FINAL REPORT

Superstore sales project

Data analyst specialist track

ONL1_ DAT1_ Md1

Group 1

رواد مصر الرقمية

DEPI | 2024

Abstract

This is a detailed report about the steps of analysis of the superstore dataset analysis using SQL, Python & visualization using tableau

2024

Group A

Group members:

1- Nehal Abdel Hady Abdelhamid (group leader)
2- Shahd Ashraf Ramadan (group presenter)
3- Eman Saad Mohamed Abbas
4- Amira Fawaz Zaki
5- Ibrahim Alaa Ibrahim
6- Al – Amir Ayman AL- Amir

Introduction

The dataset of superstore sales consists of 3 tables

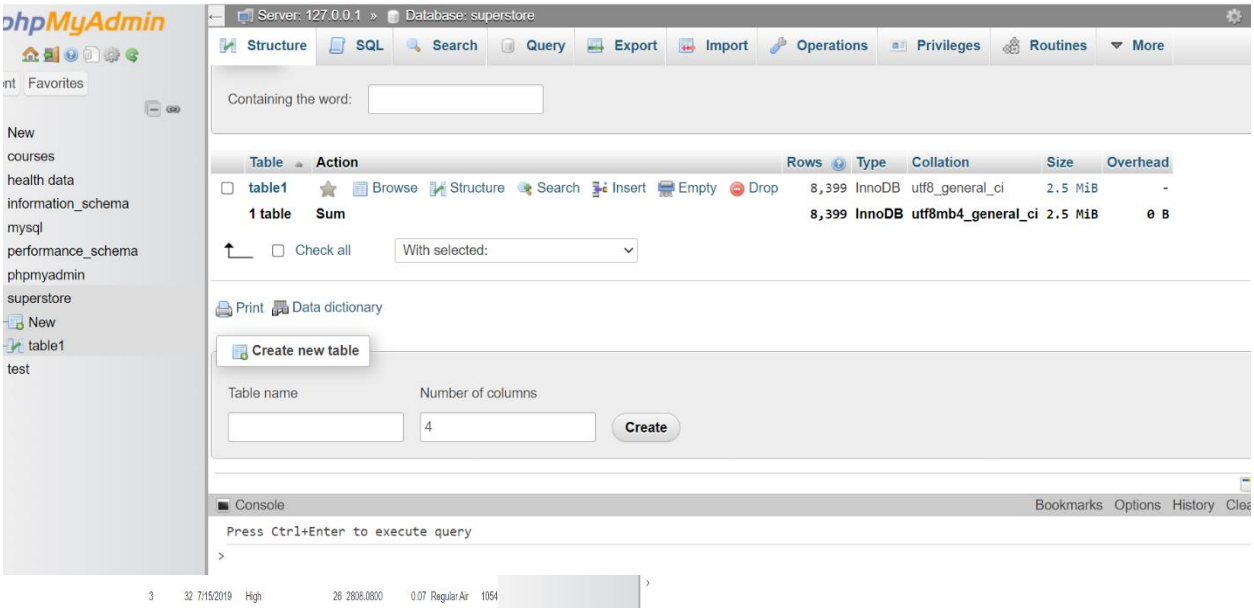
1- Tables of orders (like order id, date, customer, items, category, subcategory, profit, sales…...extra).
2- The return table (order id & status)
3- The users (area & manager)

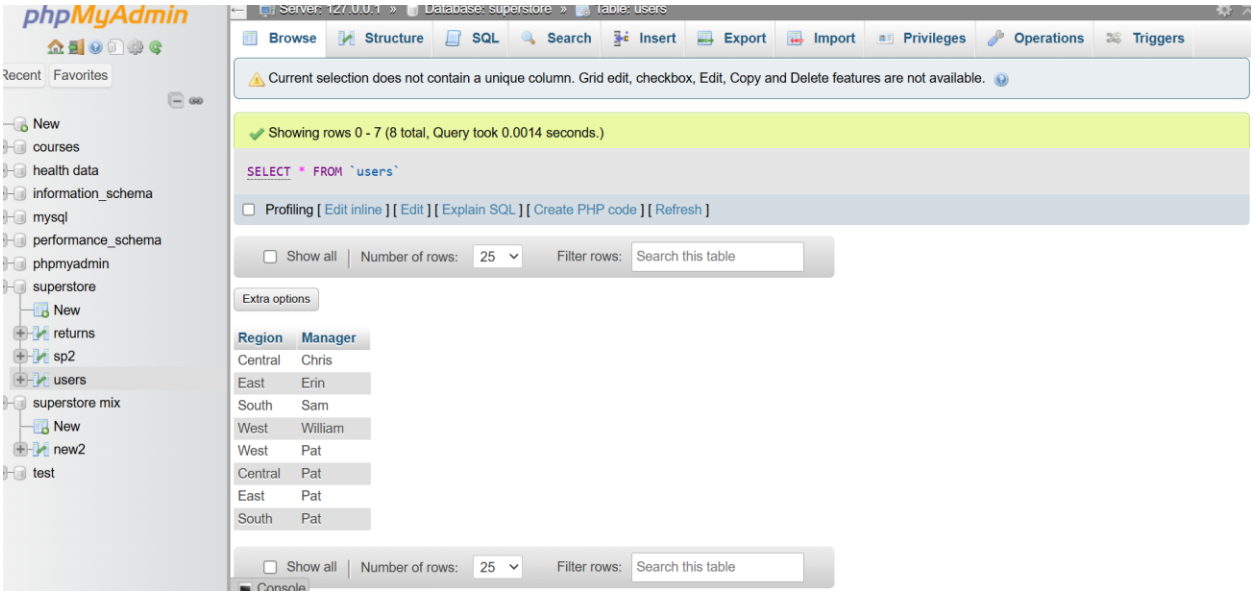
We have to merge the 3 tables together using different tools like SQL, Python & Tableau
to extract insightful pattern, here are the steps in details

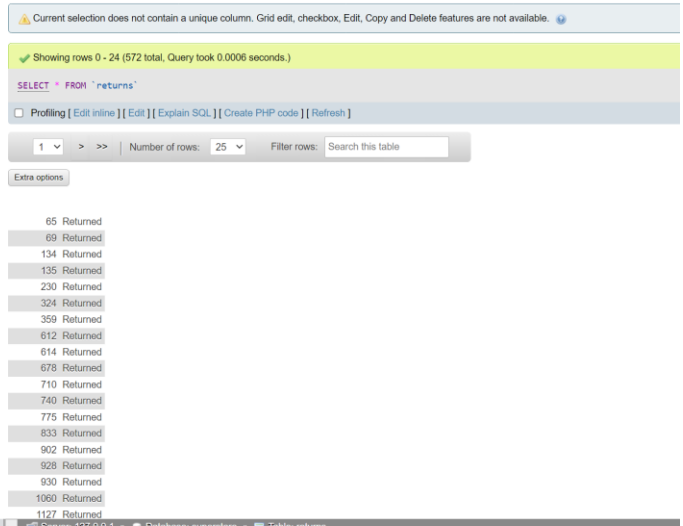
First: SQL part

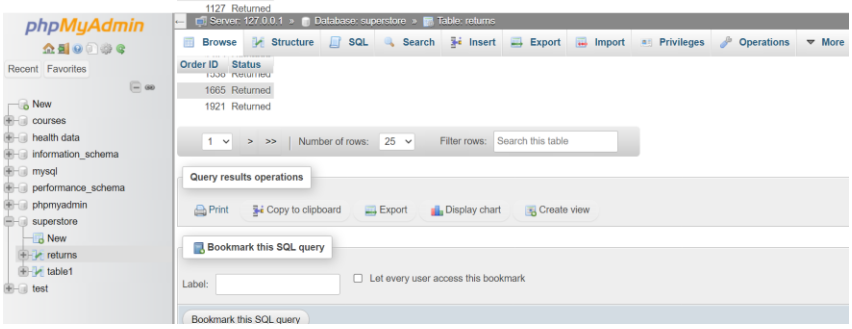
Open Xamp, phpMyAdmin

we import data, we import 8399 rows.




phpMyAdmin
Recent
Favorites
New
courses
health data
information_schema
mysql
performance_schema
phpmyadmin
superstore
New
returns
sp2
users
superstore mix
New
new2
test
Server: 127.0.0.1 » Database: superstore » Table: users
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operations
Triggers
Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.
Showing rows 0 - 7 (8 total, Query took 0.0014 seconds.)
SELECT * FROM `users`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all
Number of rows: 25
Filter rows: Search this table
Extra options
Region Manager
Central Chris
East Erin
South Sam
West William
West Pat
Central Pat
East Pat
South Pat
Show all
Number of rows: 25
Filter rows: Search this table
Console


Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.
Showing rows 0 - 24 (572 total, Query took 0.0006 seconds.)
SELECT * FROM `returns`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
1
> >>
Number of rows: 25
Filter rows: Search this table
Extra options
65 Returned
69 Returned
134 Returned
135 Returned
230 Returned
324 Returned
359 Returned
612 Returned
614 Returned
678 Returned
710 Returned
740 Returned
775 Returned
833 Returned
902 Returned
928 Returned
930 Returned
1060 Returned
1127 Returned


phpMyAdmin
Recent
Favorites
New
courses
health data
information_schema
mysql
performance_schema
phpmyadmin
superstore
New
returns
table1
test
Server: 127.0.0.1 » Database: superstore » Table: returns
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operations
More
Order ID Status
1665 Returned
1921 Returned
1
> >>
Number of rows: 25
Filter rows: Search this table
Query results operations
Print
Copy to clipboard
Export
Display chart
Create view
Bookmark this SQL query
Label:
Let every user access this bookmark
Bookmark this SQL query

step 2: we have to make left join for 3 tables in one table and export it as single data base

Show query box

Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 24 (25 total, Query took 0.2169 seconds.)

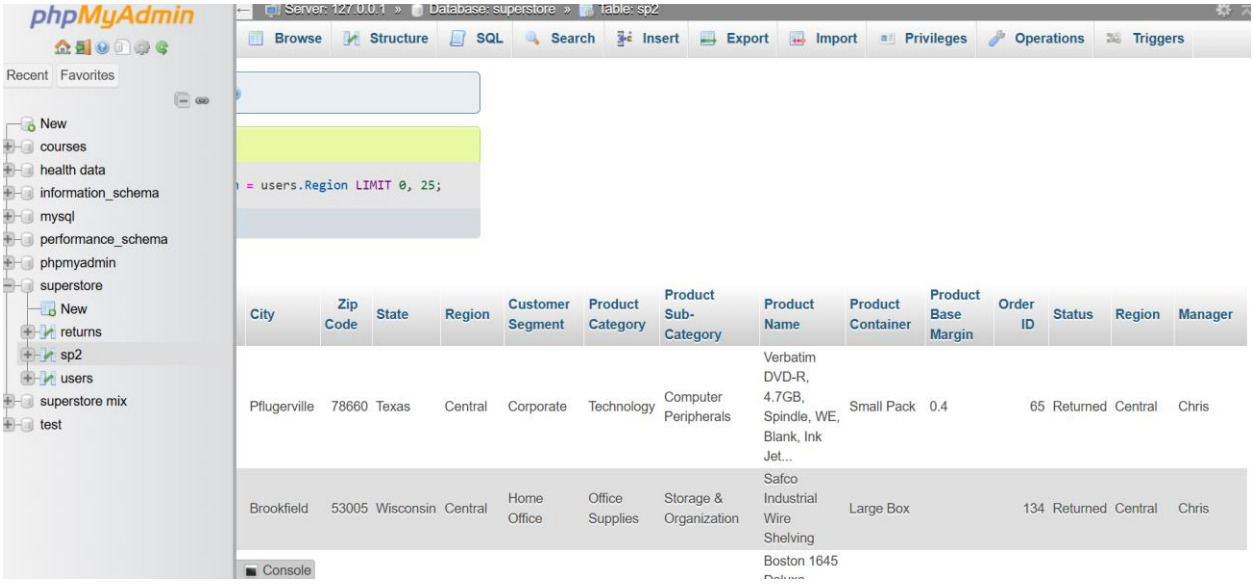
```
SELECT * FROM sp2 LEFT JOIN returns ON sp2.`Order ID` = returns.`Order ID` LEFT JOIN users ON sp2.Region = users.Region LIMIT 0, 25;
```

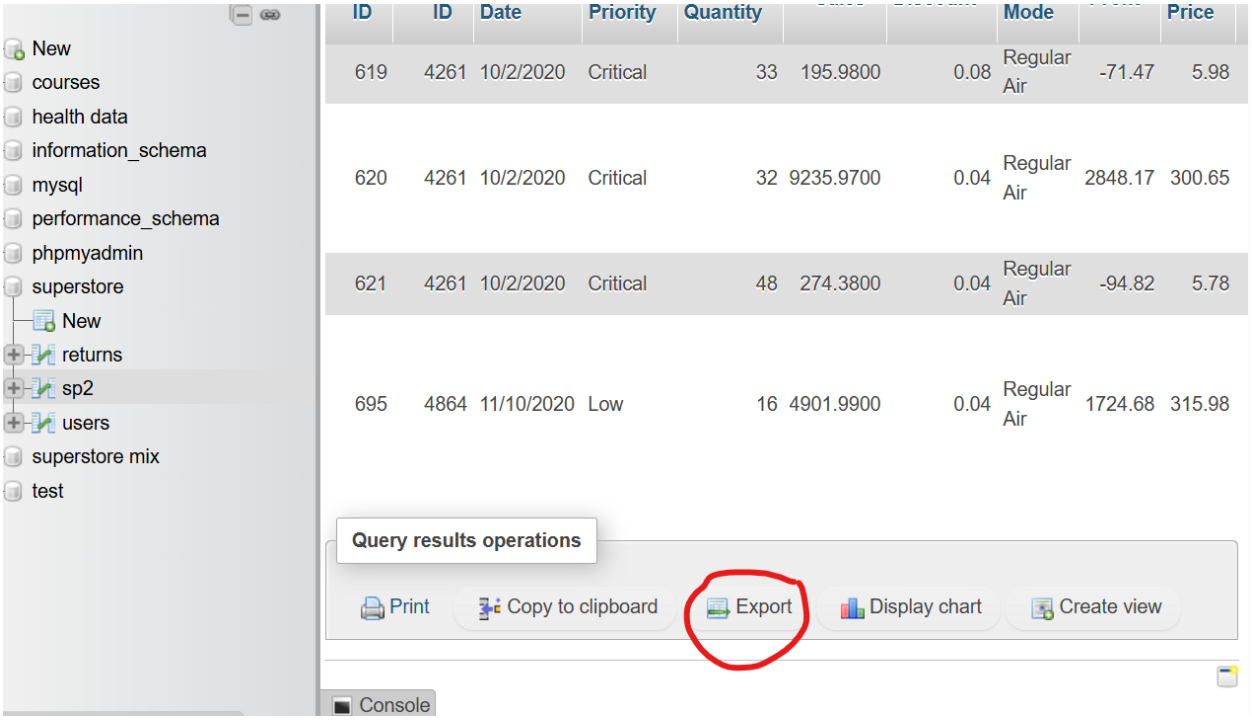
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

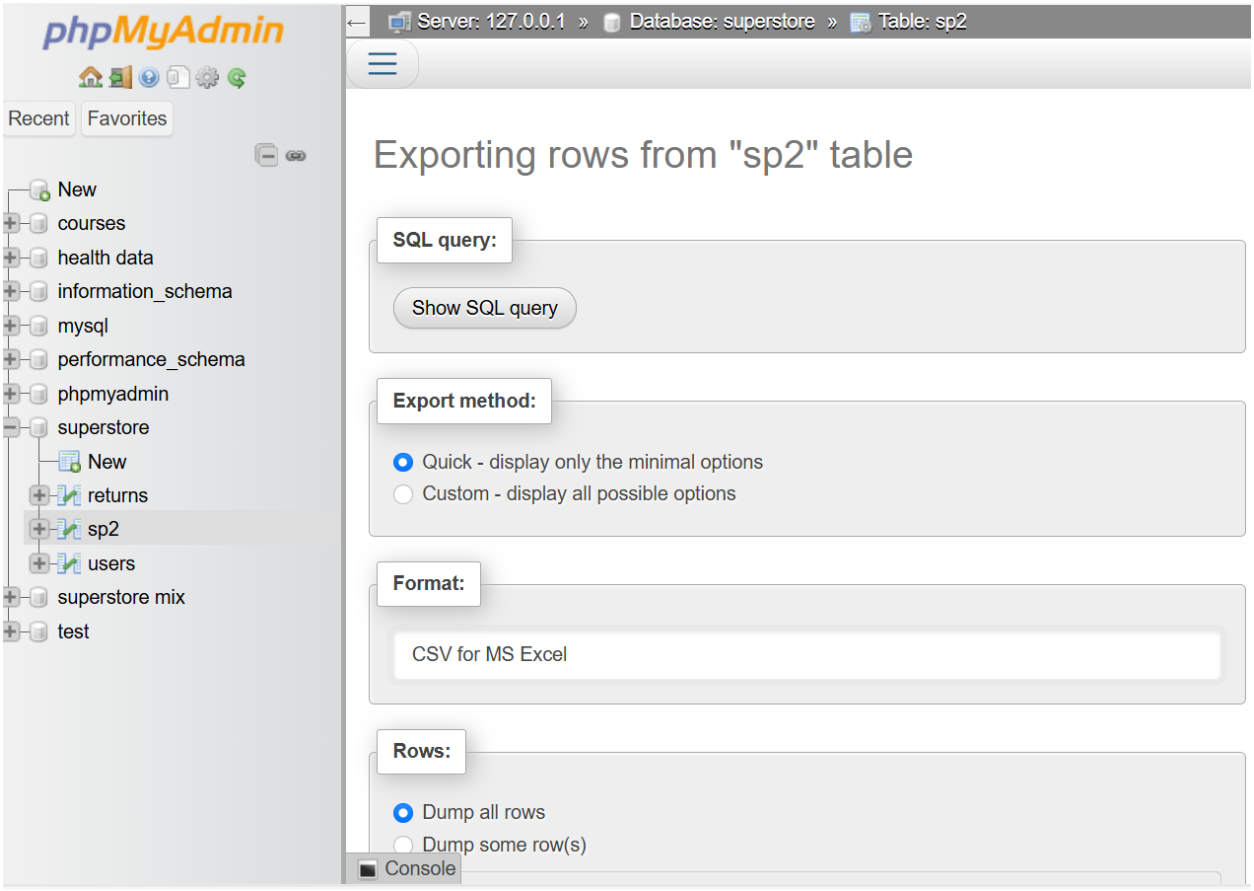
Extra options

Row ID	Order ID	Order Date	Order Priority	Order Quantity	Sales	Discount	Ship Mode	Profit	Unit Price	Shipping Cost	Customer Name	City	Zip Code	State	Region
10	65	3/17/2019	Critical	32	3812.7300	0.02	Regular Air	1470.30	115.79	1.99	Tamara Dahlen	Pflugerville	78660	Texas	Central
24	134	4/30/2020	Not Specified	11	1132.6000	0.01	Regular Air	-310.21	95.99	35.00	Michael Dominguez	Brookfield	53005	Wisconsin	Central

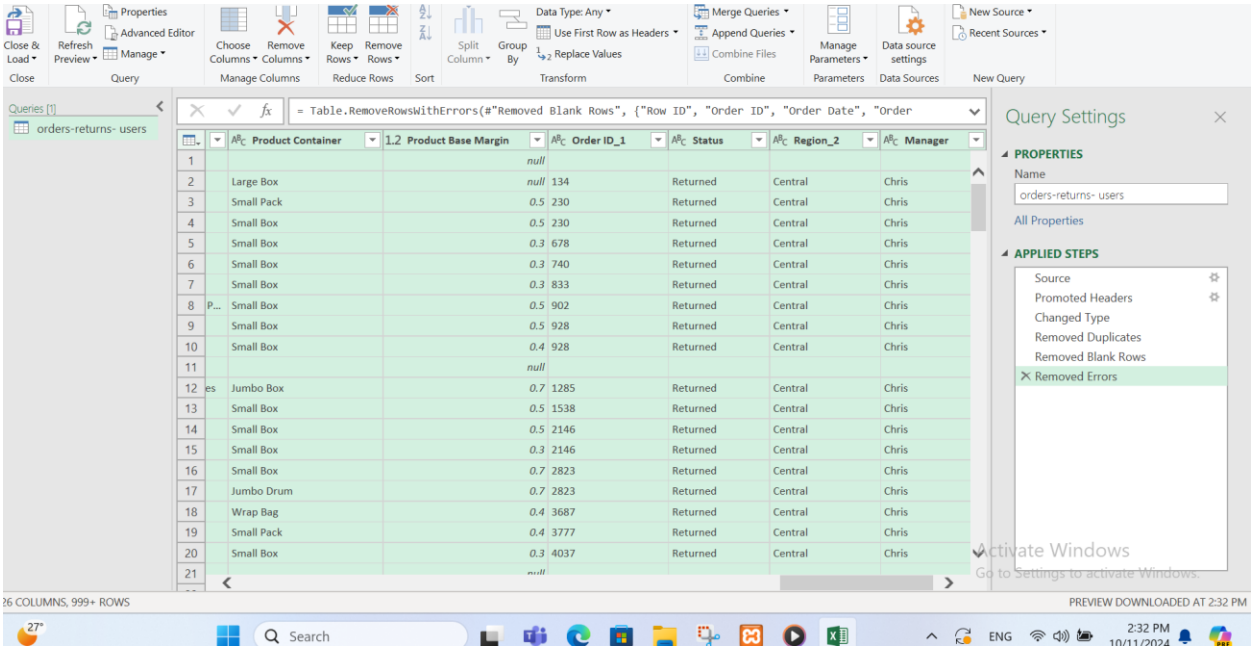
Console

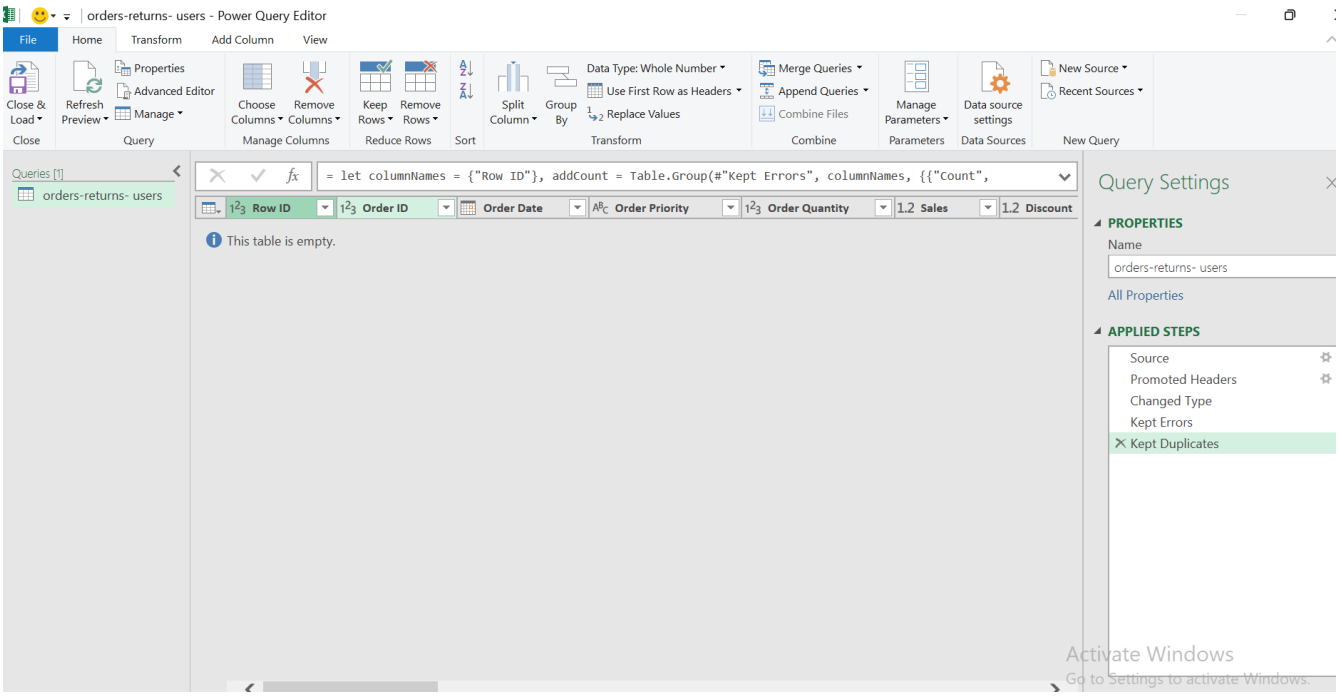



New
courses
health data
information_schema
mysql
performance_schema
phpmyadmin
superstore
New
returns
sp2
users
superstore mix
test
ID ID Date Priority Quantity Mode Price
619 4261 10/2/2020 Critical 33 195.9800 0.08 Regular Air -71.47 5.98
620 4261 10/2/2020 Critical 32 9235.9700 0.04 Regular Air 2848.17 300.65
621 4261 10/2/2020 Critical 48 274.3800 0.04 Regular Air -94.82 5.78
695 4864 11/10/2020 Low 16 4901.9900 0.04 Regular Air 1724.68 315.98
Query results operations
Print
Copy to clipboard
Export
Display chart
Create view
Console

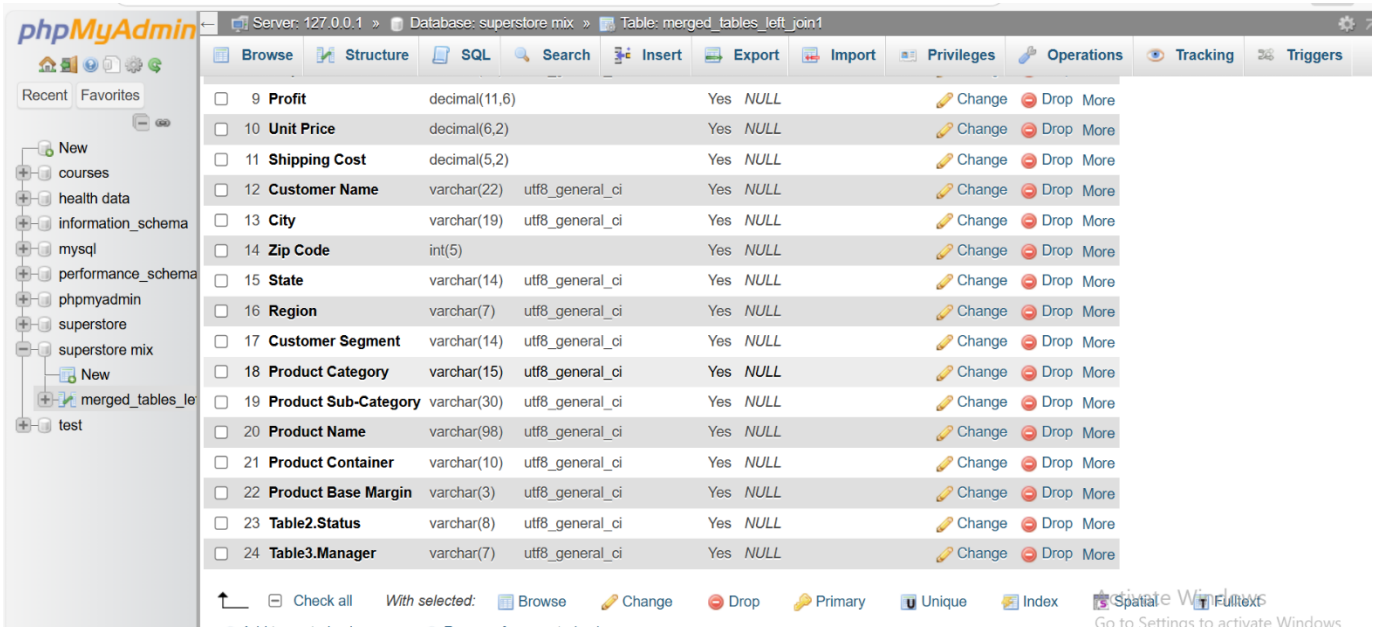

phpMyAdmin
Recent
Favorites
New
courses
health data
information_schema
mysql
performance_schema
phpmyadmin
superstore
New
returns
sp2
users
superstore mix
test
Server: 127.0.0.1 » Database: superstore » Table: sp2
Exporting rows from "sp2" table
SQL query:
Show SQL query
Export method:
Quick - display only the minimal options
Custom - display all possible options
Format:
CSV for MS Excel
Rows:
Dump all rows
Dump some row(s)
Console

Third step I use powerquey in excel to check for errors, duplicate and clean data, also I change the type of data in order date from varchar to date to be ready for analysis

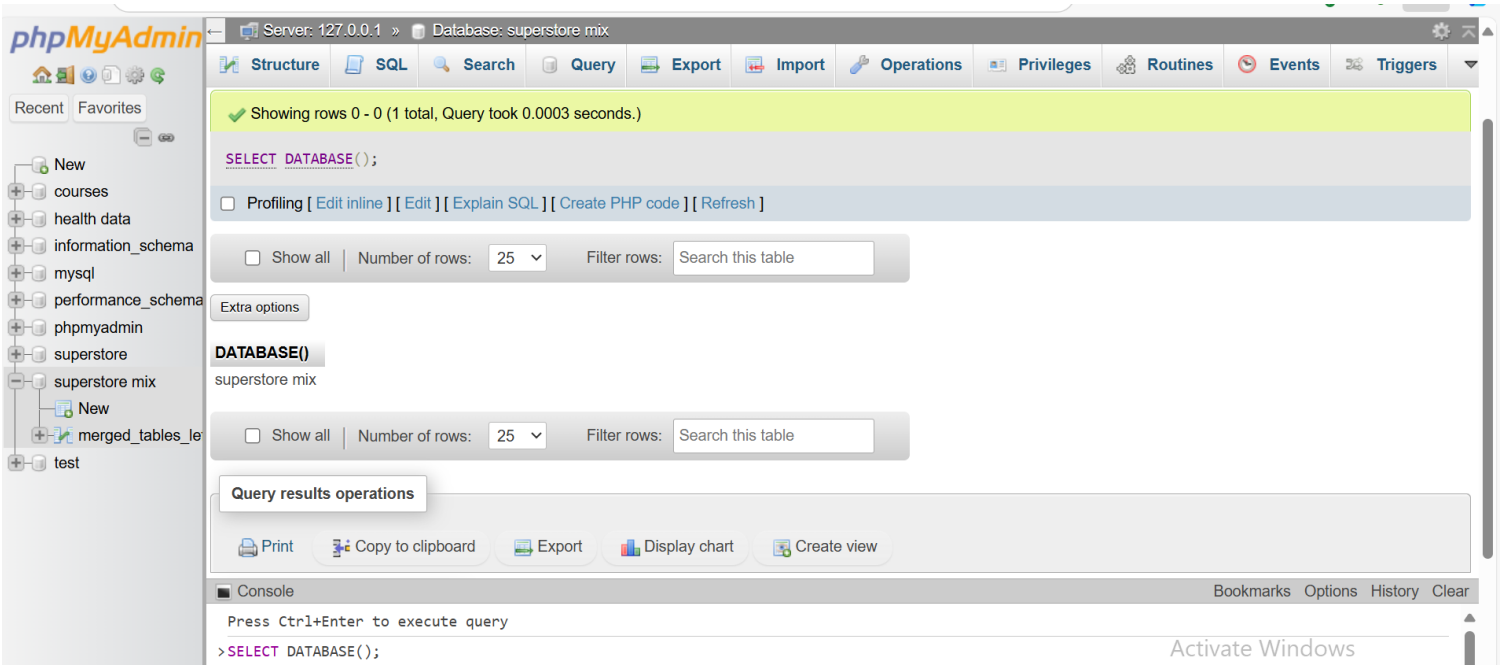




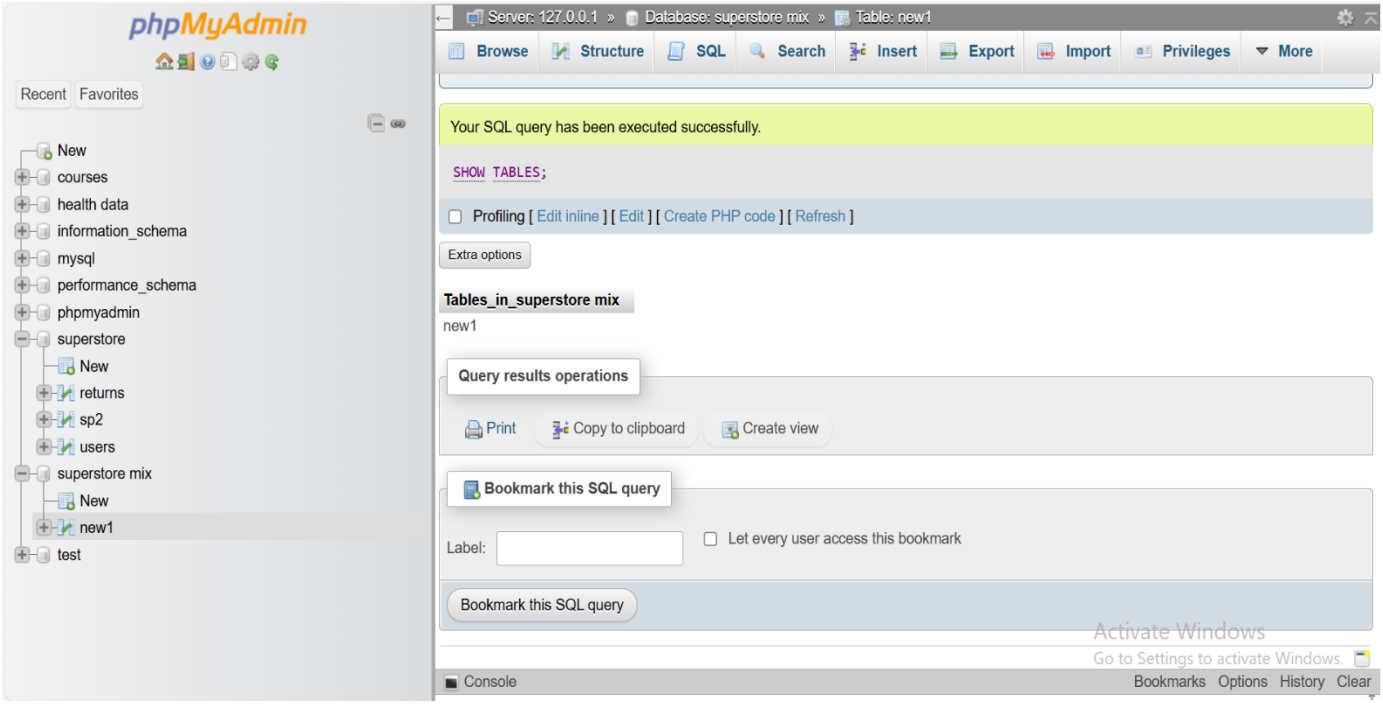
Fourth step after data cleaning now we are ready for analyzing data on phpMyAdmin with the new cleaned merged dataset, I imported it to phpMyAdmin



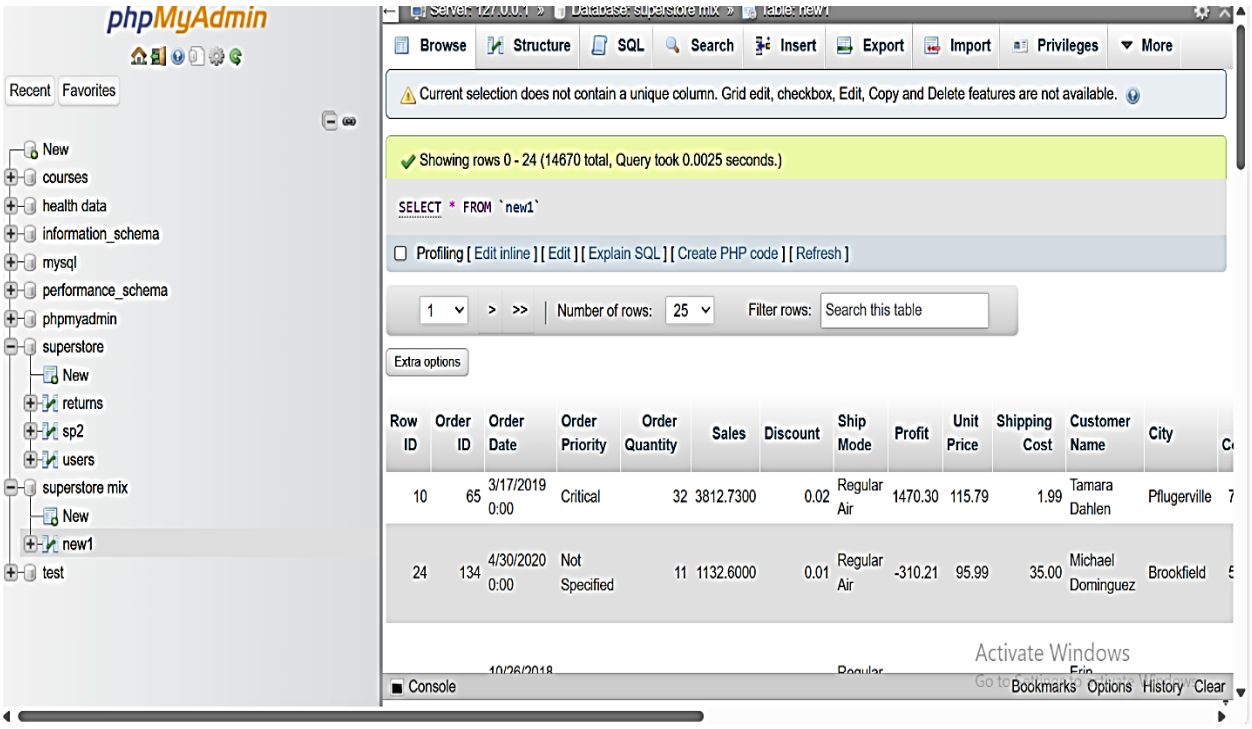
select database () show data base name



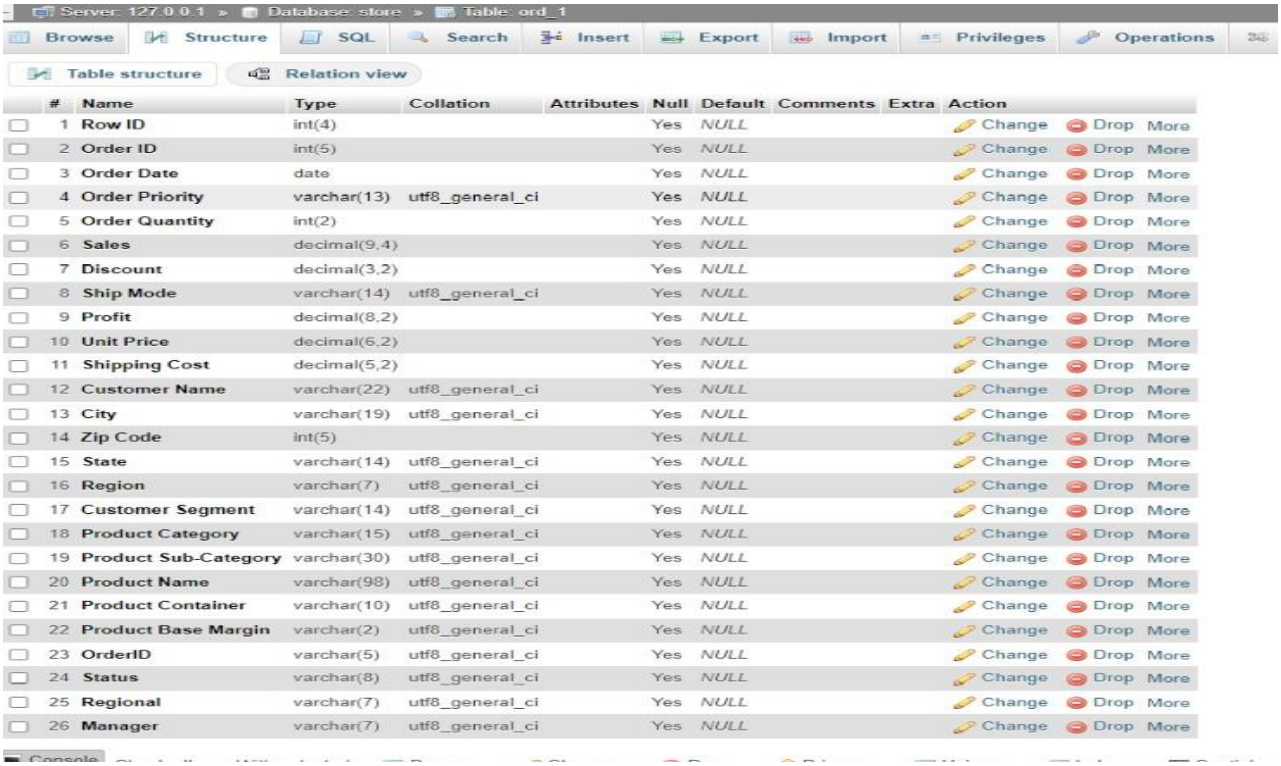
show table in my database



browse of the new table

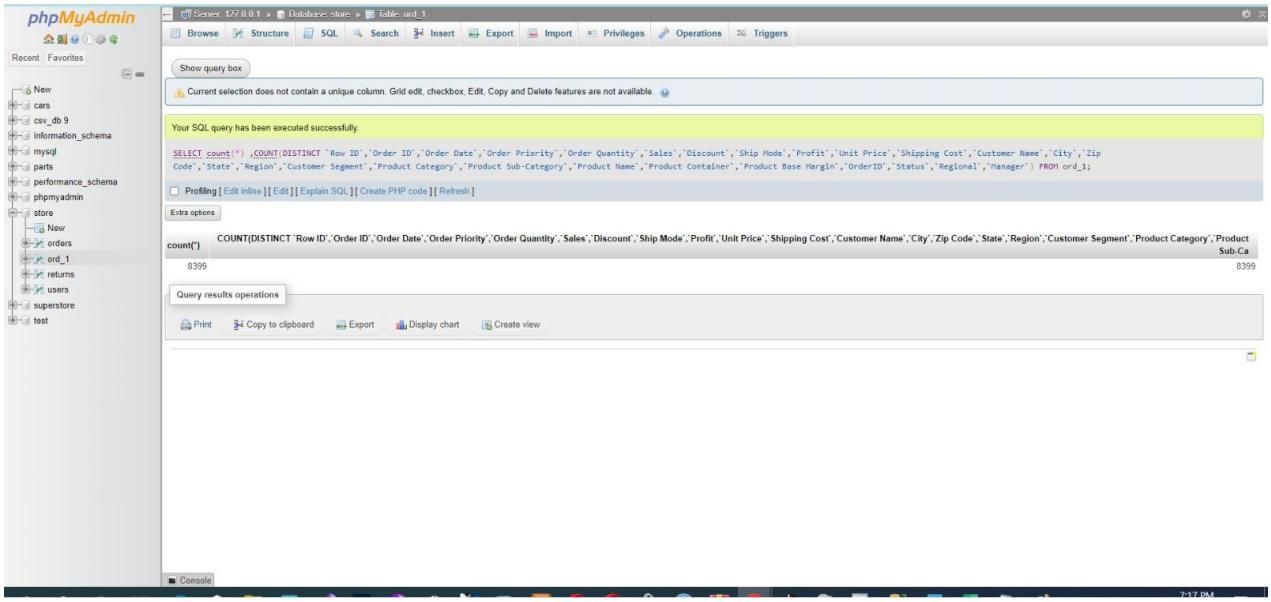


Describe new table

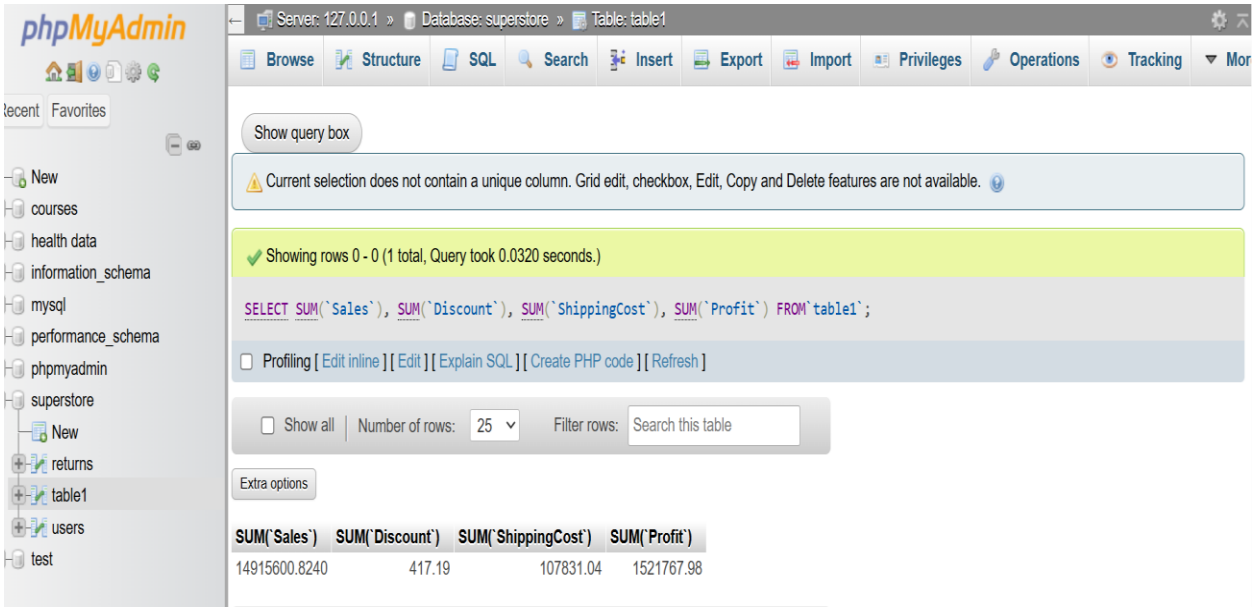


#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	Row ID	int(4)			Yes	NULL			Change Drop More
2	Order ID	int(5)			Yes	NULL			Change Drop More
3	Order Date	date			Yes	NULL			Change Drop More
4	Order Priority	varchar(13)	utf8_general_ci		Yes	NULL			Change Drop More
5	Order Quantity	int(2)			Yes	NULL			Change Drop More
6	Sales	decimal(9,4)			Yes	NULL			Change Drop More
7	Discount	decimal(3,2)			Yes	NULL			Change Drop More
8	Ship Mode	varchar(14)	utf8_general_ci		Yes	NULL			Change Drop More
9	Profit	decimal(8,2)			Yes	NULL			Change Drop More
10	Unit Price	decimal(6,2)			Yes	NULL			Change Drop More
11	Shipping Cost	decimal(5,2)			Yes	NULL			Change Drop More
12	Customer Name	varchar(22)	utf8_general_ci		Yes	NULL			Change Drop More
13	City	varchar(19)	utf8_general_ci		Yes	NULL			Change Drop More
14	Zip Code	int(5)			Yes	NULL			Change Drop More
15	State	varchar(14)	utf8_general_ci		Yes	NULL			Change Drop More
16	Region	varchar(7)	utf8_general_ci		Yes	NULL			Change Drop More
17	Customer Segment	varchar(14)	utf8_general_ci		Yes	NULL			Change Drop More
18	Product Category	varchar(15)	utf8_general_ci		Yes	NULL			Change Drop More
19	Product Sub-Category	varchar(30)	utf8_general_ci		Yes	NULL			Change Drop More
20	Product Name	varchar(98)	utf8_general_ci		Yes	NULL			Change Drop More
21	Product Container	varchar(10)	utf8_general_ci		Yes	NULL			Change Drop More
22	Product Base Margin	varchar(2)	utf8_general_ci		Yes	NULL			Change Drop More
23	OrderID	varchar(5)	utf8_general_ci		Yes	NULL			Change Drop More
24	Status	varchar(8)	utf8_general_ci		Yes	NULL			Change Drop More
25	Regional	varchar(7)	utf8_general_ci		Yes	NULL			Change Drop More
26	Manager	varchar(7)	utf8_general_ci		Yes	NULL			Change Drop More

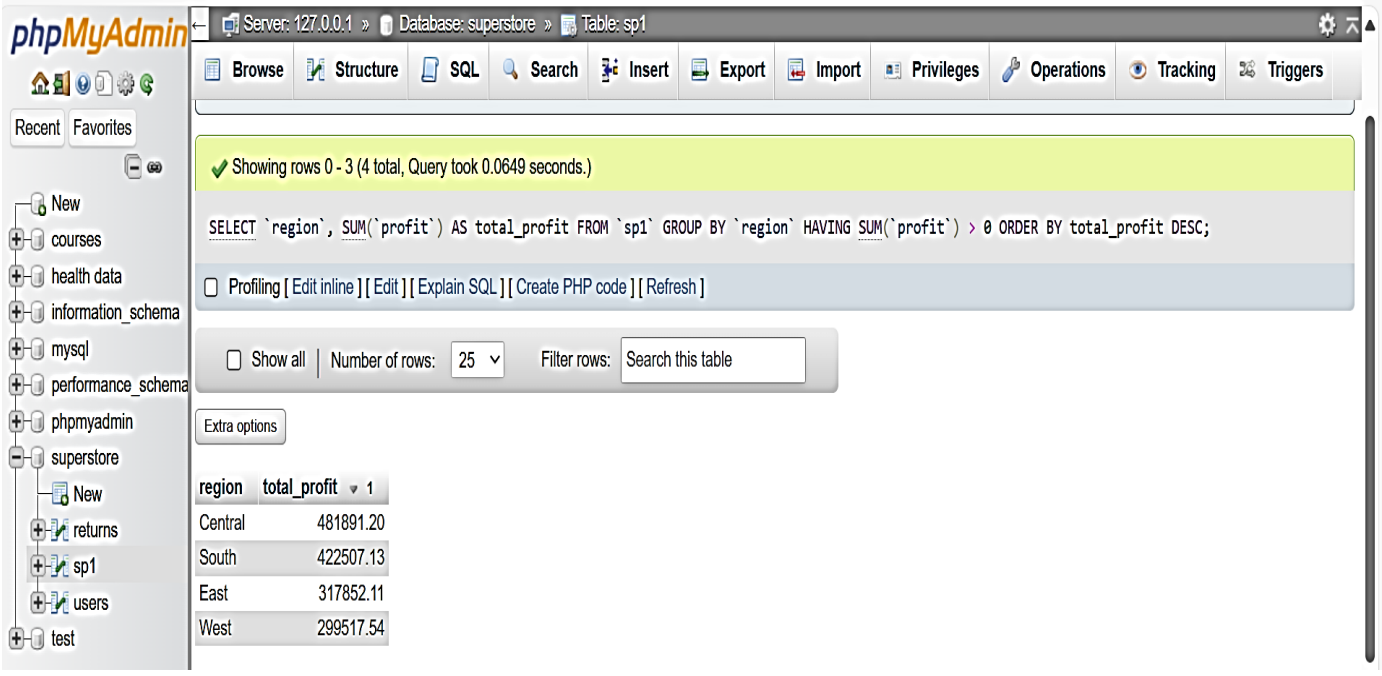
Check for duplicate using count function



Q1: what is the total profit of the store?



Q2: What is the total profit according to region?



Q3: What is the total return according to region and shipping mode and in which category?

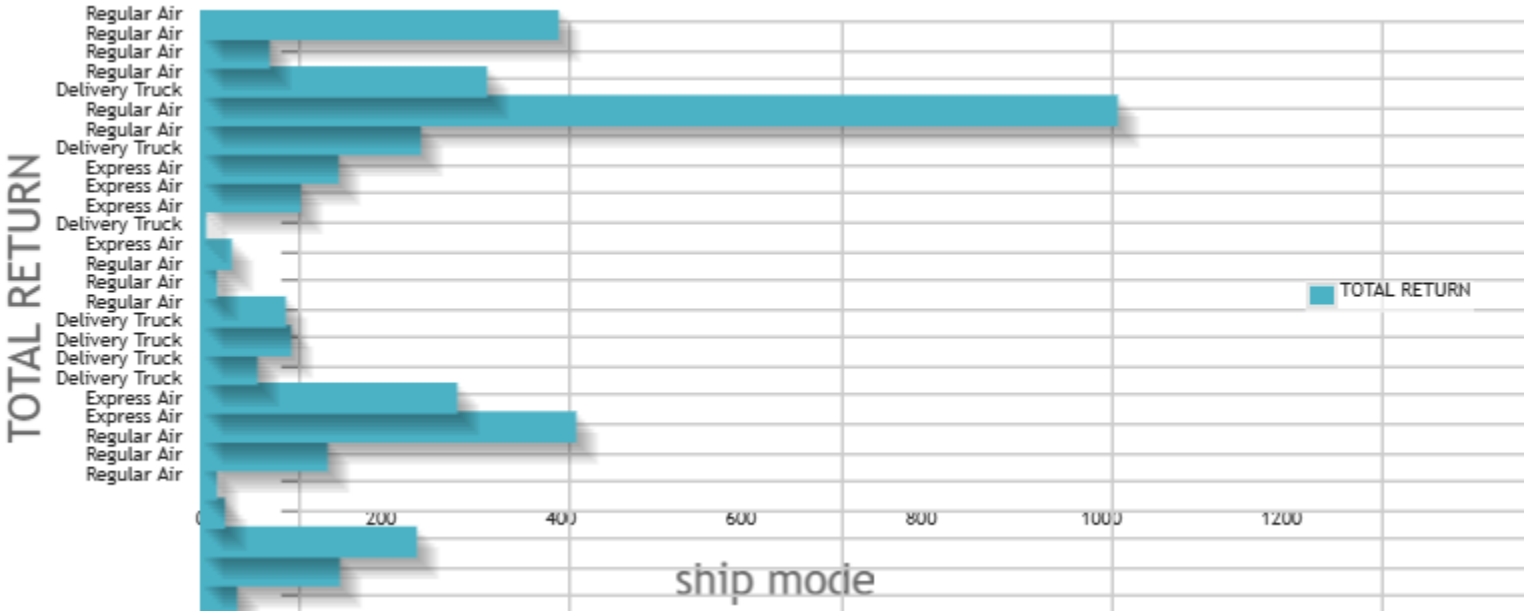
Showing rows 0 - 24 (108 total, Query took 0.0502 seconds.)

```
SELECT DISTINCT `Region`, `Manager`, COUNT(`Status`)AS'TOTAL RETURN', `product category` , `ship mode`
FROM `new1` GROUP BY `Region`, `Manager`,'TOTAL RETURN', `Product Category`, `Ship Mode` ORDER BY
'TOTAL RETURN' DESC;
```

Region	Manager	TOTAL RETURN	product category	ship mode
Central		123	Technology	Regular Air
Central	Chris	972	Office Supplies	Regular Air
Central	Chris	387	Technology	Regular Air
Central	Chris	135	Office Supplies	Express Air
Central	Chris	27	Furniture	Express Air
Central		103	Furniture	Delivery Truck
Central	Chris	160	Furniture	Delivery Truck
Central	Chris	18	Office Supplies	Delivery Truck
Central		12	Office Supplies	Delivery Truck
Central		94	Furniture	Regular Air
Central		278	Office Supplies	Regular Air
Central	Chris	190	Furniture	Regular Air
Central		42	Office Supplies	Express Air
Central	Chris	67	Technology	Delivery Truck
Central	Chris	63	Technology	Express Air
Central		12	Furniture	Express Air
Central		23	Technology	Express Air
Central		4	Technology	Delivery Truck
East		74	Technology	Regular Air
East	Erin	102	Furniture	Regular Air
East	Erin	163	Furniture	Delivery Truck
East	Erin	679	Office Supplies	Regular Air
East		212	Office Supplies	Regular Air
East		51	Furniture	Regular Air
East	Erin	265	Technology	Regular Air

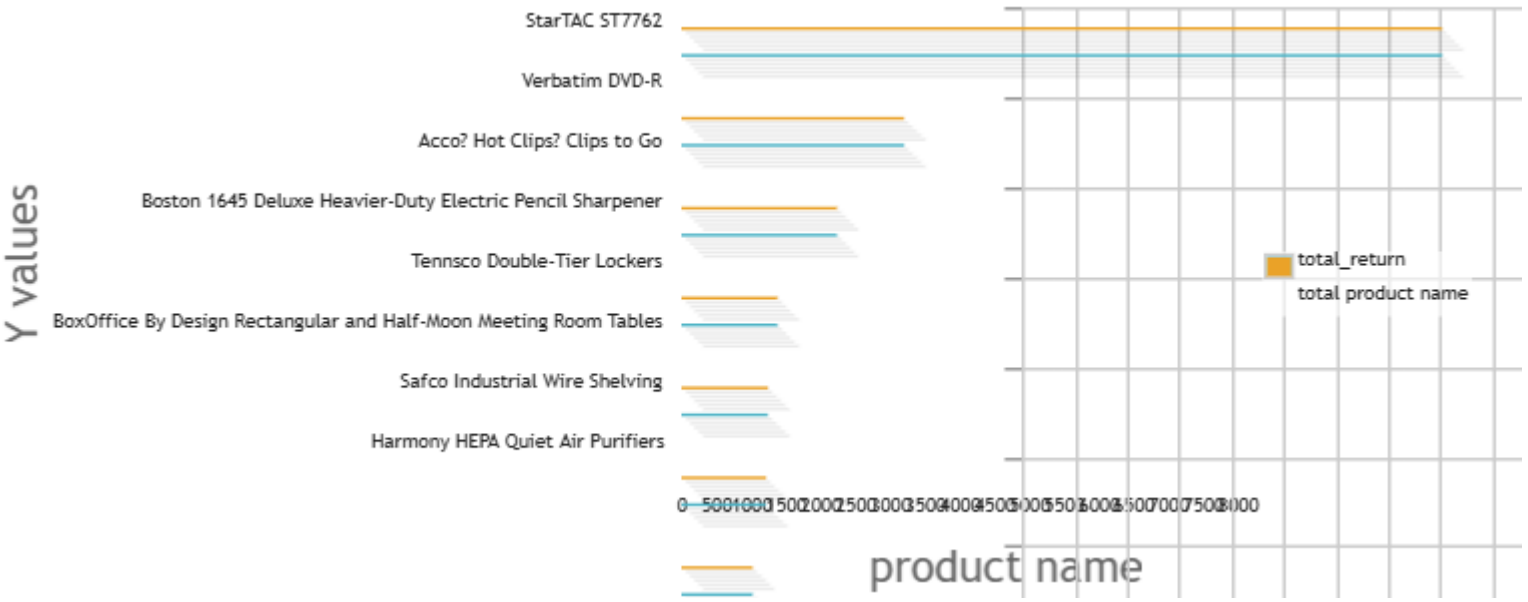
This diagram show that east are has the highest return rate

this diagram shows that Erin has the highest rate of returns



this diagram shows that regular air as a shipping mode has the highest rate of return

Q4: What are the items have the highest rate of return?



This diagram shows that star TAC ST7762 is the highest product name with the highest rate of return.

Q5: Which area and at what year has the highest profit?

```
SELECT YEAR(STR_TO_DATE(`Order date`, '%m/%d/%Y')) AS `year`, `Region`, SUM(`Profit`) AS `total
profit` FROM `new2` GROUP BY `Region`, YEAR(STR_TO_DATE(`Order date`, '%m/%d/%Y')) ORDER BY `total
profit` DESC;
```

year	Region	total profit 1
2017	South	301204.31
2018	Central	291431.14
2017	East	239479.61
2019	Central	235875.67
2019	West	208491.28
2020	Central	200056.21
2017	Central	194380.29
2020	East	186532.38
2019	South	178196.26
2020	South	166703.40
2018	West	144179.49
2018	South	121481.45
2018	East	113529.58
2019	East	96353.80
2017	West	90934.53
2020	West	86637.56

This result is arranged by descending order which means that

south area has the highest profit in 2017

has the highest totl profit while

west area has least profit in 2020

Second: Python part

1- Data Preprocessing and Cleaning**

This process involves merging datasets, handling missing values, removing duplicates, and ensuring consistent data types.

Step 1: Import Necessary Libraries

We will use `pandas` for loading and manipulating the datasets, `NumPy` to facilitate efficient numerical operations on large quantities of data, `plotly express` and `matplotlib` and `seaborn` for visualizations and allows you to create interactive plots with very little code.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
```

Step 2: Load Datasets

We have three CSV files: `orders.csv`, `returns.csv`, and `users.csv`. These will be loaded and merged for further analysis.

```
Load the datasets (adjust file paths as needed)

orders = pd.read_csv('orders.csv', encoding='ISO-8859-1')
returns = pd.read_csv('returns.csv', encoding='ISO-8859-1')
users = pd.read_csv('users.csv', encoding='ISO-8859-1')
```

Step 3: Merge Datasets

We will first merge the `orders` and `returns` datasets using the common column `Region`, then merge the result with the `users` dataset.

```
Merge orders and returns on 'Region'

merged_returns = pd.merge(orders, returns, on='Order ID', how='left')

# Merge the result with users data on 'Region'
merged_users = pd.merge(merged_returns, users, on='Region', how='left')

# Convert merged files to CSV file
merged_returns.to_csv('superstore_sales.csv', index=False)
```

Step 4: Data Exploration

Now that we have the merged data, let's explore it to understand its structure, missing values, and any necessary transformations.

```
Load the merged dataset

superstore_sales = pd.read_csv('superstore_sales.csv')
Display the rows of the dataset

superstore_sales
 Display the dataset information
superstore_sales.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8399 entries, 0 to 8398
Data columns (total 23 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Row ID               8399 non-null   int64
 1   Order ID             8399 non-null   int64
 2   Order Date           8399 non-null   object
 3   Order Priority       8399 non-null   object
 4   Order Quantity       8399 non-null   int64
 5   Sales                8399 non-null   float64
 6   Discount             8399 non-null   float64
 7   Ship Mode            8399 non-null   object
 8   Profit               8399 non-null   float64
 9   Unit Price           8399 non-null   float64
 10  Shipping Cost        8399 non-null   float64
 11  Customer Name        8399 non-null   object
 12  City                 8399 non-null   object
 13  Zip Code             8399 non-null   int64
 14  State                8399 non-null   object
 15  Region               8399 non-null   object
 16  Customer Segment     8399 non-null   object
 17  Product Category     8399 non-null   object
 18  Product Sub-Category 8399 non-null   object
 19  Product Name         8399 non-null   object
...
 21  Product Base Margin  8336 non-null   float64
 22  Status               872 non-null    object
dtypes: float64(6), int64(4), object(13)
memory usage: 1.5+ MB
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

```
Check for null values in the dataset
```

```
superstore_sales.isnull().sum()
```

```
Row ID                0
Order ID              0
Order Date            0
Order Priority        0
Order Quantity        0
Sales                 0
Discount              0
Ship Mode             0
Profit                0
Unit Price            0
Shipping Cost         0
Customer Name         0
City                  0
Zip Code              0
State                 0
Region                0
Customer Segment      0
Product Category      0
Product Sub-Category  0
Product Name          0
Product Container     0
Product Base Margin   63
Status                7527
dtype: int64
```

```
Check for duplicates in the dataset

superstore_sales.duplicated().sum()
```

```
np.int64(0)
```

```
Display summary statistics of the dataset

superstore_sales.describe(include='all')

def detect_outliers_iqr(superstore_sales, column):
    Q1 = superstore_sales[column].quantile(0.25)  # First quartile
    Q3 = superstore_sales[column].quantile(0.75)  # Third quartile
    IQR = Q3 - Q1  # Interquartile range

    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    # Identify outliers
    outliers = superstore_sales[(superstore_sales[column] < lower_bound) | (superstore_sales[column] >
upper_bound)]
    return outliers

# Example: detecting outliers in Sales, Shipping Cost, Unit Price

sales_outliers = detect_outliers_iqr(superstore_sales, 'Sales')
shipping_outliers = detect_outliers_iqr(superstore_sales, 'Shipping Cost')
price_outliers = detect_outliers_iqr(superstore_sales, 'Unit Price')
discount_outliers = detect_outliers_iqr(superstore_sales, 'Discount')
profit_outliers = detect_outliers_iqr(superstore_sales, 'Profit')

# Display number of outliers
print(f"Sales outliers: {len(sales_outliers)}")
print(f"Shipping Cost outliers: {len(shipping_outliers)}")
print(f"Unit Price outliers: {len(price_outliers)}")
print(f"Discount outliers: {len(discount_outliers)}")
print(f"Profit outliers: {len(profit_outliers)}")
```

```
Sales outliers: 1042
Shipping Cost outliers: 972
Unit Price outliers: 848
Discount outliers: 3
Profit outliers: 1704
```

Data Exploration Results:

1. The dataset has `33596` rows and `24` columns .

2. There may be numerical features like sales amounts and categorical features like regions, product categories, etc.

3. The `Order Date` column are in object format and will require conversion to `datetime` format for further analysis.

4. we have 2 columns with missing values and it is `Product Base Margin` column, which has 63 missing values and we will fill it with `?` sign and `Status` column and we will fill missing cells with Sold.

5. We have no duplicated values

6. We have Outliears in Sales, Profit, Shipping Cost and Profit columns

Step 4: Data Cleaning and Preprocessing

Based on our exploration, we will now handle missing values, convert date columns to proper formats, and ensure there are no duplicates.

```
Handling missing values by filling missing Postal Code values with '?'

superstore_sales['Product Base Margin'] = superstore_sales['Product Base Margin'].fillna(0)

superstore_sales['Status'] = superstore_sales['Status'].fillna('Sold')
Convert 'Order Date' and 'Ship Date' to datetime format if they exist

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

Check if cleaning was successful
superstore_sales.isnull().sum()
```

```
Row ID              0
Order ID            0
Order Date          0
Order Priority      0
Order Quantity      0
Sales               0
Discount            0
```

```
Ship Mode              0
Profit                 0
Unit Price             0
Shipping Cost          0
Customer Name          0
City                   0
Zip Code               0
State                  0
Region                 0
Customer Segment       0
Product Category       0
Product Sub-Category   0
Product Name           0
Product Container      0
Product Base Margin    0
Status                 0
dtype: int64
```

```
Check for duplicates in the cleaned dataset
superstore_sales.duplicated().sum()

np.int64(0)

Display the cleaned dataset information
superstore_sales.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8399 entries, 0 to 8398
Data columns (total 23 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Row ID               8399 non-null   int64
 1   Order ID             8399 non-null   int64
 2   Order Date           8399 non-null   datetime64[ns]
 3   Order Priority       8399 non-null   object
 4   Order Quantity       8399 non-null   int64
 5   Sales                8399 non-null   float64
 6   Discount             8399 non-null   float64
 7   Ship Mode            8399 non-null   object
 8   Profit               8399 non-null   float64
 9   Unit Price           8399 non-null   float64
 10  Shipping Cost        8399 non-null   float64
 11  Customer Name        8399 non-null   object
 12  City                 8399 non-null   object
 13  Zip Code             8399 non-null   int64
 14  State                8399 non-null   object
 15  Region               8399 non-null   object
 16  Customer Segment     8399 non-null   object
 17  Product Category     8399 non-null   object
 18  Product Sub-Category 8399 non-null   object
```

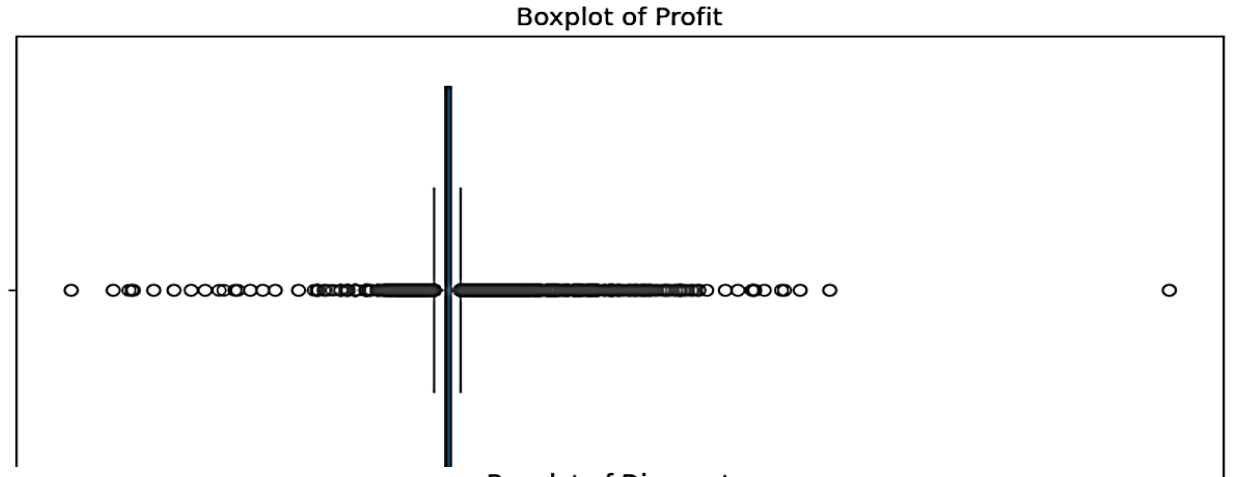
```
 19  Product Name          8399 non-null   object
...
 21  Product Base Margin   8399 non-null   float64
 22  Status                8399 non-null   object
dtypes: datetime64[ns](1), float64(6), int64(4), object(12)
memory usage: 1.5+ MB
```
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

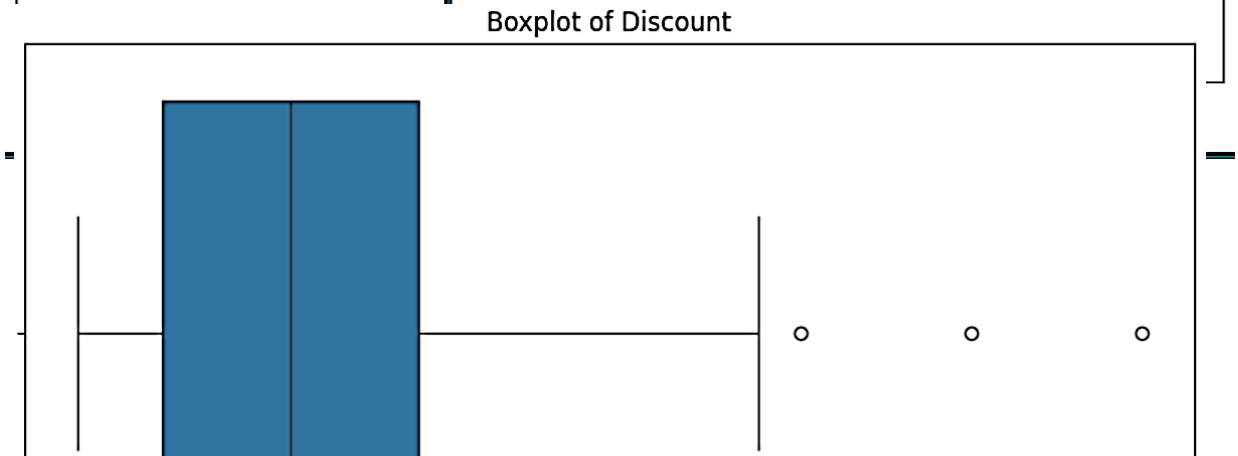
```
superstore_sales
```

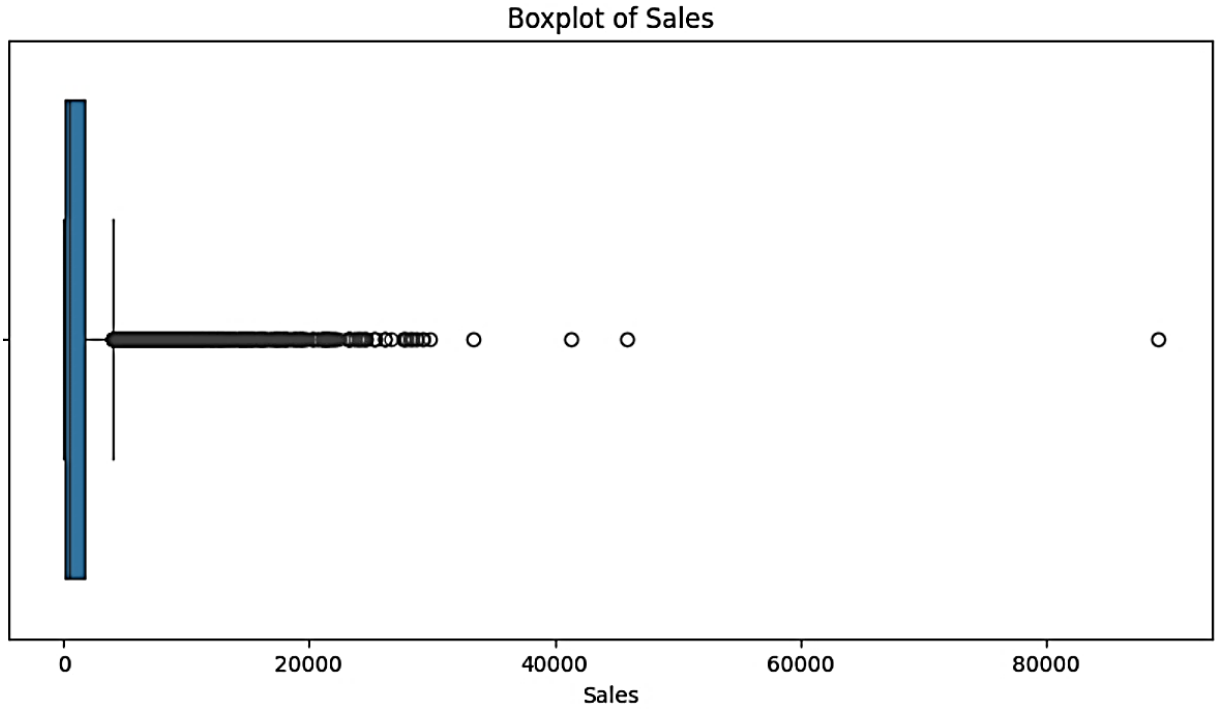
Step 5: Outliers in Sales and Profit

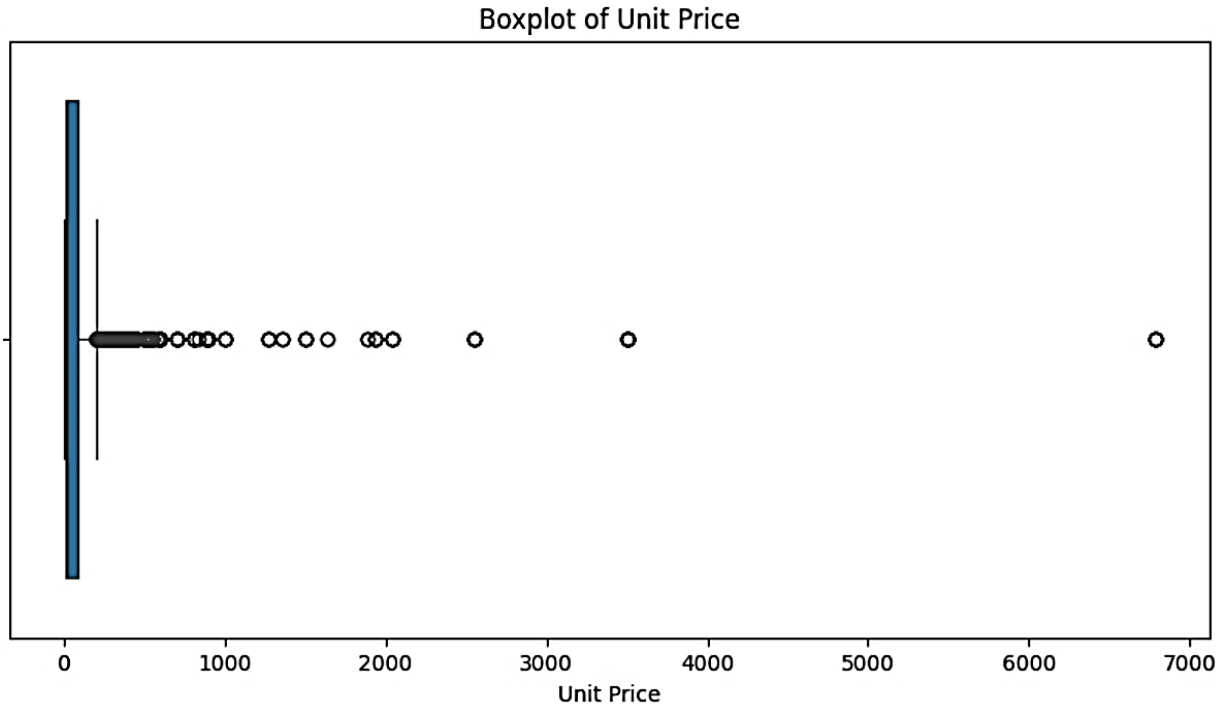
Let's create box plots for Sales and Profit to visualize any outliers. These box plots will show us the spread of the data, with the potential outliers displayed as points outside the whiskers.

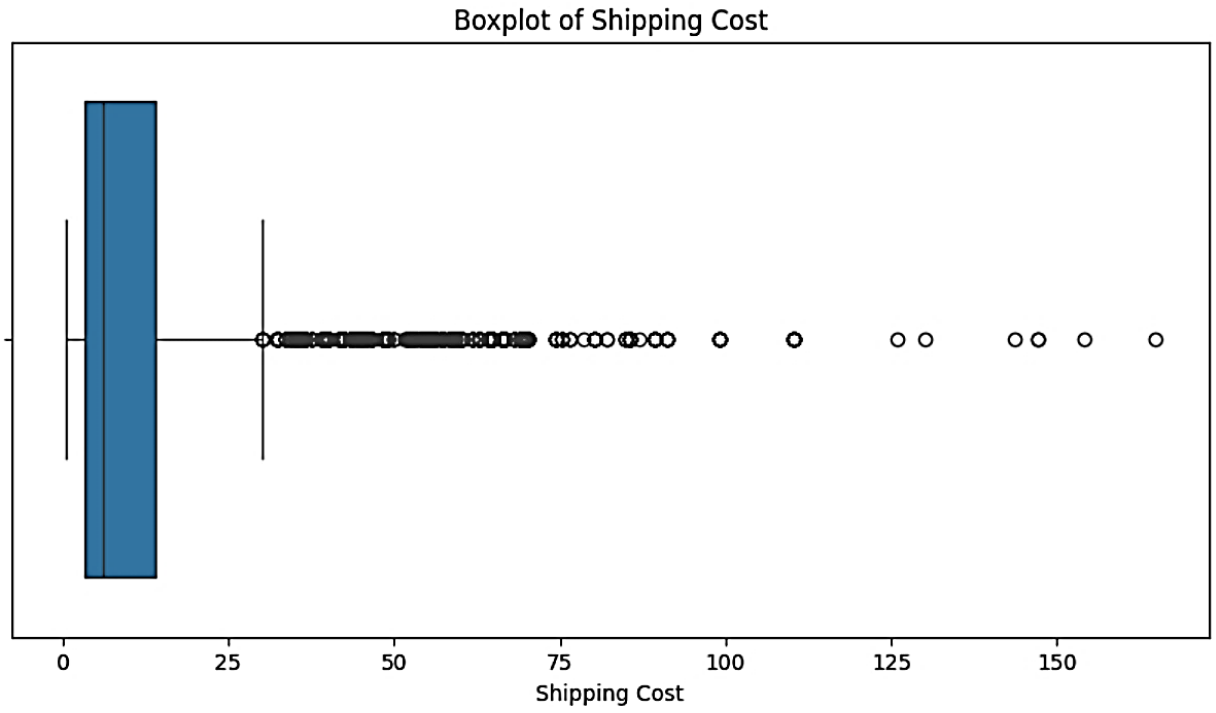
```
autoclean_outliers_columns = superstore_sales[['Profit','Discount','Sales','Unit Price','Shipping Cost']]

# Create a boxplot for each numerical feature
for col in autoclean_outliers_columns:
    plt.figure(figsize=(10, 5))
    sns.boxplot(x=superstore_sales[col])
    plt.title(f'Boxplot of {col}')
    plt.show()
```






Boxplot of Sales
0
20000
40000
60000
80000
Sales





Save dataset after cleaning as CSV file

```
superstore_sales.to_csv("cleaned_superstore_sales.csv", index=False)
```

The dataset is now cleaned and ready for further analysis.

2- Data Analysis Questions**

We will outline key analysis questions that can be derived from the dataset. These questions will provide valuable insights for decision-makers and help guide future analyses, such as understanding product performance, sales trends, and regional performance.

1. Sales Performance Analysis Questions

1- What are the trends in sales performance over time (monthly, quarterly, yearly)?

2- How do different seasons or holiday periods affect sales trends?

3- How does each product category contribute to overall sales?

4- Which regions or cities have the highest and lowest sales performance?

5- Slaes over both regoins and years

6- What is the impact of discounts on sales volume and revenue?

2. Customer Behavior Questions:

1- What are the top customer segments by total sales or frequency of purchases?

2- What are the customer retention rates across different regions?

3- What is the estimated customer lifetime value across different segments?

4- Which customers are likely to stop purchasing based on purchase frequency?

3. Product Category Analysis Questions

1- Which product categories generate the most revenue?

2- How do different product categories perform in different regions?

3- What are the stock levels across different product categories and locations?

4- How quickly are products selling out, and which categories have faster turnover?

5- What is the relationship between product price and sales volume?

****4. Profitability Questions:****

1- Which regions or product categories have the highest profit margins?

2- How do shipping costs affect profitability across different regions?

3- Are there any specific customer segments that contribute to higher profitability?

4- the relation between profit and discount

5- profit over years

6- Profit over both regoins and years

7- The highest and the lowest five product based on profit

8- The relation between shipping mode and profit

9- the relation between Unit price and Profit

10- The affect of container size on Shipment Cost and Profit Amount

11- who is the top and low five customer based on sales and profit that the company may target to improve preformance

****5. Losses analysis****

questions:

1-what is the losses of each status?

2-which mode lose the most?

3-which priority lose the most?

4-which container lose the most?

5-which segment lose the most?

6-which region lose the most?

7-which category lose the most?

8-which subcategory loses the most?

6. Returning questions

questions:

1-how many orders are returned concerned to the sold ones?

2-is the problem in a certain ship mode?

3-is the problem in a certain Category?

4-is the problem in a certain SubCategory?

5-what are the top Products returned?

6-is the problem in a certain Region?

7-is the problem in a certain Customer segment?

8-who are the most customer returning orders?

9-is the problem in a certain container?

10-is the problem in a certain priority?

3- Forecasting Questions Phase

We will determine a set of forecasting questions and answer them using the trends found in the given dataset.

1. Sales Performance Analysis Questions

1- What are the trends in sales performance over time (monthly, quarterly, yearly)?

2- How do different seasons or holiday periods affect sales trends?

3- How does each product category contribute to overall sales?

4- Which regions or cities have the highest and lowest sales performance?

5- Slaes over both regoins and years

6- What is the impact of discounts on sales volume and revenue?

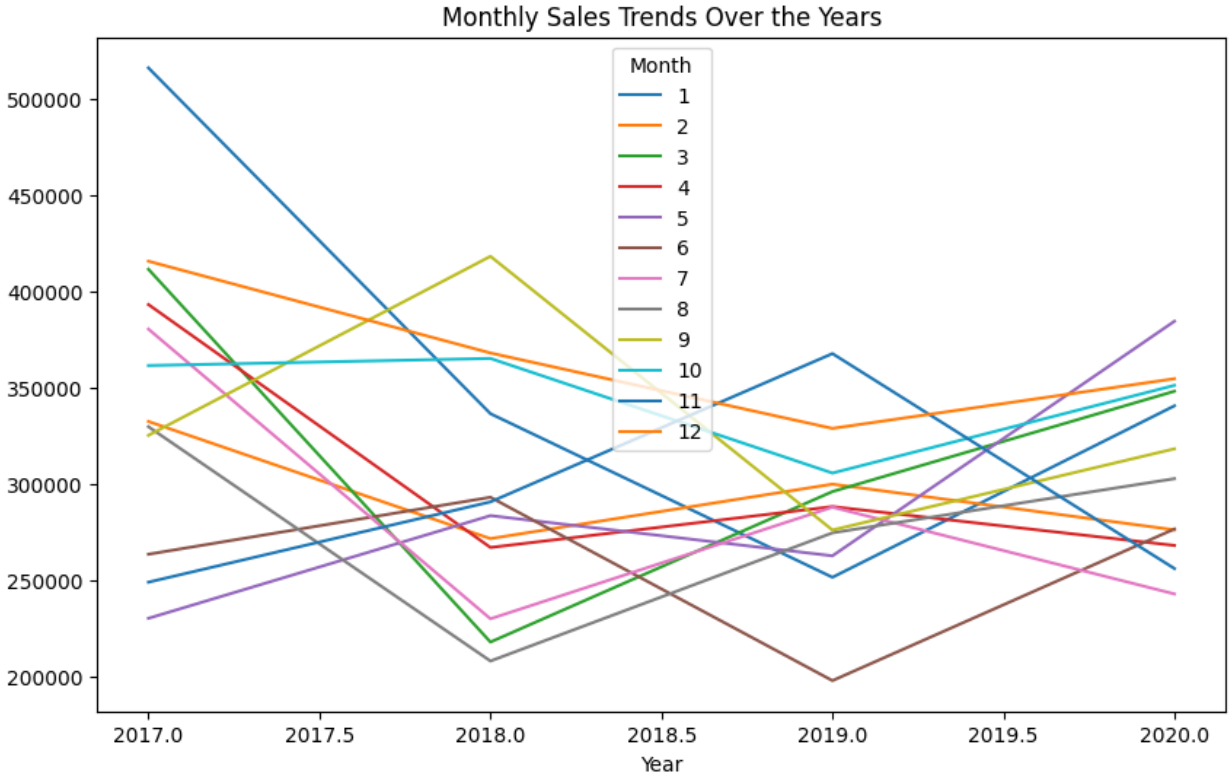
1- Sales Trends Over Time

To analyze sales trends over time, we will group the sales data by year, quarter, and month.

```
 Extract year and month from the 'Order Date' column
superstore_sales['Year'] = pd.to_datetime(superstore_sales['Order Date']).dt.year
superstore_sales['Month'] = pd.to_datetime(superstore_sales['Order Date']).dt.month
superstore_sales['Quarter'] = pd.to_datetime(superstore_sales['Order Date']).dt.quarter


sales_trends = superstore_sales.groupby(['Year', 'Month'])['Sales'].sum().unstack()
sales_trends.plot(kind='line', title='Monthly Sales Trends Over the Years', figsize=(10, 6))
```

<Axes: title={'center': 'Monthly Sales Trends Over the Years'}, xlabel='Year'>

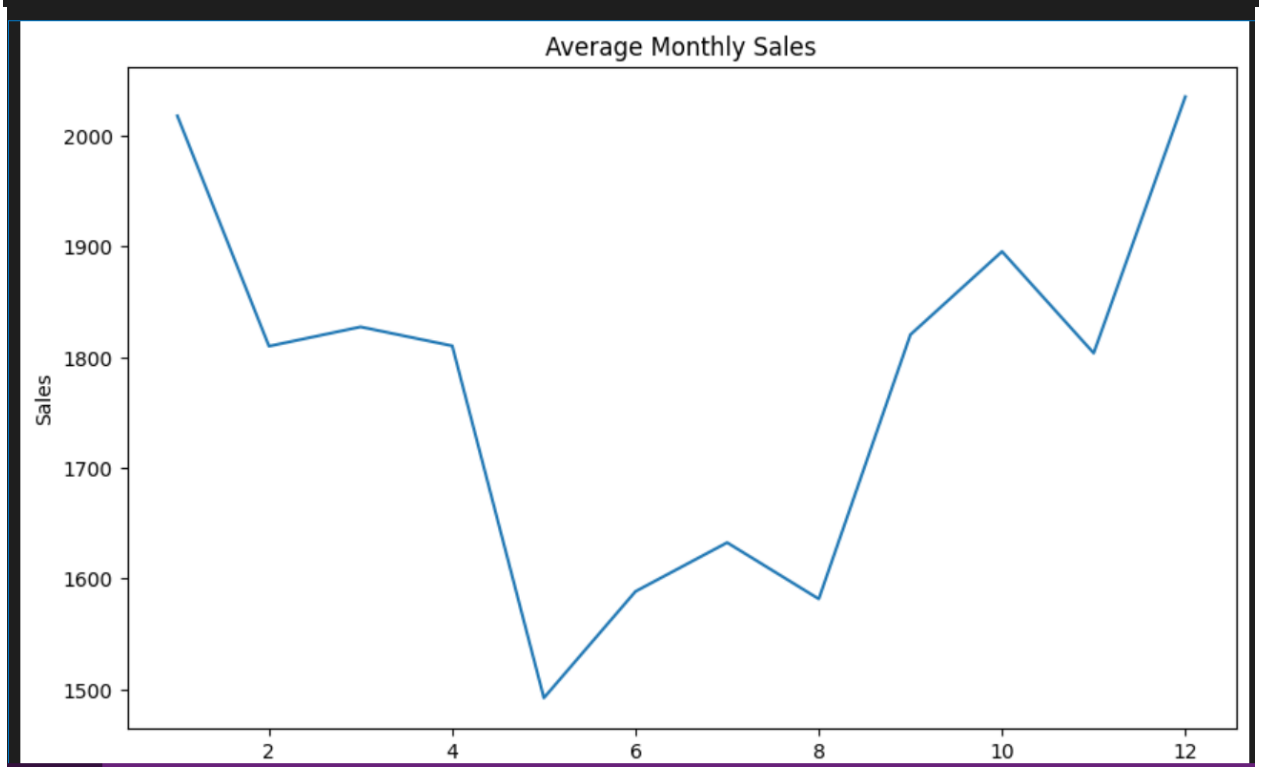
2- How do different seasons or holiday periods affect sales trends



2- How do different seasons or holiday periods affect sales trends

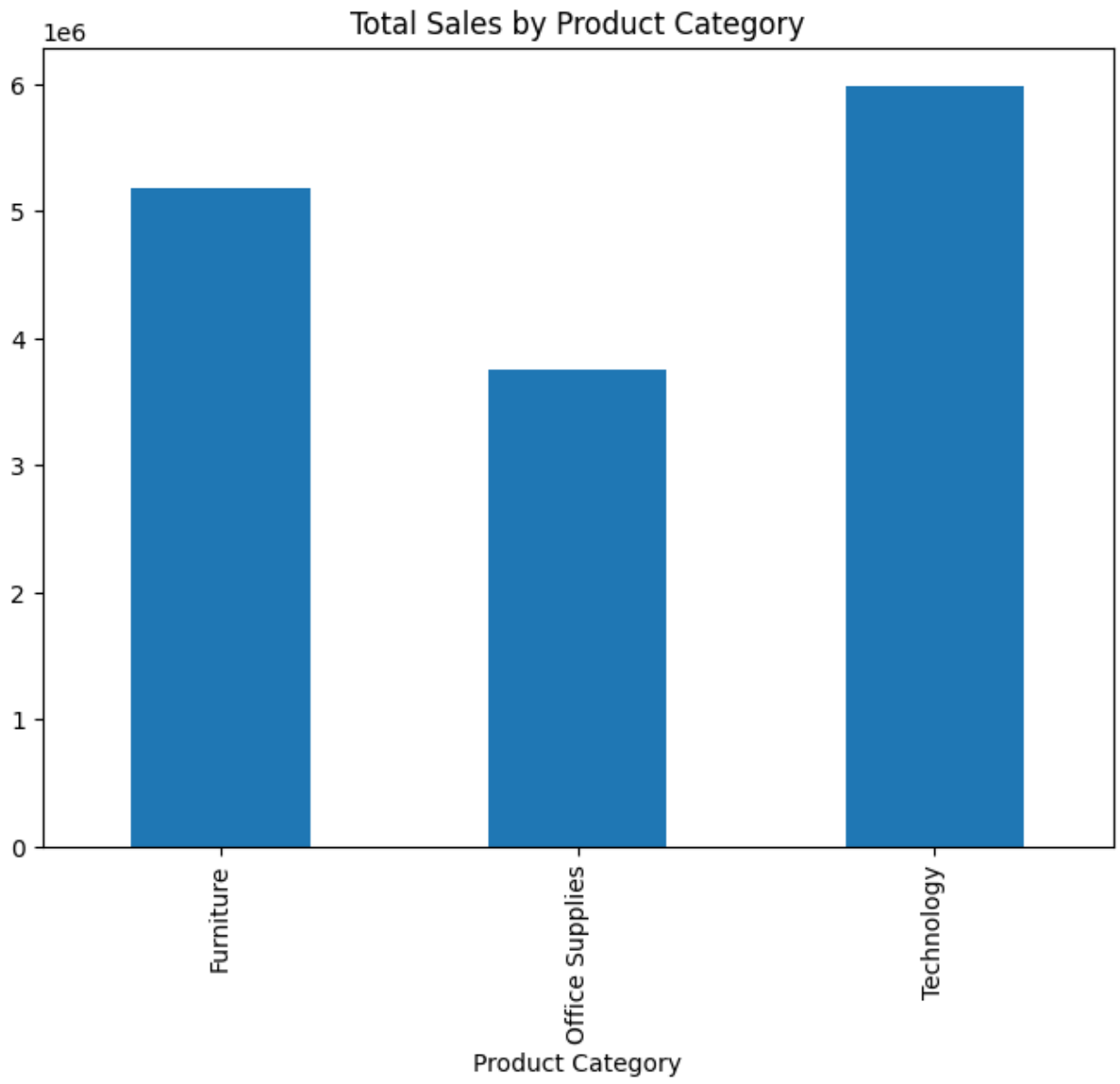
```
Extract the month and day from the 'Order Date' column
superstore_sales['Month'] = superstore_sales['Order Date'].dt.month
superstore_sales['Day'] = superstore_sales['Order Date'].dt.day

# Analyze average monthly sales to identify seasonality patterns
```

```
monthly_sales = superstore_sales.groupby('Month')['Sales'].mean()
monthly_sales.plot(kind='line', title='Average Monthly Sales', xlabel='Month', ylabel='Sales', figsize=(10, 6))

<Axes: title={'center': 'Average Monthly Sales'}, xlabel='Month', ylabel='Sales'>
```

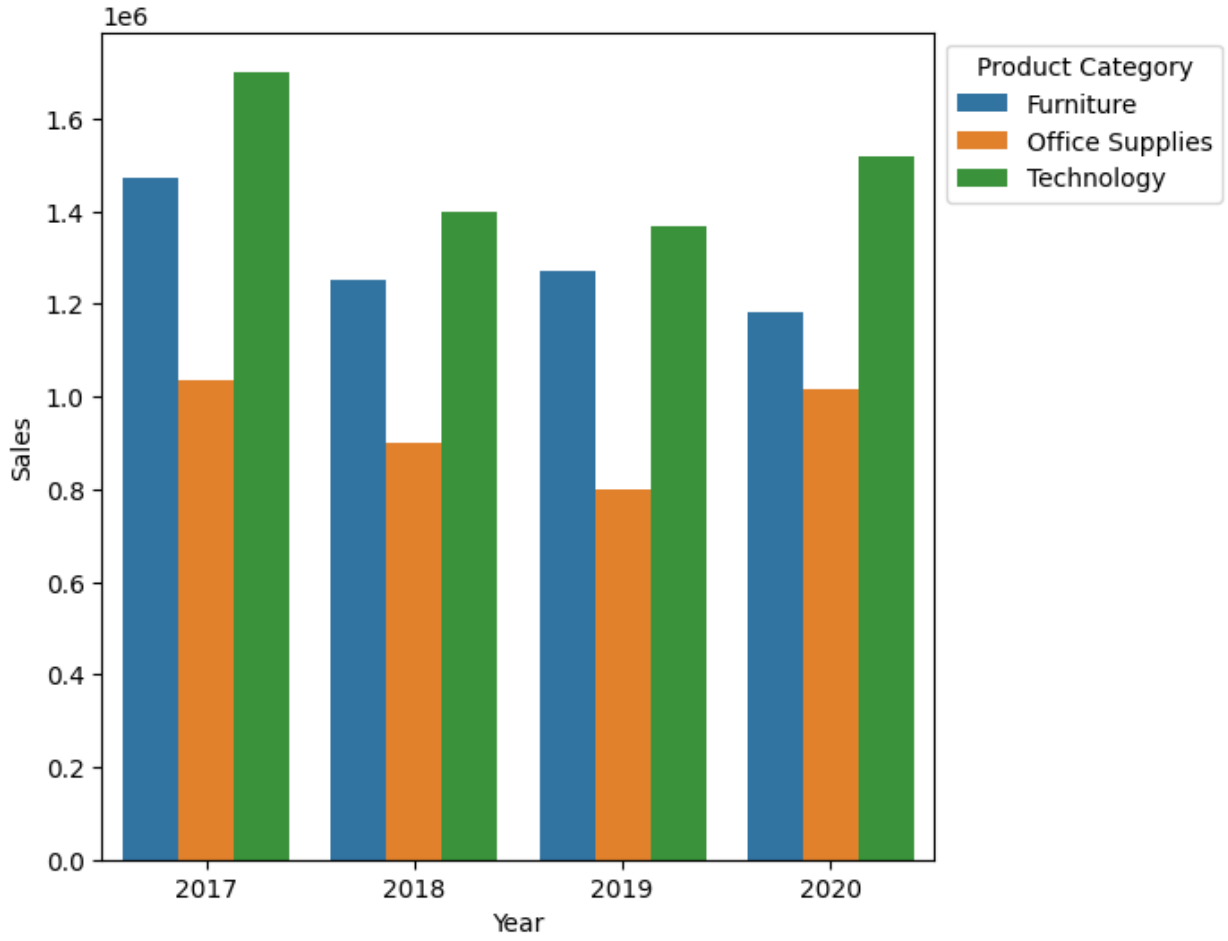


```
3- Product Category Contribution to Sales
category_sales = superstore_sales.groupby('Product Category')['Sales'].sum()
category_sales.plot(kind='bar', title='Total Sales by Product Category', figsize=(8, 6))
```

<Axes: title={'center': 'Total Sales by Product Category'}, xlabel='Product Category'>




```
The Sales of each Product category in every years
#The top Product category based on Sales
ProductCategorySalesOverYears = superstore_sales.groupby(['Year','Product
Category'])['Sales'].sum().reset_index()

plt.figure(figsize=(6,6))
sns.barplot(data = ProductCategorySalesOverYears, x ='Year', y='Sales',hue='Product Category')

plt.legend(title='Product Category', loc = 'best',bbox_to_anchor=(1,1))
plt.show()
```

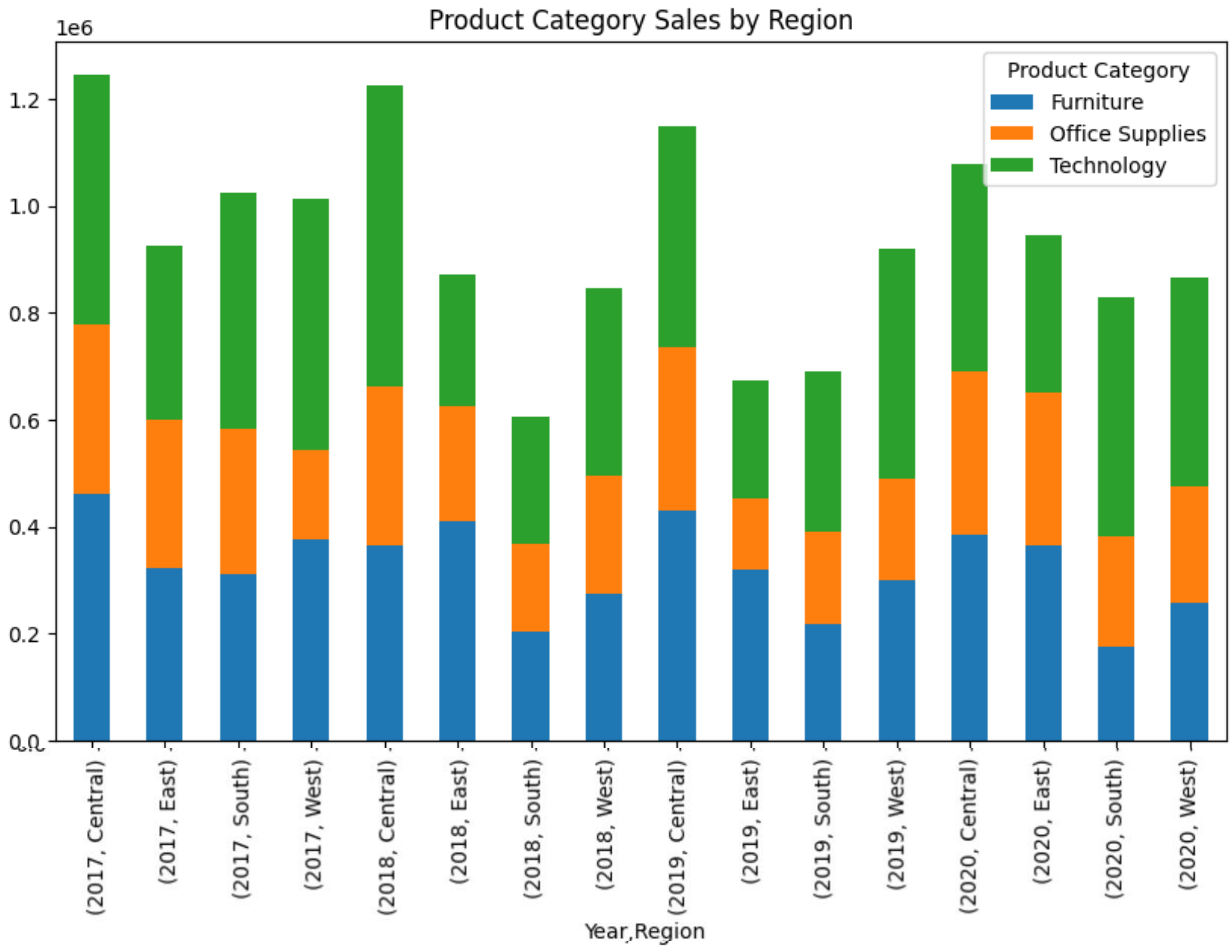


The top product sub-category over all years based on Sales is Telephones and communications and the lowest is rubber bands

The Sales of each Product category in every years and Region

```
region_category_sales = superstore_sales.groupby(['Year', 'Region', 'Product Category'])['Sales'].sum().unstack()
region_category_sales.plot(kind='bar', stacked=True, title='Product Category Sales by Region', figsize=(10, 6))
```

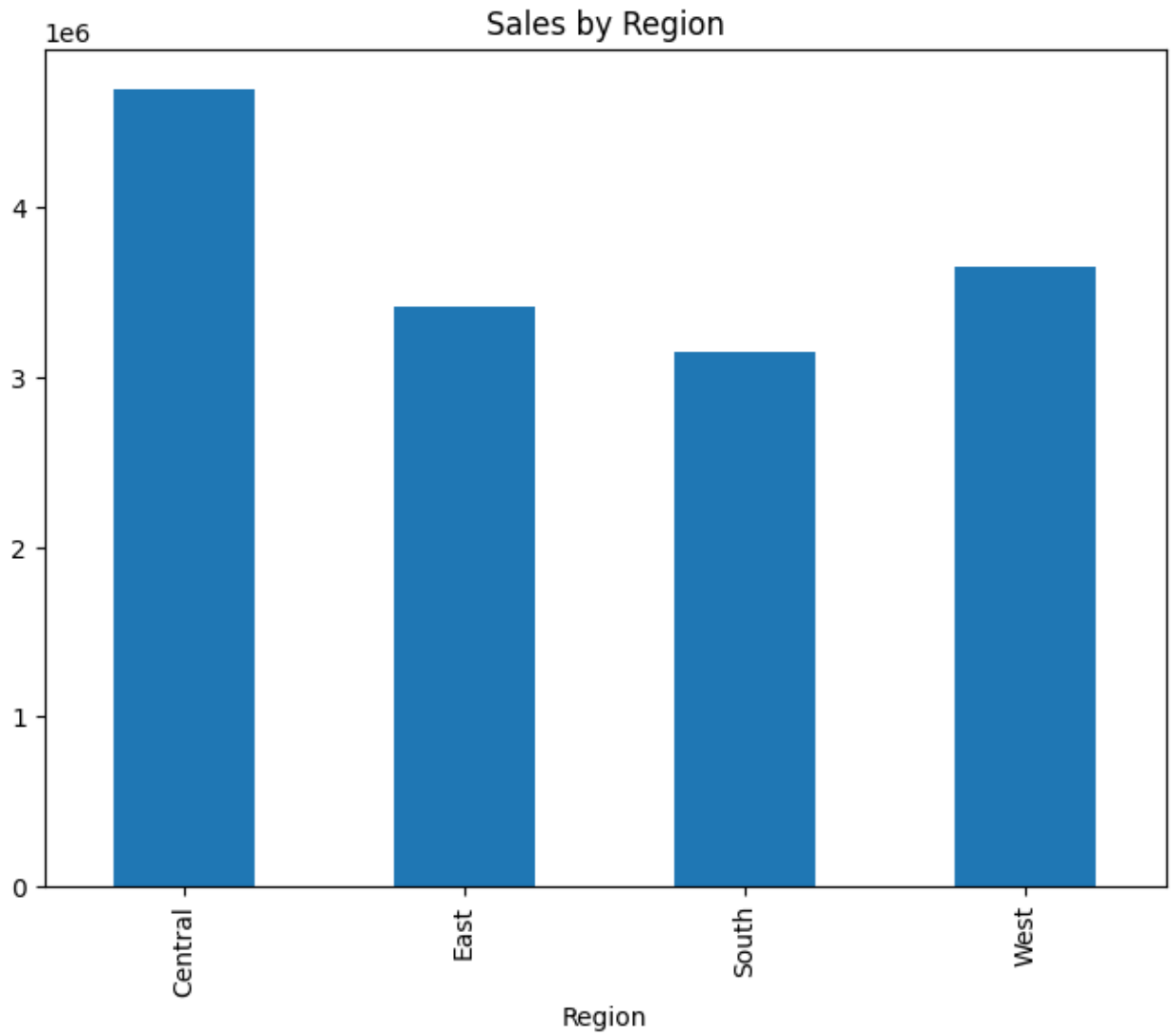
```
<Axes: title={'center': 'Product Category Sales by Region'}, xlabel='Year,Region'>
```



The Sales of each Product sub-category in every years and Region

```
region_sales = superstore_sales.groupby('Region')['Sales'].sum()
region_sales.plot(kind='bar', title='Sales by Region', figsize=(8, 6))
```

<Axes: title={'center': 'Sales by Region'}, xlabel='Region'>

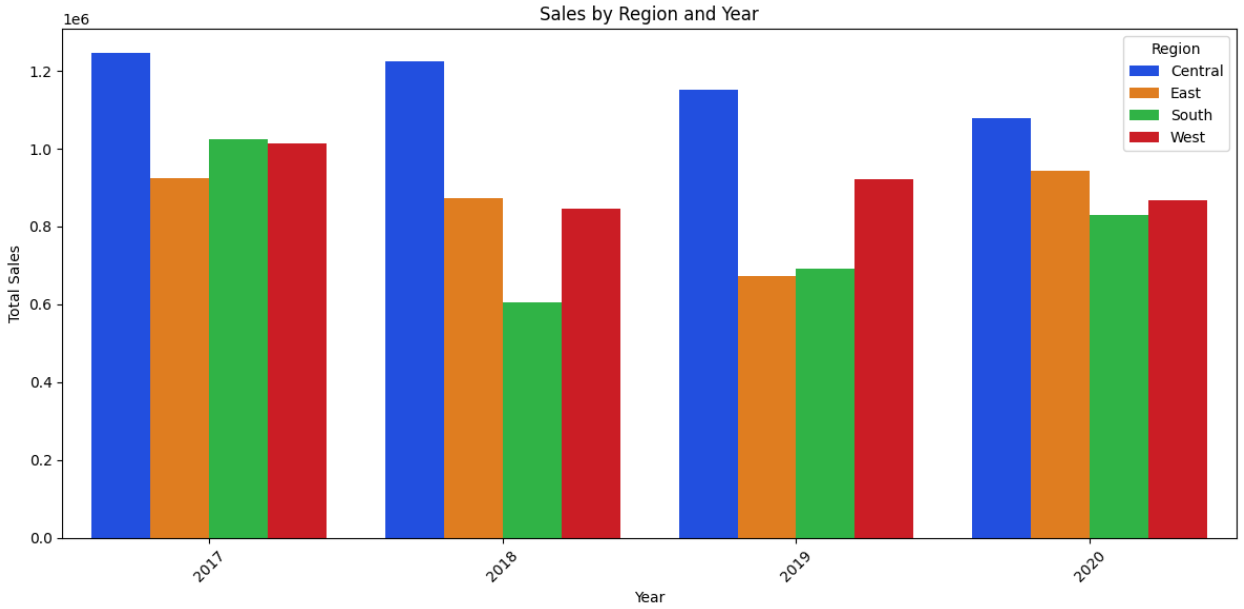



```
5- Slaes over both regoins and years
SalesByYearsAndRegion = superstore_sales.groupby(['Year','Region'])['Sales'].sum().reset_index()
print(SalesByYearsAndRegion)
```

```
   Year   Region        Sales
0  2017  Central  1.245107e+06
```

```
1   2017     East  9.255578e+05
2   2017    South  1.024364e+06
3   2017     West  1.014111e+06
4   2018  Central  1.225024e+06
5   2018     East  8.725294e+05
6   2018    South  6.049033e+05
7   2018     West  8.472239e+05
8   2019  Central  1.150174e+06
9   2019     East  6.740721e+05
10  2019    South  6.914181e+05
11  2019     West  9.211529e+05
12  2020  Central  1.078862e+06
13  2020     East  9.443072e+05
14  2020    South  8.295343e+05
15  2020     West  8.672603e+05
```

```
plt.clf()
plt.figure(figsize=(12, 6))
sns.set_palette('bright')
sns.barplot(data=SalesByYearsAndRegion, x='Year', y='Sales', hue='Region')
plt.title('Sales by Region and Year')
plt.xlabel('Year')
plt.ylabel('Total Sales')
plt.xticks(rotation=45)
plt.legend(title='Region')
plt.tight_layout()
plt.show()
```

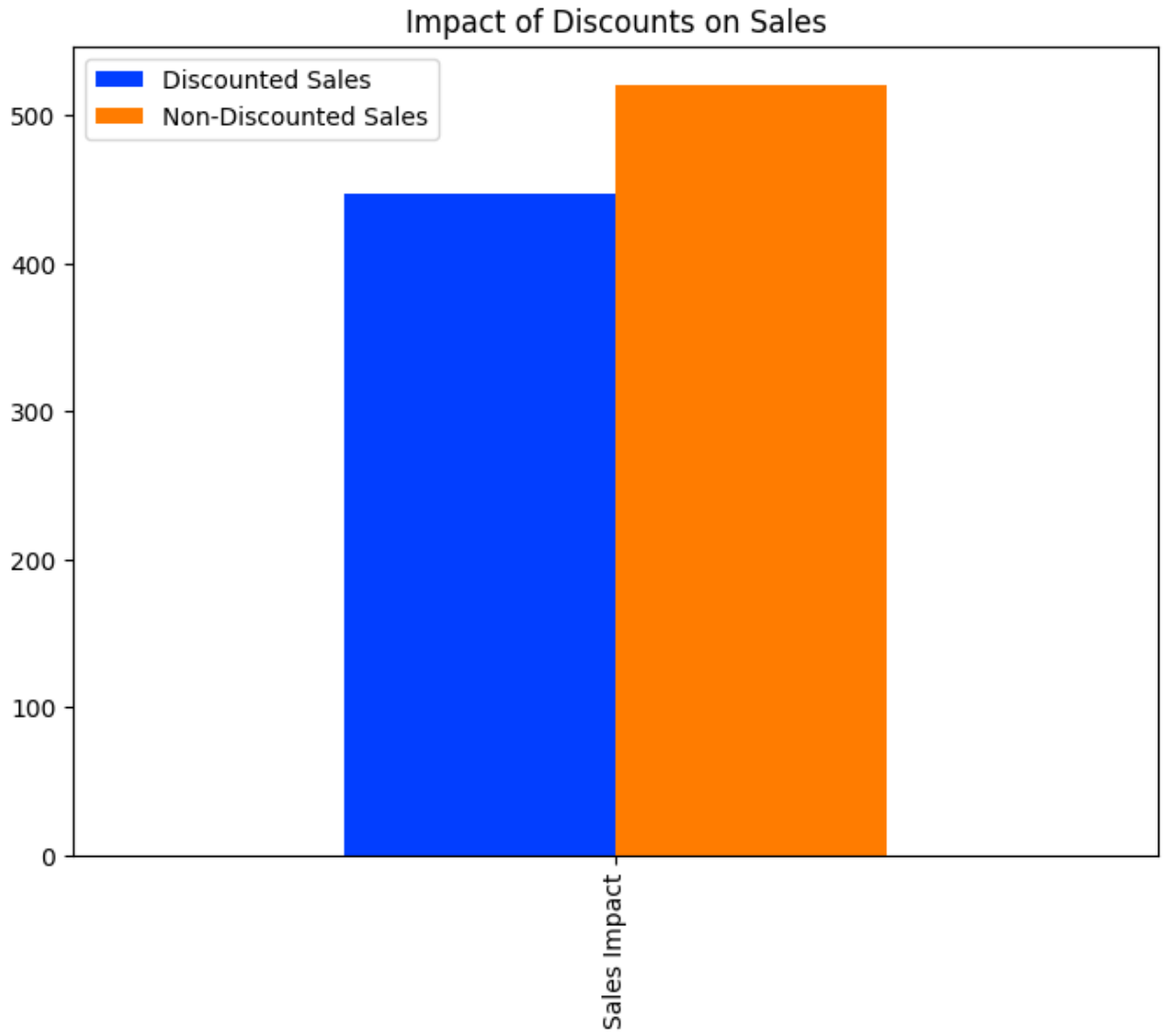



```
<Figure size 640x480 with 0 Axes>
```

```
Central had the hieghts Sales over years
6- Impact of Discounts on Sales
discount_sales = superstore_sales[superstore_sales['Discount'] > 0]
```

```
no_discount_sales = superstore_sales[superstore_sales['Discount'] == 0]

# Compare average sales with and without discounts
discount_impact = pd.DataFrame({
    'Discounted Sales': discount_sales['Sales'].median(),
    'Non-Discounted Sales': no_discount_sales['Sales'].median()
}, index=['Sales Impact'])

discount_impact.plot(kind='bar', title='Impact of Discounts on Sales', figsize=(8, 6))
```

<Axes: title={'center': 'Impact of Discounts on Sales'}>



```
correlation = superstore_sales['Discount'].corr(superstore_sales['Sales'])
print(f'Correlation between Discount and Profit: {correlation}')
```
Correlation between Discount and Profit: -0.019686336285171663

there is no correlation between Discount and Sales.

****2. Customer Behavior Questions:****

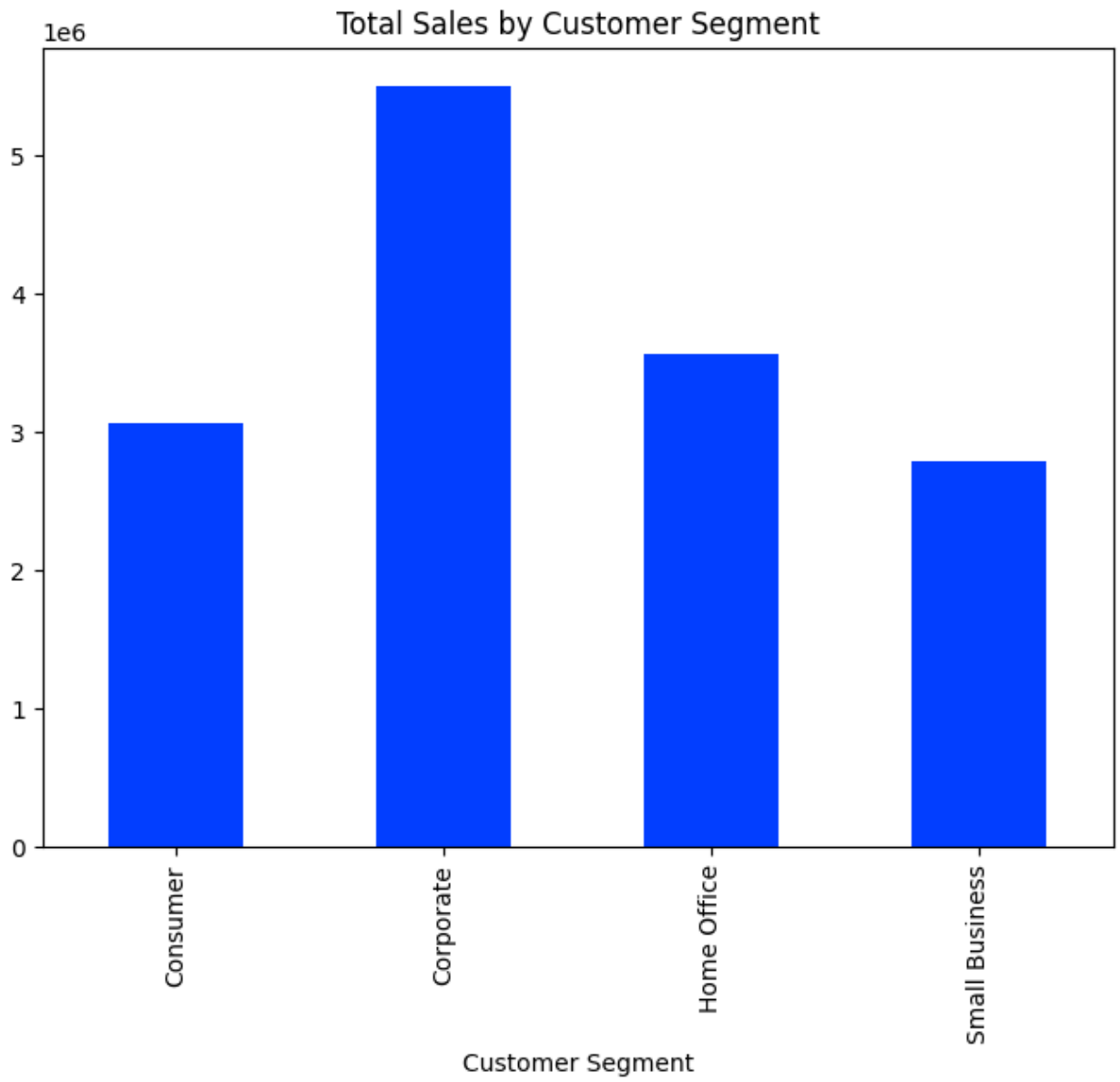
1- What are the top customer segments by total sales or frequency of purchases?

2- What are the customer retention rates across different regions?

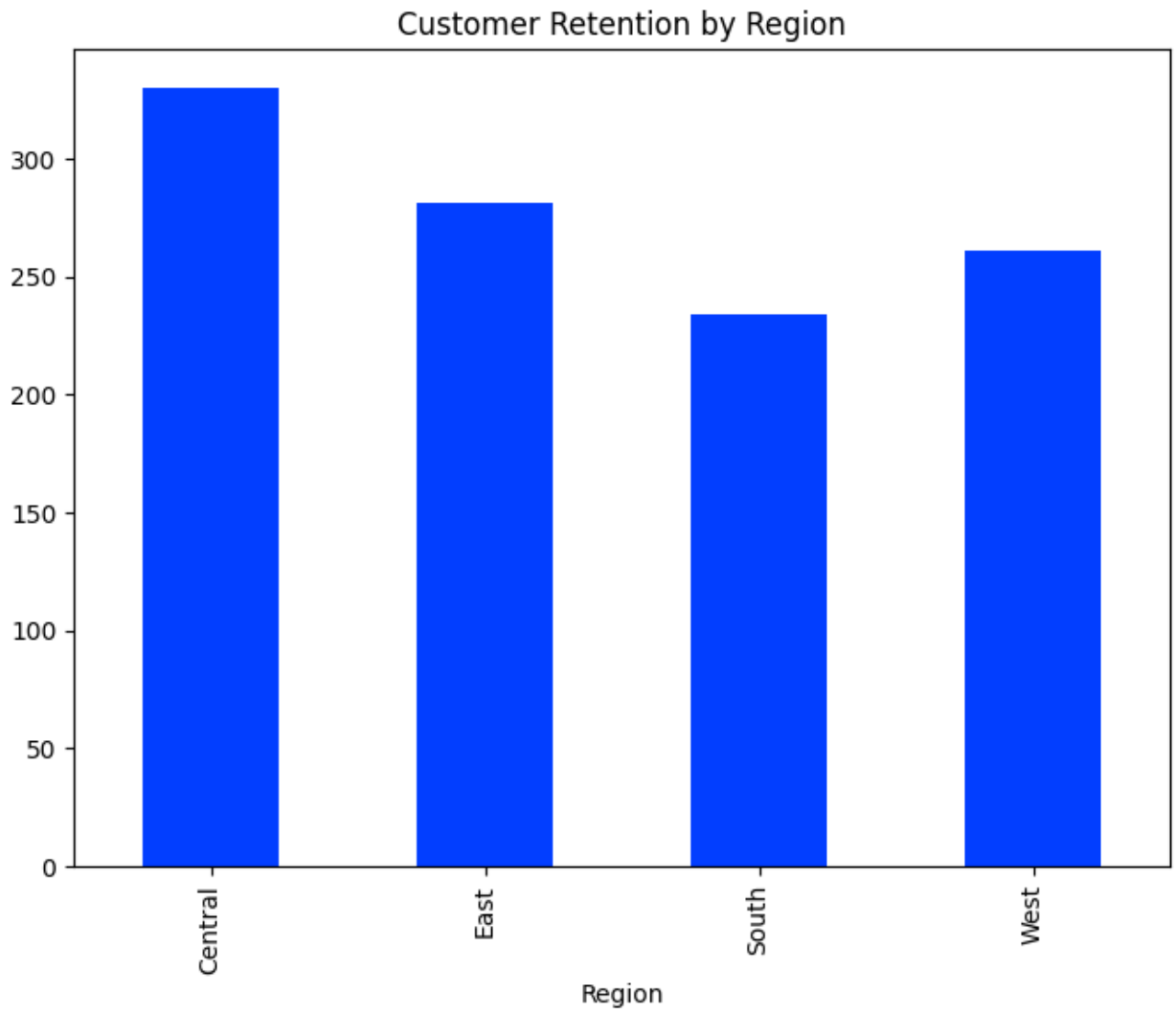
3- What is the estimated customer lifetime value across different segments?

4- Which customers are likely to stop purchasing based on purchase frequency?

```
1- Top Customer Segments by Total Sales
segment_sales = superstore_sales.groupby('Customer Segment')['Sales'].sum()
segment_sales.plot(kind='bar', title='Total Sales by Customer Segment', figsize=(8, 6))
```
<Axes: title={'center': 'Total Sales by Customer Segment'}, xlabel='Customer Segment'>




```
2- Customer Retention by Region
retention_by_region = superstore_sales.groupby('Region')['Customer Name'].nunique()
retention_by_region.plot(kind='bar', title='Customer Retention by Region', figsize=(8, 6))
```

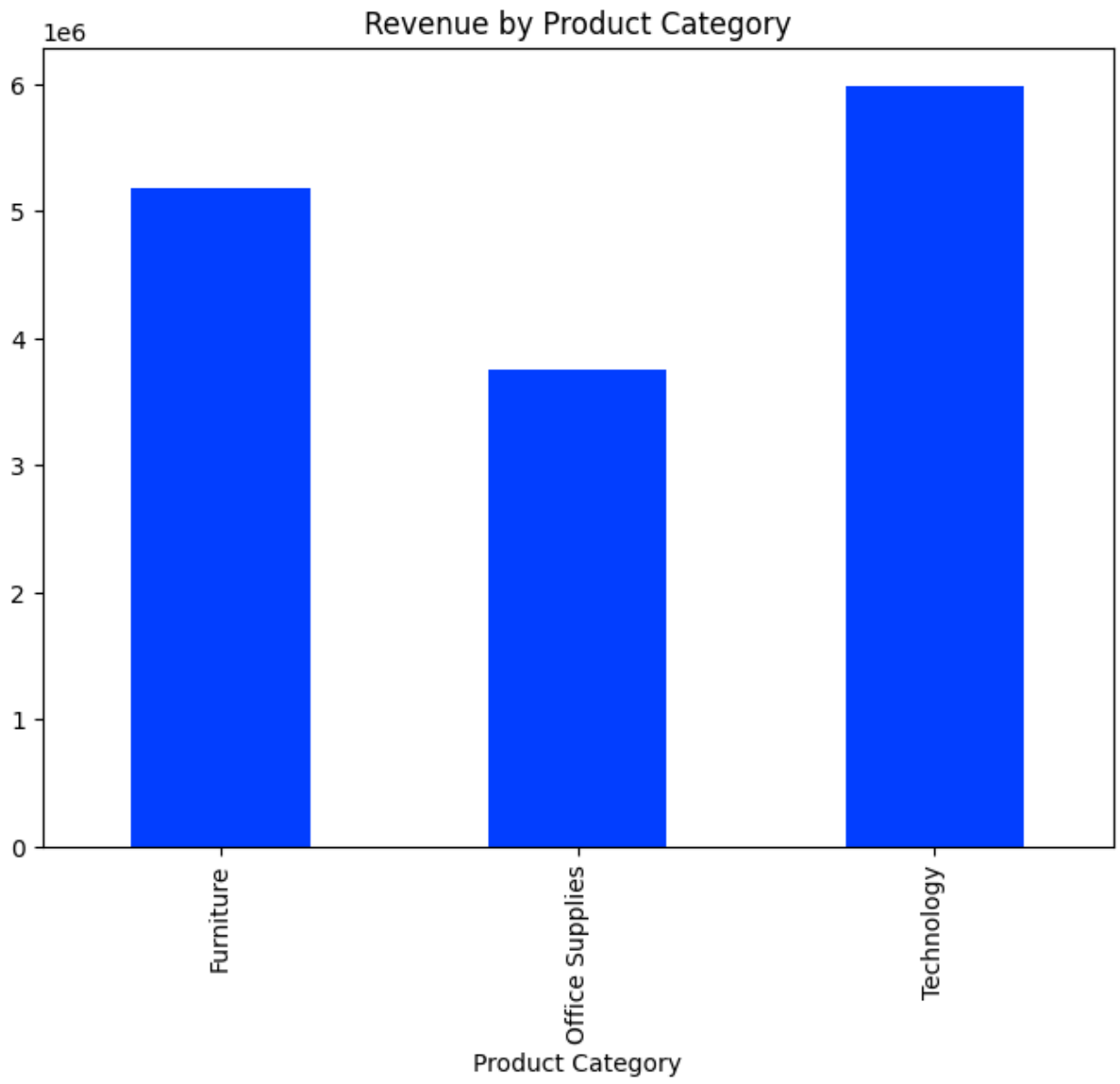



```
3- Estimated customer lifetime value across different segments
Calculate the total sales and number of purchases for each customer
customer_sales = superstore_sales.groupby('Customer Name')['Sales'].sum()
customer_freq = superstore_sales.groupby('Customer Name')['Order ID'].count()

# Estimate Customer Lifetime Value (CLV)
# Assuming a simple CLV calculation: CLV = Avg. Purchase Value * Purchase Frequency
avg_purchase_value = customer_sales / customer_freq
clv = avg_purchase_value * customer_freq

# Display the top 10 customers by CLV
top_customers_clv = clv.sort_values(ascending=False).head(10)
top_customers_clv
```

```
Customer Name
Emily Phan          117124.4380
Deborah Brumfield    97433.1355
Roy Skaria           92542.1530
```

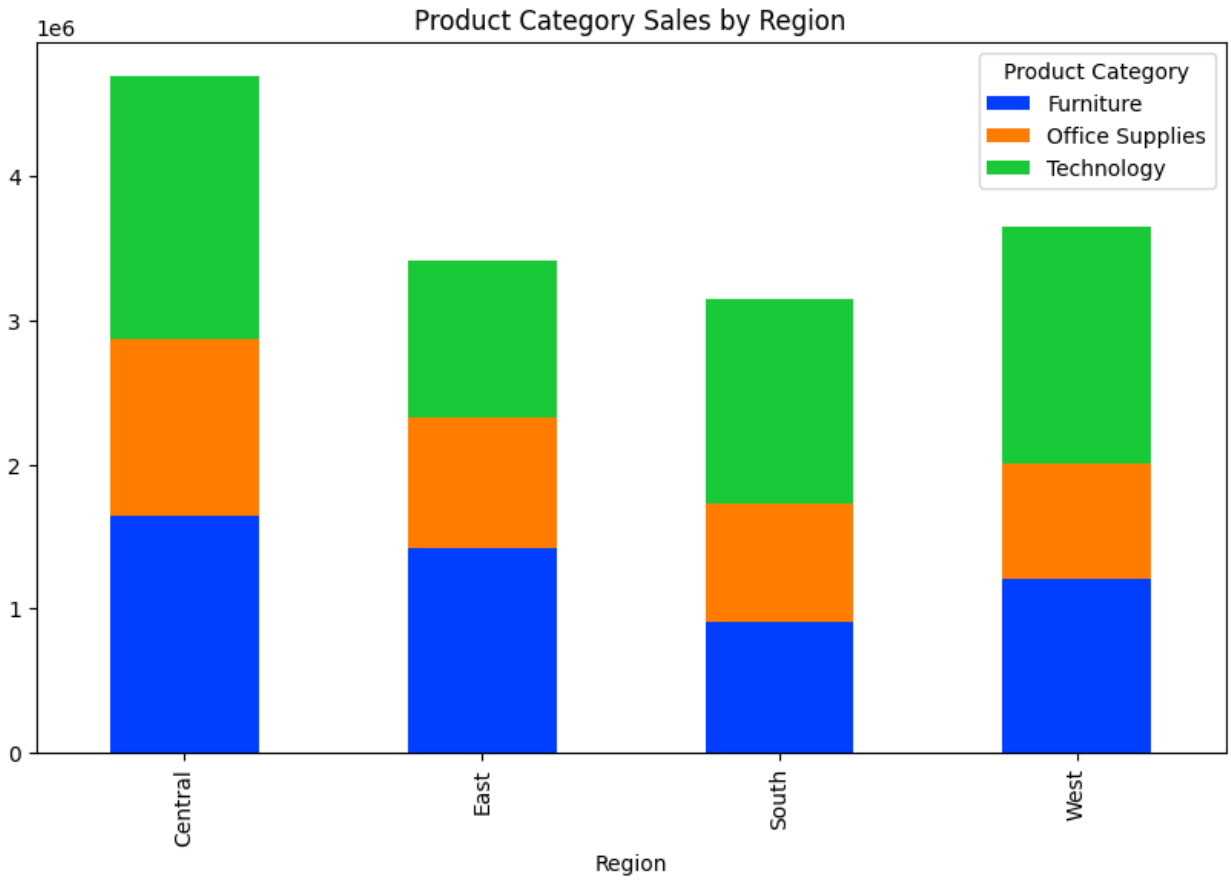
```
Sylvia Foulston      88875.7575
Grant Carroll        88417.0025
Alejandro Grove      83561.9300
Darren Budd          81577.3435
Julia Barnett        80044.4520
John Lucas           79696.1875
Liz MacKendrick      76306.4315
dtype: float64
```

```
4- Identify customers with long gaps between purchases
# Calculate the average days between orders for each customer
superstore_sales['Order Date'] = pd.to_datetime(superstore_sales['Order Date'])
customer_dates = superstore_sales.groupby('Customer Name')['Order Date'].agg(['min', 'max', 'count'])
customer_dates['days_between'] = (customer_dates['max'] - customer_dates['min']).dt.days /
customer_dates['count']

# Display customers with high average days between purchases
high_churn_risk = customer_dates[customer_dates['days_between'] >
customer_dates['days_between'].quantile(0.75)]
high_churn_risk
3. Product Category Analysis

Questions:

1- Which product categories generate the most revenue?

2- How do different product categories perform in different regions?

3- What are the stock levels across different product categories and locations?

4- How quickly are products selling out, and which categories have faster turnover?

5- What is the relationship between product price and sales volume?
1- Top Revenue Categories
category_revenue = superstore_sales.groupby('Product Category')['Sales'].sum()
category_revenue.plot(kind='bar', title='Revenue by Product Category', figsize=(8, 6))
<Axes: title={'center': 'Revenue by Product Category'}, xlabel='Product Category'>
```




```
2- Product Category Performance by Region
region_category_sales = superstore_sales.groupby(['Region', 'Product Category'])['Sales'].sum().unstack()
region_category_sales.plot(kind='bar', stacked=True, title='Product Category Sales by Region', figsize=(10, 6))
```

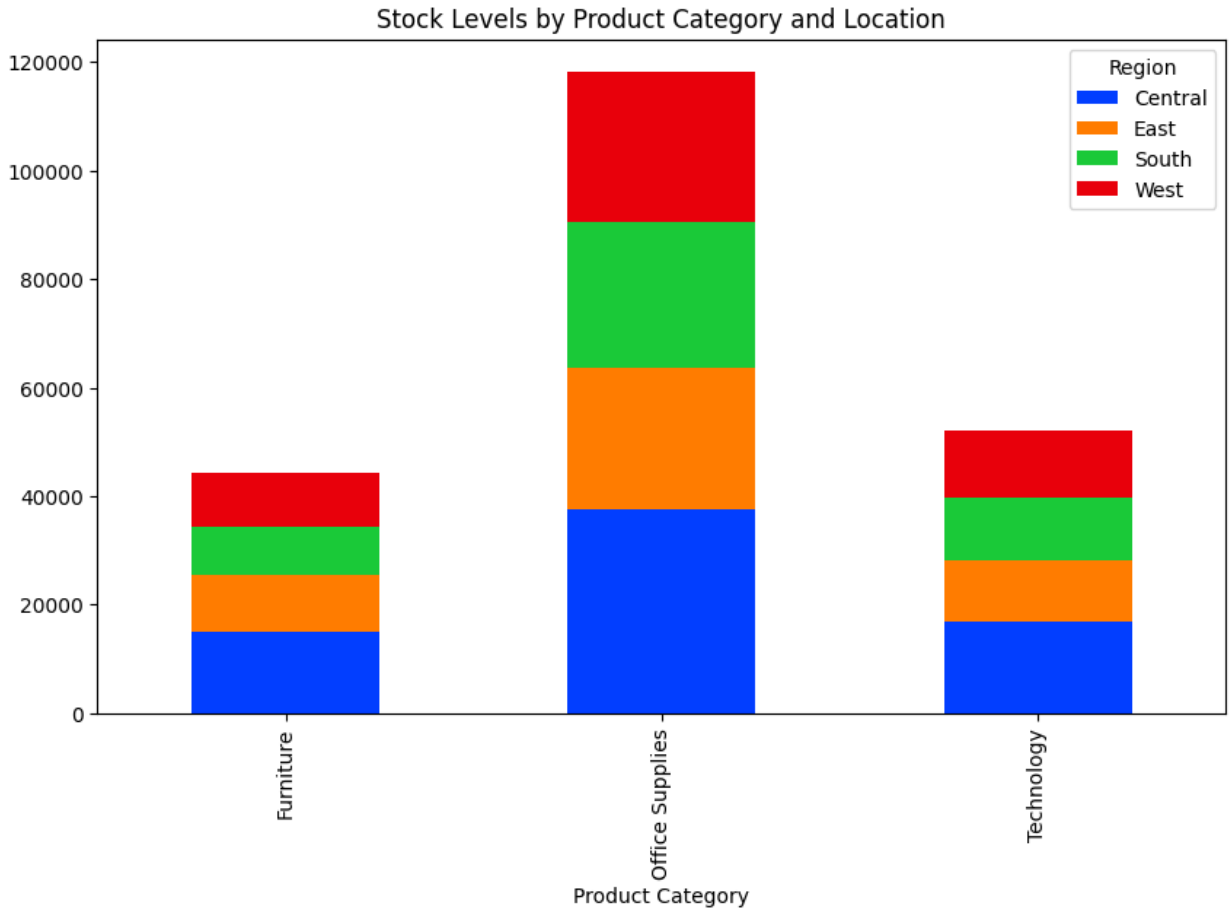
```
<Axes: title={'center': 'Product Category Sales by Region'}, xlabel='Region'>
```




```
3- Stock levels across different product categories and locations
stock_levels = superstore_sales.groupby(['Product Category', 'Region'])['Order Quantity'].sum().unstack()
stock_levels.plot(kind='bar', stacked=True, title='Stock Levels by Product Category and Location', figsize=(10, 6))
```

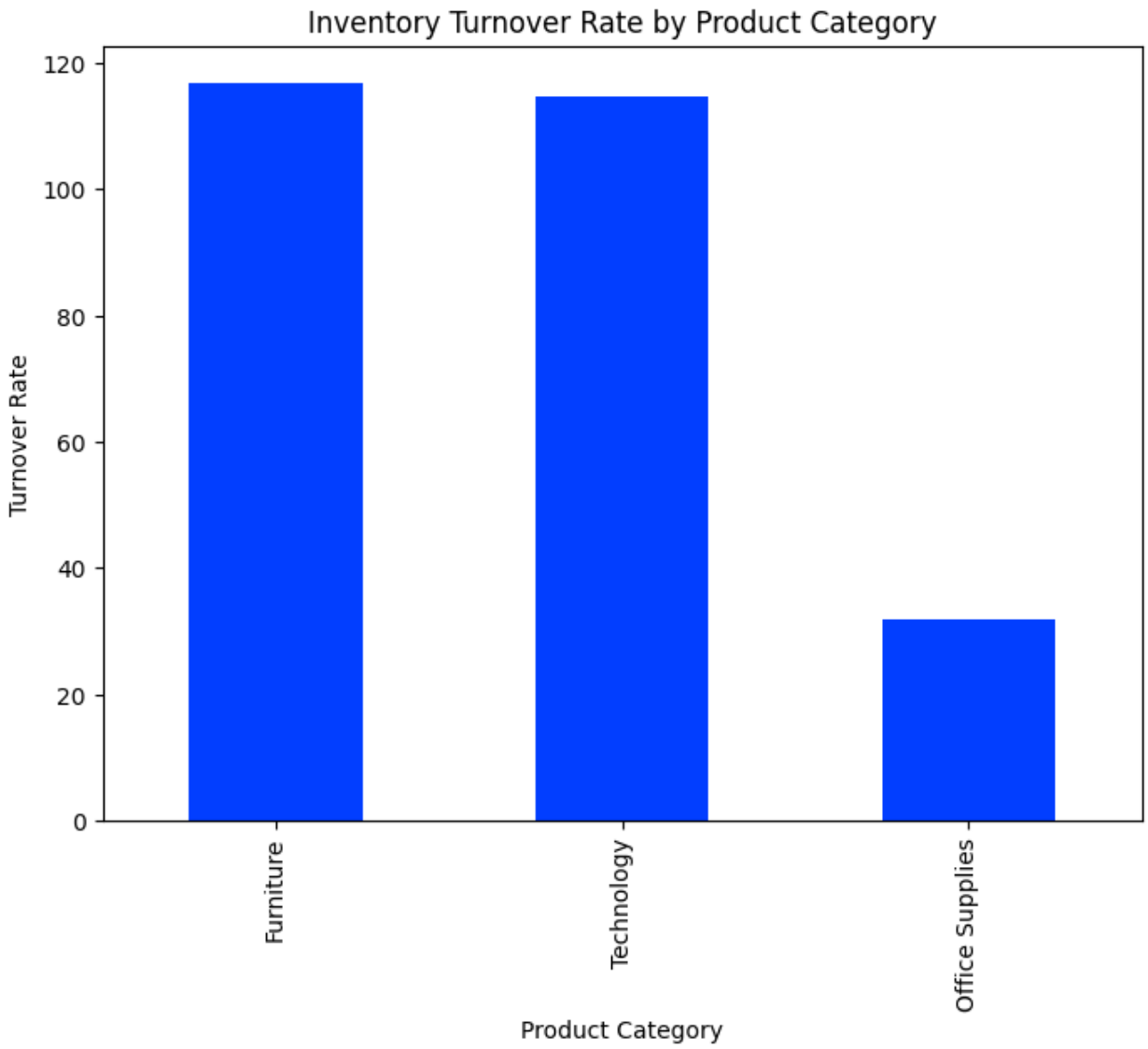
```
<Axes: title={'center': 'Stock Levels by Product Category and Location'}, xlabel='Product Category'>
```

```
4- How quickly are products selling out, and which categories have faster turnover
```




```
4- How quickly are products selling out, and which categories have faster turnover
# Calculate turnover as the ratio of total sales to total order quantity for each product category
category_sales_qty = superstore_sales.groupby('Product Category')[['Sales', 'Order Quantity']].sum()
category_sales_qty['Turnover Rate'] = category_sales_qty['Sales'] / category_sales_qty['Order Quantity']

# Plot turnover rate by product category
category_sales_qty['Turnover Rate'].sort_values(ascending=False).plot(kind='bar', title='Inventory Turnover Rate by Product Category', xlabel='Product Category', ylabel='Turnover Rate', figsize=(8, 6))
```

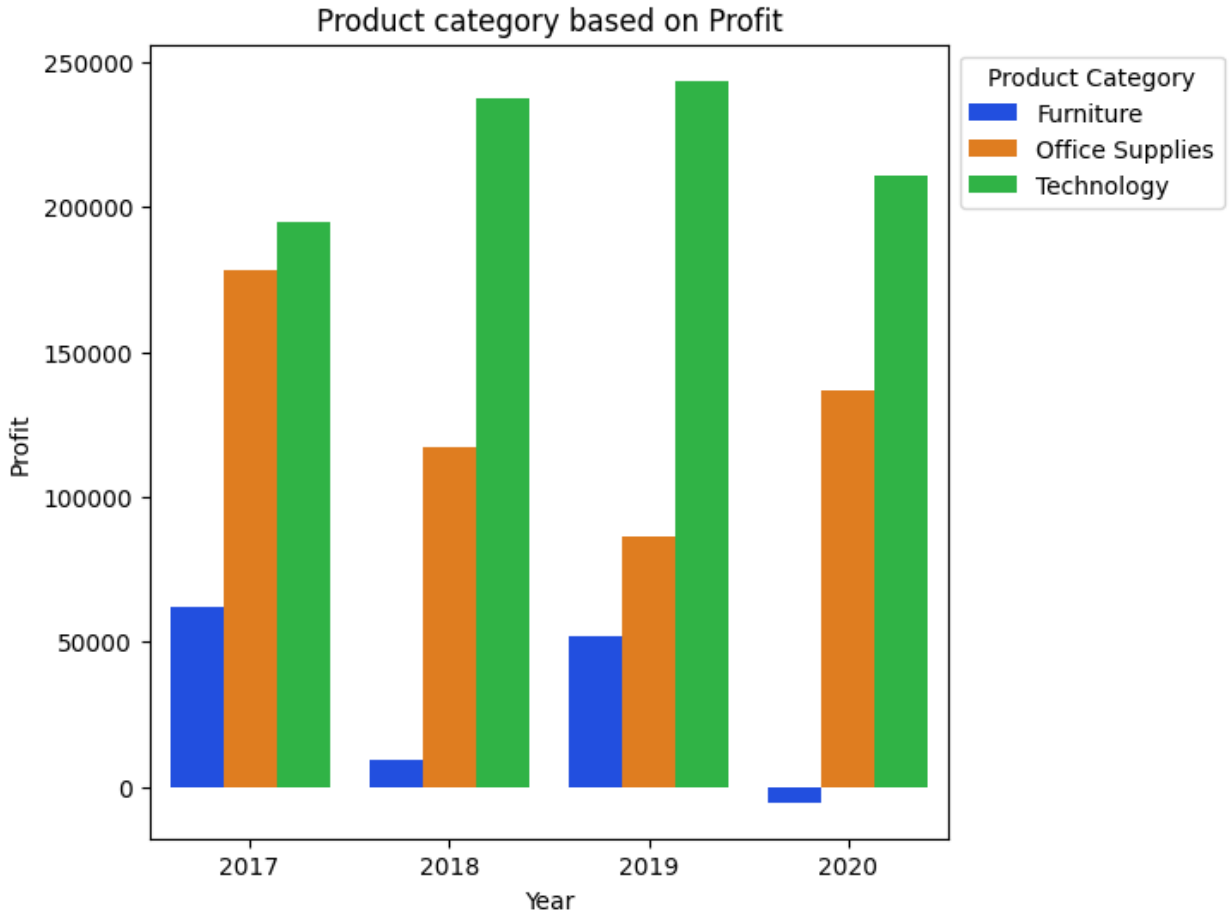
<Axes: title={'center': 'Inventory Turnover Rate by Product Category'}, xlabel='Product Category', ylabel='Turnover Rate'>



The Profit of each Product category in every years

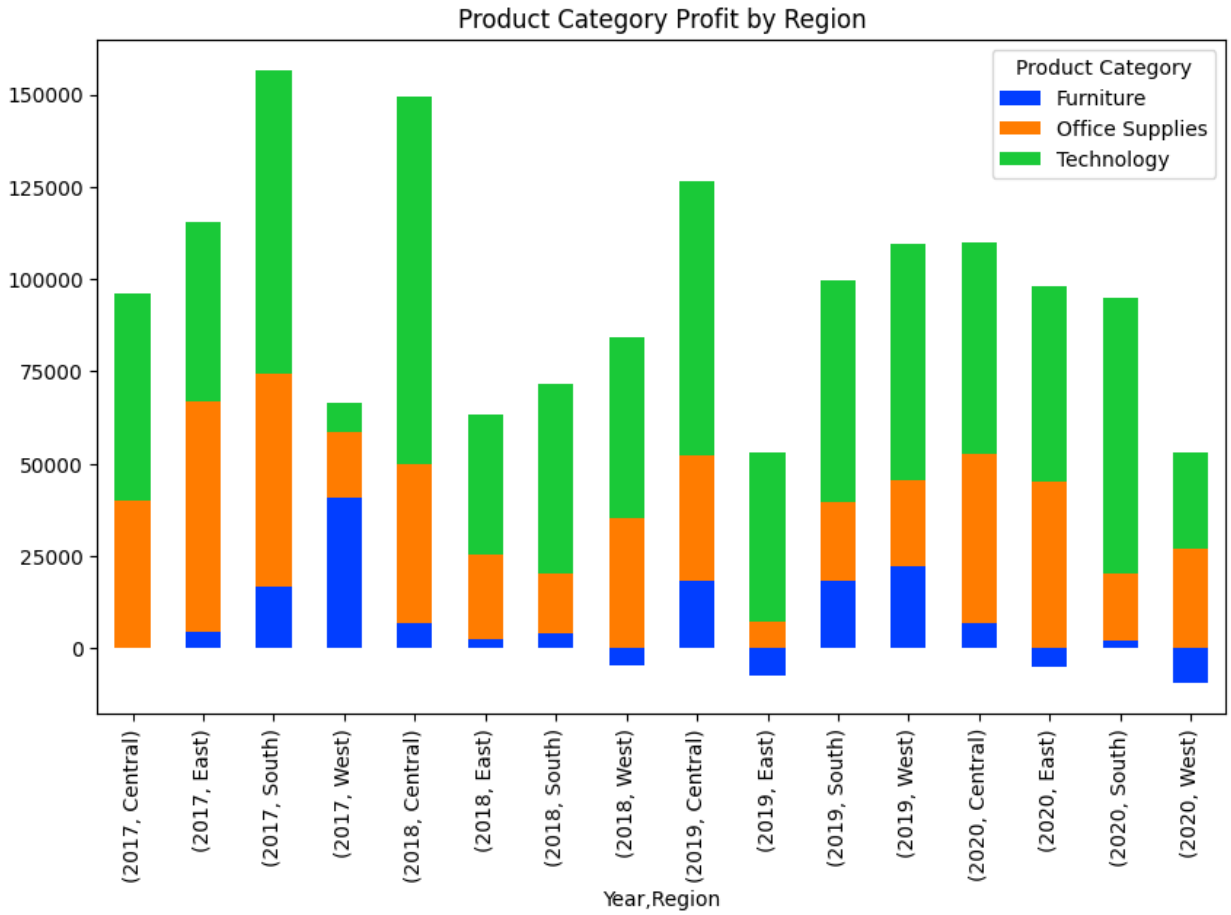
```
ProductCategoryProfitOverYears = superstore_sales.groupby(['Year','Product Category'])['Profit'].sum().reset_index()
```

```
plt.figure(figsize=(6,6))
sns.barplot(data = ProductCategoryProfitOverYears, x ='Year', y='Profit',hue='Product Category')
plt.title('Product category based on Profit')
plt.legend(title='Product Category', loc = 'best',bbox_to_anchor=(1,1))
plt.show()
```




```
The Profit of each Product category in every years and Region
region_category_sales = superstore_sales.groupby(['Year', 'Region', 'Product Category'])['Profit'].sum().unstack()
region_category_sales.plot(kind='bar', stacked=True, title='Product Category Profit by Region', figsize=(10, 6))
```

<Axes: title={'center': 'Product Category Profit by Region'}, xlabel='Year,Region'>



The Profit of each Product sub-category in every years and Region

```
# Grouping data by year and product sub-category to calculate total Profit
ProductSalesOverYears = superstore_sales.groupby(['Year', 'Product Sub-
Category'])['Profit'].sum().reset_index()

# Creating an interactive bar chart
fig = px.bar(ProductSalesOverYears,
             x='Year',
             y='Profit',
             color='Product Sub-Category',
             title='Top Product based on Profit',
             labels={'Year': 'Year', 'Profit': 'Total Profit'},
             barmode='group',  # Shows bars for each category side by side
             height=600,  # Adjust the height of the figure
             width=800)  # Adjust the width of the figure

# Update the layout to position the legend outside the plot
fig.update_layout(legend=dict(
    title='Product Sub-Category',
    x=1,  # Position it outside the plot area on the x-axis
```

```
    y=1,
    traceorder='normal'
))

# Show the interactive plot
fig.show()
```

The top product sub-category over all years based on Profit is `Telephones` and communications and `Bookcases` the product sub-category that cuses the most loss in the first year While `Tables` is cause for loses in the following 3 years
The top product sub-category over all years based on Profit is `Telephones` and communications and `Bookcases` the product sub-category that cuses the most loss in the first year While `Tables` is cause for loses in the following 3 years
The top product sub-category over all years based on Profit is `Telephones` and communications and `Bookcases` the product sub-category that cuses the most loss in the first year While `Tables` is cause for loses in the following 3 years

```
superstore_sales.plot(kind='scatter', x='Unit Price', y='Sales', title='Price vs Sales Volume', figsize=(8, 6))
```

<Axes: title={'center': 'Price vs Sales Volume'}, xlabel='Unit Price', ylabel='Sales'>

4. Profitability

Questions:

1- Which regions or product categories have the highest profit margins?

2- How do shipping costs affect profitability across different regions?

3- Are there any specific customer segments that contribute to higher profitability?

4- the relation between profit and discount

5- profit over years

6- Profit over both regoins and years

7- The highest and the lowest five product based on profit

8- The relation between shipping mode and profit

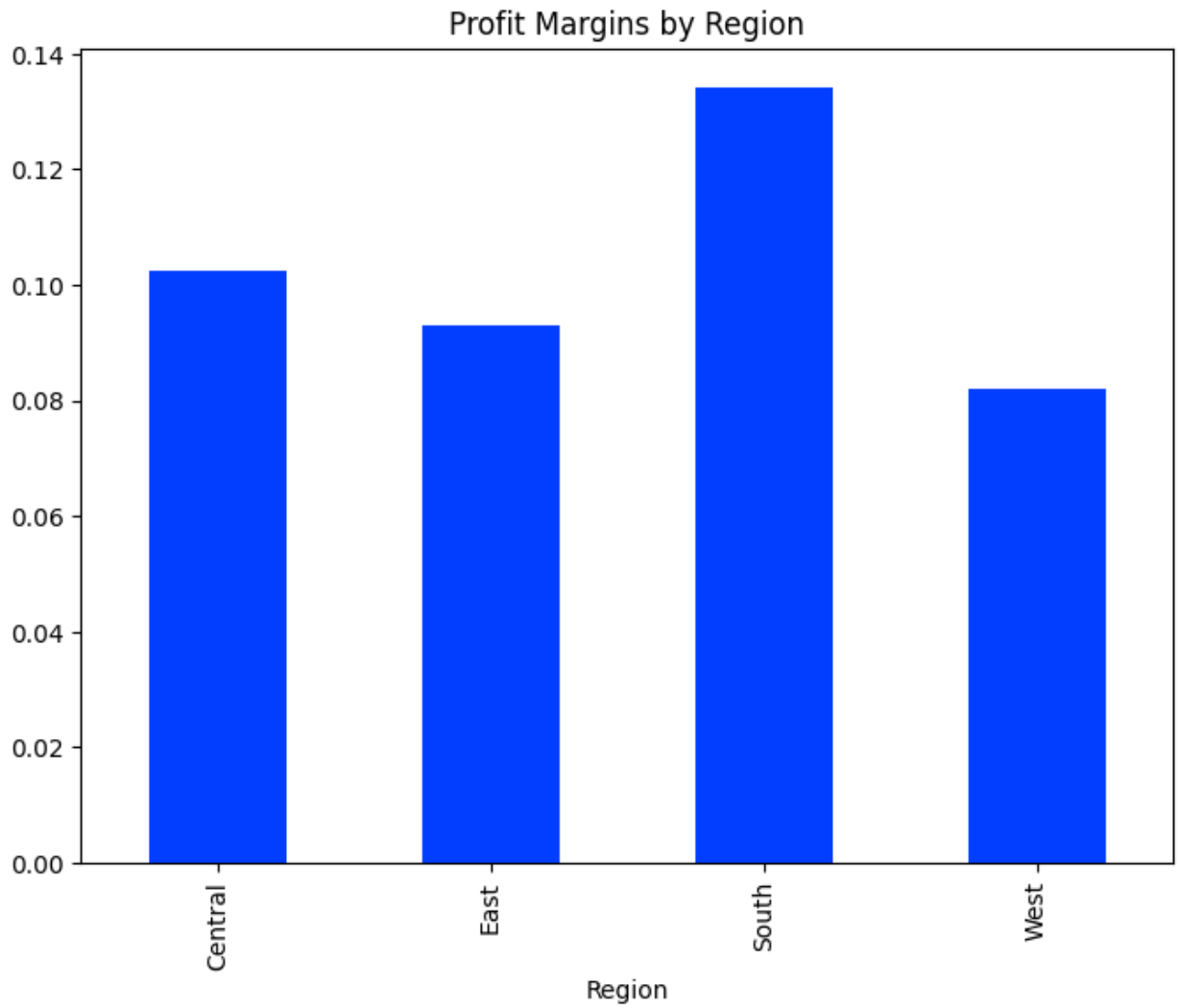
9- the relation between Unit price and Profit

10- The affect of container size on Shipment Cost and Profit Amount

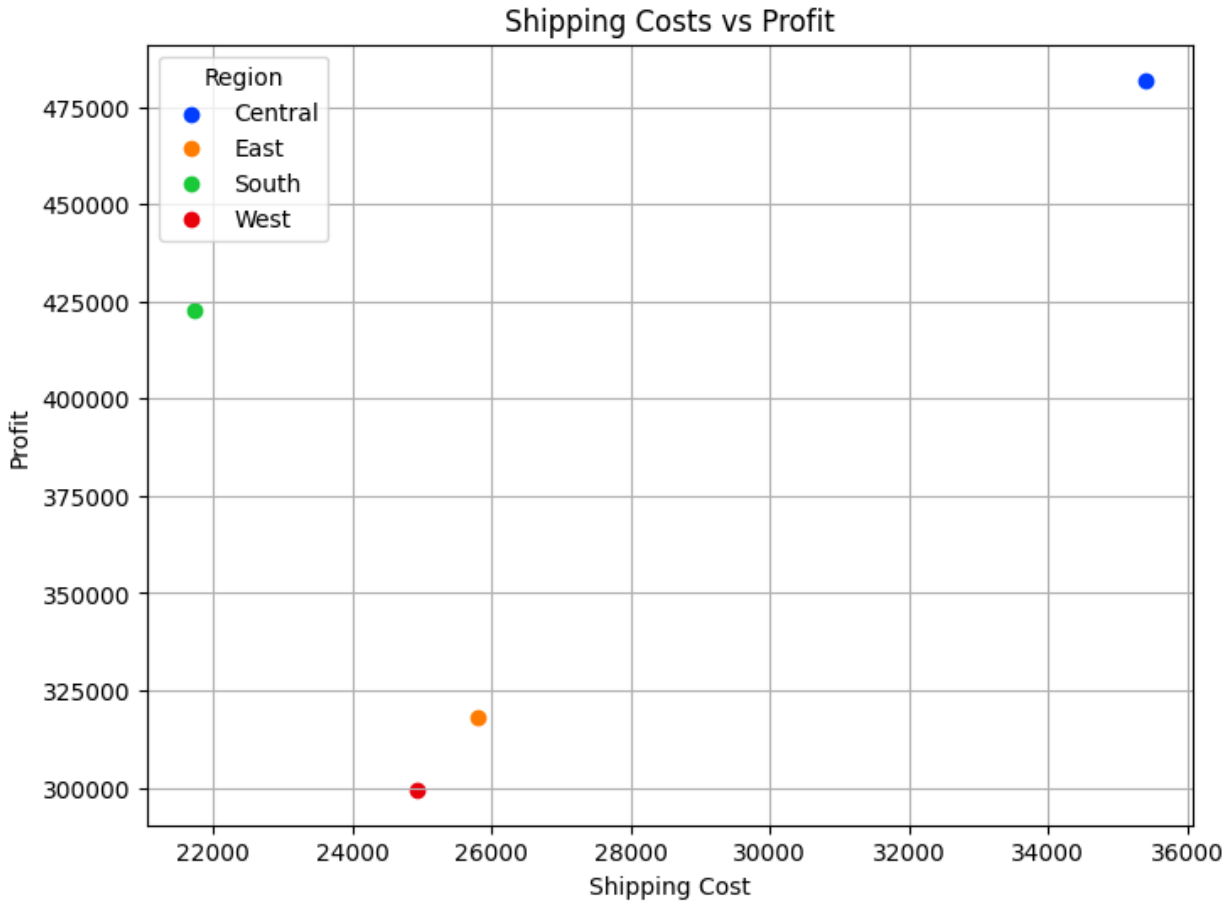
11- who is the top and low five customer based on sales and profit that the company may target to improve preformance

```
1- Profit Margins by Region
profit_margin_by_region = superstore_sales.groupby('Region')['Profit'].sum() / superstore_sales.groupby('Region')['Sales'].sum()
profit_margin_by_region.plot(kind='bar', title='Profit Margins by Region', figsize=(8, 6))
```

```
<Axes: title={'center': 'Profit Margins by Region'}, xlabel='Region'>
```

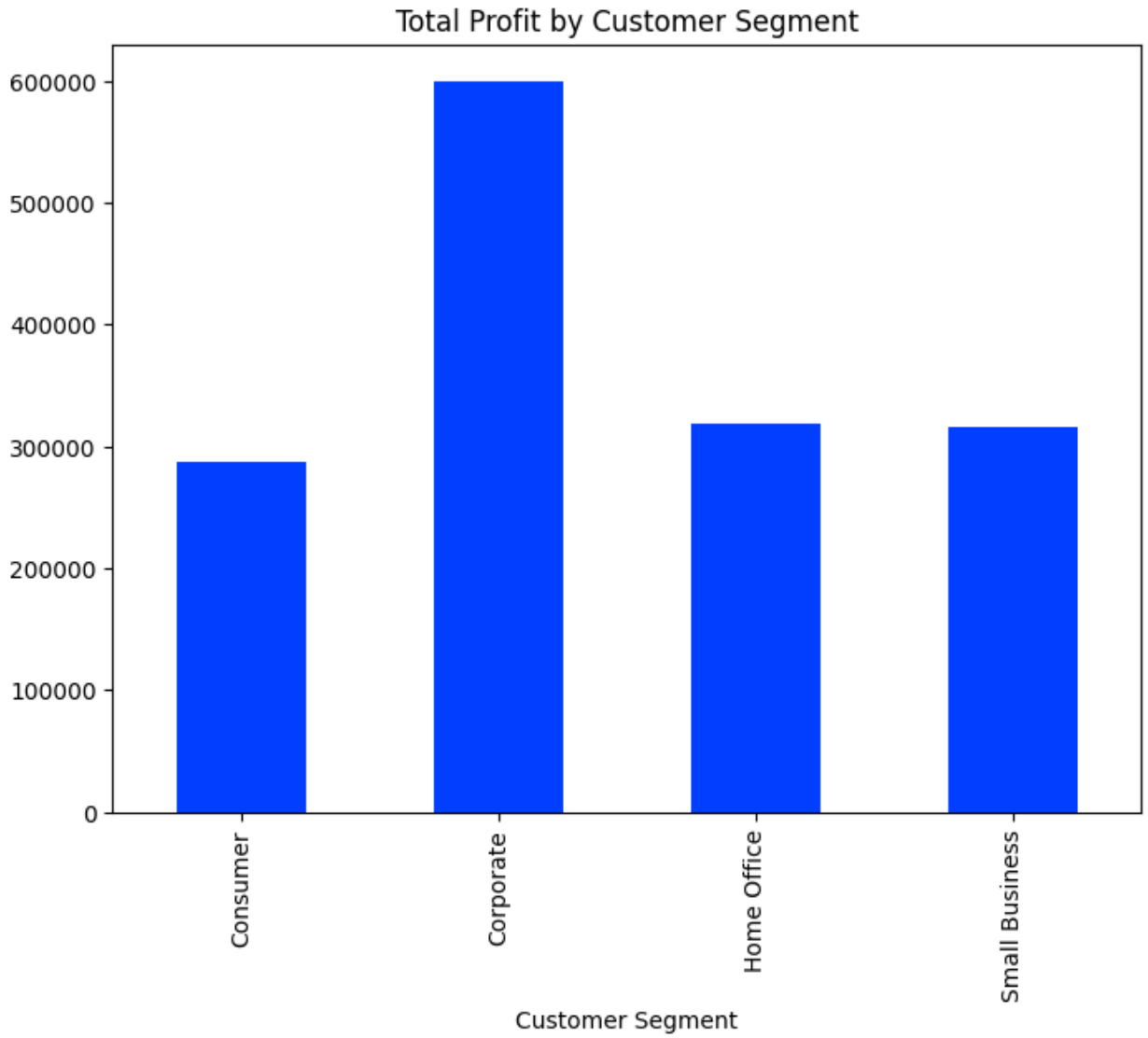



```
2- Impact of Shipping Costs on Profitability
shipping_profit = superstore_sales.groupby('Region')[['Shipping Cost', 'Profit']].sum().reset_index()

plt.figure(figsize=(8, 6))

for region in shipping_profit['Region'].unique():
    subset = shipping_profit[shipping_profit['Region'] == region]
    plt.scatter(subset['Shipping Cost'], subset['Profit'], label=region)

plt.title('Shipping Costs vs Profit')
plt.xlabel('Shipping Cost')
plt.ylabel('Profit')
plt.legend(title='Region')
plt.grid()
plt.show()
```




```
3- Calculate Profit by Customer Segment
Calculate total profit by customer segment
segment_profit = superstore_sales.groupby('Customer Segment')['Profit'].sum()

# Plot total profit by customer segment
segment_profit.plot(kind='bar', title='Total Profit by Customer Segment', figsize=(8, 6))
```

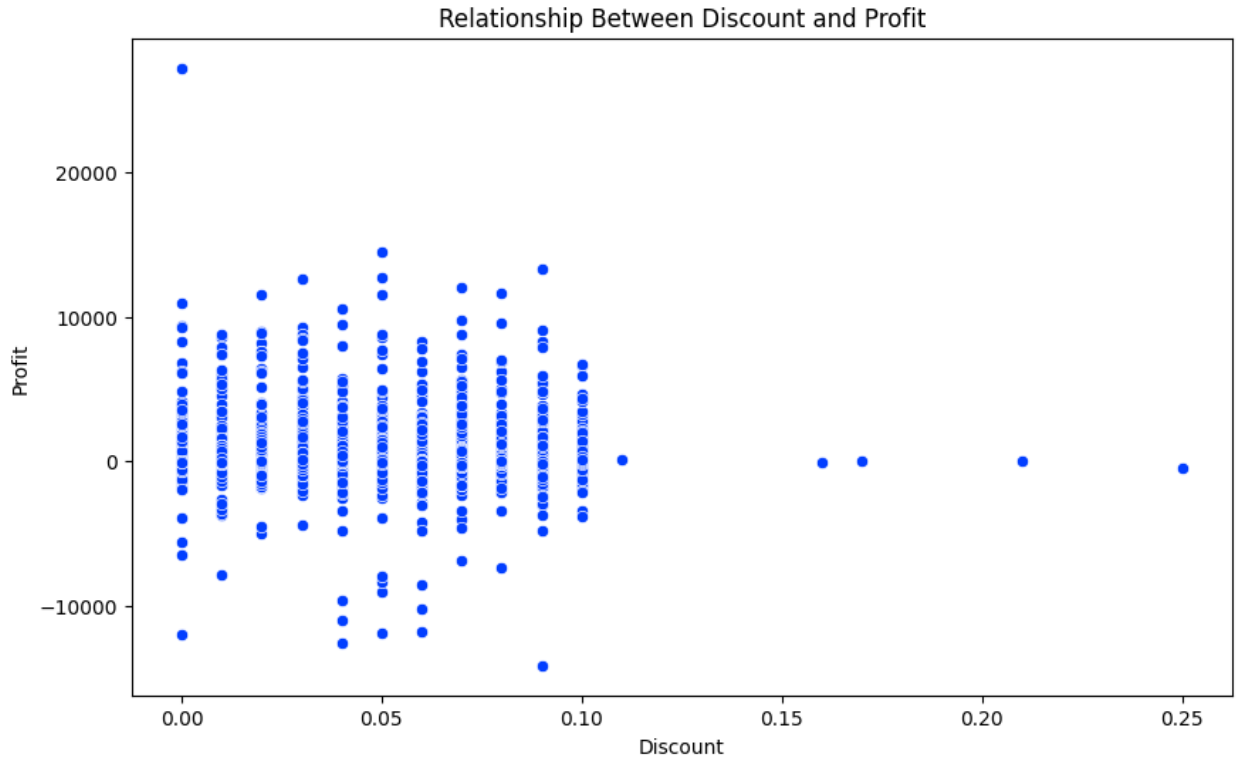
<Axes: title={'center': 'Total Profit by Customer Segment'}, xlabel='Customer Segment'>



4- the relation between profit and discount

```
# Relationship Between Discount and Profit

plt.clf()
plt.figure(figsize=(10,6))
sns.scatterplot(x='Discount', y='Profit', data = superstore_sales)
plt.title('Relationship Between Discount and Profit')
plt.xlabel('Discount')
plt.ylabel('Profit')
plt.show()
```

<Figure size 640x480 with 0 Axes>




```
the corelation between Profit and discount
correlation = superstore_sales['Discount'].corr(superstore_sales['Profit'])
print(f'Correlation between Discount and Profit: {correlation}')
```

Correlation between Discount and Profit: -0.037128373576373545

```
there is no correlation between Discount and Profit.
5- profit over years
superstore_sales['Year'] = superstore_sales['Order Date'].dt.year
profitOverYears = superstore_sales.groupby('Year')['Profit'].sum().reset_index()
print(profitOverYears)
```

```
   Year     Profit
0  2017  434538.73
1  2018  363871.48
2  2019  381455.99
3  2020  341901.78
```

```
profitOverYears['Change'] = profitOverYears['Profit'].diff()

# Print profit changes for analysis
print(profitOverYears)
```

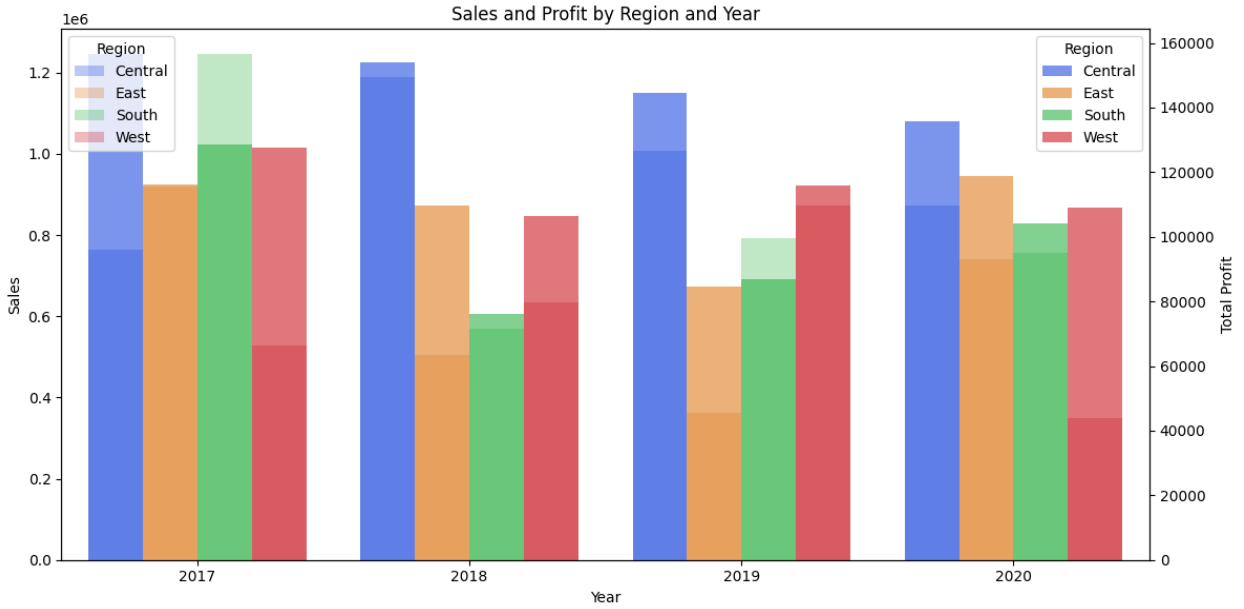
```
   Year     Profit    Change
0  2017  434538.73       NaN
1  2018  363871.48 -70667.25
2  2019  381455.99  17584.51
3  2020  341901.78 -39554.21
```

```
 sns.barplot(data=profitOverYears ,x= superstore_sales['year'] , y = superstore_sales['Profit'])
# plt.show()

# Create a bar plot
plt.clf()
plt.figure(figsize=(10, 5))
sns.barplot(data=profitOverYears, x='Year', y='Profit', palette='viridis')
plt.title('Total Profit Over Years')
plt.xlabel('Year')
plt.ylabel('Total Profit')
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
plt.tight_layout()
plt.show()
```

<Figure size 640x480 with 0 Axes>


```
2020 has the most profit and 2018 has the lowest
comparison bewteen Sales and Profit by Region and Year in every regoin
# Grouping the data by year and Region, summing Sales and Profit
SalesAndProfitByYearsAndRegion = superstore_sales.groupby(['Year', 'Region'])[['Sales',
'Profit']].sum().reset_index()

# Plotting Sales and Profit together for comparison
plt.clf()
plt.figure(figsize=(12, 6))
# Creating the bar plot for Sales
sns.barplot(data=SalesAndProfitByYearsAndRegion, x='Year', y='Sales', hue='Region', alpha=0.6)

# Creating a second y-axis for Profit
```

```
ax2 = plt.twinx()
sns.barplot(data=SalesAndProfitByYearsAndRegion, x='Year', y='Profit', hue='Region', ax=ax2, alpha=0.3)

# Setting titles and labels
plt.title('Sales and Profit by Region and Year')
plt.xlabel('Year')
plt.ylabel('Total Sales')
ax2.set_ylabel('Total Profit')

# Show legend
plt.legend(title='Region', loc='upper left')

plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

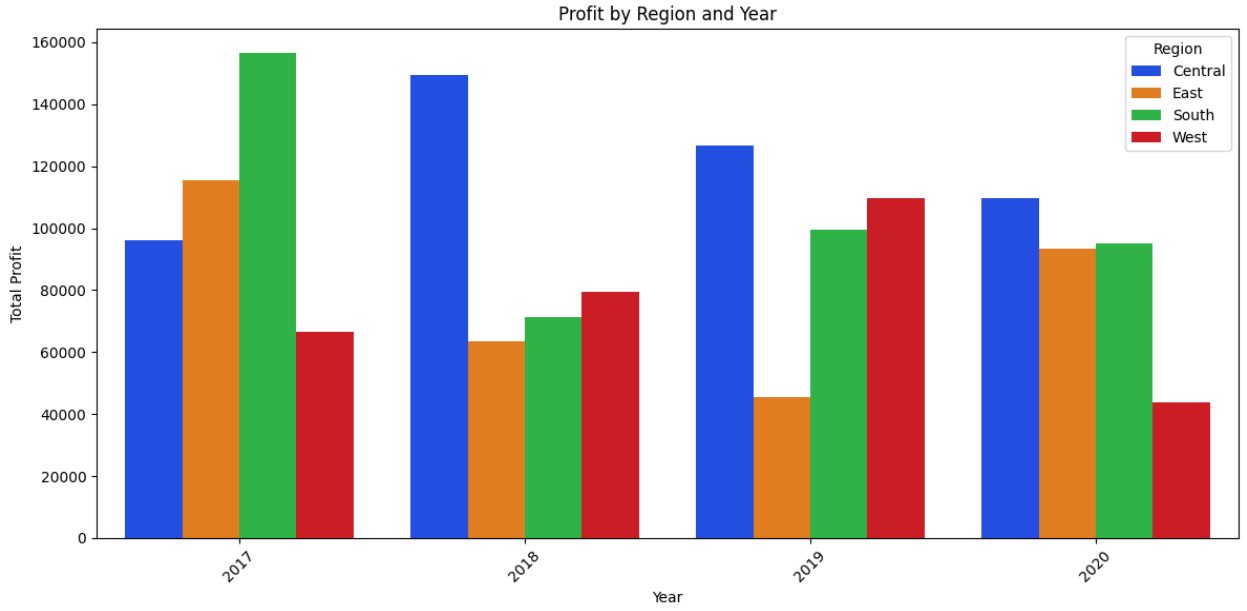



```
the Sales in 2018 in East was not the lowest , the profit was the lowest
6- Profit over both regoins and years
ProfitOverYearsRegions = superstore_sales.groupby(['Year','Region'])['Profit'].sum().reset_index()
print(ProfitOverYearsRegions)
```

```
   Year   Region     Profit
0  2017  Central   95989.26
1  2017     East  115609.93
2  2017    South  156477.53
3  2017     West   66462.01
4  2018  Central  149427.02
5  2018     East   63386.83
6  2018    South   71444.21
7  2018     West   79613.42
8  2019  Central  126631.22
9  2019     East   45621.66
```

```
10  2019    South   99588.31
11  2019     West  109614.80
12  2020  Central  109843.70
13  2020     East   93233.69
14  2020    South   94997.08
15  2020     West   43827.31
```

```
# Profit by Region and Year
plt.figure(figsize=(12, 6))
sns.set_palette('bright')
sns.barplot(data=ProfitOverYearsRegions, x='Year', y='Profit', hue='Region')
plt.title('Profit by Region and Year')
plt.xlabel('Year')
plt.ylabel('Total Profit')
plt.xticks(rotation=45)
plt.legend(title='Region')
plt.tight_layout()
plt.show()
```



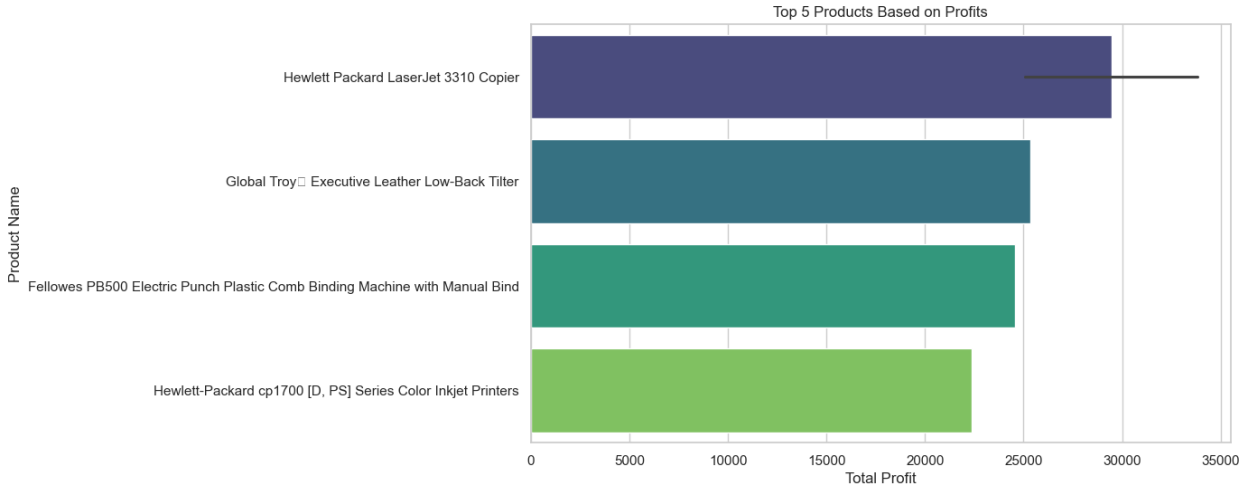
the region with the hieghests profit over all the years was Centeral and the east was the lowest till 2020
7- The highest and the lowest five product based on profit

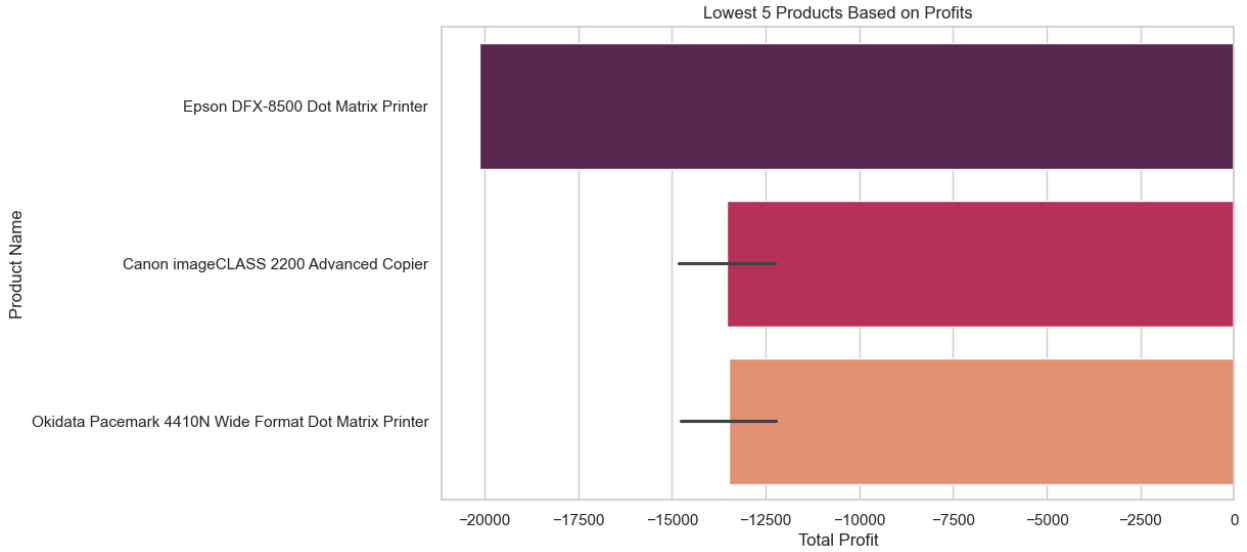
```
##The top Product  based on Profits
# Grouping the data
ProductProfitOverYears = superstore_sales.groupby(['Year','Product Name'])['Profit'].sum().reset_index()
```

```
# Find the top product
top_products = ProductProfitOverYears.nlargest(5,'Profit')
lowest_products = ProductProfitOverYears.nsmallest(5,'Profit')
# Printing the results
print("## The Top Product Based on Profits ##")
print(top_products.to_string(index=False))

print("\n## The Lowest Product Based on Profits ##")
print(lowest_products.to_string(index=False))

# Setting the visual style
sns.set(style="whitegrid")

# Plotting the Top Products
plt.figure(figsize=(10, 6))
sns.barplot(x='Profit', y='Product Name', data=top_products, palette='viridis')
plt.title('Top 5 Products Based on Profits')
plt.xlabel('Total Profit')
plt.ylabel('Product Name')
plt.show()

# Plotting the Lowest Products
plt.figure(figsize=(10, 6))
sns.barplot(x='Profit', y='Product Name', data=lowest_products, palette='rocket')
plt.title('Lowest 5 Products Based on Profits')
plt.xlabel('Total Profit')
plt.ylabel('Product Name')
plt.show()
```

```
he Top Product Based on Profits ##
 Year                                                Product Name   Profit
 2019                                Hewlett Packard LaserJet 3310 Copier 33825.10
 2020                          Global Troy™ Executive Leather Low-Back Tilter 25362.90
 2018                                Hewlett Packard LaserJet 3310 Copier 25092.25
 2017 Fellowes PB500 Electric Punch Plastic Comb Binding Machine with Manual Bind 24557.21
 2020          Hewlett-Packard cp1700 [D, PS] Series Color Inkjet Printers 22375.31

## The Lowest Product Based on Profits ##
 Year                                          Product Name    Profit
 2017                          Epson DFX-8500 Dot Matrix Printer -20133.62
 2017                Canon imageCLASS 2200 Advanced Copier -14830.99
 2017 Okidata Pacemark 4410N Wide Format Dot Matrix Printer -14761.93
 2019                Canon imageCLASS 2200 Advanced Copier -12253.38
 2020 Okidata Pacemark 4410N Wide Format Dot Matrix Printer -12227.52
C:\Users\elame\AppData\Local\Temp\ipykernel_9120\3705758638.py:21: FutureWarning:
```

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effect.





The most seller Product over years is Telephones and communications(category:Technology), the product that loses the most money is Tables (category:Furniture).

8- The relation between shipping mode and profit

```
# mean Profit by ship mode
mean_Profit_by_ship_mode = superstore_sales.groupby('Ship Mode')['Profit'].mean().reset_index()
print(mean_Profit_by_ship_mode)

sns.barplot(data= mean_Profit_by_ship_mode, x = "Ship Mode", y = "Profit")
plt.show()
```

```
Ship Mode      Profit
0  Delivery Truck  235.292208
1     Express Air  149.979969
2     Regular Air  176.187049
```

shipping modes, with Express Air and Regular Air contributing positively to profits while Delivery Truck operates at a loss.

9- the relation between Unit price and Profit

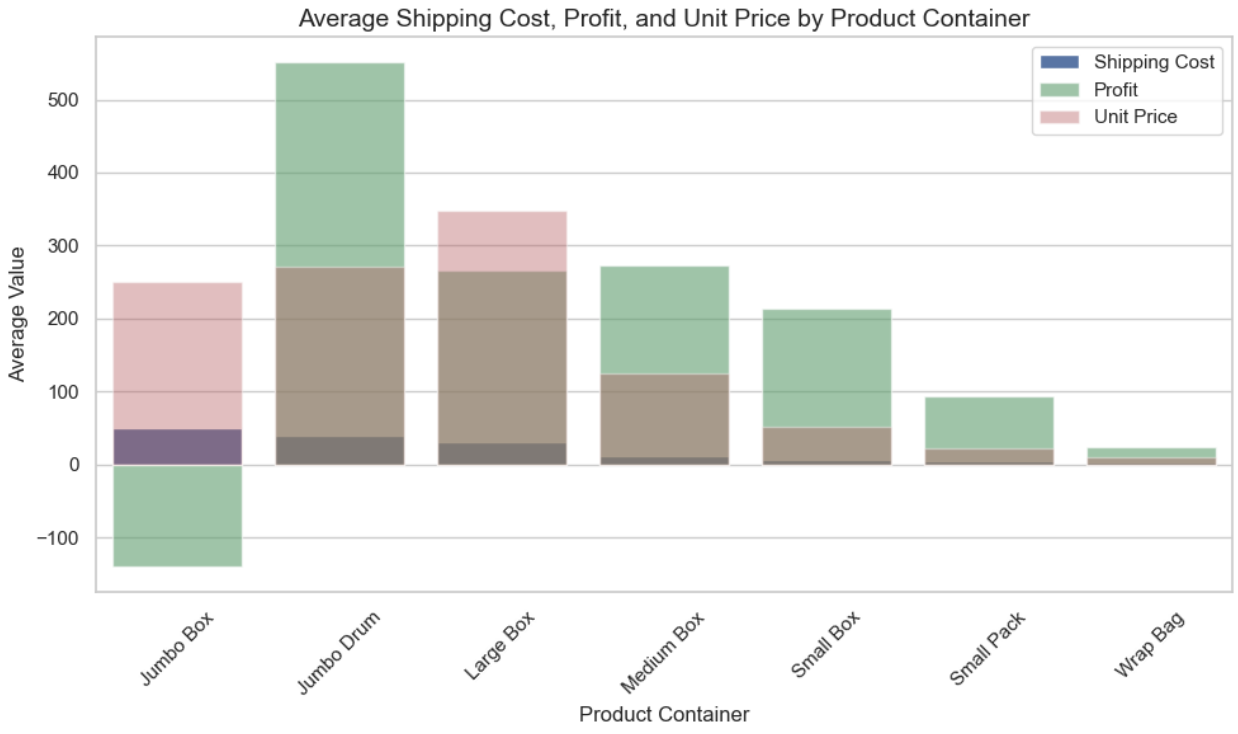
```
sns.scatterplot(data = superstore_sales , x ='Unit Price', y ='Profit')
plt.title('Relation between Unit Price and Profit')
plt.xlabel('Unit Price')
plt.ylabel('Profit ')
plt.show()
```


```
correlation = superstore_sales['Unit Price'].corr(superstore_sales['Profit'])
print(f'Correlation between Unit Price and Profit: {correlation}')
```

Correlation between Unit Price and Profit: -0.008853881167133414

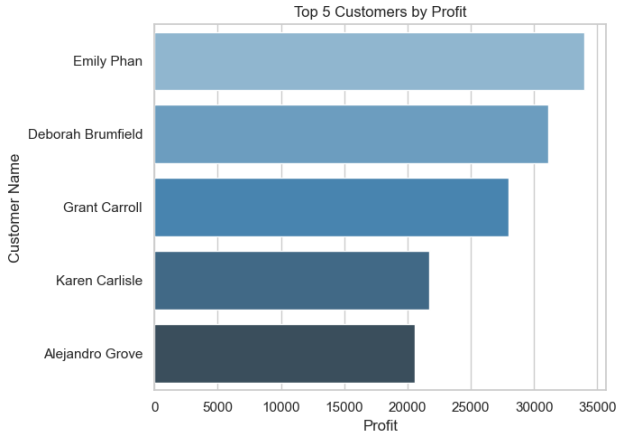
```
there is no correlation between them
10- The affect of container size on Shipment Cost and Profit Amount
CostBasedContainer  = superstore_sales.groupby('Product Container')['Shipping Cost'].mean().reset_index()
#print(CostBasedContainer)
ProfitBasedContainer  = superstore_sales.groupby('Product Container')['Profit'].mean().reset_index()
#print(ProfitBasedContainer)
UnitPriceBasedContainer = superstore_sales.groupby('Product Container')['Unit Price'].mean().reset_index()
#print(UnitPriceBasedContainer)

# Merge the three DataFrames based on 'Product Container'
merged_df = pd.merge(CostBasedContainer, ProfitBasedContainer, on='Product Container')
merged_df = pd.merge(merged_df, UnitPriceBasedContainer, on='Product Container')

# Set up the figure and size
plt.figure(figsize=(10, 6))
```

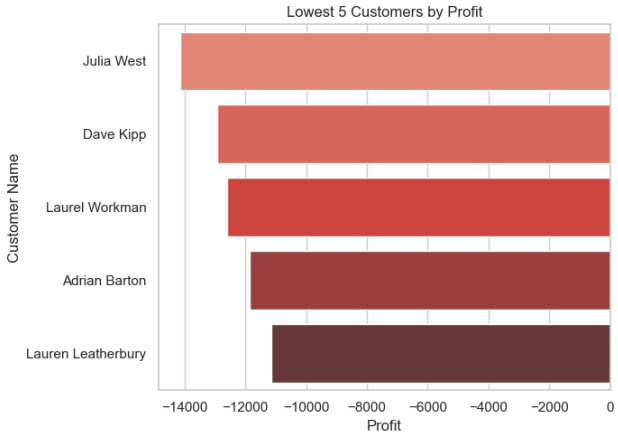
```
# Plot Shipping Cost
sns.barplot(x='Product Container', y='Shipping Cost', data=merged_df, color='b', label='Shipping Cost')

# Plot Profit (on the same axes)
sns.barplot(x='Product Container', y='Profit', data=merged_df, color='g', alpha=0.6, label='Profit')

# Plot Unit Price (on the same axes)
sns.barplot(x='Product Container', y='Unit Price', data=merged_df, color='r', alpha=0.4, label='Unit Price')

# Add title and labels
plt.title('Average Shipping Cost, Profit, and Unit Price by Product Container', fontsize=14)
plt.xlabel('Product Container', fontsize=12)
plt.ylabel('Average Value', fontsize=12)

# Show legend
plt.legend()

# Rotate x-axis labels for better readability
plt.xticks(rotation=45)

# Show the plot
plt.tight_layout()
plt.show()
```

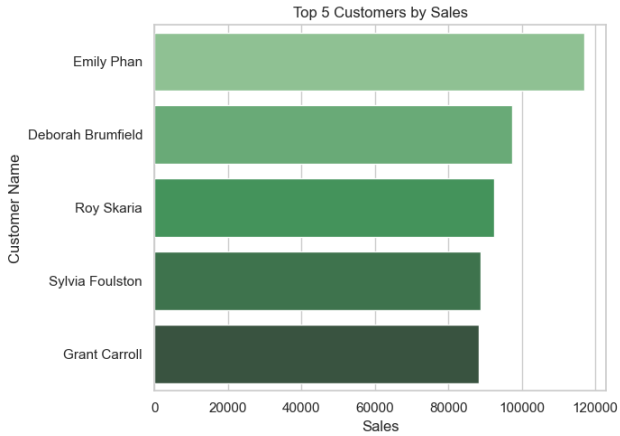


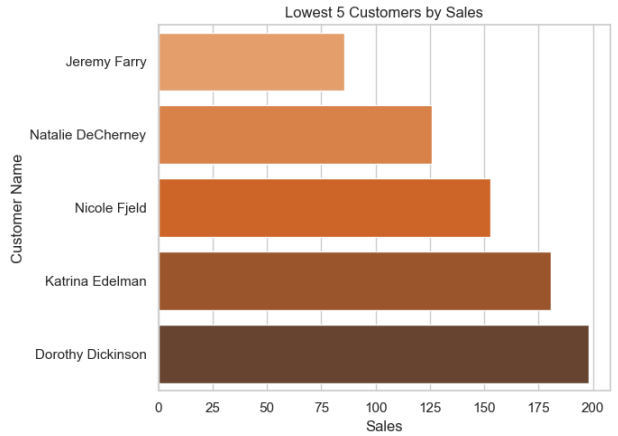
```
while Jumbo and Large boxes required `High Shipping` cost and increase `Unit Price` they lead to a `loss`. On the other hand the most `profitable` containers are `medium and smaller containers`.
correlation_shipment_unit_price = superstore_sales['Shipping Cost'].corr(superstore_sales['Unit Price'])

# Calculate the correlation between Shipment Cost and Profit
correlation_shipment_profit = superstore_sales['Shipping Cost'].corr(superstore_sales['Profit'])

# Print the results
print("Correlation between Shipment Cost and Unit Price:", correlation_shipment_unit_price)
print("Correlation between Shipment Cost and Profit:", correlation_shipment_profit)
there is a quite good correlation between Shipment Cost and Unit Price
while a little or weak correlation between Shipment Cost and Profit
11- who is the top and low five customer based on sales and profit
Find the top Customer
# Top and Lowest Customers by Profit
topCustomerProfit = superstore_sales.groupby('Customer Name')['Profit'].sum().reset_index()

# Find top 5 customers by Profit
topEltopP = topCustomerProfit.nlargest(5, 'Profit')

# Find lowest 5 customers by Profit
lowEltopP = topCustomerProfit.nsmallest(5, 'Profit')

# Top and Lowest Customers by Sales
topCustomerSales = superstore_sales.groupby('Customer Name')['Sales'].sum().reset_index()

# Find top 5 customers by Sales
topEltopS = topCustomerSales.nlargest(5, 'Sales')

# Find lowest 5 customers by Sales
lowEltopS = topCustomerSales.nsmallest(5, 'Sales')
#Create a figure for the subplots
plt.figure(figsize=(14, 10))

# Subplot for Top Customers by Profit
plt.subplot(2, 2, 1)
sns.barplot(data=topEltopP, x='Profit', y='Customer Name', palette='Blues_d')
plt.title('Top 5 Customers by Profit')
plt.xlabel('Profit')
plt.ylabel('Customer Name')

# Subplot for Lowest Customers by Profit
plt.subplot(2, 2, 2)
```

```
sns.barplot(data=lowEltopP, x='Profit', y='Customer Name', palette='Reds_d')
plt.title('Lowest 5 Customers by Profit')
plt.xlabel('Profit')
plt.ylabel('Customer Name')

# Subplot for Top Customers by Sales
plt.subplot(2, 2, 3)
sns.barplot(data=topEltopS, x='Sales', y='Customer Name', palette='Greens_d')
plt.title('Top 5 Customers by Sales')
plt.xlabel('Sales')
plt.ylabel('Customer Name')

# Subplot for Lowest Customers by Sales
plt.subplot(2, 2, 4)
sns.barplot(data=lowEltopS, x='Sales', y='Customer Name', palette='Oranges_d')
plt.title('Lowest 5 Customers by Sales')
plt.xlabel('Sales')
plt.ylabel('Customer Name')

# Adjust layout
plt.tight_layout()
plt.show()
```









5.Losses analysis**

questions:

1-what is the losses of each status?

2-which mode lose the most?

3-which priority lose the most?

4-which container lose the most?

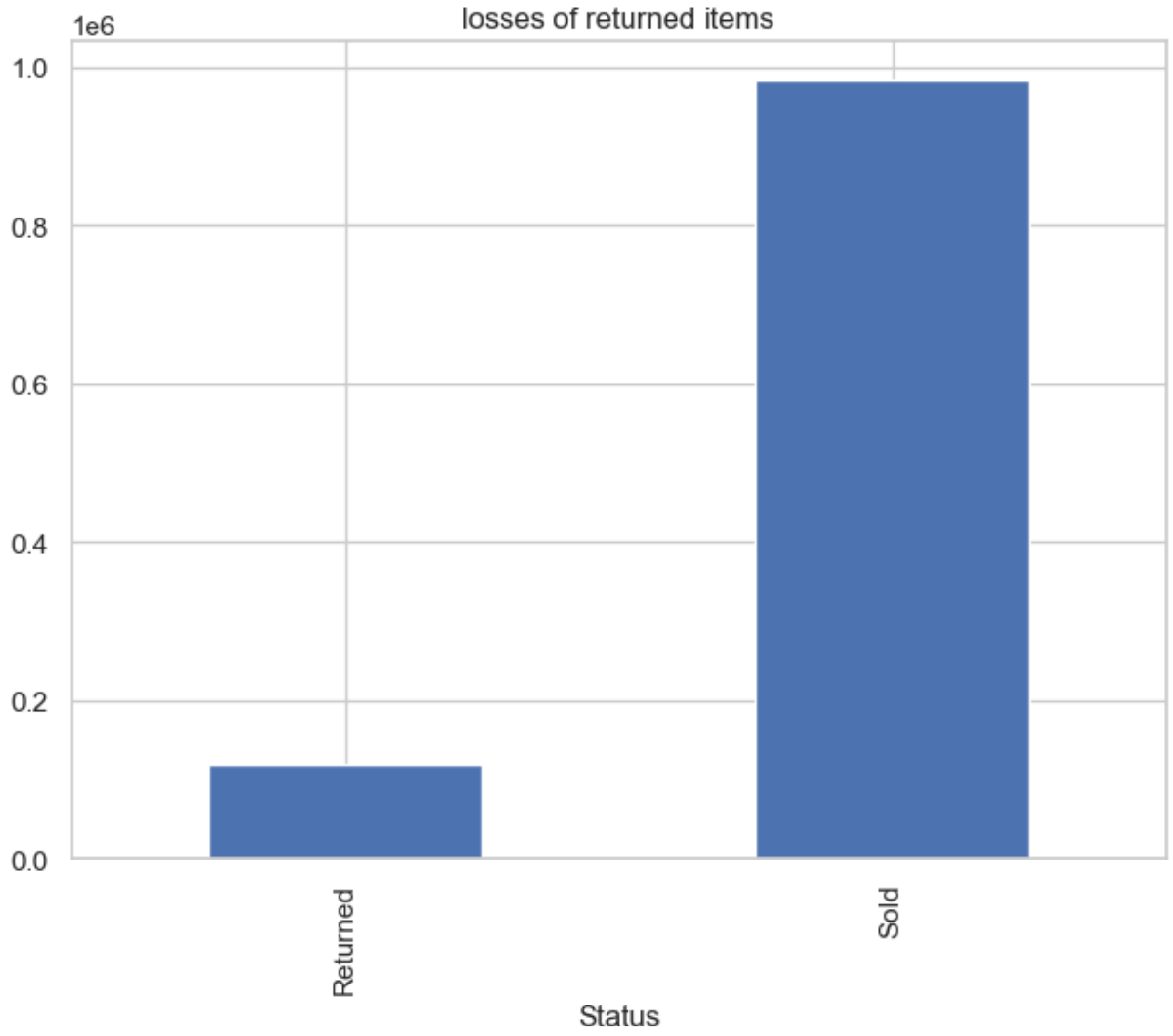
5-which segment lose the most?

6-which region lose the most?

7-which category lose the most?

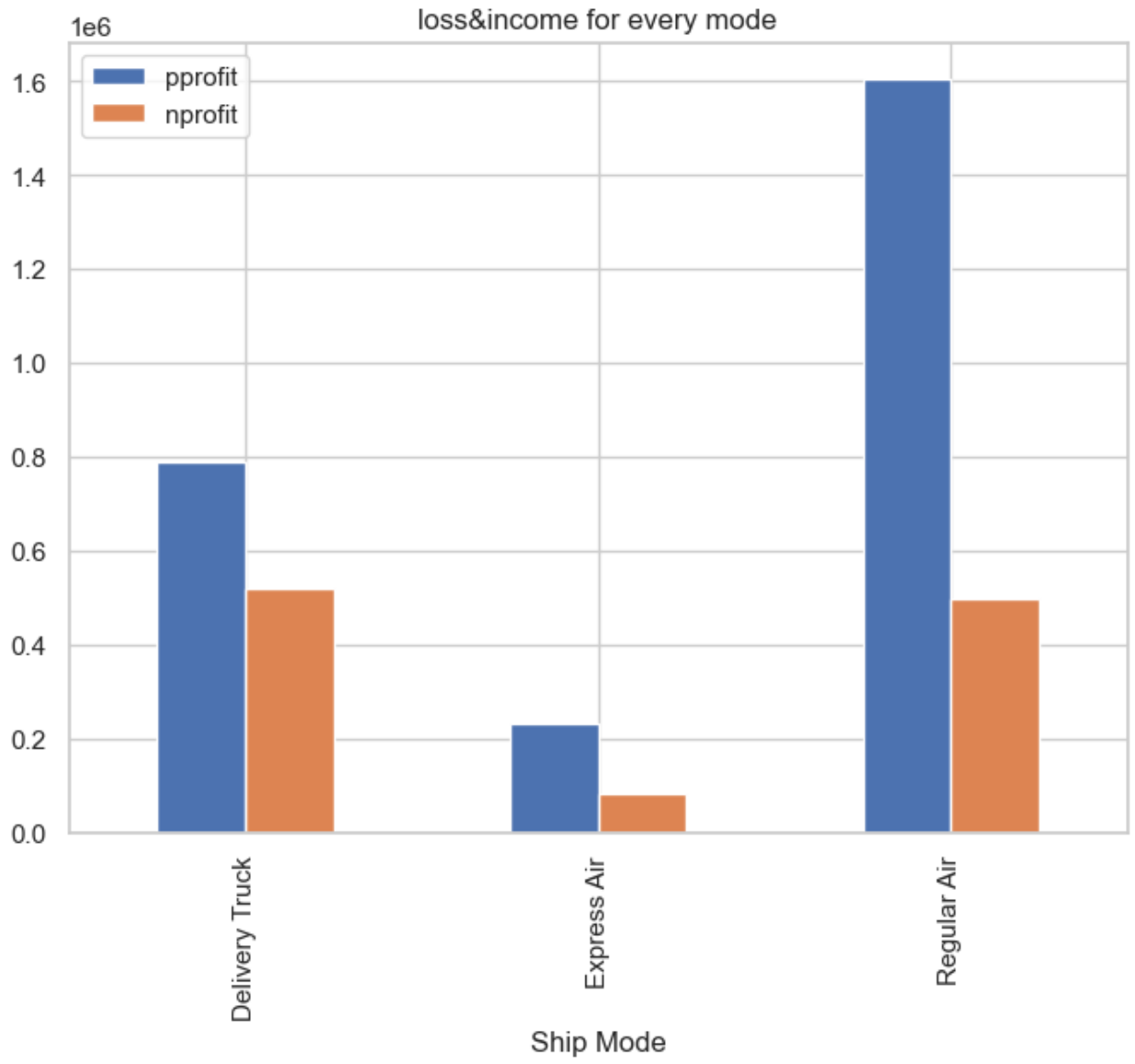
8-which subcategory loses the most?

```
--------------------------

1-what is the losses of each status?
negative=superstore_sales[superstore_sales['Profit']<0]
positive=superstore_sales[superstore_sales['Profit']>0]
return_loss=-negative.groupby('Status')['Profit'].sum()
return_loss.plot(kind='bar', title='losses of returned items', figsize=(8, 6))
```




```
the largest loss is concerned with the sold items so, the focus would be on them

2-which mode lose the most?
Ship_loss=-negative.groupby('Ship Mode')['Profit'].sum()
```

```
ship_income=positive.groupby('Ship Mode')['Profit'].sum()
mode_frame = pd.DataFrame({
    'pprofit': ship_income,
    'nprofit': Ship_loss
})

mode_frame.plot(kind='bar', title='loss&income for every mode', figsize=(8, 6))
#Ship_loss.plot(kind='bar', title='losses of returned items', figsize=(8, 6))
```



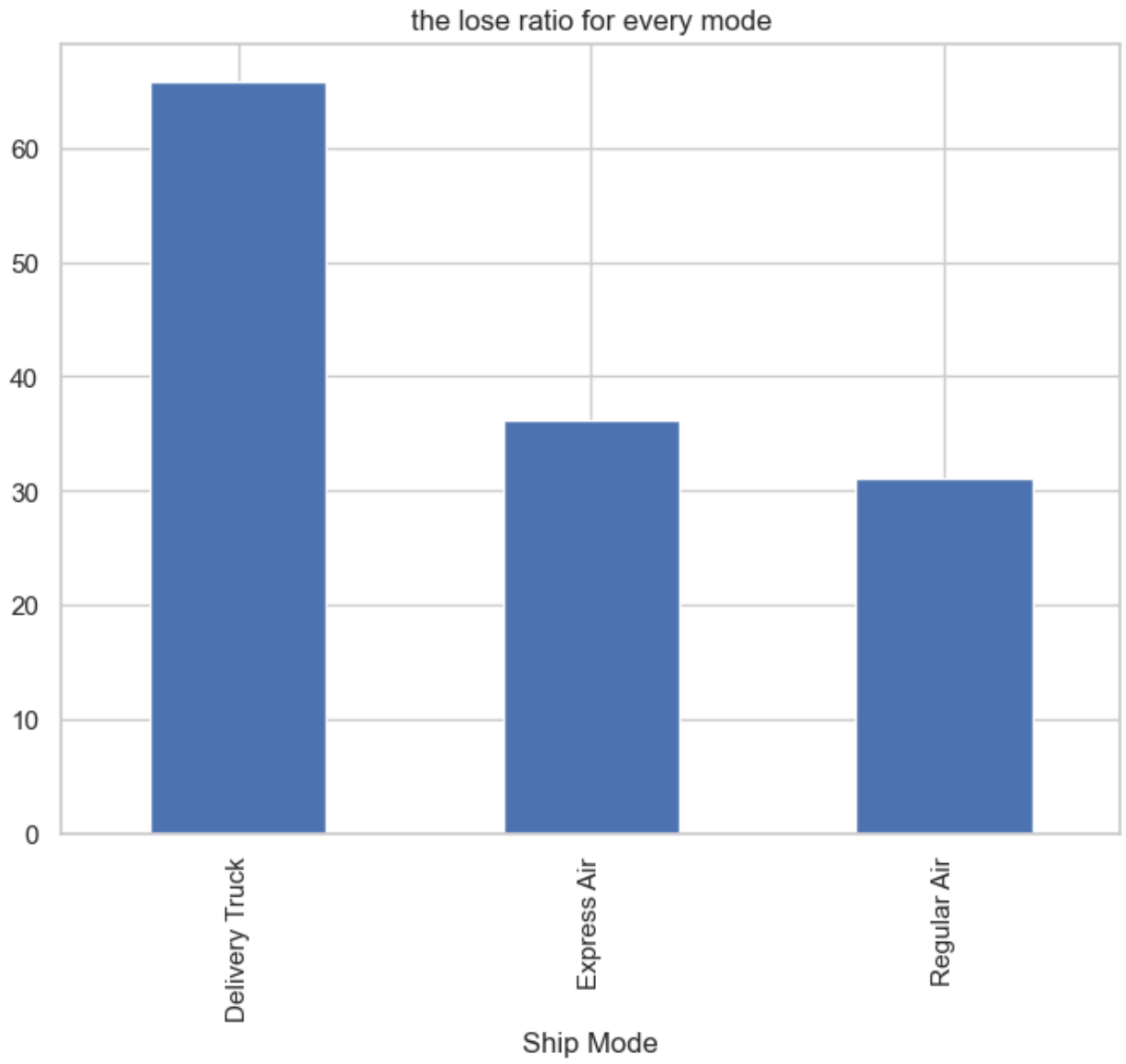
losses from delivery truck and regular are are simalar, in the other hand, regular are has alot of income so the plane losses value
is not enough

by dividing the losses over the income, it would give us the ratio by which this mode lose

ex. if I have income 100 and losses 20 then the ratio would be the 1/5 (20%)

ex. if I have total income of 2000 and the lose ratio is 15% then the losses would be 2000*0.15=300

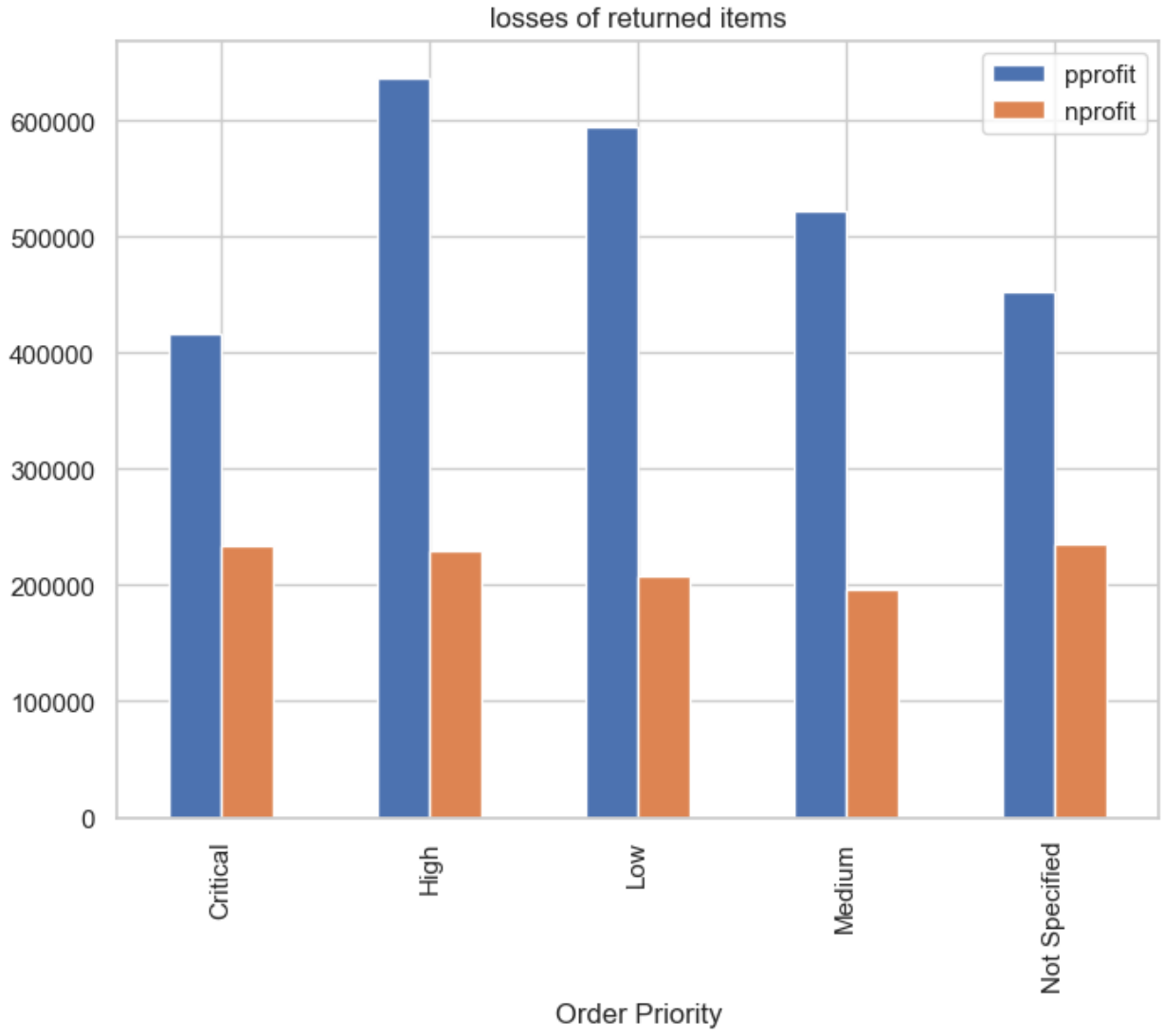
```
Ship_lossavg=-100*negative.groupby('Ship Mode')['Profit'].sum() / positive.groupby('Ship Mode')['Profit'].sum()
Ship_lossavg.plot(kind='bar', title='the lose ratio for every mode', figsize=(8, 6))
```



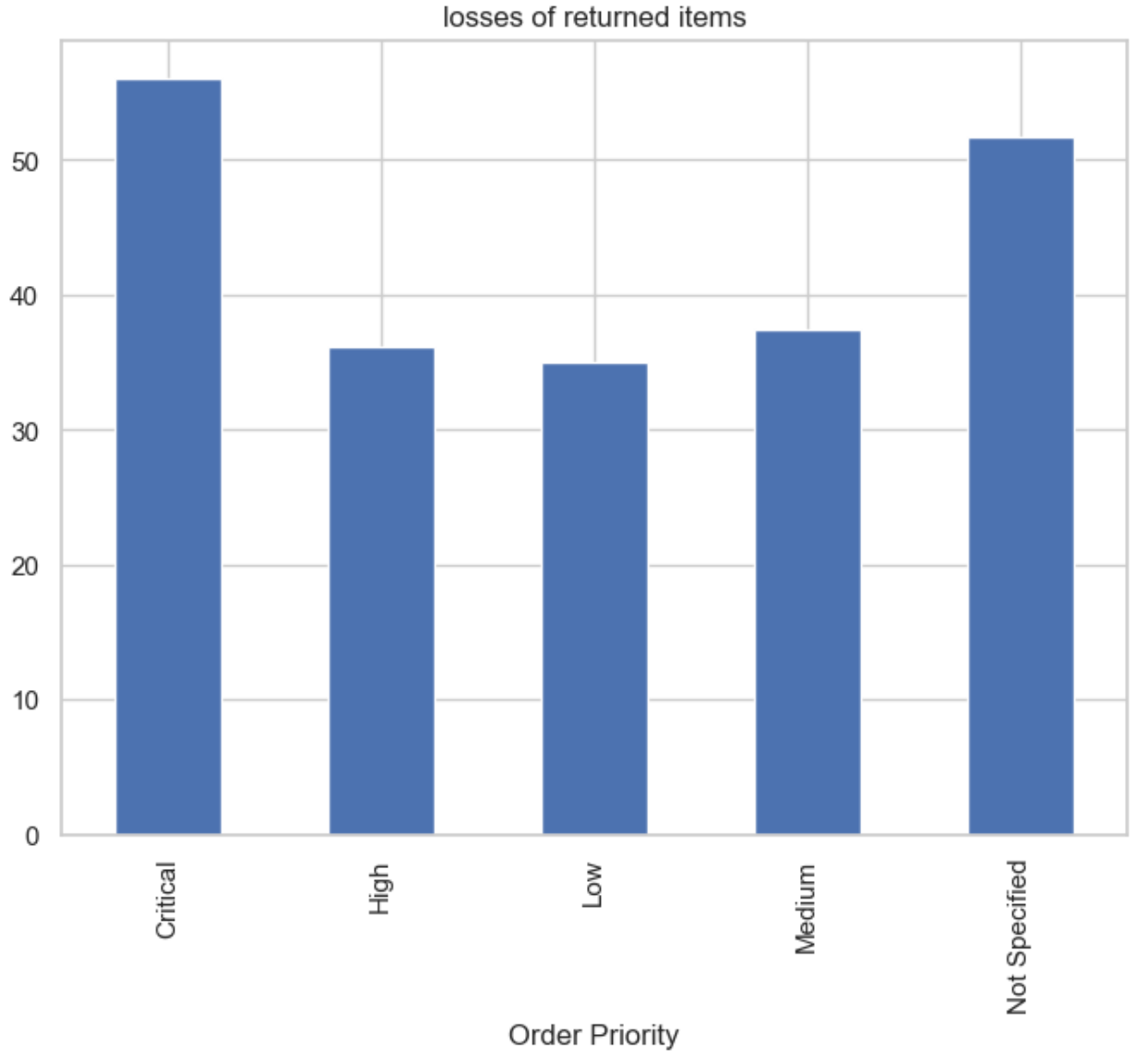
after seeing the the percent of the losses to the income, it turns out regular air is not bad. Actually it is more efficient than the express air.
the gragh shows that there is a problem with the truck as it has >60% losses which is alot

3-which priority lose the most?

```
Priority_loss=-negative.groupby('Order Priority')['Profit'].sum()
Priority_income=positive.groupby('Order Priority')['Profit'].sum()
Priority_frame = pd.DataFrame({
    'pprofit': Priority_income,
    'nprofit': Priority_loss
})
Priority_frame.plot(kind='bar', title='loss&income for every priority', figsize=(8, 6))
```



```
Priority_lossavg=-100*negative.groupby('Order Priority')['Profit'].sum() / positive.groupby('Order Priority')['Profit'].sum()
Priority_lossavg.plot(kind='bar', title='losses ratio for priorities', figsize=(8, 6))
```

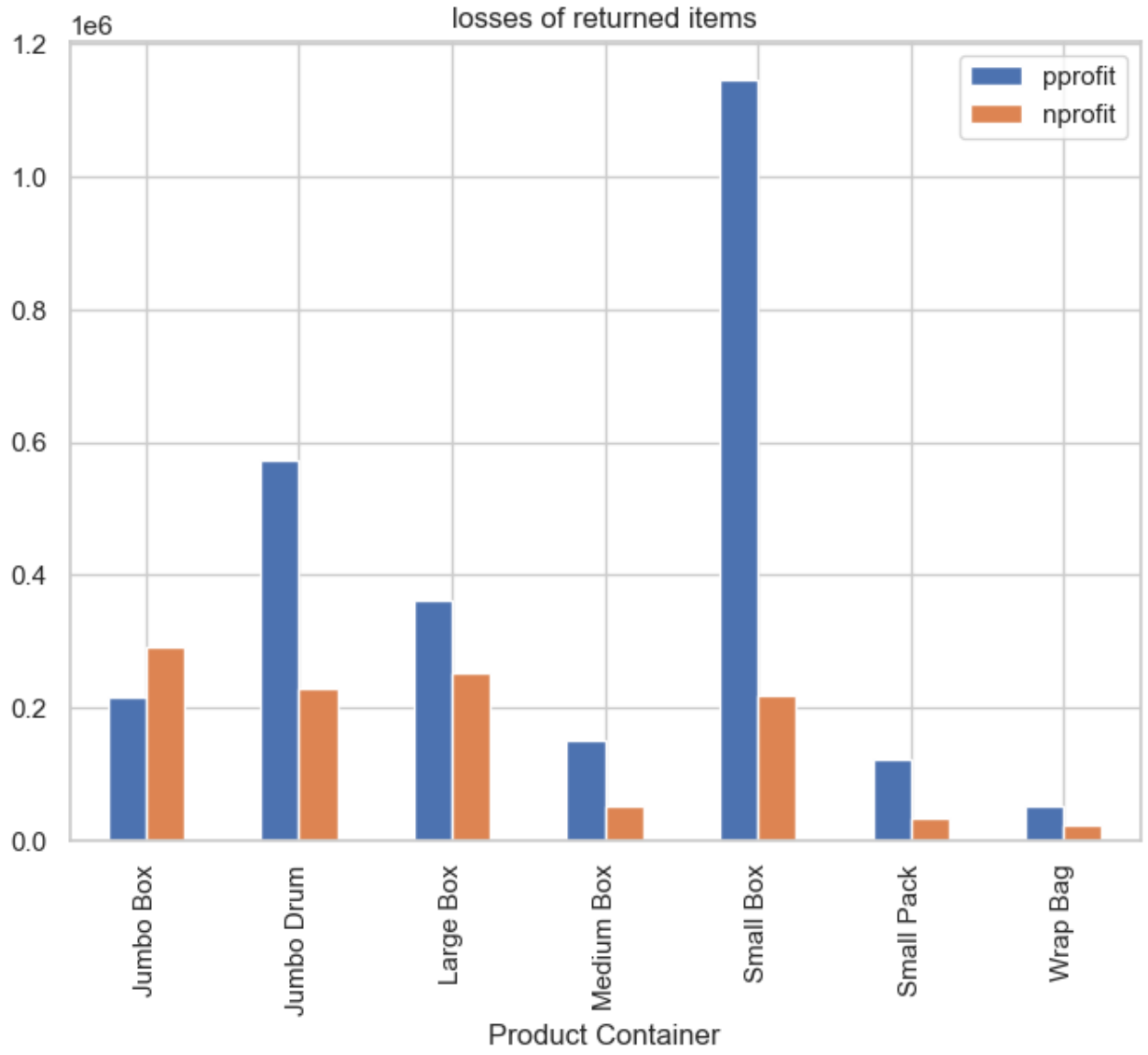


as shown, the numbers of loss are close but for the percentages, the "critical" and "not specified" sections results in losses which exceed 50%!

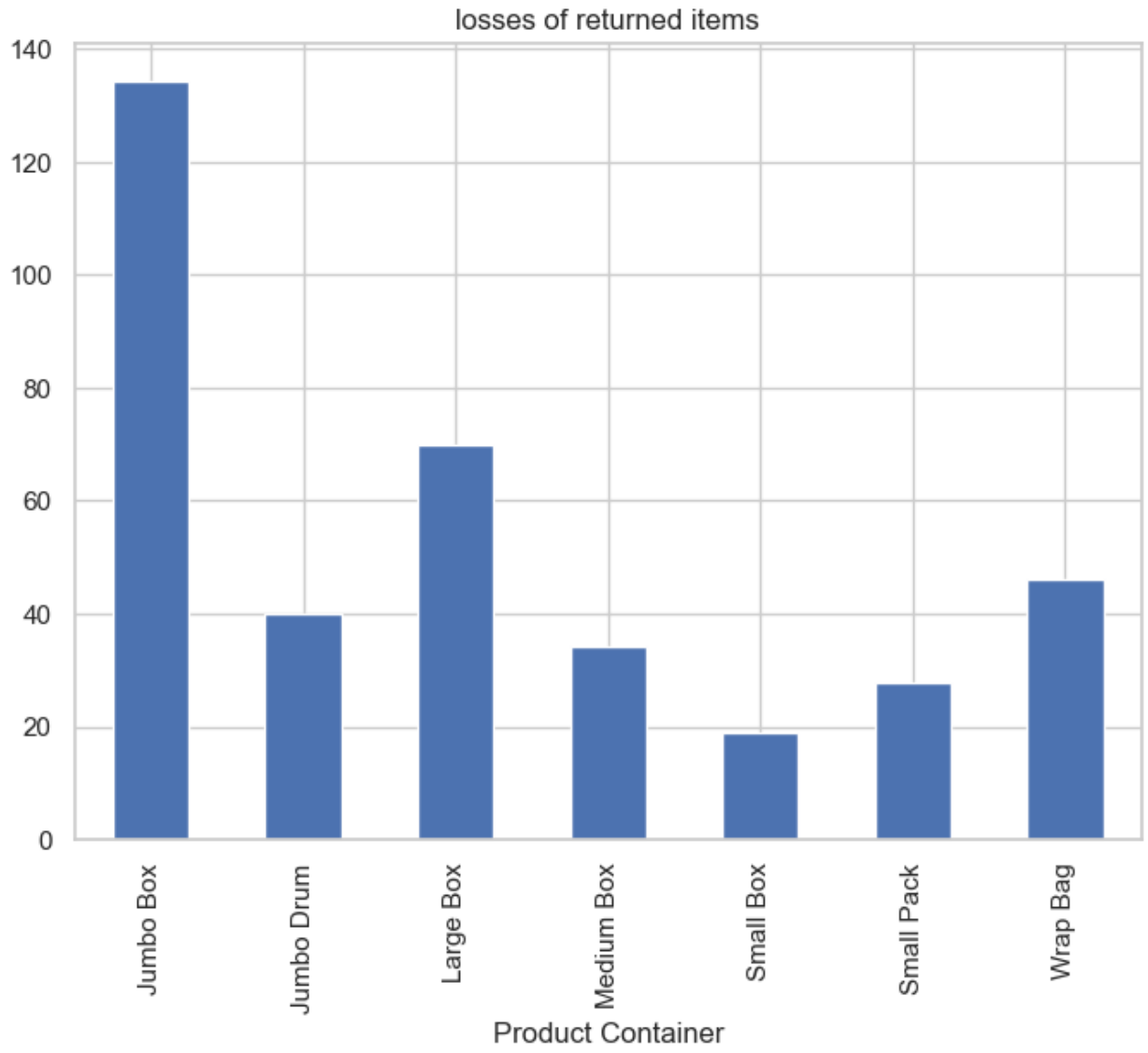
4-which container lose the most?

```
Container_loss=-negative.groupby('Product Container')['Profit'].sum()
Container_income=positive.groupby('Product Container')['Profit'].sum()
Container_frame = pd.DataFrame({
    'pprofit': Container_income,
```

```
    'nprofit': Container_loss
})
Container_frame.plot(kind='bar', title='loss&income for every container', figsize=(8, 6))
```



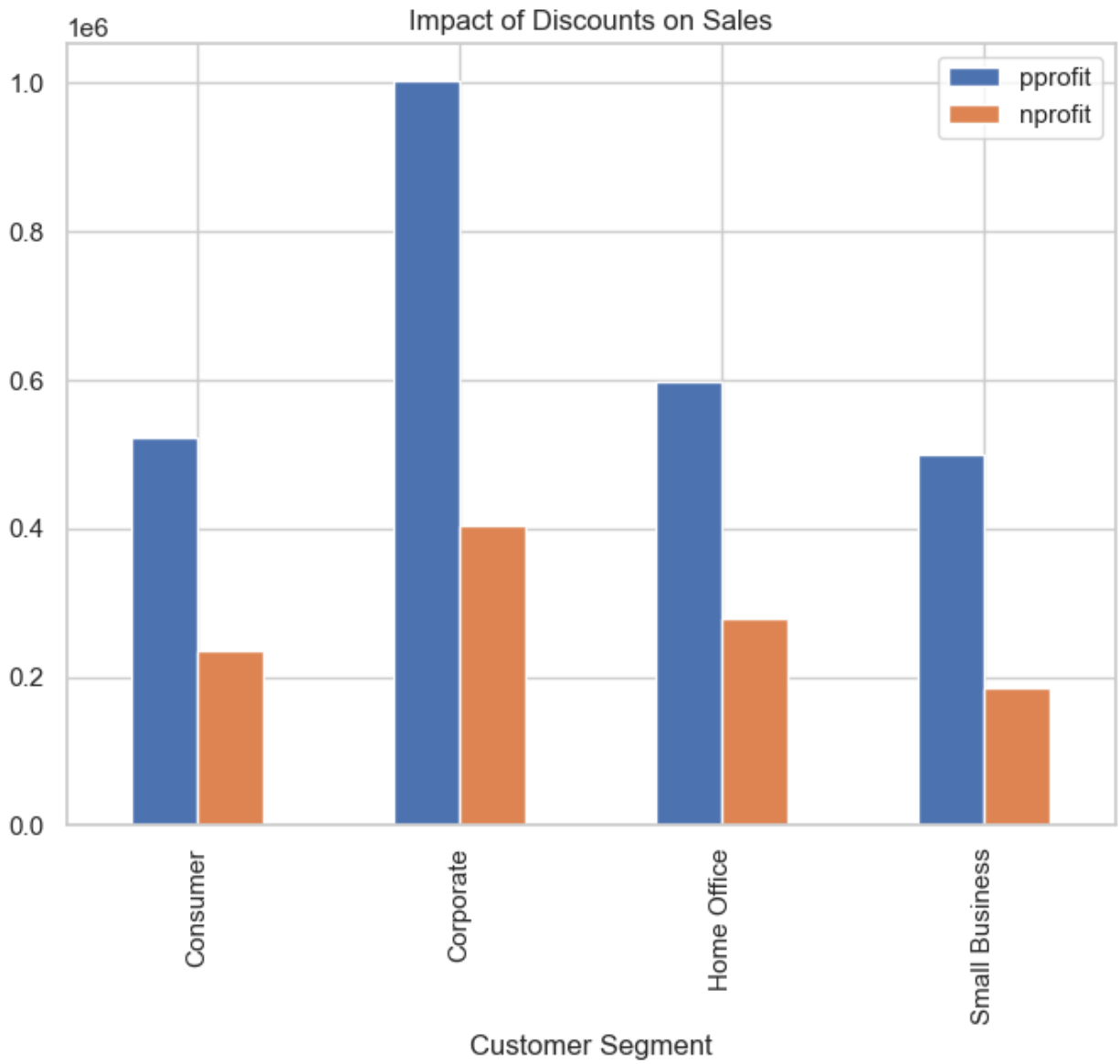

```
container_lossavg=-100*negative.groupby('Product Container')['Profit'].sum() / positive.groupby('Product Container')['Profit'].sum()
container_lossavg.plot(kind='bar', title='losses ratio for containers', figsize=(8, 6))
```



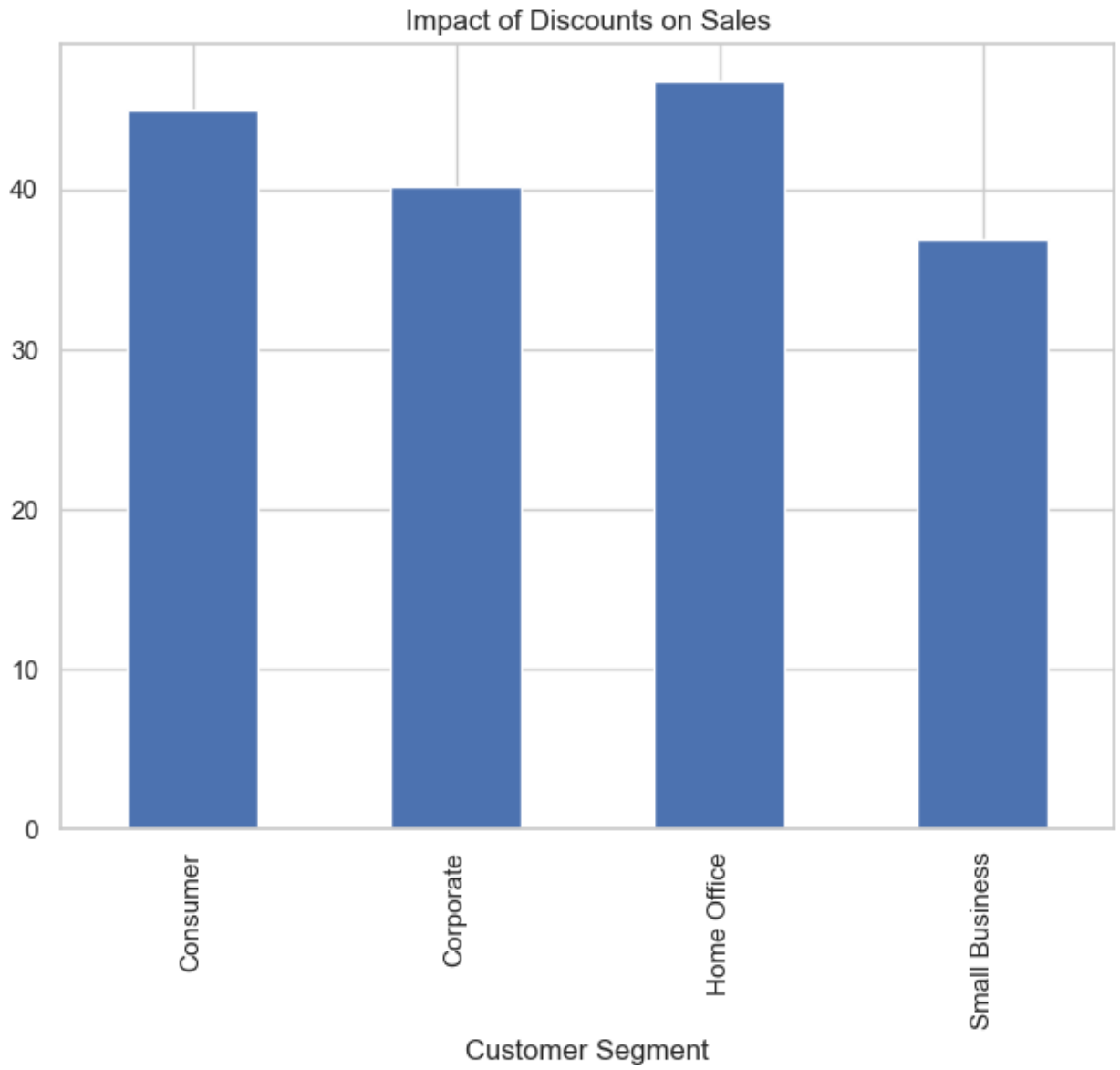
for the containers, jumbo box have a loss percent of >130% which is unacceptable
must find a quick solution for it + the large box has a lose rate of >70%

5-which segment lose the most?

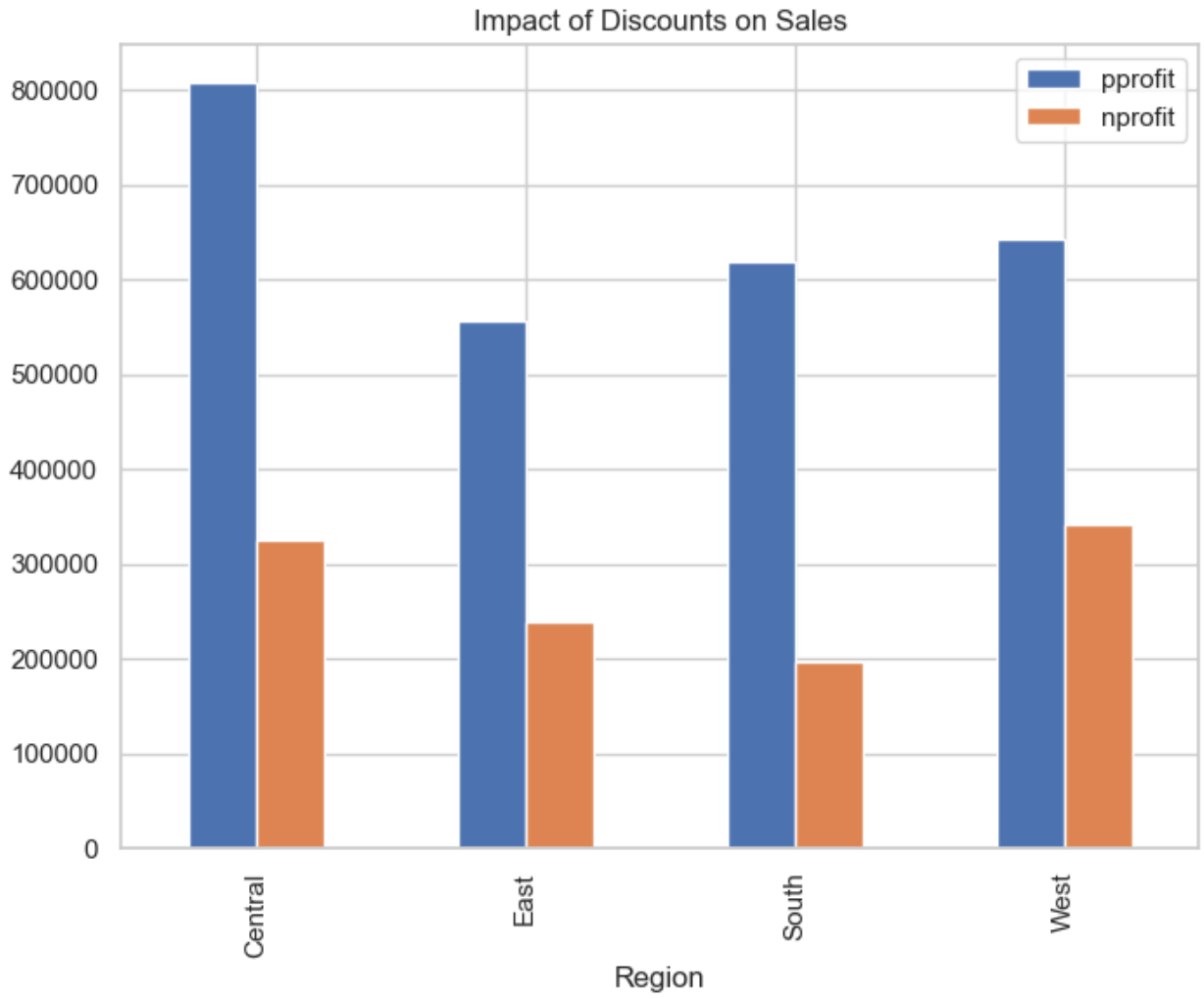
```
segment_loss = -negative.groupby('Customer Segment')['Profit'].sum()
segment_income=positive.groupby('Customer Segment')['Profit'].sum()
segment_frame = pd.DataFrame({
    'pprofit': segment_income,
    'nprofit': segment_loss
})

segment_frame.plot(kind='bar', title='loss&income for every segment', figsize=(8, 6))
```



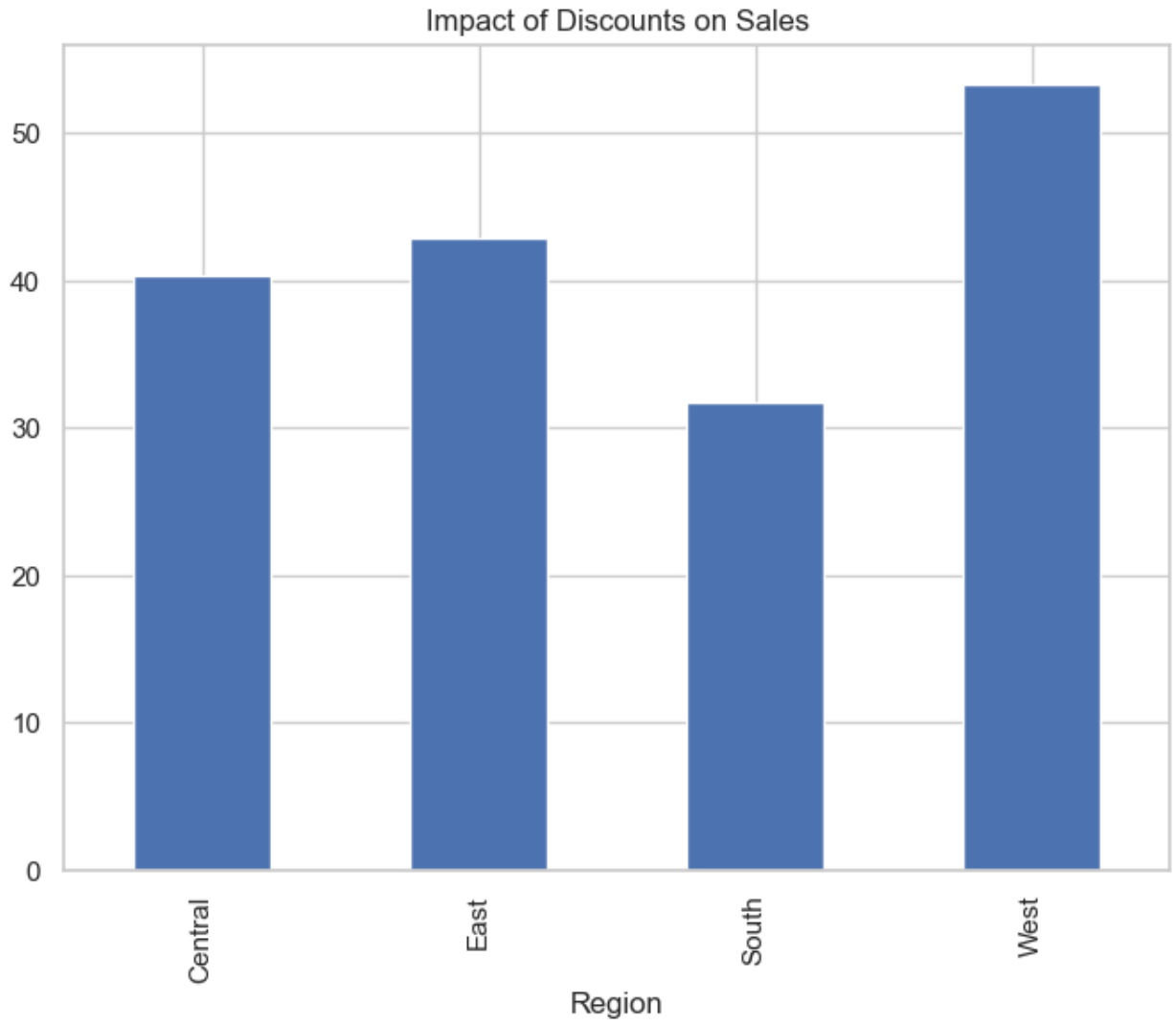

```
segment_lossavg = -100*negative.groupby('Customer Segment')['Profit'].sum() / positive.groupby('Customer Segment')['Profit'].sum()
segment_lossavg.plot(kind='bar', title='loss ratio for segments', figsize=(8, 6))
```




```
non of them exceeds 50% but the lose ratio still high

6-which region lose the most?
region_loss = -negative.groupby('Region')['Profit'].sum()
region_income=positive.groupby('Region')['Profit'].sum()
region_frame = pd.DataFrame({
    'pprofit': region_income,
    'nprofit': region_loss
})

region_frame.plot(kind='bar', title='loss&income for every region', figsize=(8, 6))
```




```
region_lossavg = -100*negative.groupby('Region')['Profit'].sum() / positive.groupby('Region')['Profit'].sum()
region_lossavg.plot(kind='bar', title='loss ratio for regions', figsize=(8, 6))
```



west has the largest ratio which exceeds 50%

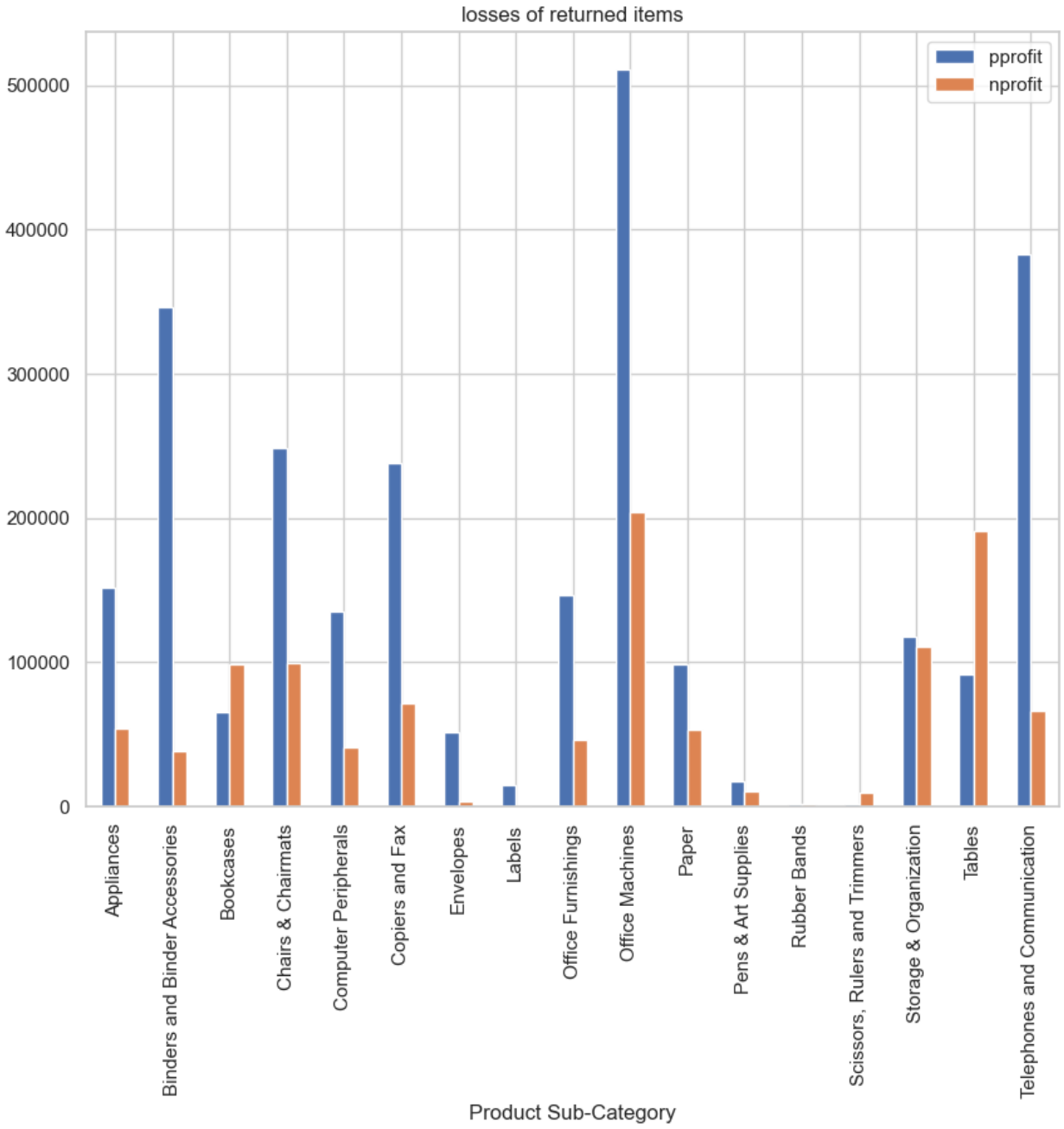
7-which category lose the most?

```
Category_loss = -negative.groupby('Product Category')['Profit'].sum()
Category_income=positive.groupby('Product Category')['Profit'].sum()
Categ = pd.DataFrame({
    'pprofit': Category_income,
    'nprofit': Category_loss
})

Categ.plot(kind='bar', title='loss&income for every category', figsize=(8, 6))
#Category_loss.plot(kind='bar', title='losses of returned items', figsize=(8, 6))
```

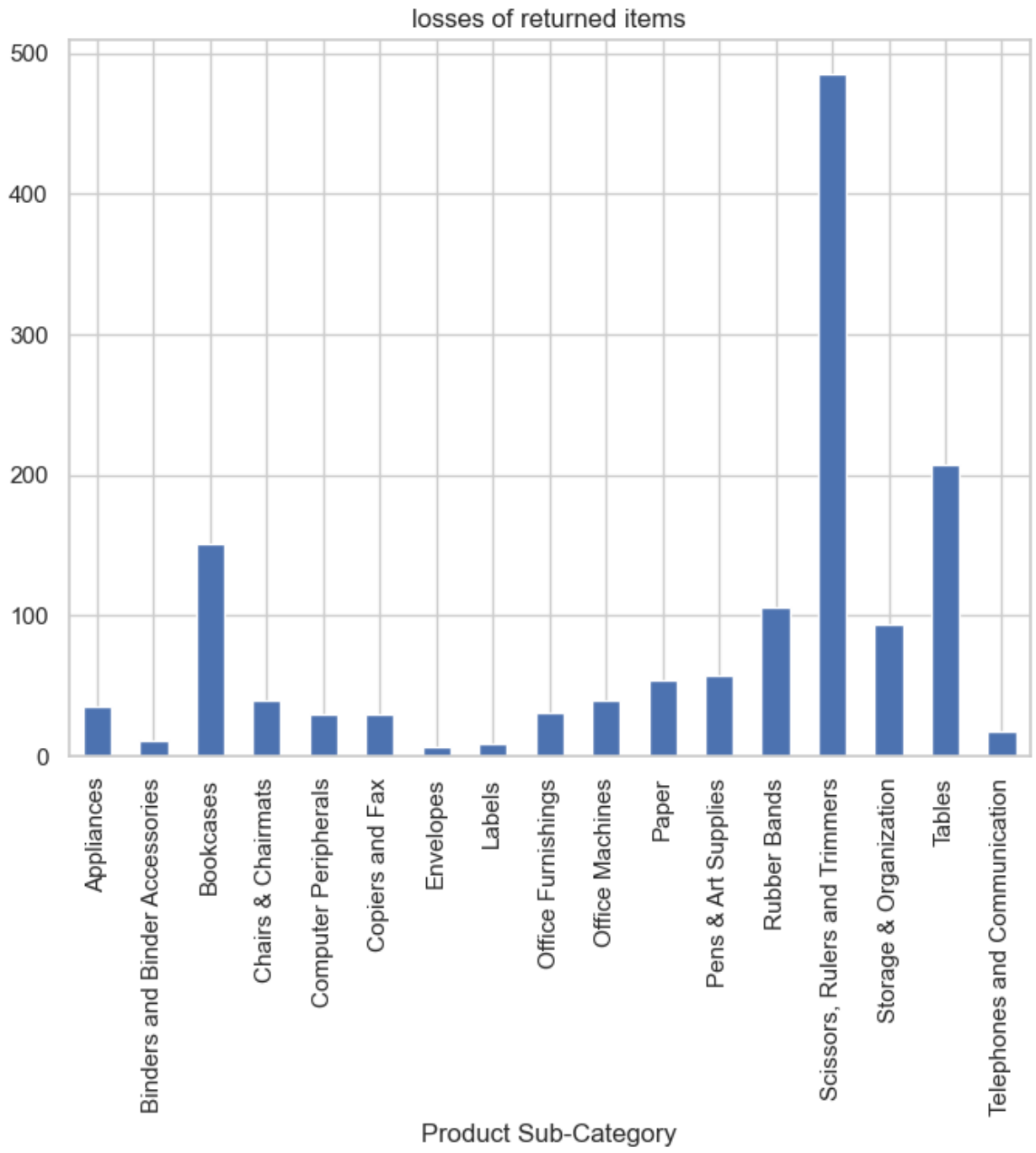
```
categ_lossavg=-100*negative.groupby('Product Category')['Profit'].sum() / positive.groupby('Product Category')['Profit'].sum()
categ_lossavg.plot(kind='bar', title='losses ratio for categories', figsize=(8, 6))
```
technology category has alot of losses but the income is so much higher as its percentage is the lowest.
for furniture, it has a lose percent of >75% and causes the biggest lose among other categories

8-which subcategory loses the most?

```
subCategory_loss = -negative.groupby('Product Sub-Category')['Profit'].sum()
subCategory_income=positive.groupby('Product Sub-Category')['Profit'].sum()
subCategory_frame = pd.DataFrame({
    'pprofit': subCategory_income,
    'nprofit': subCategory_loss
})
subCategory_frame.plot(kind='bar', title='loss&income for every subcategory', figsize=(10, 8))
```




```
subcateg_lossavg=-100*negative.groupby('Product Sub-Category')['Profit'].sum() / positive.groupby('Product Sub-Category')['Profit'].sum()
subcateg_lossavg.plot(kind='bar', title='losses rotio for subcategories', figsize=(8, 6))
```



the scissors,rulers&trimmers has a lose rate of >450% !
it doesn't affect the total loss as sales for it is small but it is nonsense to continue selling it with this lose rate
there are problems with the bookcases(it also has a problem with returns), tables , rubber bands and organizations as the lose rate exceeds 100%
****6. Returning questions****

questions:

1-how many orders are returned concerned to the sold ones?

2-is the problem in a certain ship mode?

3-is the problem in a certain Category?

4-is the problem in a certain SubCategory?

5-what are the top Products returned?

6-is the problem in a certain Region?

7-is the problem in a certain Customer segment?

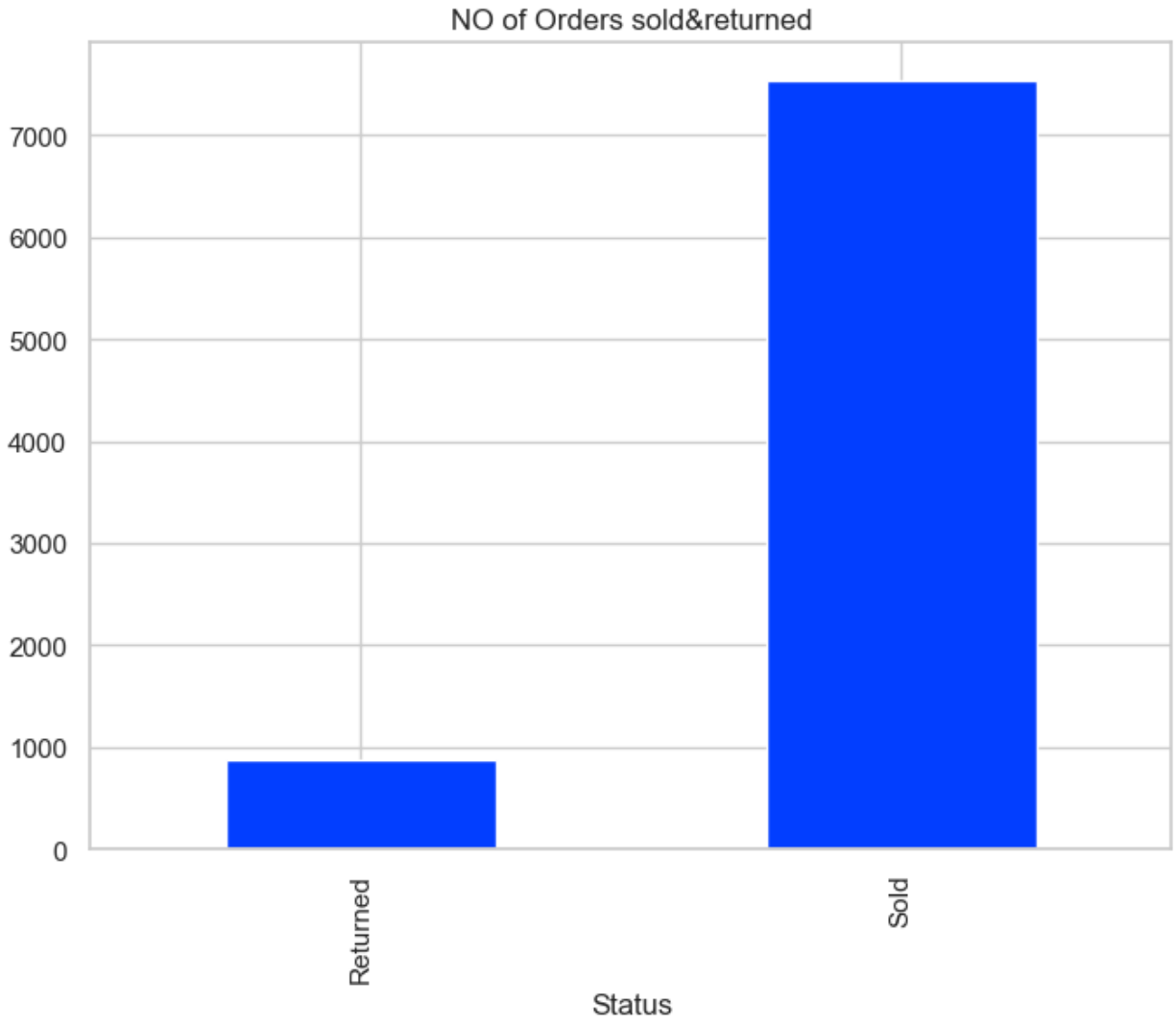
8-who are the most customer returning orders?

9-is the problem in a certain container?

10-is the problem in a certain priority?

1-how many orders are returned concerned to the sold ones?

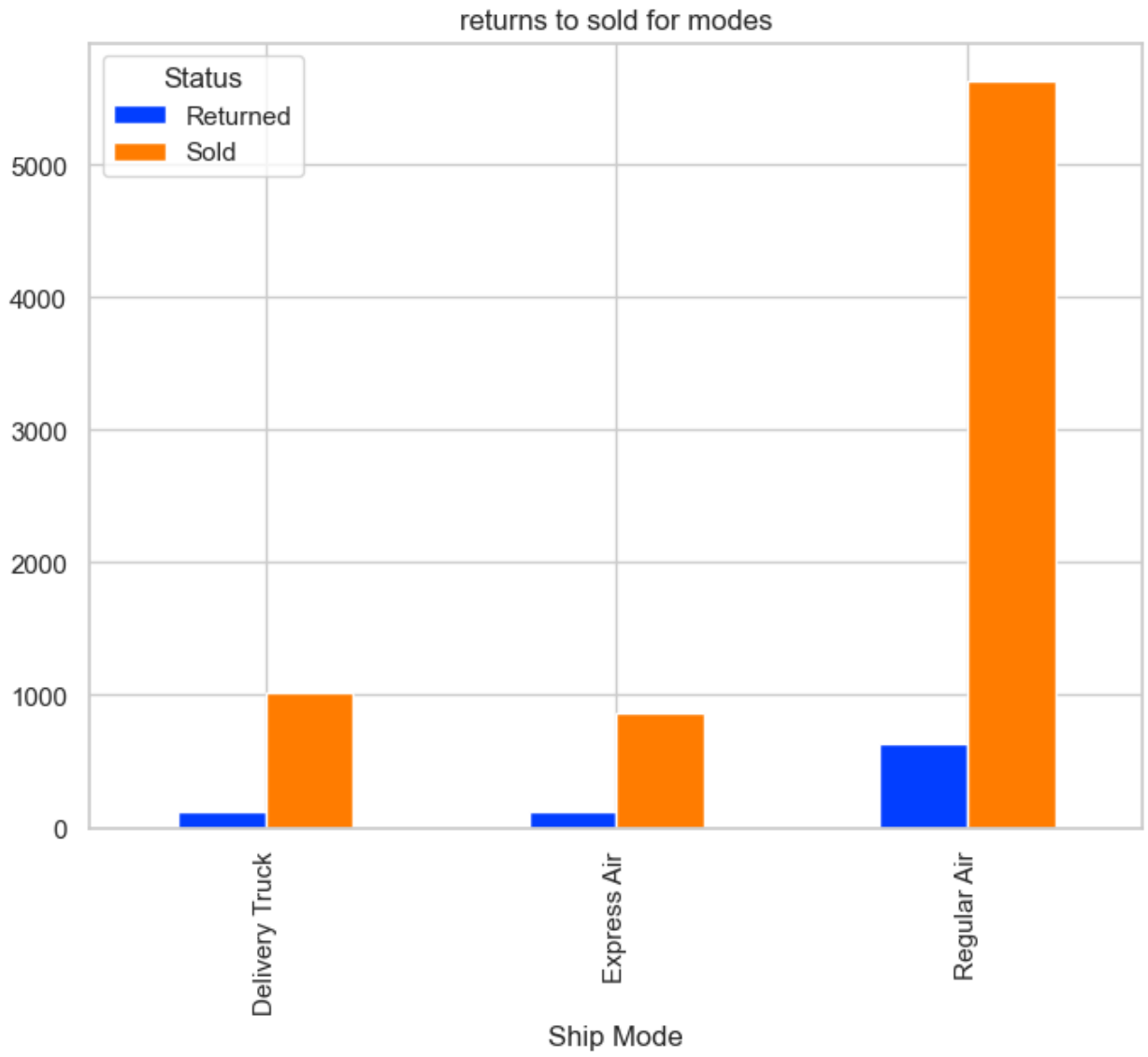
```
sales_return=superstore_sales[superstore_sales['Status']=='Returned']
bystatus=superstore_sales.groupby('Status')['Order ID'].count()
bystatus.plot(kind='bar', title='NO of Orders sold&returned', figsize=(8, 6))
```



a thousand returns among more than 8000 orders! it is around eighth of the orders and thats alot

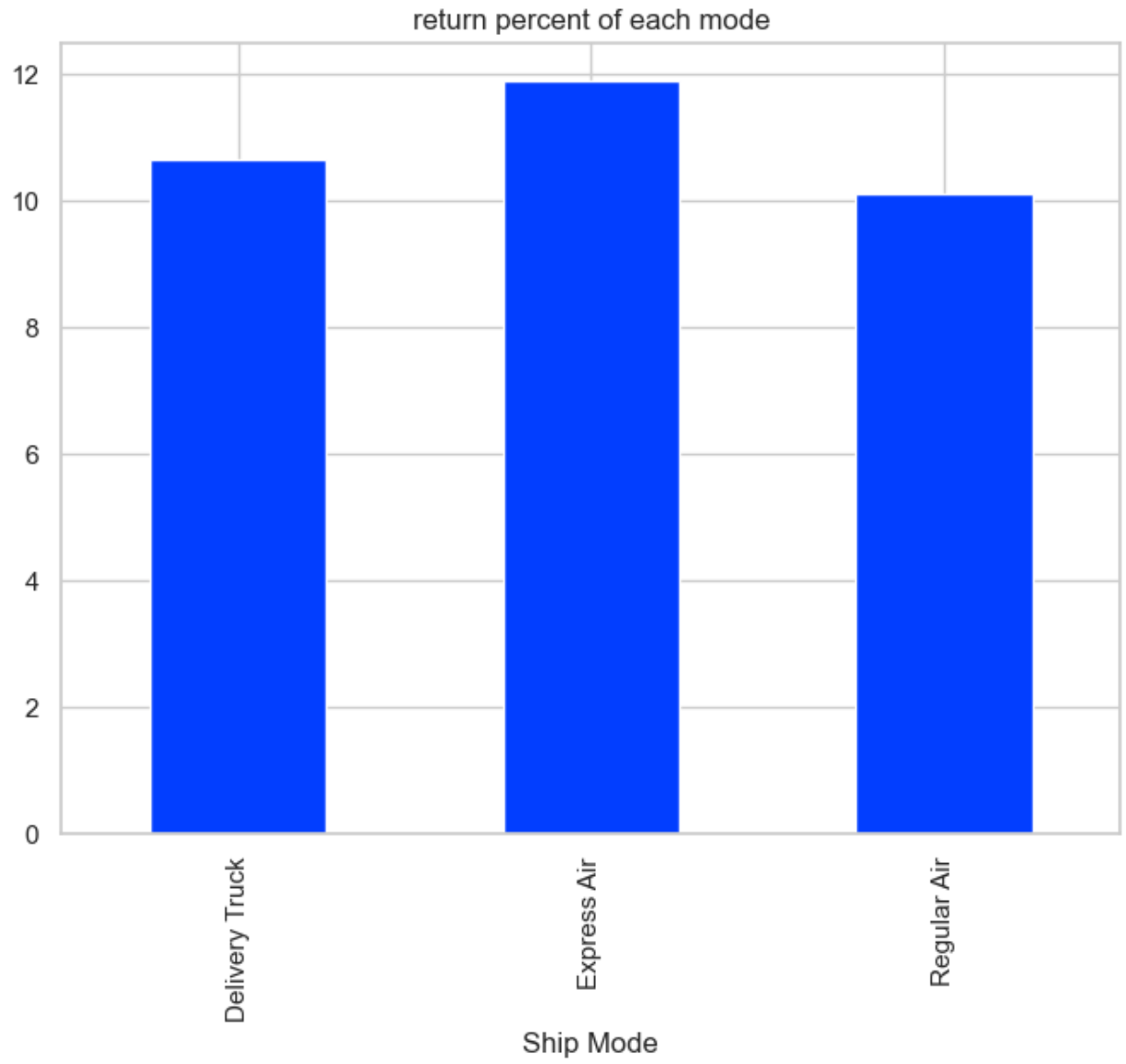
2-is the problem in a certain ship mode?

```
mode_return=superstore_sales.groupby(['Ship Mode','Status'])['Order ID'].count().unstack()
mode_return.plot(kind='bar' ,title='returns to sold for modes',figsize=(8,6))
```



from the first look, regular air has the largest returns but, it also has the largest sold orders.
so the calculations would involve a ratio between the returned and sold orders.

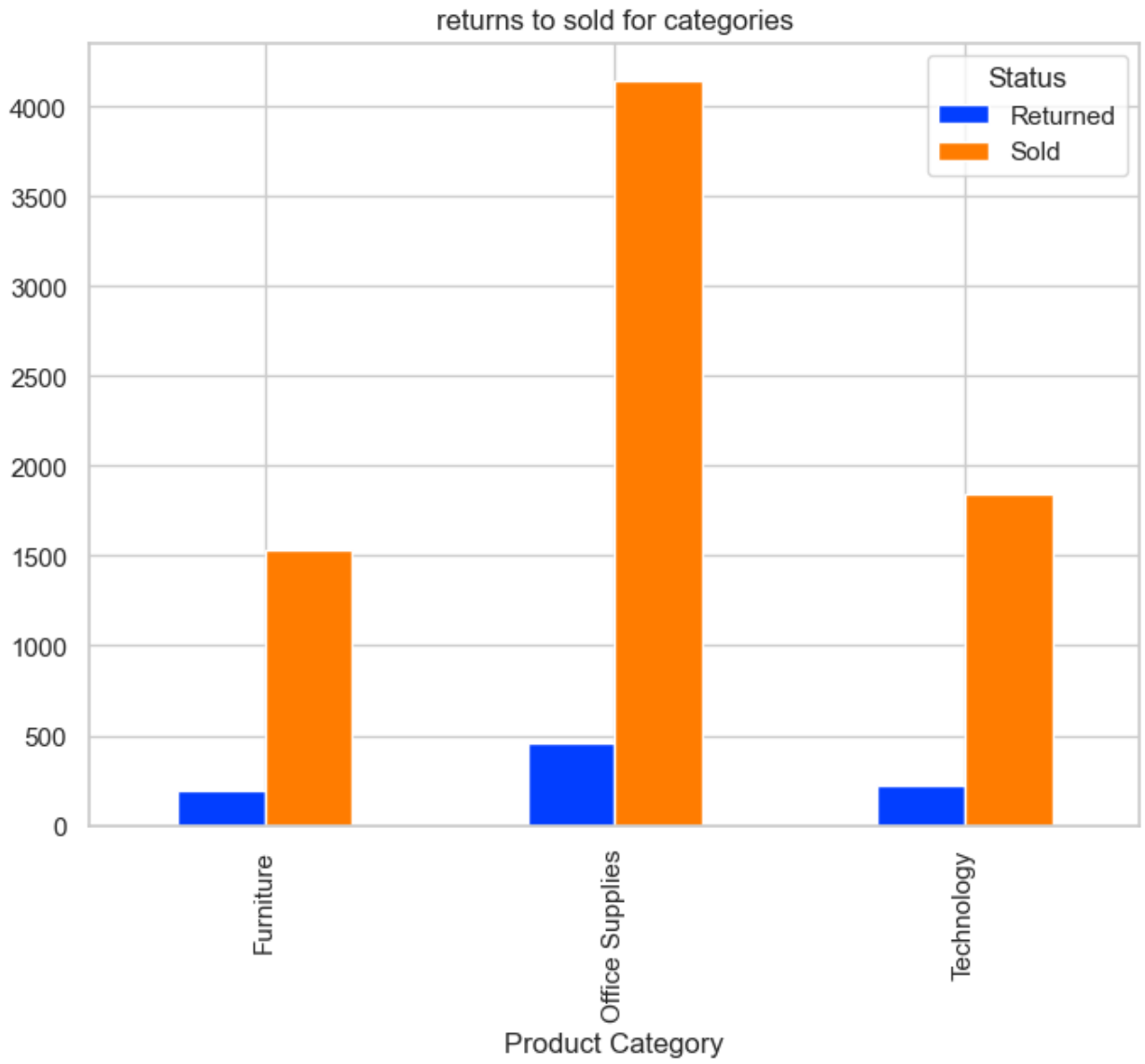
```
mode_returnavg=100*sales_return.groupby('Ship Mode')['Order ID'].count() / superstore_sales.groupby('Ship Mode')['Order ID'].count()
mode_returnavg.plot(kind='bar', title='return ratio of each mode', figsize=(8, 6))
```



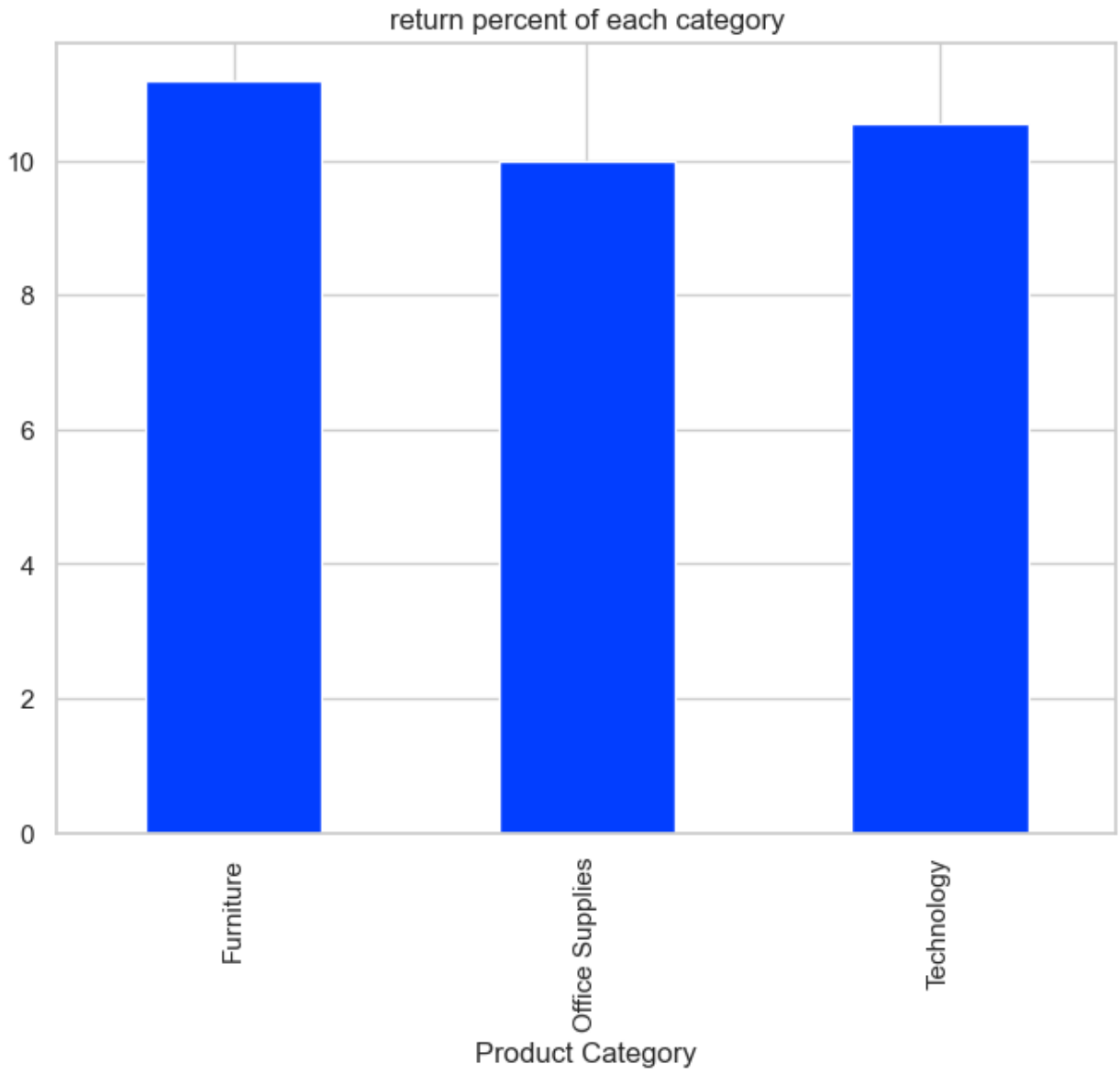
after looking at the ratios it seems like express air has the largest ratio and regular air has the lowest

3-is the problem in a certain Category?

```
categ_return=superstore_sales.groupby(['Product Category','Status'])['Order ID'].count().unstack()
categ_return.plot(kind='bar' ,title='returns to sold for categories',figsize=(8,6))
```



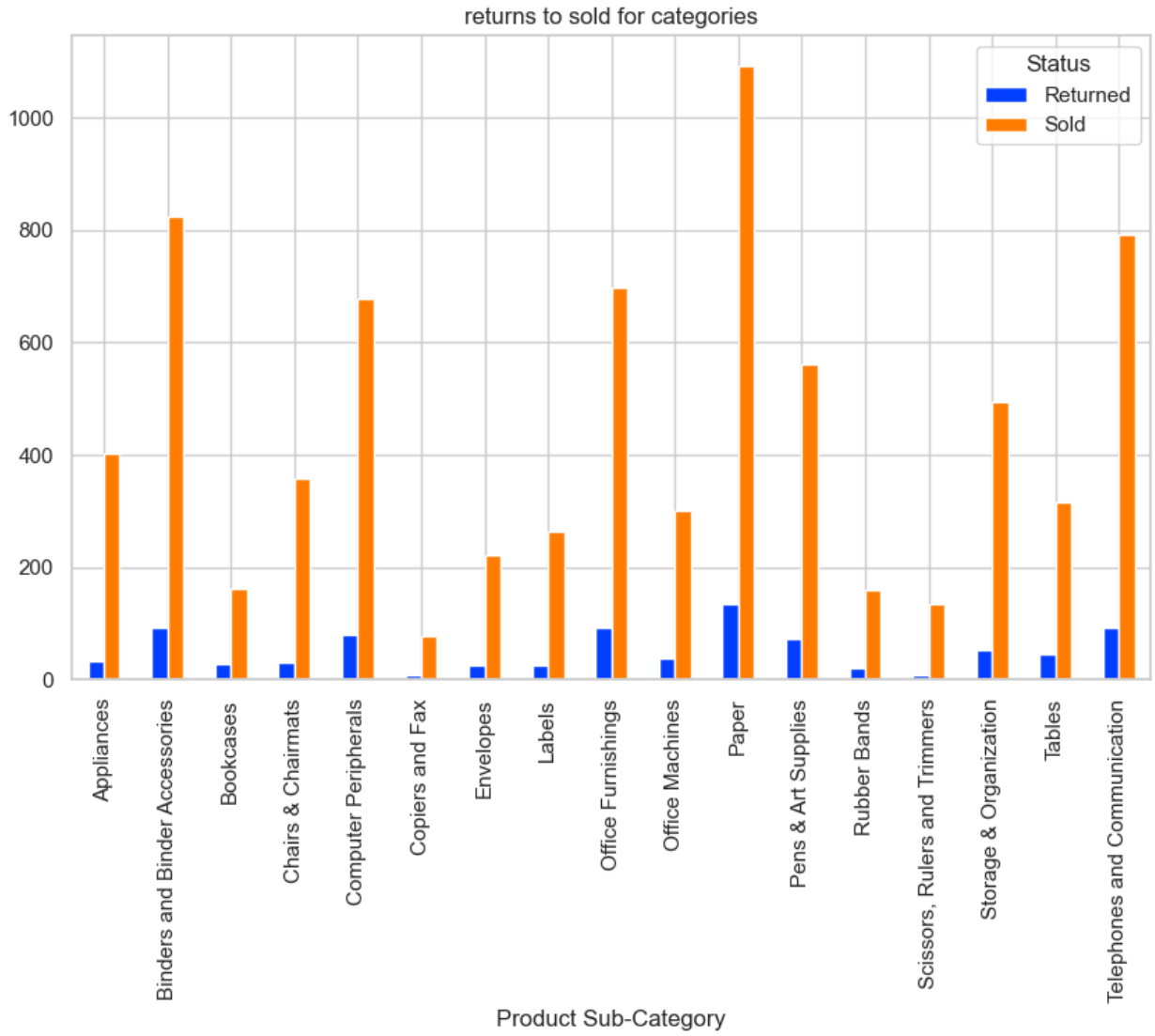

```
categ_returnavg=100*sales_return.groupby('Product Category')['Order ID'].count() /
superstore_sales.groupby('Product Category')['Order ID'].count()
categ_returnavg.plot(kind='bar', title='return ratio of each category', figsize=(8, 6))
```



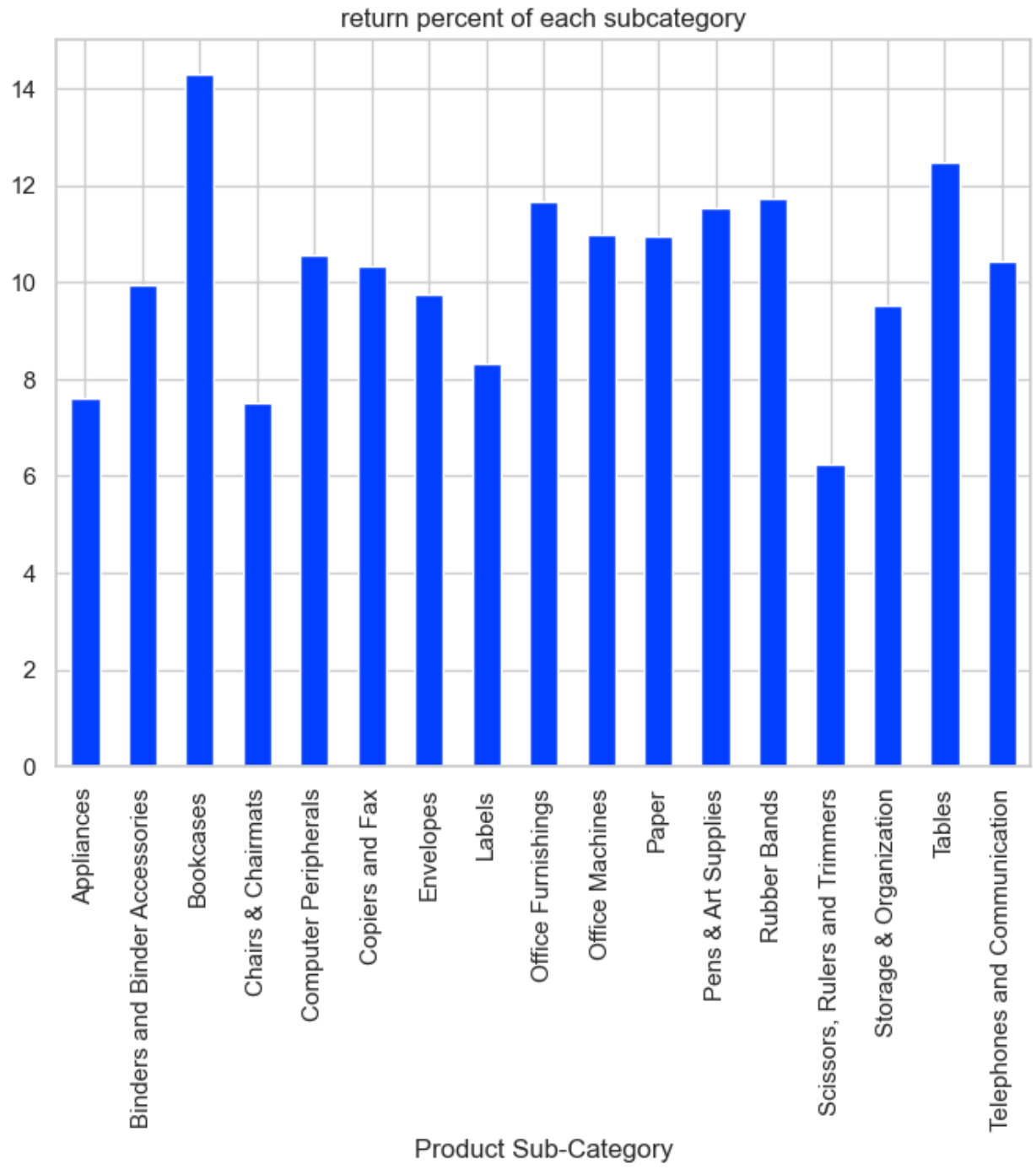
the percentages are close, no specific problem

4-is the problem in a certain SubCategory?

```
categ_return=superstore_sales.groupby(['Product Sub-Category','Status'])['Order ID'].count().unstack()
categ_return.plot(kind='bar' ,title='returns to sold for subcategories',figsize=(10,6))
```




```
subcateg_return=100*sales_return.groupby('Product Sub-Category')['Order ID'].count() /
superstore_sales.groupby('Product Sub-Category')['Order ID'].count()
subcateg_return.plot(kind='bar', title='return ratio of each subcategory', figsize=(8, 6))
```

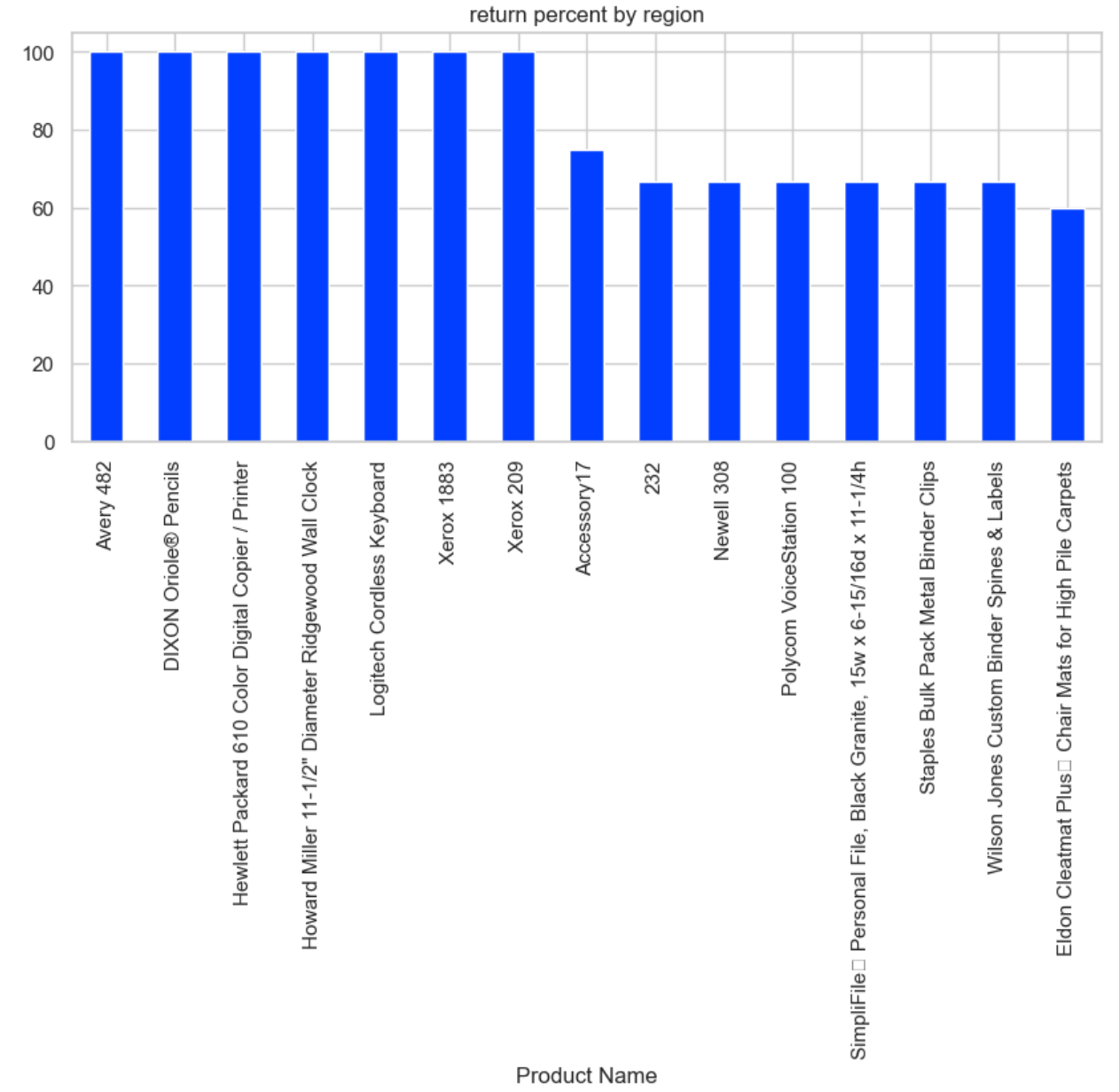


the returns of the bookcases are more than the normal
must find solution for it

5-what are the top Products returned?

```
return_prodavg=100*sales_return.groupby('Product Name')['Order ID'].count() /
superstore_sales.groupby('Product Name')['Order ID'].count()
```

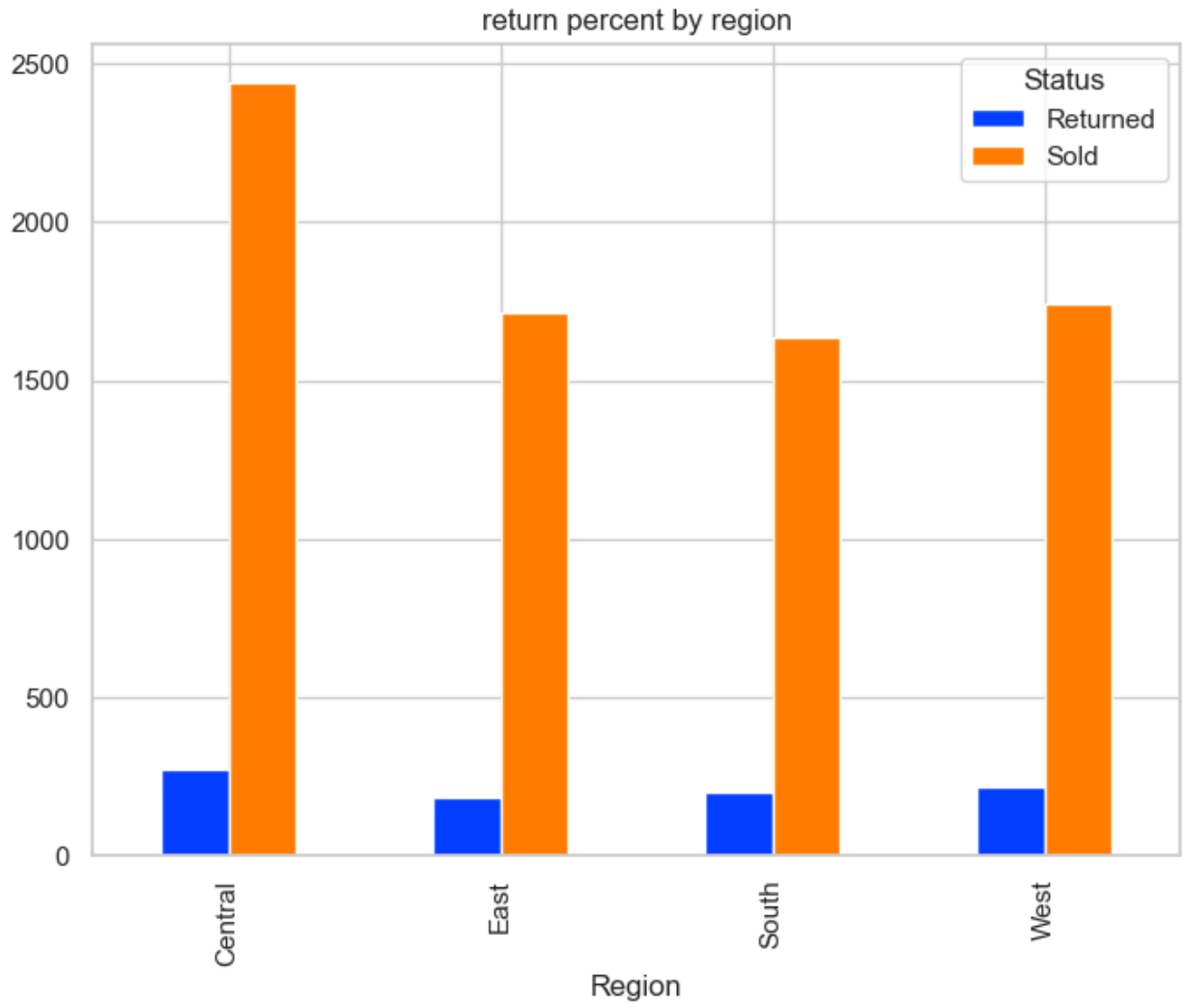
```
top10_prodavg=return_prodavg.nlargest(15)
top10_prodavg.plot(kind='bar', title='largest return ratio products', figsize=(10, 4))
```



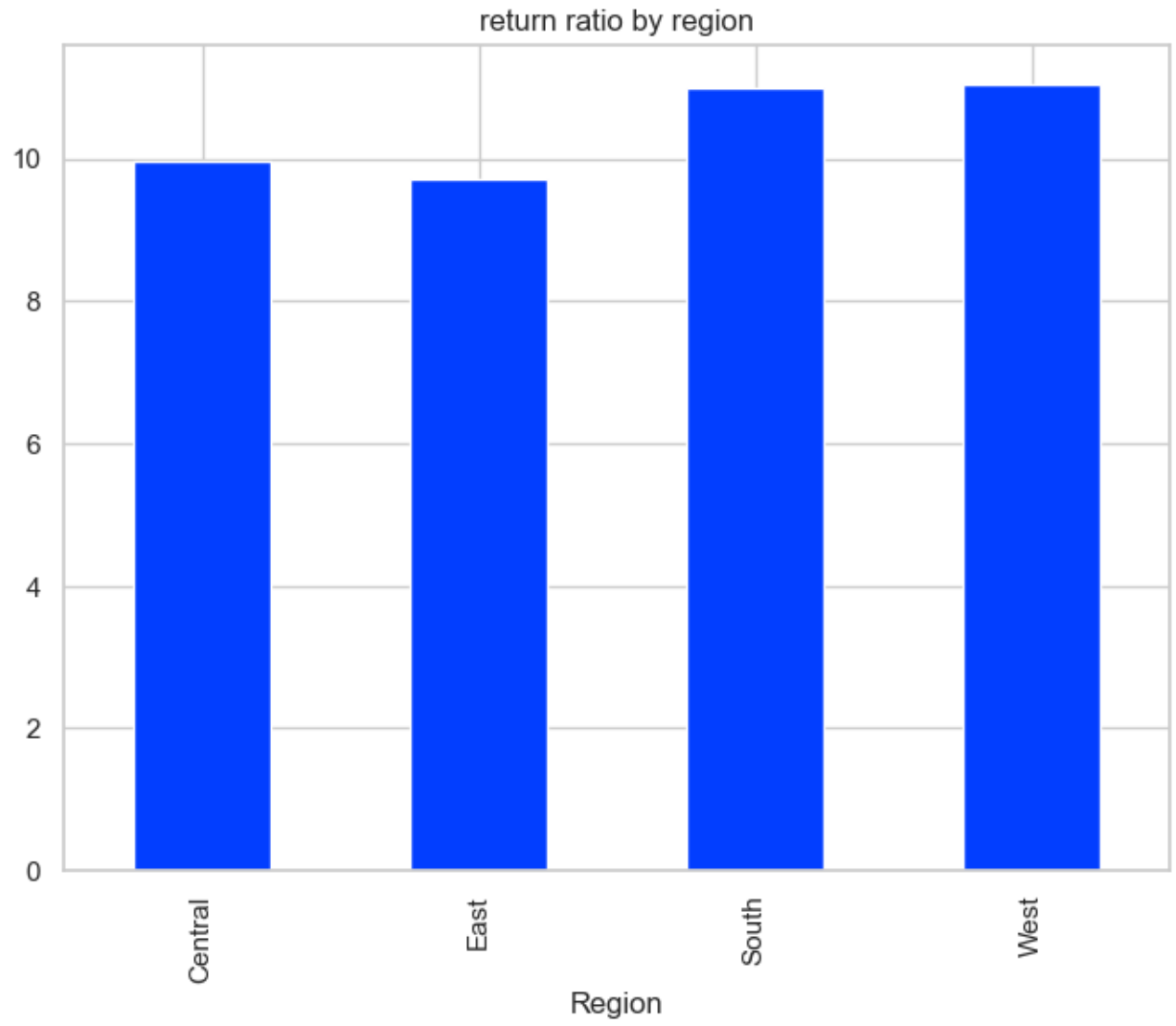
7 products are returned with 100% which means they haven't been sold a single item

6-is the problem in a certain Region?

```
region_return=superstore_sales.groupby(['Region','Status'])['Order ID'].count().unstack()
region_return.plot(kind='bar', title='returns to sold for region', figsize=(8, 6))
```

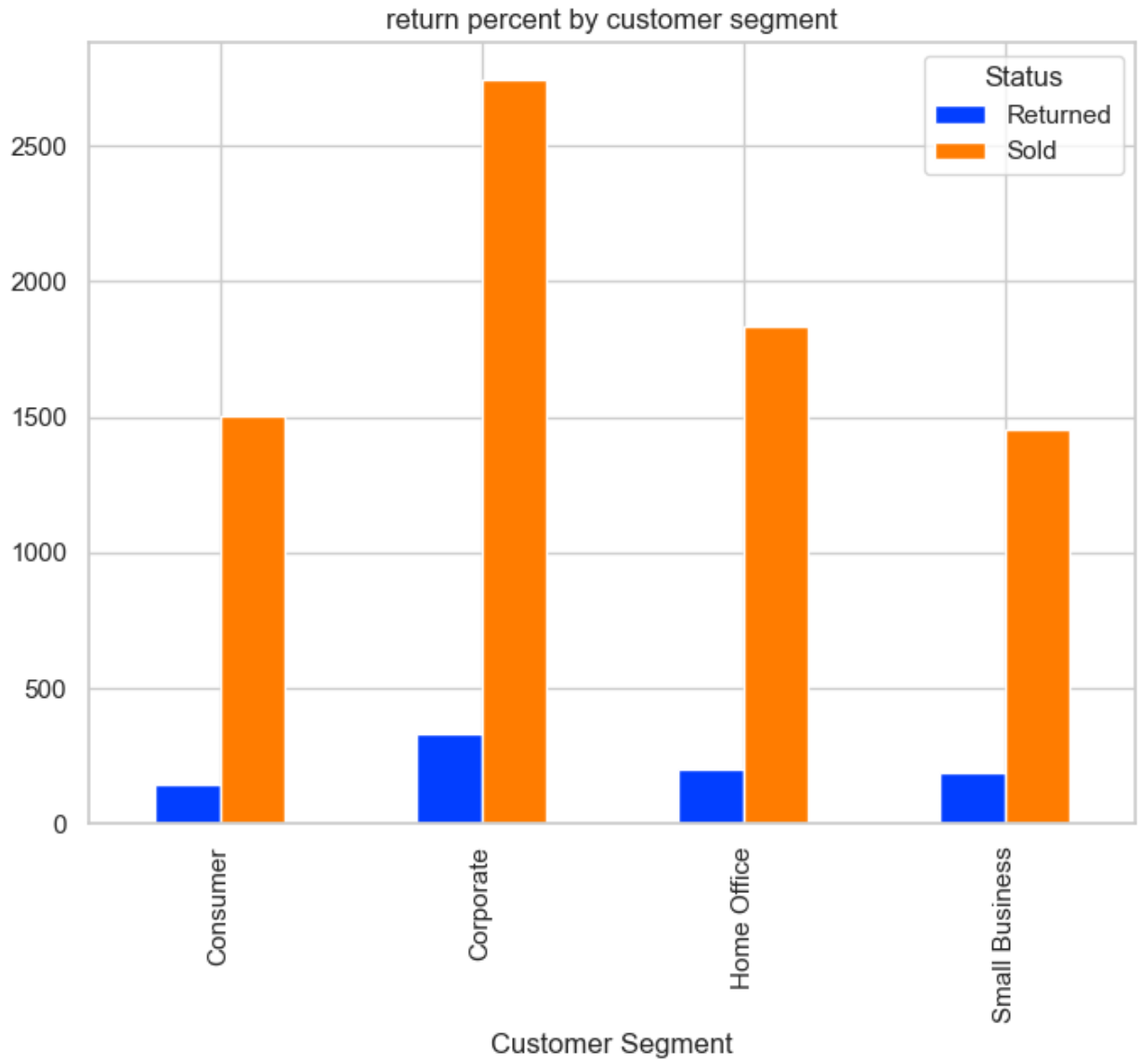



```
region_returnavg=100*sales_return.groupby('Region')['Order ID'].count() /
superstore_sales.groupby('Region')['Order ID'].count()
region_returnavg.plot(kind='bar' ,title='return ratio by region', figsize=(8, 6))
```



the percentages are close, no specific problem

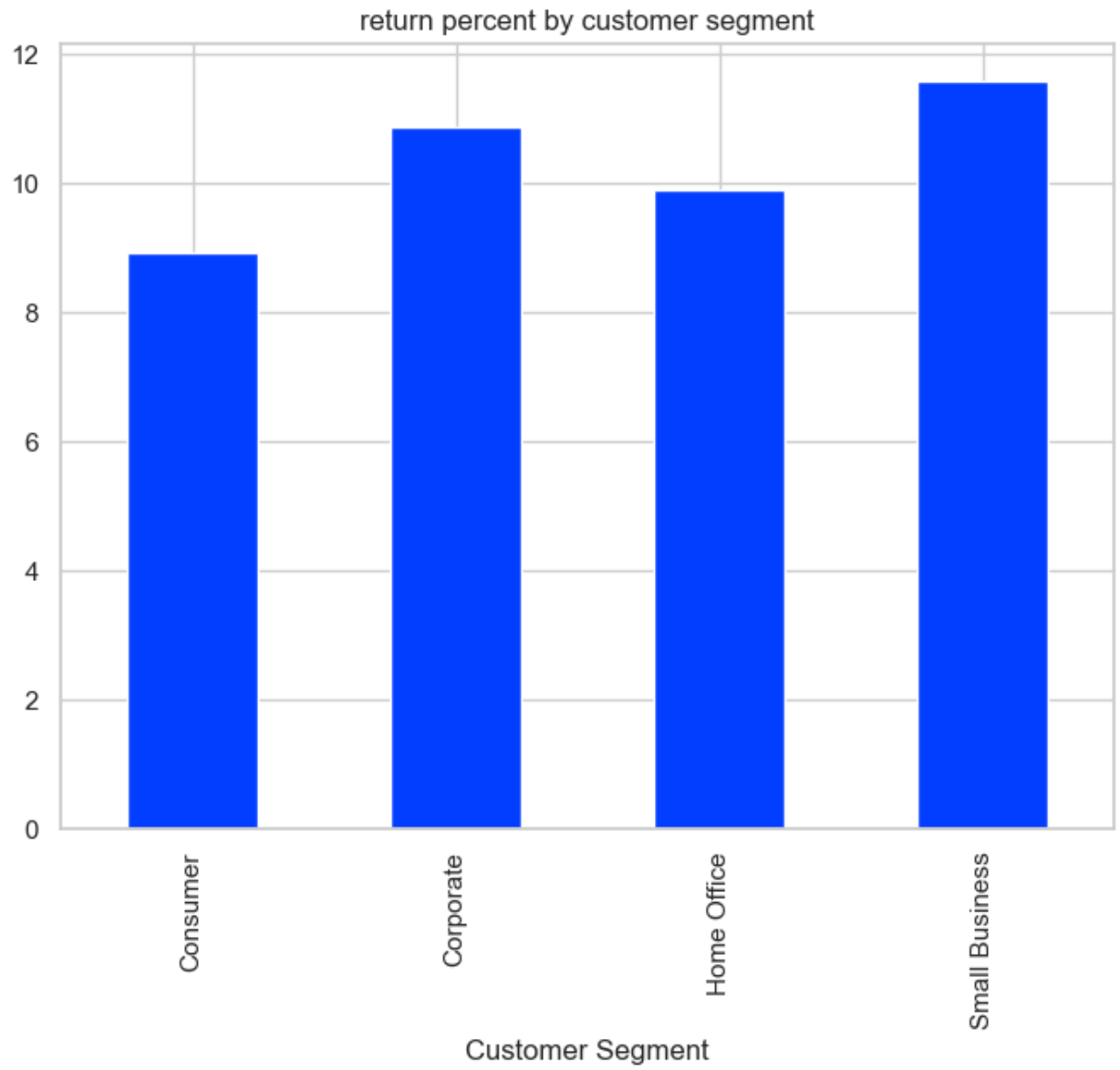
7-is the problem in a certain Customer segment?




```
segment_returnavg=100*sales_return.groupby('Customer Segment')['Order ID'].count() /
superstore_sales.groupby('Customer Segment')['Order ID'].count()
segment_returnavg.plot(kind='bar', title='return ratio by customer segment', figsize=(8, 6))
```

the difference between the largest and smallest is slightly big
so, it is required to see if their are problems with the services of small businesses

8-who are the most customer returning orders?



the difference between the largest and smallest is slightly big

so, it is required to see if their are problems with the services of small businesses

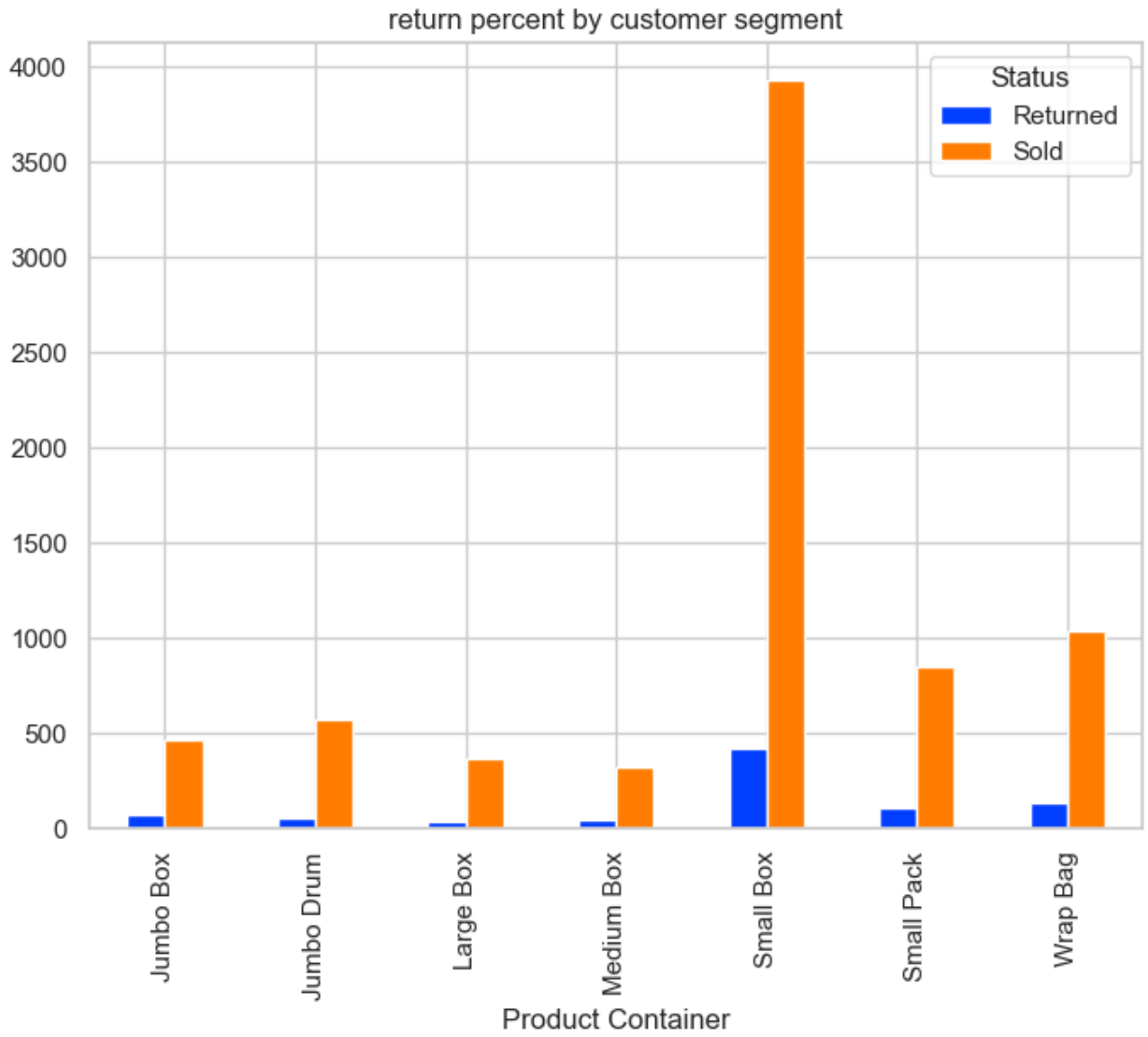
8-who are the most customer returning orders?

```
return_customavg=100*sales_return.groupby('Customer Name')['Order ID'].count() /
superstore_sales.groupby('Customer Name')['Order ID'].count()
top10_customavg=return_customavg.nlargest(15)
top10_customavg.plot(kind='bar', title='largest return ratio customers', figsize=(10, 4))
```

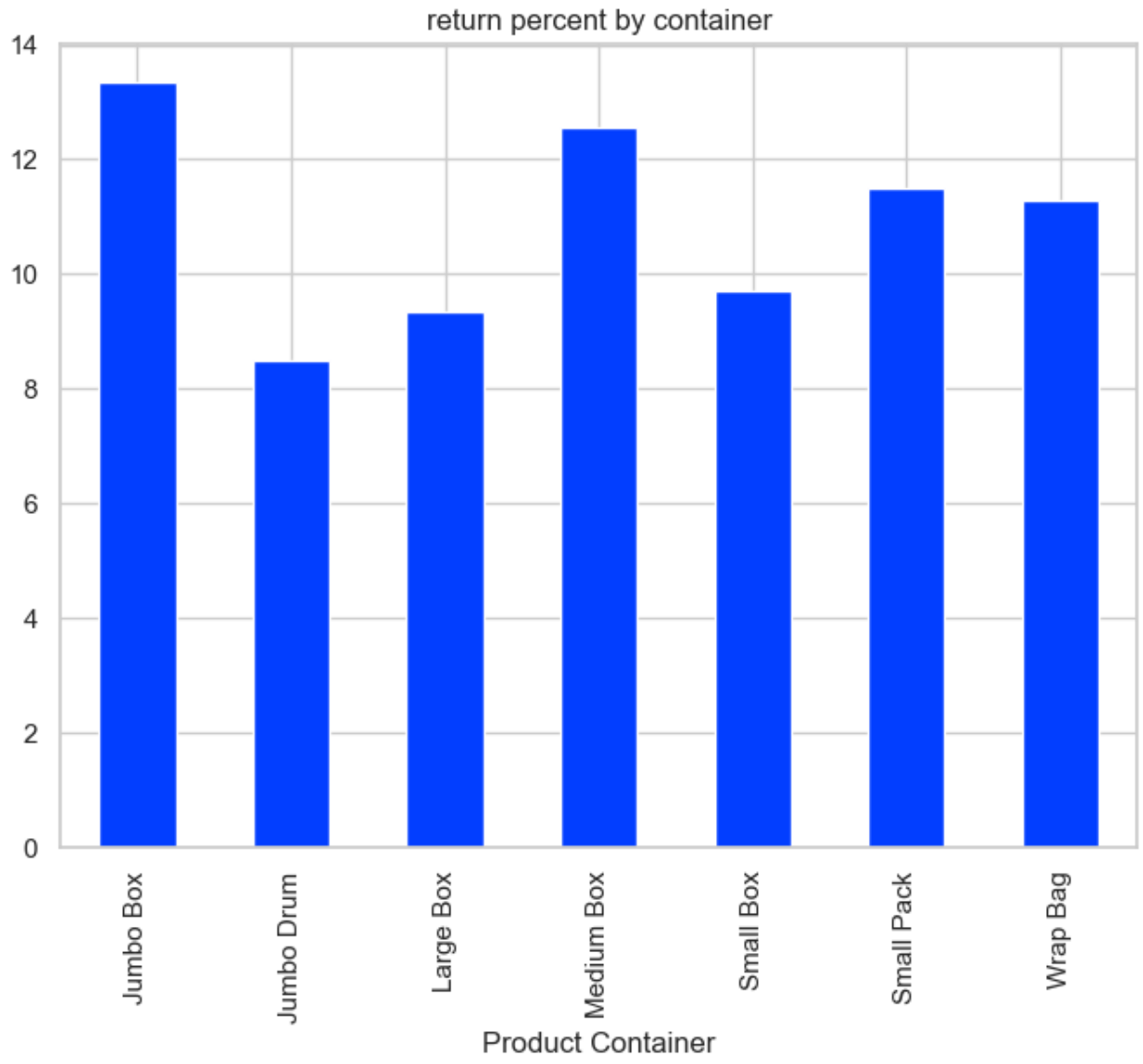
Rob lucas is returning all the orders, we return to the service see what is the problem
for the rest in the list we can call them asking about the problems they challenge and try fixing them

9-is the problem in a certain container?

```
container_return=superstore_sales.groupby(['Product Container','Status'])['Order ID'].count().unstack()
container_return.plot(kind='bar', title='returns to sold for containers', figsize=(8, 6))
```



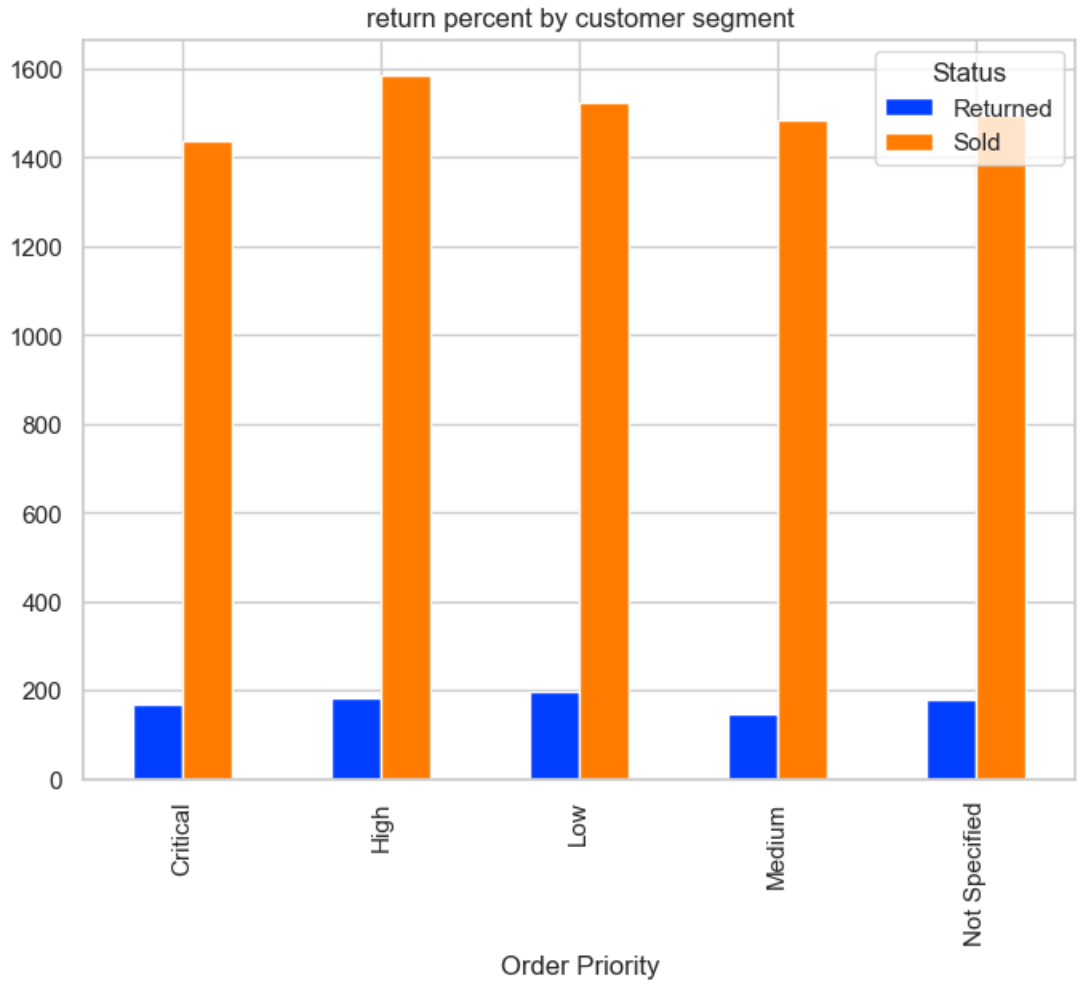

```
container_returnavg=100*sales_return.groupby('Product Container')['Order ID'].count() /
superstore_sales.groupby('Product Container')['Order ID'].count()
container_returnavg.plot(kind='bar', title='return ratio by container', figsize=(8, 6))
```


return percent by container
14
12
10
8
6
4
2
0
Jumbo Box
Jumbo Drum
Large Box
Medium Box
Small Box
Small Pack
Wrap Bag
Product Container

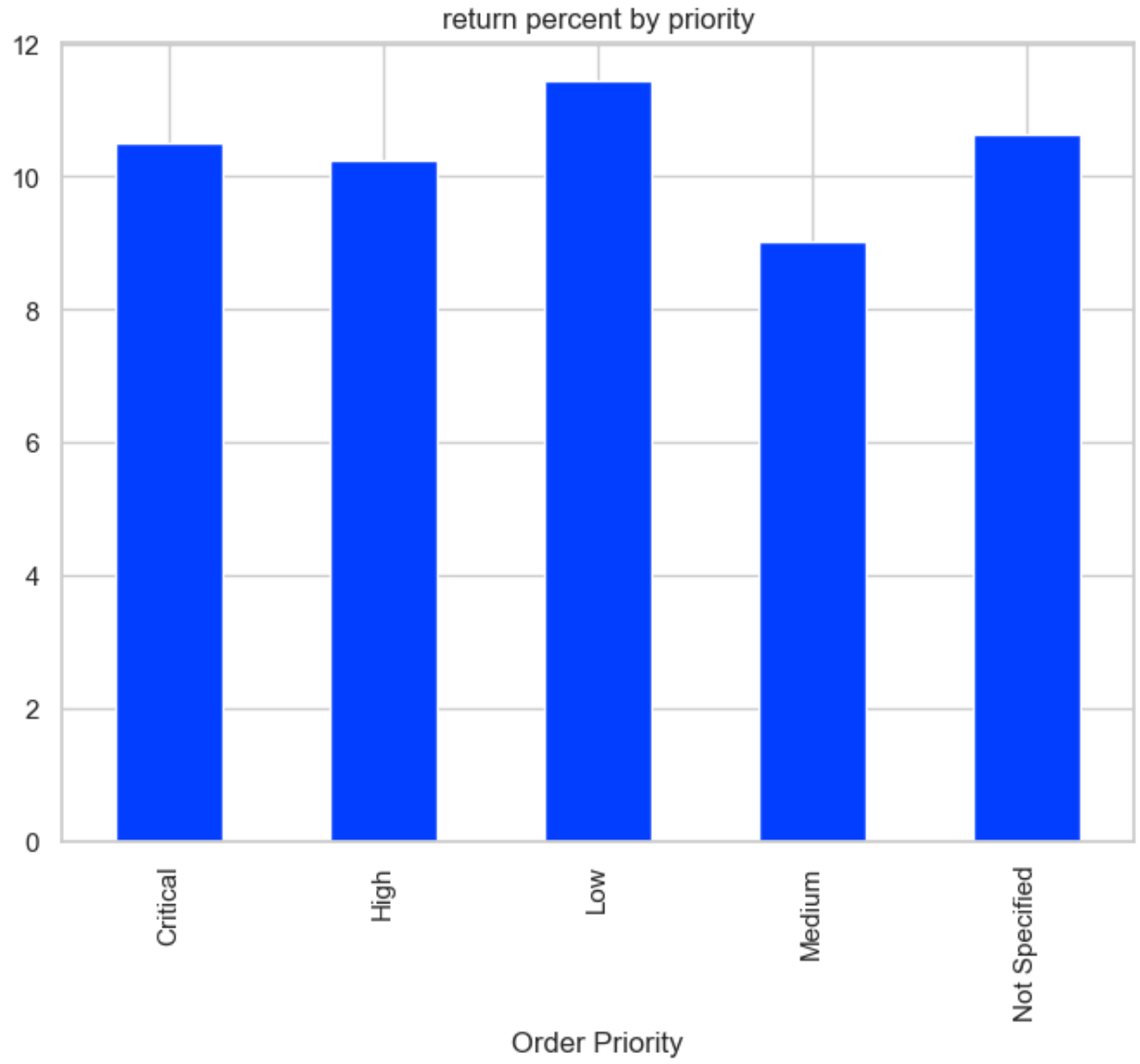
it is clear that the nubers for the jumbo box and medium box are relatively high
it must be put under consideration

10-is the problem in a certain priority?

```
priority_return=superstore_sales.groupby(['Order Priority','Status'])['Order ID'].count().unstack()
priority_return.plot(kind='bar', title='returns to sold for priority', figsize=(8, 6))
```



```
priority_return=100*sales_return.groupby('Order Priority')['Order ID'].count() /
superstore_sales.groupby('Order Priority')['Order ID'].count()
priority_return.plot(kind='bar', title='return ratio by priority', figsize=(8, 6))
```



the low priority objects have the largest percent(predictable)

Final step

Tableau Visualization

1. Sales Performance

Sales Growth: Sales have shown steady growth, peaking in 2020, with the strongest performance in the last quarter.

Seasonal Impact: There is a noticeable increase in sales during the holiday season, particularly in December.

Top Product Categories: Technology is the top-selling category, followed by Furniture and Office Supplies.

Best/Worst Cities: Madison and San Francisco are top-performing cities, while Scituate and Tinton Falls have the lowest sales.

Regional Performance: The Central region leads in sales, with the South region lagging behind.

Effect of Discounts: Discounts increase sales volume but reduce overall revenue due to smaller profit margins.

2. Customer Behavior

Top Customer Segments: Corporate and Consumer segments contribute the most to sales.

Retention Rates: Customer retention is fairly stable across regions, with some decline in the South.

Customer Lifetime Value (CLV): Corporate clients have the highest CLV, making them valuable long-term customers.

At-Risk Customers: Certain customers, such as Christine Kargatis, may stop purchasing soon based on their recent activity.

3. Product and Profitability Insights

Key Revenue Drivers: Technology is the highest revenue-generating category.

Profitability: The Central region and Technology category deliver the highest profit margins

https://public.tableau.com/views/DEPIfinalprojectGroupADashboard/Dashboard1?:language=en-US&publish=yes&:sid=&:redirect=auth&:display_count=n&:origin=viz_share_link