```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

```
from google.colab import files
uploaded = files.upload()
```

Choose Files Sales Data (KJG).xlsx

- **Sales Data (KJG).xlsx**(application/vnd.openxmlformats-officedocument.spreadsheetml.sheet) - 466893 bytes, last modified: 5/30/2025 - 100% done

```
Saving Sales Data (KJG).xlsx to Sales Data (KJG).xlsx
```

```
a=pd.read_excel('Sales Data (KJG).xlsx')
print(a.head())
```

```
   Row ID Order Priority  Discount  Unit Price  Shipping Cost  Customer ID  \
0   20847           High      0.01        2.84           0.93            3
1   20228  Not Specified      0.02      500.98          26.00            5
2   21776       Critical      0.06        9.48           7.29           11
3   24844         Medium      0.09       78.69          19.99           14
4   24846         Medium      0.08        3.28           2.31           14

       Customer Name       Ship Mode Customer Segment Product Category  ...  \
0      Bonnie Potter     Express Air        Corporate  Office Supplies  ...
1     Ronnie Proctor  Delivery Truck      Home Office        Furniture  ...
2      Marcus Dunlap     Regular Air      Home Office        Furniture  ...
3  Gwendolyn F Tyson     Regular Air   Small Business        Furniture  ...
4  Gwendolyn F Tyson     Regular Air   Small Business  Office Supplies  ...

          City Postal Code Order Date  Ship Date     Profit  \
0    Anacortes       98221 2015-01-07 2015-01-08     4.5600
1  San Gabriel       91776 2015-06-13 2015-06-15  4390.3665
2      Roselle        7203 2015-02-15 2015-02-17   -53.8096
3   Prior Lake       55372 2015-05-12 2015-05-14   803.4705
4   Prior Lake       55372 2015-05-12 2015-05-13   -24.0300

  Quantity ordered new    Sales Order ID  Return  Manager
0                    4    13.01    88522       -  William
1                   12  6362.85    90193       -  William
2                   22   211.15    90192       -     Erin
3                   16  1164.45    86838       -    Chris
4                    7    22.23    86838       -    Chris

[5 rows x 27 columns]
```

```
a.columns
```

```
Index(['Row ID', 'Order Priority', 'Discount', 'Unit Price', 'Shipping Cost',
       'Customer ID', 'Customer Name', 'Ship Mode', 'Customer Segment',
       'Product Category', 'Product Sub-Category', 'Product Container',
       'Product Name', 'Product Base Margin', 'Country', 'Region',
       'State or Province', 'City', 'Postal Code', 'Order Date', 'Ship Date',
       'Profit', 'Quantity ordered new', 'Sales', 'Order ID', 'Return',
       'Manager'],
      dtype='object')
```

Fund Null Values & drow Rows that Contains Null Values

```
print("Null Values by",a.isnull().sum())
print("\nTotal Null Values in the Data =",a.isnull().sum().sum())
```

```
Null Values by Row ID                   0
Order Priority           0
Discount                 0
Unit Price               0
Shipping Cost            0
Customer ID              0
Customer Name            0
Ship Mode                0
Customer Segment         0
Product Category         0
Product Sub-Category     0
Product Container        0
Product Name             0
Product Base Margin     16
Country                  0
Region                   0
State or Province        0
City                     0
Postal Code              0
```

```
Order Date               0
Ship Date                0
Profit                   0
Quantity ordered new     0
Sales                    0
Order ID                 0
Return                   0
Manager                  0
dtype: int64

Total Null Values in the Data = 16
```

```
#Remove rows that contains null values in a particular column
c = a.dropna(subset=['Product Base Margin'])
c.shape
```

```
(1936, 27)
```

```
#Remove rows that contains null values in any of their column
df = a[a.notnull().all(axis=1)]
print("Original Data Shape",a.shape)
print("Data Shape after remove null values",df.shape)
```

```
Original Data Shape (1952, 27)
Data Shape after remove null values (1936, 27)
```

Change String to Date format & Create Year, Month & Day as new column

```
#Convert Order date and ship date from string to date format
df['Order Date']=pd.to_datetime(df['Order Date'])
df['Ship Date']=pd.to_datetime(df['Ship Date'])

#Add 3 new columns of year, month, date from Order date
df['Year']=df['Order Date'].dt.year
df['Month']=df['Order Date'].dt.month
df['Day']=df['Order Date'].dt.day
```

Map month number to Month name & Order it

```
#Map month number to Month Name
#we have only 6 month data, so map january to june
df['Month']=df['Month'].map({1:'Jan',2:'Feb',3:'Mar',4:'Apr',5:'May',6:'Jun'})

# Define the correct order of months
month_order = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun']

# Convert 'Month' column to a categorical type with the specified order
df['Month'] = pd.Categorical(df['Month'], categories=month_order, ordered=True)
df['Month'] = df['Month']
```

```
print(df.dtypes)
```

```
Row ID                          int64
Order Priority                 object
Discount                      float64
Unit Price                    float64
Shipping Cost                 float64
Customer ID                     int64
Customer Name                  object
Ship Mode                      object
Customer Segment               object
Product Category               object
Product Sub-Category           object
Product Container              object
Product Name                   object
Product Base Margin           float64
Country                        object
Region                         object
State or Province              object
City                           object
Postal Code                     int64
Order Date             datetime64[ns]
Ship Date              datetime64[ns]
Profit                        float64
Quantity ordered new            int64
Sales                         float64
Order ID                        int64
Return                         object
Manager                        object
Year                            int32
```

```
Month                          category
Day                               int32
dtype: object
```

```
print("Shape in Original Data is",a.shape)
print("Shape after clear null values & add column is",df.shape)
```

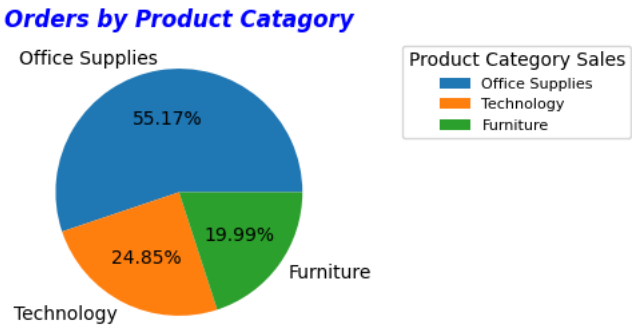
```
Shape in Original Data is (1952, 27)
Shape after clear null values & add column is (1936, 30)
```

Pie Plot for product catagery - Order, Sales & Profit

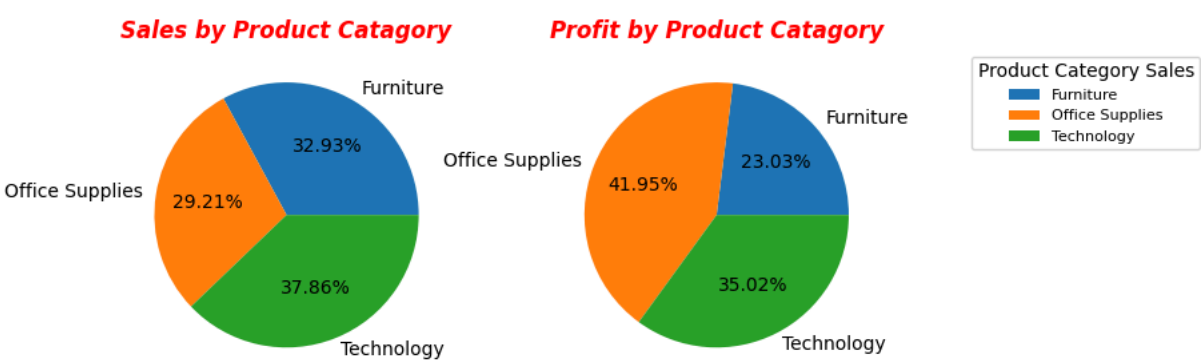
```
plt.style.use('default')
```

```
#Pie Plot Creation
plt.figure(figsize=(4,3))
pc_count = df['Product Category'].value_counts()  #or USE df.value_counts('Product Category')
          #use value_count() because the values has duplicates (count the uniques and its counts)
plt.pie(pc_count,autopct='%1.2f%%',labels=pc_count.index)
plt.title('Orders by Product Catagory',fontsize=12,color='b',style='oblique',fontweight='bold')
plt.legend(title='Product Category Sales',bbox_to_anchor=(1.2, 1),fontsize=8)
```

```
<matplotlib.legend.Legend at 0x79e7a81d6660>
```



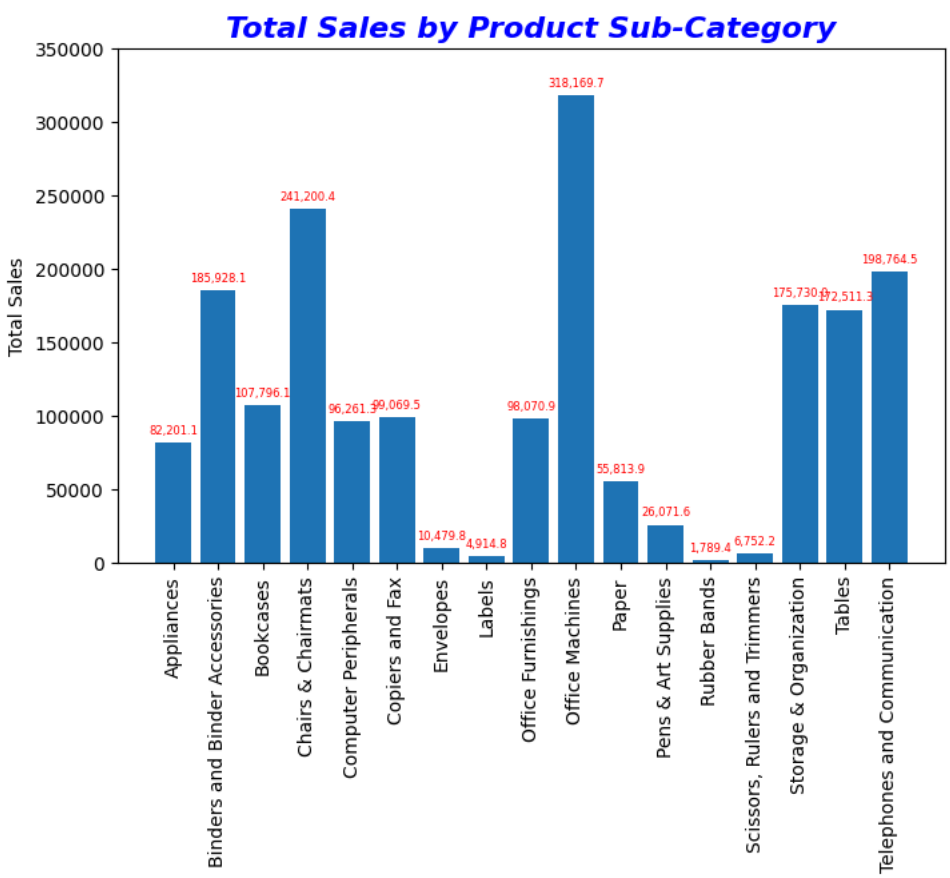

```
plt.figure(figsize=(9,3))

plt.subplot(1,2,1)
pc_sales=df.groupby('Product Category')['Sales'].sum()
plt.pie(pc_sales,autopct='%1.2f%%',labels=pc_sales.index)
plt.title('Sales by Product Catagory',fontsize=12,color='r',style='oblique',fontweight='bold')

plt.subplot(1,2,2)
pc_sales=df.groupby('Product Category')['Profit'].sum()
plt.pie(pc_sales,autopct='%1.2f%%',labels=pc_sales.index)
plt.legend(title='Product Category Sales', bbox_to_anchor=(1.25, 1),fontsize=8)
plt.title('Profit by Product Catagory',fontsize=12,color='r',style='oblique',fontweight='bold')

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



Bar Chart of Sales by Product Sub-Category

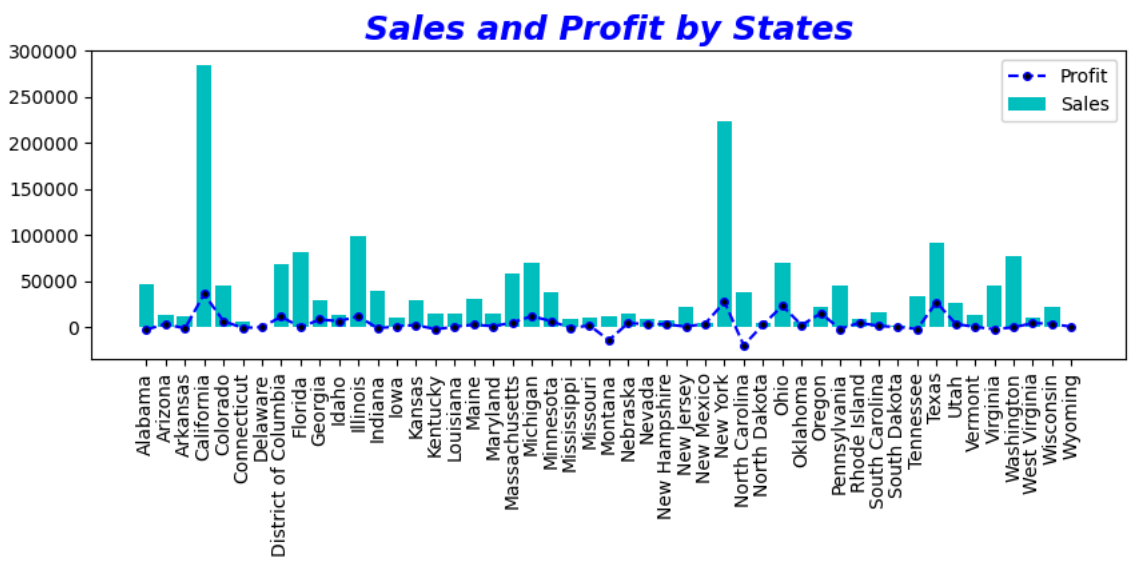
```
#Bar Plot with Labels
plt.figure(figsize=(8,5))
psc_group = df.groupby('Product Sub-Category')['Sales'].sum()
```

```
bars = plt.bar(psc_group.index, psc_group.values)
plt.ylabel('Total Sales')
plt.title('Total Sales by Product Sub-Category',fontsize=16,color='b',style='oblique',fontweight='bold')
plt.xticks(rotation=90)
plt.bar_label(bars, fmt='{:,.1f}', fontsize=6, color='r', padding = 3)
plt.ylim(0,350000)
plt.show()
```



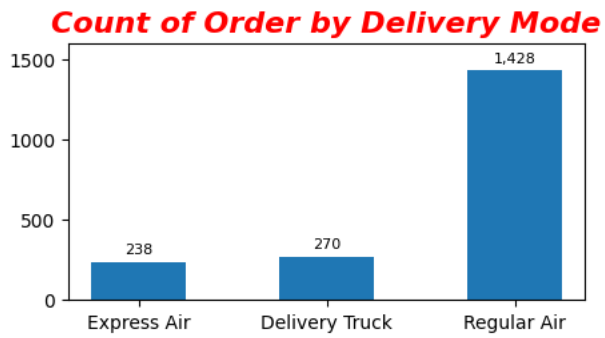
Bar & Line chart of Sales and Profit Country Wise

```
plt.figure(figsize=(10,3))  #Sets the size of the figure in inch
state_sales = df.groupby('State or Province')['Sales'].sum()
state_profit = df.groupby('State or Province')['Profit'].sum()
plt.bar(state_sales.index,state_sales.values,color='c')
ax2 = plt.gca() #-- plt.gca().twinx() is for dual axis
ax2.plot(state_profit.index,state_profit.values,'o--b',ms=4,mfc='k')
plt.title('Sales and Profit by States',fontsize=18,color='b',style='oblique',fontweight='bold')
plt.legend(['Profit','Sales'])
plt.xticks(rotation=90)
plt.show()
```

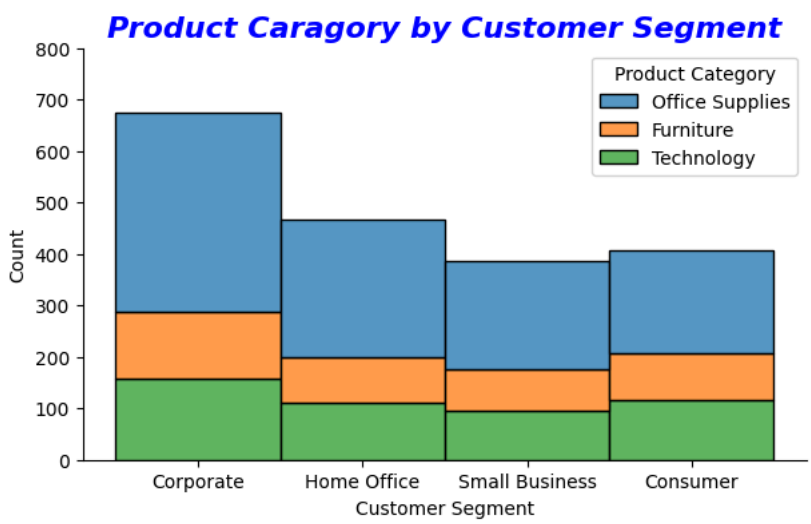


Order count by Ship Mode & Customer Segment

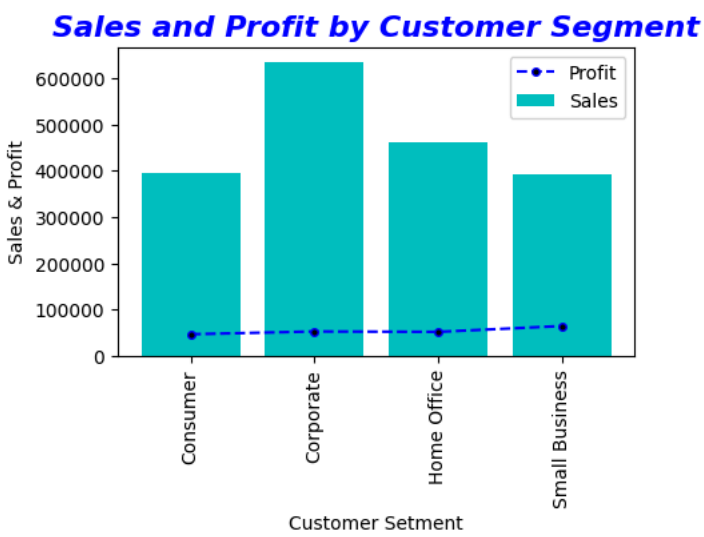
```
plt.figure(figsize=(5,2.5))  #Sets the size of the figure in inch
ship_count = df['Ship Mode'].value_counts(ascending=True)  #ascending=False for Desending
lable = plt.bar(ship_count.index,ship_count.values, width=0.5)
plt.title('Count of Order by Delivery Mode',fontsize=16,color='r',style='oblique',fontweight='bold')
plt.bar_label(lable, fmt='{:,.0f}', fontsize=8, color='k', padding = 3)
plt.ylim(0,1600)
plt.show()
```



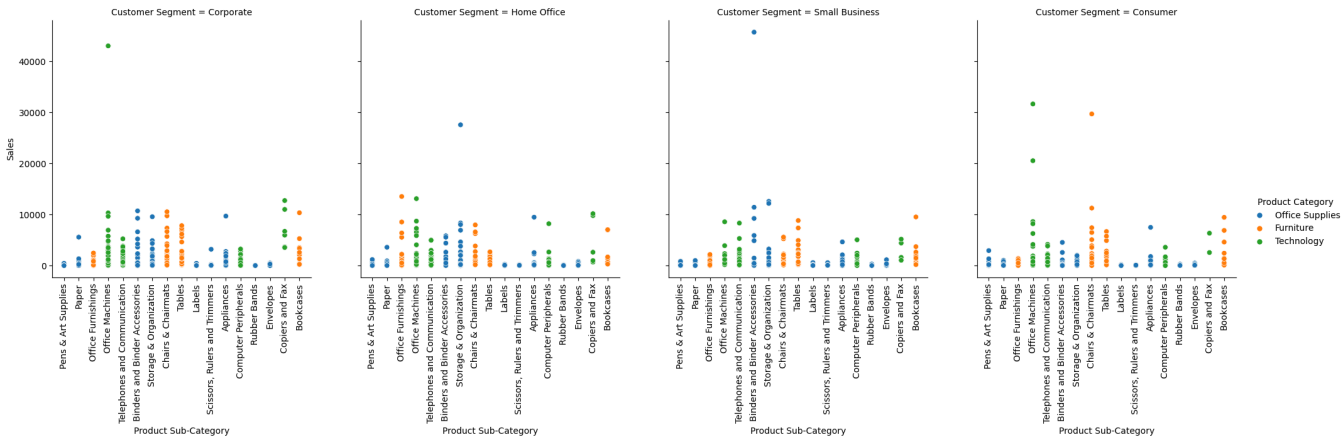

```
plt.figure(figsize=(7,4))
sns.histplot(data=df,x=df['Customer Segment'],hue=df['Product Category'],multiple="stack")
plt.title('Product Caragory by Customer Segment',fontsize=16,color='b',style='oblique',fontweight='bold')
plt.ylim(0,800)
sns.despine()
```



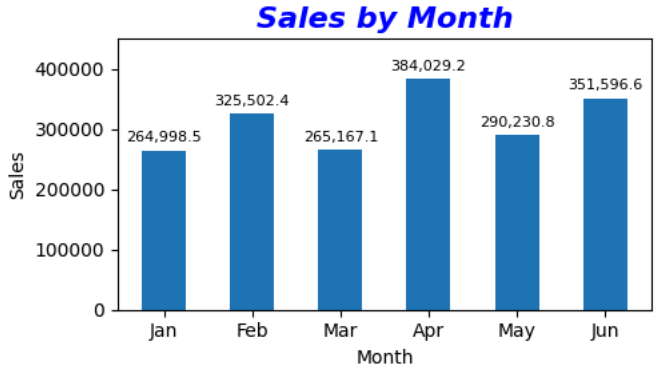

```
plt.figure(figsize=(5,3))  #Sets the size of the figure in inch
cs_sales = df.groupby('Customer Segment')['Sales'].sum()
cs_profit = df.groupby('Customer Segment')['Profit'].sum()
plt.bar(cs_sales.index,cs_sales.values,color='c')
ax2 = plt.gca() #-- plt.gca().twinx() is for dual axis
ax2.plot(cs_profit.index,cs_profit.values,'o--b',ms=4,mfc='k')
plt.xlabel('Customer Setment')
plt.ylabel('Sales & Profit')
plt.title('Sales and Profit by Customer Segment',fontsize=16,color='b',style='oblique',fontweight='bold')
plt.legend(['Profit','Sales'])
plt.xticks(rotation=90)
plt.show()
```

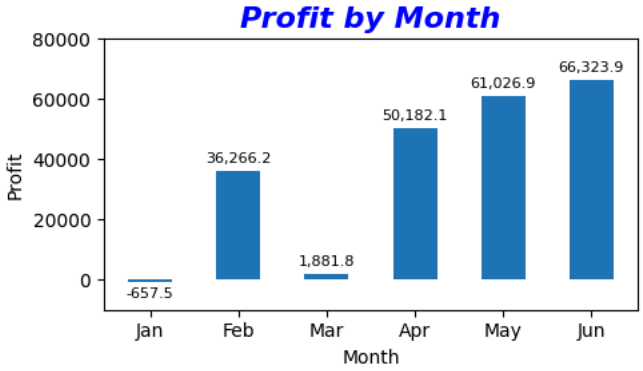



```
sns.relplot(data=df,x=df['Product Sub-Category'],y=df['Sales'],col=df['Customer Segment'],hue=df['Product Category'])
for ax in plt.gcf().axes:  #plt.gcf() is Get Current Figure
    for label in ax.get_xticklabels():
        label.set_rotation(90);
```



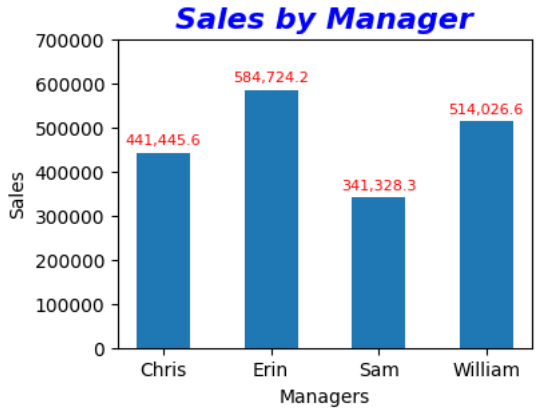

```
plt.figure(figsize=(10,3)) # Adjust figure size as needed


plt.subplot(1,2,1)
month_sales = df.groupby('Month')['Sales'].sum()
ms_lable = plt.bar(month_sales.index,month_sales.values,width=.5)
plt.bar_label(ms_lable, fmt='{:,.1f}', fontsize=8, color='k', padding = 3)
plt.title('Sales by Month',fontsize=16,color='b',style='oblique',fontweight='bold')
plt.ylim(0,450000)
plt.xlabel('Month')
plt.ylabel('Sales')


plt.subplot(1,2,2)
month_profit = df.groupby('Month')['Profit'].sum()
mp_label = plt.bar(month_profit.index,month_profit.values,width=.5)
plt.title('Profit by Month',fontsize=16,color='b',style='oblique',fontweight='bold')
plt.bar_label(mp_label, fmt='{:,.1f}', fontsize=8, color='k', padding = 3)
plt.ylim(-10000,80000)
plt.xlabel('Month')
plt.ylabel('Profit')

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





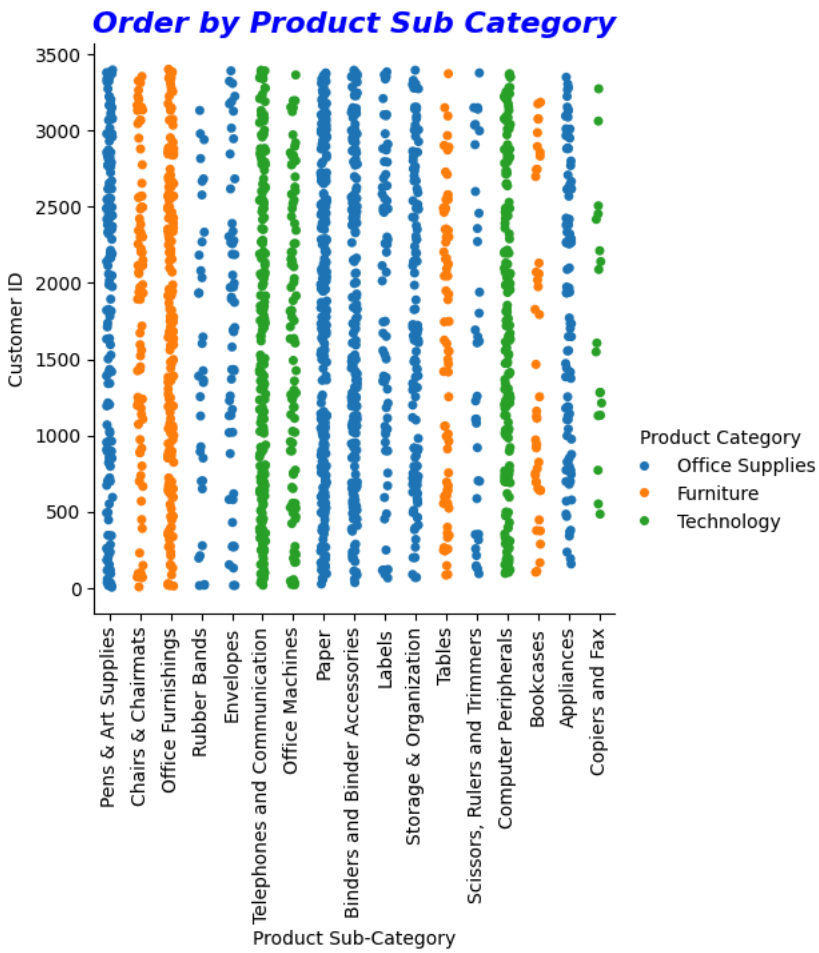

```
plt.figure(figsize=(4,3))
manager_sales = df.groupby('Manager')['Sales'].sum()
msale_label = plt.bar(manager_sales.index,manager_sales.values,width=.5)
plt.title('Sales by Manager',fontsize=16,color='b',style='oblique',fontweight='bold')
plt.bar_label(msale_label, fmt='{:,.1f}', fontsize=8, color='r', padding = 3)
plt.ylim(0,700000)
plt.xlabel('Managers')
plt.ylabel('Sales')
```

```
Text(0, 0.5, 'Sales')
```

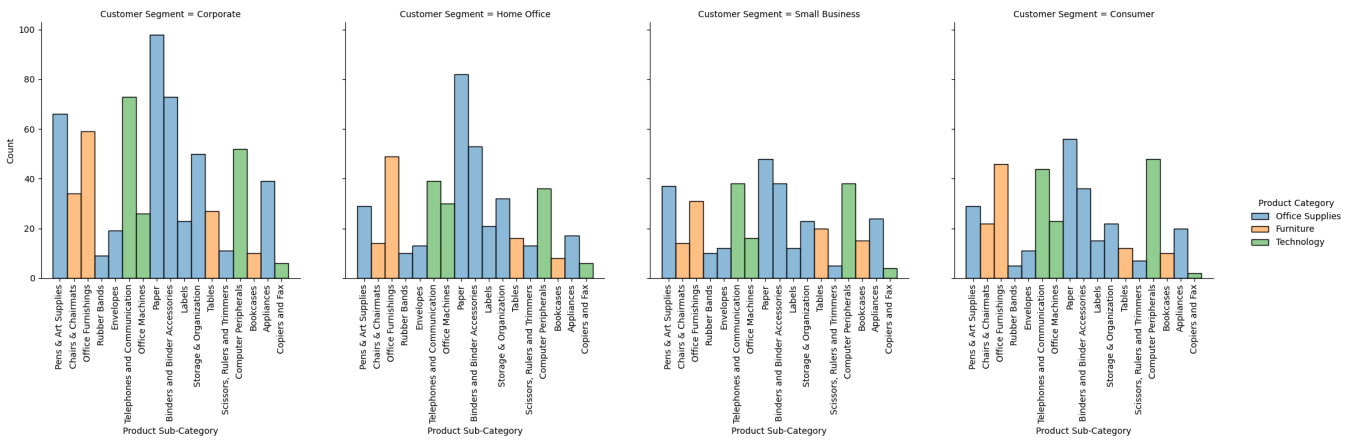


Customer Behaviour

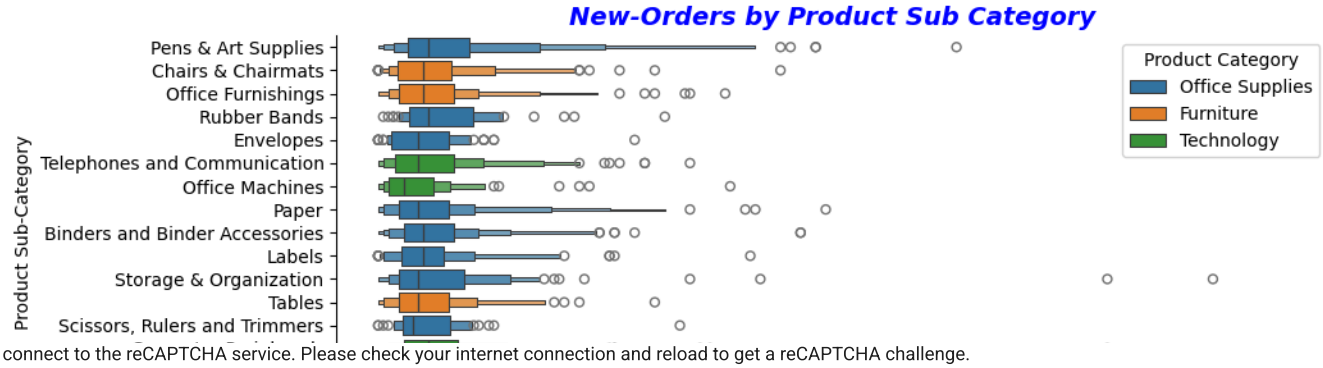
```
sns.catplot(data=df,x=df['Product Sub-Category'],y=df['Customer ID'], hue=df['Product Category'])
plt.title('Order by Product Sub Category',fontsize=16,color='b',style='oblique',fontweight='bold')
plt.xticks(rotation=90); #semi-colon(;) - used to Remove descriptive text of the FacetGrid object appears before the chart
```




```
sns.displot(data=df,x=df['Product Sub-Category'],col=df['Customer Segment'],hue=df['Product Category'])
for ax in plt.gcf().axes:
    for label in ax.get_xticklabels():
        label.set_rotation(90);
```




```
plt.figure(figsize=(10,4)) # Adjust figure size as needed
sns.boxenplot(x=df['Quantity ordered new'],y=df['Product Sub-Category'],hue=df['Product Category'],data=df)
plt.title('New-Orders by Product Sub Category',fontsize=14,color='b',style='oblique',fontweight='bold')
lim_x=np.arange(0,200,10)
sns.despine()
plt.xticks(lim_x,rotation=90);
```



Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.