GRIP @ The Spark Foundation

Author : Cheekireddy Dhamini

Sample Super Store - Retail

Importing the packages

In [1]:
```
# Importing the packages
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

Reading the Data set

In [2]:
```
# Reading the data set
data = pd.read_csv("SampleSuperstore.csv")
```

In [3]:
```
# Reading the first 5 attributes
data.head()
```

Out[3]:

	Ship Mode	Segment	Country	City	State	Postal Code	Region	Category	Sub-Category	Sales	Quantity	Discount	Profit
0	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Bookcases	261.9600	2	0.00	41.9136
1	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Chairs	731.9400	3	0.00	219.5820
2	Second Class	Corporate	United States	Los Angeles	California	90036	West	Office Supplies	Labels	14.6200	2	0.00	6.8714
3	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Furniture	Tables	957.5775	5	0.45	-383.0310
4	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Office Supplies	Storage	22.3680	2	0.20	2.5164

Exploratary Data Analysis

In [4]:
```
# Getting the total number of attributes in the data
data.columns
```

Out[4]:
```
Index(['Ship Mode', 'Segment', 'Country', 'City', 'State', 'Postal Code',
       'Region', 'Category', 'Sub-Category', 'Sales', 'Quantity', 'Discount',
       'Profit'],
      dtype='object')
```

In [5]:
```
# Getting the information about the data set
data.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 [6]:
```
# Getting the statistical data of the data set
data.describe()
```

Out[6]:

	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 [7]:
```
# Checking the number of null values in the given data set
data.isnull().sum()
```

Out[7]:
```
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 [8]:
```
# To find the duplicate values in the data
data.duplicated().sum()
```

Out[8]:
```
17
```

In [9]:
```
# Getting the duplicated values
dup = data.duplicated()
data[dup]
```

Out[9]:

	Ship Mode	Segment	Country	City	State	Postal Code	Region	Category	Sub-Category	Sales	Quantity	Discount	
950	Standard Class	Home Office	United States	Philadelphia	Pennsylvania	19120	East	Office Supplies	Paper	15.552	3	0.2	5
3406	Standard Class	Home Office	United States	Columbus	Ohio	43229	East	Furniture	Chairs	281.372	2	0.3	-12
3670	Standard Class	Consumer	United States	Salem	Oregon	97301	West	Office Supplies	Paper	10.368	2	0.2	3
4117	Standard Class	Consumer	United States	Los Angeles	California	90036	West	Office Supplies	Paper	19.440	3	0.0	9
4553	Standard Class	Consumer	United States	San Francisco	California	94122	West	Office Supplies	Paper	12.840	3	0.0	5
5905	Same Day	Home Office	United States	San Francisco	California	94122	West	Office Supplies	Labels	41.400	4	0.0	19
6146	Standard Class	Corporate	United States	San Francisco	California	94122	West	Office Supplies	Art	11.760	4	0.0	3
6334	Standard Class	Consumer	United States	New York City	New York	10011	East	Office Supplies	Paper	49.120	4	0.0	23
6357	Standard Class	Corporate	United States	Seattle	Washington	98103	West	Office Supplies	Paper	25.920	4	0.0	12
7608	Standard Class	Consumer	United States	San Francisco	California	94122	West	Office Supplies	Paper	25.920	4	0.0	12
7735	Standard Class	Corporate	United States	Seattle	Washington	98105	West	Office Supplies	Paper	19.440	3	0.0	9
7759	Standard Class	Corporate	United States	Houston	Texas	77041	Central	Office Supplies	Paper	15.552	3	0.2	5
8032	First Class	Consumer	United States	Houston	Texas	77041	Central	Office Supplies	Paper	47.952	3	0.2	16
8095	Second Class	Consumer	United States	Seattle	Washington	98115	West	Office Supplies	Paper	12.960	2	0.0	6
9262	Standard Class	Consumer	United States	Detroit	Michigan	48227	Central	Furniture	Chairs	389.970	3	0.0	35

	Ship Mode	Segment	Country	City	State	Postal Code	Region	Category	Sub-Category	Sales	Quantity	Discount	
9363	Standard Class	Home Office	United States	Seattle	Washington	98105	West	Furniture	Furnishings	22.140	3	0.0	
9477	Second Class	Corporate	United States	Chicago	Illinois	60653	Central	Office Supplies	Binders	3.564	3	0.8	

In [10]:
```
# Dropping the duplicated columns from the dataset
data.drop_duplicates(inplace=True)
```

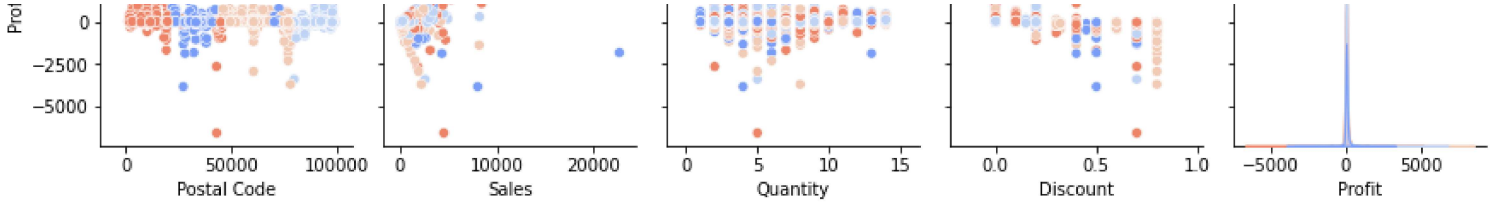
In [11]:
```
# Checking whether duplicates are removed or not
data.duplicated().sum()
```

Out[11]: 0

Data Visualization

In [12]:
```
# Pair plot
sns.pairplot(data,hue='Region',palette='coolwarm')
```

Out[12]:
```
<seaborn.axisgrid.PairGrid at 0x23d059269d0>
```


Prof
0
−2500
−5000
0
50000
100000
Postal Code
0
10000
20000
Sales
0
5
10
15
Quantity
0.0
0.5
1.0
Discount
−5000
0
5000
Profit

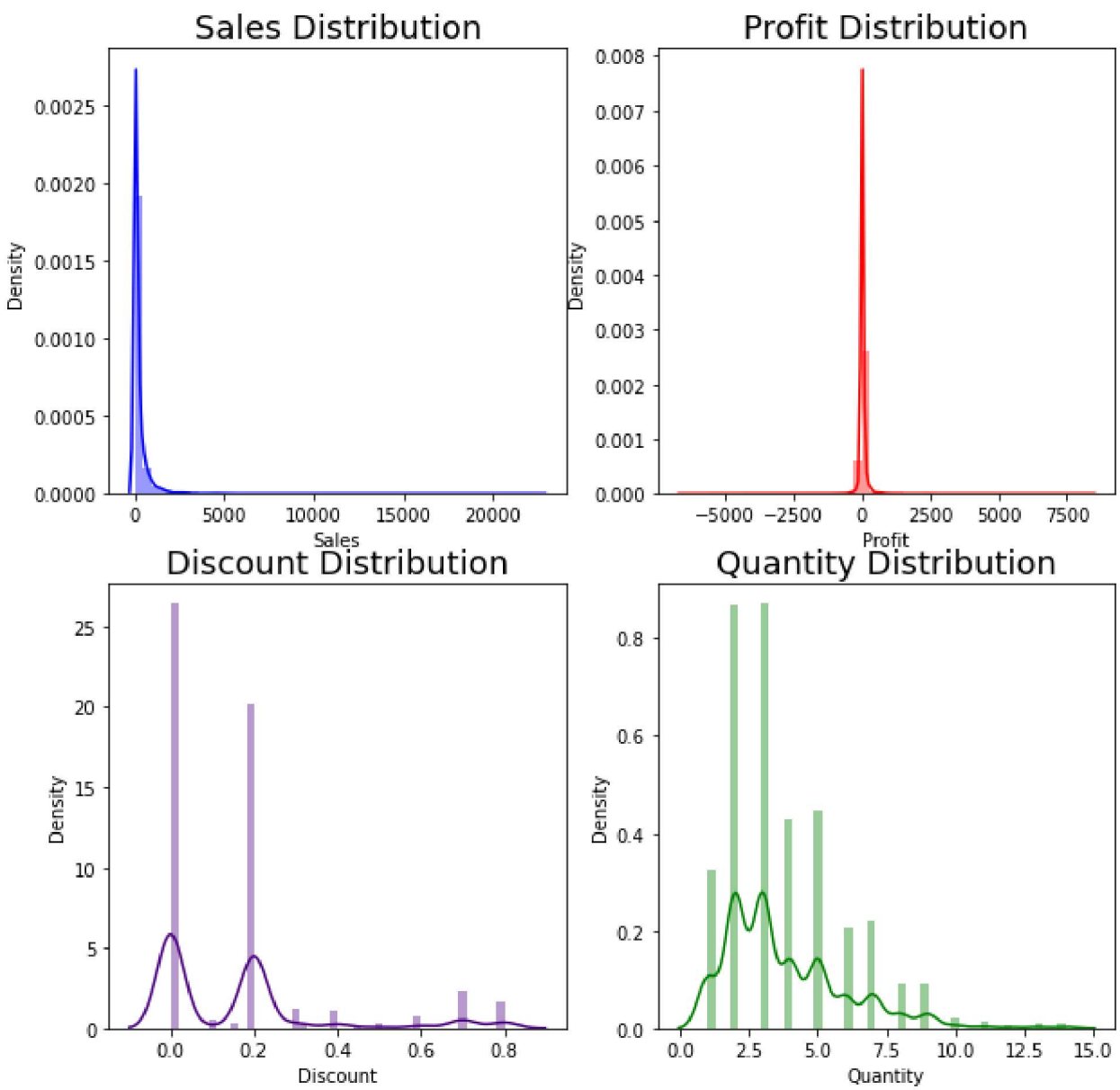
In [13]:

```
# Comparing with Distribution Plots
fig, axs = plt.subplots(2,2,figsize=(10,10))
sns.distplot(data['Sales'],ax=axs[0,0],color='blue')
sns.distplot(data['Profit'],ax=axs[0,1],color='red')
sns.distplot(data['Discount'],ax=axs[1,0],color='indigo')
sns.distplot(data['Quantity'],ax=axs[1,1],color='green')
axs[0,0].set_title("Sales Distribution",fontsize=18)
axs[0,1].set_title("Profit Distribution",fontsize=18)
axs[1,0].set_title("Discount Distribution",fontsize=18)
axs[1,1].set_title("Quantity Distribution",fontsize=18)
```

```
C:\Users\mr\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a depre
cated function and will be removed in a future version. Please adapt your code to use either `displot` (a f
igure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\mr\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a depre
cated function and will be removed in a future version. Please adapt your code to use either `displot` (a f
igure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\mr\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a depre
cated function and will be removed in a future version. Please adapt your code to use either `displot` (a f
igure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
C:\Users\mr\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a depre
cated function and will be removed in a future version. Please adapt your code to use either `displot` (a f
igure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[13]:

```
Text(0.5, 1.0, 'Quantity Distribution')
```


Sales Distribution
Density
Sales
0.0000
0.0005
0.0010
0.0015
0.0020
0.0025
0
5000
10000
15000
20000
Profit Distribution
Density
Profit
0.000
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
−5000
−2500
0
2500
5000
7500
Discount Distribution
Density
Discount
0
5
10
15
20
25
0.0
0.2
0.4
0.6
0.8
Quantity Distribution
Density
Quantity
0.0
0.2
0.4
0.6
0.8
0.0
2.5
5.0
7.5
10.0
12.5
15.0

In [14]:
```
# Visualizing using heat map
correlation = data.corr()
sns.heatmap(correlation, annot=True, cmap='coolwarm')
```

Out[14]: <AxesSubplot:>

Values closer to zero shows that there is no linear correlation between those two items. The values clo
se to 1 shows
that they are positively correlated to each other and the values closer to -1 shows they are negatively
correleate to
each other i.e., if the value of one variable increases then the value of another decreases

Conclusion

- Discount and profit are negatively correlated i.e, if discount increases profit decreases and vice versa

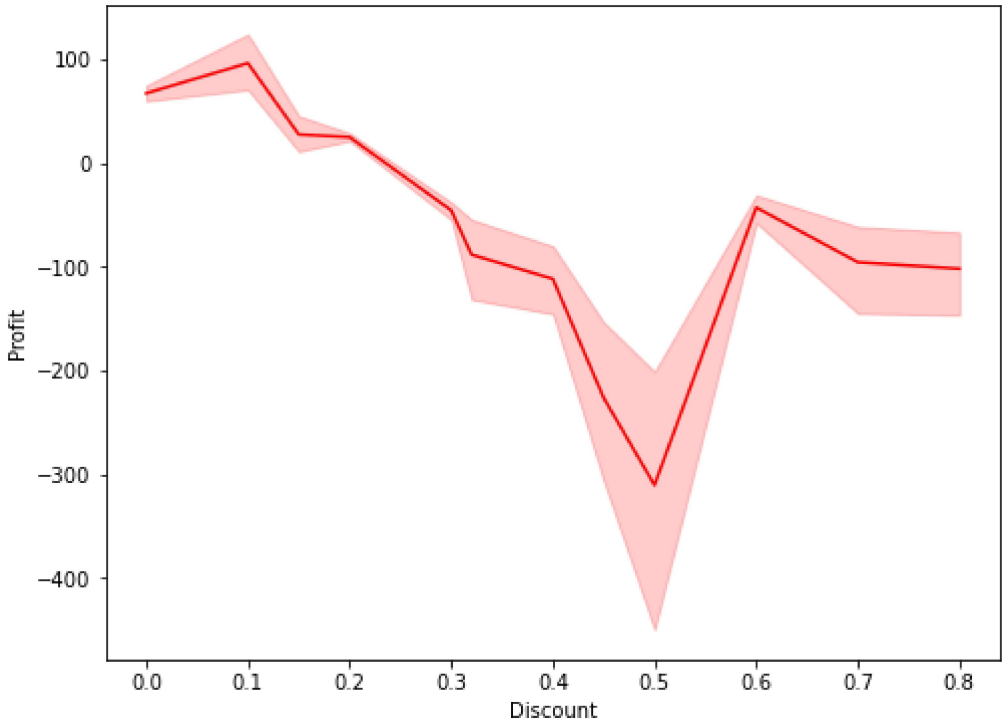
- Sales and profits are moderately correlated

- Discount and sales are negatively correlated

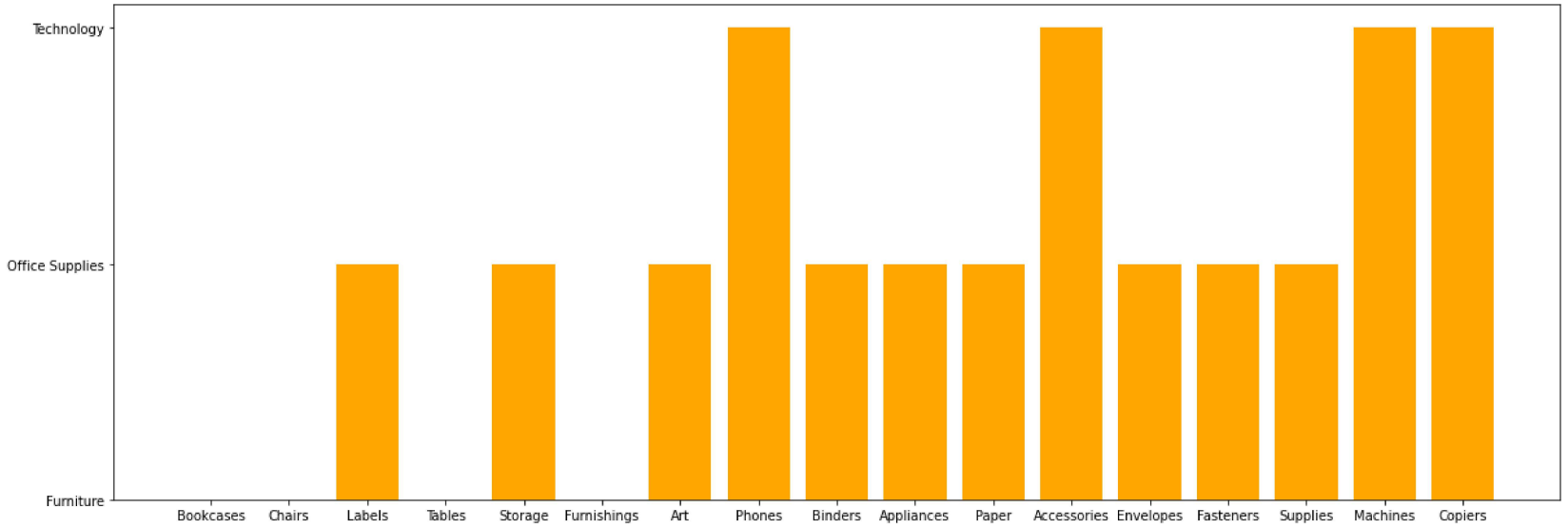
In [15]:
```
# Visualizing the discount and profit
plt.figure(figsize=(8,6))
sns.lineplot(data['Discount'], data['Profit'], data=data, color='red')
```

```
C:\Users\mr\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variab
les as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passin
g other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

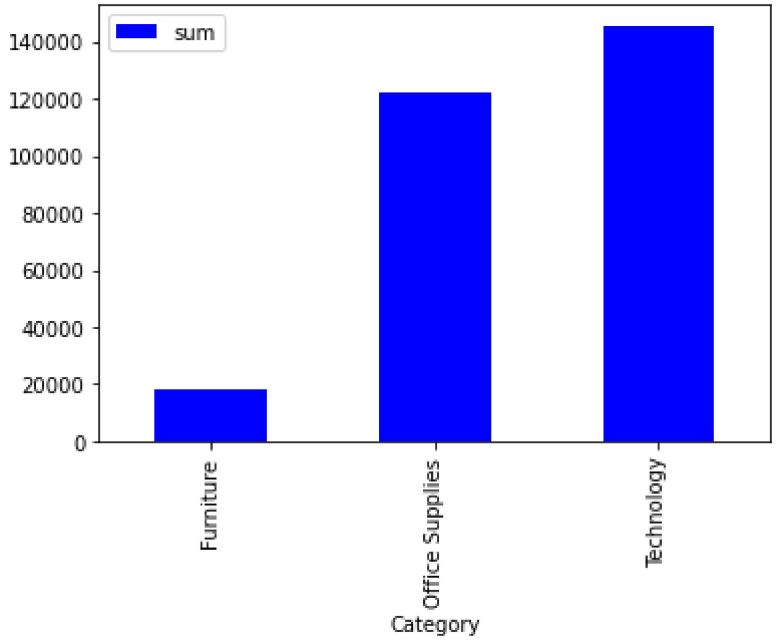
Out[15]: <AxesSubplot:xlabel='Discount', ylabel='Profit'>



In [16]:
```
# Bar Graph between Sub-Category and Category
plt.figure(figsize=(20,7))
plt.bar('Sub-Category','Category',data=data,color='orange')
plt.show()
```



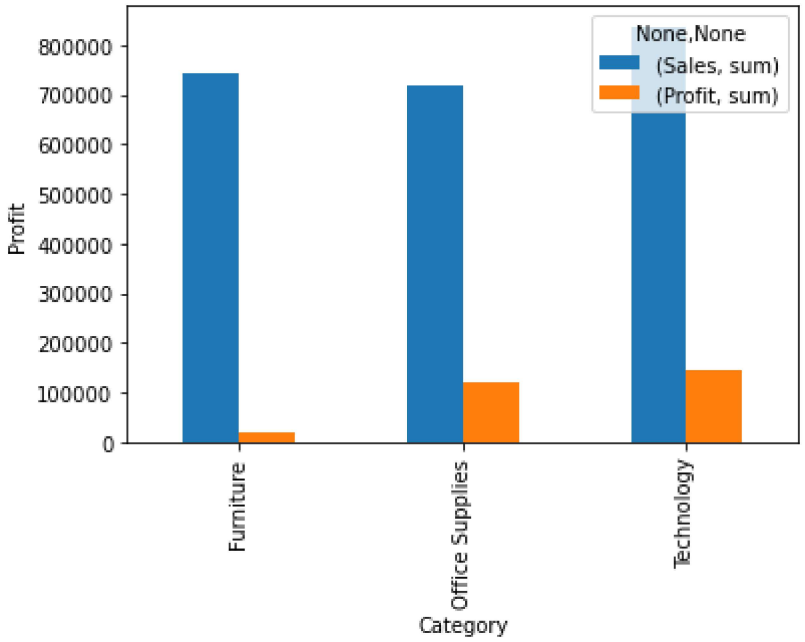
In [17]:
```
# Bar Graph between Category and Profit
data.groupby("Category")['Profit'].agg(['sum']).plot.bar(color='blue')
plt.show()
```



From the above graph we can say that most of the profits are from the Category of Technology

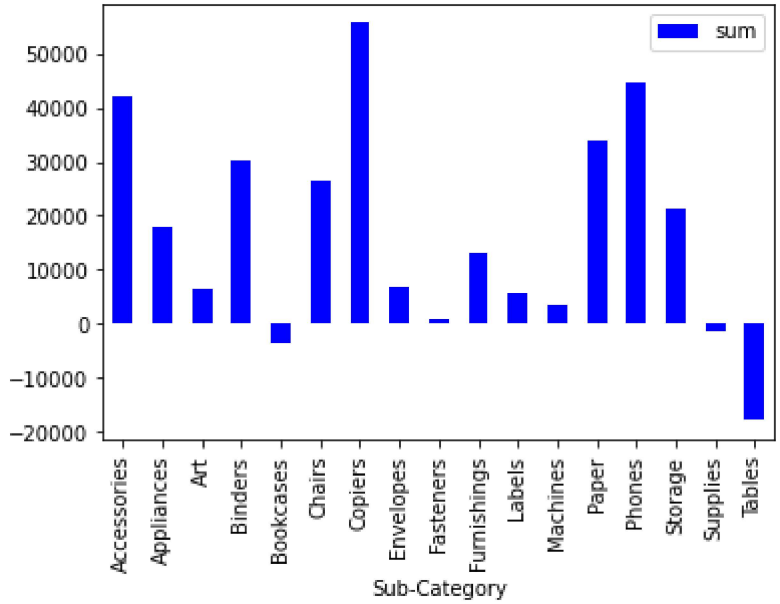
In [18]:
```
# Category wise analysis between Sales and Profit
data.groupby('Category')['Sales','Profit'].agg(['sum']).plot.bar()
plt.ylabel("Profit")
plt.show()
```

```
<ipython-input-18-6cf68a8624b2>:2: FutureWarning: Indexing with multiple keys (implicitly converted to a tu
ple of keys) will be deprecated, use a list instead.
  data.groupby('Category')['Sales','Profit'].agg(['sum']).plot.bar()
```



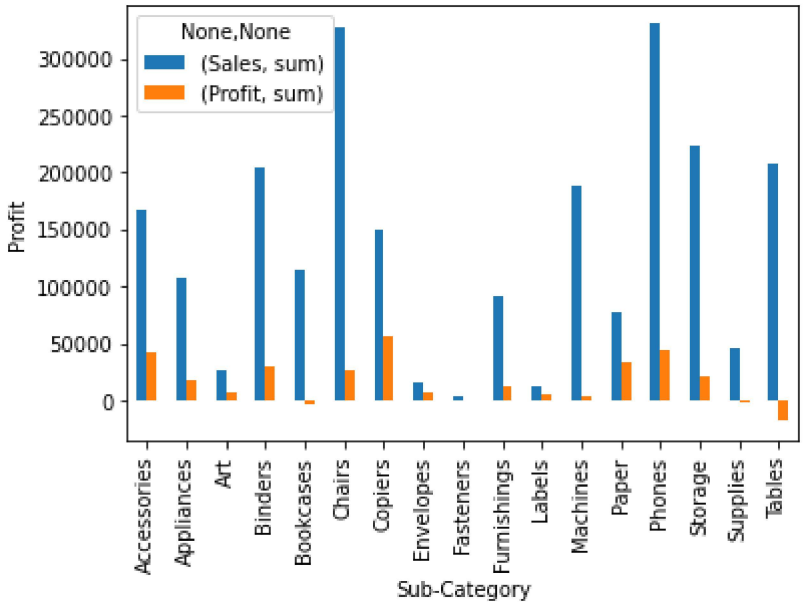
Most of the customers are likely to buy Technological things and also most of the profit is from Technology category

In [19]:
```
# Bar Graph between Category and Profit
data.groupby("Sub-Category")['Profit'].agg(['sum']).plot.bar(color='blue')
plt.show()
```



In [20]:
```
#  Sub-Category wise analysis of Sales and Profit
data.groupby('Sub-Category')['Sales','Profit'].agg(['sum']).plot.bar()
plt.ylabel("Profit")
plt.show()
```

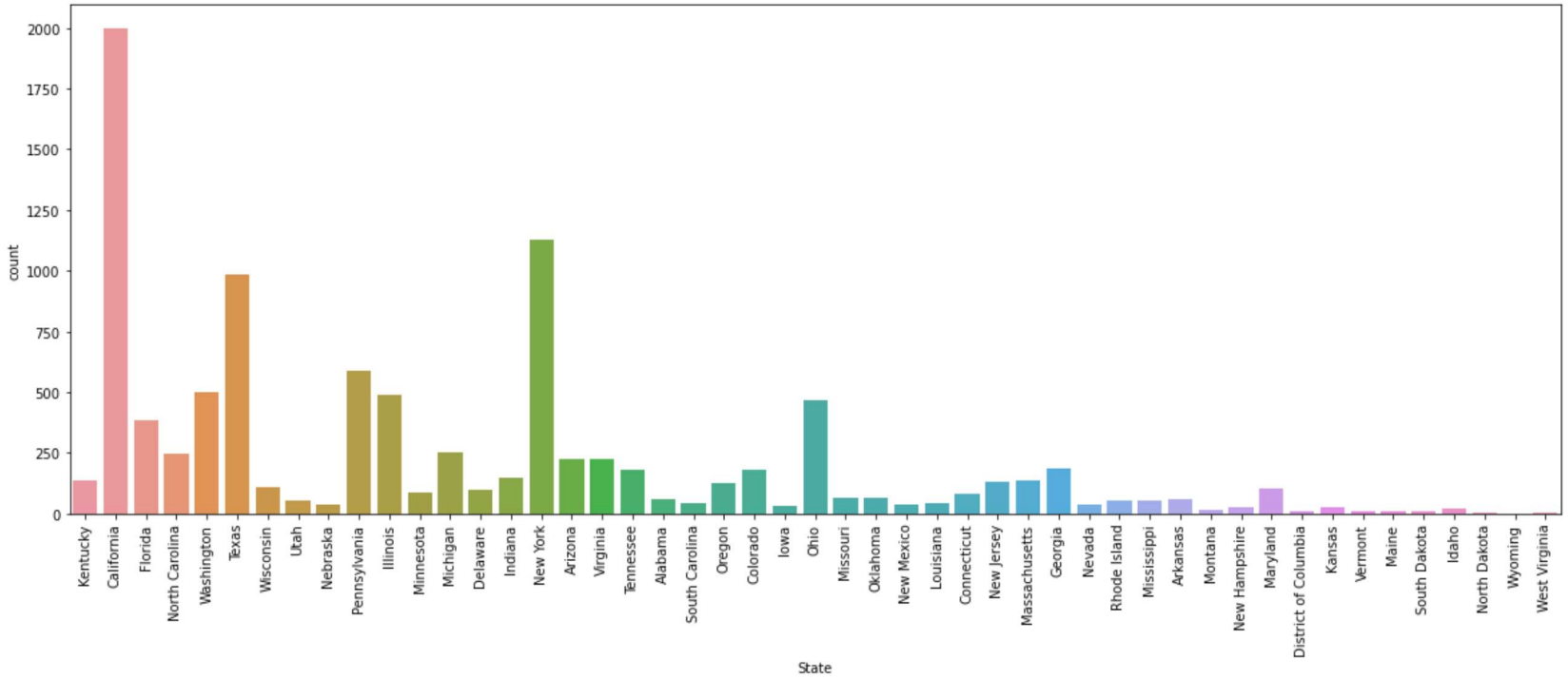
```
<ipython-input-20-5c33f9023653>:2: FutureWarning: Indexing with multiple keys (implicitly converted to a tu
ple of keys) will be deprecated, use a list instead.
  data.groupby('Sub-Category')['Sales','Profit'].agg(['sum']).plot.bar()
```



Conclusion

- Maximum sales are noted in the phones category,
- But maximum profits are from copiers,
- Minimum Profit / loss is from Tabels.

In [21]:
```
# State wise dealings
plt.figure(figsize=(20,7))
sns.countplot(x=data['State'])
plt.xticks(rotation=90)
plt.show()
```



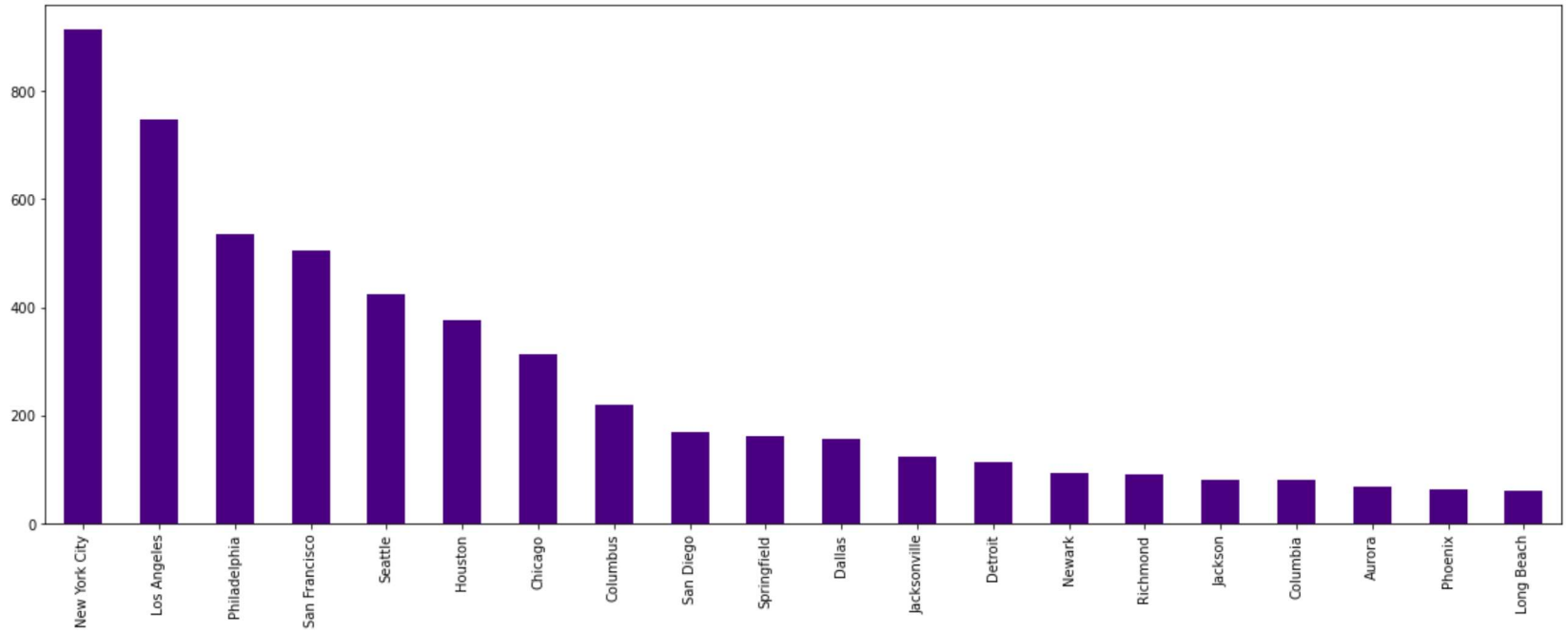
Most of the dealings are made from the state california

In [22]:
```
# Calculating the average number of dealings per state
state = data['State'].value_counts()
state.mean()
```

Out[22]: 203.6122448979592

Average number of delayings per state is 203.612244

In [23]:
```
# Graph Between state and the number of dealings per state
df = data['City'].value_counts()
df = df.head(20)
df.plot(kind='bar',figsize=(20,7), color='Indigo')
plt.xticks(rotation=90)
plt.show()
```



New York, Los Angles, Philadelphia are the top 3 City which have maximum number of dealings

State wise analysis of profit, Discount and Sales

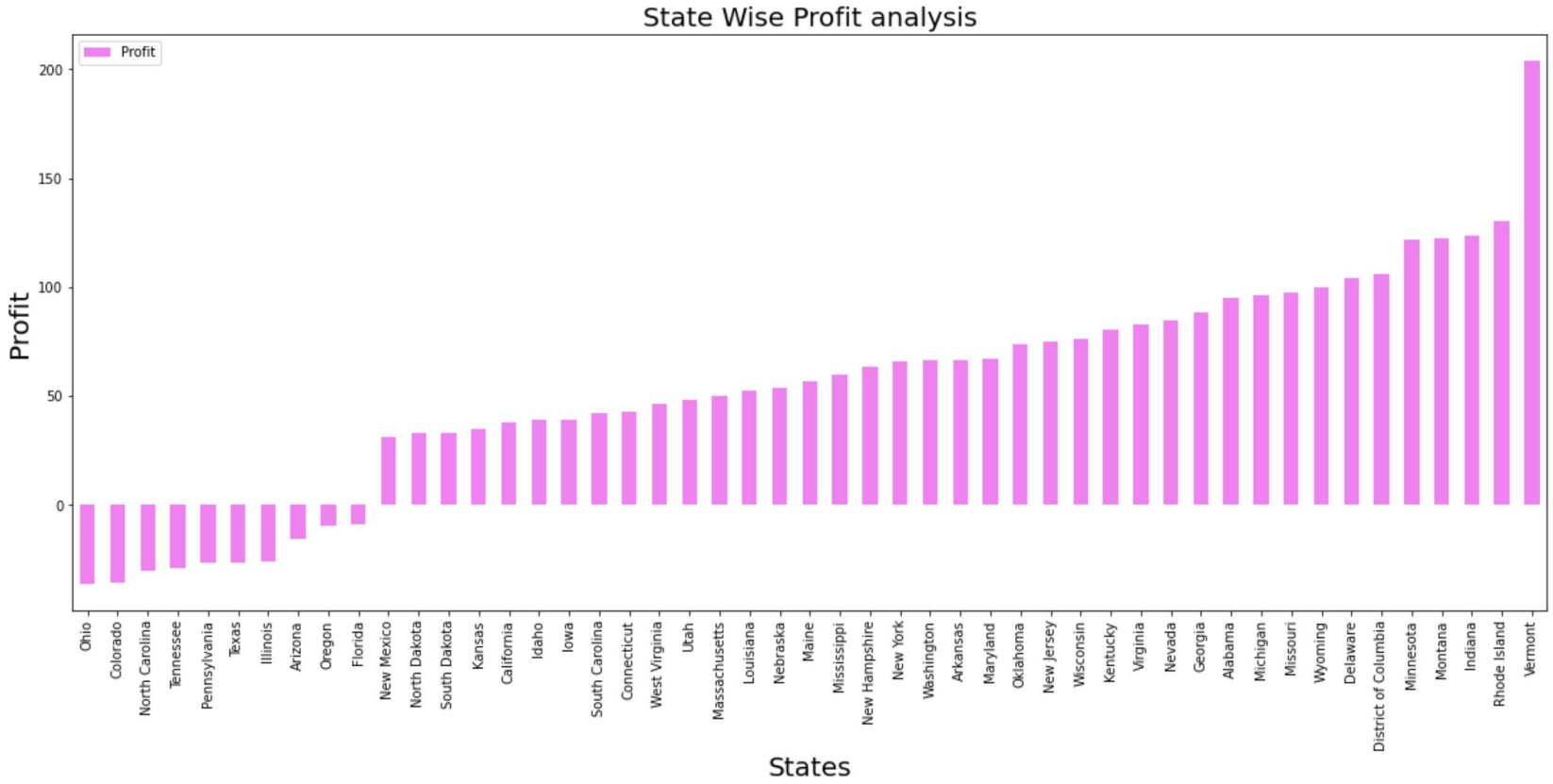
In [24]:
```
# State wise analysis of profit, sales, discount
df1 = data.groupby(['State'])[['Profit','Discount','Sales']].mean()
df1.head()
```

Out[24]:

State	Profit	Discount	Sales
Alabama	94.865989	0.000000	319.846557
Arizona	-15.303235	0.303571	157.508933
Arkansas	66.811452	0.000000	194.635500
California	38.241878	0.072946	229.246629
Colorado	-35.867351	0.316484	176.418231

In [25]:
```
 # State wise Profit Analysis
state_profit = df1.sort_values('Profit')
state_profit[['Profit']].plot(kind='bar',figsize=(20,8), color='violet')
plt.title("State Wise Profit analysis",fontsize=20)
plt.xlabel("States",fontsize=20)
plt.ylabel("Profit",fontsize=20)
```

Out[25]: Text(0, 0.5, 'Profit')

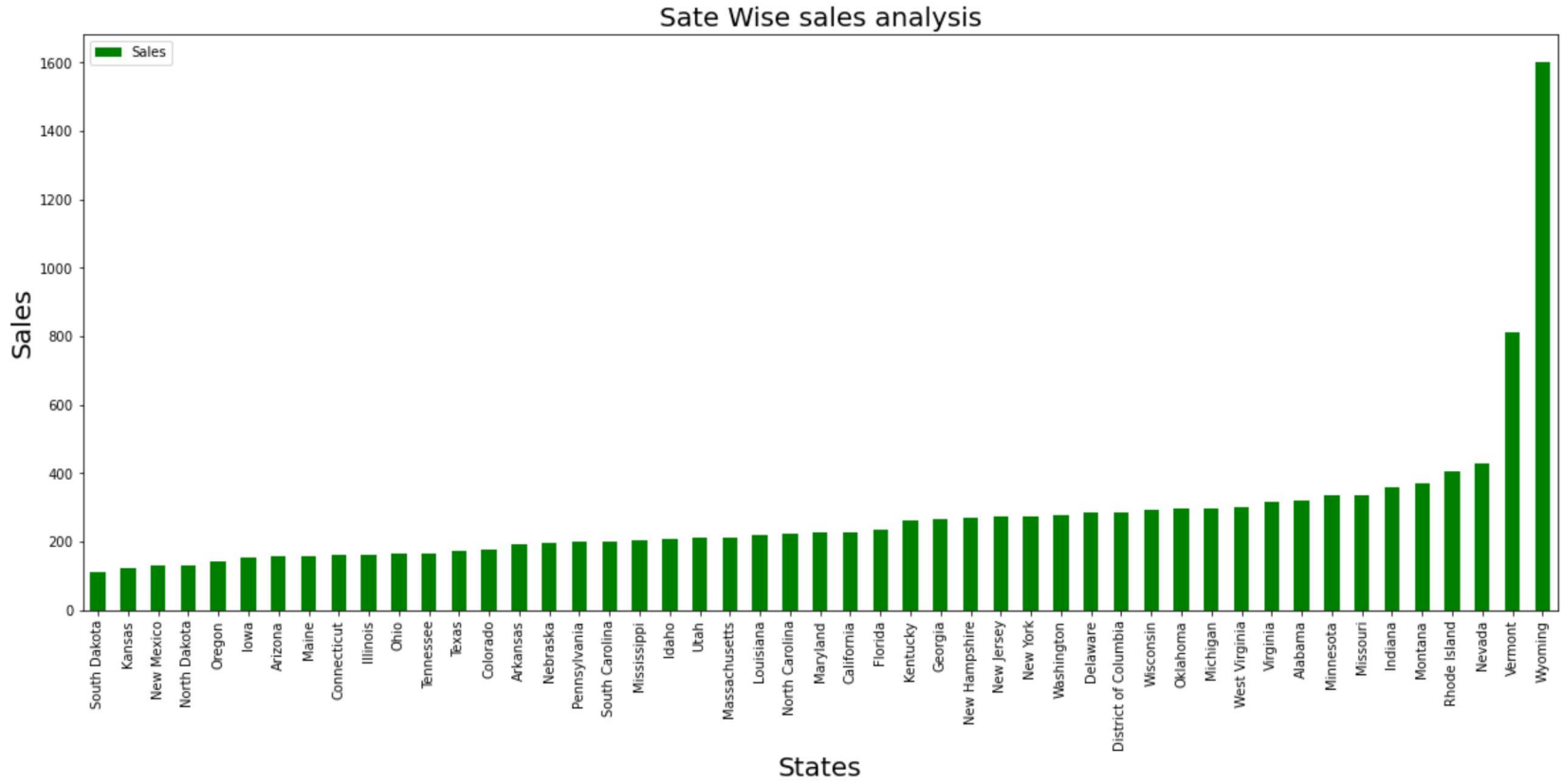


Highest Profit State : Vermont

Lowest Profit State : Ohio

In [26]:
```
# State wise Sales Analysis
state_sales = df1.sort_values('Sales')
state_sales[['Sales']].plot(kind='bar',figsize=(20,8), color='green')
plt.title("Sate Wise sales analysis", fontsize=20)
plt.xlabel("States",fontsize=20)
plt.ylabel("Sales",fontsize=20)
```

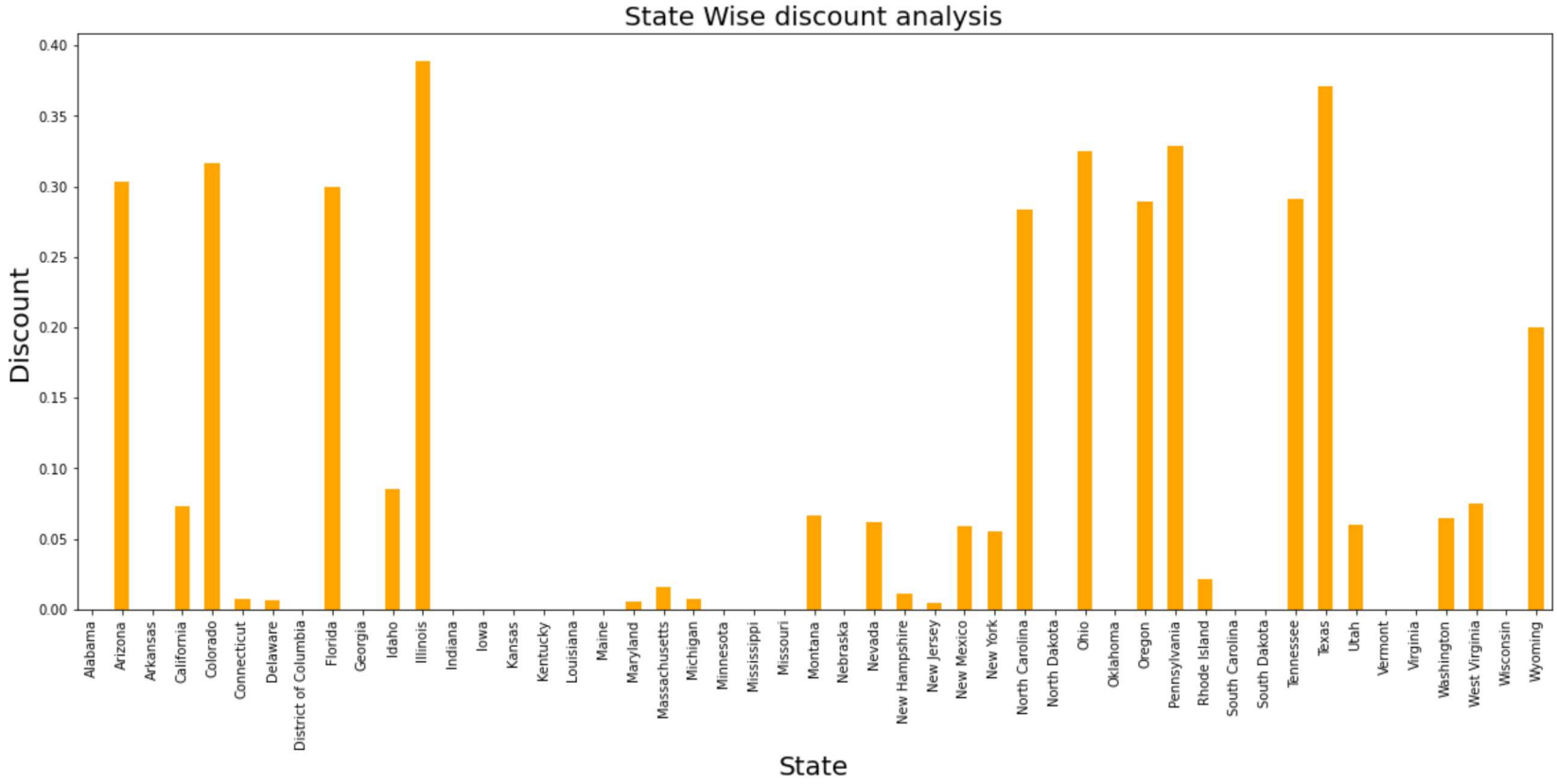
Out[26]:
```
Text(0, 0.5, 'Sales')
```



State with Highest Sales : Wyoming

In [27]:
```
# State  wise Discount Analysis
df1['Discount'].plot(kind='bar',figsize=(20,8), color='orange')
plt.title("State Wise discount analysis", fontsize=20)
plt.xlabel("State",fontsize=20)
plt.ylabel("Discount",fontsize=20)
```

Out[27]: Text(0, 0.5, 'Discount')



Highest Discount providing state - LIlinois

Lowest Discount providing state - Lowa, Kanas...

City wise Profit Analysis

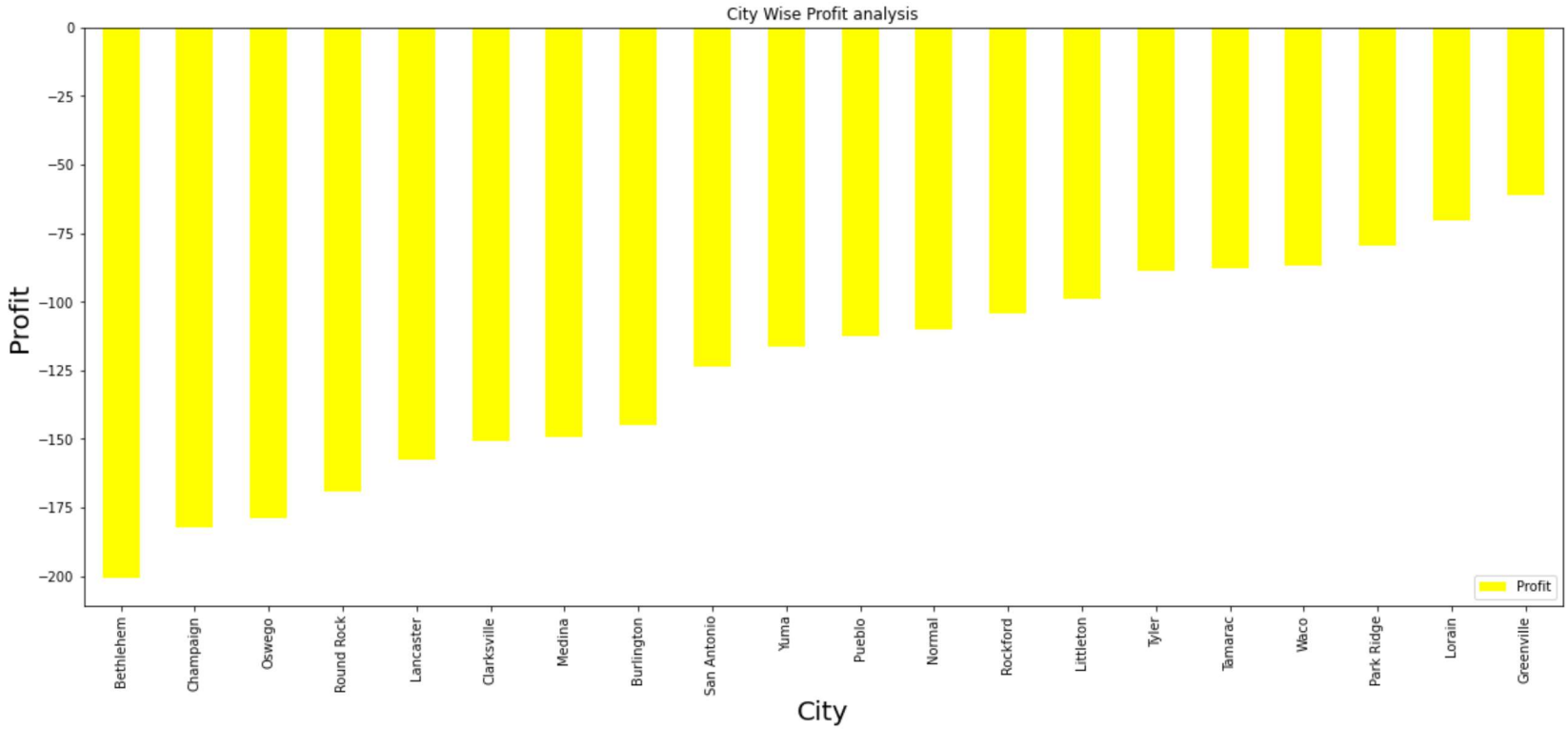
In [28]:
```
# City wise profit analysis
df2 = data.groupby(['City'])[['Profit']].mean()
df2.head()
```

Out[28]:

City	Profit
Aberdeen	6.630000
Abilene	-3.758400
Akron	-8.887410
Albuquerque	45.292007
Alexandria	19.913644

In [29]:
```
# City wise profit analysis which states are under loss
city_profit = df2.sort_values('Profit').head(20)
city_profit[['Profit']].plot(kind='bar',figsize=(20,8), color='yellow')
plt.title("City Wise Profit analysis")
plt.xlabel("City",fontsize=20)
plt.ylabel("Profit",fontsize=20)
```

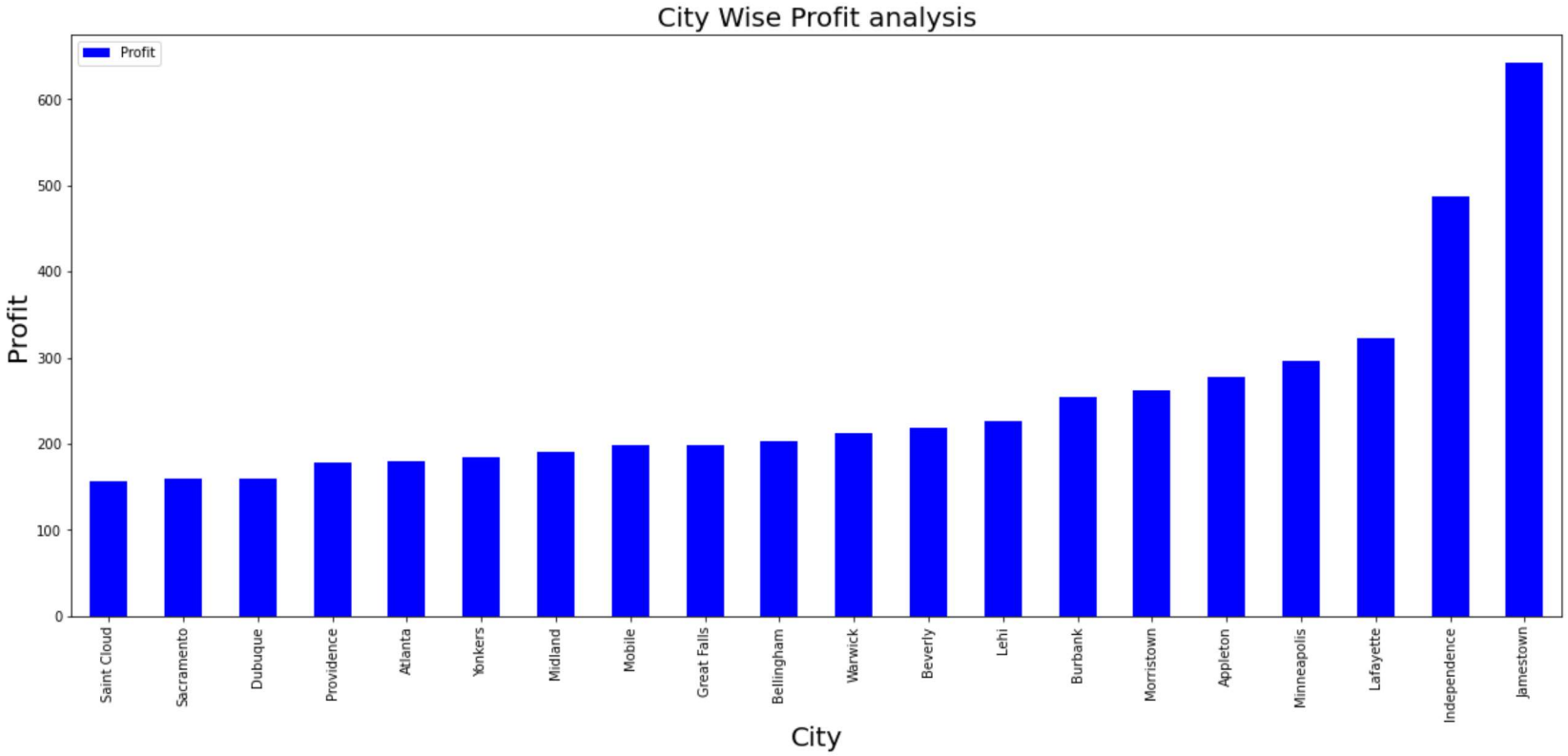
Out[29]: Text(0, 0.5, 'Profit')



City with Lowest Profit : Bethlehern

In [30]:
```
# City wise analysis of which cities are more profitable
city_profit = df2.sort_values('Profit').tail(20)
city_profit[['Profit']].plot(kind='bar',figsize=(20,8), color='blue')
plt.title("City Wise Profit analysis",fontsize=20)
plt.xlabel("City",fontsize=20)
plt.ylabel("Profit",fontsize=20)
```

Out[30]: Text(0, 0.5, 'Profit')



City with Highest Profit : Jamestown

Category wise profit

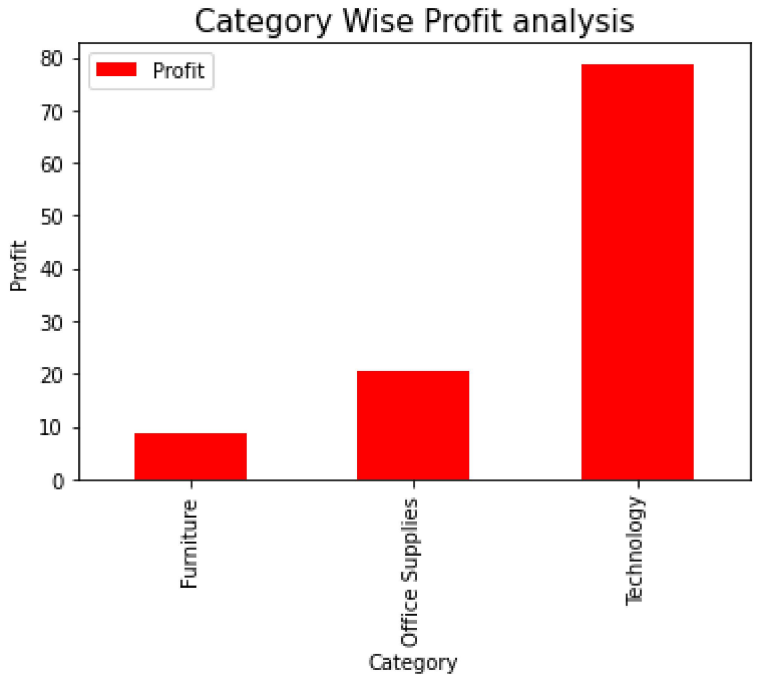
In [31]:
```
# Category wise profit
df3 = data.groupby(['Category'])[['Profit']].mean()
df3
```

Out[31]:

Category	Profit
Furniture	8.697740
Office Supplies	20.353403
Technology	78.752002

In [32]:
```
# Visualizing Category wise profit analysis
category_profit = df3.sort_values('Profit').tail(20)
category_profit[['Profit']].plot(kind='bar', color='red')
plt.title("Category Wise Profit analysis",fontsize=15)
plt.xlabel("Category")
plt.ylabel("Profit")
```

Out[32]:
```
Text(0, 0.5, 'Profit')
```



Highest Profit providing Category: Technology

Lowest Profit providing Category : Furniture

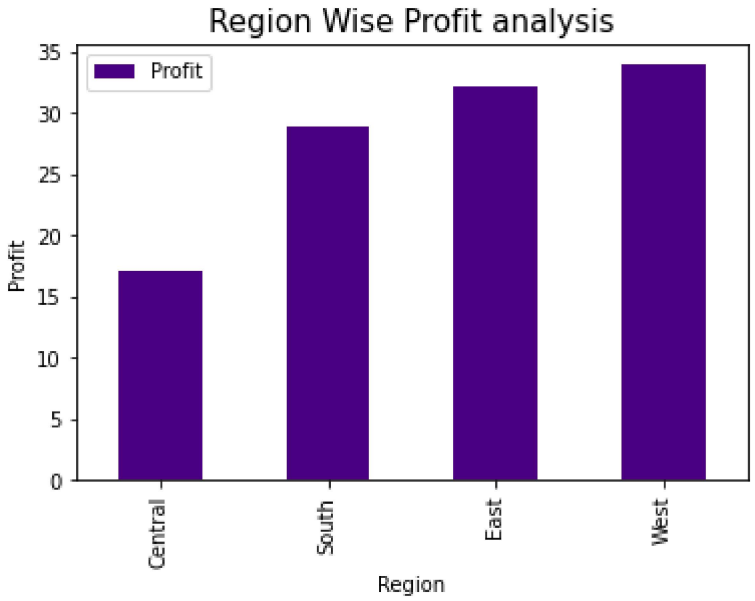
In [33]:
```
# Region wise profit
df4 = data.groupby(['Region'])[['Profit']].mean()
df4
```

Out[33]:

Region	Profit
Central	17.100421
East	32.163905
South	28.857673
West	33.927281

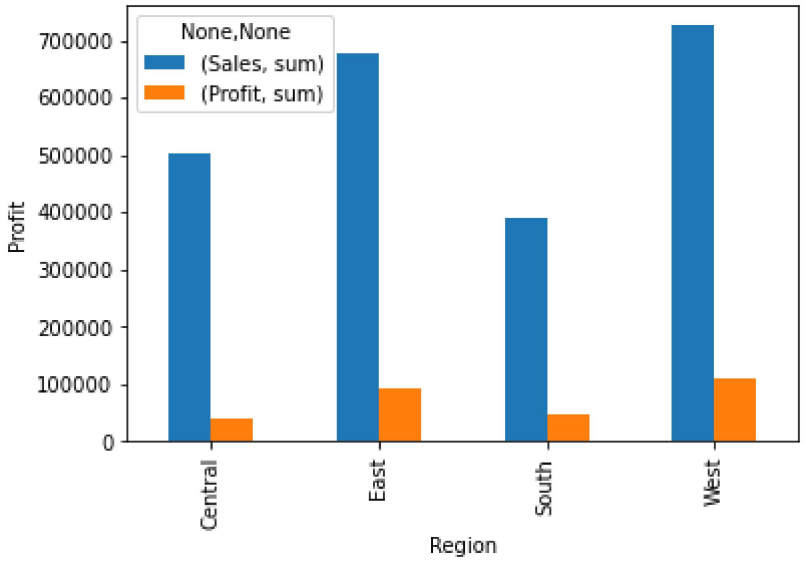
In [34]:
```
# Region wise Profit analysis
region_profit = df4.sort_values('Profit')
region_profit[['Profit']].plot(kind='bar', color='Indigo')
plt.title("Region Wise Profit analysis",fontsize=15)
plt.xlabel("Region")
plt.ylabel("Profit")
```

Out[34]:
```
Text(0, 0.5, 'Profit')
```



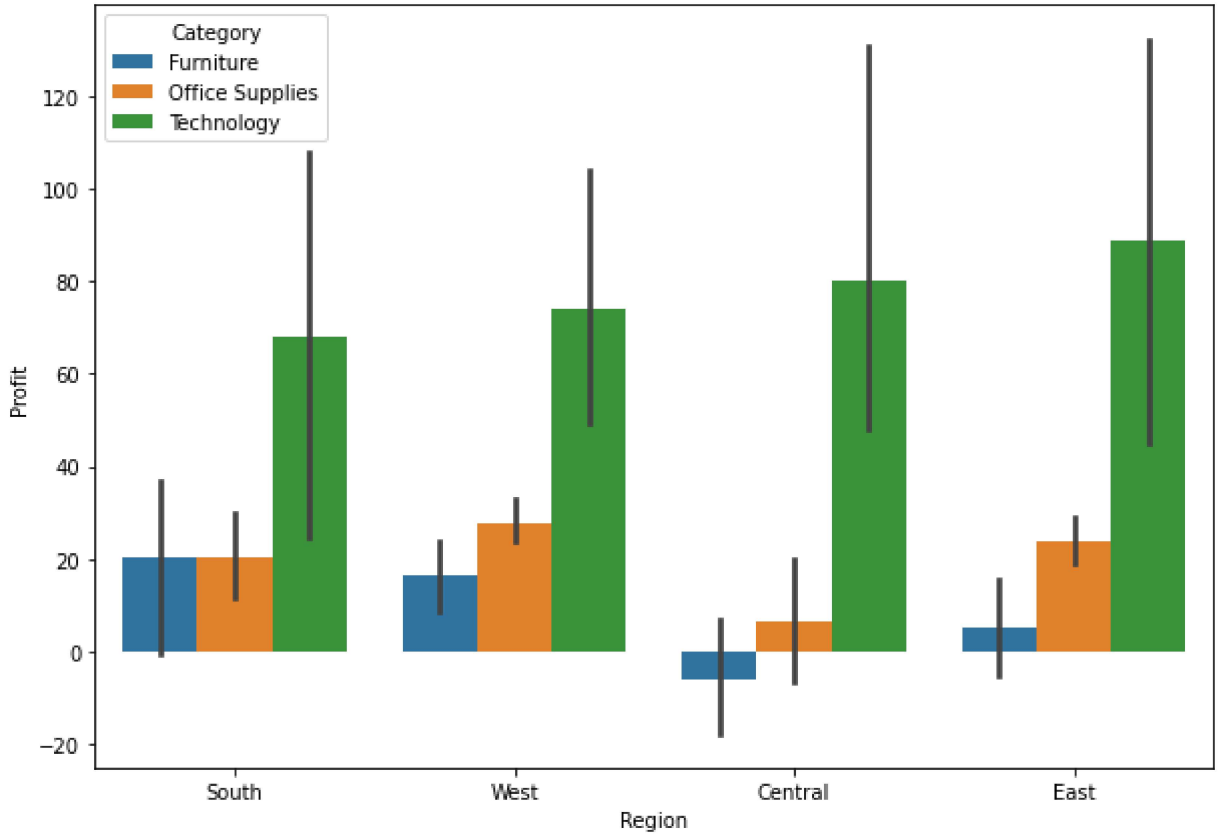
In [35]:
```
#  Region wise analysis between Sales and Profit
data.groupby('Region')['Sales','Profit'].agg(['sum']).plot.bar()
plt.ylabel("Profit")
plt.show()
```

```
<ipython-input-35-fb83d9d3aa21>:2: FutureWarning: Indexing with multiple keys (implicitly converted to a tu
ple of keys) will be deprecated, use a list instead.
  data.groupby('Region')['Sales','Profit'].agg(['sum']).plot.bar()
```

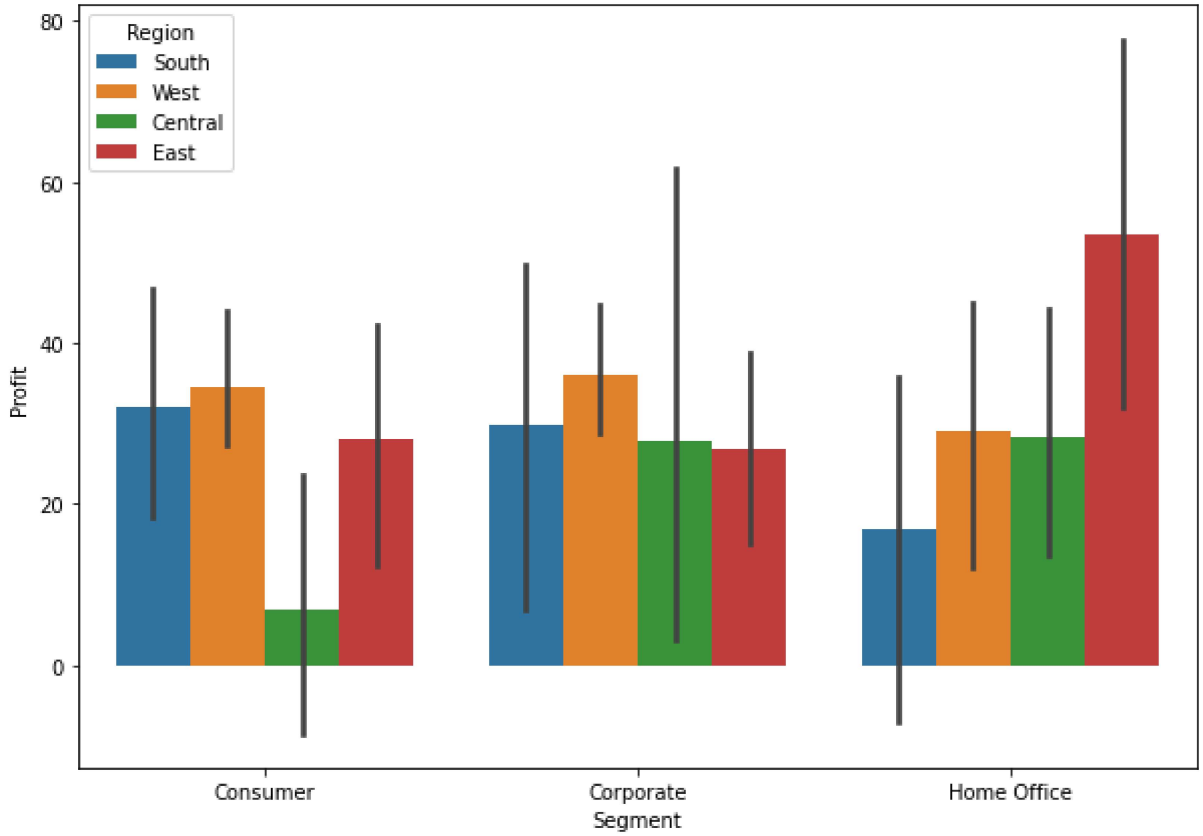


Highest Sales and Profit is from the Region,West

In [36]:
```
# Graph of Region Profit
plt.figure(figsize=(10,7))
sns.barplot(x=data.Region, y=data.Profit, hue=data.Category)
plt.show()
```



In [37]:
```
plt.figure(figsize=(10,7))
sns.barplot(x=data.Segment, y=data.Profit, hue=data.Region)
plt.show()
```



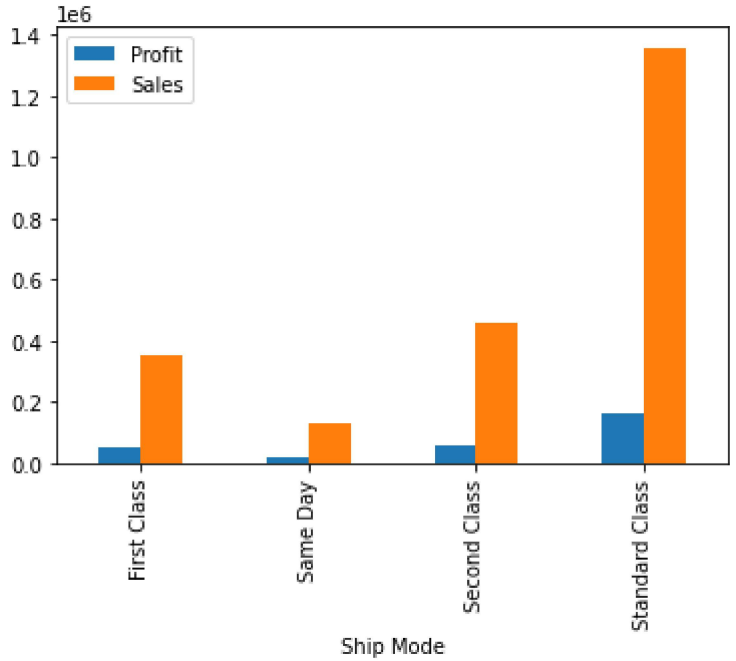
In [38]:
```
# Ship Mode wise Profit and Sales wise Analysis
df5 = data.groupby(['Ship Mode'])[['Profit','Sales']].mean()
df5
```

Out[38]:

Ship Mode	Profit	Sales
First Class	31.850134	228.614490
Same Day	29.283924	236.755950
Second Class	29.565956	236.323750
Standard Class	27.534715	227.928858

In [39]:
```
# Ship Mode Analysis of Sales and profit
data.groupby(['Ship Mode'])[['Profit','Sales']].sum().plot.bar()
```

Out[39]: <AxesSubplot:xlabel='Ship Mode'>



- Maximum number of shipments belongs to standard class
- Very less number of shipments are done in the same day

Business problems that can be derive by exploring the data are...

- We should limit sales of furniture and focus on Technology category to increase the profit

- Consedering the Sub-Categories it generates maximum loss, sales of tables should be minimized. Should increase sales in the west, since it is the maximum profit area.

- We should concentrate on improving the sales of the states like New York and California so that we can make more profits

- Discount should be less than or equal to 50% brings us the highest sales and profit marging
- Provide optimal Discount to technology and furniture to attract more number of customers
- The top 5 Sub-Categories with the sales less than 50%, these catogires should accelerate according to the markeing strategies or they should introduce additional products in those categories
- Profits can be made if the ship mode is Standard Class
- As furniture has low profit margin and have more storage cost, and people don't prefer to buy furniture, books and tables from the super store
- In House-office segment we have high profit and sales, we can promot these to improve for better results