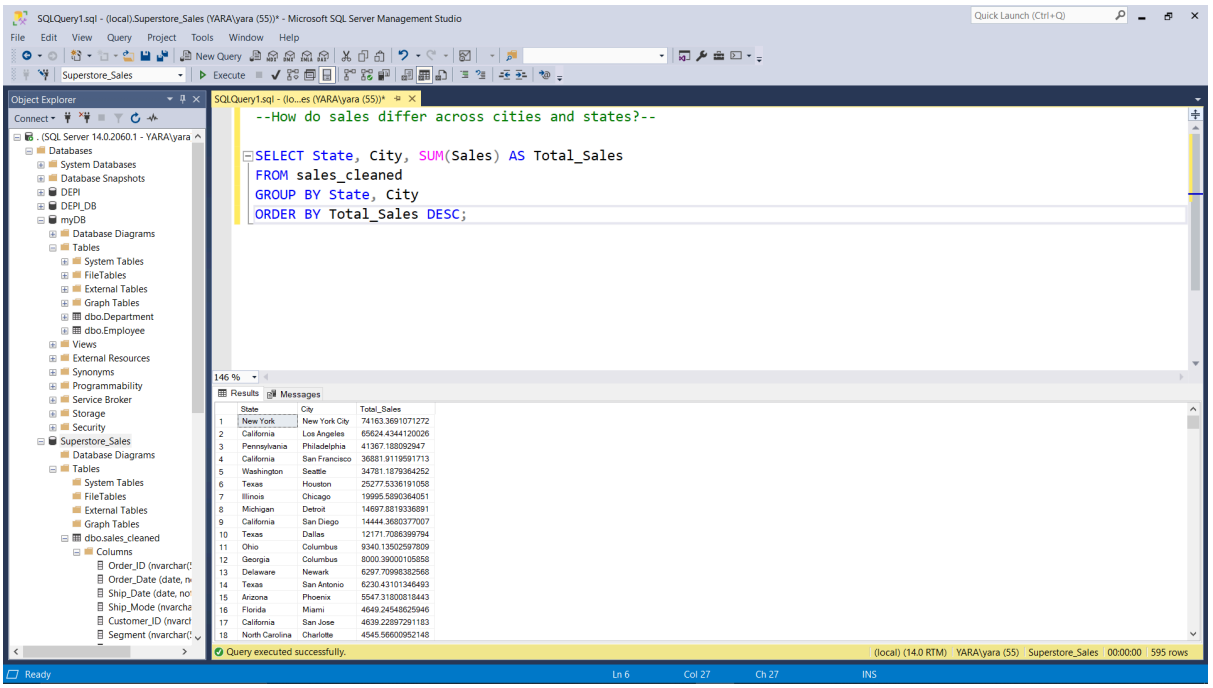
To see which region in the states has the highest sales, the total sales in each region were summed and a code was created on SQL to help us answer this question and it was as follows:



As we can see the highest sales of (269677.13$) were in the West region. Followed by the East with sales (229351.96$). The Central region had the 3rd highest sales of (180207.395$). Finally in fourth place was the South region with sales of (34781.19$)

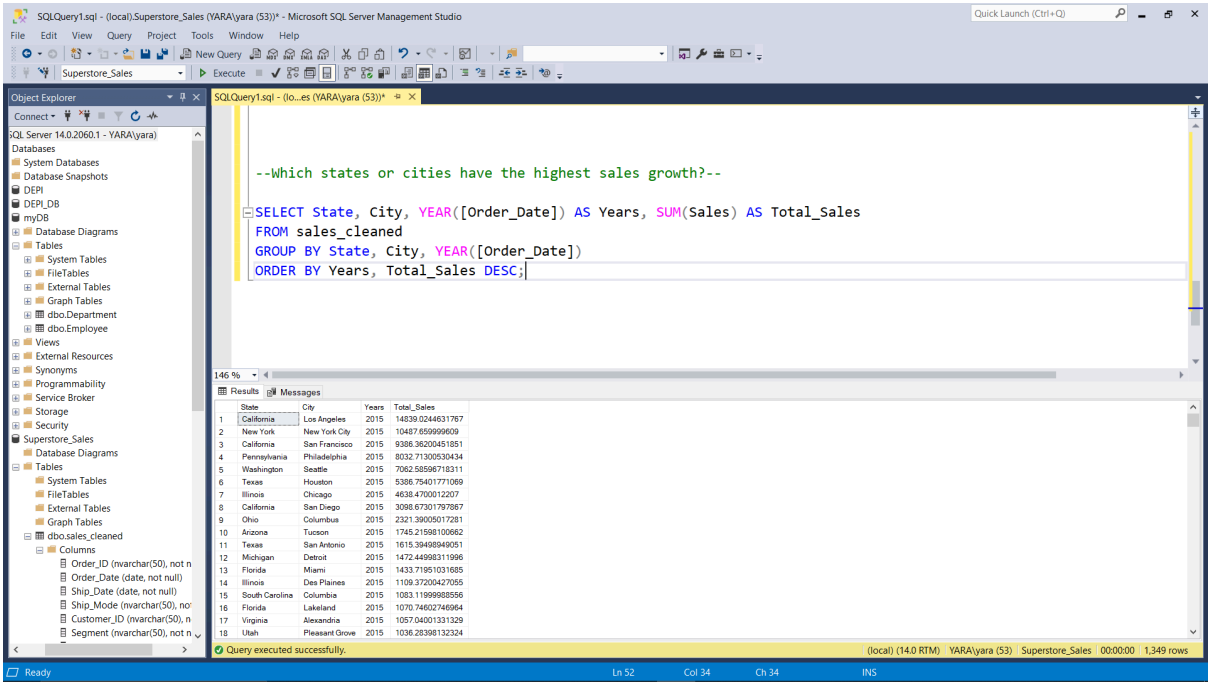
12. How do sales differ across Cities and States?

This question was answered using a code that shows the different Cities and States to see which states and cities should be focused on. The highest sales of (74163.37$) were in New York City in the state of New York. while the lowest were in The city of Abilene in the state of Texas with sales being only (1.39$).



13. Which states or cities have the highest sales growth?

For this question we wanted to know which States and cities had the highest growth overtime, so a code was written to show the highest sales in every state and city in all the years available in the dataset. The results and code written were as follows:

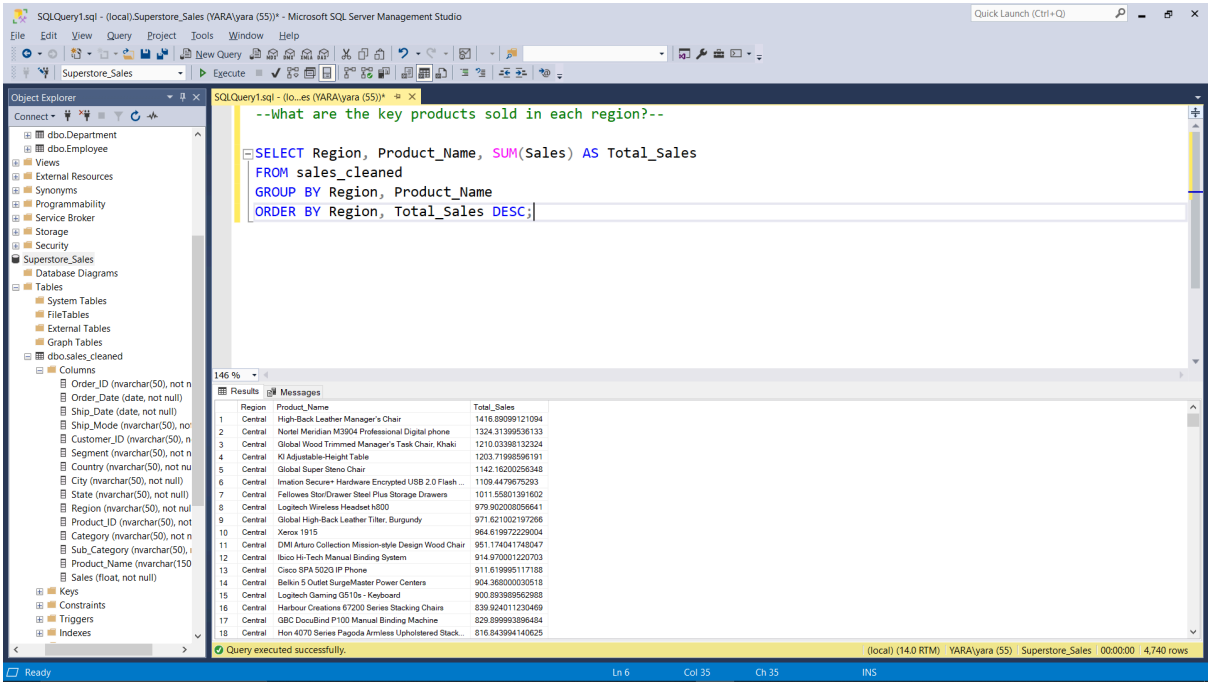


As we can see the results show the highest sales in every year grouped by every city and state. In 2015 Los Angeles was in the lead with total sales (14839.02$). In 2016 New York City took the lead with total sales of (15216.41$). It took the lead again in 2017 and further increased its sales to a total of (23182.29$). Once again in 2018 New York City had the highest sales in 2018 that reached a total of (25277$). Making New York City the region with the highest sales growth.

14. What are the key products sold in each region?

This question is used to find the highest selling product in each region. Starting with the Central Region the highest selling product is “High-Back Leather Manager's Chair” with total sales of (1416.89$). The East Region had the highest sales of (1483.48$) from the product “Situations Contoured Folding Chairs, 4/Set”. In the South Region the highest selling product was “Gould Plastics 18-Pocket Panel Bin, 34w x 5-1/4d x 20-1/2h” with sales of (1232.67$). Finally the West Region’s highest sales were (1777.05$) with the product “Fellowes Super Stor/Drawer Files”.

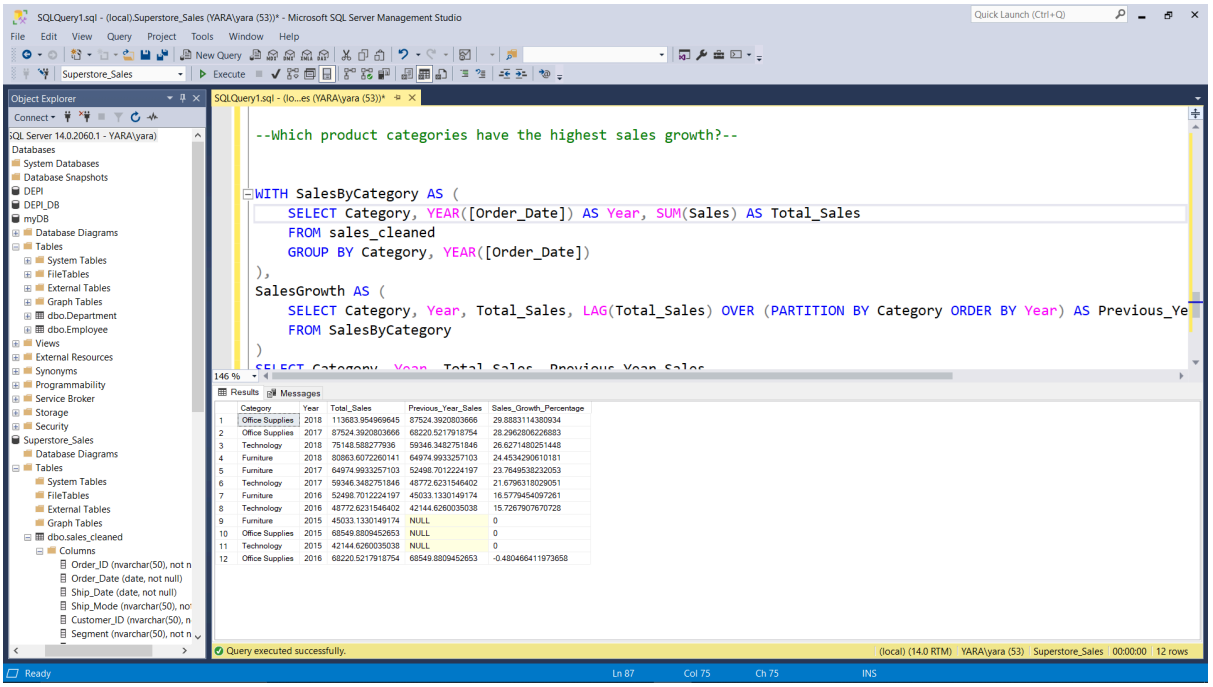
The following shows the code written on SQL and the results:

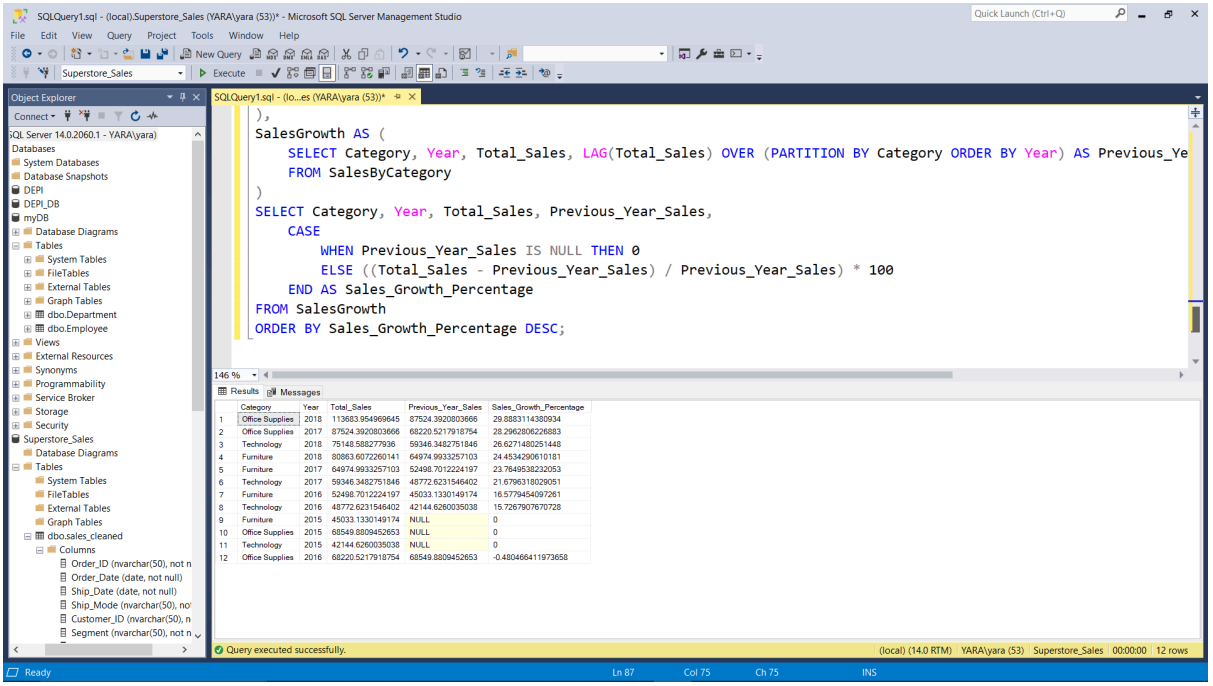


4. Product Performance Analysis:

15. Which product categories have the highest sales growth?

For this question we needed to calculate the total sales of each product category over the years so a code was used to group the sales by the category and years. It was then followed by a code to calculate the sales growth and its percentage every year and for this case the LAG function was used to calculate the previous year's sales. Finally the percentage was calculated and then arranged in descending order.



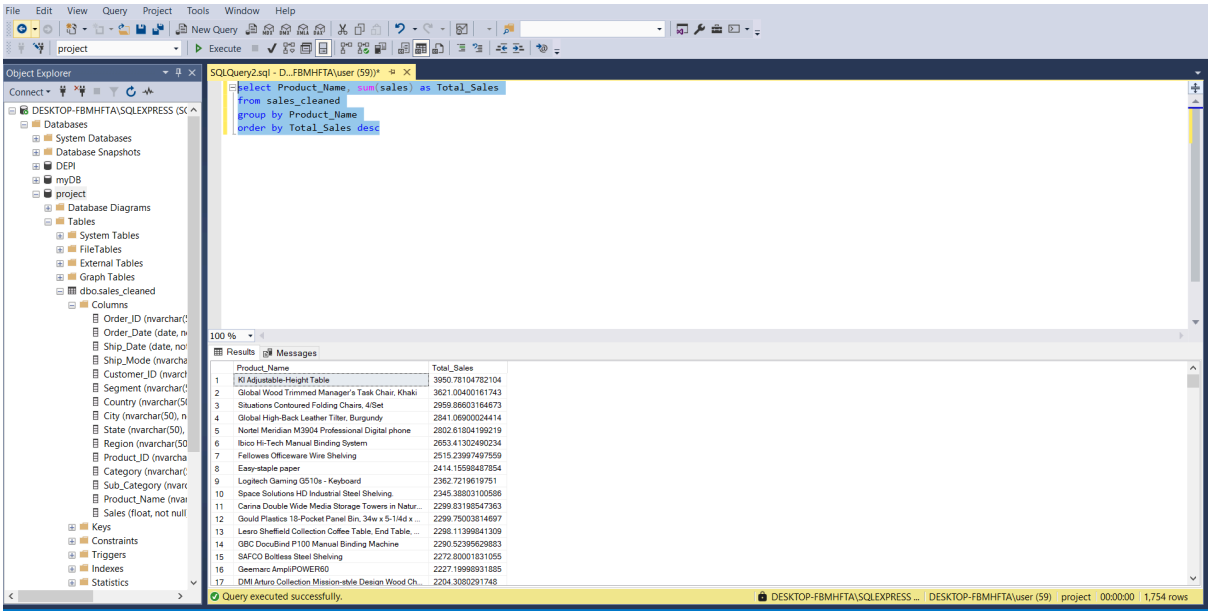


As we can see in the results the year 2015 can't be considered as there's no previous year's data. However we can see that in 2018 the office supplies had the highest growth rate of (29.9%). While the office supplies had a negative growth in rate in 2016 of (-0.84%) putting it last place even though it has higher total sales than the other categories.

16. Which products have the highest return rates or lowest sales performance?

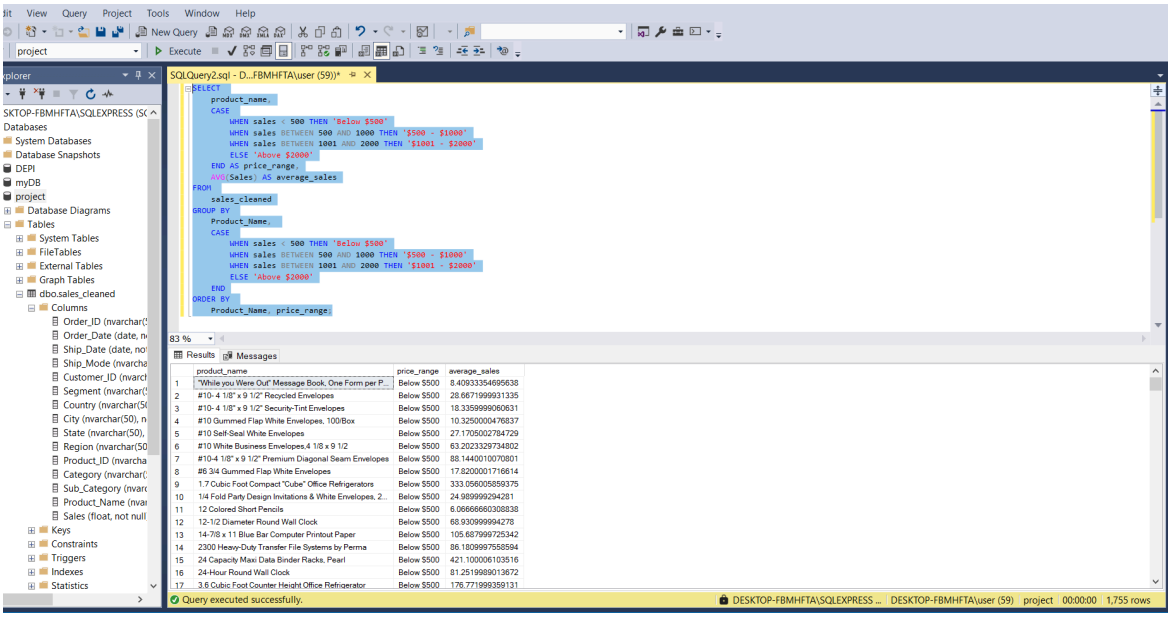
Product with highest return: KI Adjustable - height table

Product with lowest sales performance: Eureka Disposable Bags



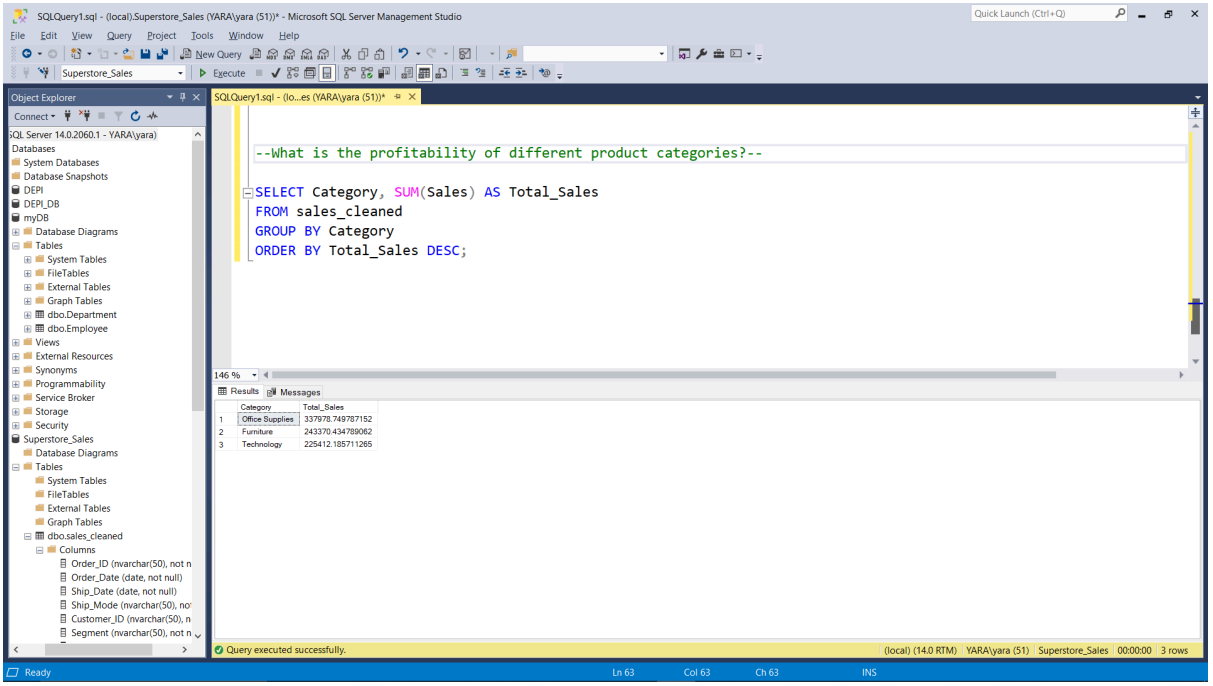
17. How does product price impact sales?

Average sales per product price range. First we indicated the price ranges and calculated the average sales for each product in said ranges.



18. What is the profitability of different product categories?

There is no column for costs in the table, so it's not possible to calculate profit which is why we'll use the sales instead to determine which product category has the highest sales.

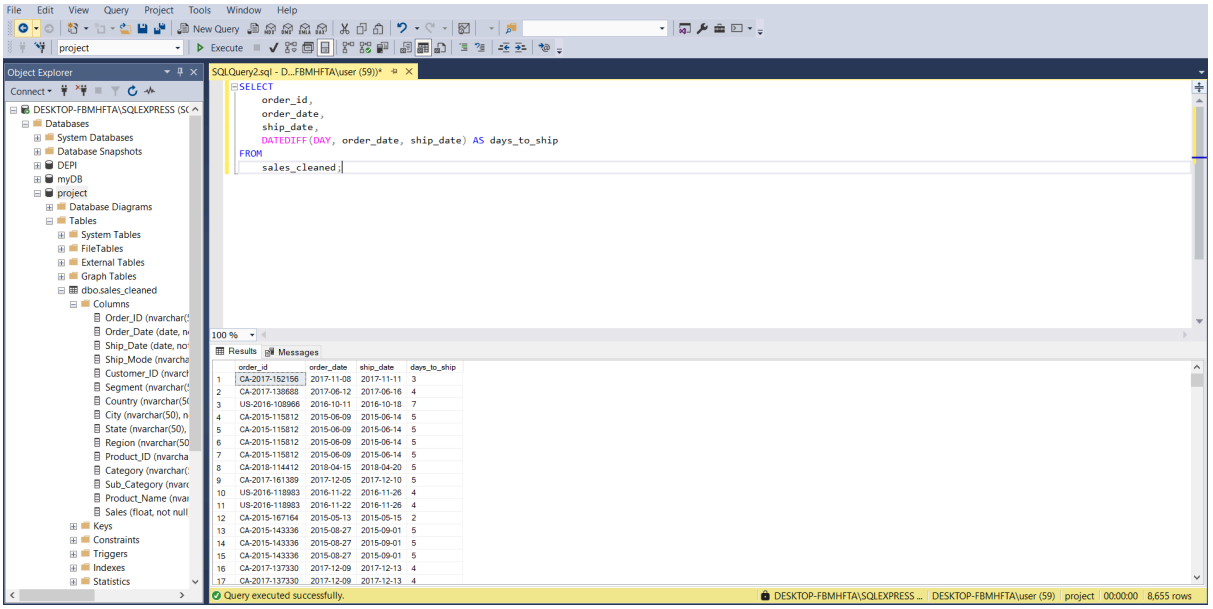


As we can see, the Office supplies category has the highest total sales of (337978.75$) followed by Furniture and Technology with sales respectively (243370.43$) and (225412.19$).

5. Logistics and Operational Analysis:

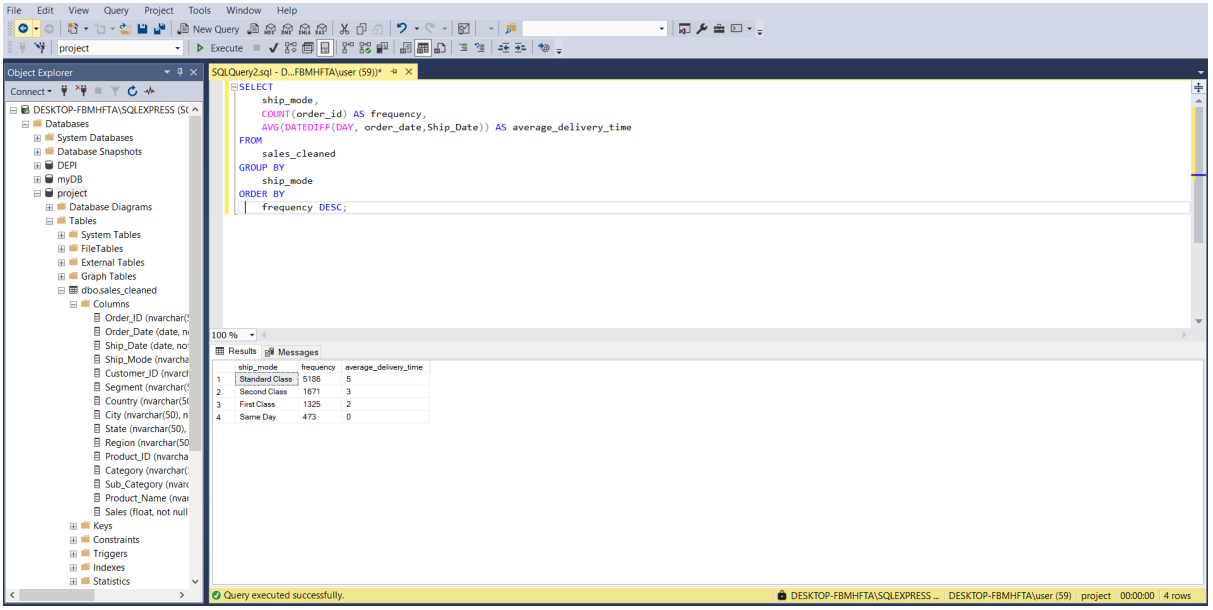
19. What is the average delivery time from order to shipping for each shipping mode?

Calculated the time difference between order date and shipping date using the built in function DATEDIFF.

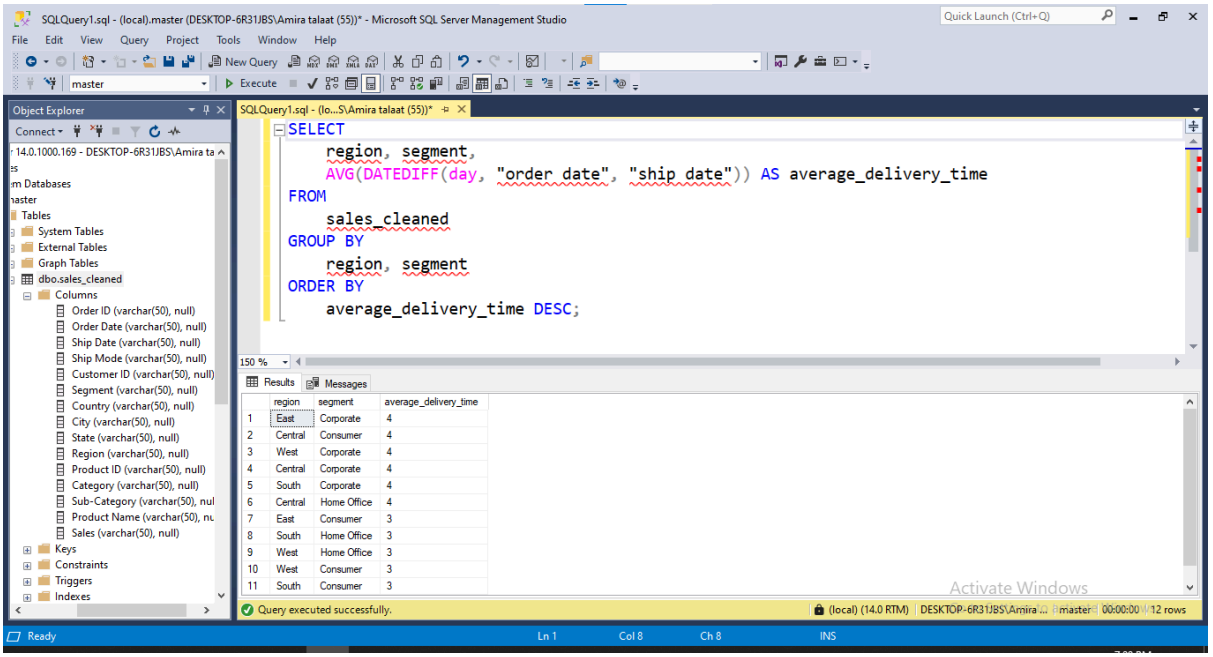


20. Which shipping methods are the most used by customers, and how do they impact delivery times?

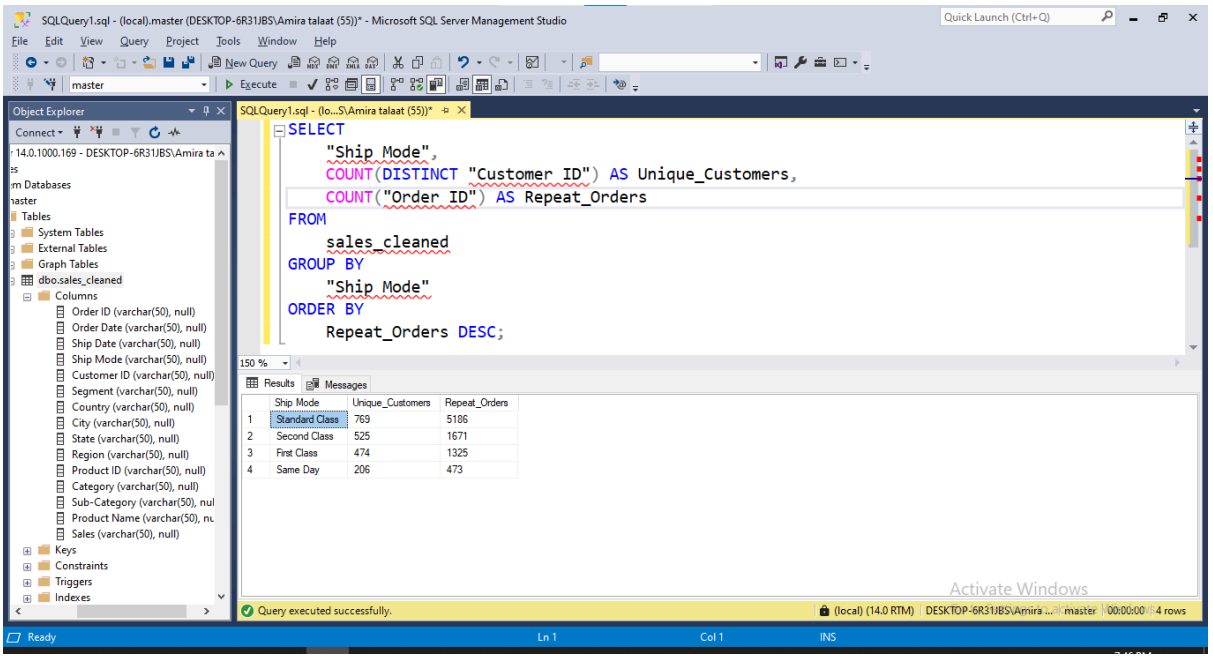
Shipping mode frequency and its impact on delivery times by calculating the number of orders for each shipping mode and the average delivery time in days for each delivery mode.



21. Are there any trends between shipping time and specific regions or customer segments?

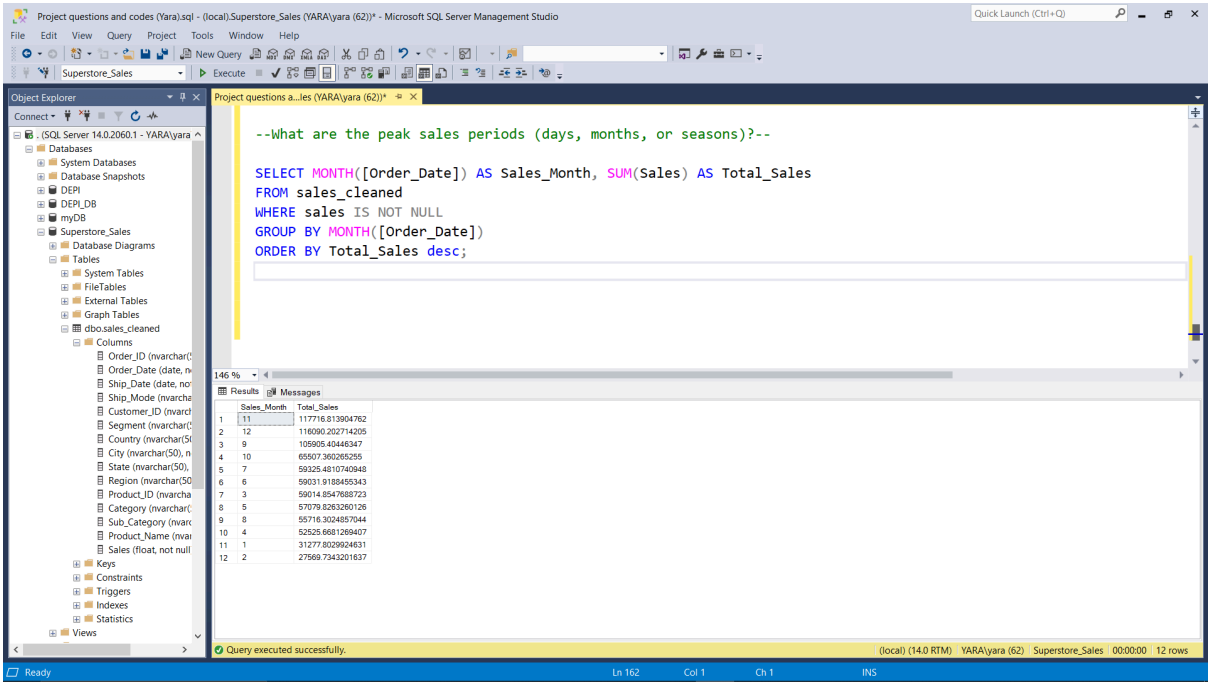


22. What is the correlation between shipping mode and customer satisfaction (using repeat orders)?



6. Time-Based Analysis:

23. What are the peak sales periods (days, months, or seasons)?



24. How do sales change over holidays or promotional periods?

As we can see in the previous results the months of (9, 10, 11 and 12) had the highest sales and these months coincide with the christmas holiday showing how the holidays can have a large effect in increasing the customers willingness to buy products and the store should make preparations for the holiday seasons by increasing their products.

Task 3:

Week 3: Forecasting Questions Phase

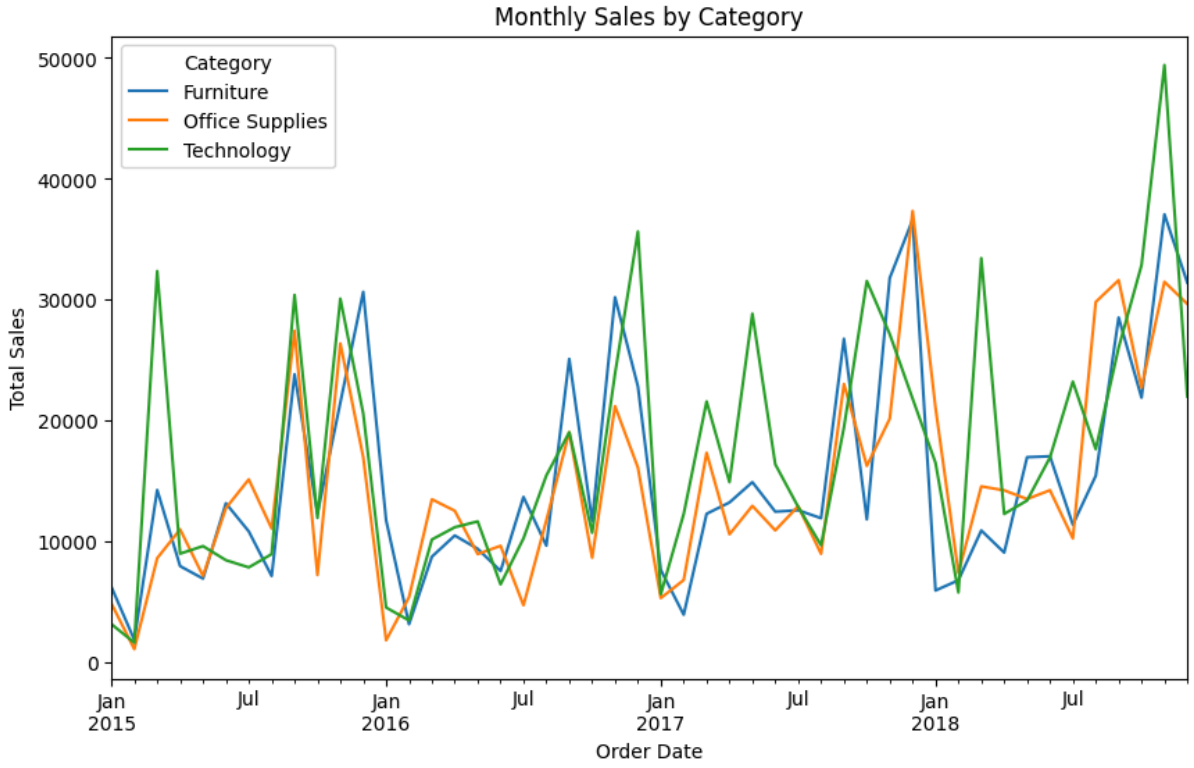
- **Tasks:**
  - ➔ Determine a set of forecasting questions and answer them using the trends found in the given dataset.
  - ➔ Tools: Python (scikit-learn, pandas, Matplotlib).
- **Deliverables:**
  - ➔ Visualization plots answering forecasting questions.

For our data set the following questions will be used to give us insight into the trends we have in the data and help the store predict the future sales to be prepared with appropriate inventory that can match the demand in the holiday seasons as well as in normal months.

Forecasting Questions:

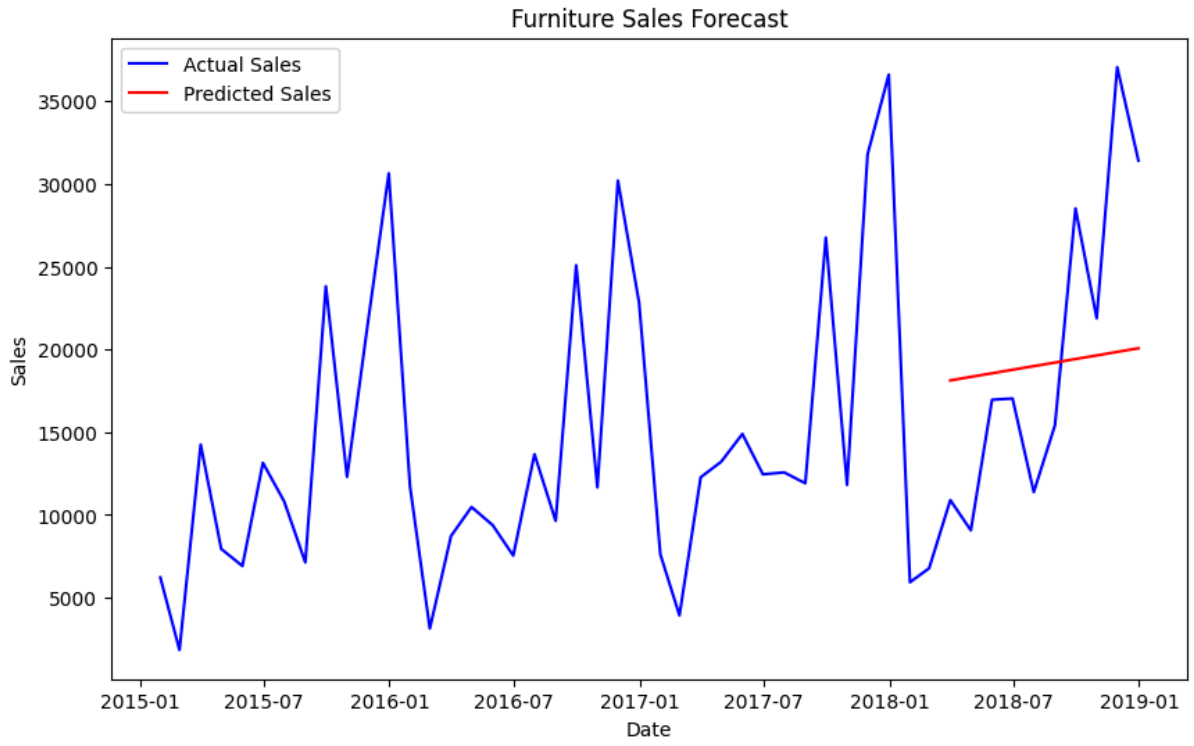
1. **What will the sales be in the next month/quarter/year for each product category?**
    - **Forecasting sales trends helps plan for inventory, marketing, and resource allocation.**

```
df_cleaned['Order Date'] = pd.to_datetime(df_cleaned['Order Date'],
errors='coerce')

monthly_sales = df_cleaned.groupby([pd.Grouper(key='Order Date',
freq='M'), 'Category'])['Sales'].sum().reset_index()

     monthly_sales_pivot = monthly_sales.pivot(index='Order Date',
columns='Category', values='Sales')

monthly_sales_pivot.plot(figsize=(10, 6), title='Monthly Sales by
Category')

plt.ylabel('Total Sales')

plt.show()
```

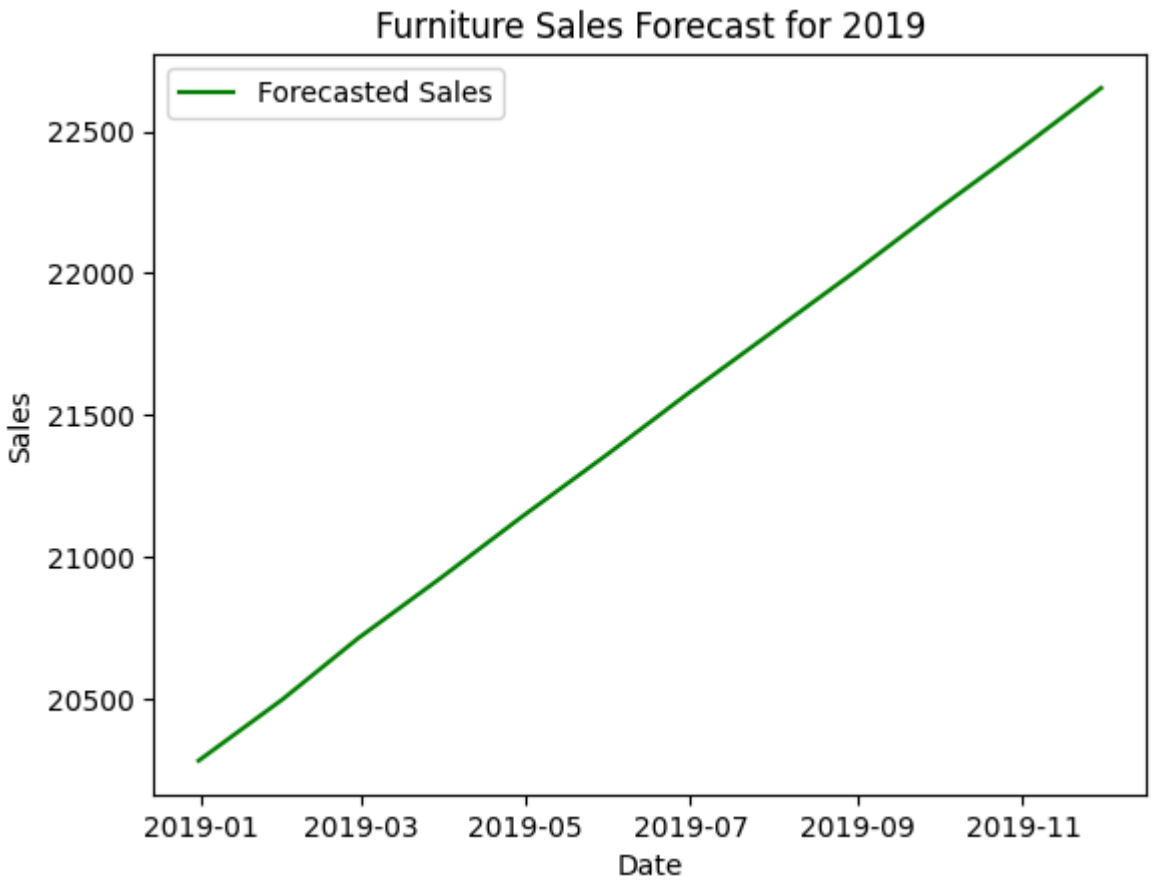



```
from sklearn.model_selection import train_test_split
```

```
from sklearn.linear_model import LinearRegression

furniture_sales =
monthly_sales_pivot['Furniture'].dropna().reset_index()

furniture_sales['Month_Num'] = np.arange(len(furniture_sales))

X = furniture_sales[['Month_Num']]

y = furniture_sales['Furniture']

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, shuffle=False)

model = LinearRegression()

model.fit(X_train, y_train)

y_pred = model.predict(X_test)

plt.figure(figsize=(10, 6))

plt.plot(furniture_sales['Order Date'], y, label='Actual Sales',
color='blue')

plt.plot(furniture_sales['Order Date'][len(X_train):], y_pred,
label='Predicted Sales', color='red')

plt.title('Furniture Sales Forecast')

plt.xlabel('Date')

plt.ylabel('Sales')

plt.legend()

plt.show()
```

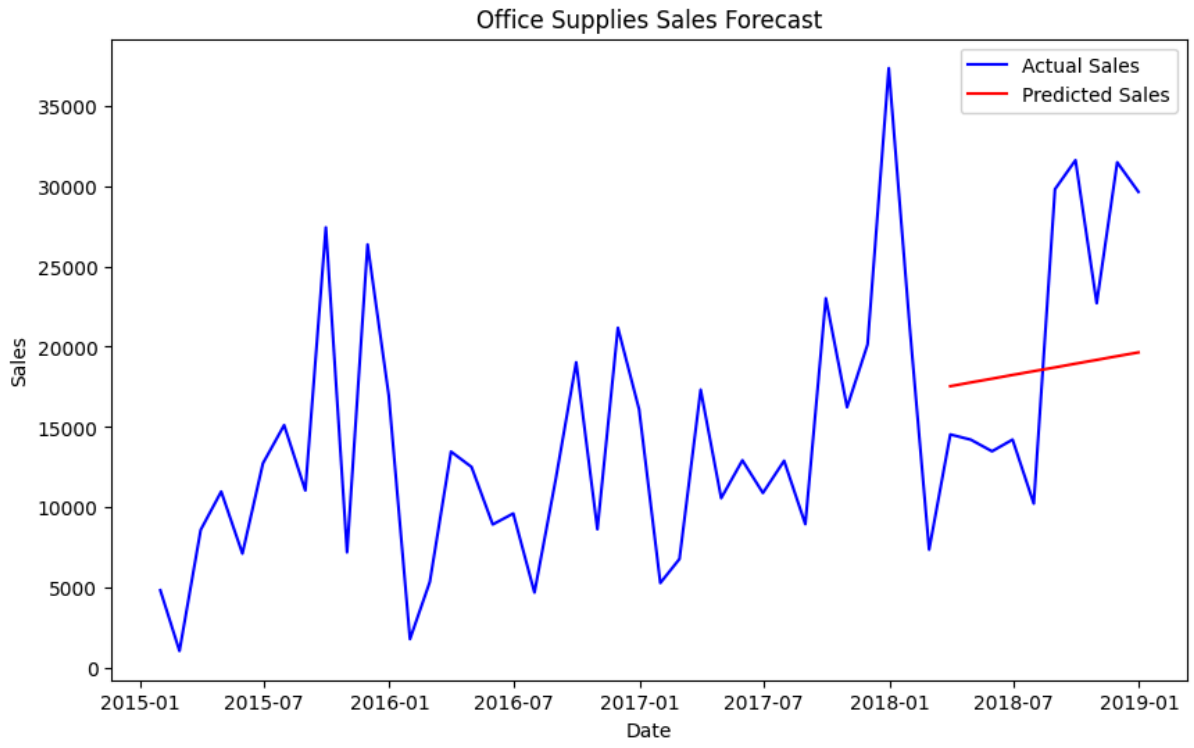



```
future_months = np.arange(len(furniture_sales), len(furniture_sales) +
12).reshape(-1, 1)

future_sales_pred = model.predict(future_months)

future_dates = pd.date_range(furniture_sales['Order Date'].max(),
periods=12, freq='ME')

plt.plot(future_dates, future_sales_pred, label='Forecasted Sales',
color='green')

plt.title('Furniture Sales Forecast for 2019')

plt.xlabel('Date')

plt.ylabel('Sales')

plt.legend()

plt.show()
```

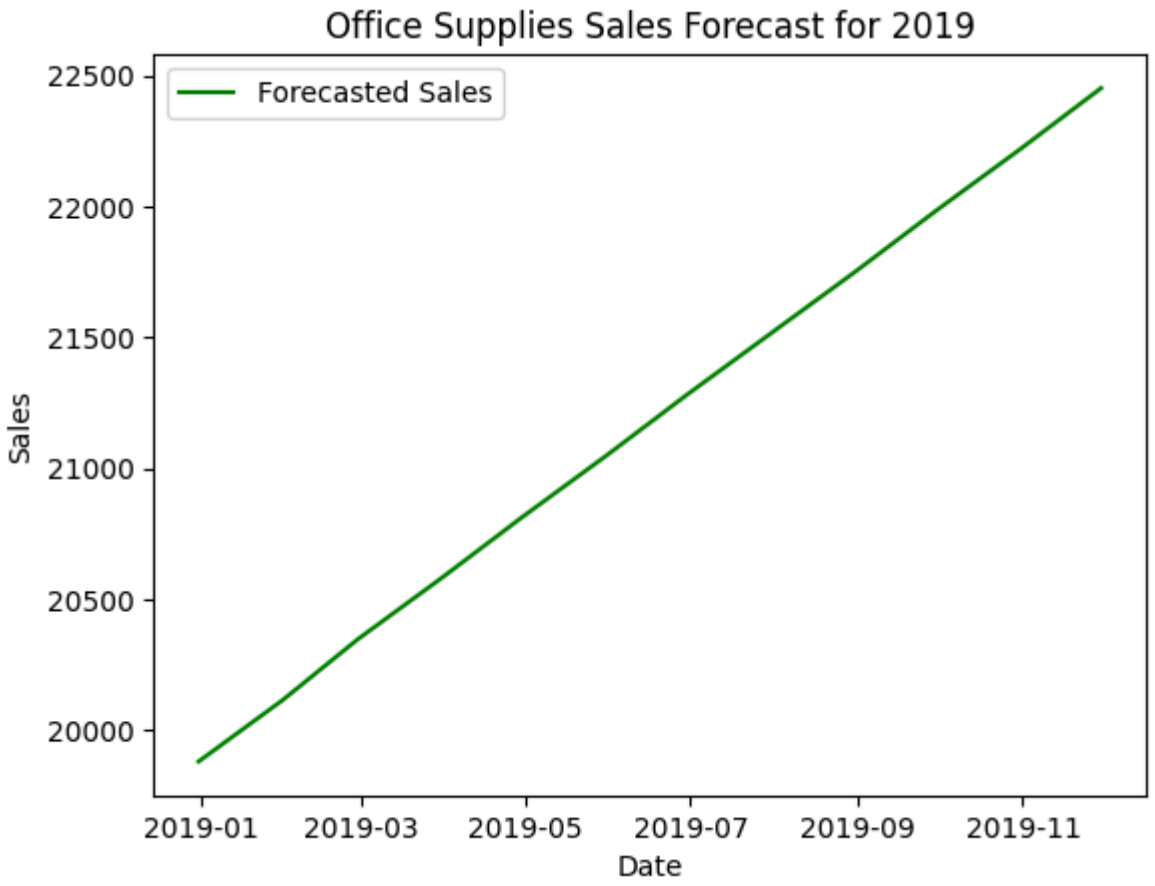



```
office_supplies_sales = monthly_sales_pivot['Office
Supplies'].dropna().reset_index()


office_supplies_sales['Month_Num'] =
np.arange(len(office_supplies_sales))


X = office_supplies_sales[['Month_Num']]

y = office_supplies_sales['Office Supplies']


X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, shuffle=False)


model = LinearRegression()
```

```
model.fit(X_train, y_train)



y_pred = model.predict(X_test)



plt.figure(figsize=(10, 6))

plt.plot(office_supplies_sales['Order Date'], y, label='Actual Sales',
color='blue')

plt.plot(office_supplies_sales['Order Date'][len(X_train):], y_pred,
label='Predicted Sales', color='red')

plt.title('Office Supplies Sales Forecast')

plt.xlabel('Date')

plt.ylabel('Sales')

plt.legend()

plt.show()
```

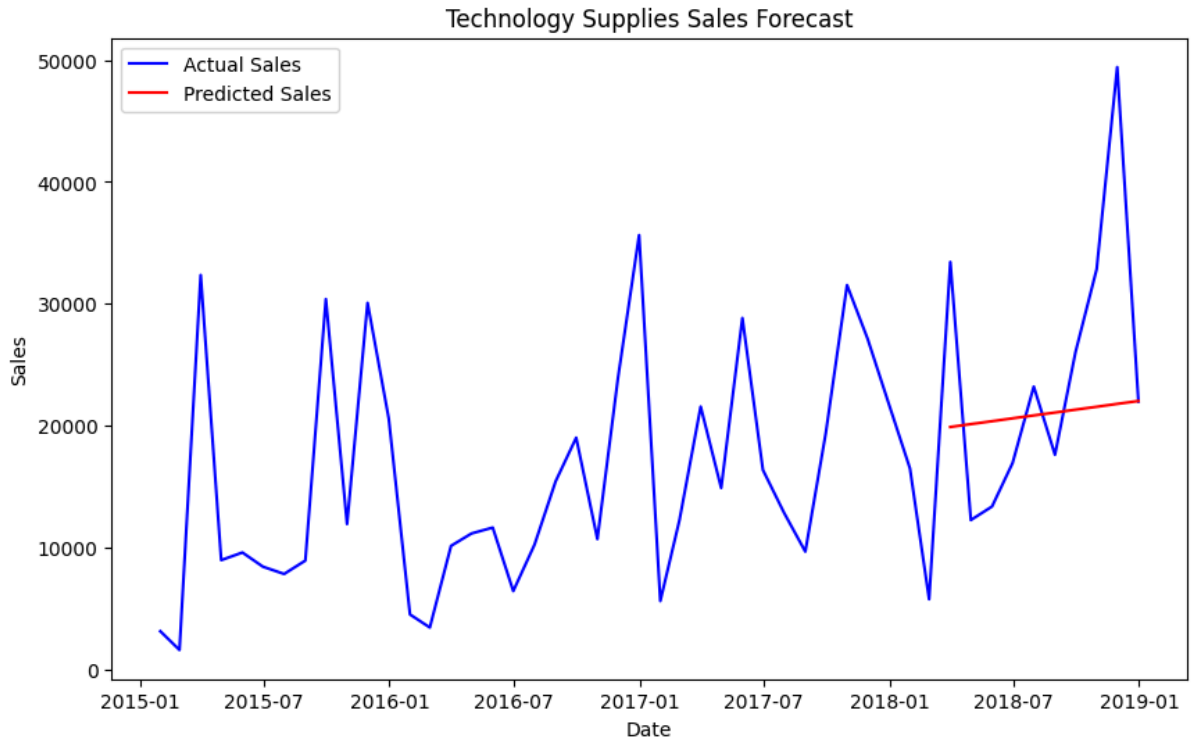


```
future_months = np.arange(len(office_supplies_sales),
len(office_supplies_sales) + 12).reshape(-1, 1)

future_sales_pred = model.predict(future_months)

future_dates = pd.date_range(office_supplies_sales['Order Date'].max(),
periods=12, freq='ME')

plt.plot(future_dates, future_sales_pred, label='Forecasted Sales',
color='green')

plt.title('Office Supplies Sales Forecast for 2019')

plt.xlabel('Date')

plt.ylabel('Sales')

plt.legend()

plt.show()
```

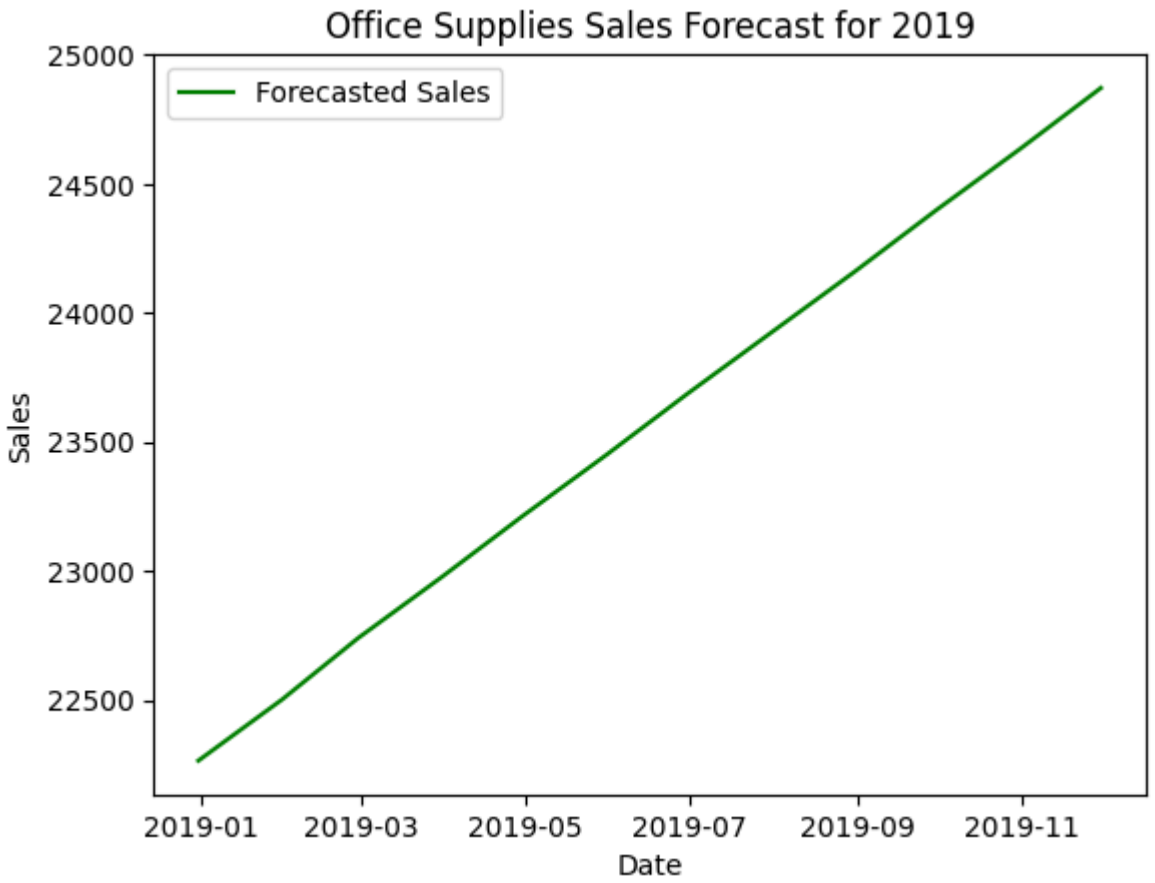



```
Technology_sales =
monthly_sales_pivot['Technology'].dropna().reset_index()
```

```
Technology_sales['Month_Num'] = np.arange(len(Technology_sales))


X = Technology_sales[['Month_Num']]

y = Technology_sales['Technology']


X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, shuffle=False)


model = LinearRegression()

model.fit(X_train, y_train)


y_pred = model.predict(X_test)


plt.figure(figsize=(10, 6))

plt.plot(Technology_sales['Order Date'], y, label='Actual Sales',
color='blue')

plt.plot(Technology_sales['Order Date'][len(X_train):], y_pred,
label='Predicted Sales', color='red')

plt.title('Technology Supplies Sales Forecast')

plt.xlabel('Date')

plt.ylabel('Sales')

plt.legend()

plt.show()
```

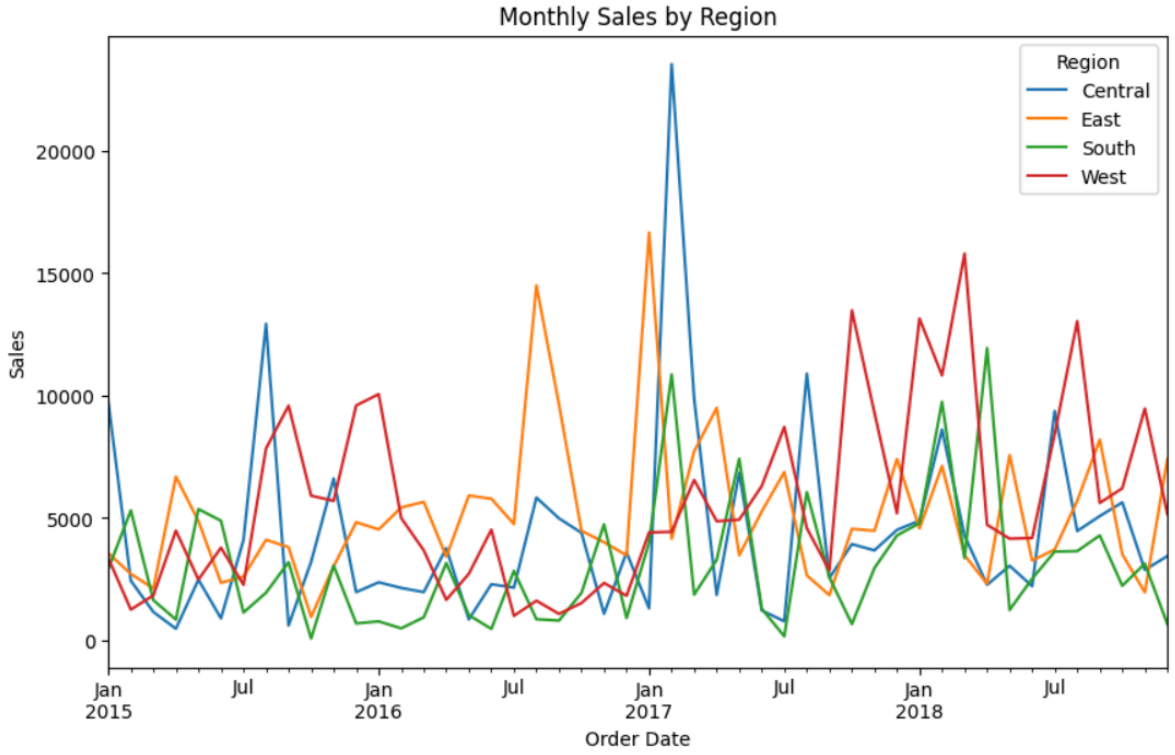



```
future_months = np.arange(len(Technology_sales), len(Technology_sales)
+ 12).reshape(-1, 1)

future_sales_pred = model.predict(future_months)

future_dates = pd.date_range(Technology_sales['Order Date'].max(),
periods=12, freq='ME')

plt.plot(future_dates, future_sales_pred, label='Forecasted Sales',
color='green')

plt.title('Office Supplies Sales Forecast for 2019')

plt.xlabel('Date')

plt.ylabel('Sales')

plt.legend()

plt.show()
```



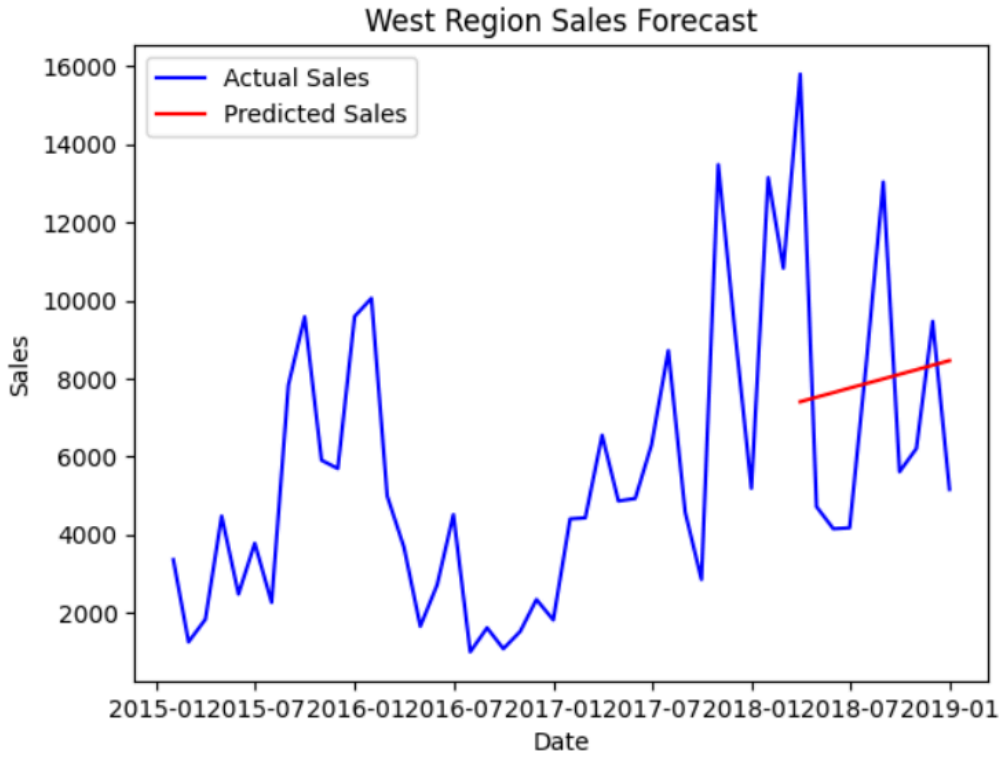
2. **Which regions are expected to see the most growth in sales?**
    - **Regional sales forecasting can help target high-potential areas and optimize resource distribution.**

```
monthly_sales_region = df.groupby([pd.Grouper(key='Order Date',
freq='M'), 'Region']).sum()['Sales'].reset_index()



monthly_sales_region_pivot = monthly_sales_region.pivot(index='Order
Date', columns='Region', values='Sales')



monthly_sales_region_pivot.plot(figsize=(10, 6), title='Monthly Sales
by Region')

plt.ylabel('Sales')

plt.show()
```

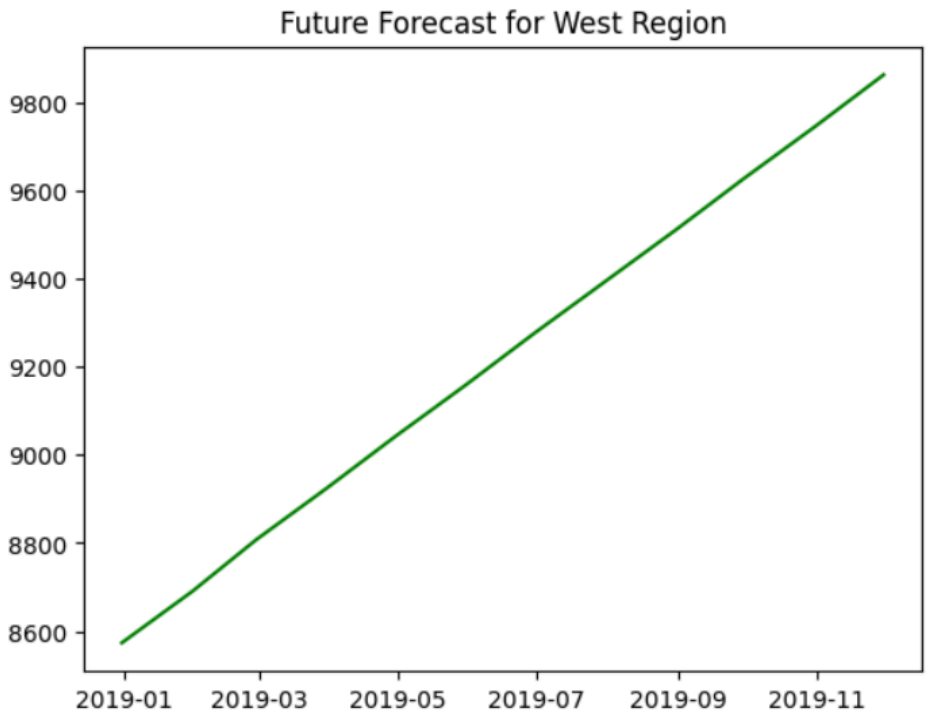



```
west_sales = monthly_sales_region_pivot['West'].dropna().reset_index()

west_sales['Month_Num'] = range(len(west_sales))

X = west_sales[['Month_Num']]

y = west_sales['West']



X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, shuffle=False)



model = LinearRegression()

model.fit(X_train, y_train)

y_pred = model.predict(X_test)

plt.plot(west_sales['Order Date'], y, label='Actual Sales',
color='blue')

plt.plot(west_sales['Order Date'][len(X_train):], y_pred,
label='Predicted Sales', color='red')
```

```
plt.title('West Region Sales Forecast')

plt.xlabel('Date')

plt.ylabel('Sales')

plt.legend()

plt.show()
```



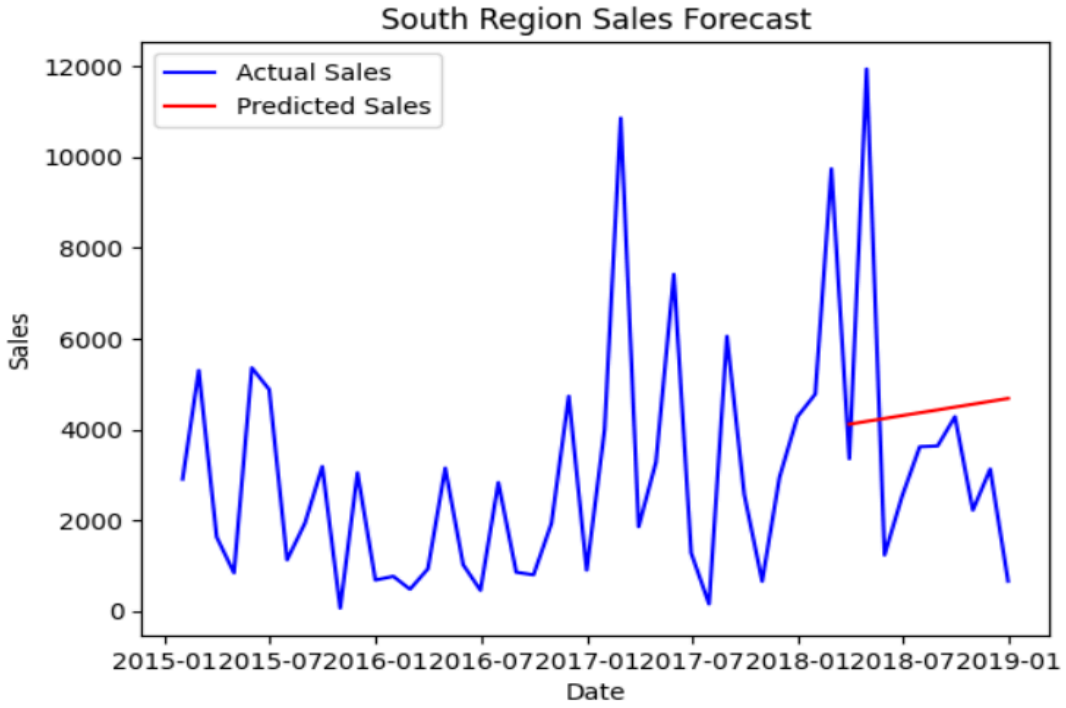

```
future_sales_pred = model.predict(future_months)

plt.plot(future_dates, future_sales_pred, label='Future Forecast for
West Region', color='green')

plt.title('Future Forecast for West Region')

plt.show()
```



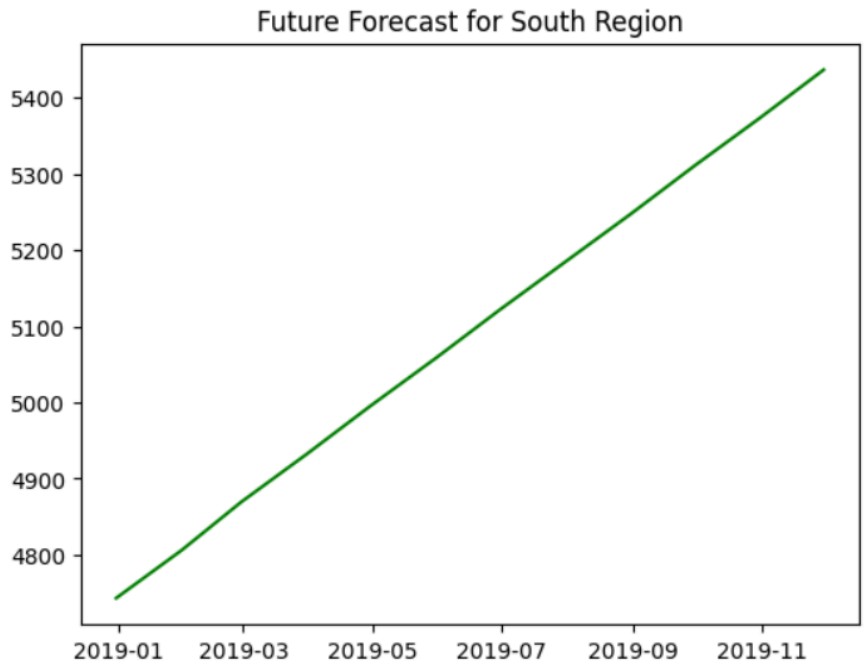

```
south_sales =
monthly_sales_region_pivot['South'].dropna().reset_index()

south_sales['Month_Num'] = range(len(south_sales))

X = south_sales[['Month_Num']]

y = south_sales['South']



X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, shuffle=False)



model = LinearRegression()

model.fit(X_train, y_train)
```

```
y_pred = model.predict(X_test)

plt.plot(south_sales['Order Date'], y, label='Actual Sales',
color='blue')
```

```
plt.plot(south_sales['Order Date'][len(X_train):], y_pred,
label='Predicted Sales', color='red')

plt.title('South Region Sales Forecast')

plt.xlabel('Date')

plt.ylabel('Sales')

plt.legend()

plt.show()
```



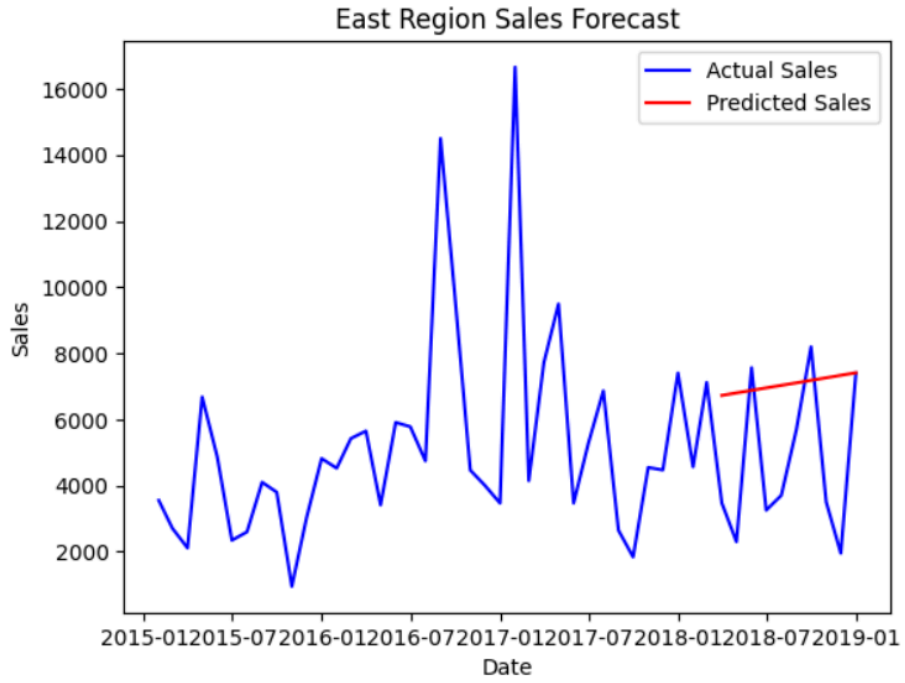

```
future_sales_pred = model.predict(future_months)

plt.plot(future_dates, future_sales_pred, label='Future Forecast for
West Region', color='green')

plt.title('Future Forecast for South Region')

plt.show()
```



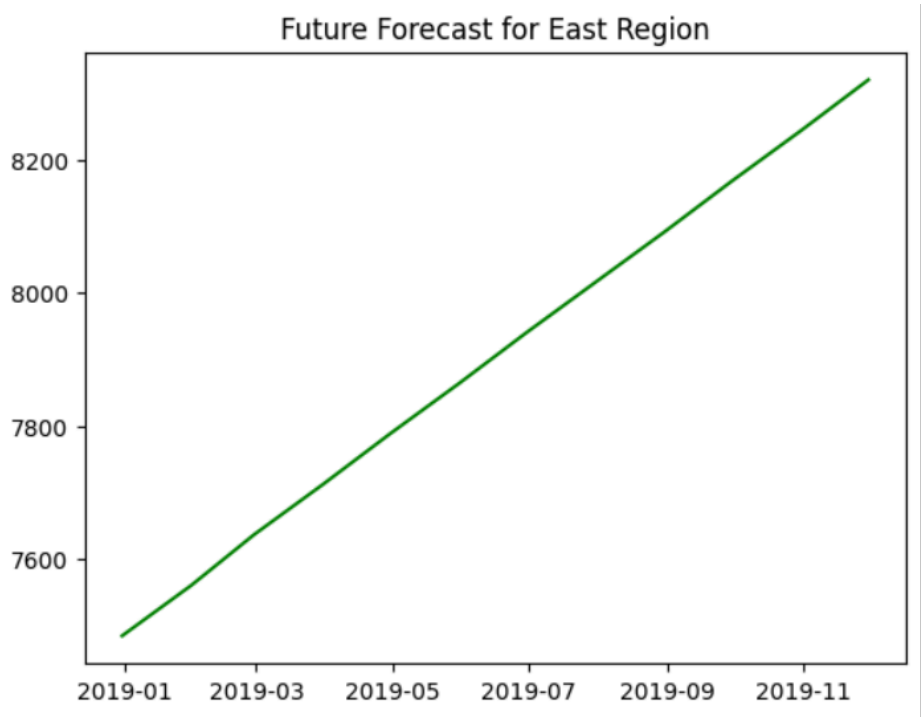

```
East_sales = monthly_sales_region_pivot['East'].dropna().reset_index()

East_sales['Month_Num'] = range(len(East_sales))

X = East_sales[['Month_Num']]

y = East_sales['East']



X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, shuffle=False)



model = LinearRegression()

model.fit(X_train, y_train)
```

```
y_pred = model.predict(X_test)

plt.plot(East_sales['Order Date'], y, label='Actual Sales',
color='blue')

plt.plot(East_sales['Order Date'][len(X_train):], y_pred,
label='Predicted Sales', color='red')

plt.title('East Region Sales Forecast')
```

```
plt.xlabel('Date')

plt.ylabel('Sales')

plt.legend()

plt.show()
```



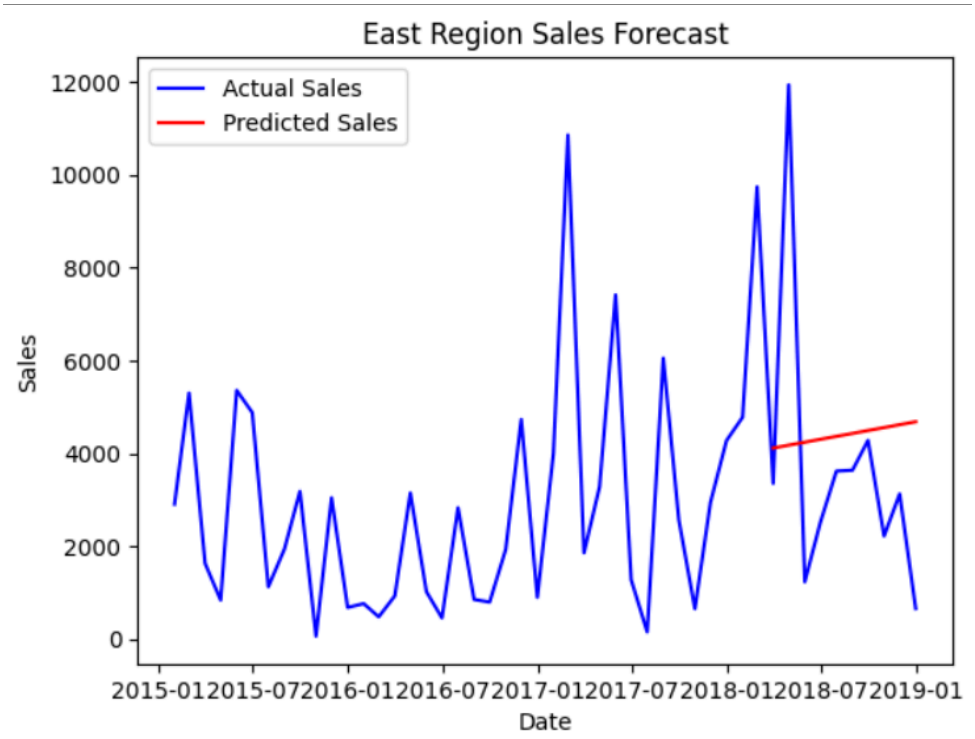

```
future_sales_pred = model.predict(future_months)

plt.plot(future_dates, future_sales_pred, label='Future Forecast for
East Region', color='green')

plt.title('Future Forecast for East Region')

plt.show()
```



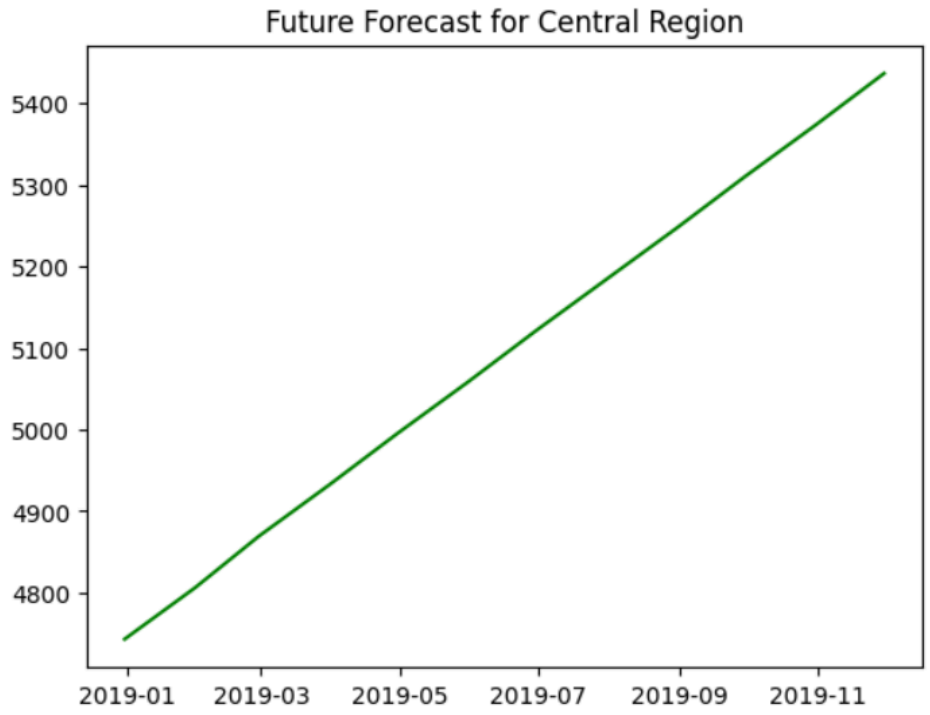

```
central_sales =
monthly_sales_region_pivot['South'].dropna().reset_index()

central_sales['Month_Num'] = range(len(central_sales))

X = central_sales[['Month_Num']]

y = central_sales['South']



X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, shuffle=False)



model = LinearRegression()

model.fit(X_train, y_train)
```

```
y_pred = model.predict(X_test)

plt.plot(central_sales['Order Date'], y, label='Actual Sales',
color='blue')
```

```
plt.plot(central_sales['Order Date'][len(X_train):], y_pred,
label='Predicted Sales', color='red')

plt.title('East Region Sales Forecast')

plt.xlabel('Date')

plt.ylabel('Sales')

plt.legend()

plt.show()
```



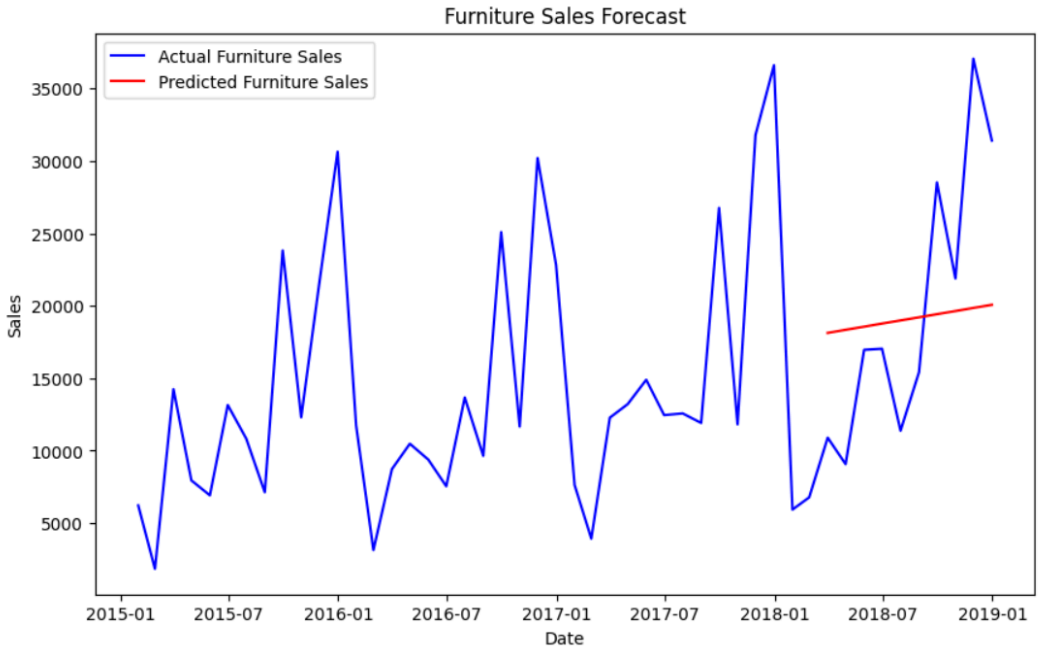

```
future_sales_pred = model.predict(future_months)

plt.plot(future_dates, future_sales_pred, label='Future Forecast for
Central Region', color='green')

plt.title('Future Forecast for Central Region')

plt.show()
```

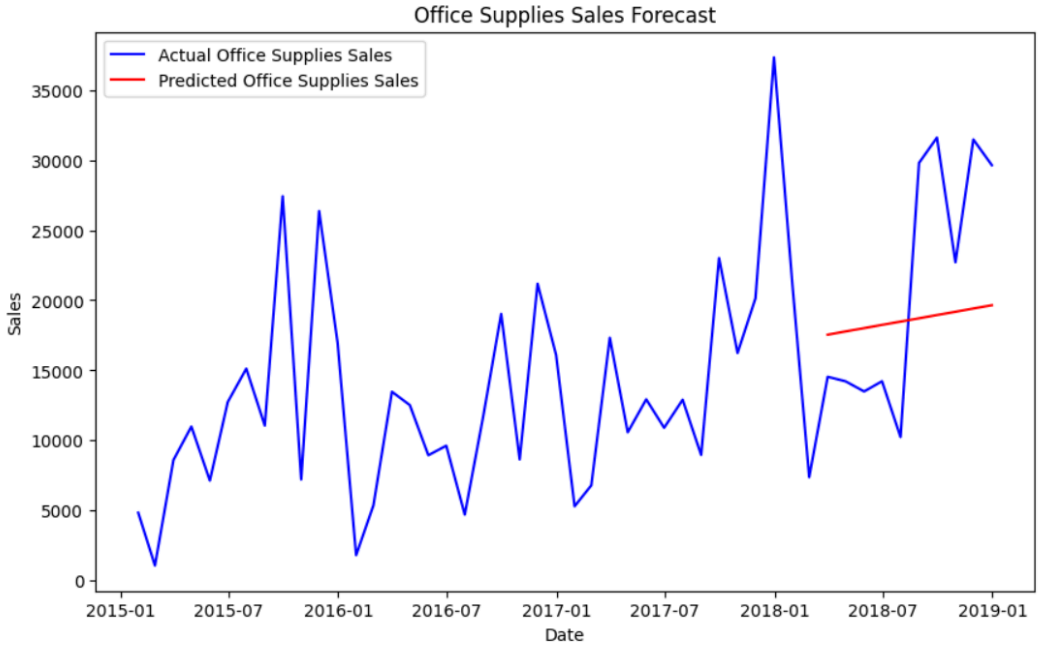


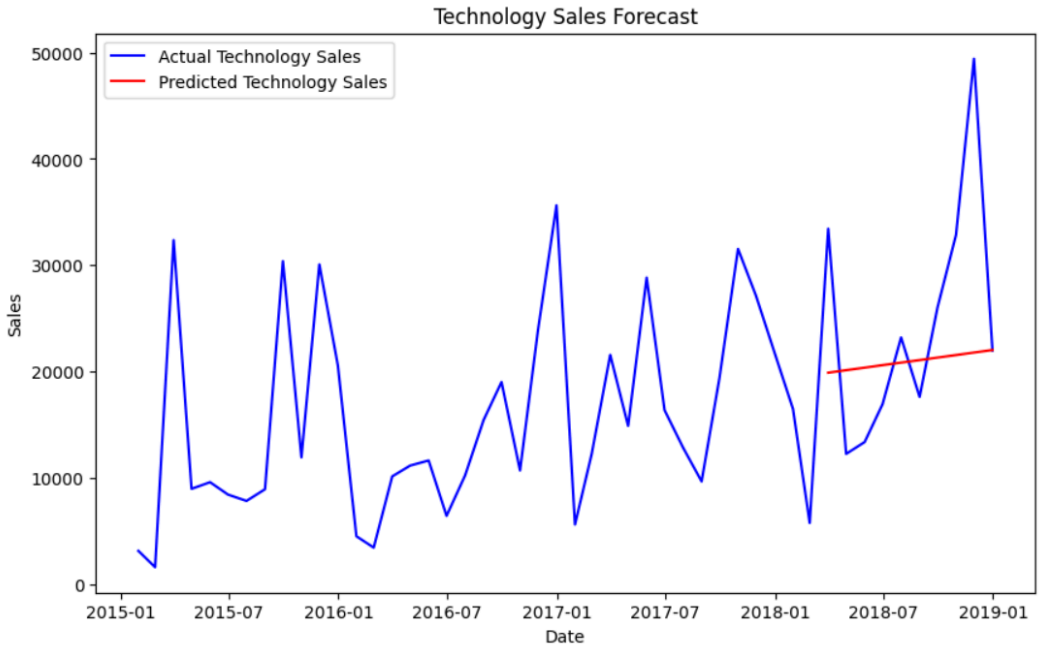
3. **How will the demand for different product categories evolve over time?**
    - **Predicting demand changes by category will help with supply chain and inventory management.**

```
for category in monthly_sales_pivot.columns:

    category_sales =
monthly_sales_pivot[category].dropna().reset_index()

    category_sales['Month_Num'] = range(len(category_sales))



    X = category_sales[['Month_Num']]

    y = category_sales[category]



    X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, shuffle=False)



    model = LinearRegression()
```

```
    model.fit(X_train, y_train)



    y_pred = model.predict(X_test)



    plt.figure(figsize=(10, 6))

    plt.plot(category_sales['Order Date'], y, label=f'Actual {category}
Sales', color='blue')

    plt.plot(category_sales['Order Date'][len(X_train):], y_pred,
label=f'Predicted {category} Sales', color='red')

    plt.title(f'{category} Sales Forecast')

    plt.xlabel('Date')

    plt.ylabel('Sales')

    plt.legend()

    plt.show()
```

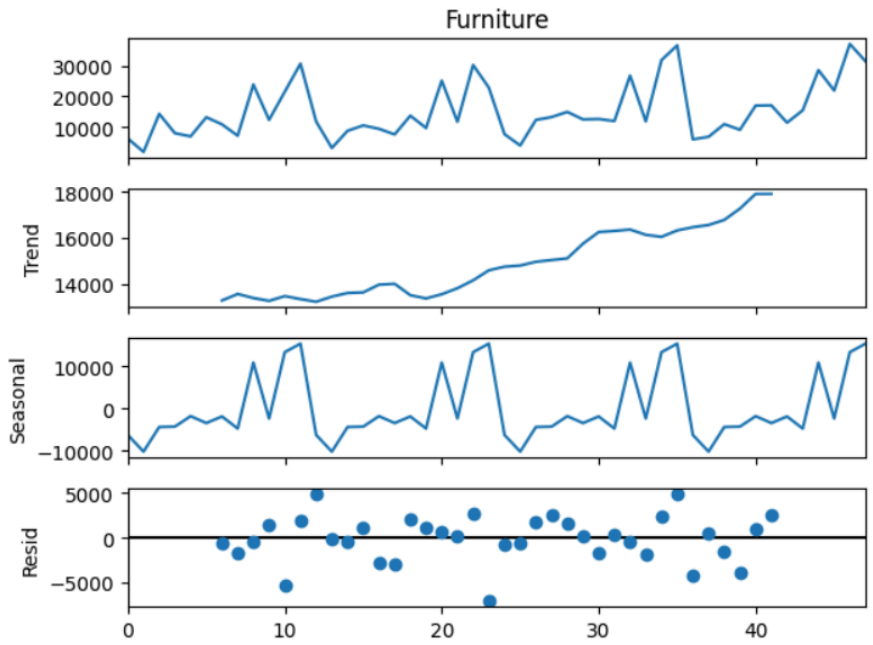

Furniture Sales Forecast
Actual Furniture Sales
Predicted Furniture Sales
Sales
Date
35000
30000
25000
20000
15000
10000
5000
2015-01
2015-07
2016-01
2016-07
2017-01
2017-07
2018-01
2018-07
2019-01


Office Supplies Sales Forecast
Actual Office Supplies Sales
Predicted Office Supplies Sales
Sales
Date
35000
30000
25000
20000
15000
10000
5000
0
2015-01
2015-07
2016-01
2016-07
2017-01
2017-07
2018-01
2018-07
2019-01

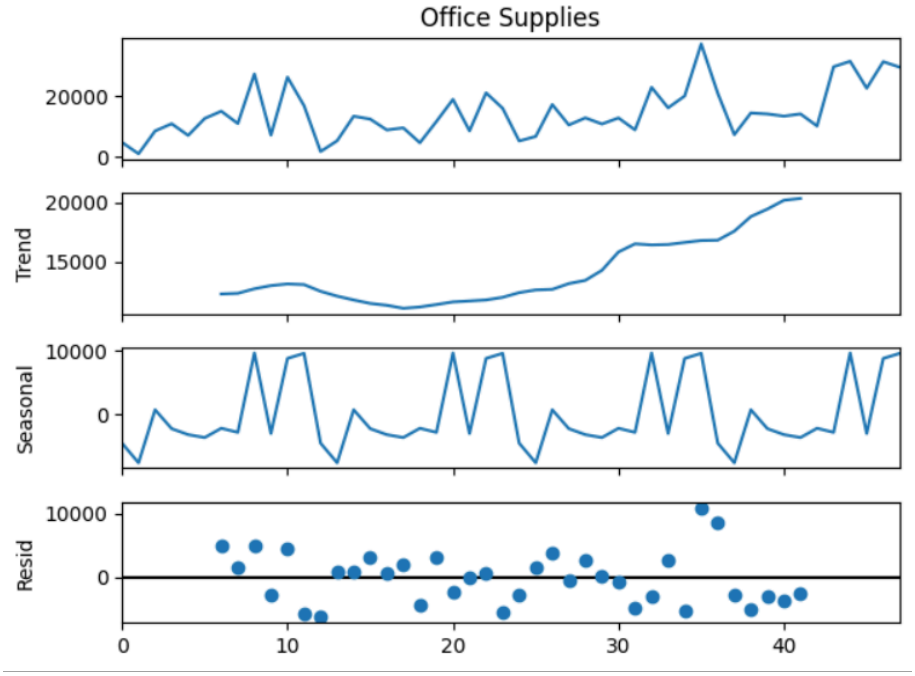


4. **What are the expected peak sales periods in the upcoming months?**
    - **Knowing peak sales periods can assist in inventory planning and promotional strategies.**

```
from statsmodels.tsa.seasonal import seasonal_decompose

decomposition = seasonal_decompose(furniture_sales['Furniture'],
model='additive', period=12)

decomposition.plot()

plt.show()
```

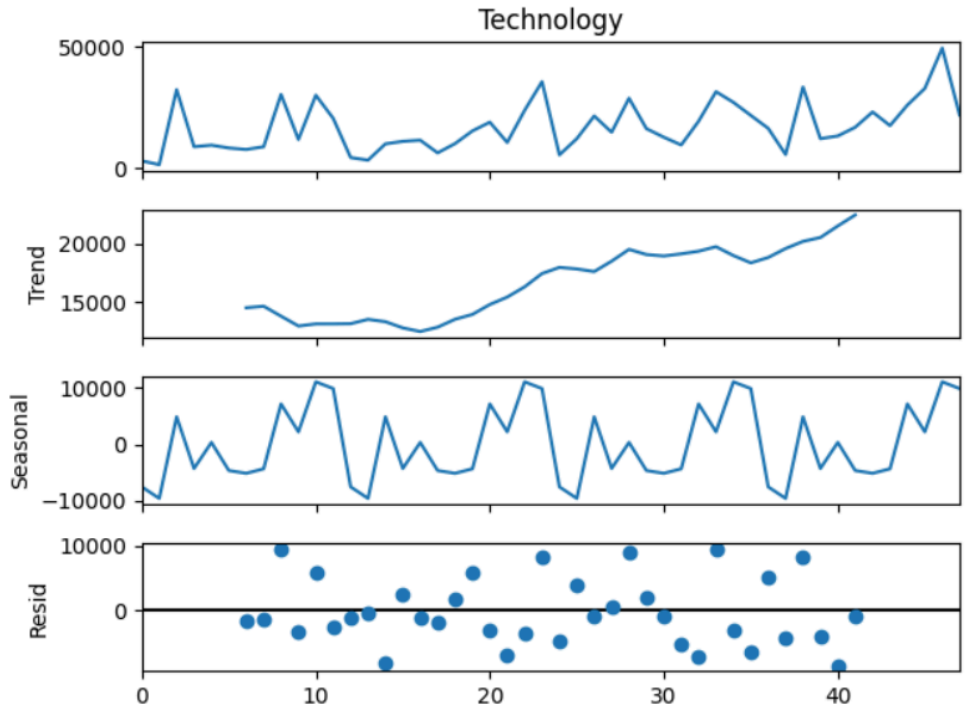



```
decomposition = seasonal_decompose(office_supplies_sales['Office
Supplies'], model='additive', period=12)

decomposition.plot()

plt.show()
```



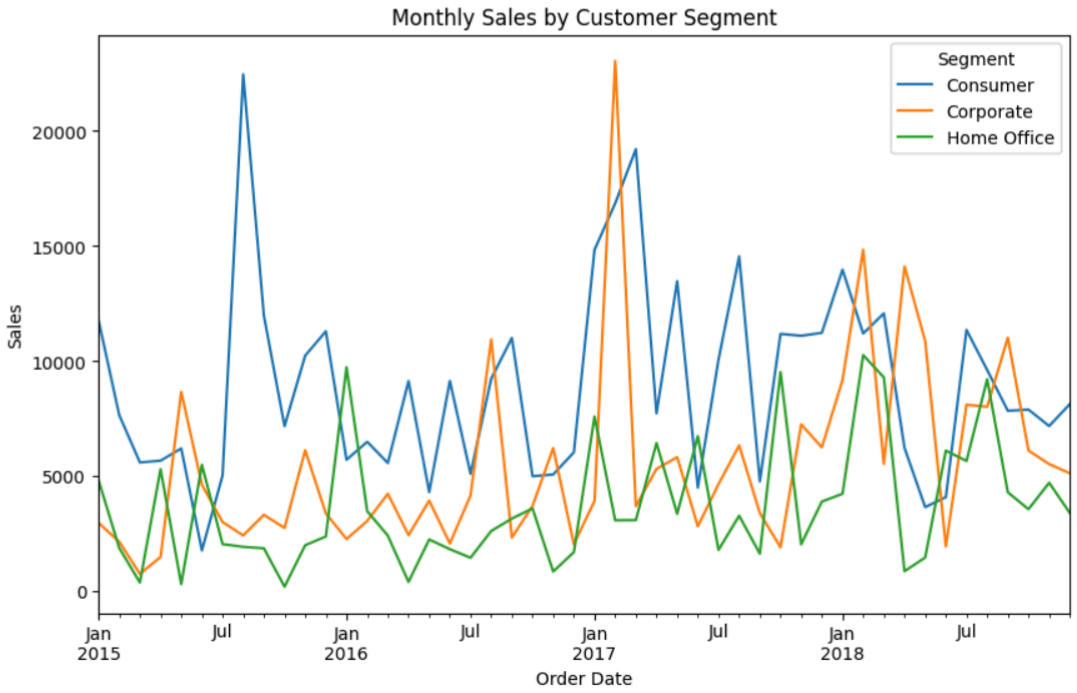

```
decomposition = seasonal_decompose(Technology_sales['Technology'],
model='additive', period=12)

decomposition.plot()
```

```
plt.show()
```



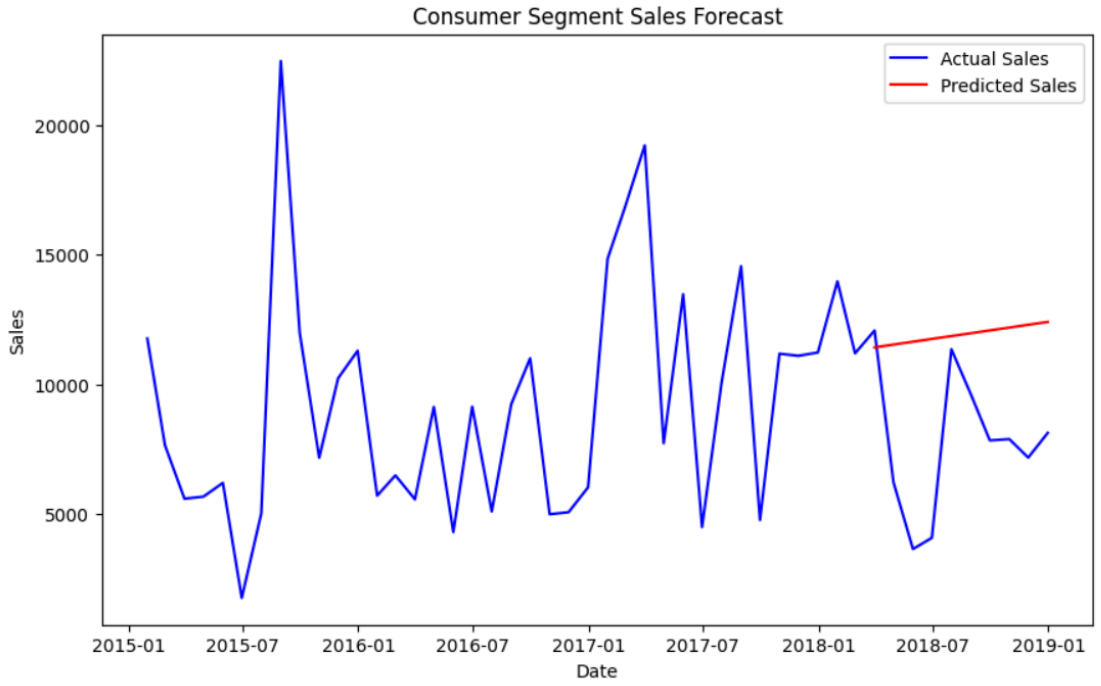
5. **What will be the expected revenue for key customer segments (e.g., Consumer, Corporate, Home Office)?**
    - **Segment-based forecasting helps customize marketing campaigns for specific customer groups.**

```
monthly_sales_segment = df.groupby([pd.Grouper(key='Order Date',
freq='M'), 'Segment']).sum()['Sales'].reset_index()

monthly_sales_segment_pivot = monthly_sales_segment.pivot(index='Order
Date', columns='Segment', values='Sales')

monthly_sales_segment_pivot.plot(figsize=(10, 6), title='Monthly Sales
by Customer Segment')

plt.ylabel('Sales')

plt.show()
```



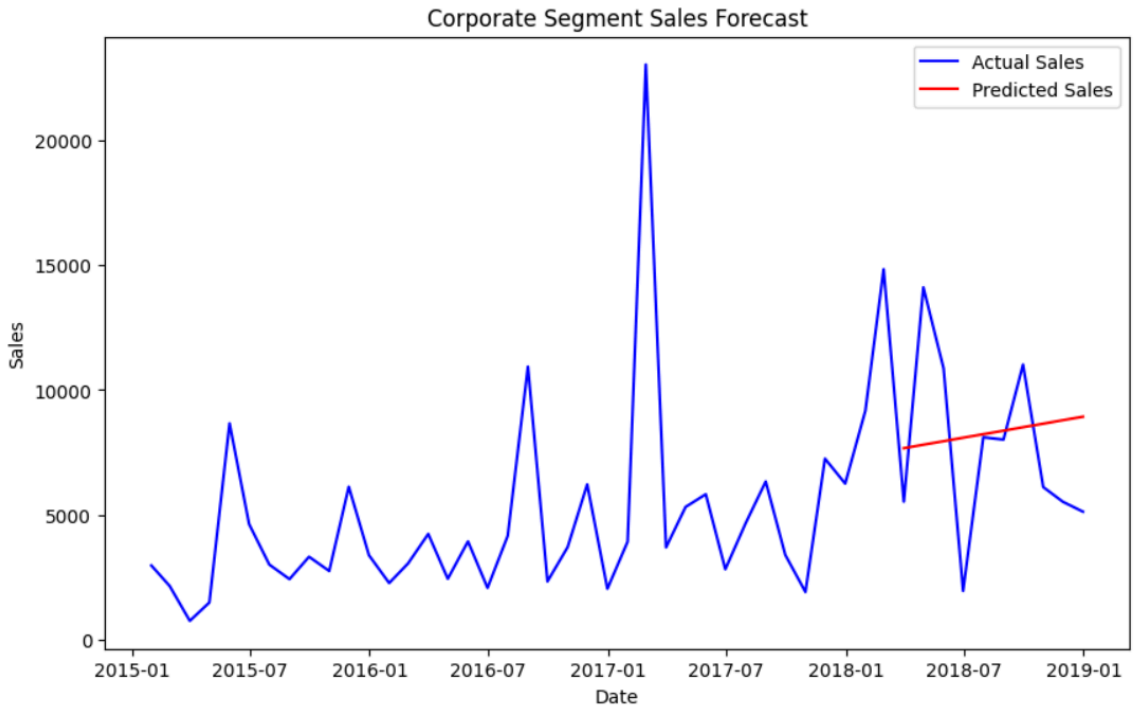

```
consumer_sales =
monthly_sales_segment_pivot['Consumer'].dropna().reset_index()

consumer_sales['Month_Num'] = np.arange(len(consumer_sales))

X = consumer_sales[['Month_Num']]

y = consumer_sales['Consumer']

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, shuffle=False)

model = LinearRegression()

model.fit(X_train, y_train)

y_pred = model.predict(X_test)



plt.figure(figsize=(10, 6))

plt.plot(consumer_sales['Order Date'], y, label='Actual Sales',
color='blue')

plt.plot(consumer_sales['Order Date'][len(X_train):], y_pred,
label='Predicted Sales', color='red')

plt.title('Consumer Segment Sales Forecast')
```

```
plt.xlabel('Date')

plt.ylabel('Sales')

plt.legend()

plt.show()
```



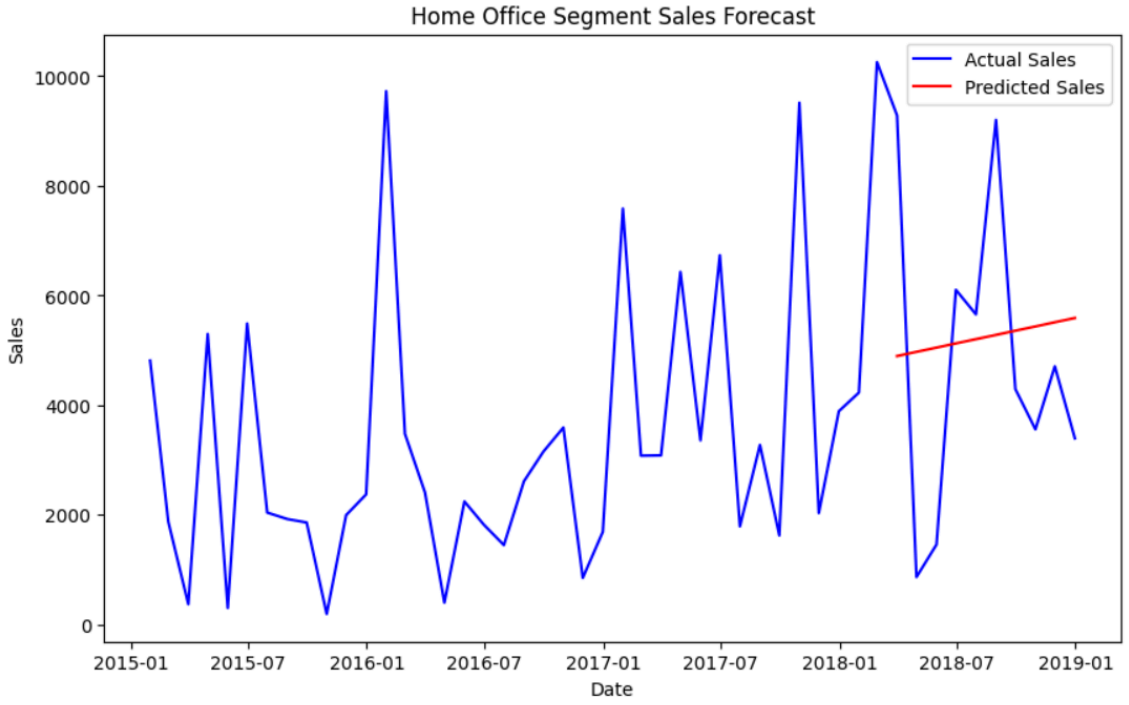

```
corporate_sales =
monthly_sales_segment_pivot['Corporate'].dropna().reset_index()

corporate_sales['Month_Num'] = np.arange(len(corporate_sales))

X = corporate_sales[['Month_Num']]

y = corporate_sales['Corporate']

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, shuffle=False)

model = LinearRegression()

model.fit(X_train, y_train)

y_pred = model.predict(X_test)
```

```
plt.figure(figsize=(10, 6))

plt.plot(corporate_sales['Order Date'], y, label='Actual Sales',
color='blue')

plt.plot(corporate_sales['Order Date'][len(X_train):], y_pred,
label='Predicted Sales', color='red')

plt.title('Corporate Segment Sales Forecast')

plt.xlabel('Date')

plt.ylabel('Sales')

plt.legend()

plt.show()
```




```
home_office_sales = monthly_sales_segment_pivot['Home
Office'].dropna().reset_index()

home_office_sales['Month_Num'] = np.arange(len(home_office_sales))

X = home_office_sales[['Month_Num']]

y = home_office_sales['Home Office']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, shuffle=False)

model = LinearRegression()

model.fit(X_train, y_train)

y_pred = model.predict(X_test)



plt.figure(figsize=(10, 6))

plt.plot(home_office_sales['Order Date'], y, label='Actual Sales',
color='blue')

plt.plot(home_office_sales['Order Date'][len(X_train):], y_pred,
label='Predicted Sales', color='red')

plt.title('Home Office Segment Sales Forecast')

plt.xlabel('Date')

plt.ylabel('Sales')

plt.legend()

plt.show()
```



Task 4

Week 4: Visualization Dashboard and Final Presentation

- **Tasks:**

  ➔ Build a Visualization Dashboard: Build a Tableau visualization dashboard that visualizes the answers to all answered questions.

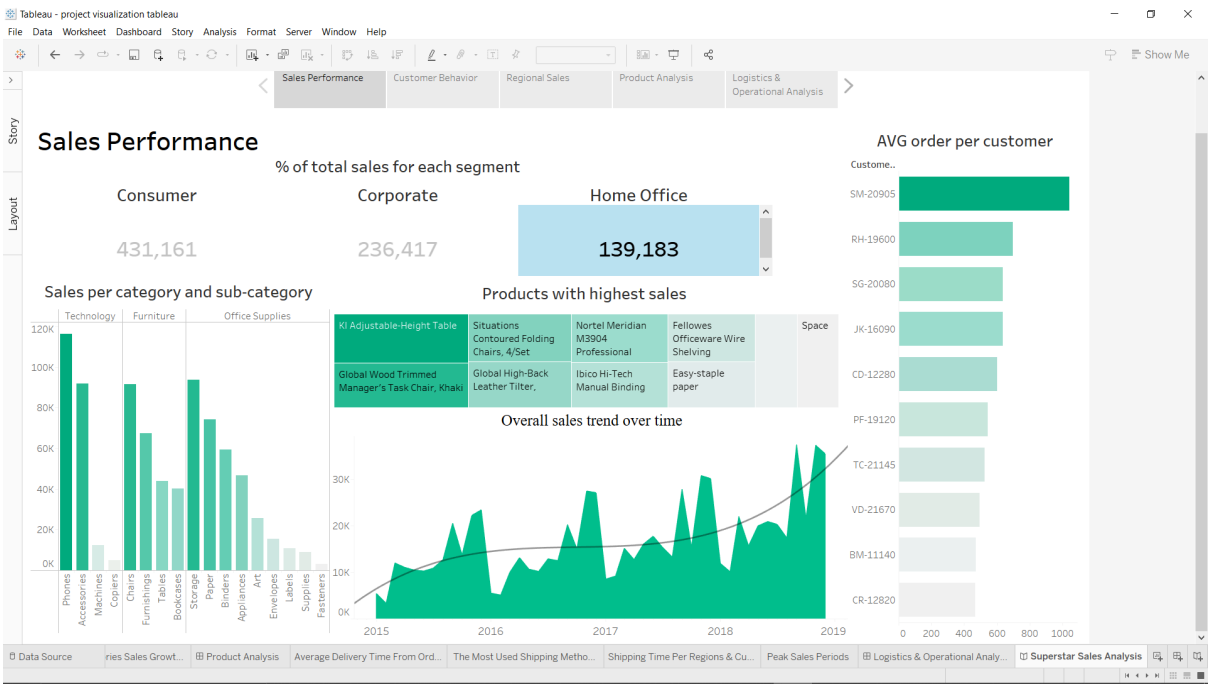
  ➔ Final Presentation: Prepare a report and presentation summarizing the project work, including data analysis, model development, and deployment.

  ➔ Tools: SQL, Python (pandas, Matplotlib), Tableau.

- **Deliverables:**

  ➔ Visualization dashboard.

Final report and presentation.



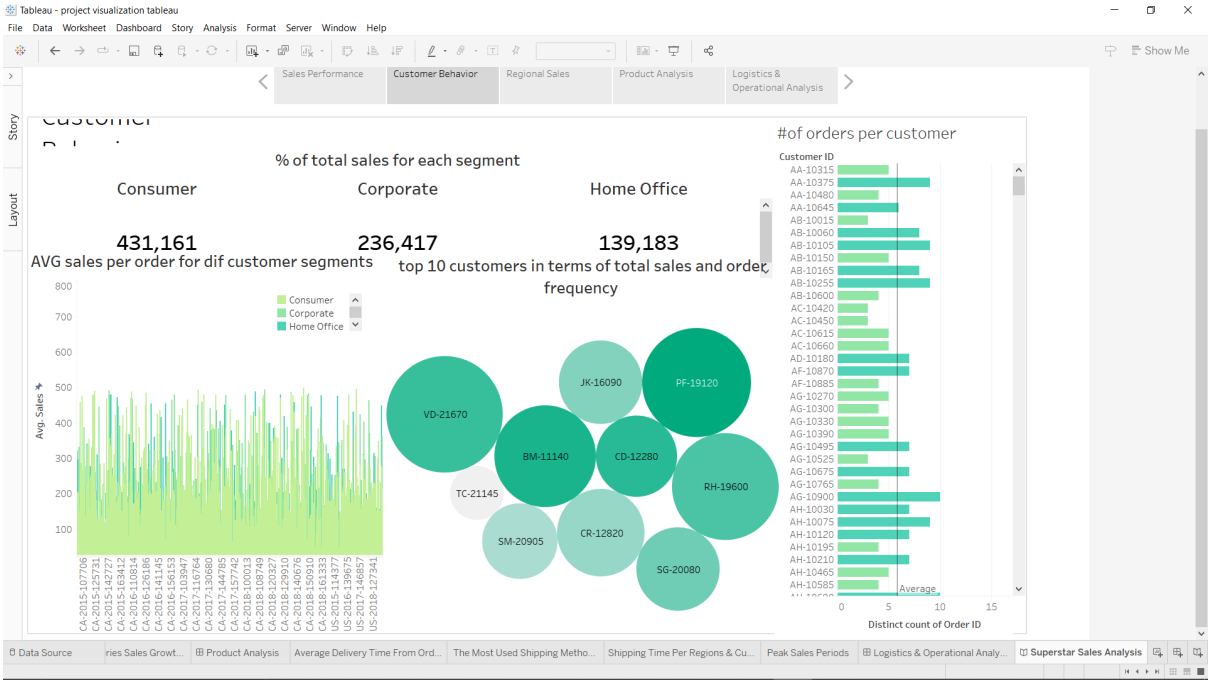


Tableau - project visualization tableau
File Data Worksheet Dashboard Story Analysis Format Server Window Help
Sales Performance
Customer Behavior
Regional Sales
Product Analysis
Logistics & Operational Analysis
% of total sales for each segment
Consumer
Corporate
Home Office
431,161
236,417
139,183
#of orders per customer
Customer ID
AVG sales per order for dif customer segments
Consumer
Corporate
Home Office
Avg. Sales
top 10 customers in terms of total sales and order frequency
VD-21670
JK-16090
PF-19120
BM-11140
CD-12280
RH-19600
TC-21145
CR-12820
SM-20905
SG-20080
Average
Distinct count of Order ID
Data Source
Product Analysis
Average Delivery Time From Ord...
The Most Used Shipping Metho...
Shipping Time Per Regions & Cu...
Peak Sales Periods
Logistics & Operational Analy...
Superstar Sales Analysis

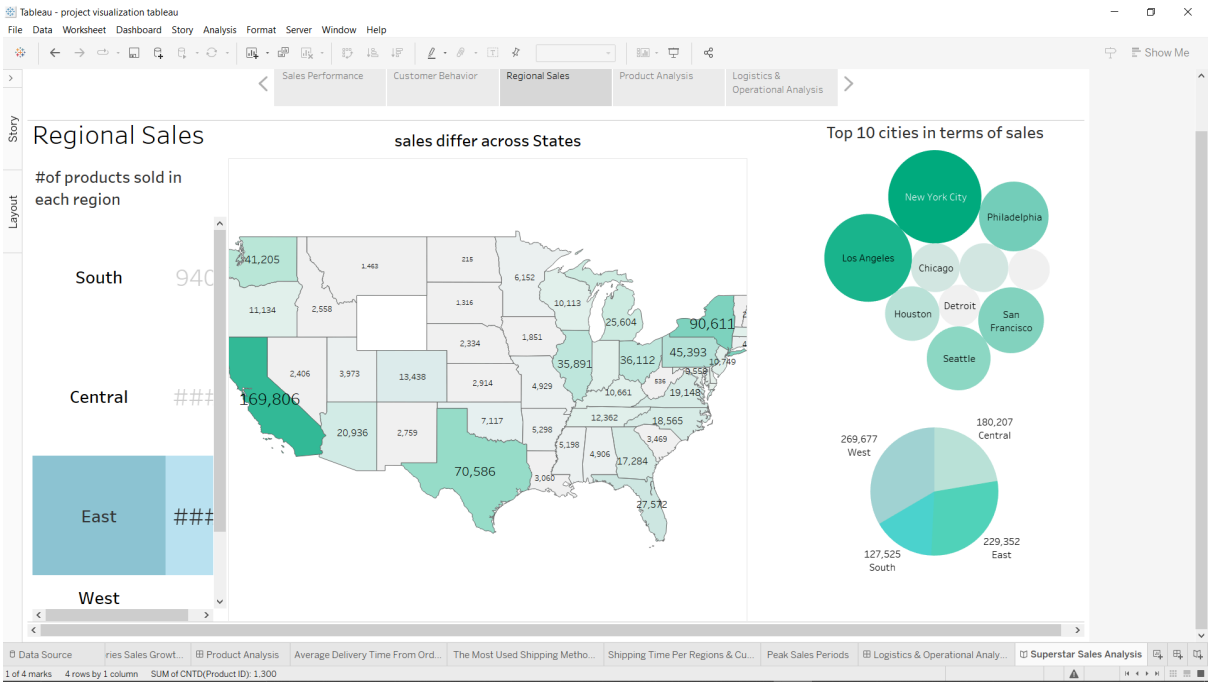


Tableau - project visualization tableau
File Data Worksheet Dashboard Story Analysis Format Server Window Help
Sales Performance
Customer Behavior
Regional Sales
Product Analysis
Logistics & Operational Analysis
Regional Sales
#of products sold in each region
South
Central
East
West
sales differ across States
Top 10 cities in terms of sales
New York City
Philadelphia
Los Angeles
Chicago
Houston
Detroit
San Francisco
Seattle
180,207
Central
269,677
West
229,352
East
127,525
South
Data Source
Product Analysis
Average Delivery Time From Ord...
The Most Used Shipping Metho...
Shipping Time Per Regions & Cu...
Peak Sales Periods
Logistics & Operational Analy...
Superstar Sales Analysis
1 of 4 marks
4 rows by 1 column
SUM of CNTD(Product ID): 1,300

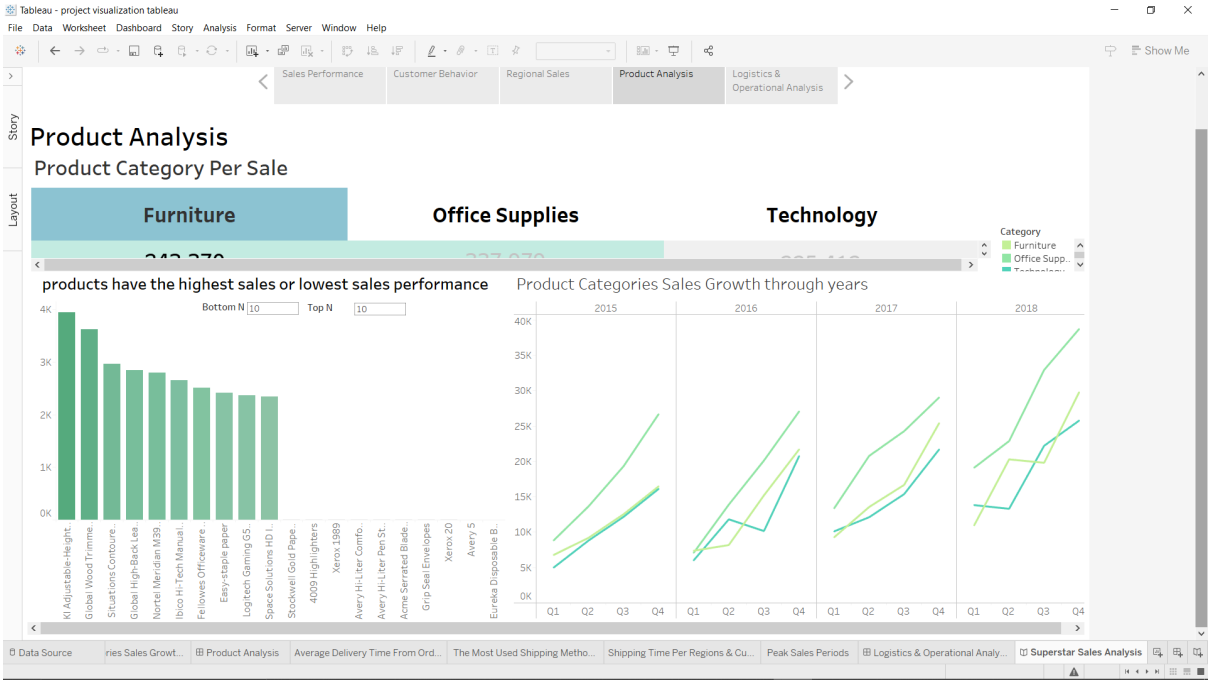


Tableau - project visualization tableau
File Data Worksheet Dashboard Story Analysis Format Server Window Help
Sales Performance
Customer Behavior
Regional Sales
Product Analysis
Logistics & Operational Analysis
Story
Layout
Product Analysis
Product Category Per Sale
Furniture
Office Supplies
Technology
Category
Furniture
Office Supp..
products have the highest sales or lowest sales performance
Bottom N 10
Top N 10
Product Categories Sales Growth through years
2015
2016
2017
2018
Q1 Q2 Q3 Q4
Data Source
Product Analysis
Average Delivery Time From Ord...
The Most Used Shipping Metho...
Shipping Time Per Regions & Cu...
Peak Sales Periods
Logistics & Operational Analy...
Superstar Sales Analysis

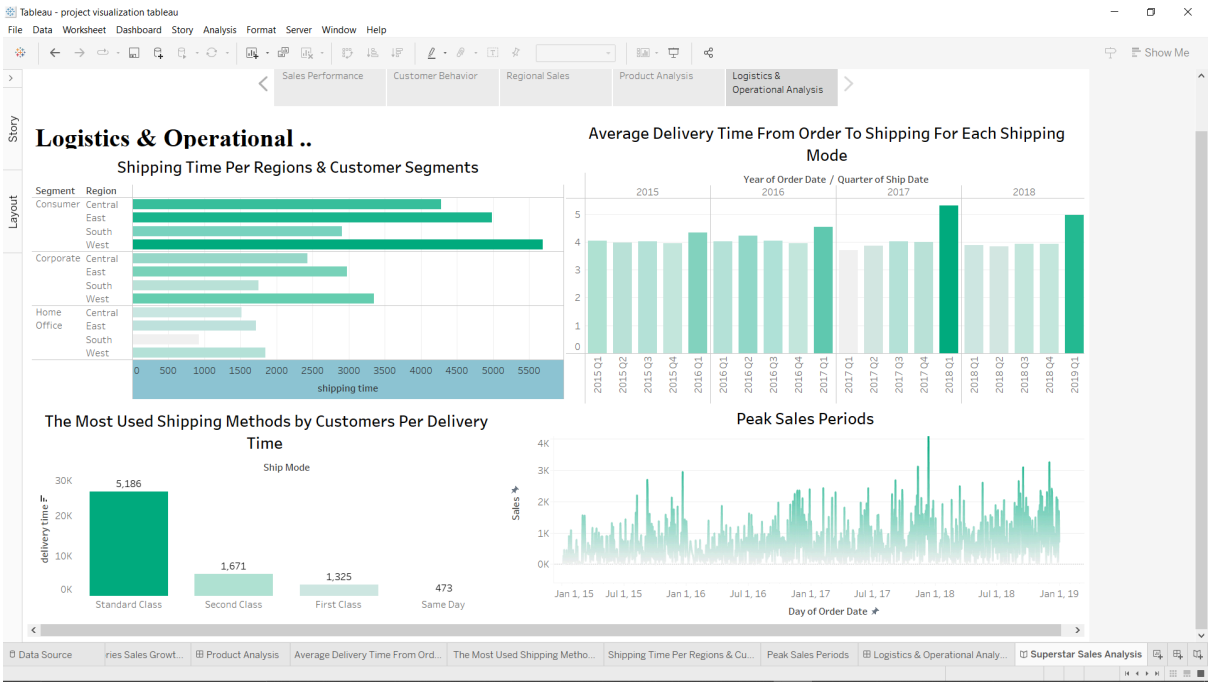


Tableau - project visualization tableau
File Data Worksheet Dashboard Story Analysis Format Server Window Help
Sales Performance
Customer Behavior
Regional Sales
Product Analysis
Logistics & Operational Analysis
Story
Layout
Logistics & Operational ..
Shipping Time Per Regions & Customer Segments
Segment Region
Consumer Central East South West
Corporate Central East South West
Home Office Central East South West
shipping time
Average Delivery Time From Order To Shipping For Each Shipping Mode
Year of Order Date / Quarter of Ship Date
2015 2016 2017 2018
The Most Used Shipping Methods by Customers Per Delivery Time
Ship Mode
delivery time
5,186
1,671
1,325
473
Standard Class
Second Class
First Class
Same Day
Peak Sales Periods
Sales
Day of Order Date
Data Source
Product Analysis
Average Delivery Time From Ord...
The Most Used Shipping Metho...
Shipping Time Per Regions & Cu...
Peak Sales Periods
Logistics & Operational Analy...
Superstar Sales Analysis