

```
In [55]: # Import Libraries:  
import numpy as np  
import pandas as pd  
from matplotlib import pyplot as plt  
from matplotlib import style  
import seaborn as sns
```

Importing CSV file

In [56]: # Read the CSV file from the file C drive
df = pd.read_csv(r"C:\Users\HP\Downloads\Superstore data.csv")
df

Out[56]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	
0	1	CA-2013-152156	09-11-2013	12-11-2013	Second Class	CG-12520	Claire Gute	Consumer	United States	Hend
1	2	CA-2013-152156	09-11-2013	12-11-2013	Second Class	CG-12520	Claire Gute	Consumer	United States	Hend
2	3	CA-2013-138688	13-06-2013	17-06-2013	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los An
3	4	US-2012-108966	11-10-2012	18-10-2012	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Laude
4	5	US-2012-108966	11-10-2012	18-10-2012	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Laude
...
9989	9990	CA-2011-110422	22-01-2011	24-01-2011	Second Class	TB-21400	Tom Boeckenhauer	Consumer	United States	
9990	9991	CA-2014-121258	27-02-2014	04-03-2014	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa
9991	9992	CA-2014-121258	27-02-2014	04-03-2014	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa
9992	9993	CA-2014-121258	27-02-2014	04-03-2014	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa
9993	9994	CA-2014-119914	05-05-2014	10-05-2014	Second Class	CC-12220	Chris Cortes	Consumer	United States	Westm

9994 rows × 20 columns



In [57]: *#Information about the each column and its data types*
df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 20 columns):
 #   Column            Non-Null Count  Dtype  
--- 
 0   Row ID             9994 non-null    int64  
 1   Order ID           9991 non-null    object  
 2   Order Date         9993 non-null    object  
 3   Ship Date          9993 non-null    object  
 4   Ship Mode          9991 non-null    object  
 5   Customer ID        9994 non-null    object  
 6   Customer Name      9993 non-null    object  
 7   Segment             9994 non-null    object  
 8   Country             9994 non-null    object  
 9   City                9994 non-null    object  
 10  State               9994 non-null    object  
 11  Postal Code        9994 non-null    int64  
 12  Region              9994 non-null    object  
 13  Product ID          9994 non-null    object  
 14  Category            9994 non-null    object  
 15  Sub-Category        9994 non-null    object  
 16  Sales               9989 non-null    float64 
 17  Quantity            9994 non-null    int64  
 18  Discount            9994 non-null    float64 
 19  Profit              9994 non-null    float64 
dtypes: float64(3), int64(3), object(14)
memory usage: 1.5+ MB
```

Cleaning Process

```
In [58]: # It shows the number of number of null values in the each column  
null_counts = df.isnull().sum()  
print("Rows with null values:")  
null_counts
```

Rows with null values:

```
Out[58]: Row ID      0  
Order ID      3  
Order Date    1  
Ship Date     1  
Ship Mode     3  
Customer ID   0  
Customer Name 1  
Segment        0  
Country        0  
City           0  
State          0  
Postal Code   0  
Region         0  
Product ID    0  
Category       0  
Sub-Category   0  
Sales          5  
Quantity       0  
Discount       0  
Profit         0  
dtype: int64
```

```
In [59]: # Drop the null- values rows  
df.dropna(inplace=True)
```

In [60]: `df.info()`

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 9983 entries, 0 to 9993
Data columns (total 20 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Row ID            9983 non-null    int64  
 1   Order ID          9983 non-null    object  
 2   Order Date        9983 non-null    object  
 3   Ship Date         9983 non-null    object  
 4   Ship Mode         9983 non-null    object  
 5   Customer ID      9983 non-null    object  
 6   Customer Name    9983 non-null    object  
 7   Segment           9983 non-null    object  
 8   Country           9983 non-null    object  
 9   City               9983 non-null    object  
 10  State              9983 non-null    object  
 11  Postal Code       9983 non-null    int64  
 12  Region             9983 non-null    object  
 13  Product ID        9983 non-null    object  
 14  Category           9983 non-null    object  
 15  Sub-Category      9983 non-null    object  
 16  Sales              9983 non-null    float64 
 17  Quantity           9983 non-null    int64  
 18  Discount           9983 non-null    float64 
 19  Profit              9983 non-null    float64 
dtypes: float64(3), int64(3), object(14)
memory usage: 1.6+ MB
```

In [61]: `# know the unique value in the each column
df.nunique(axis=0)`

Out[61]:

Row ID	9983
Order ID	5003
Order Date	1238
Ship Date	1333
Ship Mode	4
Customer ID	793
Customer Name	793
Segment	3
Country	1
City	531
State	49
Postal Code	631
Region	4
Product ID	1862
Category	3
Sub-Category	17
Sales	5819
Quantity	14
Discount	12
Profit	7281
dtype: int64	

In [62]: #Add Profit margin column in :

```
df['Profit margin'] = df['Profit'] / df['Sales']*100
df
```

Out[62]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	
0	1	CA-2013-152156	09-11-2013	12-11-2013	Second Class	CG-12520	Claire Gute	Consumer	United States	Hend
1	2	CA-2013-152156	09-11-2013	12-11-2013	Second Class	CG-12520	Claire Gute	Consumer	United States	Hend
2	3	CA-2013-138688	13-06-2013	17-06-2013	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los An
3	4	US-2012-108966	11-10-2012	18-10-2012	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Laude
4	5	US-2012-108966	11-10-2012	18-10-2012	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Laude
...
9989	9990	CA-2011-110422	22-01-2011	24-01-2011	Second Class	TB-21400	Tom Boeckenhauer	Consumer	United States	TM
9990	9991	CA-2014-121258	27-02-2014	04-03-2014	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa
9991	9992	CA-2014-121258	27-02-2014	04-03-2014	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa
9992	9993	CA-2014-121258	27-02-2014	04-03-2014	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa
9993	9994	CA-2014-119914	05-05-2014	10-05-2014	Second Class	CC-12220	Chris Cortes	Consumer	United States	Westm

9983 rows × 21 columns



In [63]: # it shows the description of all numerical values in the data

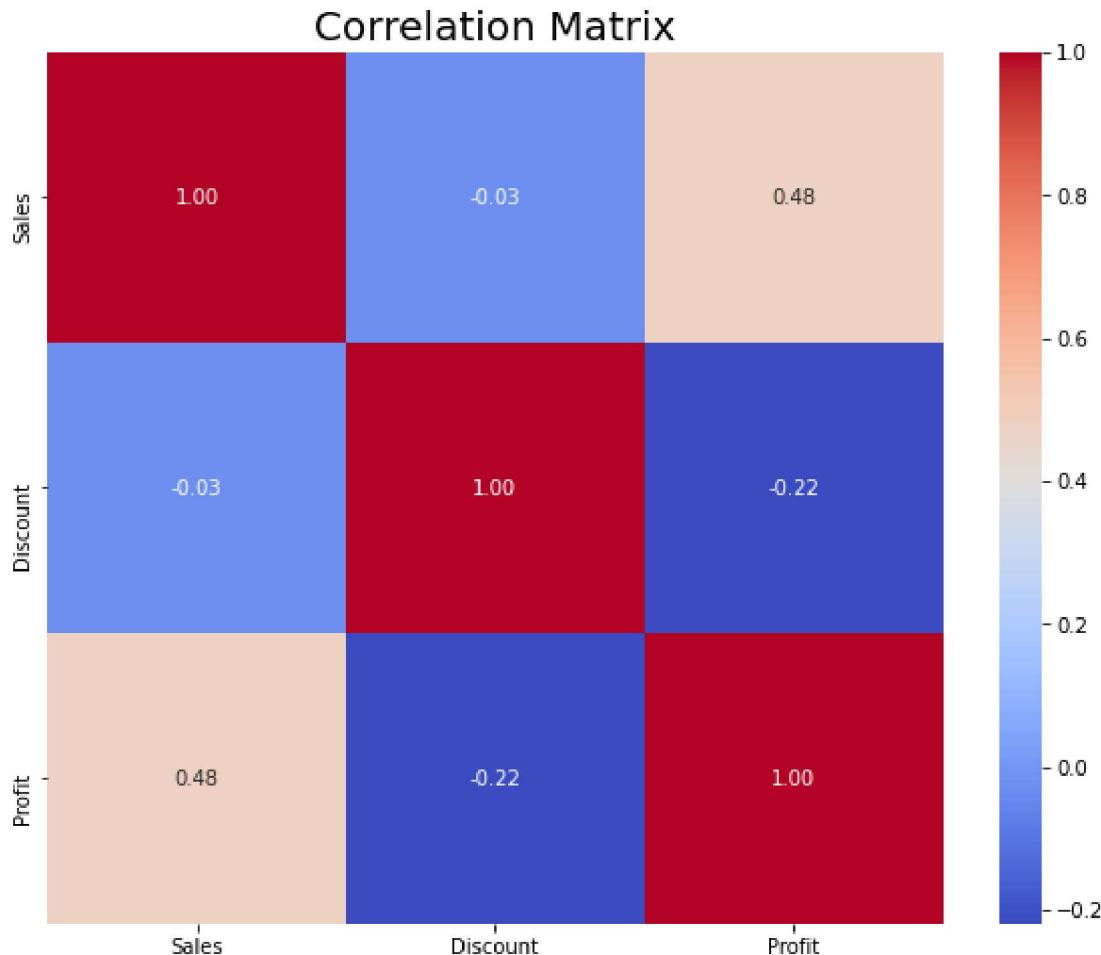
```
description = df[['Sales', 'Quantity', 'Discount', 'Profit','Profit margin']].  
description
```

Out[63]:

	Sales	Quantity	Discount	Profit	Profit margin
count	9983.000000	9983.000000	9983.000000	9983.000000	9983.000000
mean	229.619321	3.789242	0.156174	28.644303	12.042697
std	623.318674	2.225167	0.206439	234.376183	46.659218
min	0.444000	1.000000	0.000000	-6599.978000	-275.000000
25%	17.248000	2.000000	0.000000	1.727100	7.500000
50%	54.384000	3.000000	0.200000	8.643600	27.000000
75%	209.880000	5.000000	0.200000	29.349600	36.250000
max	22638.480000	14.000000	0.800000	8399.976000	50.000000

In [64]: # Correlation of sales, discount and profit

```
correlation_matrix = df[['Sales', 'Discount', 'Profit']].corr()  
plt.figure(figsize=(10, 8))  
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")  
plt.title('Correlation Matrix', fontsize=20)  
plt.show()
```



Valuable Insights

```
In [118]: # Top 10 Best Selling Products:  
best_selling_products = df.groupby('Product ID')['Sales'].sum().sort_values(as  
print('Top 10 Best Selling Products:')
```

Top 10 Best Selling Products:

```
Out[118]: Product ID  
TEC-CO-10004722      61599.824  
OFF-BI-10003527      27453.384  
TEC-MA-10002412      22638.480  
FUR-CH-10002024      21870.576  
OFF-BI-10001359      19823.479  
OFF-BI-10000545      19024.500  
TEC-CO-10001449      18839.686  
TEC-MA-10001127      18374.895  
OFF-BI-10004995      17965.068  
OFF-SU-10000151      17030.312  
Name: Sales, dtype: float64
```

```
In [66]: # Top 10 Customers by Sales:  
top_customers = df.groupby('Customer Name')['Sales'].sum().sort_values(ascendi  
print('Top 10 Customers Based on Sales:')
```

Top 10 Customers Based on Sales:

```
Customer Name  
Sean Miller          25043.050  
Tamara Chand         19052.218  
Raymond Buch         15117.339  
Tom Ashbrook         14595.620  
Adrian Barton        14473.571  
Ken Lonsdale         14175.229  
Sanjit Chand         14142.334  
Hunter Lopez          12873.298  
Sanjit Engle          12209.438  
Christopher Conant    12129.072  
Name: Sales, dtype: float64
```

```
In [67]: # Top 10 most profitable and 10 least profitable cities:  
most_profitable_cities = df.groupby('City')['Profit'].sum().sort_values(ascending=False).head(10)  
least_profitable_cities = df.groupby('City')['Profit'].sum().sort_values().head(10)  
print('Top 10 Most Profitable Cities:')  
print(most_profitable_cities)  
print('\n10 Least Profitable Cities:')  
print(least_profitable_cities)
```

Top 10 Most Profitable Cities:

City	Profit
New York City	62026.0741
Los Angeles	30287.0919
Seattle	28858.1232
San Francisco	17489.4262
Detroit	13181.7908
Lafayette	10018.3876
Jackson	7581.6828
Atlanta	6993.6629
Minneapolis	6824.5846
San Diego	6377.1960

Name: Profit, dtype: float64

10 Least Profitable Cities:

City	Profit
Philadelphia	-13837.7674
Houston	-10153.5485
San Antonio	-7299.0502
Lancaster	-7239.0684
Chicago	-6654.5688
Burlington	-3622.8772
Dallas	-2846.5257
Phoenix	-2790.8832
Aurora	-2691.7386
Jacksonville	-2323.8350

Name: Profit, dtype: float64

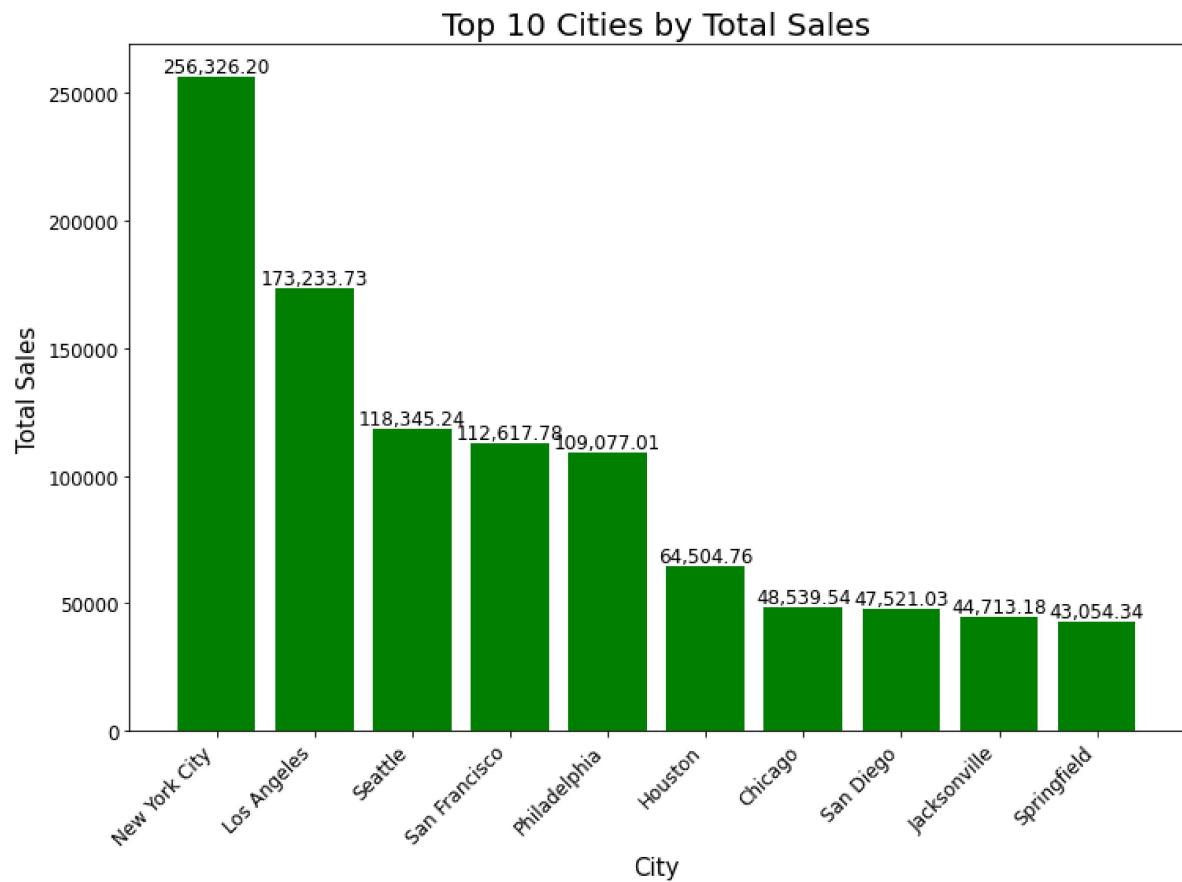
```
In [85]: # Grouping by City and calculating the sum of Sales, then sorting by Sales in
top_cities = df.groupby(['City'], as_index=False)[ 'Sales' ].sum().sort_values(b

# Plotting the bar chart
plt.figure(figsize=(12, 8))
bars = plt.bar(top_cities['City'], top_cities['Sales'], color='green')

plt.xlabel('City', fontsize=15)
plt.ylabel('Total Sales', fontsize=15)
plt.title('Top 10 Cities by Total Sales', fontsize=20)
plt.xticks(rotation=45, ha='right', fontsize=12)
plt.yticks(fontsize=12)

# Adding data labels to the bars
for bar in bars:
    plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height(),
             f'{bar.get_height():,.2f}', ha='center', va='bottom', fontsize=12)

plt.show()
```



```
In [69]: segment_wise_quantity = df.groupby('Segment')['Quantity'].sum()
print("Total Quantity sold by each Segment :")
print(segment_wise_quantity)
print()
segment_wise_sales = df.groupby('Segment')['Sales'].sum()
print("Total Sales by Segments :")
print(segment_wise_sales)
print()
segment_wise_sales = df.groupby('Segment')['Profit'].sum()
print("Total Profit by Segments :")
print(segment_wise_sales)
```

Total Quantity sold by each Segment :

Segment
Consumer 19483
Corporate 11606
Home Office 6739
Name: Quantity, dtype: int64

Total Sales by Segments :

Segment
Consumer 19483
Corporate 11606
Home Office 6739
Name: Sales, dtype: int64

Total Profit by Segments :

Segment
Consumer 19483
Corporate 11606
Home Office 6739
Name: Profit, dtype: int64

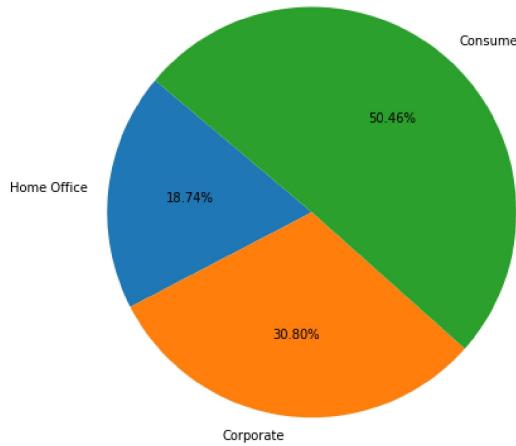
In [70]: #Sales and profit distribution by Segment

```
segment_wise_sales = df.groupby('Segment')['Sales'].sum().sort_values(ascending=True)
segment_wise_profit = df.groupby('Segment')['Profit'].sum().sort_values(ascending=True)
fig, axs = plt.subplots(1, 2, figsize=(16, 8))
axs[0].pie(segment_wise_sales, labels=segment_wise_sales.index, autopct='%.1f%%')
axs[0].set_title('Segment-wise Sales Distribution', fontsize=16)

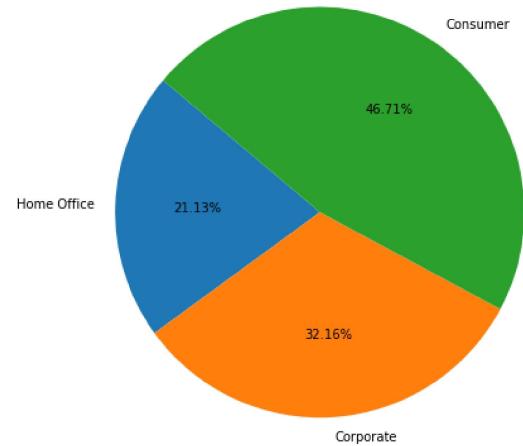
axs[1].pie(segment_wise_profit, labels=segment_wise_profit.index, autopct='%.1f%%')
axs[1].set_title('Segment-wise Profit Distribution', fontsize=16)

plt.show()
```

Segment-wise Sales Distribution



Segment-wise Profit Distribution



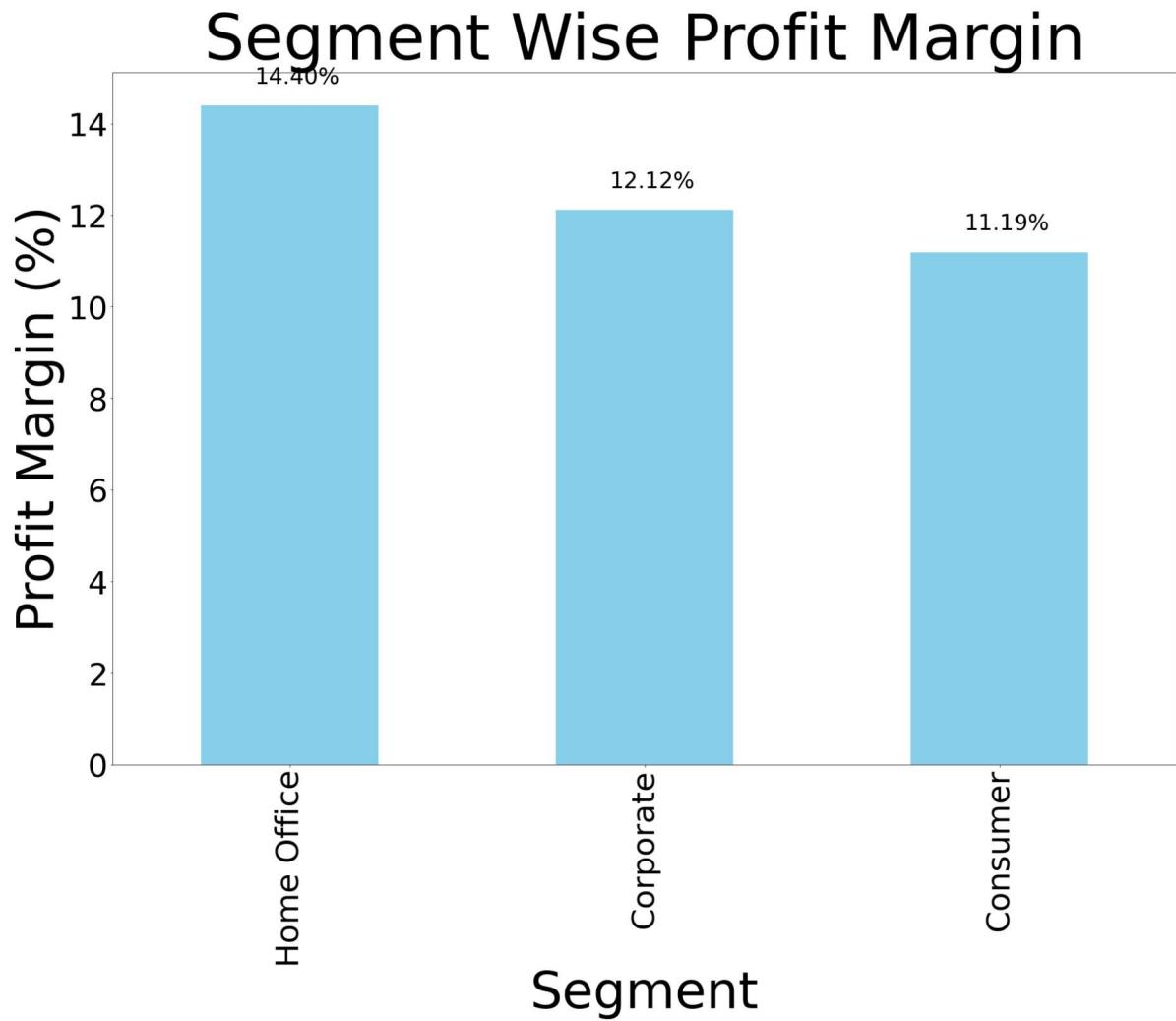
```
In [71]: # Segment wise profit margin :
```

```
df['Profit Margin'] = (df['Profit'] / df['Sales']) * 100

segment_profit_margin = df.groupby('Segment')['Profit Margin'].mean().sort_values()
plt.figure(figsize=(30,20))
bars = segment_profit_margin.plot(kind='bar', color='skyblue')
plt.xlabel('Segment', fontsize=80)
plt.ylabel('Profit Margin (%)', fontsize=80)
plt.title('Segment Wise Profit Margin', fontsize=100)
plt.xticks(fontsize=50)
plt.yticks(fontsize=50)

for bar in bars.patches:
    plt.text(bar.get_x() + bar.get_width() / 2 - 0.1, bar.get_height() + 0.5,
             f'{bar.get_height():.2f}%', fontsize=35, color='black')

plt.show()
```



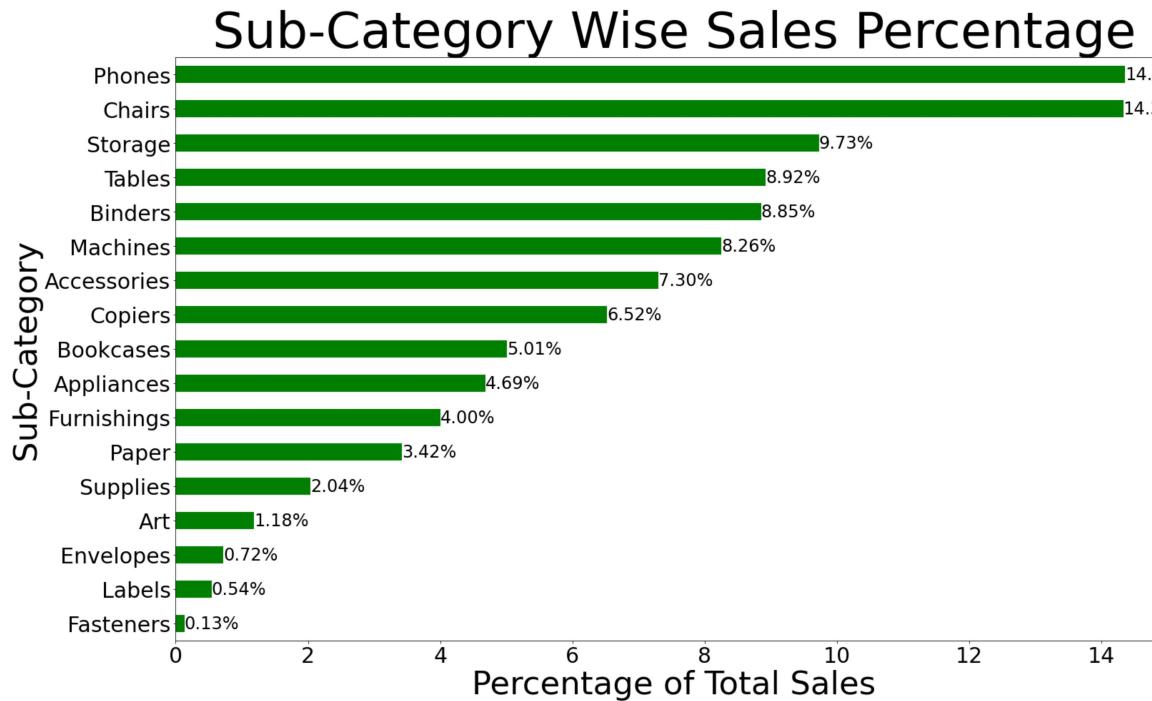
In [72]: #Sub-category wise sales and its percentage:

```
sub_category_wise_sales = df.groupby('Sub-Category')['Sales'].sum().sort_values()
print('Sub-category wise Sales :')
print()
print(sub_category_wise_sales)

total_sales = df['Sales'].sum()
sub_category_wise_sales_percentage = (df.groupby('Sub-Category')['Sales'].sum() / total_sales) * 100
plt.figure(figsize=(25, 15))
ax = sub_category_wise_sales_percentage.sort_values(ascending=True).plot(kind='bar')
for rect in ax.patches:
    width = rect.get_width()
    plt.text(width, rect.get_y() + rect.get_height() / 2, f'{width:.2f}%', ha='left', va='center')
plt.xlabel('Percentage of Total Sales', fontsize=45)
plt.ylabel('Sub-Category', fontsize=45)
plt.title('Sub-Category Wise Sales Percentage', fontsize=70)
plt.xticks(fontsize=30)
plt.yticks(fontsize=30)
plt.show()
```

Sub-category wise Sales :

Sub-Category	Sales
Phones	329067.6380
Chairs	328449.1030
Storage	222951.1680
Tables	204471.8180
Binders	202953.4450
Machines	189238.6310
Accessories	167380.3180
Copiers	149528.0300
Bookcases	114879.9963
Appliances	107463.3510
Furnishings	91663.2040
Paper	78463.6540
Supplies	46673.5380
Art	27118.7920
Envelopes	16476.4020
Labels	12486.3120
Fasteners	3024.2800
Name: Sales, dtype: float64	

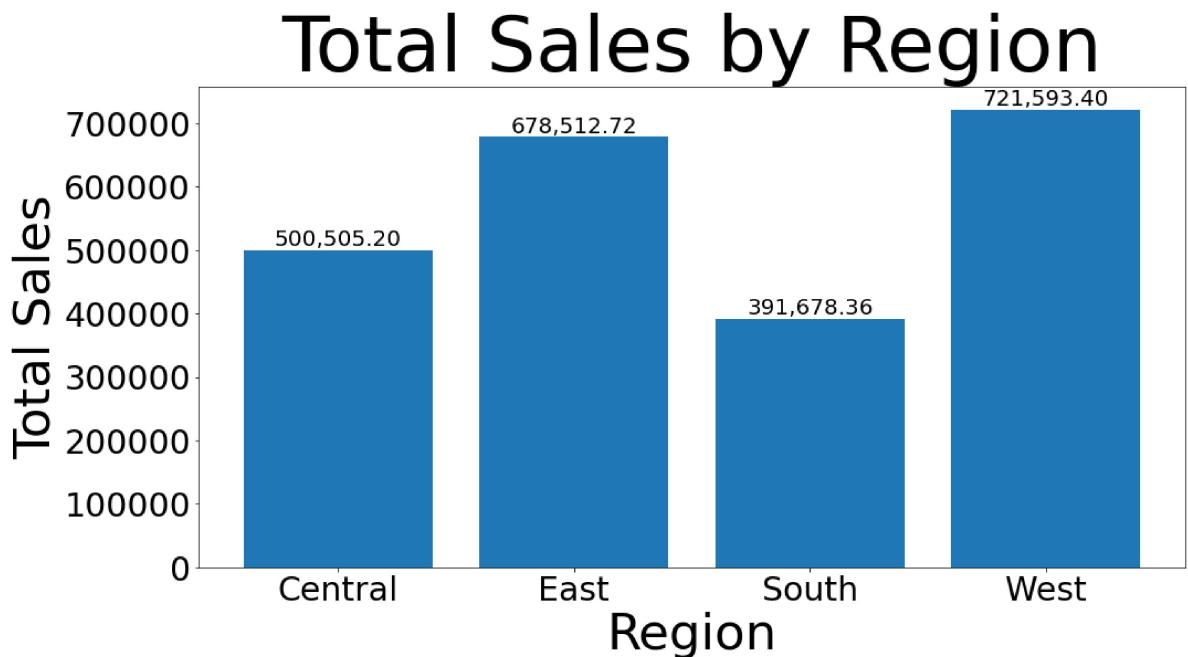


```
In [73]: plt.figure(figsize=(16, 8))
d1 = df.groupby('Region', as_index=False)[['Sales']].sum()
bars = plt.bar(d1['Region'], d1['Sales'])

plt.xlabel('Region', fontsize=45)
plt.ylabel('Total Sales', fontsize=45)
plt.title('Total Sales by Region', fontsize=70)
plt.xticks(fontsize=30)
plt.yticks(fontsize=30)

# Adding data labels to the bars
for bar in bars:
    plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height(),
             f'{bar.get_height():,.2f}', ha='center', va='bottom', fontsize=20)

plt.show()
```



```
In [74]: import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

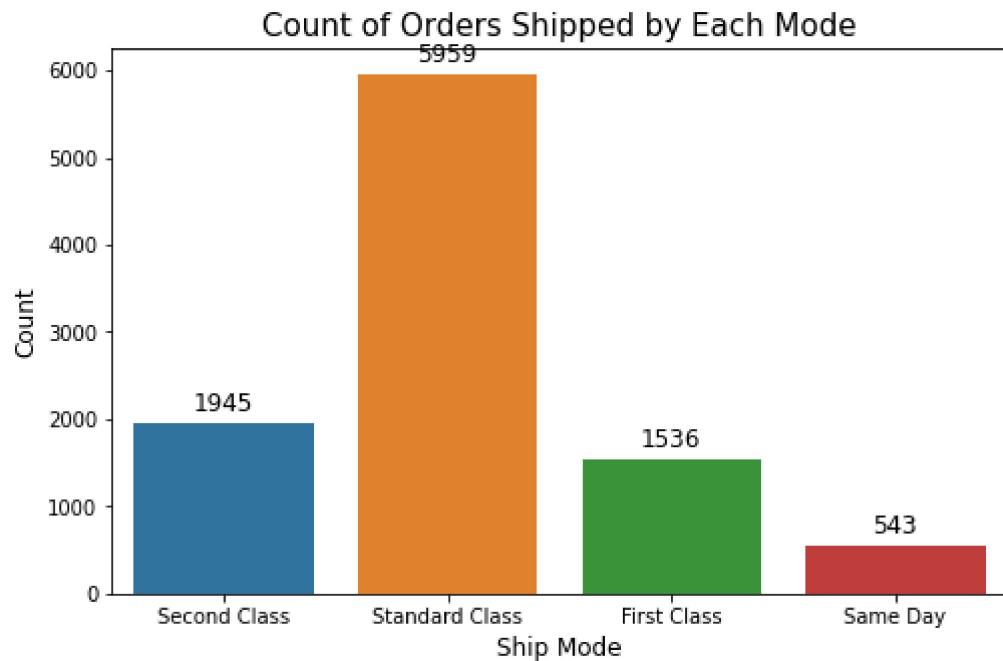
# Assuming you have a DataFrame named df with a column 'Ship Mode'

# Plotting the count plot
plt.figure(figsize=(8, 5))
ax = sns.countplot(x='Ship Mode', data=df)

plt.title('Count of Orders Shipped by Each Mode', fontsize=15)
plt.xlabel('Ship Mode', fontsize=12)
plt.ylabel('Count', fontsize=12)
plt.xticks(fontsize=10)
plt.yticks(fontsize=10)

# Adding data labels to the bars
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_he
        ha='center', va='center', fontsize=12, color='black', xytext=(textcoords='offset points'))

plt.show()
```

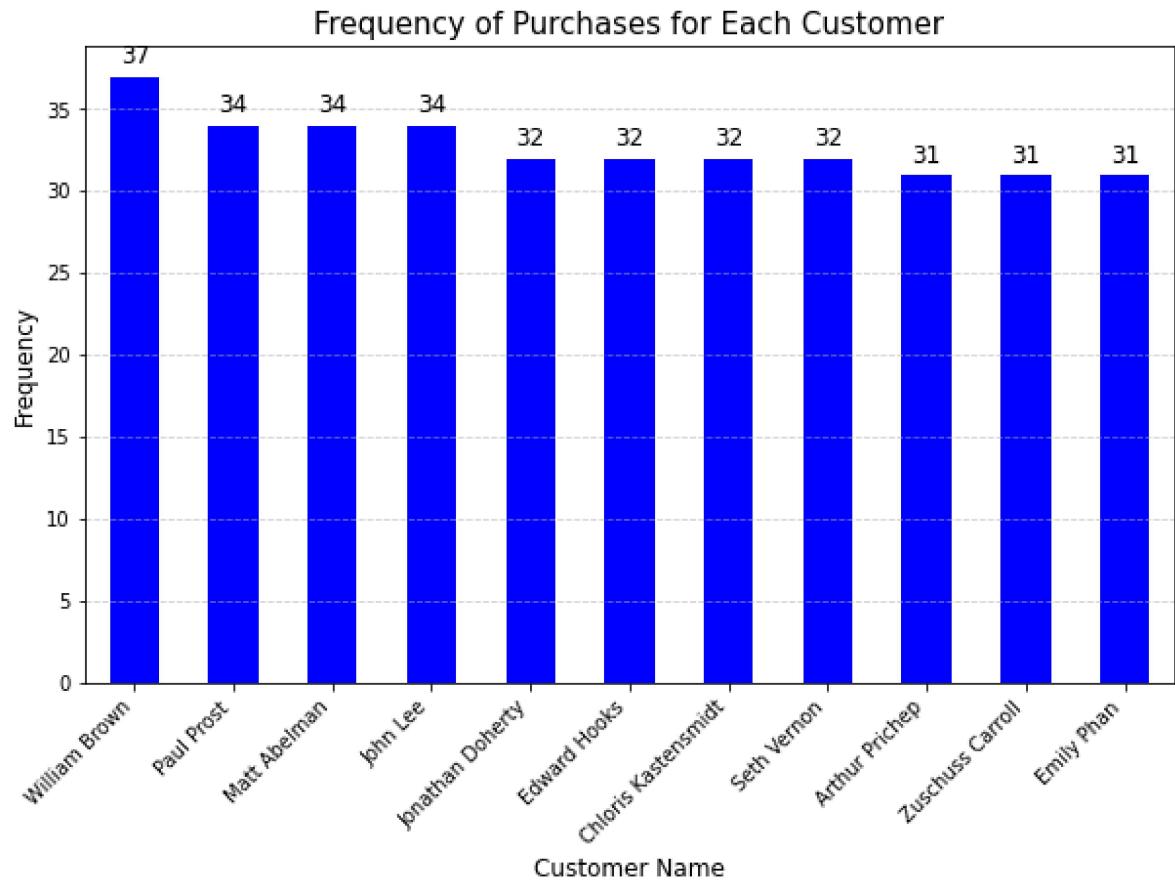


```
In [75]: # Plotting the bar chart
plt.figure(figsize=(10, 6))
ax = df['Customer Name'].value_counts().head(11).plot(kind='bar', color='blue')

plt.title('Frequency of Purchases for Each Customer', fontsize=15)
plt.xlabel('Customer Name', fontsize=12)
plt.ylabel('Frequency', fontsize=12)
plt.xticks(rotation=45, ha='right', fontsize=10)
plt.yticks(fontsize=10)
plt.grid(axis='y', linestyle='--', alpha=0.6)

# Adding data labels to the bars
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height(),
                                      ha='center', va='center', fontsize=12, color='black', xytext=(0, 5),
                                      textcoords='offset points')

plt.show()
```



```
In [76]: grouped_data = df.groupby(['Region', 'Category']).agg({'Sales': 'sum'}).reset_index()

# Find the maximum sales for each Region
max_sales_by_region = grouped_data.groupby('Region')['Sales'].idxmax()

# Get the corresponding row for each maximum sales entry
top_selling_products = grouped_data.loc[max_sales_by_region]

top_selling_products
#print("Top selling products in each region:")
#print(top_selling_products[['Region', 'Category', 'Sales']])
```

Out[76]:

	Region	Category	Sales
2	Central	Technology	170416.312
5	East	Technology	264973.981
8	South	Technology	148743.916
11	West	Technology	251080.408

```
In [77]: import matplotlib.pyplot as plt
# Sub-category-wise sales
sub_category_wise_sales = df.groupby('Sub-Category')['Sales'].sum()
print(sub_category_wise_sales)

# Average discount per sub-category
avg_discount_per_sub_category = df.groupby('Sub-Category')['Discount'].sum()
print(avg_discount_per_sub_category)

# Calculate discount percentage per sub-category
discount_percentage_per_sub_category = (avg_discount_per_sub_category / sub_ca
```

```
Sub-Category
Accessories      167380.3180
Appliances       107463.3510
Art              27118.7920
Binders          202953.4450
Bookcases        114879.9963
Chairs           328449.1030
Copiers          149528.0300
Envelopes        16476.4020
Fasteners        3024.2800
Furnishings      91663.2040
Labels            12486.3120
Machines          189238.6310
Paper             78463.6540
Phones            329067.6380
Storage           222951.1680
Supplies          46673.5380
Tables            204471.8180
Name: Sales, dtype: float64

Sub-Category
Accessories      60.80
Appliances       76.80
Art              59.60
Binders          566.60
Bookcases        48.14
Chairs           105.00
Copiers          11.00
Envelopes        20.40
Fasteners        17.80
Furnishings      132.40
Labels            25.00
Machines          35.20
Paper             102.40
Phones            137.00
Storage           63.20
Supplies          14.60
Tables            83.15
Name: Discount, dtype: float64
```

```
In [78]: import pandas as pd
import matplotlib.pyplot as plt

# Assuming you have a DataFrame named df with columns 'OrderDate', 'Sales', and
df['Order Date'] = pd.to_datetime(df['Order Date'])
df['YearMonth'] = df['Order Date'].dt.to_period('M')

monthly_data = df.groupby('YearMonth').agg({'Sales': 'sum', 'Profit': 'sum'})

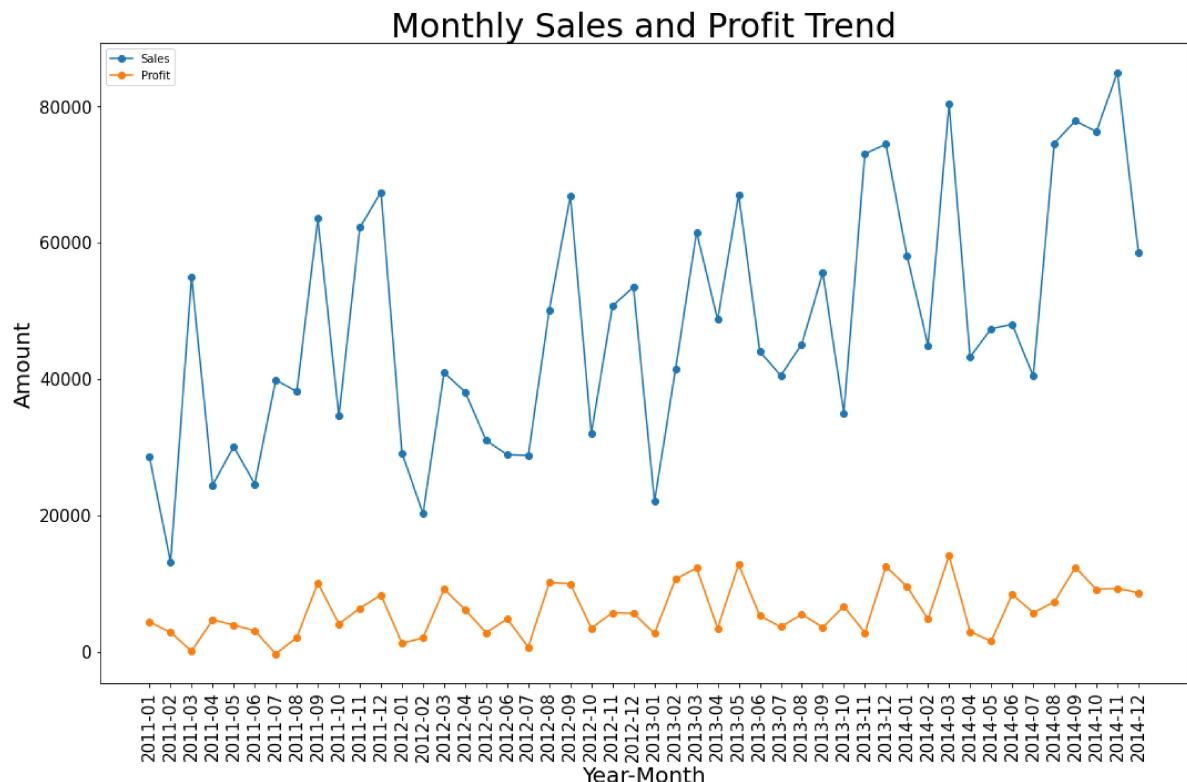
plt.figure(figsize=(15, 10))

plt.plot(monthly_data.index.astype(str), monthly_data['Sales'], marker='o', linestyle='solid')
plt.plot(monthly_data.index.astype(str), monthly_data['Profit'], marker='o', linestyle='solid')

plt.title('Monthly Sales and Profit Trend', fontsize=30)
plt.xlabel('Year-Month', fontsize=20)
plt.ylabel('Amount', fontsize=20)
plt.xticks(rotation=90, fontsize=15)
plt.yticks(rotation=0, fontsize=15)

plt.legend()

plt.tight_layout()
plt.show()
```

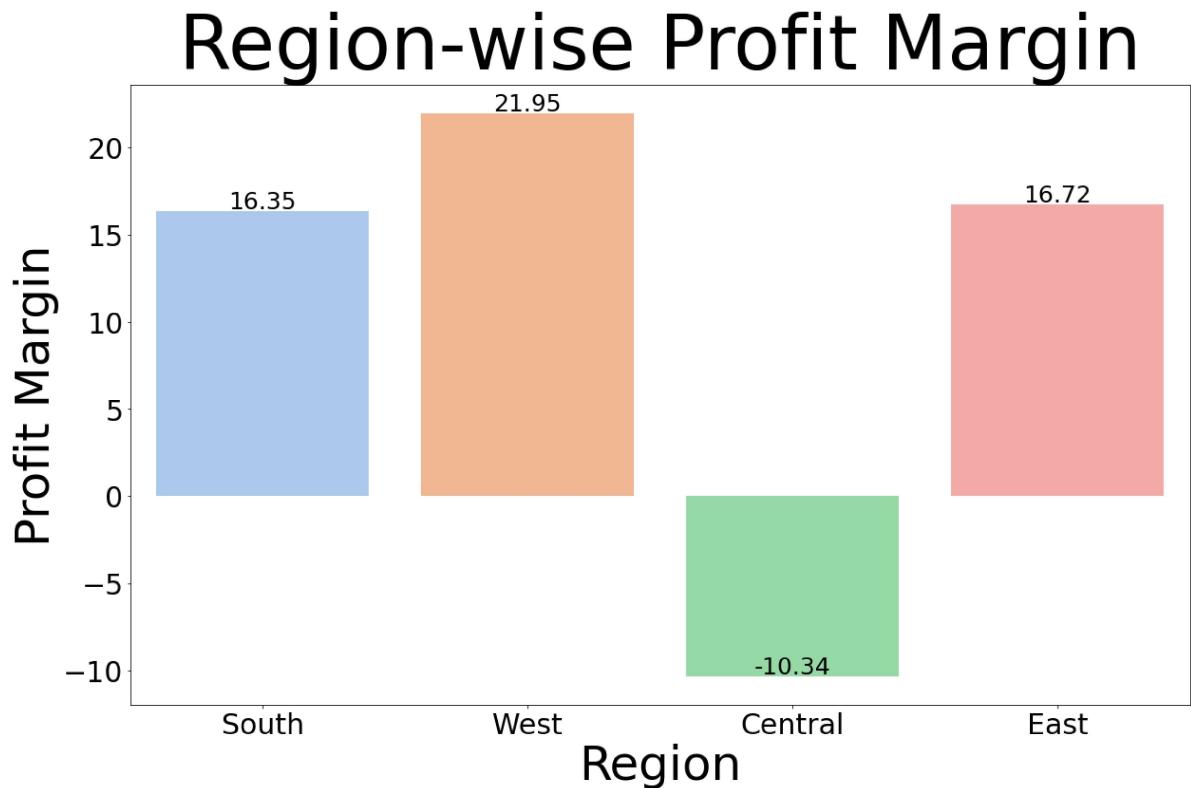


```
In [80]: plt.figure(figsize=(20, 12))
ax = sns.barplot(x='Region', y='Profit margin', data=df, ci=None, palette='pastel')

plt.title('Region-wise Profit Margin', fontsize=80)
plt.xlabel('Region', fontsize=50)
plt.ylabel('Profit Margin', fontsize=50)
plt.xticks(fontsize=30)
plt.yticks(fontsize=30)

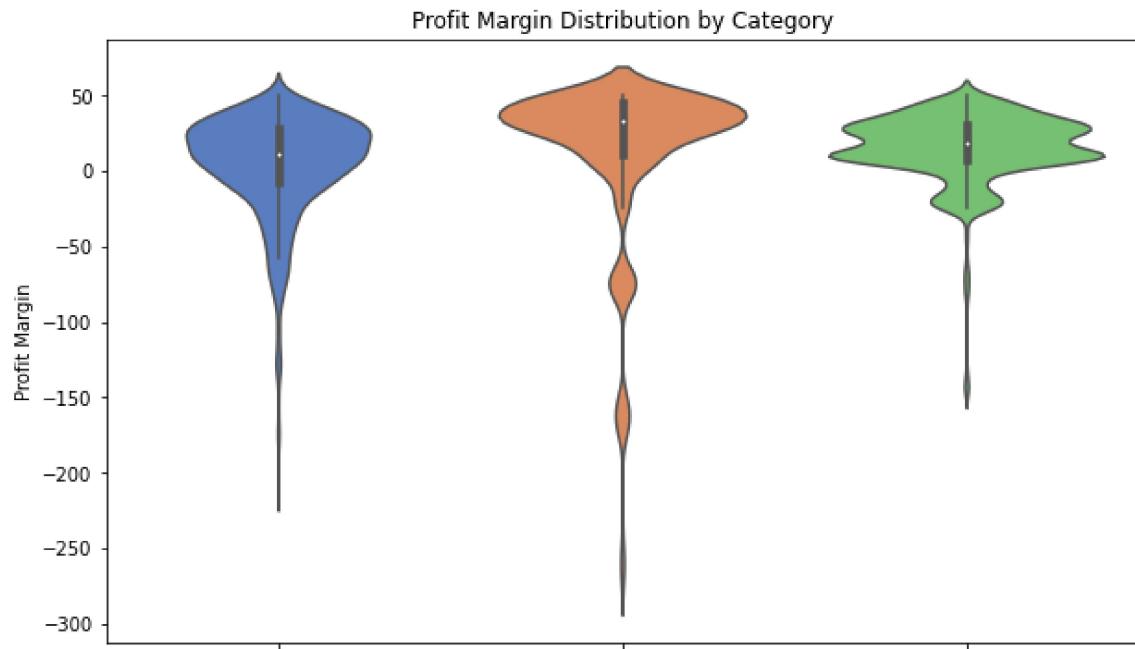
# Adding data labels to the bars
for p in ax.patches:
    ax.annotate(f'{p.get_height():.2f}', (p.get_x() + p.get_width() / 2., p.get_y() + p.get_height() / 2.), ha='center', va='center', fontsize=25, color='black', xytext=(0, 10), textcoords='offset points')

plt.show()
```



In [79]: #Profit Margin Distribution by Category :

```
plt.figure(figsize=(10, 6))
sns.violinplot(x='Category', y='Profit margin', data=df, palette='muted')
plt.title('Profit Margin Distribution by Category')
plt.xlabel('Category')
plt.ylabel('Profit Margin')
plt.show()
```



In [81]: customer_segments = df.groupby('Segment')['Sales'].sum()
print("Customer Segmentation:")
print(customer_segments)

```
Customer Segmentation:  
Segment  
Consumer      1.156601e+06  
Corporate     7.061044e+05  
Home Office   4.295843e+05  
Name: Sales, dtype: float64
```

```
In [82]: import matplotlib.pyplot as plt

# Sub-category-wise sales
sub_category_wise_sales = df.groupby('Sub-Category')['Sales'].sum()
print(sub_category_wise_sales)
# Average discount per sub-category
avg_discount_per_sub_category = df.groupby('Sub-Category')['Discount'].sum()
print(avg_discount_per_sub_category)
# Plotting Sales Line Chart
plt.figure(figsize=(20, 10))

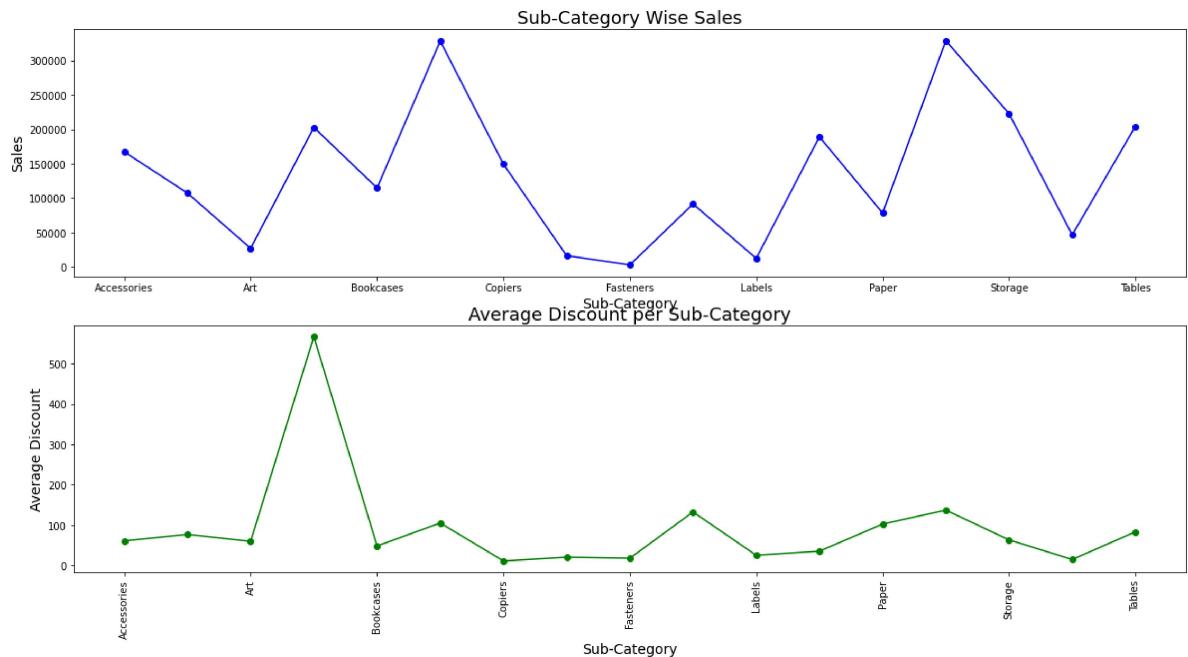
plt.subplot(2, 1, 1) # 2 rows, 1 column, subplot 1
sub_category_wise_sales.plot(kind='line', marker='o', color='blue')
plt.title('Sub-Category Wise Sales', fontsize=18)
plt.xlabel('Sub-Category', fontsize=14)
plt.ylabel('Sales', fontsize=14)

# Plotting Discount Line Chart
plt.subplot(2, 1, 2) # 2 rows, 1 column, subplot 2
avg_discount_per_sub_category.plot(kind='line', marker='o', color='green')
plt.title('Average Discount per Sub-Category', fontsize=18)
plt.xlabel('Sub-Category', fontsize=14)
plt.ylabel('Average Discount', fontsize=14)

# plt.tight_layout() # Adjust Layout for better spacing
plt.xticks(rotation=90)
plt.show()
```

```
Sub-Category
Accessories      167380.3180
Appliances       107463.3510
Art              27118.7920
Binders          202953.4450
Bookcases        114879.9963
Chairs           328449.1030
Copiers          149528.0300
Envelopes        16476.4020
Fasteners         3024.2800
Furnishings      91663.2040
Labels            12486.3120
Machines          189238.6310
Paper             78463.6540
Phones            329067.6380
Storage           222951.1680
Supplies          46673.5380
Tables            204471.8180
Name: Sales, dtype: float64
```

```
Sub-Category
Accessories      60.80
Appliances       76.80
Art              59.60
Binders          566.60
Bookcases        48.14
Chairs           105.00
Copiers          11.00
Envelopes        20.40
Fasteners         17.80
Furnishings      132.40
Labels            25.00
Machines          35.20
Paper             102.40
Phones            137.00
Storage           63.20
Supplies          14.60
Tables            83.15
Name: Discount, dtype: float64
```



```
In [83]: #quater wise sales, profit , profit margin trends
import pandas as pd
import matplotlib.pyplot as plt

df['Order Date'] = pd.to_datetime(df['Order Date'])
df.set_index('Order Date', inplace=True)
quarterly_data = df.resample('Q').agg({
    'Sales': 'sum',
    'Profit': 'sum',
    'Profit Margin': 'mean'
})

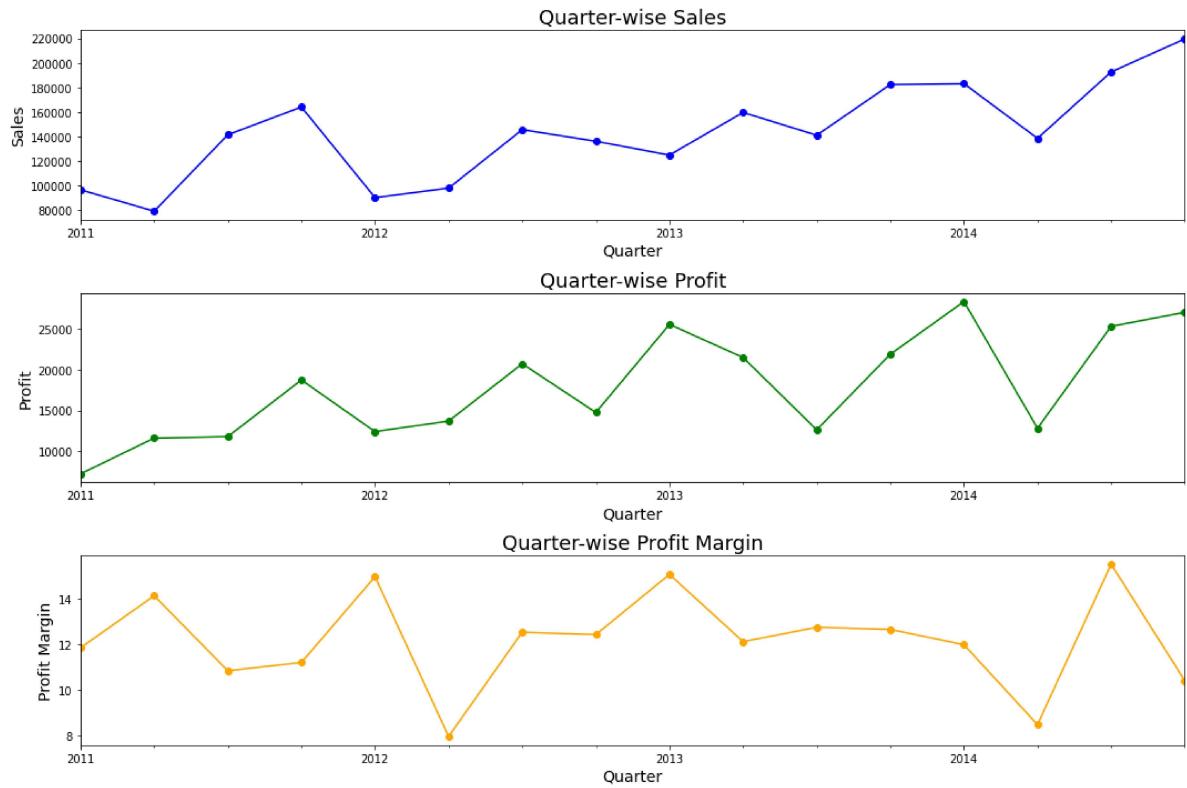
plt.figure(figsize=(15, 10))

# Sales Line Chart
plt.subplot(3, 1, 1)
quarterly_data['Sales'].plot(kind='line', marker='o', color='blue')
plt.title('Quarter-wise Sales', fontsize=18)
plt.xlabel('Quarter', fontsize=14)
plt.ylabel('Sales', fontsize=14)

# Profit Line Chart
plt.subplot(3, 1, 2)
quarterly_data['Profit'].plot(kind='line', marker='o', color='green')
plt.title('Quarter-wise Profit', fontsize=18)
plt.xlabel('Quarter', fontsize=14)
plt.ylabel('Profit', fontsize=14)

# Profit Margin Line Chart
plt.subplot(3, 1, 3)
quarterly_data['Profit Margin'].plot(kind='line', marker='o', color='orange')
plt.title('Quarter-wise Profit Margin', fontsize=18)
plt.xlabel('Quarter', fontsize=14)
plt.ylabel('Profit Margin', fontsize=14)

plt.tight_layout()
plt.show()
```

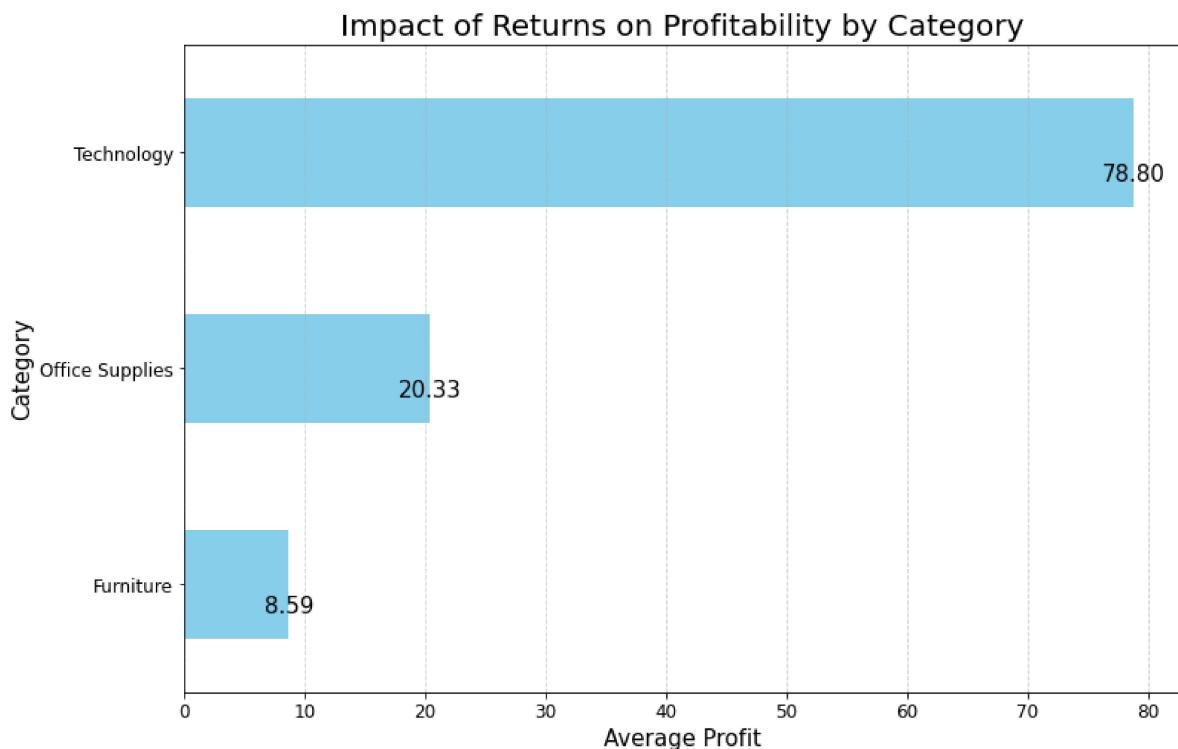


```
In [84]: returns_analysis = df.groupby('Category')['Profit'].mean()

# Plotting the bar chart
plt.figure(figsize=(12, 8))
bars = returns_analysis.sort_values().plot(kind='barh', color='skyblue')
plt.title('Impact of Returns on Profitability by Category', fontsize=20)
plt.xlabel('Average Profit', fontsize=15)
plt.ylabel('Category', fontsize=15)
plt.xticks(rotation=0, fontsize=12)
plt.yticks(fontsize=12)
plt.grid(axis='x', linestyle='--', alpha=0.6)

# Adding data Labels to the bars
for bar in bars.patches:
    plt.text(bar.get_width() + 0.1, bar.get_y() + bar.get_height()/2 - 0.1,
             f'{bar.get_width():,.2f}', ha='center', va='center', fontsize=15,
             color='black')

plt.show()
```



```
In [ ]:
```