Exploratory Data Analysis

Perform ‘Exploratory Data Analysis’ on dataset ‘SampleSuperstore’

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

SHITAL MORE

In [ ]:

In [169]:
```
#Import Libraries
```

For this project I have used the following Libaries

In [96]:
```
import os  #Standard imports
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Data Information

Importing the csv file in dataframe

In [97]:
```
df=pd.read_csv("C:\\Users\\Shri Krupa\\Downloads\\SampleSuperstore.csv")
```

In [98]:
```
df.head()
```

Out[98]:

	Ship Mode	Segment	Country	City	State	Postal Code	Region	Category	Sub-Category	Sa
0	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Bookcases	261.96
1	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Chairs	731.94
2	Second Class	Corporate	United States	Los Angeles	California	90036	West	Office Supplies	Labels	14.62
3	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Furniture	Tables	957.57
4	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Office Supplies	Storage	22.36

In [99]:
```
df.describe()   #is used to view some basic statistical details like percentile,
                 #mean, std etc. of a data frame or a series of numeric values
```

Out[99]:

	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

Pandas dataframe.info() function is used to get a concise summary of the dataframe

In [100]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 13 columns):
Ship Mode       9994 non-null object
Segment         9994 non-null object
Country         9994 non-null object
City            9994 non-null object
State           9994 non-null object
Postal Code     9994 non-null int64
Region          9994 non-null object
Category        9994 non-null object
Sub-Category    9994 non-null object
Sales           9994 non-null float64
Quantity        9994 non-null int64
Discount        9994 non-null float64
Profit          9994 non-null float64
dtypes: float64(3), int64(2), object(8)
memory usage: 1015.1+ KB
```

In [173]:
```
df['Ship Mode'].unique
```

Out[173]:
```
<bound method Series.unique of 0         Second Class
1         Second Class
2         Second Class
3       Standard Class
4       Standard Class
5       Standard Class
6       Standard Class
7       Standard Class
8       Standard Class
9       Standard Class
10      Standard Class
11      Standard Class
12      Standard Class
13      Standard Class
14      Standard Class
15      Standard Class
16      Standard Class
17        Second Class
18        Second Class
19        Second Class
20        Second Class
21      Standard Class
22      Standard Class
23        Second Class
24      Standard Class
25        Second Class
26        Second Class
27      Standard Class
28      Standard Class
29      Standard Class
             ...
9964      Second Class
9965      Second Class
9966      Second Class
9967    Standard Class
9968    Standard Class
9969    Standard Class
9970    Standard Class
9971    Standard Class
9972    Standard Class
9973    Standard Class
9974    Standard Class
9975    Standard Class
9976    Standard Class
9977    Standard Class
9978    Standard Class
9979    Standard Class
9980      Second Class
9981       First Class
9982    Standard Class
9983    Standard Class
9984    Standard Class
9985    Standard Class
9986    Standard Class
9987    Standard Class
```

```
9988    Standard Class
9989      Second Class
9990    Standard Class
9991    Standard Class
9992    Standard Class
9993      Second Class
Name: Ship Mode, Length: 9977, dtype: object>
```

df. isnull(). sum() will give the column-wise sum of missing values. This returns the counts of non-NA, NA and total number of entries per group.

In [101]:
```
df.isnull().sum()    # 0 False 1 True
```

Out[101]:
```
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 [102]:
```
df.shape
```

Out[102]:
```
(9994, 13)
```

In [103]:
```
df.columns
```

Out[103]:
```
Index(['Ship Mode', 'Segment', 'Country', 'City', 'State', 'Postal Code',
       'Region', 'Category', 'Sub-Category', 'Sales', 'Quantity', 'Discount',
       'Profit'],
      dtype='object')
```

In [104]:
```
df.duplicated().sum()
```

Out[104]:
```
17
```

In [105]:
```
df.nunique()
```

Out[105]:
```
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
```

An important part of Data analysis is analyzing Duplicate Values and removing them. Pandas drop_duplicates() method helps in removing duplicates from the data frame.

In [106]:
```
df.drop_duplicates(keep='first',inplace=True)
df.head()
```

Out[106]:

	Ship Mode	Segment	Country	City	State	Postal Code	Region	Category	Sub-Category	Sa
0	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Bookcases	261.96
1	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Chairs	731.94
2	Second Class	Corporate	United States	Los Angeles	California	90036	West	Office Supplies	Labels	14.62
3	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Furniture	Tables	957.57
4	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Office Supplies	Storage	22.36

In [107]:
```
df.duplicated().sum()  #df.duplicated() 0 False 1 True types=bool
```

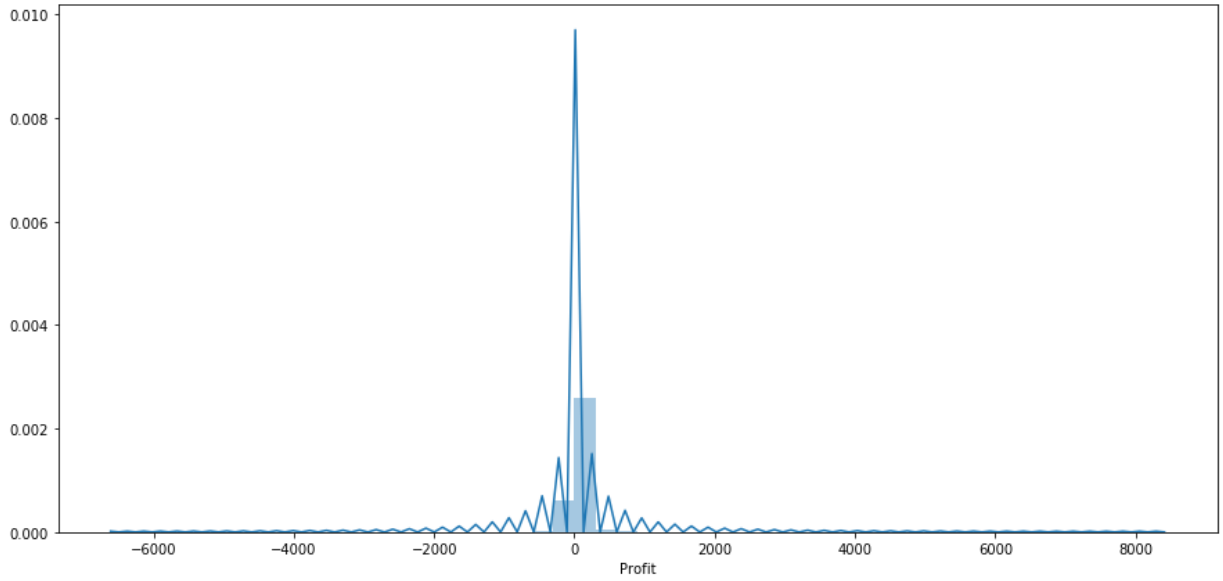
Out[107]:
```
0
```

Before we dive into the details of creating visualizations with Matplotlib,

Since it is a financial dataset lets have a look at profit and loss statements

In [108]:
```
fig,axes=plt.subplots(figsize=(15,7))
sns.distplot(df.Profit)
```

Out[108]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2119ab79940>
```



In [109]:
```
df1=df.loc[:,['Profit','Sales']]
```

In [110]:
```
df1.head()
```

Out[110]:

	Profit	Sales
0	41.9136	261.9600
1	219.5820	731.9400
2	6.8714	14.6200
3	-383.0310	957.5775
4	2.5164	22.3680

In [111]:
```
df2=df1.sort_values(['Profit'],ascending=False)
```

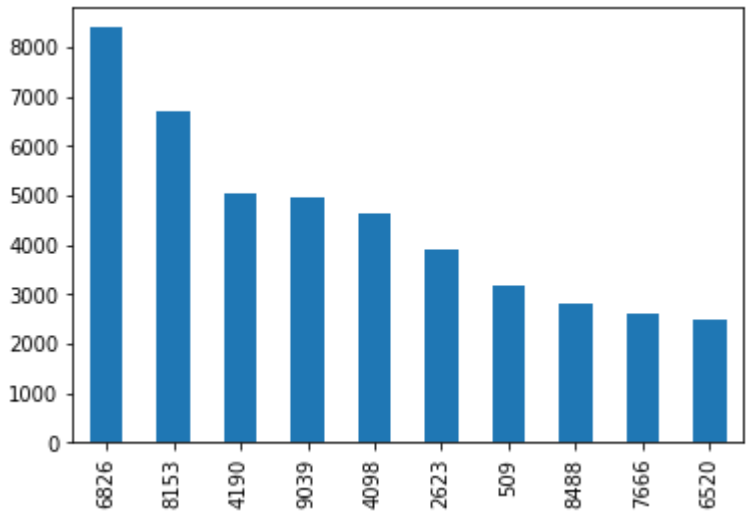
In [112]:
```
df2.head(10)
```

Out[112]:

	Profit	Sales
6826	8399.9760	17499.950
8153	6719.9808	13999.960
4190	5039.9856	10499.970
9039	4946.3700	9892.740
4098	4630.4755	9449.950
2623	3919.9888	11199.968
509	3177.4750	6354.950
8488	2799.9840	8749.950
7666	2591.9568	5399.910
6520	2504.2216	5443.960

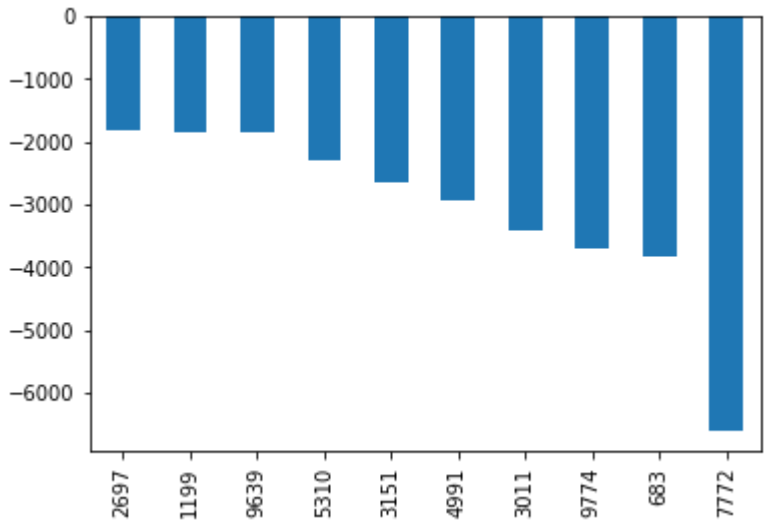
In [113]:
```
df2['Profit'].head(10).plot(kind='bar')
```

Out[113]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2119cc37fd0>
```



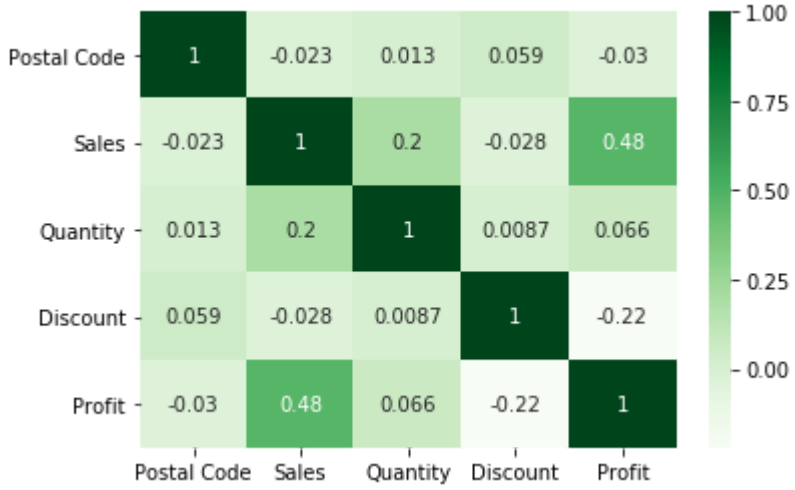
In [114]:
```
df2['Profit'].tail(10).plot(kind='bar')
```

Out[114]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2119a68fbe0>
```



In [116]:
```
corr=df.corr()
sns.heatmap(corr,annot=True,cmap='Greens')
```

Out[116]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2119a71a710>
```

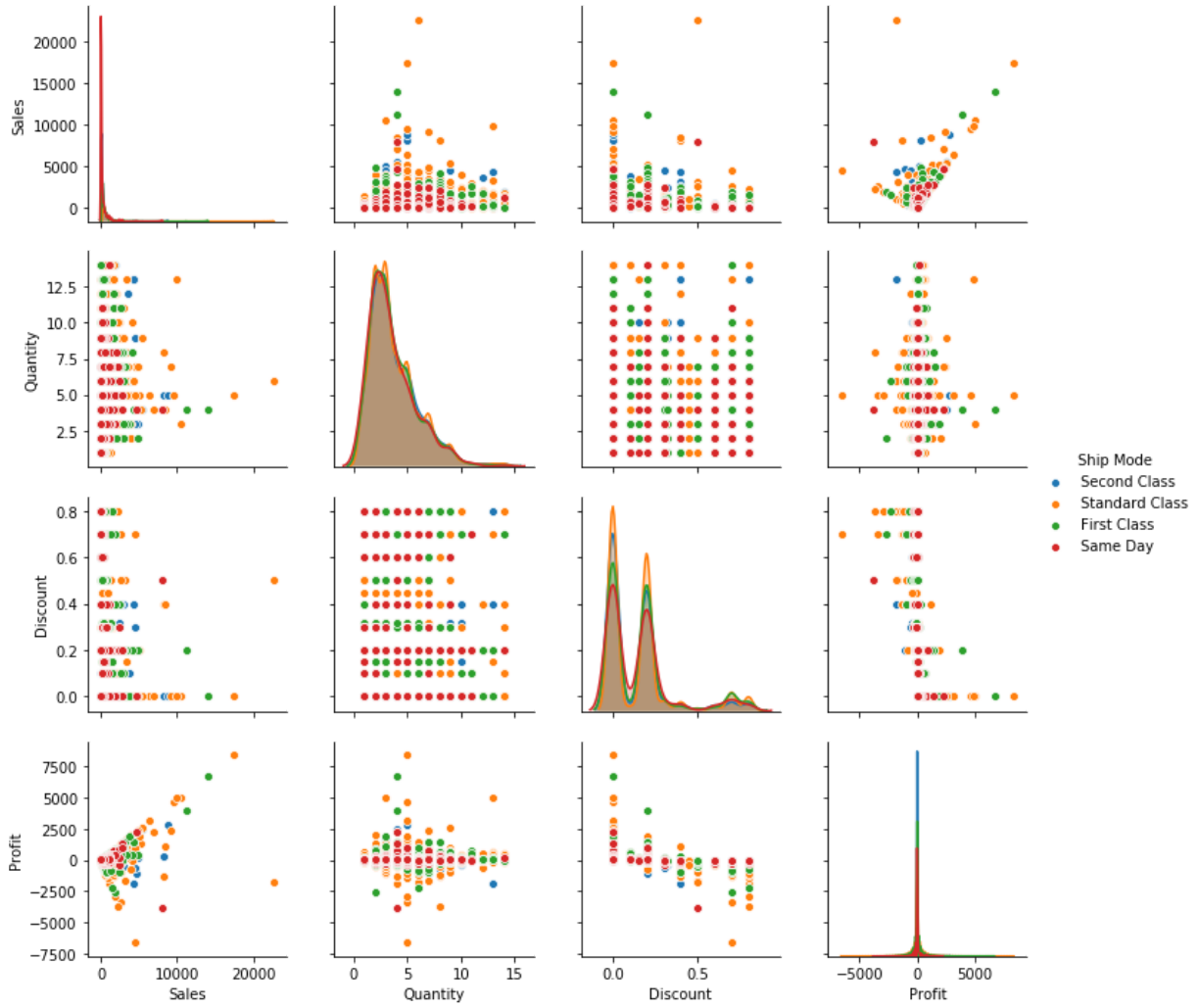


In [117]:
```
df['Postal Code']=df['Postal Code'].astype('object')
```

In [118]:
```
df=df.drop(['Postal Code'],axis=1)
```

In [119]:
```
sns.pairplot(df,hue='Ship Mode')
```

Out[119]:
```
<seaborn.axisgrid.PairGrid at 0x2119acaec18>
```



In [120]:
```
df['Ship Mode'].value_counts()
```

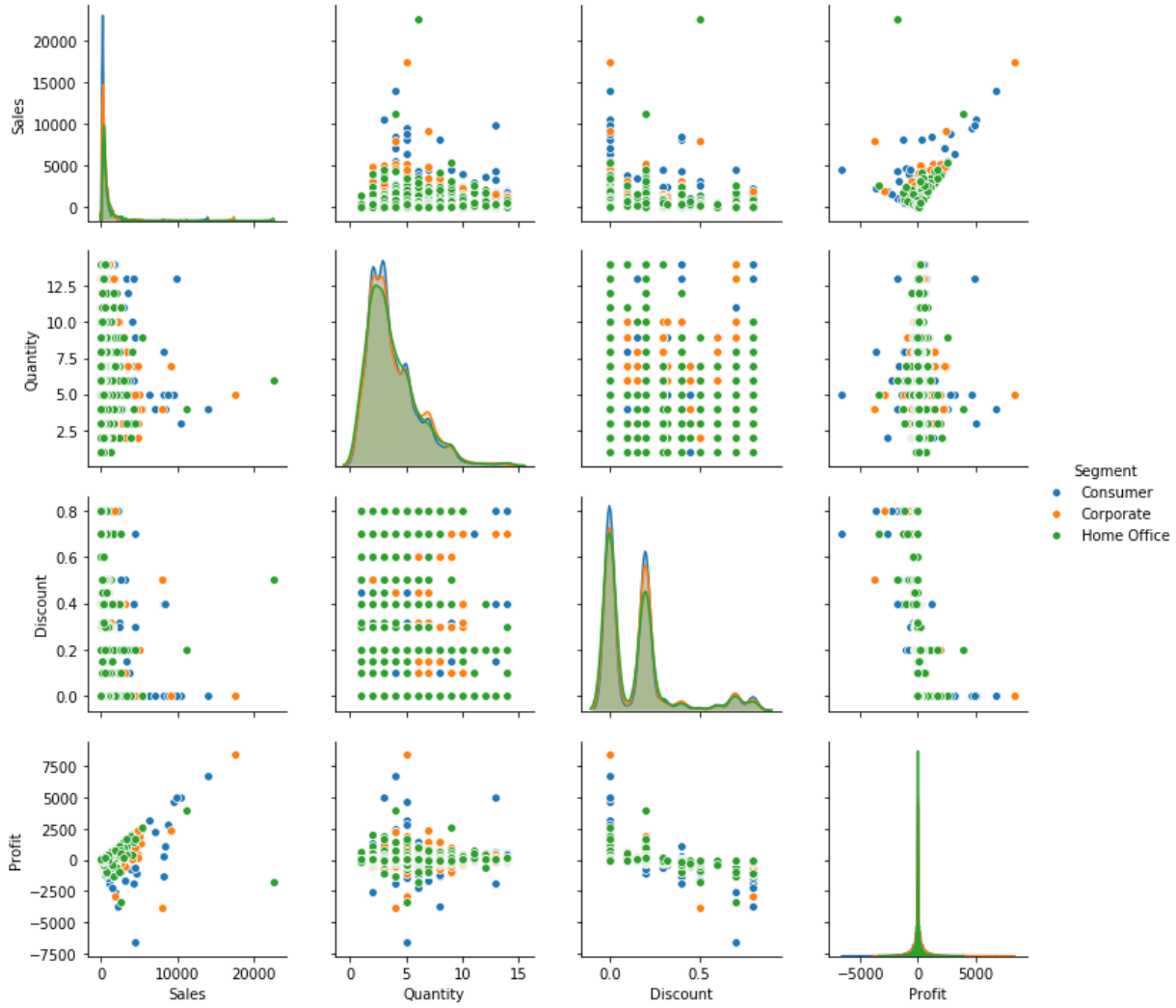
Out[120]:
```
Standard Class    5955
Second Class      1943
First Class       1537
Same Day           542
Name: Ship Mode, dtype: int64
```

In [121]:
```
df['Segment'].value_counts()
```

Out[121]:
```
Consumer       5183
Corporate      3015
Home Office    1779
Name: Segment, dtype: int64
```

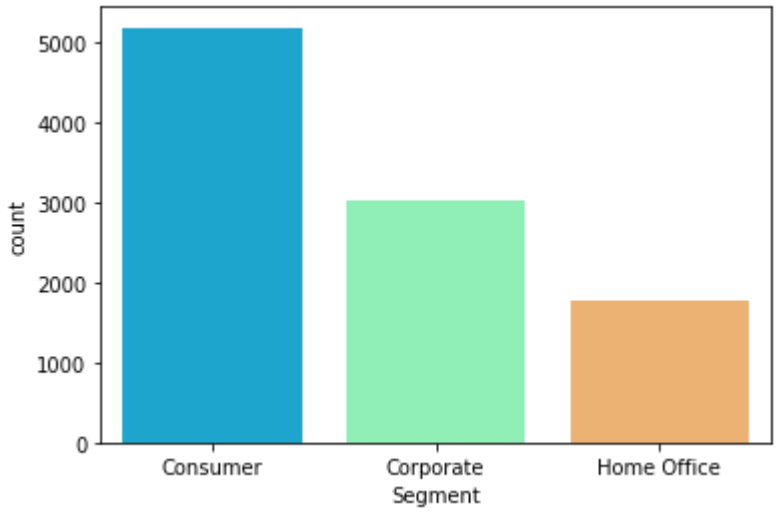
In [123]:
```
sns.pairplot(df,hue='Segment')
```

Out[123]: <seaborn.axisgrid.PairGrid at 0x2119b1f2c18>



In [124]:
```
sns.countplot(x='Segment',data=df,palette='rainbow')
```

Out[124]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2119e638780>
```

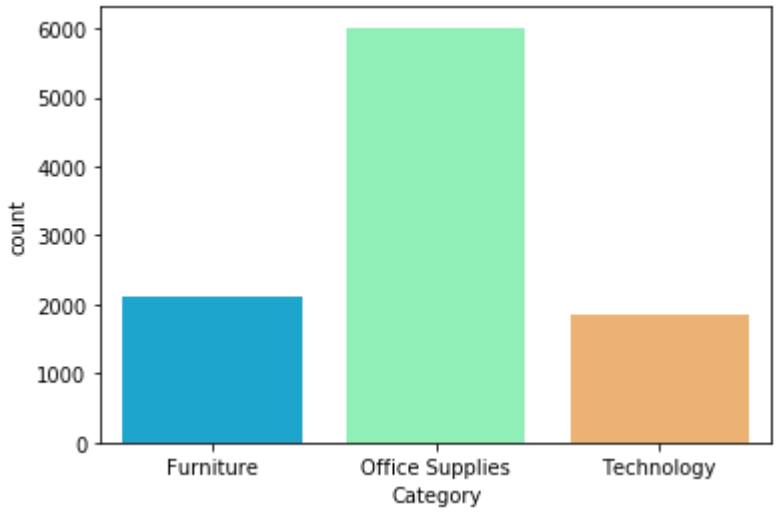


In [125]:
```
df['Category'].value_counts()
```

Out[125]:
```
Office Supplies    6012
Furniture          2118
Technology         1847
Name: Category, dtype: int64
```

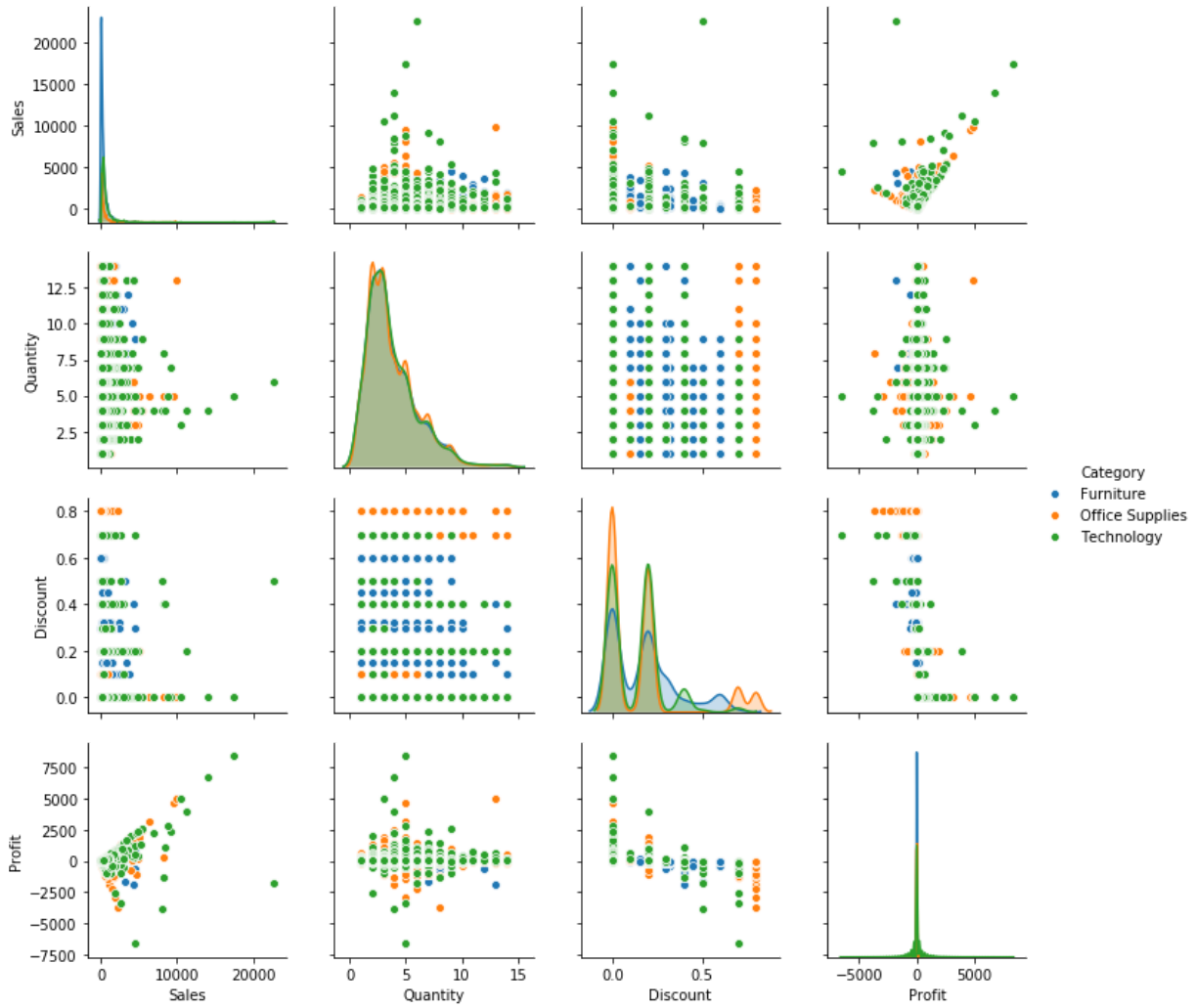
In [126]:
```
sns.countplot(x='Category',data=df,palette='rainbow')
```

Out[126]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2119e8c05c0>
```

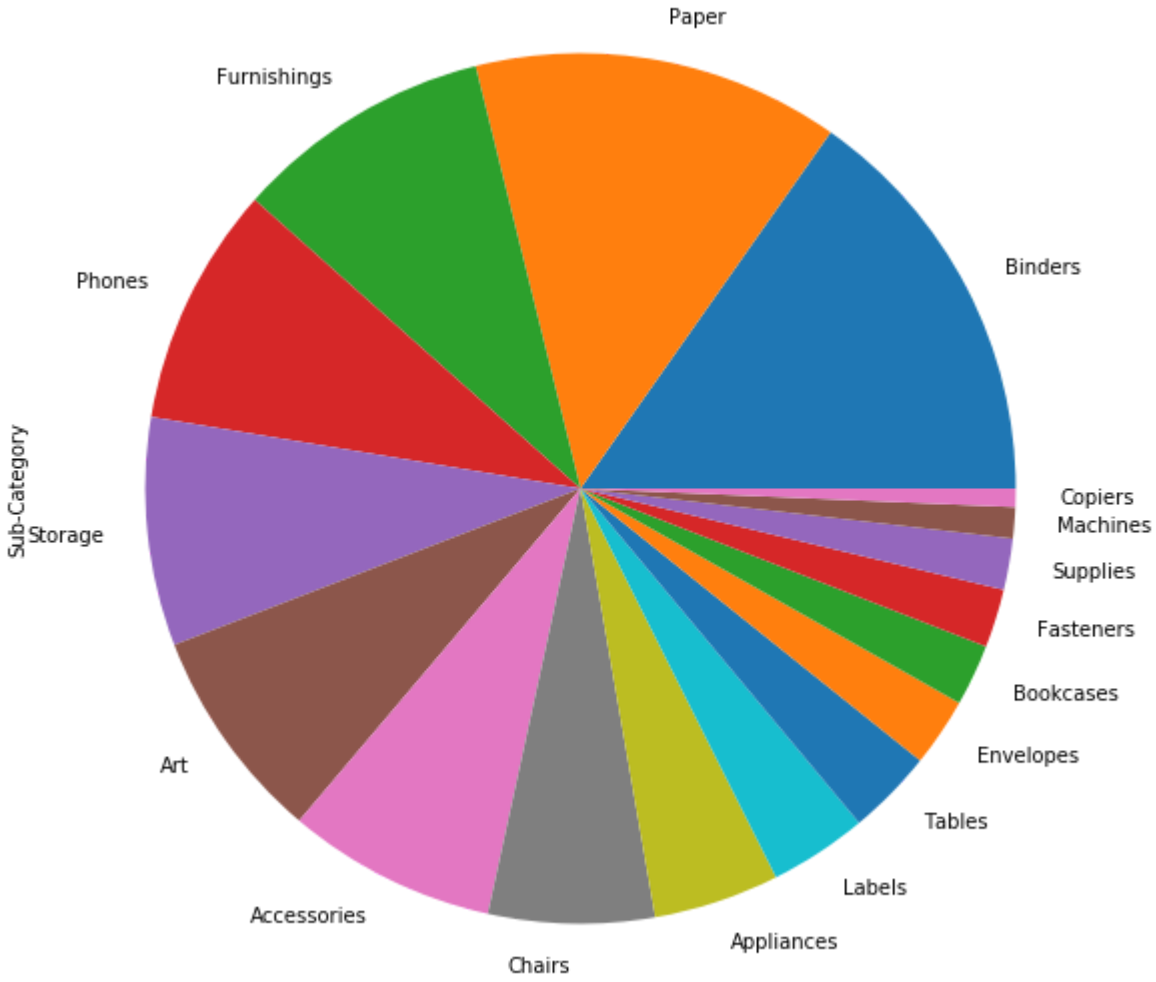


In [127]:
```
sns.pairplot(df,hue='Category')
```

Out[127]: <seaborn.axisgrid.PairGrid at 0x2119ebae780>



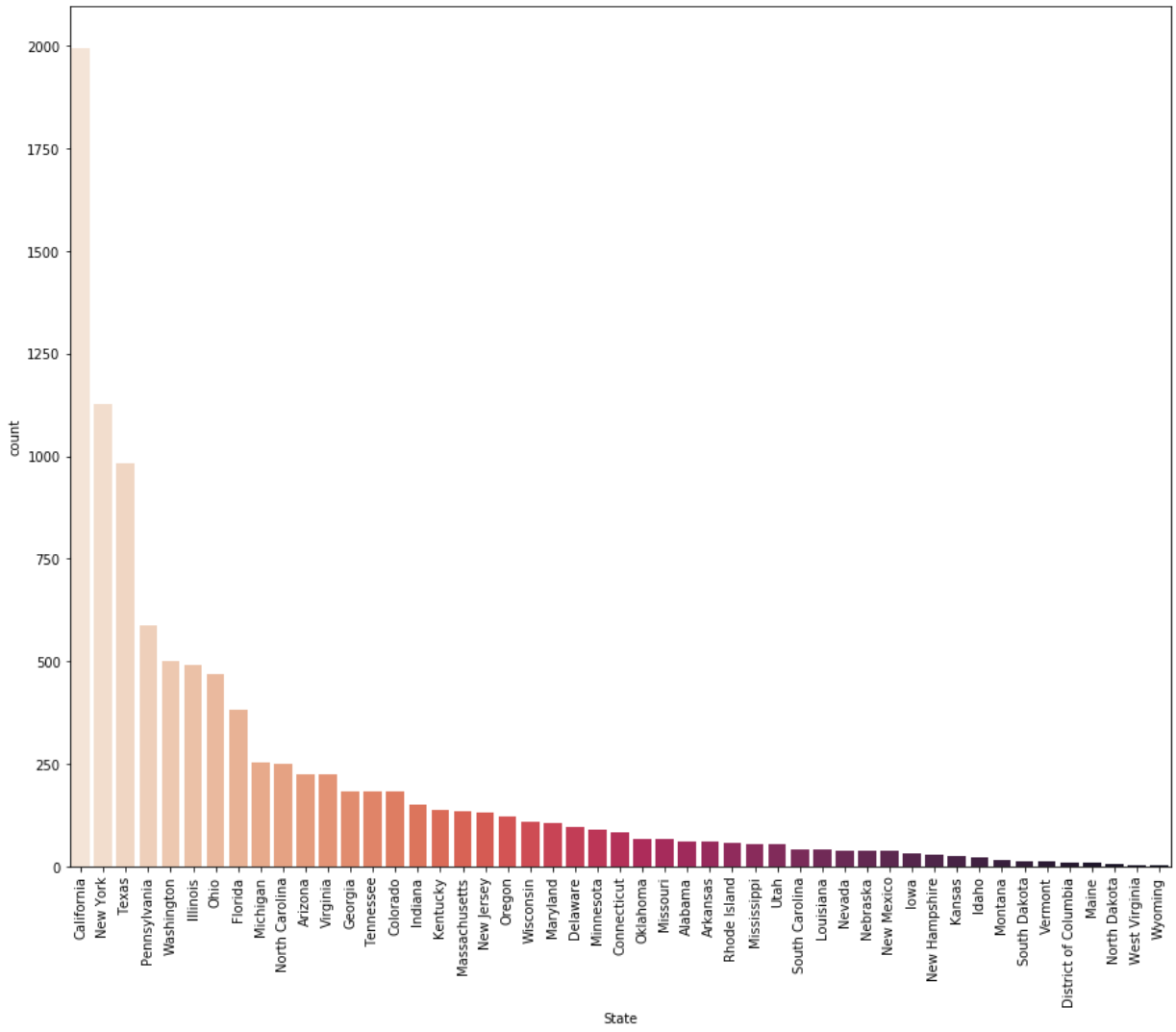
In [129]:
```
plt.figure(figsize=(12,10))
df['Sub-Category'].value_counts().plot.pie()
plt.show()
```



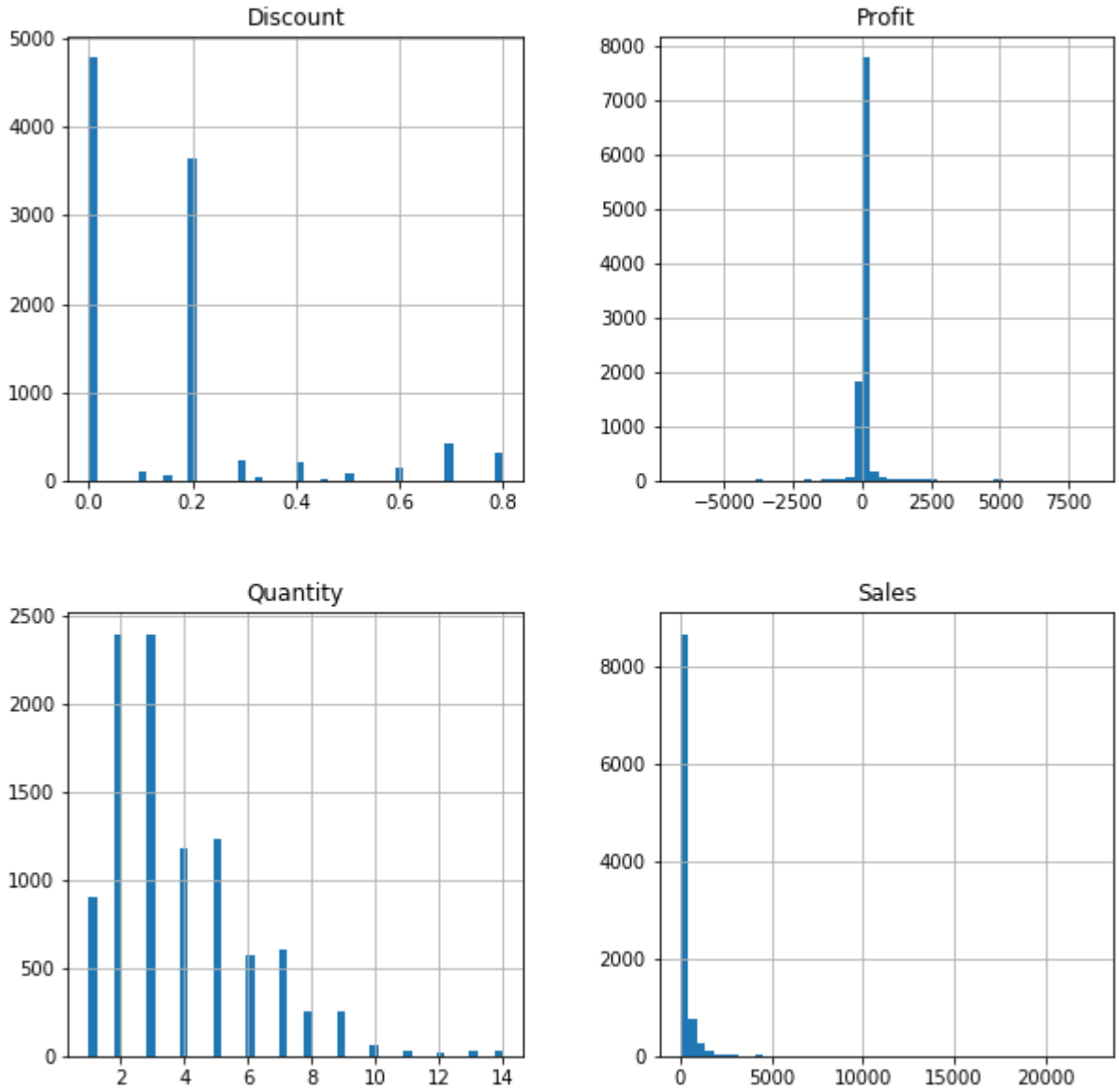
In [130]:
```
df['State'].value_counts()
```

Out[130]:
```
California              1996
New York                1127
Texas                    983
Pennsylvania             586
Washington               502
Illinois                 491
Ohio                     468
Florida                  383
Michigan                 254
North Carolina           249
Arizona                  224
Virginia                 224
Georgia                  184
Tennessee                183
Colorado                 182
Indiana                  149
Kentucky                 139
Massachusetts            135
New Jersey               130
Oregon                   123
Wisconsin                110
Maryland                 105
Delaware                  96
Minnesota                 89
Connecticut               82
Oklahoma                  66
Missouri                  66
Alabama                   61
Arkansas                  60
Rhode Island              56
Mississippi               53
Utah                      53
South Carolina            42
Louisiana                 42
Nevada                    39
Nebraska                  38
New Mexico                37
Iowa                      30
New Hampshire             27
Kansas                    24
Idaho                     21
Montana                   15
South Dakota              12
Vermont                   11
District of Columbia      10
Maine                      8
North Dakota               7
West Virginia              4
Wyoming                    1
Name: State, dtype: int64
```

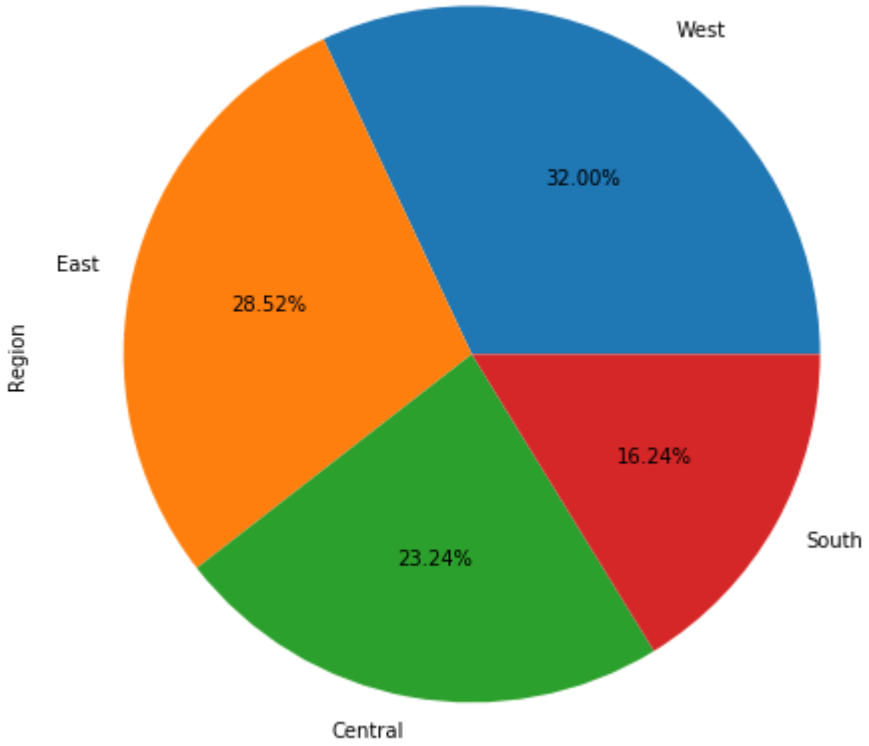
In [132]:
```
plt.figure(figsize=(15,12))
sns.countplot(x='State',data=df,palette='rocket_r',order=df['State'].value_counts
plt.xticks(rotation=90)
plt.show()
```



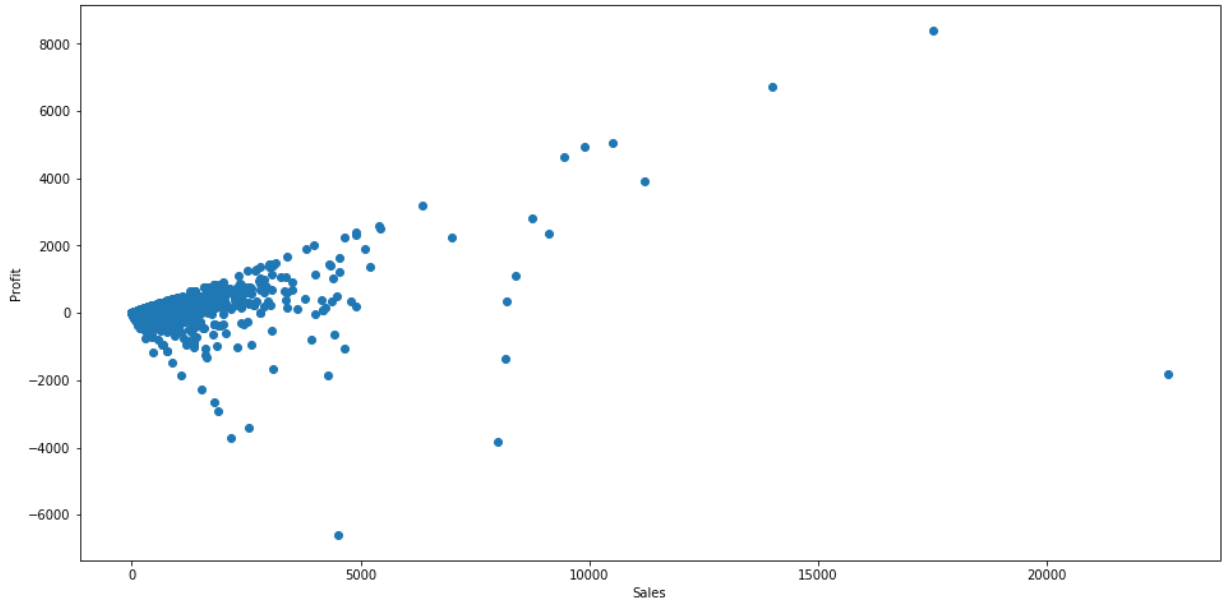
In [133]:
```
df.hist(figsize=(10,10),bins=50)
plt.show()
```



In [167]:
```
plt.figure(figsize=(10,8))
df['Region'].value_counts().plot(kind='pie',autopct='%1.2f%%')
plt.show()
```

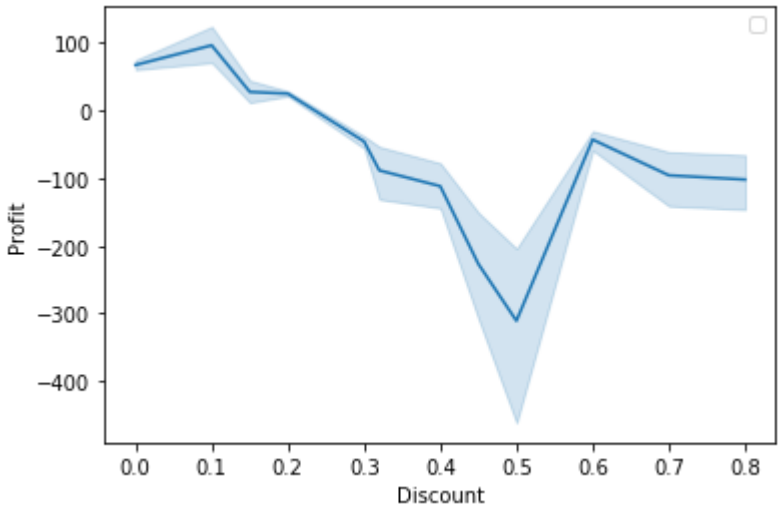


In [136]:
```
from matplotlib import style
fig,ax=plt.subplots(figsize=(16,8))
ax.scatter(df['Sales'],df['Profit'])
ax.set_xlabel('Sales')
ax.set_ylabel('Profit')
plt.show()
```

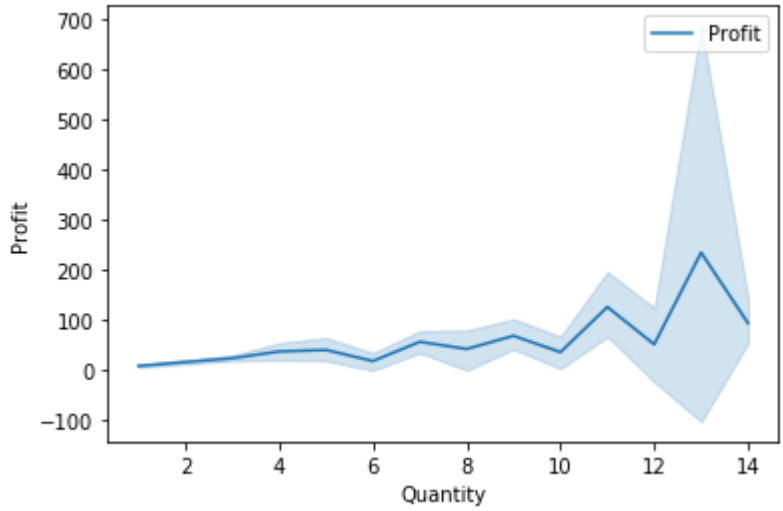


In [137]:
```
sns.lineplot(x='Discount',y='Profit' ,data=df)
plt.legend()
plt.show()
```

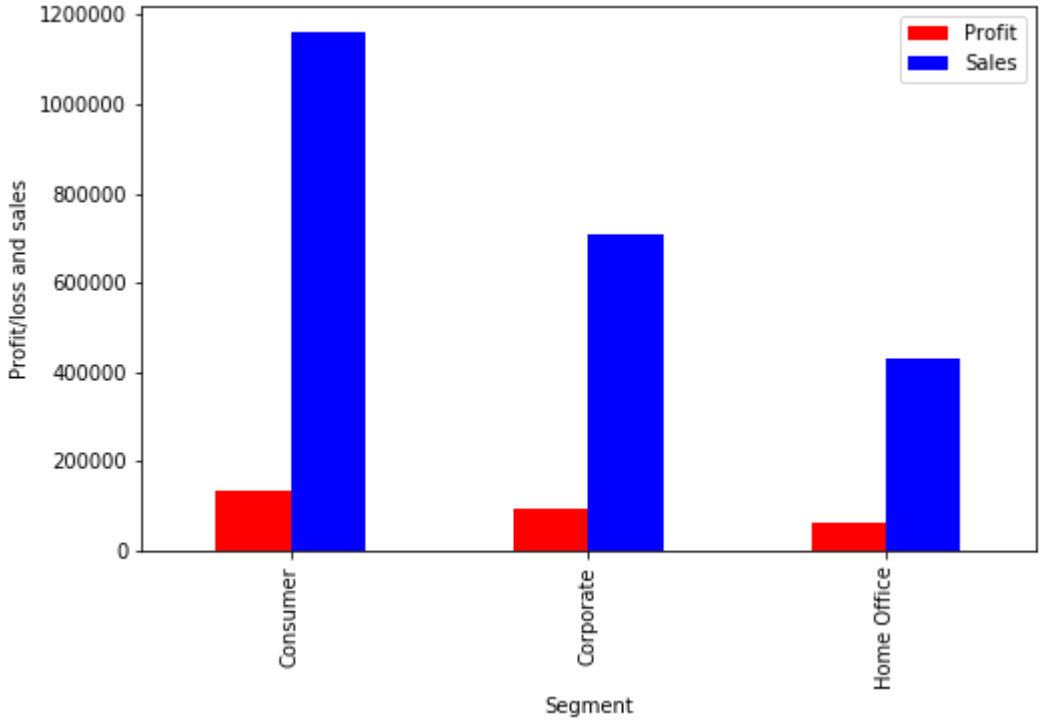
```
No handles with labels found to put in legend.
```



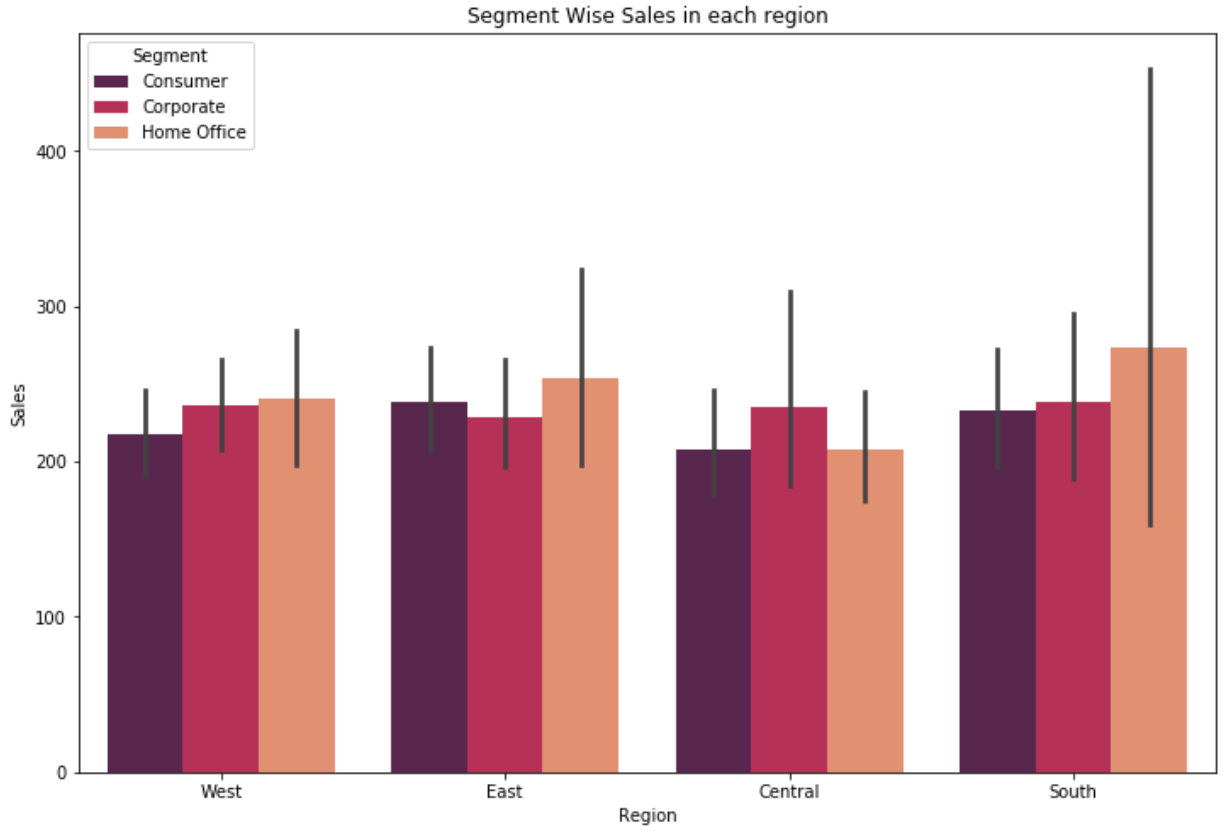
In [138]:
```
sns.lineplot(x='Quantity',y='Profit',label='Profit',data=df)
plt.show()
```



In [139]:
```
df.groupby('Segment')[['Profit','Sales']].sum().plot.bar(color=['red','blue'],fi
plt.ylabel('Profit/loss and sales')
plt.show()
```



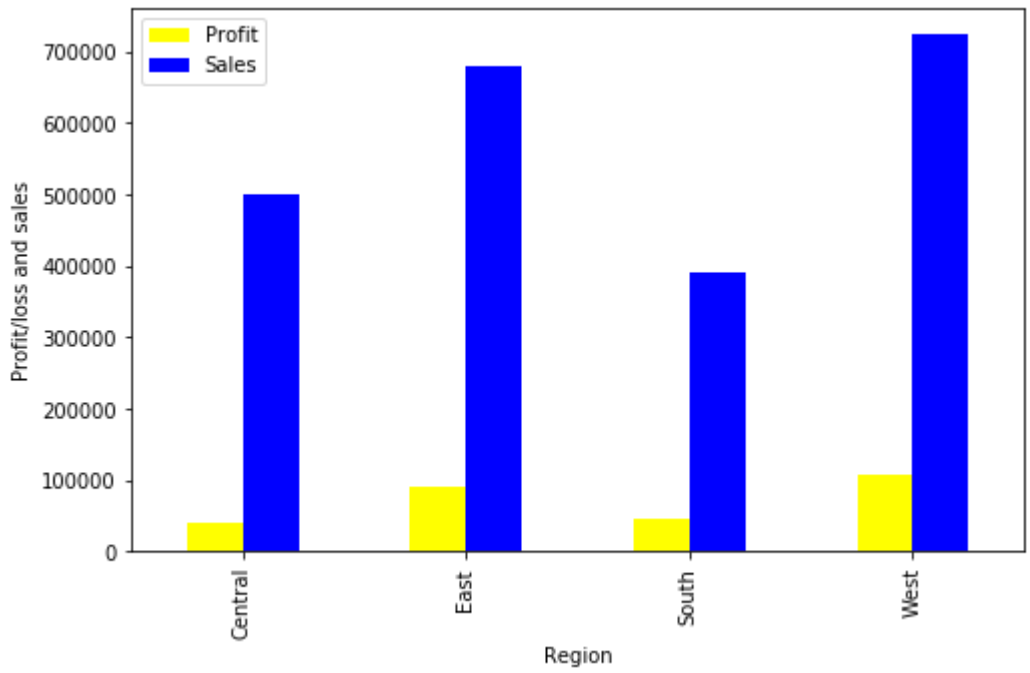
In [155]:
```
plt.figure(figsize=(12,8))
plt.title('Segment Wise Sales in each region')
sns.barplot(x='Region', y='Sales' , data=df ,hue='Segment',order=df['Region'].va
plt.xlabel('Region')
plt.show()
```



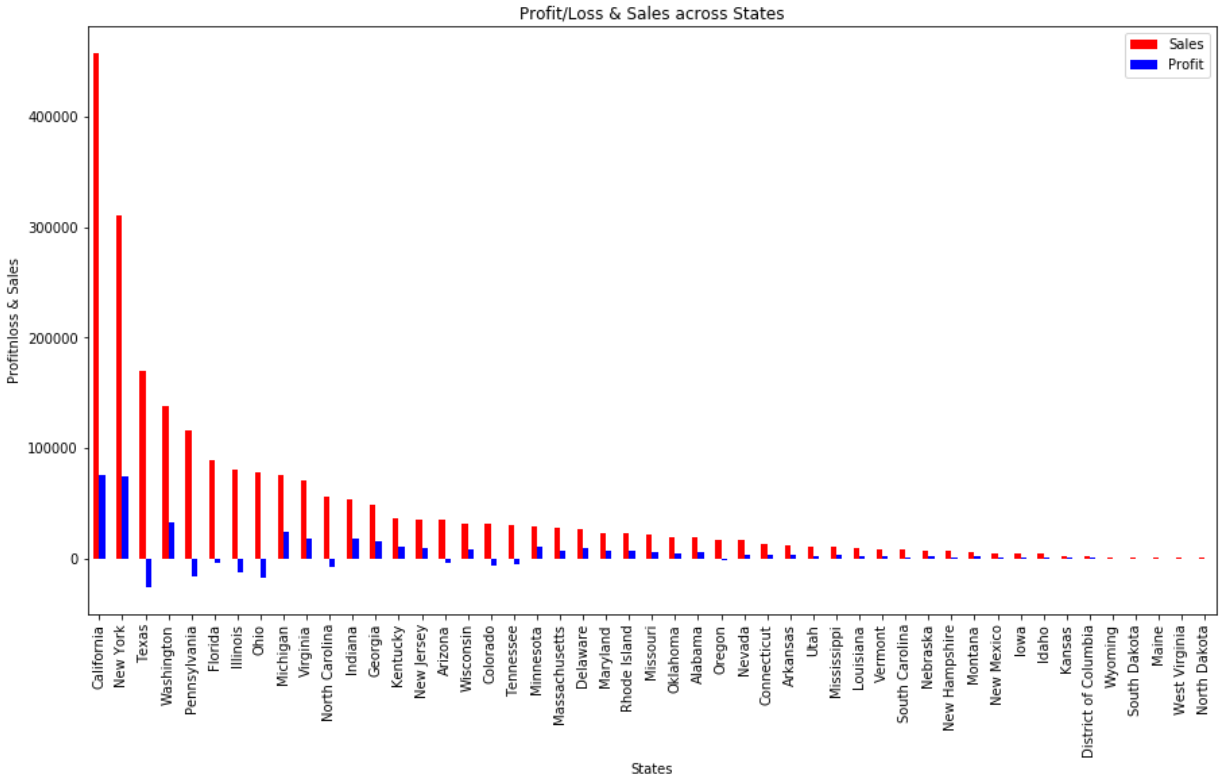
In [ ]:
```

```

In [143]:
```
df.groupby('Region')[['Profit','Sales']].sum().plot.bar(color=['yellow','blue'],
plt.ylabel('Profit/loss and sales')
plt.show()
```



In [164]:
```
ps=df.groupby('State')[['Sales','Profit']].sum().sort_values(by='Sales',ascendin
ps[:].plot.bar(color=['red','blue'],figsize=(15,8))
plt.title('Profit/Loss & Sales across States')
plt.xlabel('States')
plt.ylabel('Profitnloss & Sales')
plt.show()
```



In [151]:
```
top_states=df['State'].value_counts()
```

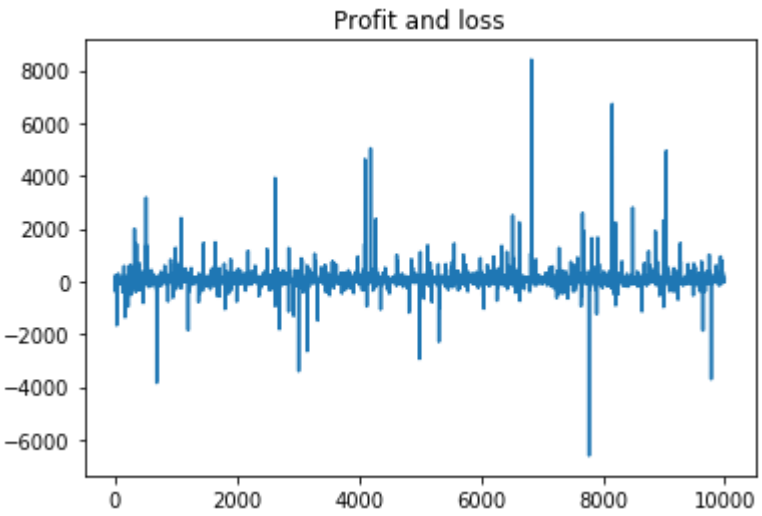
In [154]:
```
top_states.head(10)
```

Out[154]:
```
California        1996
New York          1127
Texas              983
Pennsylvania       586
Washington         502
Illinois           491
Ohio               468
Florida            383
Michigan           254
North Carolina     249
Name: State, dtype: int64
```

In [168]:
```
# Lets see how money is divided in these ship mode categories
```

In [159]:
```
Profitandloss=df['Profit']
plt.plot(Profitandloss)
plt.title("Profit and loss")
```

Out[159]:
```
Text(0.5, 1.0, 'Profit and loss')
```



In [160]:
```
Profitandloss.max()
```

Out[160]:
```
8399.976
```

In [161]:
```
Profitandloss.min()
```

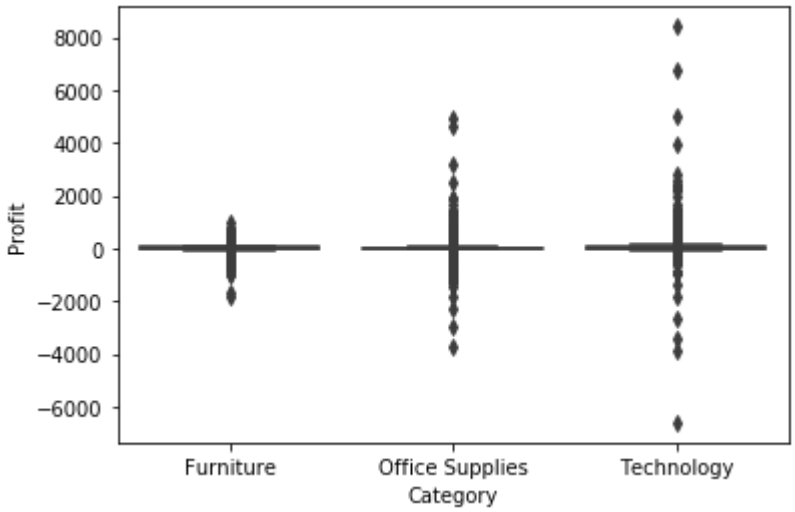
Out[161]:
```
-6599.978
```

In [ ]:
```

```

In [162]:
```
sns.boxplot(df['Category'],Profitandloss) #here we can see that Technology has t
```

Out[162]: <matplotlib.axes._subplots.AxesSubplot at 0x211a1d545c0>

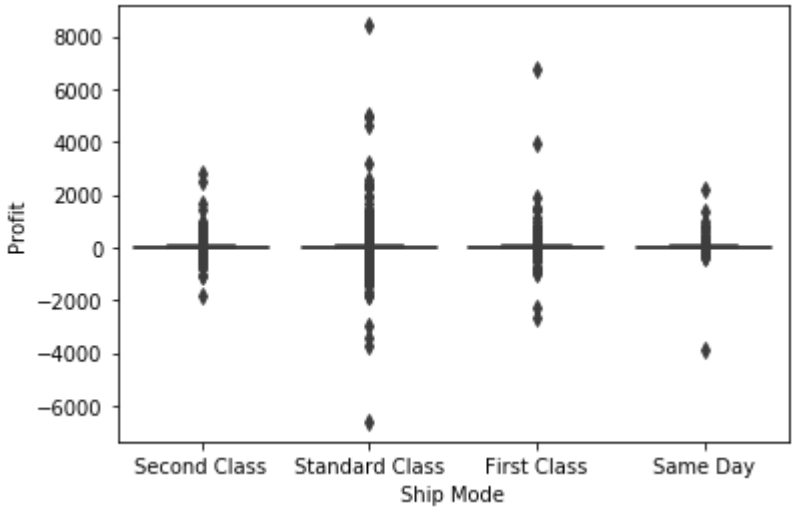


In [ ]:
```

```

In [163]:
```
sns.boxplot(df['Ship Mode'],Profitandloss) #From Boxplot we can see that standar
                                           #it has both max profit and maxloss
```

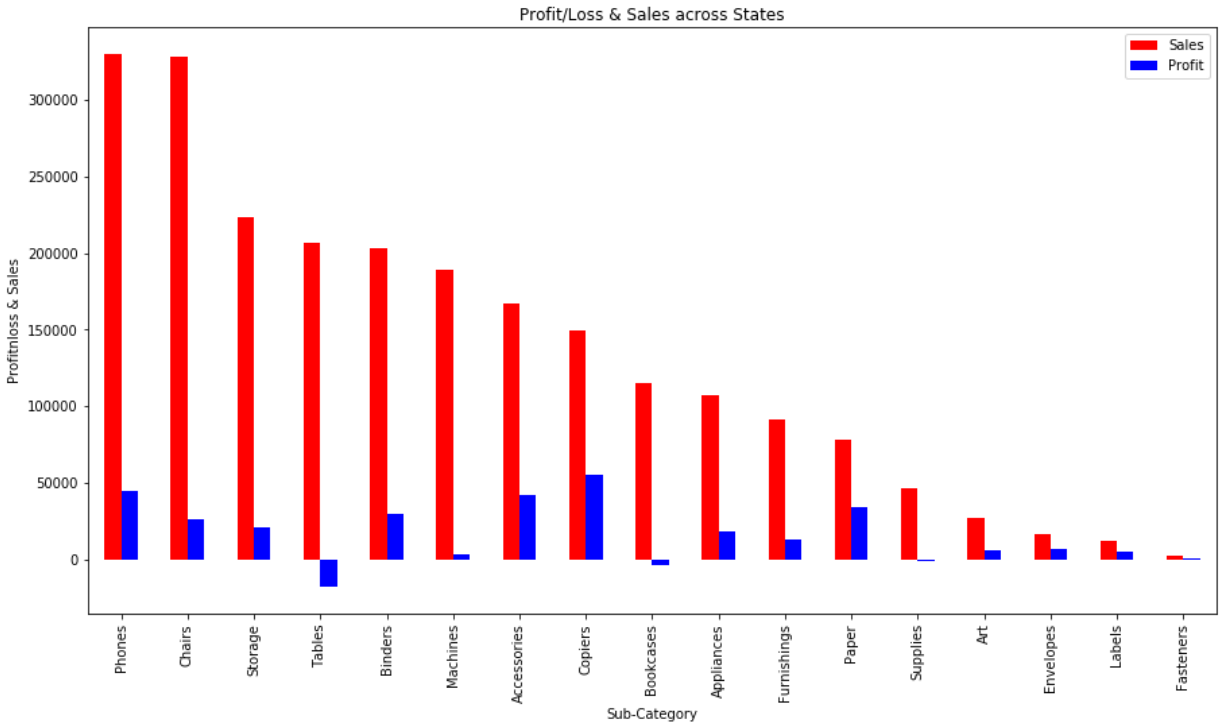
Out[163]: <matplotlib.axes._subplots.AxesSubplot at 0x2119f97d908>



In [ ]:
```

```

In [165]:
```
ps=df.groupby('Sub-Category')[['Sales','Profit']].sum().sort_values(by='Sales',as
ps[:].plot.bar(color=['red','blue'],figsize=(15,8))
plt.title('Profit/Loss & Sales across States')
plt.xlabel('Sub-Category')
plt.ylabel('Profitnloss & Sales')
plt.show()
```



conculsion

we need to invest more in Technology Category.most people se standard ship mode of delivery .All three modes of First Class,Second Class,standard a major role in our superstore ship mode.Each offers benefits that the other mode of transport .It is up to you to make a well-informed decision of choosing the right mode of shipping that will be beneficial for your business.