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
```

# Importing CSV file

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 | City | State | Postal Code | Region | Prod |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | CA-2013-152156 | 09-11-2013 | 12-11-2013 | Second Class | CG-12520 | Claire Gute | Consumer | United States | Henderson | Kentucky | 42420 | South | FUR-B 10001 |
| **1** | 2 | CA-2013-152156 | 09-11-2013 | 12-11-2013 | Second Class | CG-12520 | Claire Gute | Consumer | United States | Henderson | Kentucky | 42420 | South | FUR-C 10000 |
| **2** | 3 | CA-2013-138688 | 13-06-2013 | 17-06-2013 | Second Class | DV-13045 | Darrin Van Huff | Corporate | United States | Los Angeles | California | 90036 | West | OFF- 10000 |
| **3** | 4 | US-2012-108966 | 11-10-2012 | 18-10-2012 | Standard Class | SO-20335 | Sean O'Donnell | Consumer | United States | Fort Lauderdale | Florida | 33311 | South | FUR- 10000 |
| **4** | 5 | US-2012-108966 | 11-10-2012 | 18-10-2012 | Standard Class | SO-20335 | Sean O'Donnell | Consumer | United States | Fort Lauderdale | Florida | 33311 | South | OFF- 10000 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| **9989** | 9990 | CA-2011-110422 | 22-01-2011 | 24-01-2011 | Second Class | TB-21400 | Tom Boeckenhauer | Consumer | United States | Miami | Florida | 33180 | South | FUR-F 10001 |
| **9990** | 9991 | CA-2014-121258 | 27-02-2014 | 04-03-2014 | Standard Class | DB-13060 | Dave Brooks | Consumer | United States | Costa Mesa | California | 92627 | West | FUR-F 10000 |
| **9991** | 9992 | CA-2014-121258 | 27-02-2014 | 04-03-2014 | Standard Class | DB-13060 | Dave Brooks | Consumer | United States | Costa Mesa | California | 92627 | West | TEC-F 10003 |
| **9992** | 9993 | CA-2014-121258 | 27-02-2014 | 04-03-2014 | Standard Class | DB-13060 | Dave Brooks | Consumer | United States | Costa Mesa | California | 92627 | West | OFF-F 10004 |
| **9993** | 9994 | CA-2014-119914 | 05-05-2014 | 10-05-2014 | Second Class | CC-12220 | Chris Cortes | Consumer | United States | Westminster | California | 92683 | West | OFF-A 10002 |

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
```

# Cleaning Process

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)
```

In [6]:
```
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 [7]:
```
# know the unique value in the each column
df.nunique(axis=0)
```

Out[7]:
```
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 [8]:
```
#Profit margin :

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

Out[8]:

| | Row ID | Order ID | Order Date | Ship Date | Ship Mode | Customer ID | Customer Name | Segment | Country | City | ... | Postal Code | Region | Product ID | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | CA-2013-152156 | 09-11-2013 | 12-11-2013 | Second Class | CG-12520 | Claire Gute | Consumer | United States | Henderson | ... | 42420 | South | FUR-BO-10001798 | |
| **1** | 2 | CA-2013-152156 | 09-11-2013 | 12-11-2013 | Second Class | CG-12520 | Claire Gute | Consumer | United States | Henderson | ... | 42420 | South | FUR-CH-10000454 | |
| **2** | 3 | CA-2013-138688 | 13-06-2013 | 17-06-2013 | Second Class | DV-13045 | Darrin Van Huff | Corporate | United States | Los Angeles | ... | 90036 | West | OFF-LA-10000240 | |
| **3** | 4 | US-2012-108966 | 11-10-2012 | 18-10-2012 | Standard Class | SO-20335 | Sean O'Donnell | Consumer | United States | Fort Lauderdale | ... | 33311 | South | FUR-TA-10000577 | |
| **4** | 5 | US-2012-108966 | 11-10-2012 | 18-10-2012 | Standard Class | SO-20335 | Sean O'Donnell | Consumer | United States | Fort Lauderdale | ... | 33311 | South | OFF-ST-10000760 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| **9989** | 9990 | CA-2011-110422 | 22-01-2011 | 24-01-2011 | Second Class | TB-21400 | Tom Boeckenhauer | Consumer | United States | Miami | ... | 33180 | South | FUR-FU-10001889 | |
| **9990** | 9991 | CA-2014-121258 | 27-02-2014 | 04-03-2014 | Standard Class | DB-13060 | Dave Brooks | Consumer | United States | Costa Mesa | ... | 92627 | West | FUR-FU-10000747 | |
| **9991** | 9992 | CA-2014-121258 | 27-02-2014 | 04-03-2014 | Standard Class | DB-13060 | Dave Brooks | Consumer | United States | Costa Mesa | ... | 92627 | West | TEC-PH-10003645 | Te |
| **9992** | 9993 | CA-2014-121258 | 27-02-2014 | 04-03-2014 | Standard Class | DB-13060 | Dave Brooks | Consumer | United States | Costa Mesa | ... | 92627 | West | OFF-PA-10004041 | |
| **9993** | 9994 | CA-2014-119914 | 05-05-2014 | 10-05-2014 | Second Class | CC-12220 | Chris Cortes | Consumer | United States | Westminster | ... | 92683 | West | OFF-AP-10002684 | |

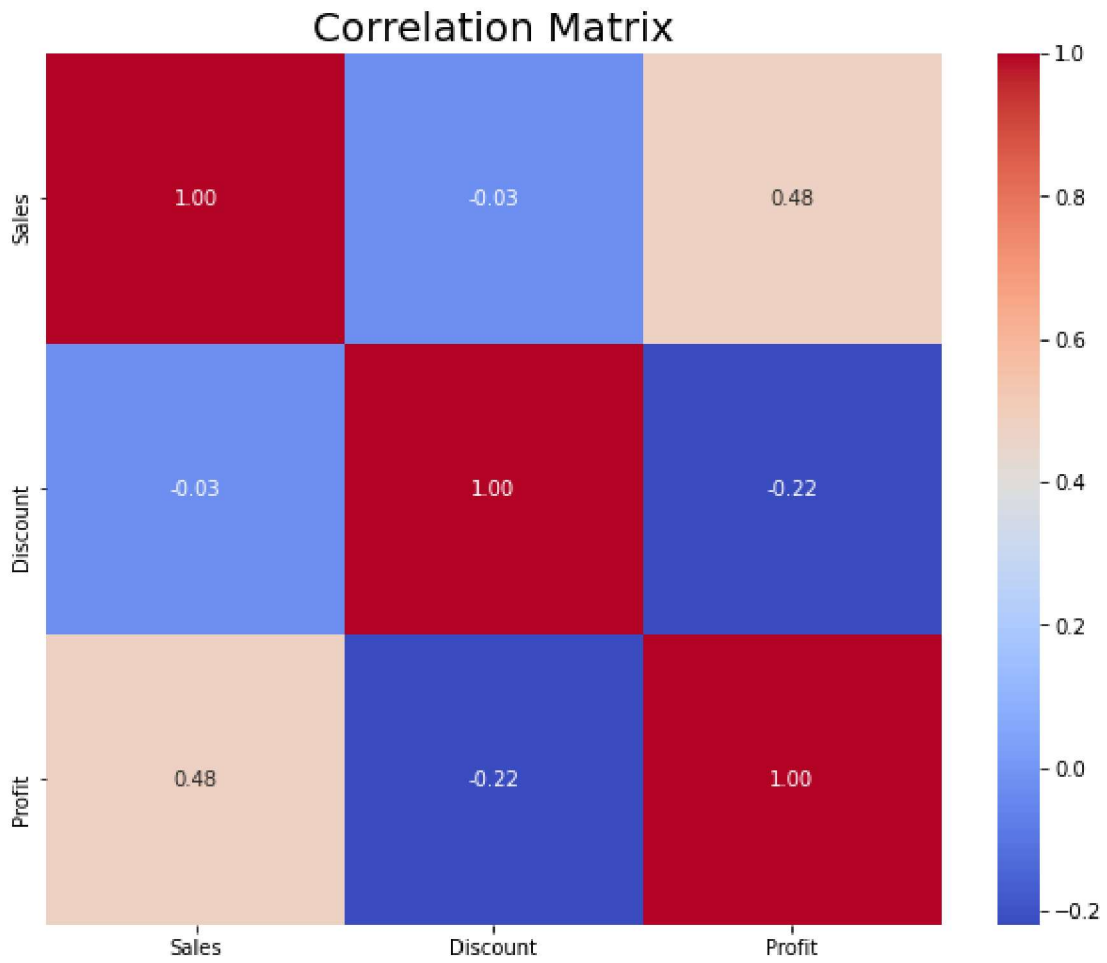
9983 rows × 21 columns

In [9]:
```
# it shows the description of all numerical values in the data
description = df[['Sales', 'Quantity', 'Discount', 'Profit','Profit margin']].describe()
description
```

Out[9]:

| | 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 [34]:
```
# 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 [10]:
```
# Top 10 Best Selling Products:
best_selling_products = df.groupby('Product ID')['Sales'].sum().sort_values(ascending=False).head(10)
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 [11]:
```
# Top 10 Customers by Sales:
top_customers = df.groupby('Customer Name')['Sales'].sum().sort_values(ascending=False).head(10)
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 [12]:
```
# 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
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 [13]:
```
# Top 10 cities by sales:
df.groupby(['City'], as_index=False)['Sales'].sum().sort_values(by='Sales', ascending=False).head(10)
```

Out[13]:

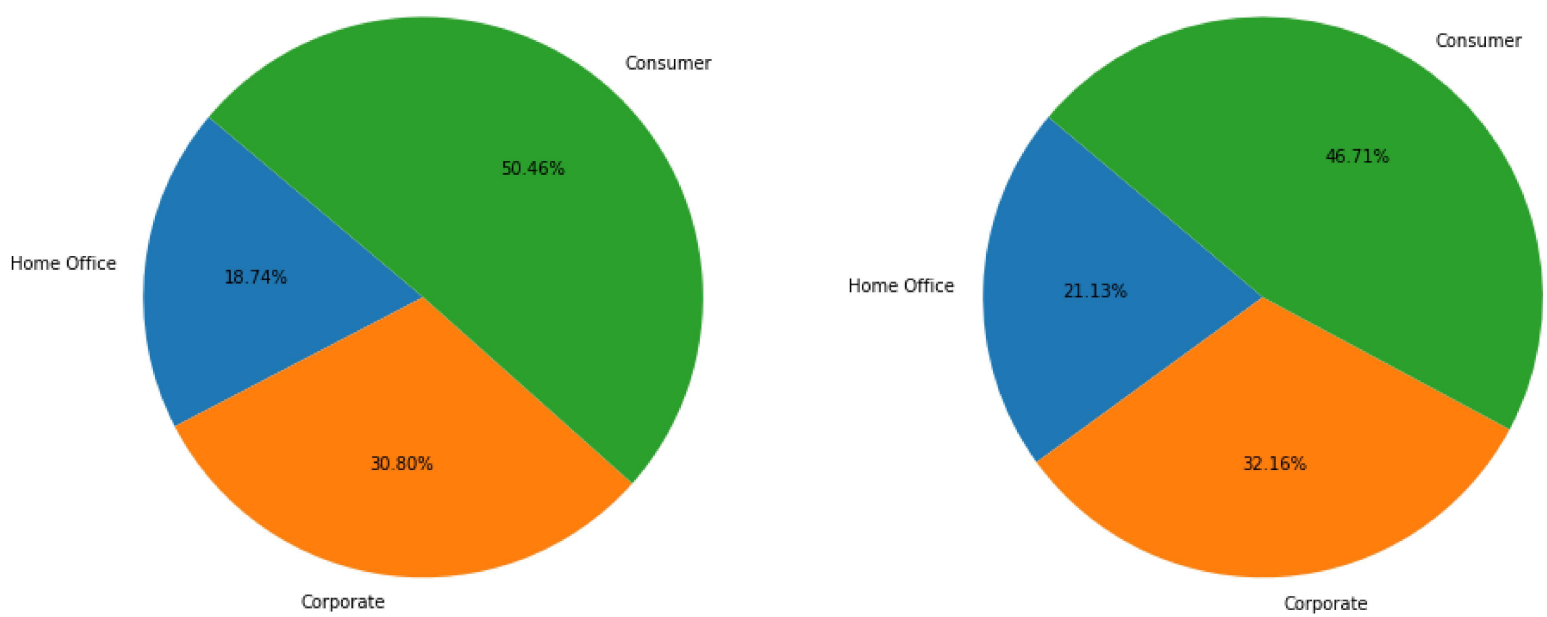
| | City | Sales |
|---|---|---|
| **329** | New York City | 256326.2010 |
| **266** | Los Angeles | 173233.7330 |
| **452** | Seattle | 118345.2360 |
| **438** | San Francisco | 112617.7800 |
| **374** | Philadelphia | 109077.0130 |
| **207** | Houston | 64504.7604 |
| **80** | Chicago | 48539.5410 |
| **437** | San Diego | 47521.0290 |
| **216** | Jacksonville | 44713.1830 |
| **464** | Springfield | 43054.3420 |

In [14]:
```
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_quantity)
print()
segment_wise_sales = df.groupby('Segment')['Profit'].sum()
print("Total Profit by Segments :")
print(segment_wise_quantity)
```

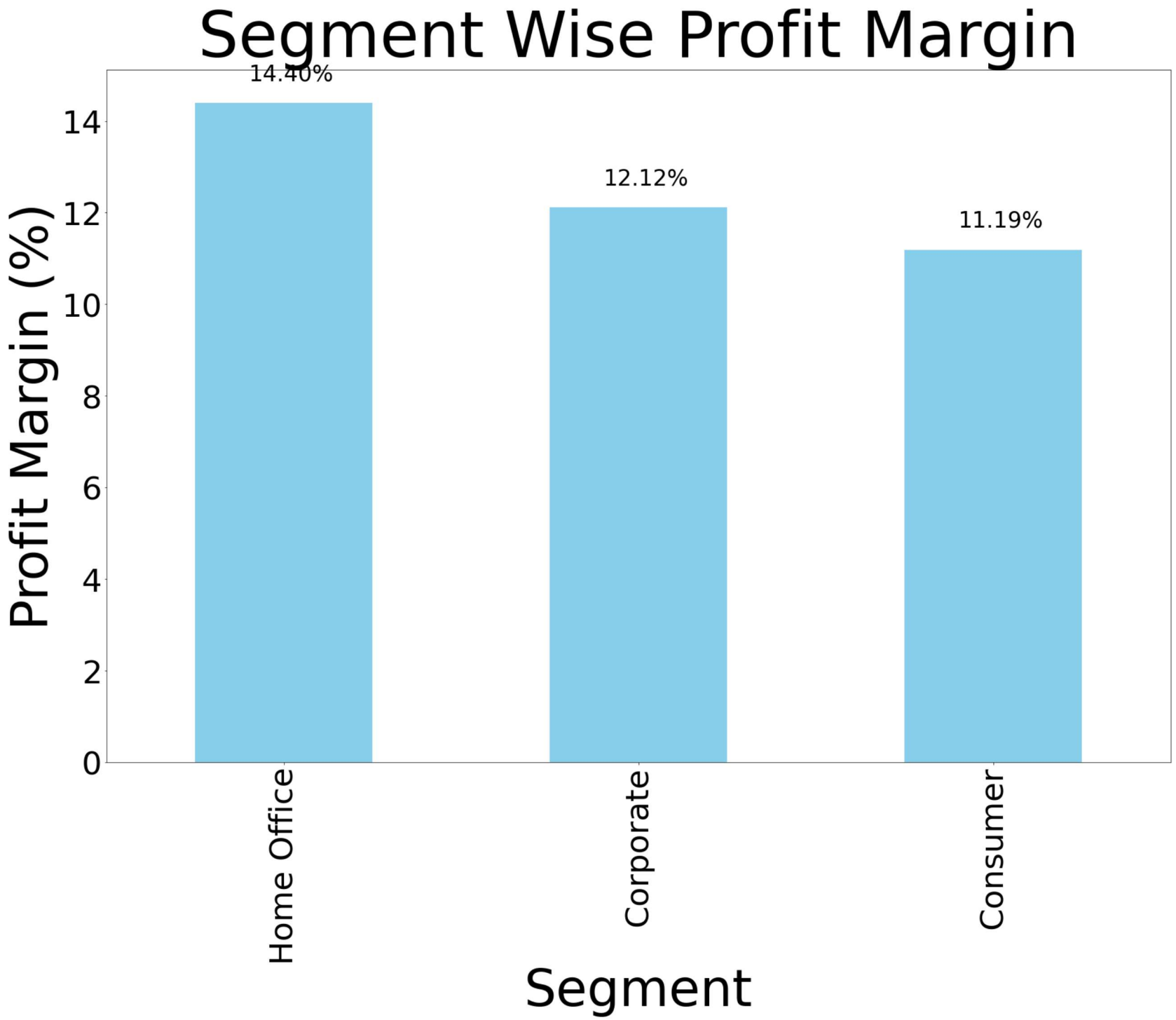
```
Total Quantity sold by each Segment :
Segment
Consumer       19483
Corporate      11606
Home Office     6739
Name: Quantity, dtype: int64

Total Sales by Segments :
Segment
Consumer       19483
Corporate      11606
Home Office     6739
Name: Quantity, dtype: int64

Total Profit by Segments :
Segment
Consumer       19483
Corporate      11606
Home Office     6739
Name: Quantity, dtype: int64
```

In [15]:
```
#Sales based on Customer Segmentation :

plt.figure(figsize=(10, 6))
sns.scatterplot(x='Sales', y='Profit', hue='Segment', data=df, palette='Set2')
plt.title('Sales based on Customer Segmentation')
plt.xlabel('Sales')
plt.ylabel('Profit')
plt.show()
```

In [16]:
```
#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='%1.2f%%', startangle=140)
axs[0].set_title('Segment-wise Sales Distribution', fontsize=16)

axs[1].pie(segment_wise_profit, labels=segment_wise_profit.index, autopct='%1.2f%%', startangle=140)
axs[1].set_title('Segment-wise Profit Distribution', fontsize=16)

plt.show()
```

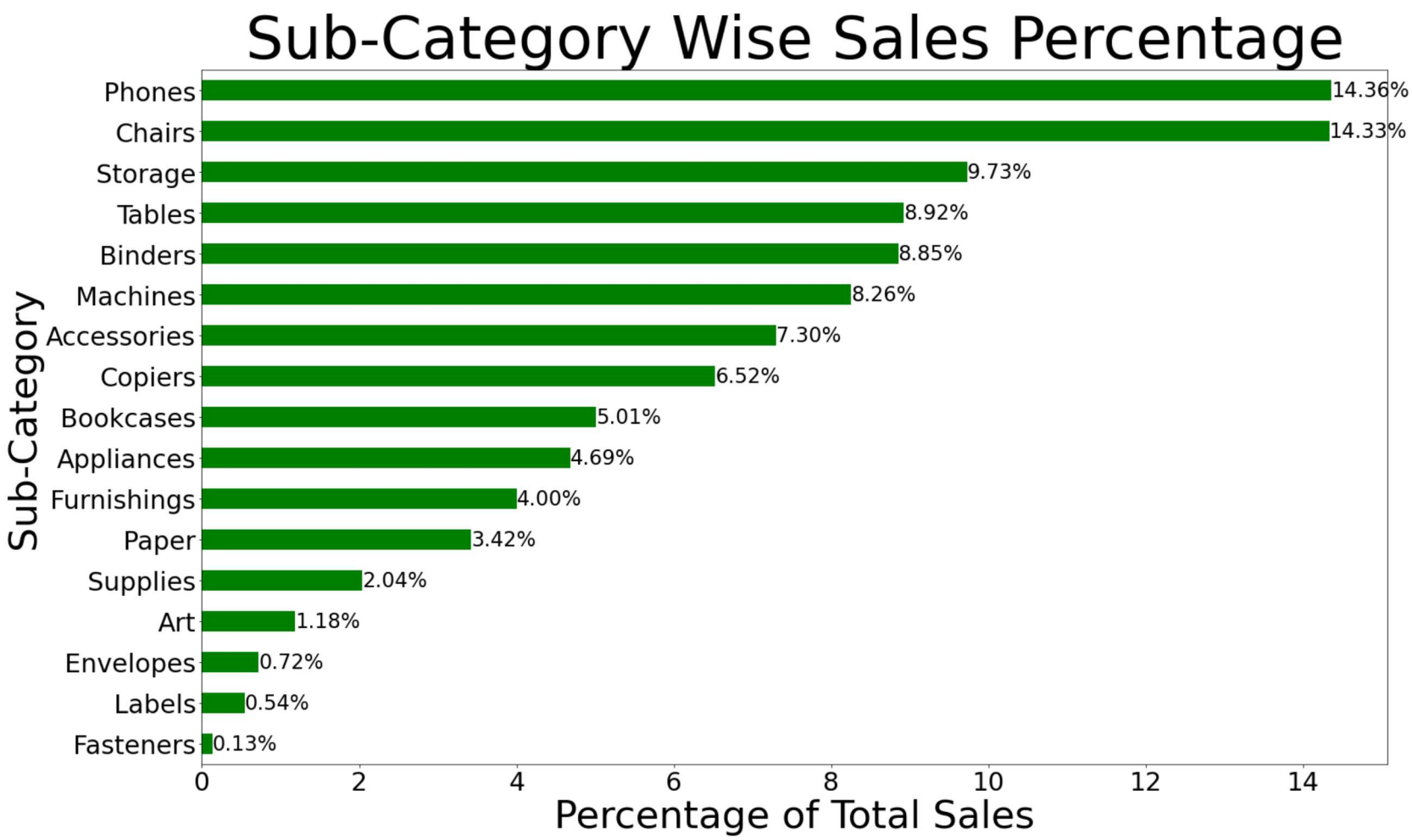


In [17]:
```
# Segment wise profit margin :

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

segment_profit_margin = df.groupby('Segment')['Profit Margin'].mean().sort_values(ascending=False)
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()
```


Segment Wise Profit Margin
14.40%
12.12%
11.19%
Profit Margin (%)
0
2
4
6
8
10
12
14
Home Office
Corporate
Consumer
Segment

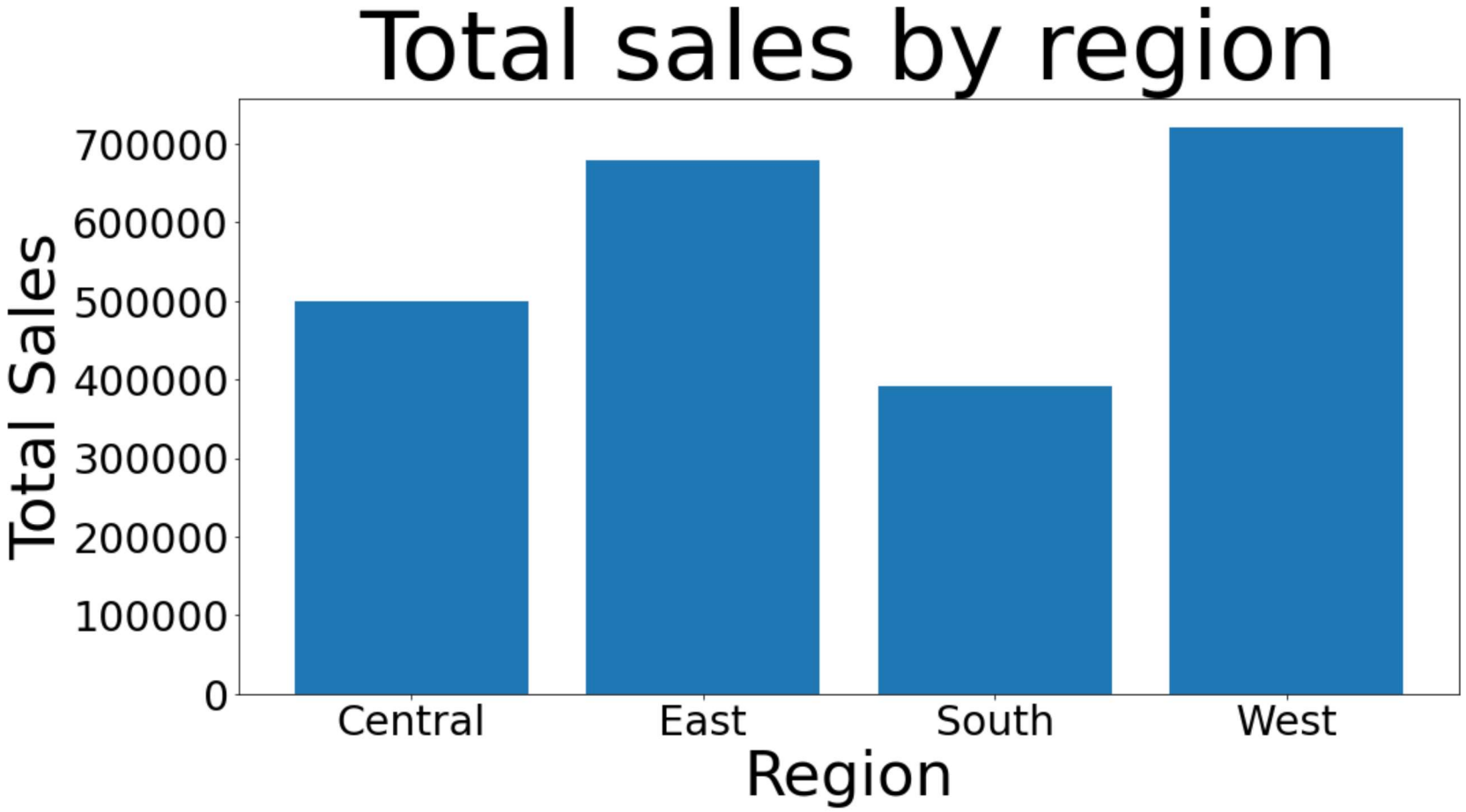
In [18]:
```
#Sub-category wise sales and its percentage:

sub_category_wise_sales = df.groupby('Sub-Category')['Sales'].sum().sort_values(ascending=False)
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='barh', color='green')
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', fontsize=2
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
```


Sub-Category Wise Sales Percentage
Phones
Chairs
Storage
Tables
Binders
Machines
Accessories
Copiers
Bookcases
Appliances
Furnishings
Paper
Supplies
Art
Envelopes
Labels
Fasteners
14.36%
14.33%
9.73%
8.92%
8.85%
8.26%
7.30%
6.52%
5.01%
4.69%
4.00%
3.42%
2.04%
1.18%
0.72%
0.54%
0.13%
Sub-Category
0
2
4
6
8
10
12
14
Percentage of Total Sales

In [19]:
```
#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)
```

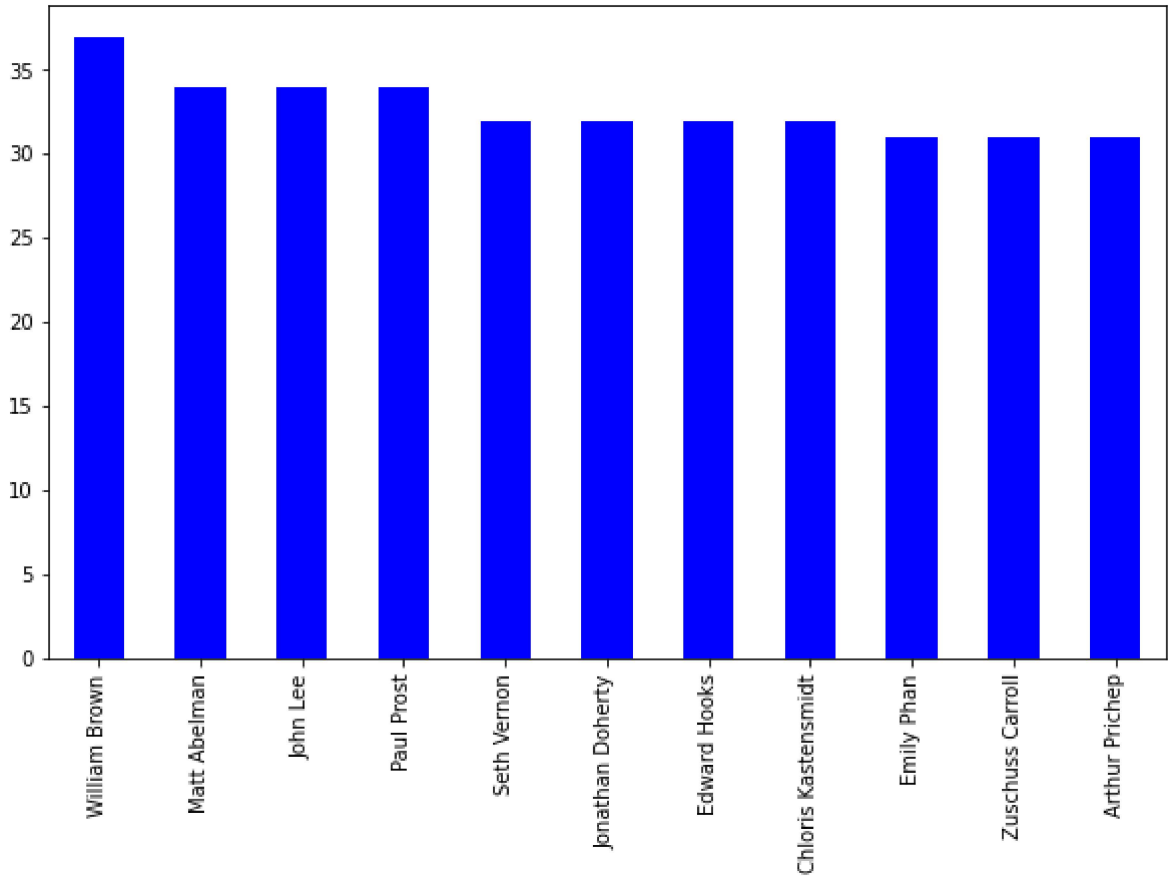
Out[19]:
```
(array([     0., 100000., 200000., 300000., 400000., 500000., 600000.,
        700000., 800000.]),
 [Text(0, 0.0, '0'),
  Text(0, 100000.0, '100000'),
  Text(0, 200000.0, '200000'),
  Text(0, 300000.0, '300000'),
  Text(0, 400000.0, '400000'),
  Text(0, 500000.0, '500000'),
  Text(0, 600000.0, '600000'),
  Text(0, 700000.0, '700000'),
  Text(0, 800000.0, '800000')])
```


Total sales by region
Total Sales
700000
600000
500000
400000
300000
200000
100000
0
Central
East
South
West
Region

In [20]:
```
# 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 [32]:
```
# The frequency of purchases for each specific customer

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

Out[32]: <Axes: >



In [33]:
```
import seaborn as sns
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]


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

In [23]:
```
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_category_wise_sales) * 100
```

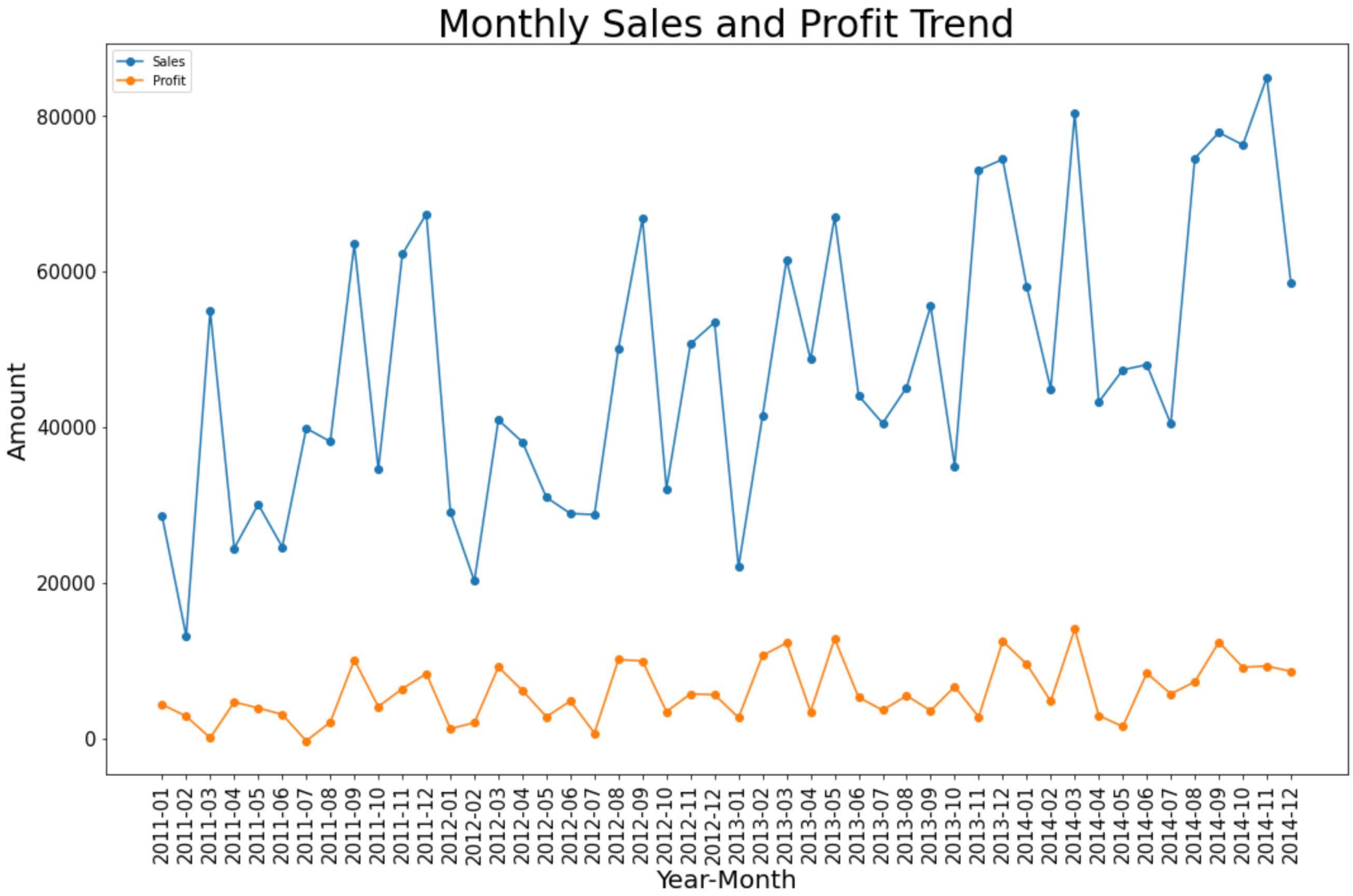
```
Sub-Category
Accessories    167380.3180
Appliances     107463.3510
Art             27118.7920
Binders        202953.4450
Bookcases      114879.9963
Chairs         328449.1030
Copiers        149528.0300
Envelopes       16476.4020
Fasteners        3024.2800
Furnishings     91663.2040
Labels          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 [24]:
```
# Pivot table :

pivot_table = pd.pivot_table(df, values=['Sales', 'Quantity', 'Discount', 'Profit'],
                             index=['Country', 'Region'],
                             columns=['Category', 'Sub-Category'],
                             aggfunc={'Sales': 'sum', 'Quantity': 'sum', 'Discount': 'mean', 'Profit': 'sum'}
                             fill_value=0)
print(pivot_table)
```

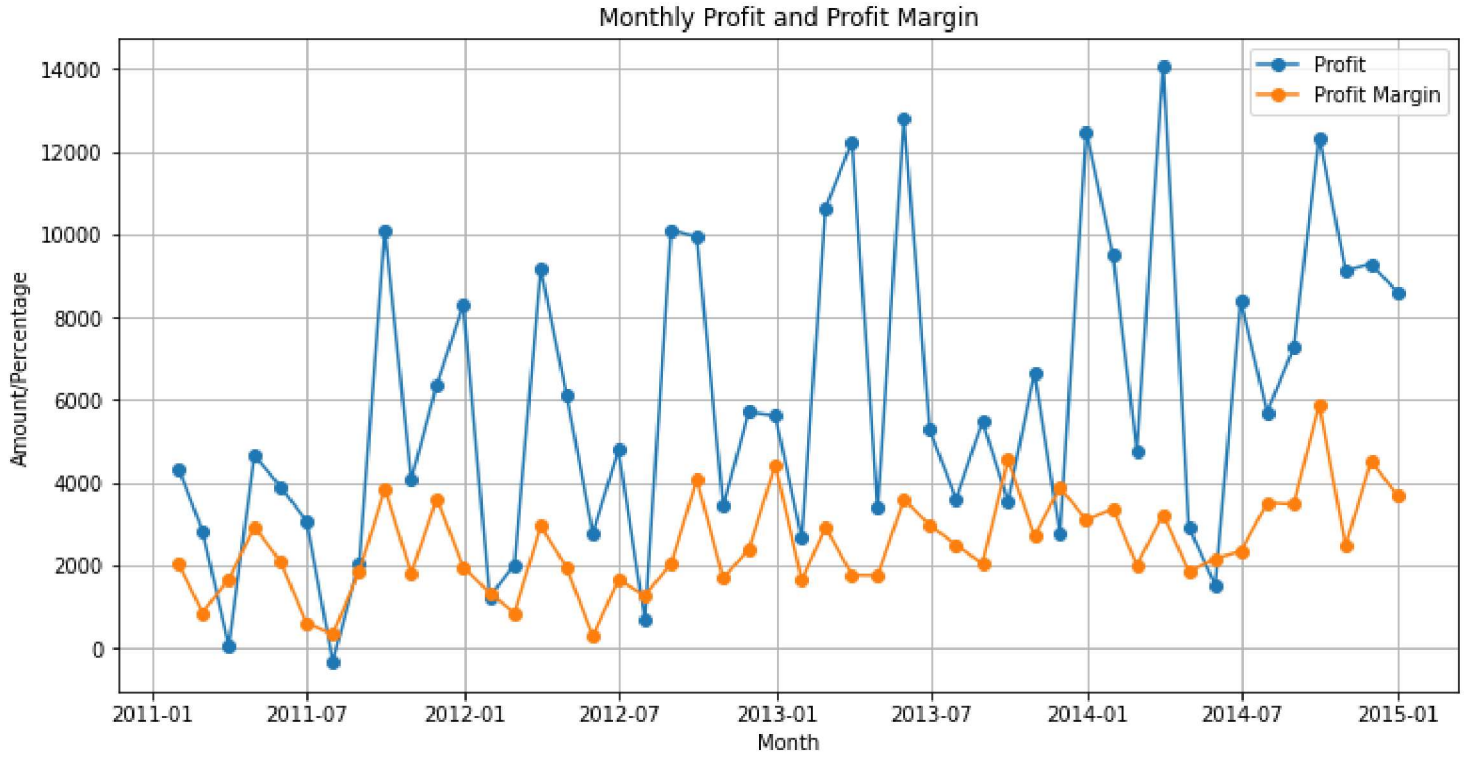
```
                     Discount                                    \
Category             Furniture
Sub-Category         Bookcases    Chairs Furnishings    Tables
Country       Region
United States Central   0.23280  0.192857    0.403902  0.262500
              East      0.22000  0.150595    0.078014  0.373750
              South     0.10000  0.097727    0.106667  0.222549
              West      0.22875  0.200000    0.032895  0.201754

                                                                  \
Category             Office Supplies
Sub-Category              Appliances       Art   Binders Envelopes Fasteners
Country       Region
United States Central       0.445902  0.122727  0.509290  0.128814  0.134545
              East          0.073016  0.069298  0.353409  0.081081  0.091803
              South         0.111111  0.101408  0.375610  0.074074  0.082759
              West          0.030882  0.031200  0.282090  0.041791  0.033333

                                ...           Sales                     \
Category                        ... Office Supplies
Sub-Category            Labels  ...       Envelopes Fasteners    Labels
Country       Region            ...
United States Central  0.113158  ...        4636.872   778.030  2451.472
              East     0.067290  ...        4375.874   819.718  2602.934
              South    0.107692  ...        3345.556   503.316  2353.180
              West     0.018966  ...        4118.100   923.216  5078.726

                                                                  \
Category                                            Technology
Sub-Category                Paper    Storage   Supplies Accessories    Copiers
Country       Region
United States Central  17491.902  45264.232   9467.372   33956.076  37259.570
              East     20172.602  71386.024  10760.116   45033.372  53219.462
              South    14135.432  35768.060   8318.928   27276.754   9299.756
              West     26663.718  70532.852  18127.122   61114.116  49749.242


Category
Sub-Category             Machines      Phones
Country       Region
United States Central  26797.384   72403.282
              East     66106.165  100614.982
              South    53890.960   58276.446
```

```
                West     42444.122   97772.928

[4 rows x 68 columns]
```

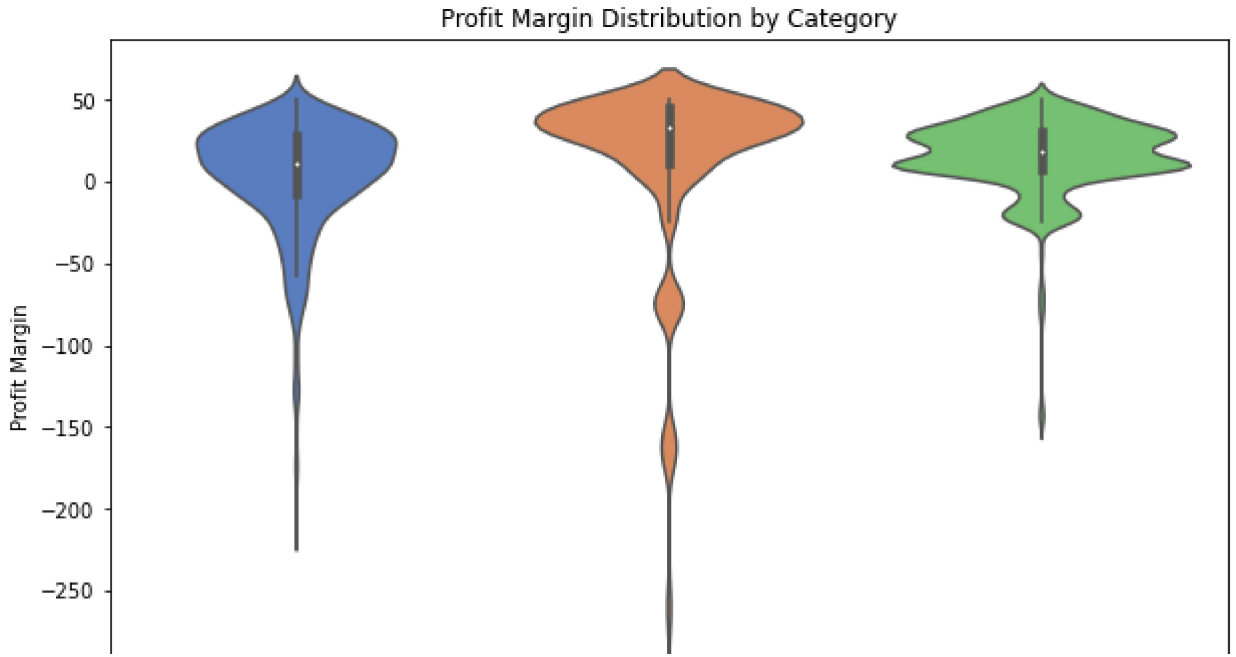
In [38]:
```
#monthly ales and profit trends
df['Order Date'] = pd.to_datetime(df['Order Date'])
df['YearMonth'] = df['Order Date'].dt.to_period('M')

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='-', label='Sales')

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

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()
```


Monthly Sales and Profit Trend
Sales
Profit
Amount
80000
60000
40000
20000
0
2011-01
2011-02
2011-03
2011-04
2011-05
2011-06
2011-07
2011-08
2011-09
2011-10
2011-11
2011-12
2012-01
2012-02
2012-03
2012-04
2012-05
2012-06
2012-07
2012-08
2012-09
2012-10
2012-11
2012-12
2013-01
2013-02
2013-03
2013-04
2013-05
2013-06
2013-07
2013-08
2013-09
2013-10
2013-11
2013-12
2014-01
2014-02
2014-03
2014-04
2014-05
2014-06
2014-07
2014-08
2014-09
2014-10
2014-11
2014-12
Year-Month

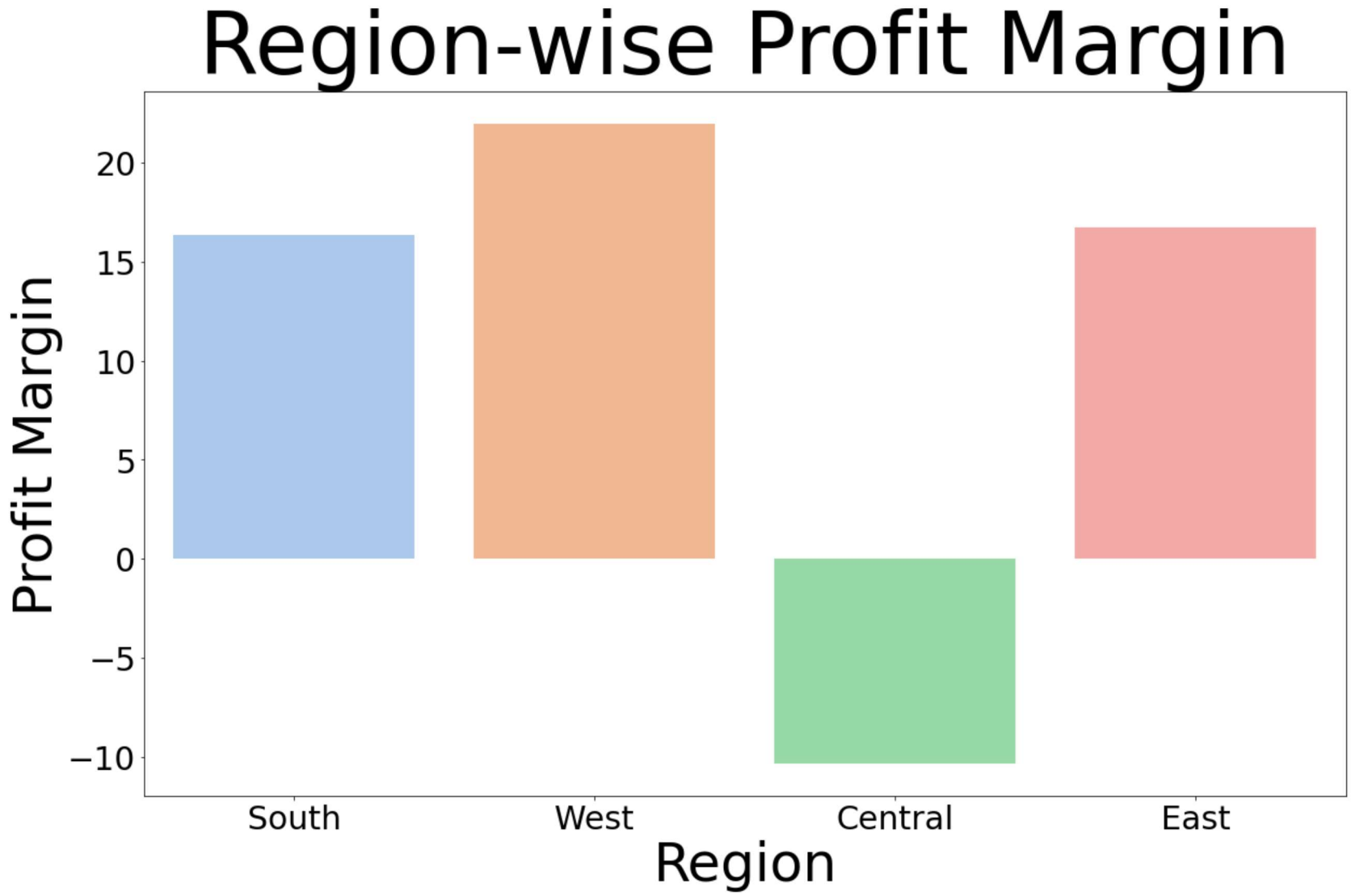
In [27]:
```
#Monthly Profit and Profit Margin :

df['Order Date'] = pd.to_datetime(df['Order Date'])
monthly_profit = df.resample('M', on='Order Date')[['Profit', 'Profit margin']].sum()
plt.figure(figsize=(12, 6))
plt.plot(monthly_profit.index, monthly_profit['Profit'], label='Profit', marker='o')
plt.plot(monthly_profit.index, monthly_profit['Profit margin'], label='Profit Margin', marker='o')
plt.title('Monthly Profit and Profit Margin')
plt.xlabel('Month')
plt.ylabel('Amount/Percentage')
plt.legend()
plt.grid(True)
plt.show()
```



In [28]:
```
#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 [29]:
```
#Region wise profit margin
plt.figure(figsize=(20, 12))
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)
plt.show()
```

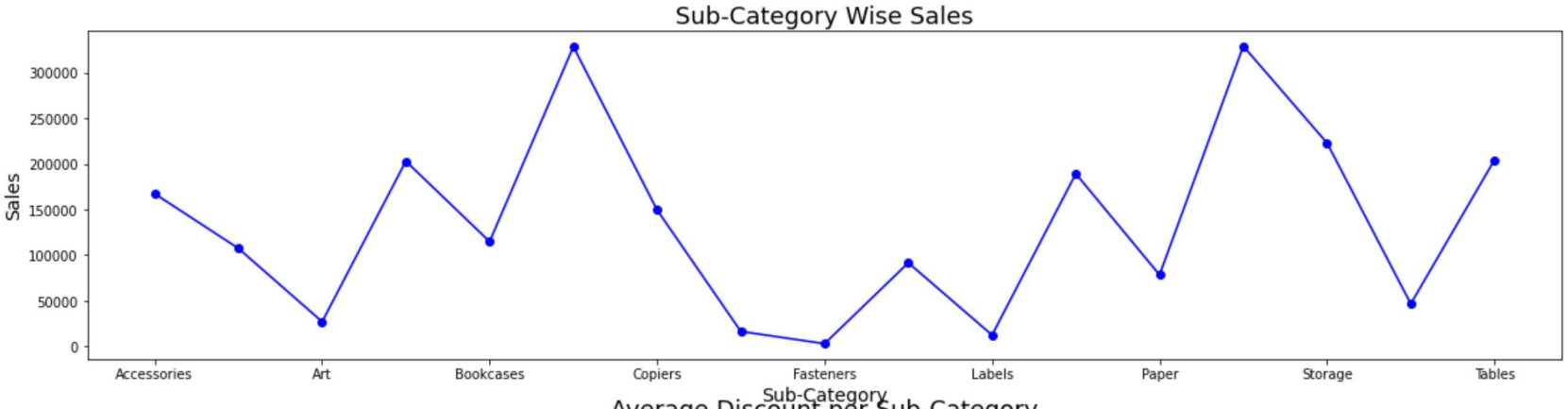

Region-wise Profit Margin
Profit Margin
20
15
10
5
0
−5
−10
South
West
Central
East
Region

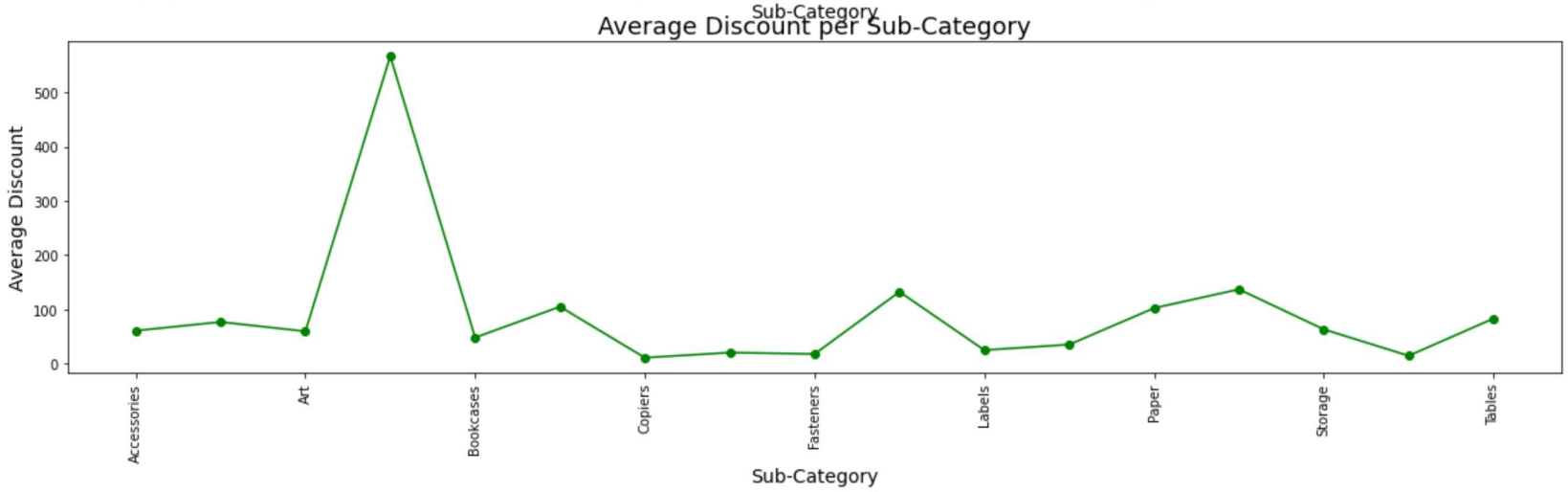
In [46]:
```
customer_segments = df.groupby('Segment')['Sales'].sum()
print("Customer Segmentation:")
print(customer_segments)
```

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

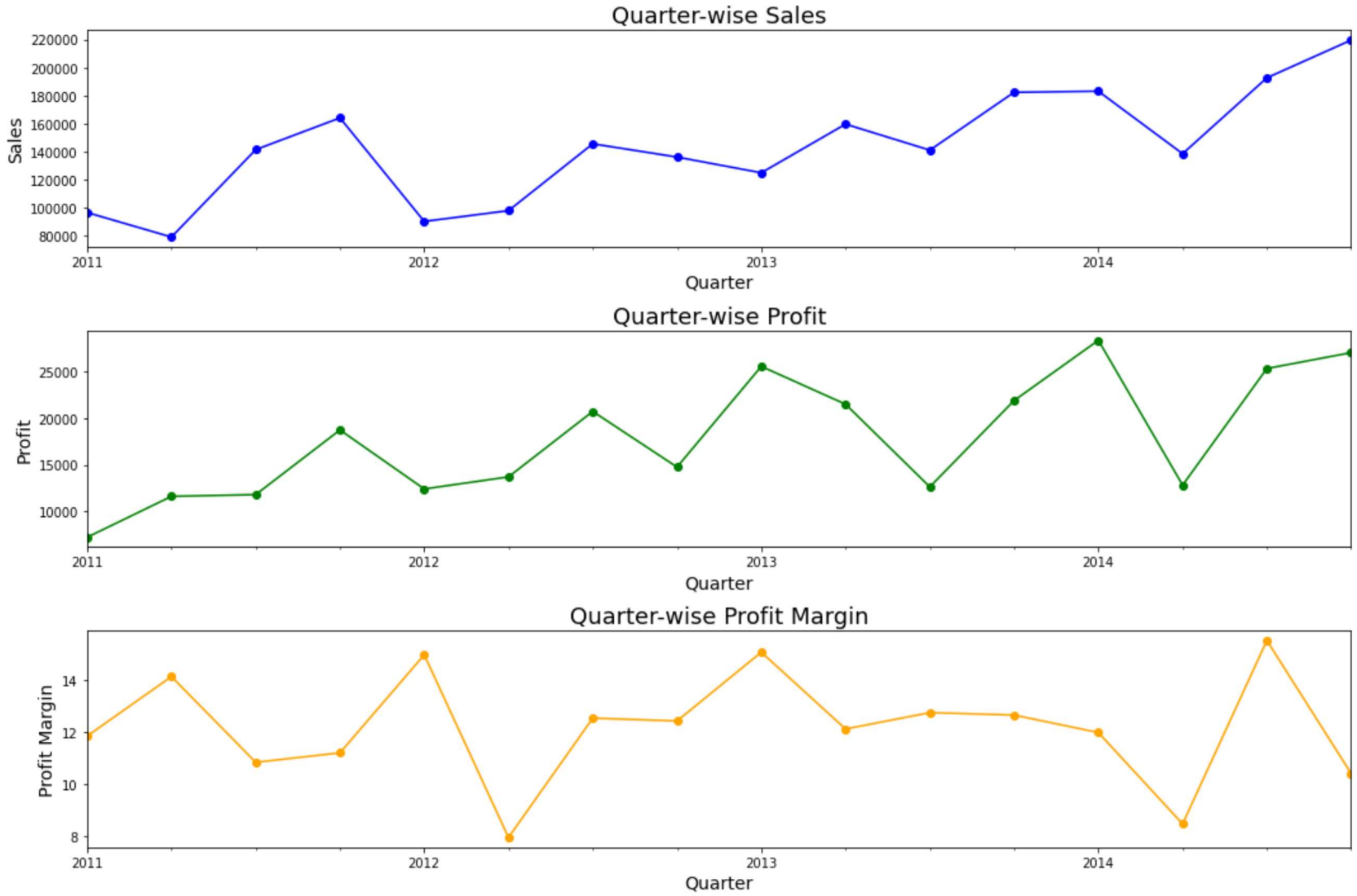
In [47]:
```
import matplotlib.pyplot as plt

# 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
```


Sub-Category Wise Sales
Sales
300000
250000
200000
150000
100000
50000
0
Accessories
Art
Bookcases
Copiers
Fasteners
Labels
Paper
Storage
Tables
Sub-Category


Average Discount per Sub-Category
Average Discount
500
400
300
200
100
0
Accessories
Art
Bookcases
Copiers
Fasteners
Labels
Paper
Storage
Tables
Sub-Category

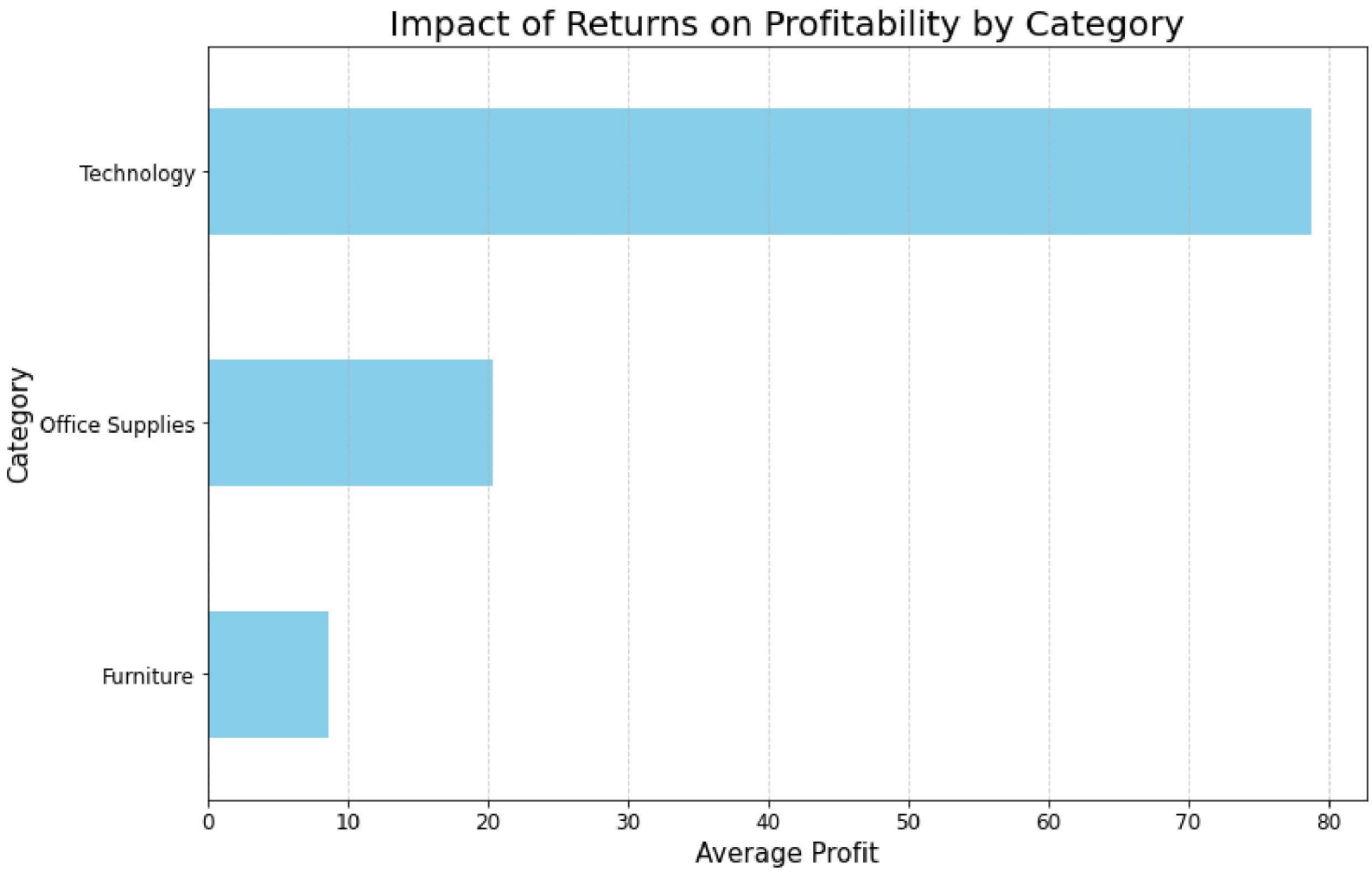
In [54]:
```
#quater wise sales, profit , profit margin trends
import pandas as pd
import matplotlib.pyplot as plt


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 [51]:
```
customer_retention = df.groupby('Customer ID')['Order Date'].min()
retention_rate = (customer_retention < '2023-01-01').sum() / len(customer_retention)
print(f"Customer retention rate: {retention_rate:.2%}")
```

```
Customer retention rate: 100.00%
```

In [53]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Assuming you have a DataFrame named df with columns 'Category' and 'Profit'

returns_analysis = df.groupby('Category')['Profit'].mean()

# Plotting the bar chart
plt.figure(figsize=(12, 8))
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)

plt.show()
```


Impact of Returns on Profitability by Category
Technology
Office Supplies
Furniture
Category
0
10
20
30
40
50
60
70
80
Average Profit