In [1]:

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

In [2]:

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

Out[2]:

	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	H
1	2	CA-2013-152156	09-11-2013	12-11-2013	Second Class	CG-12520	Claire Gute	Consumer	United States	H
2	3	CA-2013-138688	13-06-2013	17-06-2013	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los
3	4	US-2012-108966	11-10-2012	18-10-2012	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	La
4	5	US-2012-108966	11-10-2012	18-10-2012	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	La
...	...	...	...	...	...	...	...	...	...	
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	Co
9991	9992	CA-2014-121258	27-02-2014	04-03-2014	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Co
9992	9993	CA-2014-121258	27-02-2014	04-03-2014	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Co
9993	9994	CA-2014-119914	05-05-2014	10-05-2014	Second Class	CC-12220	Chris Cortes	Consumer	United States	We

9994 rows × 20 columns

In [3]:
```
#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
```

In [4]:
```
# 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[4]:
```
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 [5]:

```
# Drop the null- values rows
df.dropna(inplace=True)
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 [6]:

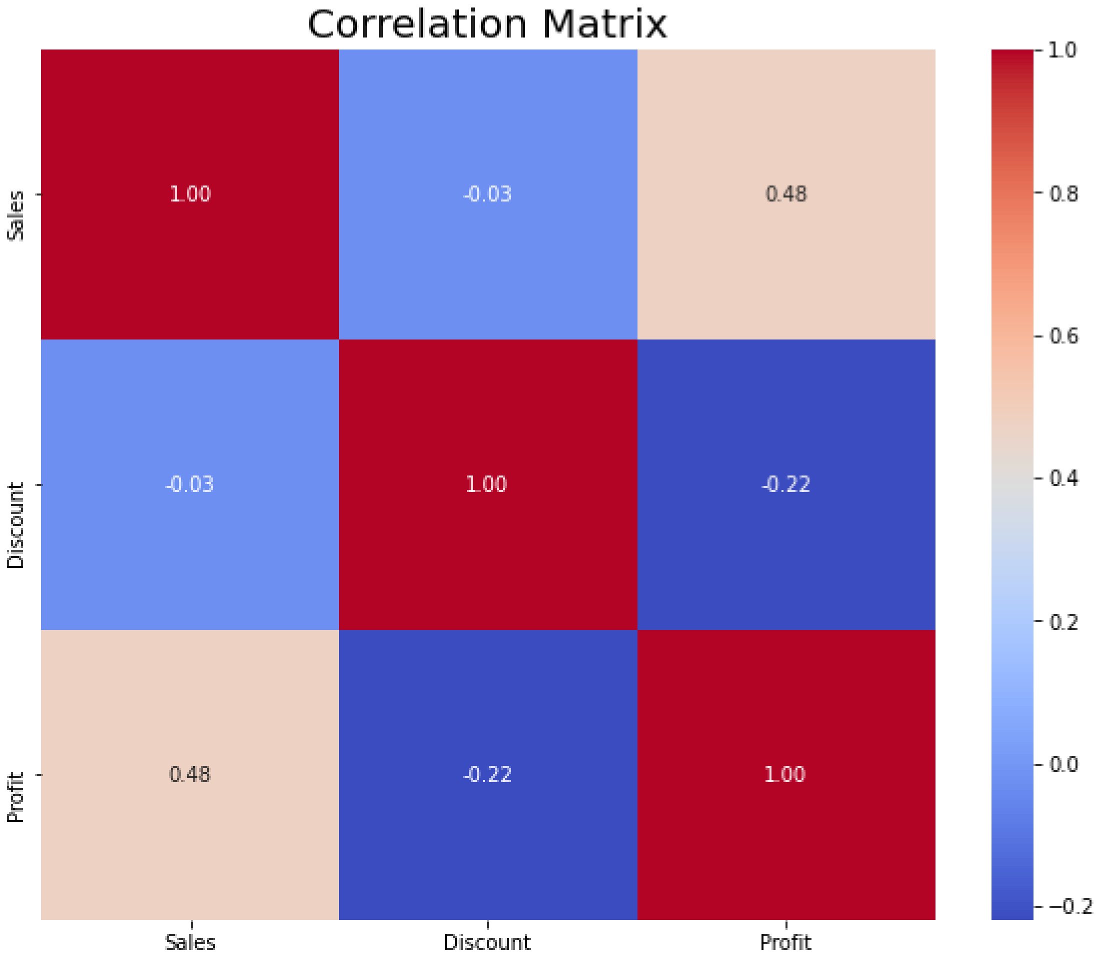
```
#Profit margin :

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

Out[6]:

	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	H
1	2	CA-2013-152156	09-11-2013	12-11-2013	Second Class	CG-12520	Claire Gute	Consumer	United States	H
2	3	CA-2013-138688	13-06-2013	17-06-2013	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los
3	4	US-2012-108966	11-10-2012	18-10-2012	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	La
4	5	US-2012-108966	11-10-2012	18-10-2012	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	La
...	...	...	...	...	...	...	...	...	...	
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	Co
9991	9992	CA-2014-121258	27-02-2014	04-03-2014	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Co
9992	9993	CA-2014-121258	27-02-2014	04-03-2014	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Co
9993	9994	CA-2014-119914	05-05-2014	10-05-2014	Second Class	CC-12220	Chris Cortes	Consumer	United States	We

9983 rows × 21 columns

In [7]:
```
# 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()
```



In [8]:
```
# Top 10 Customers by Sales:
top_customers = df.groupby('Customer Name')['Sales'].sum().sort_values(asce
print('Top 10 Customers Based on Sales:')
print(top_customers)
```

```
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
```

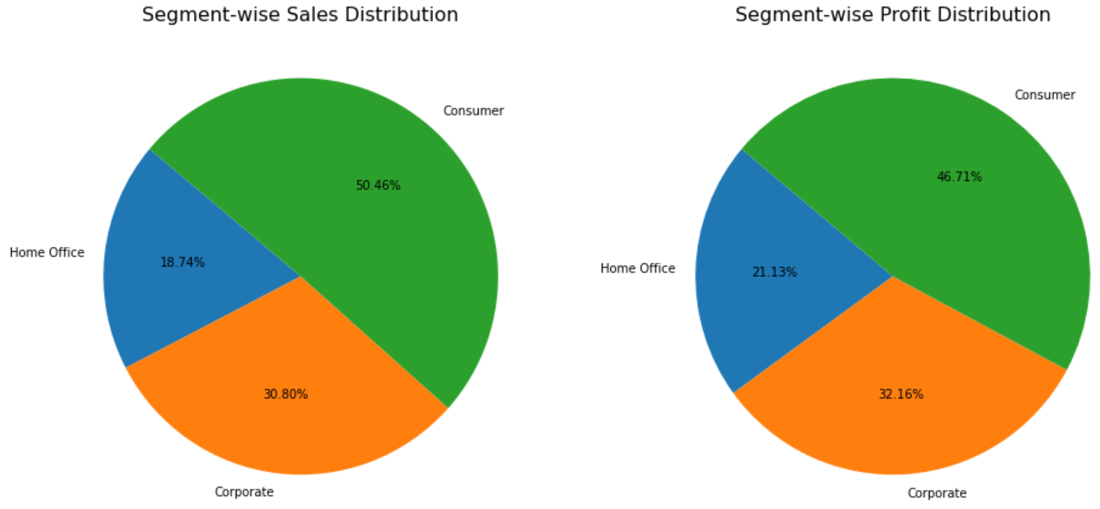
In [9]:

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

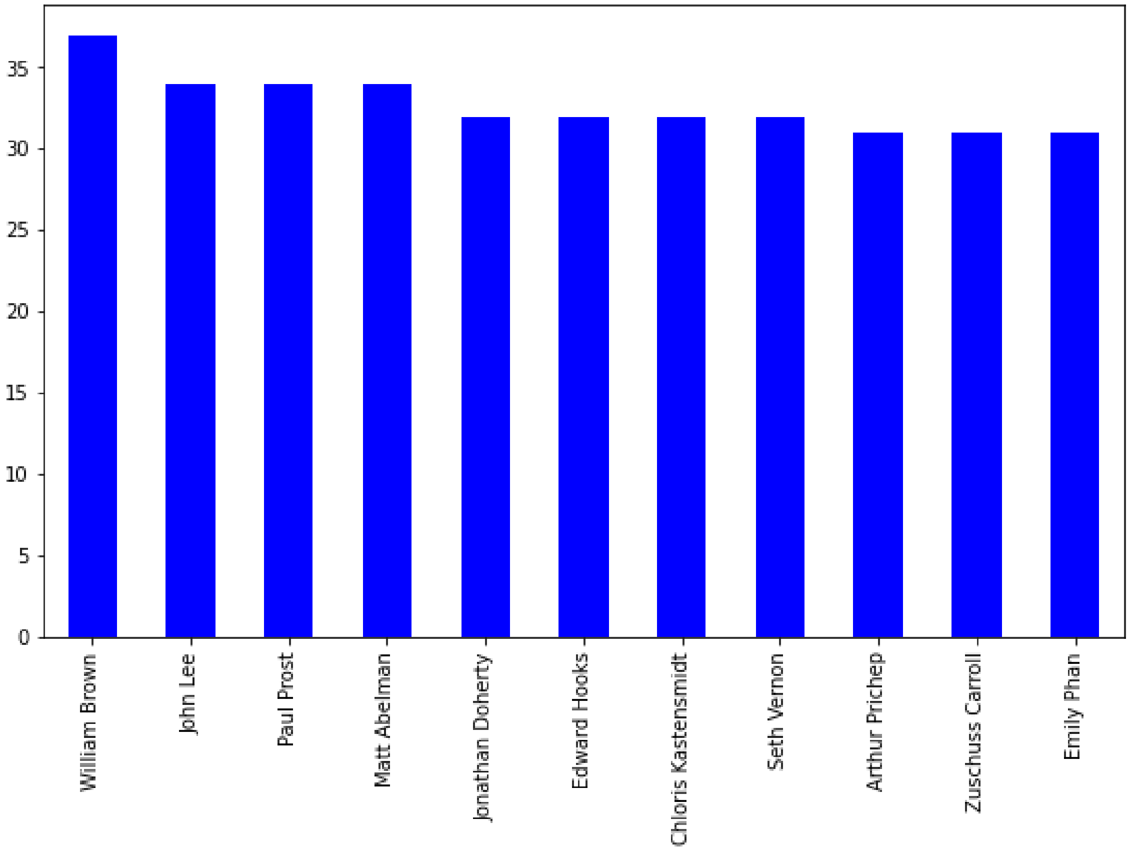
```
Top 10 Best Selling Products:
Product ID
TEC-CO-10004722    61599.824
OFF-BI-10003527    27453.384
TEC-MA-10002412    22638.480
FUR-CH-10002024    21870.576
OFF-BI-10001359    19823.479
OFF-BI-10000545    19024.500
TEC-CO-10001449    18839.686
TEC-MA-10001127    18374.895
OFF-BI-10004995    17965.068
OFF-SU-10000151    17030.312
Name: Sales, dtype: float64
```

In [10]:

```
#Sales and profit distribution by Segment

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

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

plt.show()
```

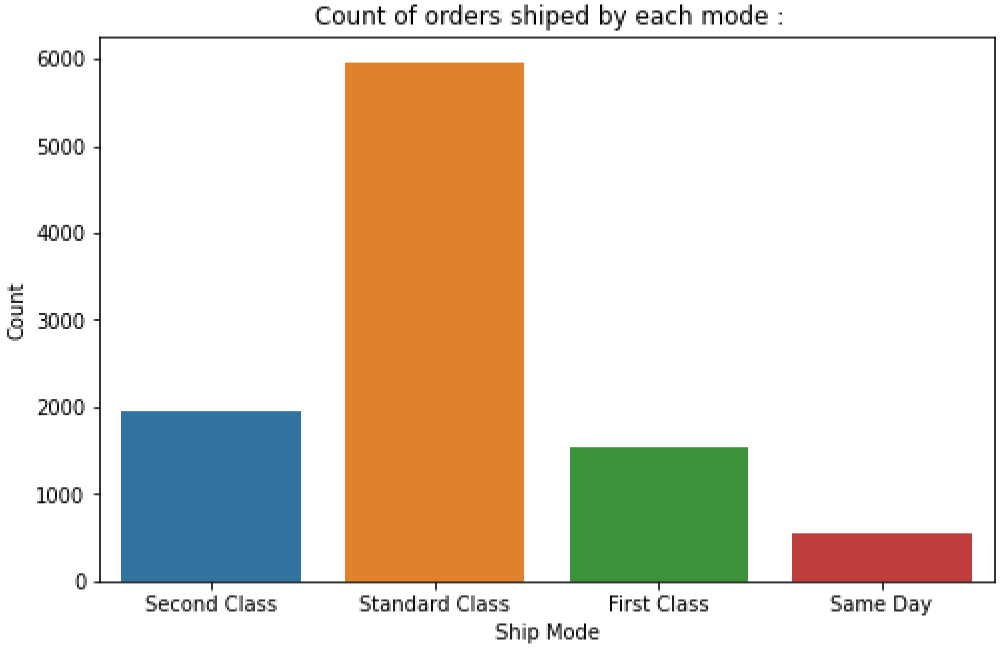


```
In [11]: # The frequency of purchases for each specific customer

         df['Customer Name'].value_counts().head(11).plot(kind='bar', figsize=(10, 6

Out[11]: <Axes: >
```



In [12]:

```
# Count of orders shiped by each mode :

plt.figure(figsize=(8, 5))
sns.countplot(x='Ship Mode', data=df)
plt.title(' Count of orders shiped by each mode :')
plt.xlabel('Ship Mode')
plt.ylabel('Count')
plt.show()
```



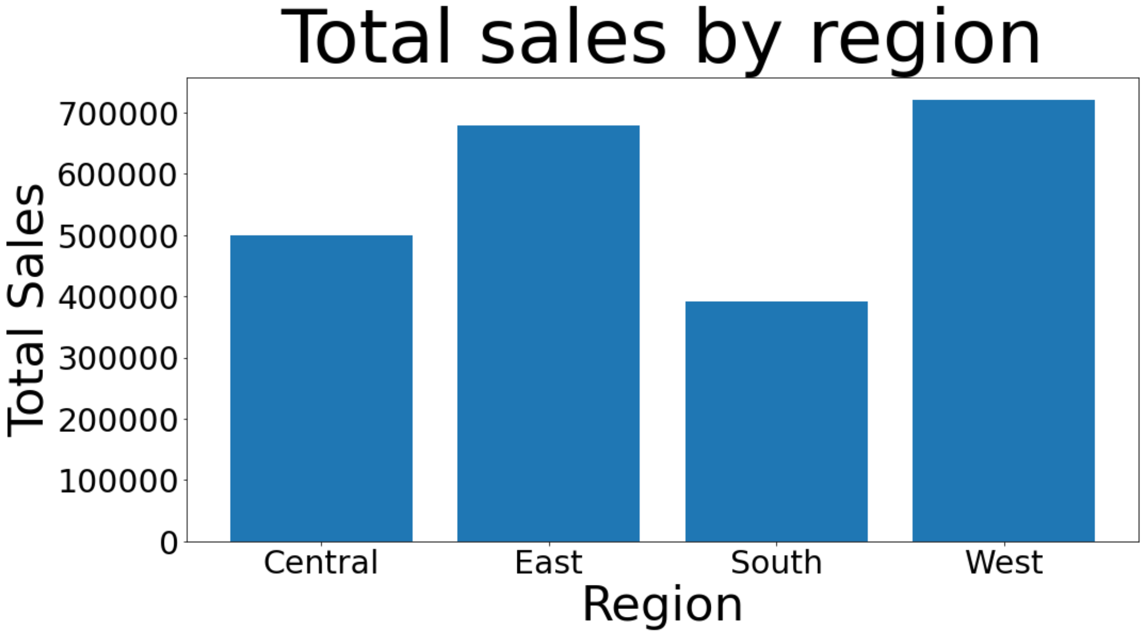
In [13]:

```
# Top 10 most profitable and 10 least profitable cities:
most_profitable_cities = df.groupby('City')['Profit'].sum().sort_values(asc
least_profitable_cities = df.groupby('City')['Profit'].sum().sort_values().
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
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
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
```

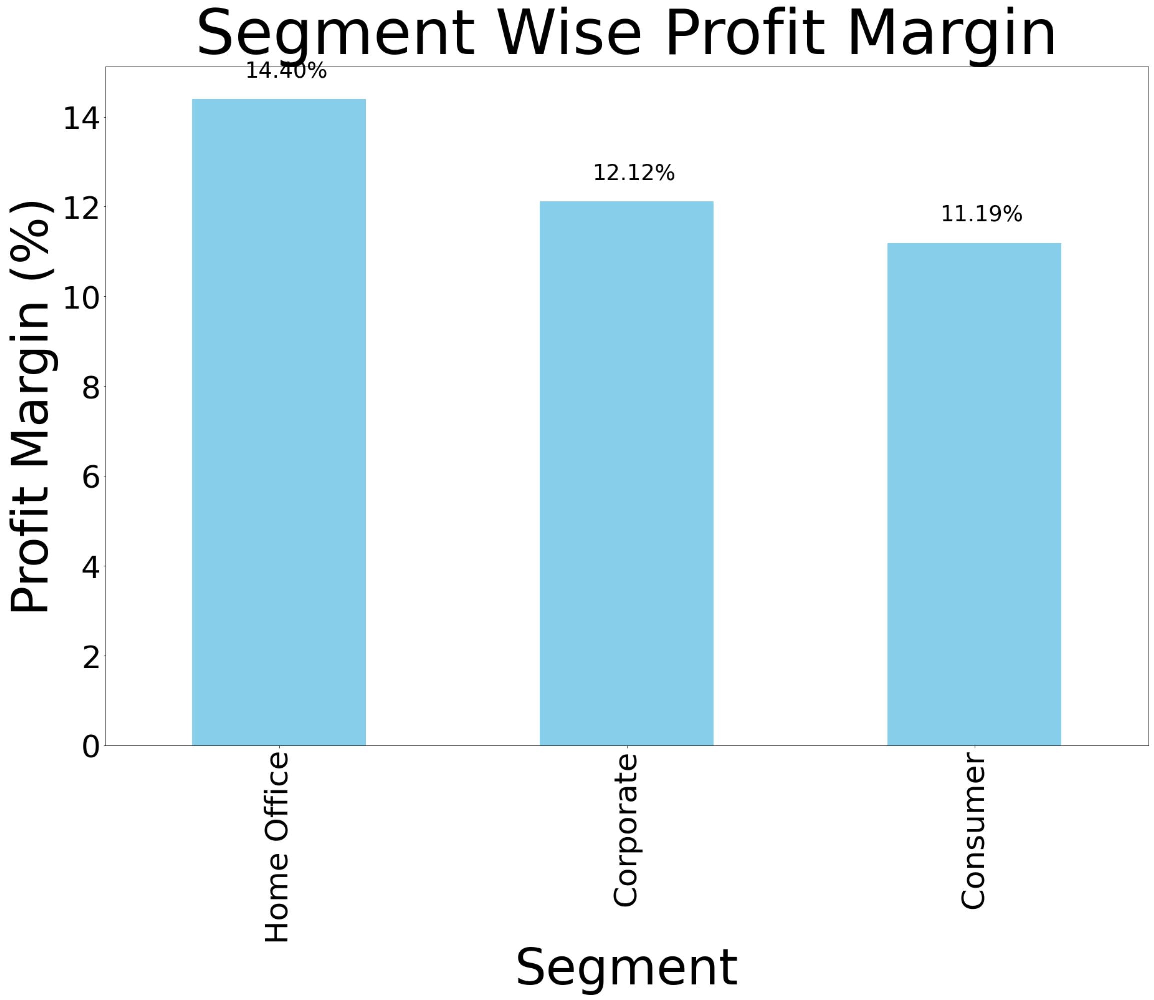
```
In [16]: #Total sales by region :

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

```
Out[16]: (array([     0., 100000., 200000., 300000., 400000., 500000., 600000.,
                 700000., 800000.]),
          [Text(0, 0.0, '0'),
           Text(0, 100000.0, '100000'),
           Text(0, 200000.0, '200000'),
           Text(0, 300000.0, '300000'),
           Text(0, 400000.0, '400000'),
           Text(0, 500000.0, '500000'),
           Text(0, 600000.0, '600000'),
           Text(0, 700000.0, '700000'),
           Text(0, 800000.0, '800000')])
```



In [17]:

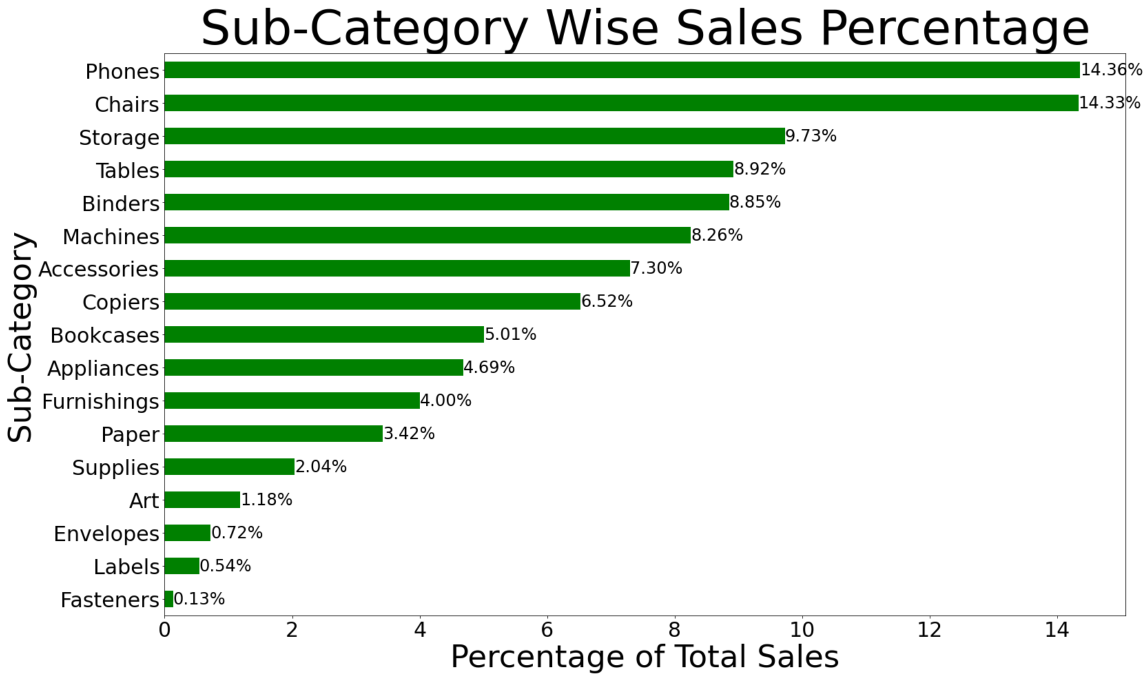
```
# Segment wise profit margin :

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

segment_profit_margin = df.groupby('Segment')['Profit Margin'].mean().sort_
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.
             f'{bar.get_height():.2f}%', fontsize=35,color='black')

plt.show()
```



In [18]:

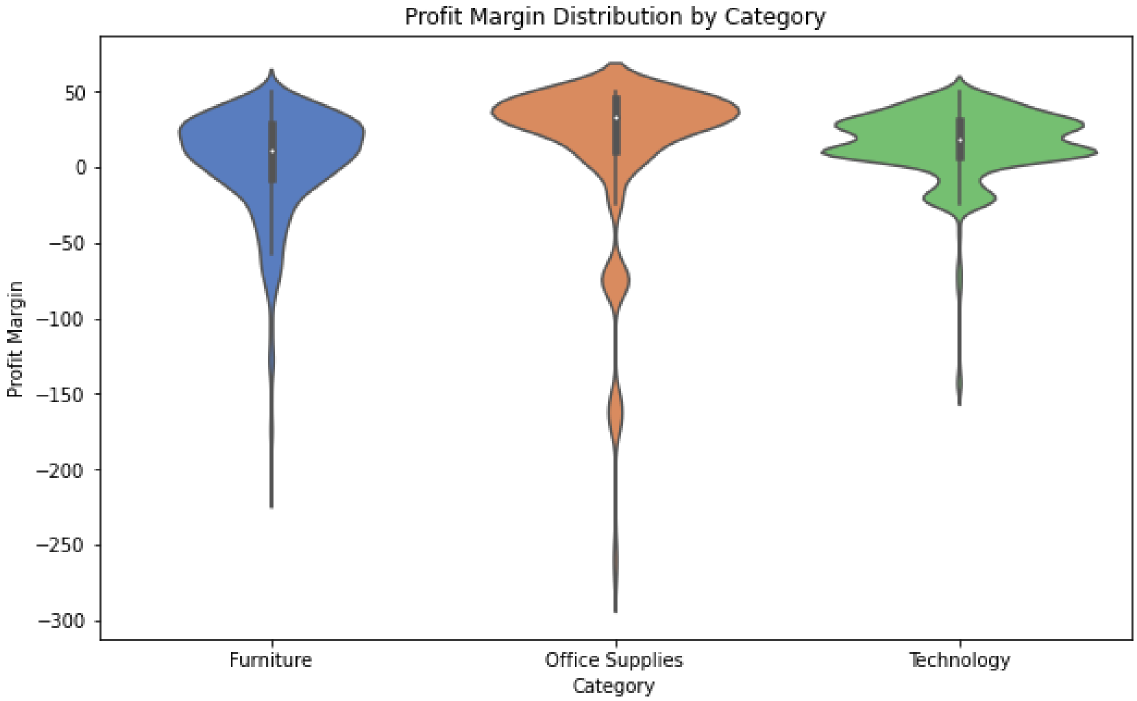
```
#Sub-category wise sales and its percentage:

sub_category_wise_sales = df.groupby('Sub-Category')['Sales'].sum().sort_va
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'].s
plt.figure(figsize=(25, 15))
ax = sub_category_wise_sales_percentage.sort_values(ascending=True).plot(ki
for rect in ax.patches:
    width = rect.get_width()
    plt.text(width, rect.get_y() + rect.get_height() / 2, f'{width:.2f}%', 
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
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 [19]:

```
#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 [ ]: