Exploratory Data Analysis on Dataset 'SampleSuperstore'

Aim As a business manager, try to find out the weak areas where you can work to make more profit.

Exploratory Data Analysis (EDA) is a data science technique that involves analyzing and summarizing datasets to uncover patterns, trends, and insights.The main advantage of EDA is providing the data visualization of data after conducting the analysis.

Author: Ramchandra Darade

Importing Required libraries

In [6]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('ggplot')
```

Loading the dataset

In [9]:
```
df= pd.read_csv("C:/Users/ARCHANA/Desktop/The Sparks Foundation/Sample
```

Data Exploration/ Understanding

In [10]:
```
df.head()
```

Out[10]:

	Ship Mode	Segment	Country	City	State	Postal Code	Region	Category	Sub-Category
0	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Bookcases
1	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Chairs
2	Second Class	Corporate	United States	Los Angeles	California	90036	West	Office Supplies	Labels
3	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Furniture	Tables
4	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Office Supplies	Storage

In [11]:
```
df.tail()
```

Out[11]:

	Ship Mode	Segment	Country	City	State	Postal Code	Region	Category	Cat
9989	Second Class	Consumer	United States	Miami	Florida	33180	South	Furniture	Furni
9990	Standard Class	Consumer	United States	Costa Mesa	California	92627	West	Furniture	Furni
9991	Standard Class	Consumer	United States	Costa Mesa	California	92627	West	Technology	P
9992	Standard Class	Consumer	United States	Costa Mesa	California	92627	West	Office Supplies	
9993	Second Class	Consumer	United States	Westminster	California	92683	West	Office Supplies	Appli

Model Building

In [12]:
```
df.describe()
```

Out[12]:

	Postal Code	Sales	Quantity	Discount	Profit
count	9994.000000	9994.000000	9994.000000	9994.000000	9994.000000
mean	55190.379428	229.858001	3.789574	0.156203	28.656896
std	32063.693350	623.245101	2.225110	0.206452	234.260108
min	1040.000000	0.444000	1.000000	0.000000	-6599.978000
25%	23223.000000	17.280000	2.000000	0.000000	1.728750
50%	56430.500000	54.490000	3.000000	0.200000	8.666500
75%	90008.000000	209.940000	5.000000	0.200000	29.364000
max	99301.000000	22638.480000	14.000000	0.800000	8399.976000

In [20]:
```
df.dtypes
```

Out[20]:
```
Ship Mode        object
Segment          object
Country          object
City             object
State            object
Postal Code       int64
Region           object
Category         object
Sub-Category     object
Sales           float64
Quantity          int64
Discount        float64
Profit          float64
dtype: object
```

In [13]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 13 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Ship Mode     9994 non-null   object
 1   Segment       9994 non-null   object
 2   Country       9994 non-null   object
 3   City          9994 non-null   object
 4   State         9994 non-null   object
 5   Postal Code   9994 non-null   int64
 6   Region        9994 non-null   object
 7   Category      9994 non-null   object
 8   Sub-Category  9994 non-null   object
 9   Sales         9994 non-null   float64
 10  Quantity      9994 non-null   int64
 11  Discount      9994 non-null   float64
 12  Profit        9994 non-null   float64
dtypes: float64(3), int64(2), object(8)
memory usage: 1015.1+ KB
```

In [14]:
```
df.isnull().sum()
```

Out[14]:
```
Ship Mode       0
Segment         0
Country         0
City            0
State           0
Postal Code     0
Region          0
Category        0
Sub-Category    0
Sales           0
Quantity        0
Discount        0
Profit          0
dtype: int64
```

In [16]:
```
df.columns
```

Out[16]:
```
Index(['Ship Mode', 'Segment', 'Country', 'City', 'State', 'Postal Code',
       'Region', 'Category', 'Sub-Category', 'Sales', 'Quantity', 'Discou
nt',
       'Profit'],
      dtype='object')
```

In [17]:
```
df.nunique()
```

Out[17]:
```
Ship Mode          4
Segment            3
Country            1
City             531
State             49
Postal Code      631
Region             4
Category           3
Sub-Category      17
Sales           5825
Quantity          14
Discount          12
Profit          7287
dtype: int64
```

In [21]:

```
df = df[['Ship Mode', 'Segment', 'City', 'State','Postal Code',
    'Category', 'Sub-Category', 'Sales', 'Quantity', 'Discount','Profi
df
```

Out[21]:

	Ship Mode	Segment	City	State	Postal Code	Category	Sub-Category	Sales	(
0	Second Class	Consumer	Henderson	Kentucky	42420	Furniture	Bookcases	261.9600	
1	Second Class	Consumer	Henderson	Kentucky	42420	Furniture	Chairs	731.9400	
2	Second Class	Corporate	Los Angeles	California	90036	Office Supplies	Labels	14.6200	
3	Standard Class	Consumer	Fort Lauderdale	Florida	33311	Furniture	Tables	957.5775	
4	Standard Class	Consumer	Fort Lauderdale	Florida	33311	Office Supplies	Storage	22.3680	
...	...	...	...	...	...	...	...	...	
9989	Second Class	Consumer	Miami	Florida	33180	Furniture	Furnishings	25.2480	
9990	Standard Class	Consumer	Costa Mesa	California	92627	Furniture	Furnishings	91.9600	
9991	Standard Class	Consumer	Costa Mesa	California	92627	Technology	Phones	258.5760	
9992	Standard Class	Consumer	Costa Mesa	California	92627	Office Supplies	Paper	29.6000	
9993	Second Class	Consumer	Westminster	California	92683	Office Supplies	Appliances	243.1600	

9994 rows × 11 columns

In [23]:

```
df.loc[df.duplicated()].shape
```

Out[23]: (17, 11)

In [25]:

```
df.drop_duplicates(inplace=True)
df.reset_index(drop=True,inplace=True)
df
```

Out[25]:

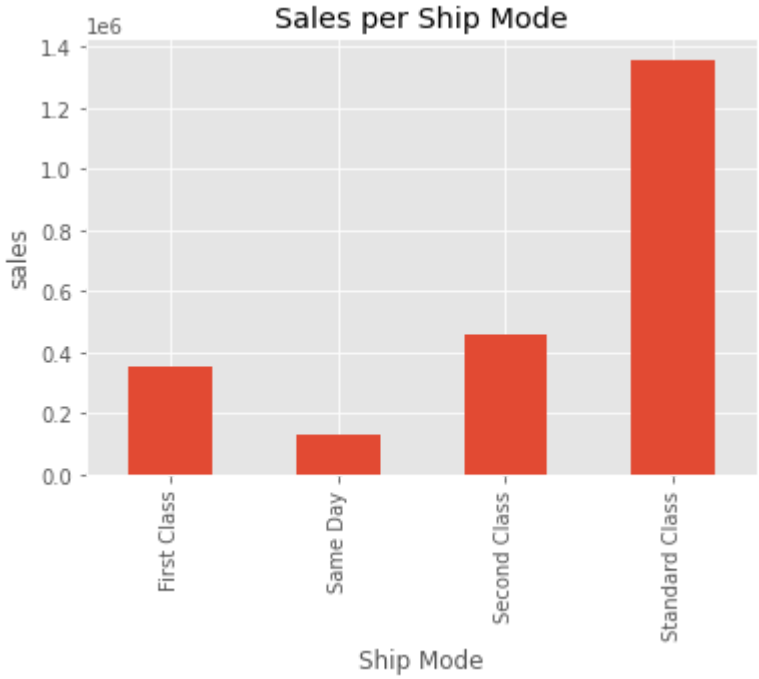
	Ship Mode	Segment	City	State	Postal Code	Category	Sub-Category	Sales	C
0	Second Class	Consumer	Henderson	Kentucky	42420	Furniture	Bookcases	261.9600	
1	Second Class	Consumer	Henderson	Kentucky	42420	Furniture	Chairs	731.9400	
2	Second Class	Corporate	Los Angeles	California	90036	Office Supplies	Labels	14.6200	
3	Standard Class	Consumer	Fort Lauderdale	Florida	33311	Furniture	Tables	957.5775	
4	Standard Class	Consumer	Fort Lauderdale	Florida	33311	Office Supplies	Storage	22.3680	
...	...	...	...	...	...	...	...	...	
9972	Second Class	Consumer	Miami	Florida	33180	Furniture	Furnishings	25.2480	
9973	Standard Class	Consumer	Costa Mesa	California	92627	Furniture	Furnishings	91.9600	
9974	Standard Class	Consumer	Costa Mesa	California	92627	Technology	Phones	258.5760	
9975	Standard Class	Consumer	Costa Mesa	California	92627	Office Supplies	Paper	29.6000	
9976	Second Class	Consumer	Westminster	California	92683	Office Supplies	Appliances	243.1600	

9977 rows × 11 columns

In [33]:

```
sales_by_shipmode=  df.groupby('Ship Mode')['Sales'].sum()
ax= sales_by_shipmode.plot(kind='bar',title= 'Sales per Ship Mode')
ax.set_xlabel('Ship Mode')
ax.set_ylabel('sales')
print('sales_by_ shipmode')
plt.show()

```

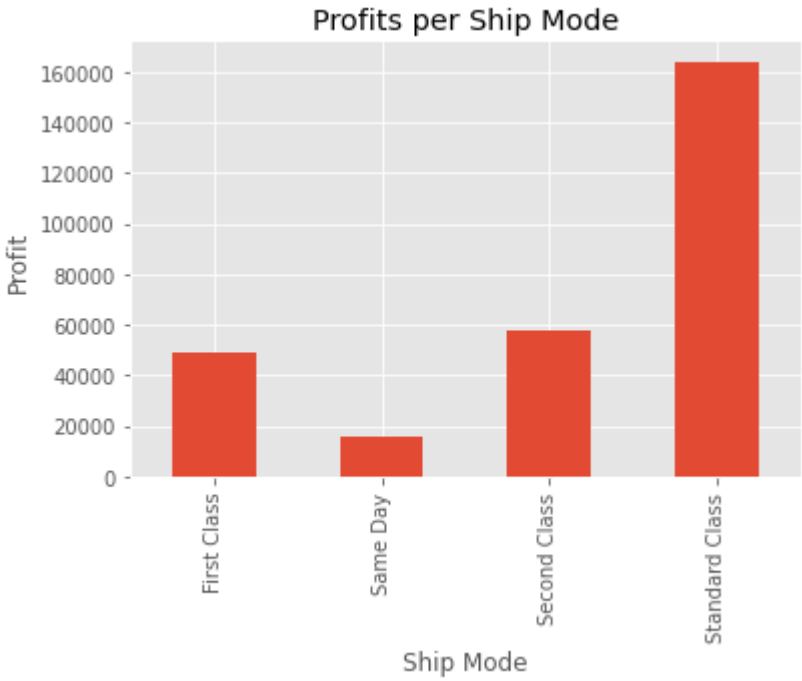
```
sales_by_ shipmode
```



In [34]:

```
profit_by_shipmode = df.groupby('Ship Mode')['Profit'].sum()
ax = profit_by_shipmode.plot(kind='bar',title='Profits per Ship Mode')
ax.set_xlabel('Ship Mode')
ax.set_ylabel('Profit')
print(profit_by_shipmode)
plt.show()
```

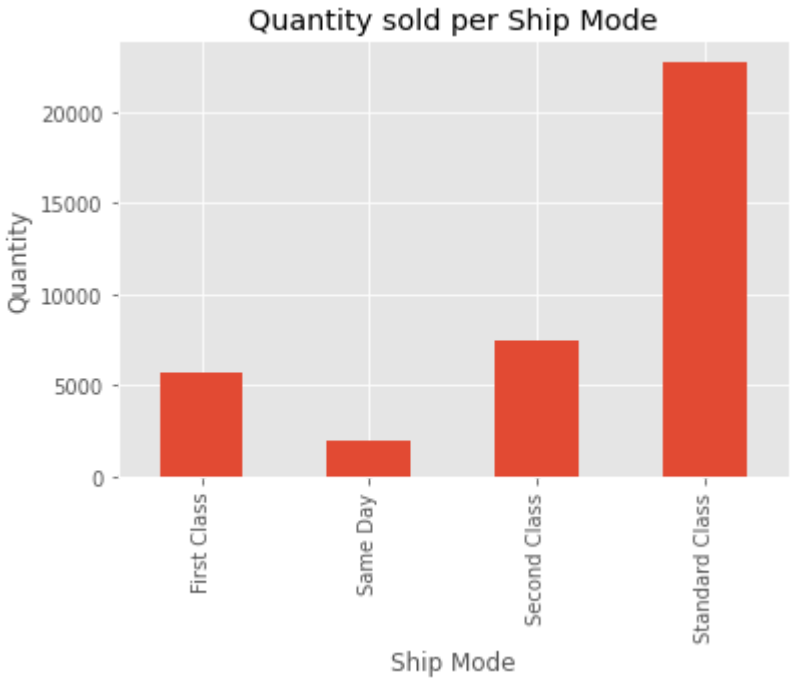
```
Ship Mode
First Class        48953.6561
Same Day           15871.8869
Second Class       57446.6516
Standard Class    163969.2280
Name: Profit, dtype: float64
```



In [35]:

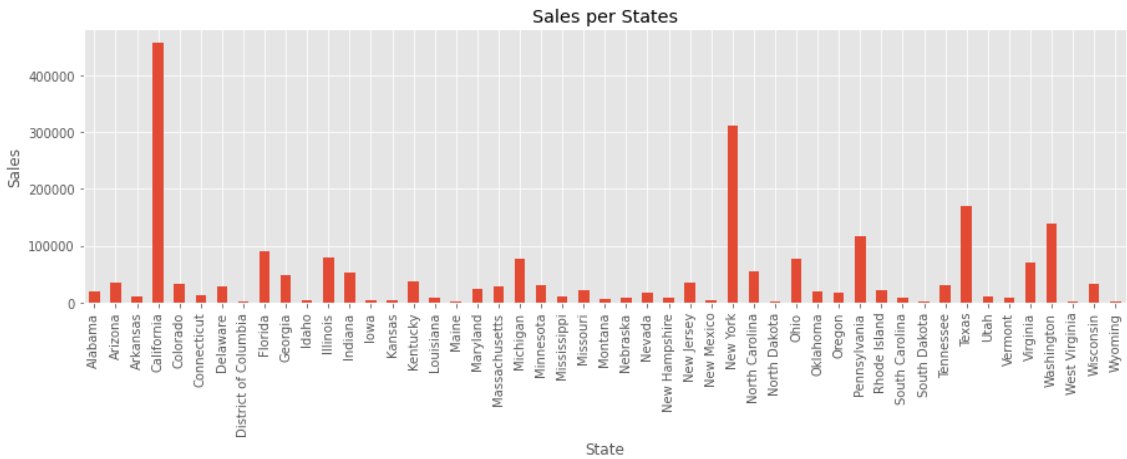
```
quantity_by_shipmode = df.groupby('Ship Mode')['Quantity'].sum()
ax = quantity_by_shipmode.plot(kind='bar',title='Quantity sold per Shi
ax.set_xlabel('Ship Mode')
ax.set_ylabel('Quantity')
print(quantity_by_shipmode)
plt.show()
```

```
Ship Mode
First Class        5690
Same Day           1956
Second Class       7418
Standard Class    22756
Name: Quantity, dtype: int64
```



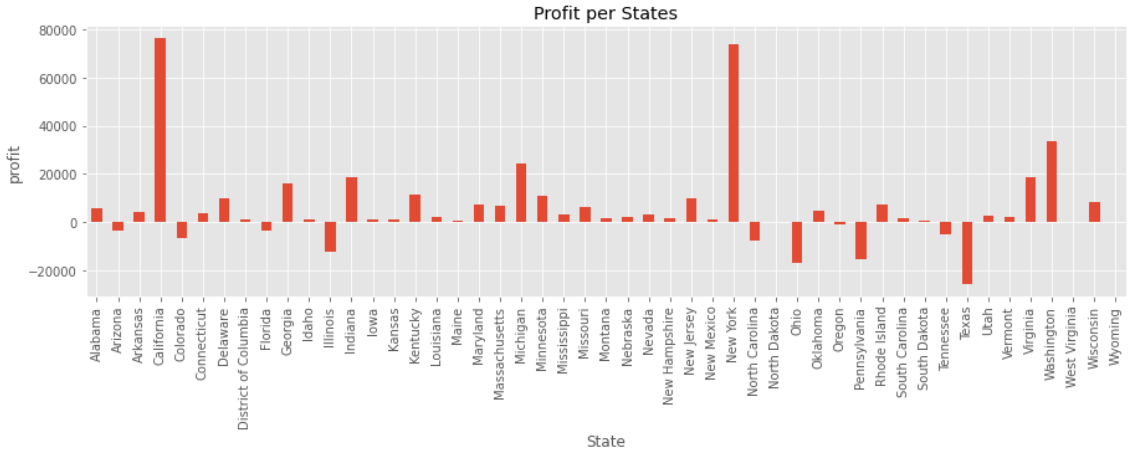
In [36]:

```
Sales_per_state = df.groupby('State')['Sales'].sum()
cx=Sales_per_state.plot(kind='bar',title='Sales per States',figsize=(1
cx.set_ylabel('Sales')
plt.show()
```



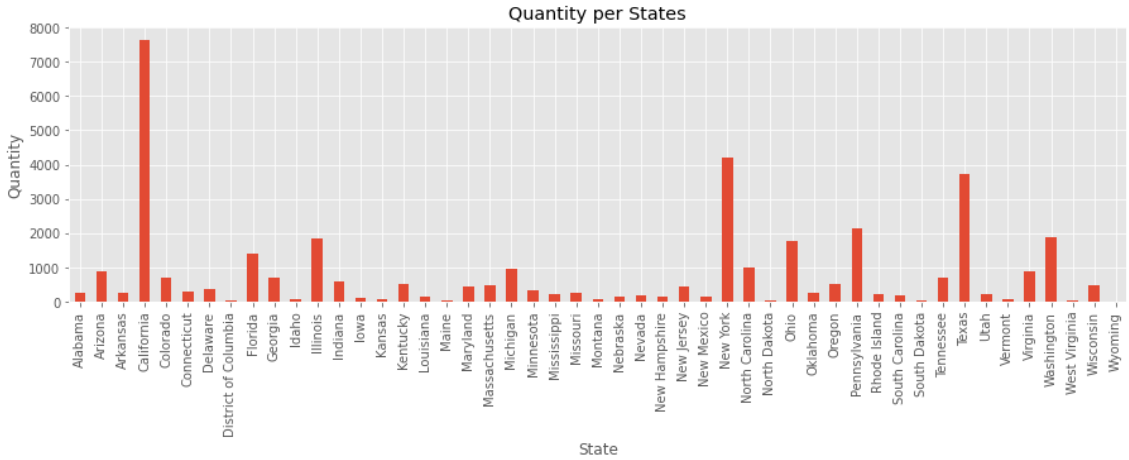
In [37]:

```
Profit_per_state = df.groupby('State')['Profit'].sum()
dx=Profit_per_state.plot(kind='bar',title='Profit per States',figsize=
dx.set_ylabel('profit')
plt.show()
```



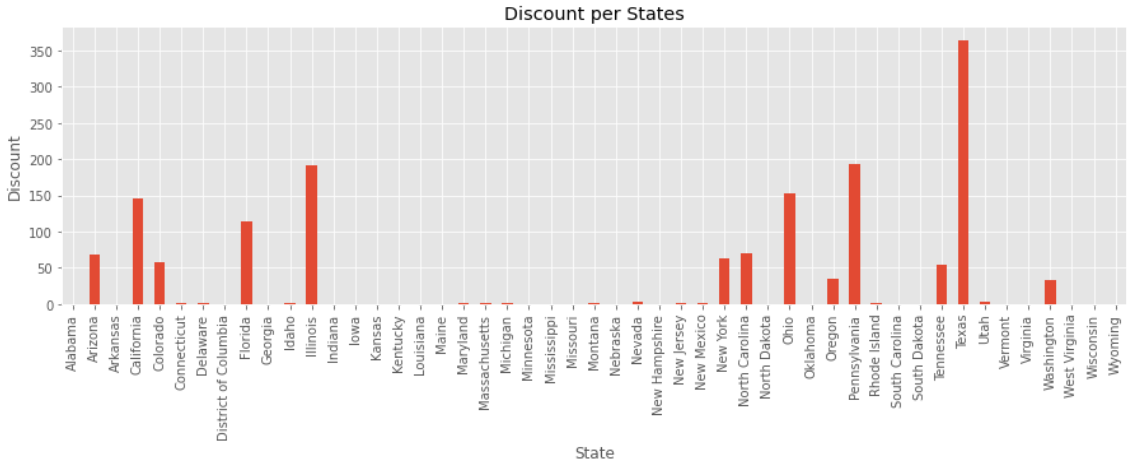
In [38]:

```
Quantity_per_state = df.groupby('State')['Quantity'].sum()
dx=Quantity_per_state.plot(kind='bar',title='Quantity per States',figs
dx.set_ylabel('Quantity')
plt.show()
```



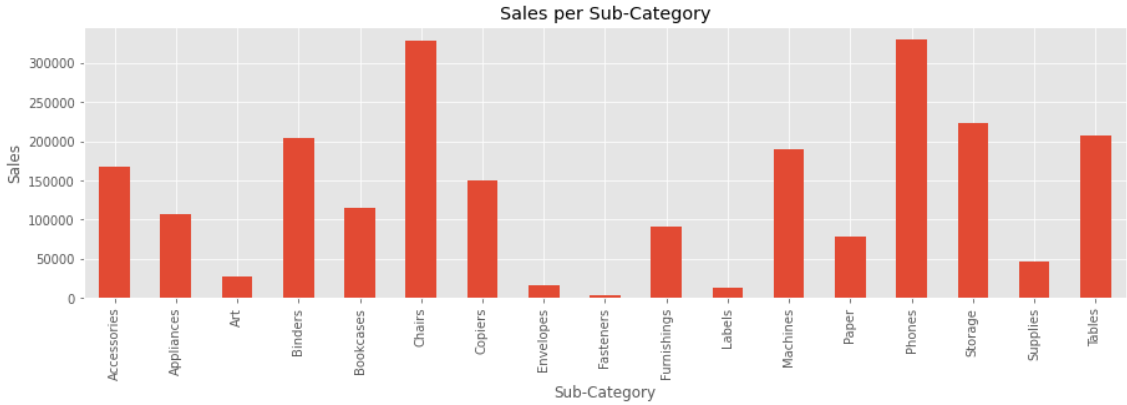
In [39]:

```
Discount_per_state = df.groupby('State')['Discount'].sum()
dx=Discount_per_state.plot(kind='bar',title='Discount per States',figs
dx.set_ylabel('Discount')
plt.show()

```



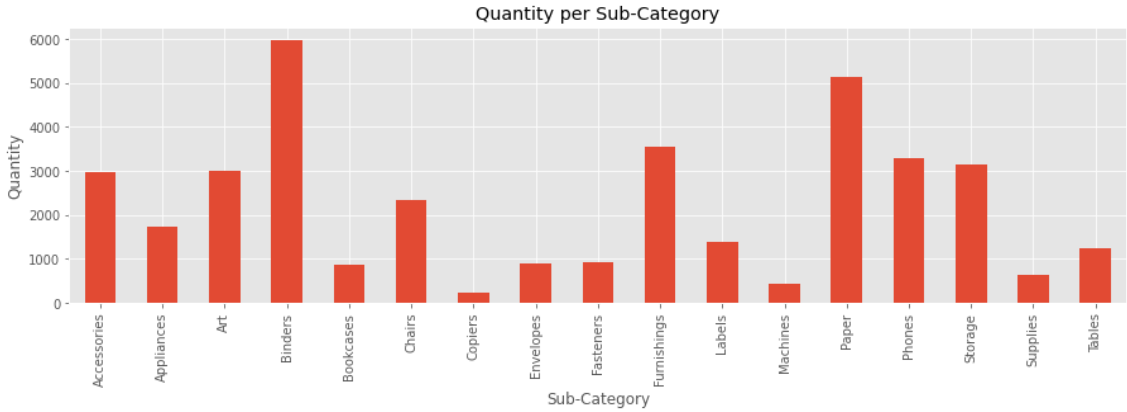
In [40]:

```
Sales_per_Sub_Category = df.groupby('Sub-Category')['Sales'].sum()
cx=Sales_per_Sub_Category.plot(kind='bar',title='Sales per Sub-Categor
cx.set_ylabel('Sales')
plt.show()

```



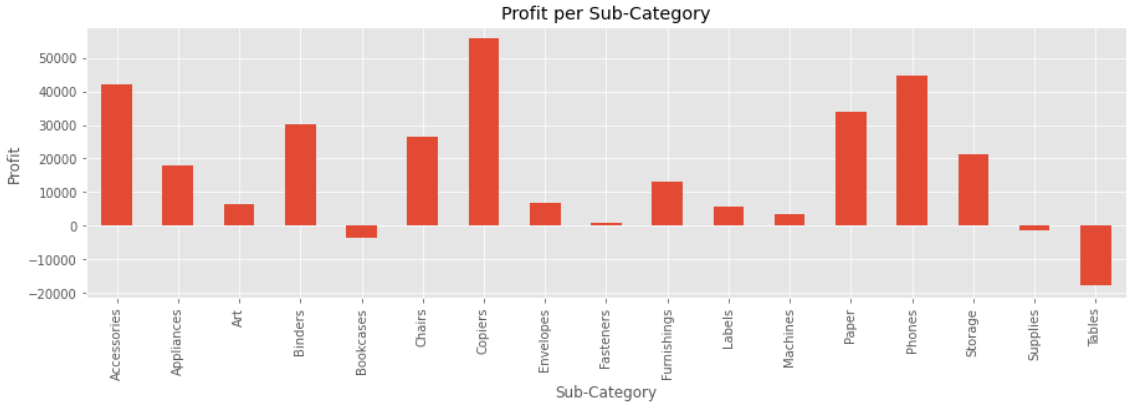
In [41]:

```
Quantity_per_Sub_Category = df.groupby('Sub-Category')['Quantity'].sum
cx=Quantity_per_Sub_Category.plot(kind='bar',title='Quantity per Sub-C
cx.set_ylabel('Quantity')
plt.show()
```



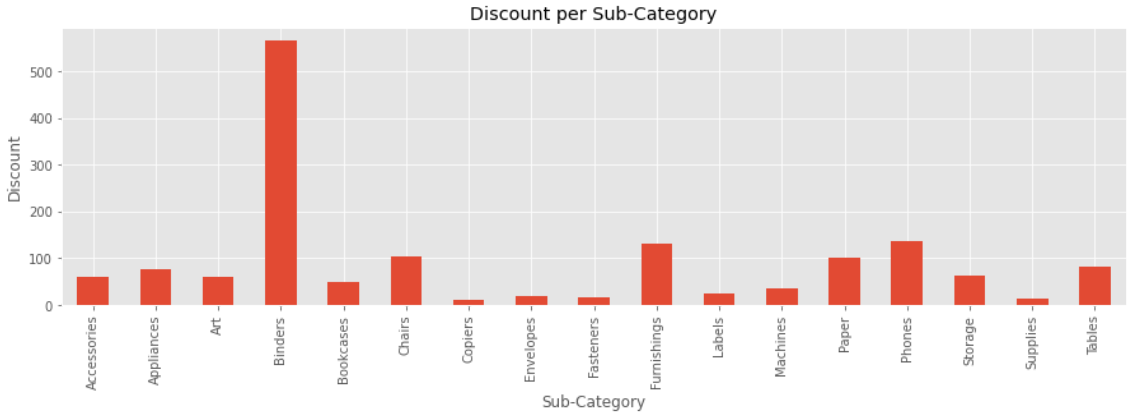
In [42]:

```
Profit_per_Sub_Category = df.groupby('Sub-Category')['Profit'].sum()
cx=Profit_per_Sub_Category.plot(kind='bar',title='Profit per Sub-Categ
cx.set_ylabel('Profit')
plt.show()
```



In [43]:

```
Discount_per_Sub_Category = df.groupby('Sub-Category')['Discount'].sum
cx=Discount_per_Sub_Category.plot(kind='bar',title='Discount per Sub-C
cx.set_ylabel('Discount')
plt.show()
```



In [45]:

```
sns.pairplot(df,vars=['Sales','Quantity','Discount', 'Profit'],hue='Sh
plt.show()
```

In [46]:

```
df_corr = df[['Sales','Quantity','Discount', 'Profit']].corr()
df_corr
```

Out[46]:

	Sales	Quantity	Discount	Profit
Sales	1.000000	0.200722	-0.028311	0.479067
Quantity	0.200722	1.000000	0.008678	0.066211
Discount	-0.028311	0.008678	1.000000	-0.219662
Profit	0.479067	0.066211	-0.219662	1.000000

In [47]:
```
sns.heatmap(df_corr,annot=True)

```

Out[47]: <Axes: >

In [ ]:
```

```