Task:Exploratory Data Analysis - Retail

In this task, we will perform ‘Exploratory Data Analysis’ on dataset ‘SampleSuperstore’. As a business manager, we will try to find out the weak areas where we can work to make more profit. Also, what all business problems can be derived by exploring the data.

By Gayathri R

In [1]:

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

In [2]:

```
#Reading the dataset
retail = pd.read_csv('SampleSuperstore.csv')
```

In [3]:

```
retail
```

Out[3]:

	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

	Ship Mode	Segment	Country	City	State	Postal Code	Region	Category	Sub Category
4	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Office Supplies	Storage
...	...	...	...	...	...	...	...	...	..
9989	Second Class	Consumer	United States	Miami	Florida	33180	South	Furniture	Furnishings
9990	Standard Class	Consumer	United States	Costa Mesa	California	92627	West	Furniture	Furnishings
9991	Standard Class	Consumer	United States	Costa Mesa	California	92627	West	Technology	Phones
9992	Standard Class	Consumer	United States	Costa Mesa	California	92627	West	Office Supplies	Paper
9993	Second Class	Consumer	United States	Westminster	California	92683	West	Office Supplies	Appliances

9994 rows × 13 columns

In [4]:

```
#top five observations
retail.head()
```

Out[4]:

	Ship Mode	Segment	Country	City	State	Postal Code	Region	Category	Sub-Category	S
0	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Bookcases	261.
1	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Chairs	731.
2	Second Class	Corporate	United States	Los Angeles	California	90036	West	Office Supplies	Labels	14.
3	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Furniture	Tables	957.
4	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Office Supplies	Storage	22.

```
In [5]: #bottom five observations
        retail.tail()
```

Out[5]:

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

```
In [6]: #data size
        retail.shape
```

```
Out[6]: (9994, 13)
```

Reading the information that is provided in the dataset

```
In [7]: #data info
        retail.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 [8]:

```
#checking for missing data
retail.isnull()
```

Out[8]:

	Ship Mode	Segment	Country	City	State	Postal Code	Region	Category	Sub-Category	Sales	Quanti
0	False	False	False	False	False	False	False	False	False	False	Fals
1	False	False	False	False	False	False	False	False	False	False	Fals
2	False	False	False	False	False	False	False	False	False	False	Fals
3	False	False	False	False	False	False	False	False	False	False	Fals
4	False	False	False	False	False	False	False	False	False	False	Fals
...	...	...	...	...	...	...	...	...	...	...	
9989	False	False	False	False	False	False	False	False	False	False	Fals
9990	False	False	False	False	False	False	False	False	False	False	Fals
9991	False	False	False	False	False	False	False	False	False	False	Fals
9992	False	False	False	False	False	False	False	False	False	False	Fals
9993	False	False	False	False	False	False	False	False	False	False	Fals

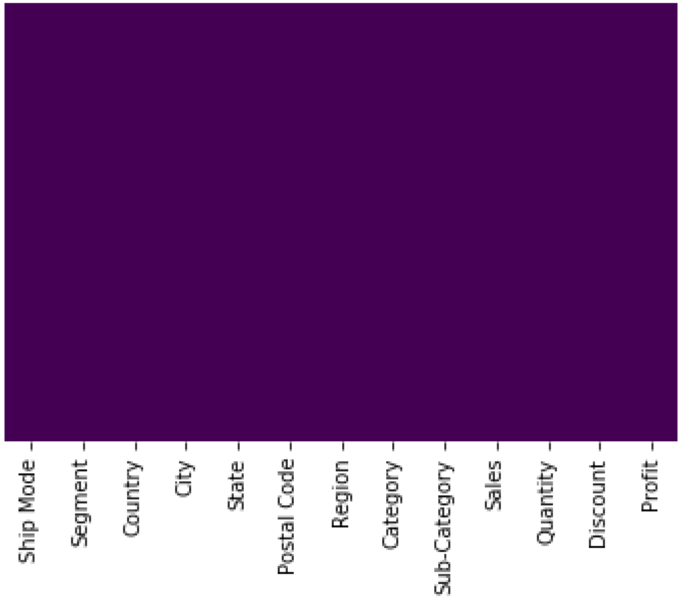
9994 rows × 13 columns

In [9]:
```
retail.isnull().sum()
```

Out[9]:
```
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 [10]:
```
sns.heatmap(retail.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[10]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x21efb8bd248>
```



In [11]:
```
#statistics of the data
retail.describe()
```

Out[11]:

	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

Checking for the duplicate data

In [12]:
```
#checking for duplicate data
retail.duplicated().sum()
```

Out[12]:
```
17
```

In [13]:
```
duplicate=retail.duplicated()
retail[duplicate]
```

Out[13]:

	Ship Mode	Segment	Country	City	State	Postal Code	Region	Category	Su Catego
950	Standard Class	Home Office	United States	Philadelphia	Pennsylvania	19120	East	Office Supplies	Pa
3406	Standard Class	Home Office	United States	Columbus	Ohio	43229	East	Furniture	Cha

	Ship Mode	Segment	Country	City	State	Postal Code	Region	Category	Su Catego
3670	Standard Class	Consumer	United States	Salem	Oregon	97301	West	Office Supplies	Pa
4117	Standard Class	Consumer	United States	Los Angeles	California	90036	West	Office Supplies	Pa
4553	Standard Class	Consumer	United States	San Francisco	California	94122	West	Office Supplies	Pa
5905	Same Day	Home Office	United States	San Francisco	California	94122	West	Office Supplies	Lab
6146	Standard Class	Corporate	United States	San Francisco	California	94122	West	Office Supplies	
6334	Standard Class	Consumer	United States	New York City	New York	10011	East	Office Supplies	Pa
6357	Standard Class	Corporate	United States	Seattle	Washington	98103	West	Office Supplies	Pa
7608	Standard Class	Consumer	United States	San Francisco	California	94122	West	Office Supplies	Pa
7735	Standard Class	Corporate	United States	Seattle	Washington	98105	West	Office Supplies	Pa
7759	Standard Class	Corporate	United States	Houston	Texas	77041	Central	Office Supplies	Pa
8032	First Class	Consumer	United States	Houston	Texas	77041	Central	Office Supplies	Pa
8095	Second Class	Consumer	United States	Seattle	Washington	98115	West	Office Supplies	Pa
9262	Standard Class	Consumer	United States	Detroit	Michigan	48227	Central	Furniture	Cha
9363	Standard Class	Home Office	United States	Seattle	Washington	98105	West	Furniture	Furnishir
9477	Second Class	Corporate	United States	Chicago	Illinois	60653	Central	Office Supplies	Bind

```
In [14]: retail.drop_duplicates(inplace = True)
```

```
In [15]: #confirming all duplicates are removed
         rd = retail.duplicated()
         rd.sum()
```

```
Out[15]: 0
```

```
In [16]: #checking the unique values
         retail.nunique()
```

```
Out[16]: 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 [17]: retail.State.unique()
```

```
Out[17]: array(['Kentucky', 'California', 'Florida', 'North Carolina',
                'Washington', 'Texas', 'Wisconsin', 'Utah', 'Nebraska',
                'Pennsylvania', 'Illinois', 'Minnesota', 'Michigan', 'Delaware',
                'Indiana', 'New York', 'Arizona', 'Virginia', 'Tennessee',
                'Alabama', 'South Carolina', 'Oregon', 'Colorado', 'Iowa', 'Ohi
         o',
                'Missouri', 'Oklahoma', 'New Mexico', 'Louisiana', 'Connecticu
         t',
                'New Jersey', 'Massachusetts', 'Georgia', 'Nevada', 'Rhode Islan
         d',
                'Mississippi', 'Arkansas', 'Montana', 'New Hampshire', 'Marylan
         d',
                'District of Columbia', 'Kansas', 'Vermont', 'Maine',
                'South Dakota', 'Idaho', 'North Dakota', 'Wyoming',
                'West Virginia'], dtype=object)
```

In [18]:
```
retail.Country.unique()
```

Out[18]:
```
array(['United States'], dtype=object)
```

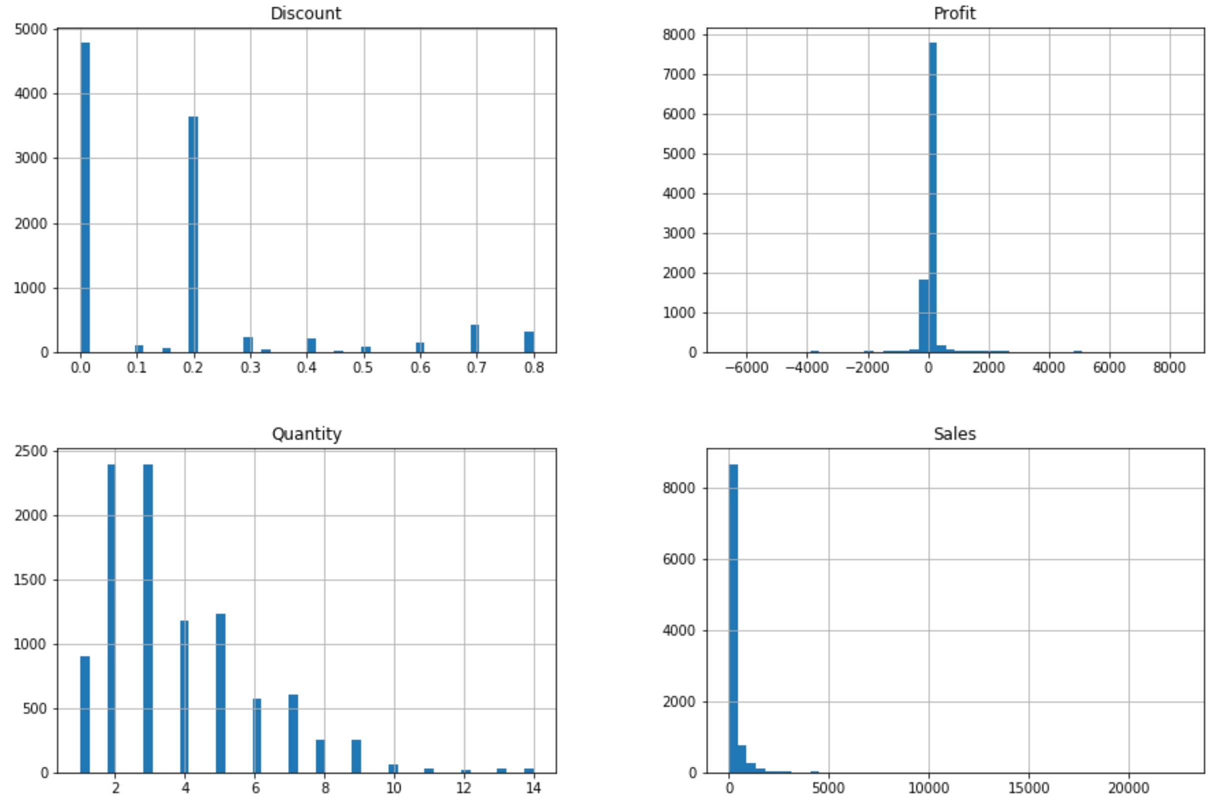
In [19]:
```
#removing the unimportant columns
retail=retail.drop(['Country', 'Postal Code'], axis=1)
```

In [20]:
```
#summary of dataset
retail.describe()
```

Out[20]:

	Sales	Quantity	Discount	Profit
count	9977.000000	9977.000000	9977.000000	9977.00000
mean	230.148902	3.790719	0.156278	28.69013
std	623.721409	2.226657	0.206455	234.45784
min	0.444000	1.000000	0.000000	-6599.97800
25%	17.300000	2.000000	0.000000	1.72620
50%	54.816000	3.000000	0.200000	8.67100
75%	209.970000	5.000000	0.200000	29.37200
max	22638.480000	14.000000	0.800000	8399.97600

In [21]:
```
retail.hist(figsize=(15, 10), bins=50)
plt.show()
```



Checking the statistical relation between the various rows & columns

```
In [22]: #Correlation between the Sales, Quantity, Discount and Profit
         retail.corr()
```

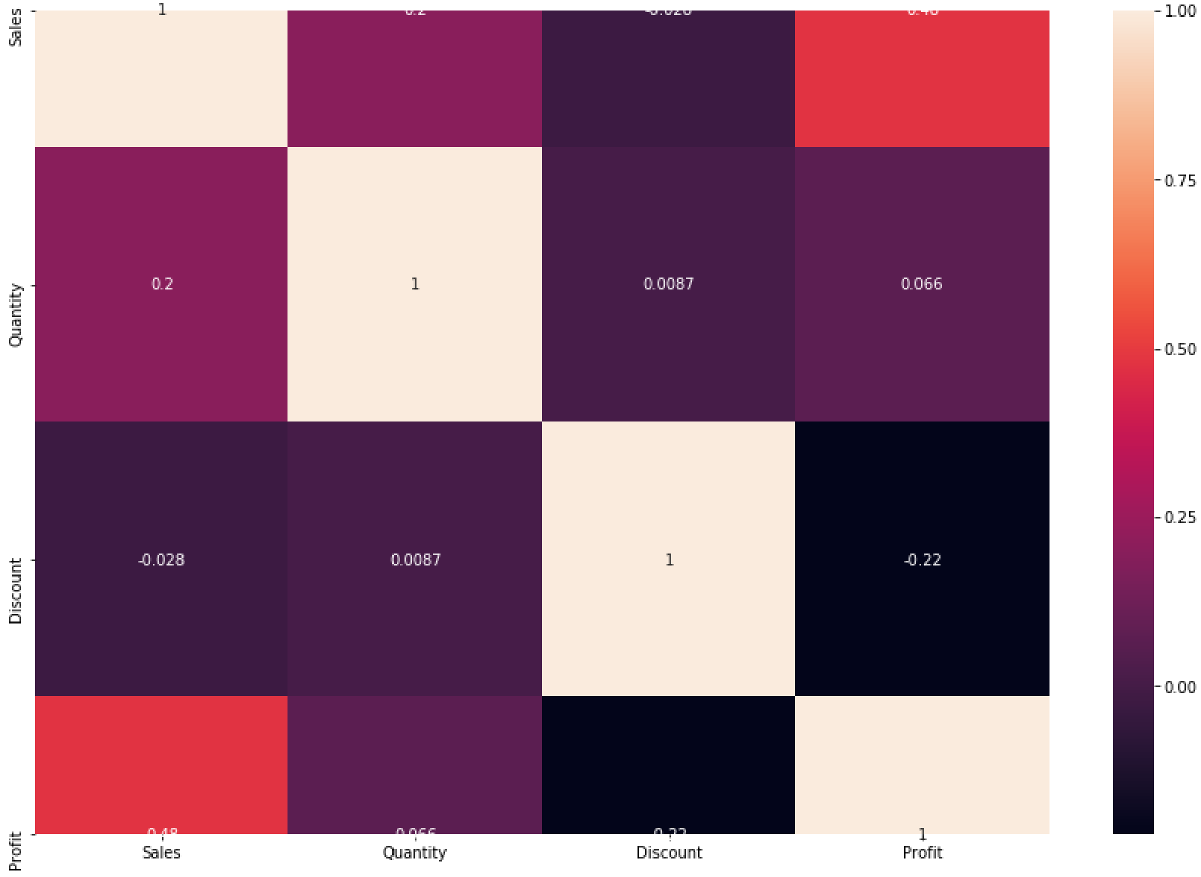
Out[22]:

	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

	Sales	Quantity	Discount	Profit
Profit	0.479067	0.066211	-0.219662	1.000000

In [23]:
```
#Checking correlation between columns visually
f,ax = plt.subplots(figsize=(15, 10))
sns.heatmap(retail.corr(),annot=True)
```

Out[23]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x21efe65e848>
```



In [24]:
```
#Covariance between the Sales, Quantity, Discount and Profit
retail.cov()
```

Out[24]:

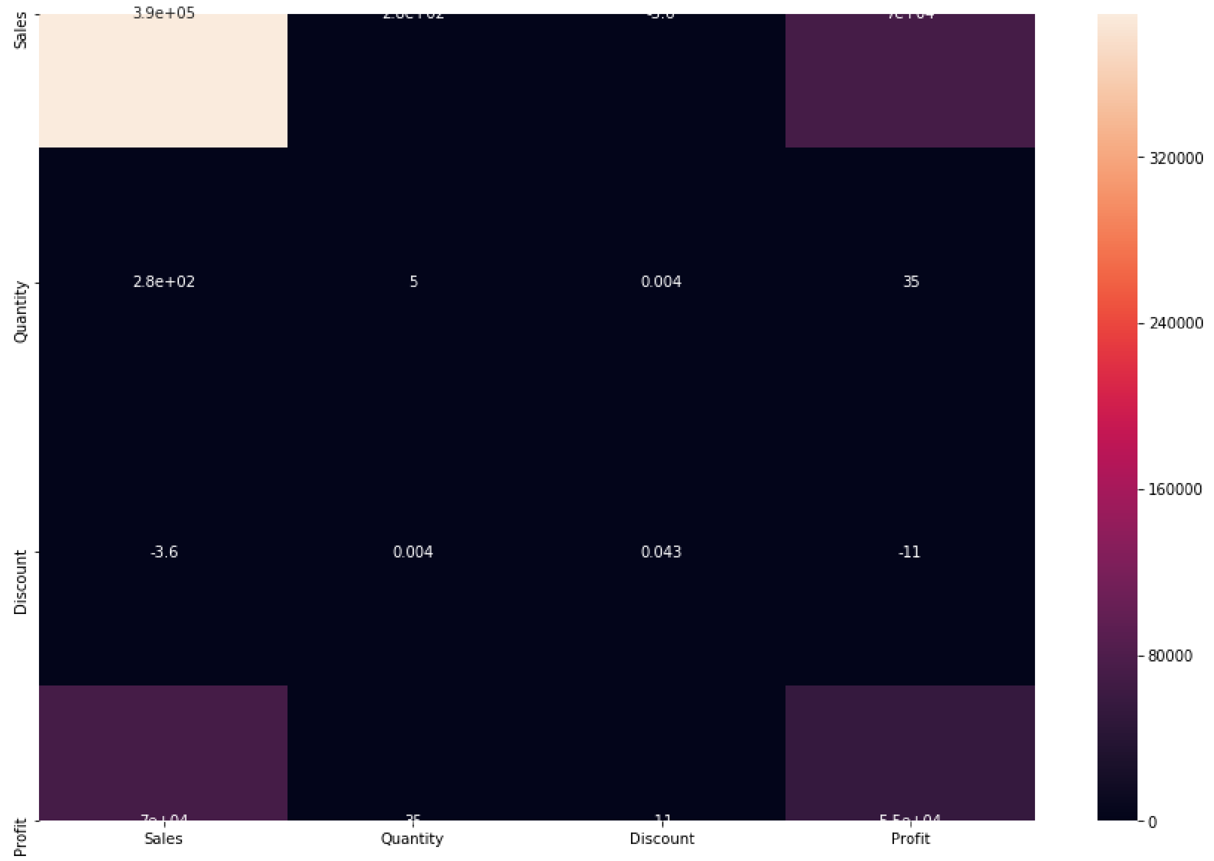
	Sales	Quantity	Discount	Profit
Sales	389028.396022	278.765576	-3.645637	70057.067126
Quantity	278.765576	4.958001	0.003990	34.565743
Discount	-3.645637	0.003990	0.042624	-10.632751
Profit	70057.067126	34.565743	-10.632751	54970.478824

In [25]:

```
#Checking correlation between columns visually
f,ax = plt.subplots(figsize=(15, 10))
sns.heatmap(retail.cov(),annot=True)
```

Out[25]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x21efe6d54c8>
```



Data Visualization & Analysis

```
In [26]: retail.shape
```

```
Out[26]: (9977, 11)
```

```
In [27]: retail.head()
```

Out[27]:

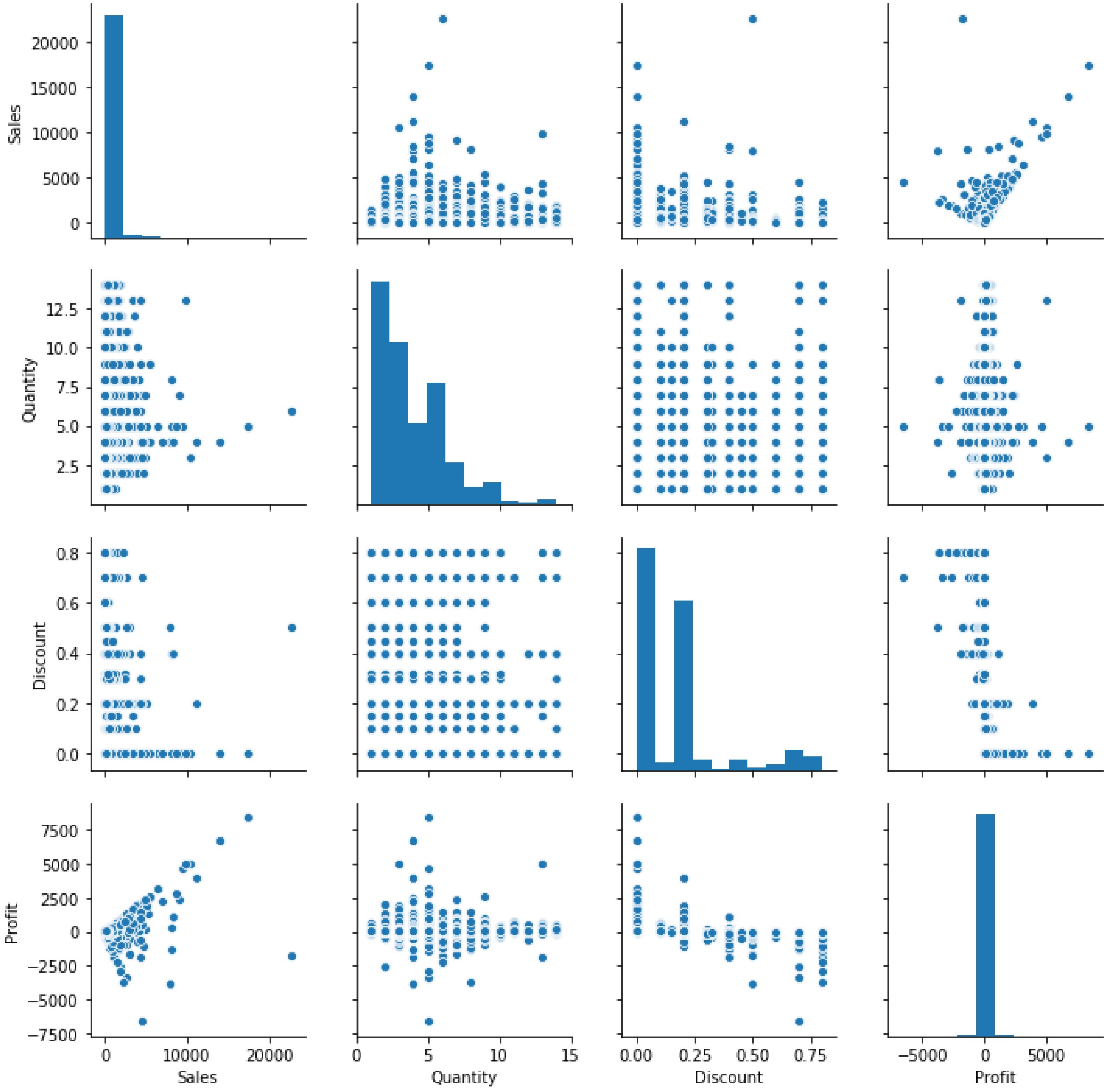
	Ship Mode	Segment	City	State	Region	Category	Sub-Category	Sales	Quantity	D

	Ship Mode	Segment	City	State	Region	Category	Sub-Category	Sales	Quantity	D
0	Second Class	Consumer	Henderson	Kentucky	South	Furniture	Bookcases	261.9600	2	
1	Second Class	Consumer	Henderson	Kentucky	South	Furniture	Chairs	731.9400	3	
2	Second Class	Corporate	Los Angeles	California	West	Office Supplies	Labels	14.6200	2	
3	Standard Class	Consumer	Fort Lauderdale	Florida	South	Furniture	Tables	957.5775	5	
4	Standard Class	Consumer	Fort Lauderdale	Florida	South	Office Supplies	Storage	22.3680	2	

Establishing the relationship between Sales, Quantity, Discount & Profit

```
In [28]: sns.pairplot(retail)
```

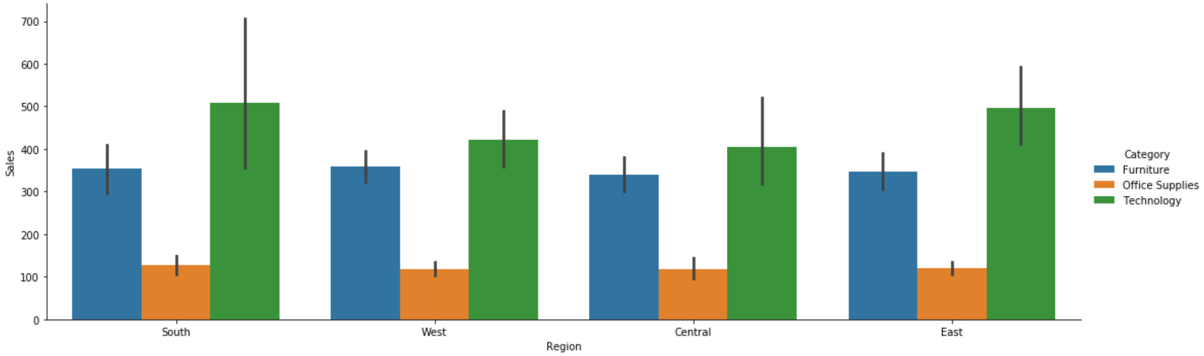
```
Out[28]: <seaborn.axisgrid.PairGrid at 0x21efe965f08>
```



Analyzing Sales

In [29]:
```
#category wise sales in different regions
plt.figure(figsize=[20,15])
ax = sns.catplot(x="Region", y="Sales", hue="Category", data=retail, kind='bar', aspect=3, height=5)
```

```
<Figure size 1440x1080 with 0 Axes>
```

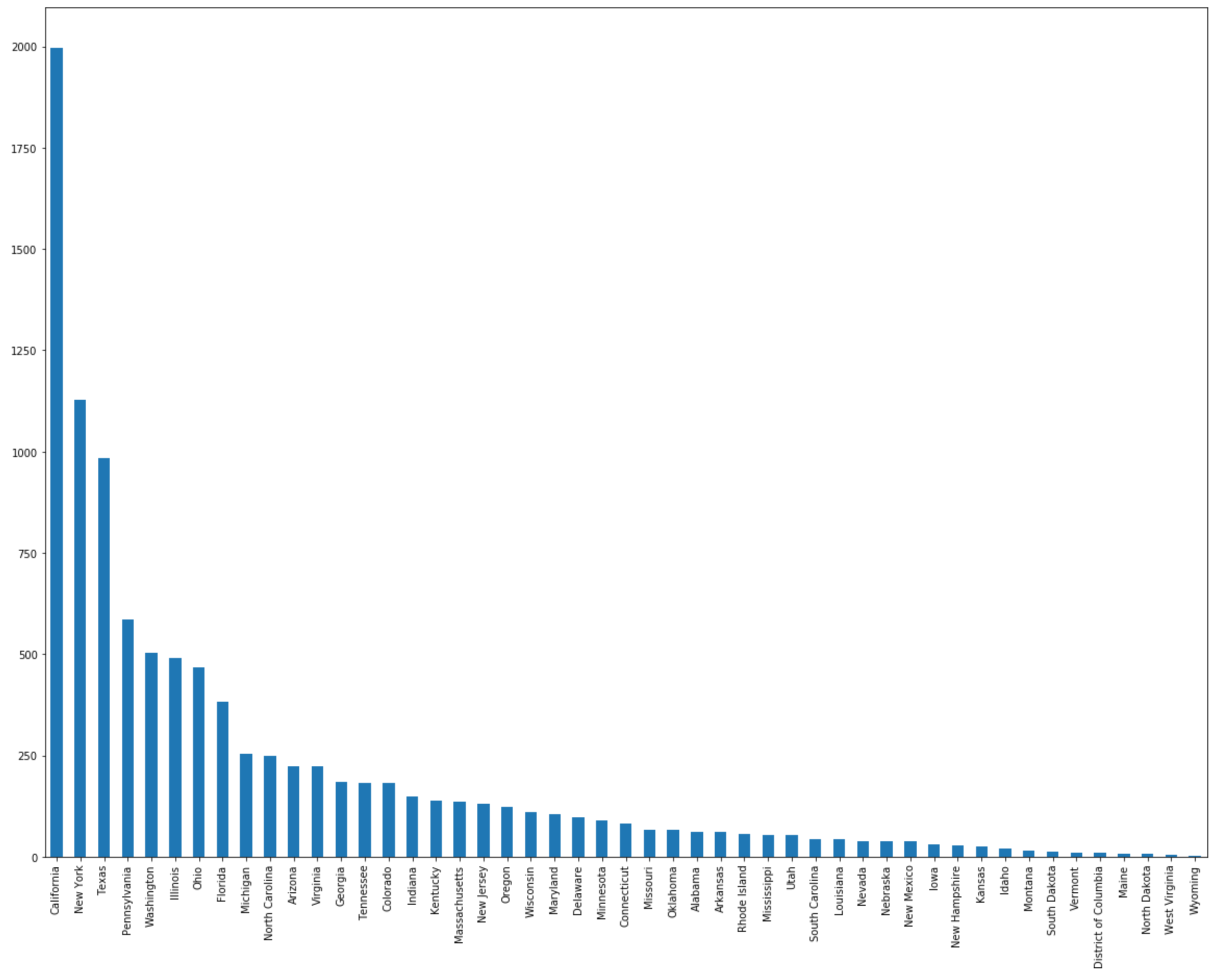


Analyzing Orders

Orders of different states in the USA

In [30]:
```
retail['State'].value_counts().plot(kind = 'bar', figsize=(20,15))
```

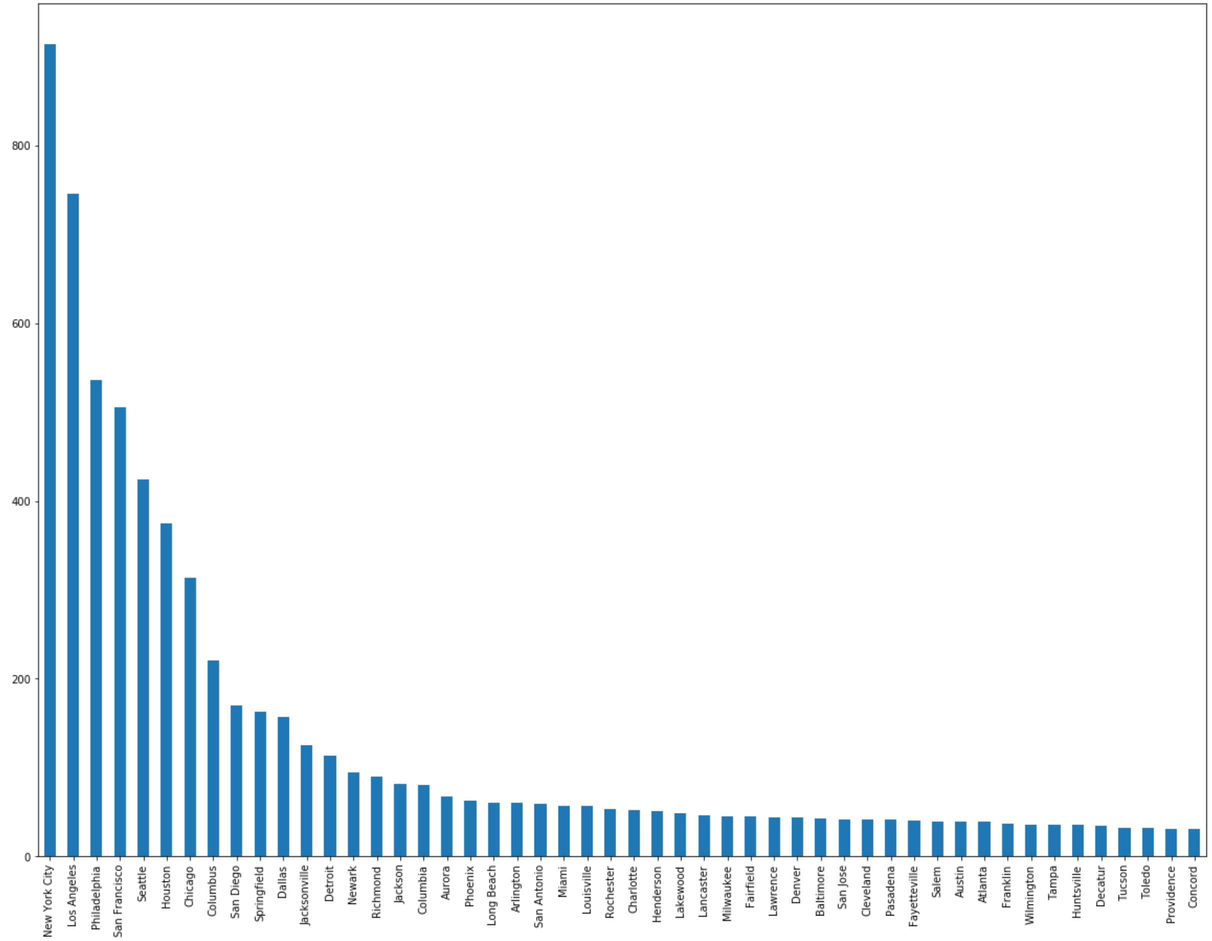
Out[30]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x21efdcd8748>
```



Top 50 cities with maximum orders

```
In [31]: retail['City'].value_counts().head(50).plot(kind = 'bar', figsize=(20,1
         5))
```

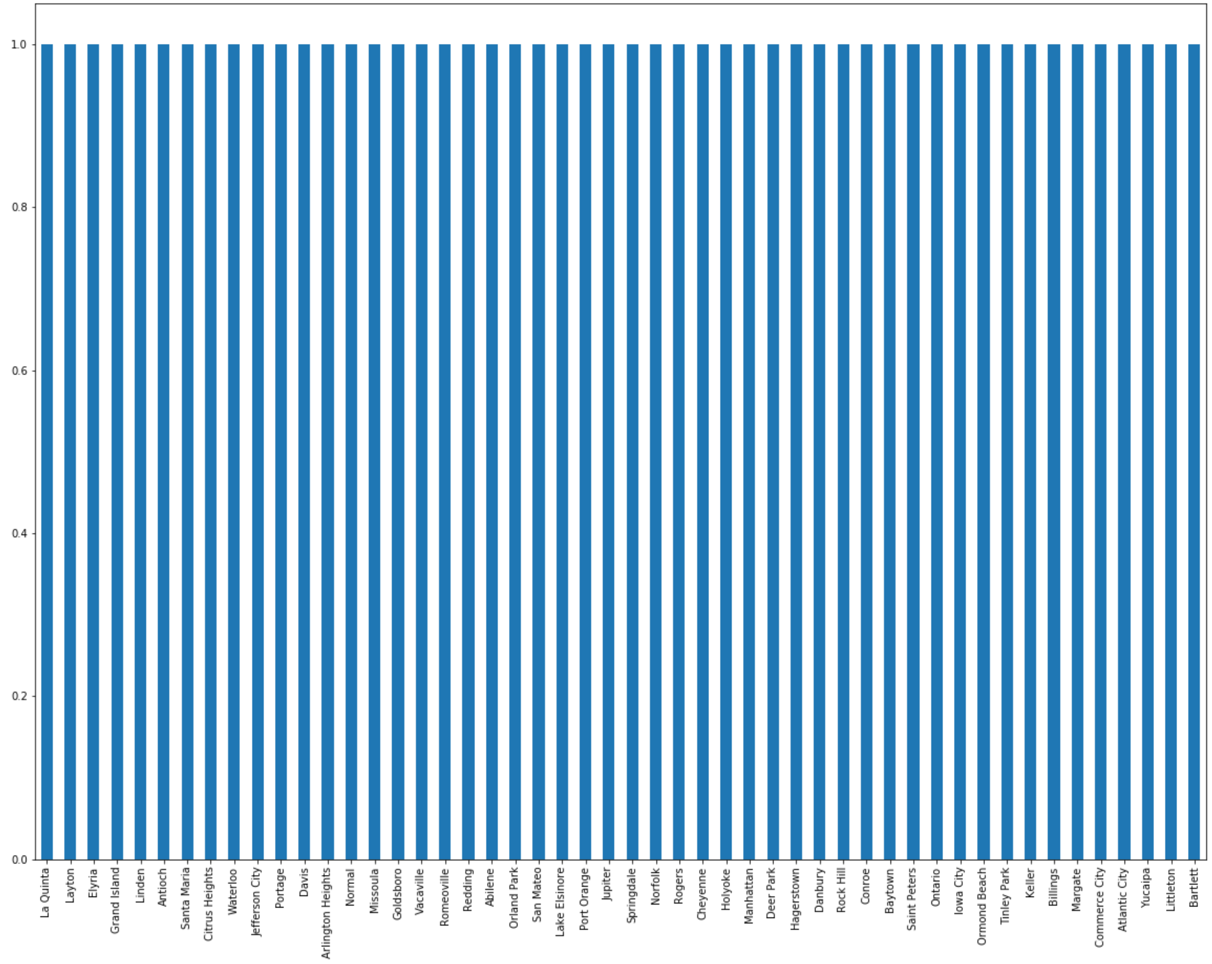
```
Out[31]: <matplotlib.axes._subplots.AxesSubplot at 0x21e81709248>
```



Bottom 50 cities with minimum orders

In [32]:
```
retail['City'].value_counts().tail(50).plot(kind = 'bar', figsize=(20,1
5))
```

Out[32]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x21e811df748>
```



Analyzing

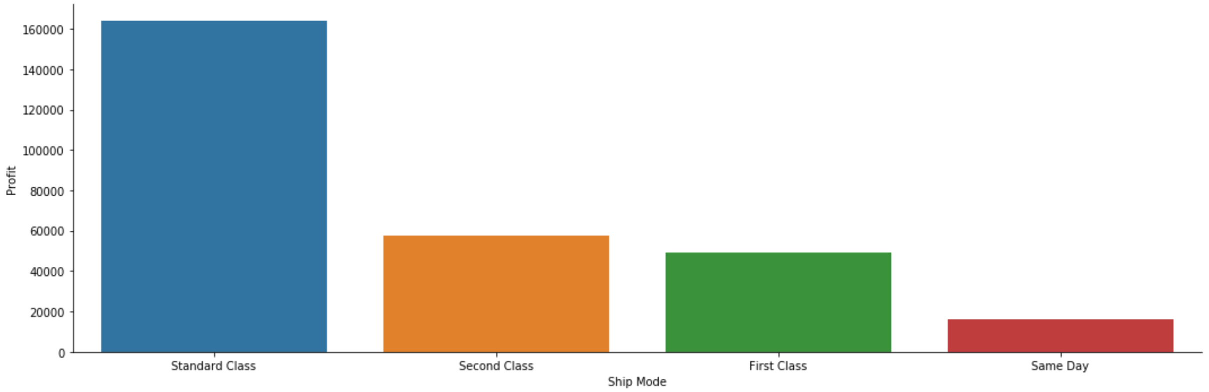
Profits by Shipmode

In [33]:
```
shipmode_profit=retail.groupby('Ship Mode')['Profit'].sum().reset_index
().sort_values(by='Profit', ascending=False)
```

```
sns.catplot('Ship Mode', 'Profit', data=shipmode_profit, kind='bar', as
pect=3, height=5)
```

Out[33]:

```
<seaborn.axisgrid.FacetGrid at 0x21e82405d48>
```



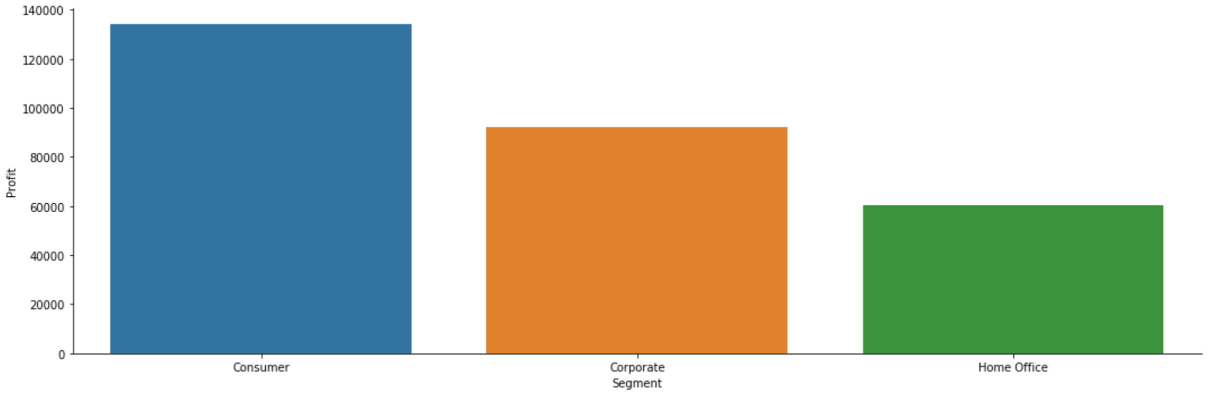
Profits by Segment

In [34]:

```
segment_profit=retail.groupby('Segment')['Profit'].sum().reset_index().
sort_values(by='Profit', ascending=False)
sns.catplot('Segment', 'Profit', data=segment_profit, kind='bar', aspec
t=3, height=5)
```

Out[34]:

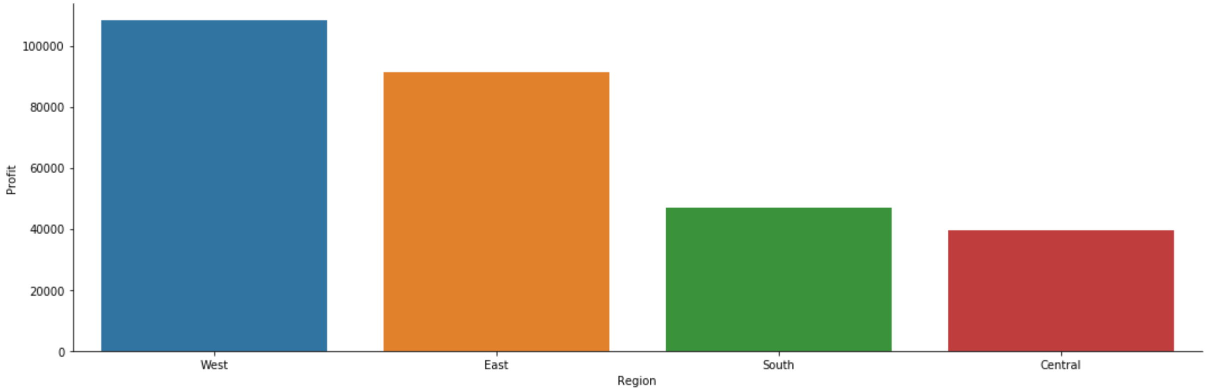
```
<seaborn.axisgrid.FacetGrid at 0x21e81dc9cc8>
```



Profits by Region

In [36]:
```
region_profit=retail.groupby('Region')['Profit'].sum().reset_index().so
rt_values(by='Profit', ascending=False)
sns.catplot('Region', 'Profit', data=region_profit, kind='bar', aspect=
3, height=5)
```

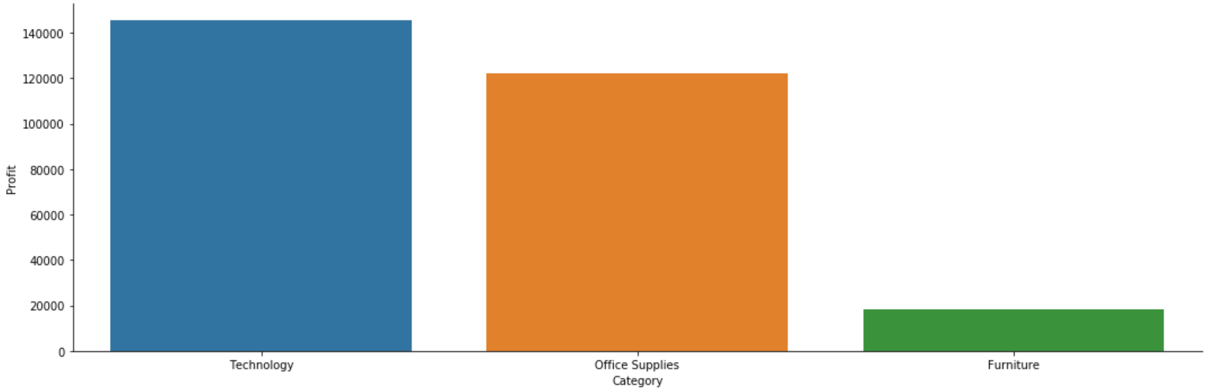
Out[36]:
```
<seaborn.axisgrid.FacetGrid at 0x21e82336ec8>
```



Profits by Category

In [37]:
```
category_profit=retail.groupby('Category')['Profit'].sum().reset_index
().sort_values(by='Profit', ascending=False)
sns.catplot('Category', 'Profit', data=category_profit, kind='bar', asp
ect=3, height=5)
```

Out[37]:
```
<seaborn.axisgrid.FacetGrid at 0x21e823362c8>
```



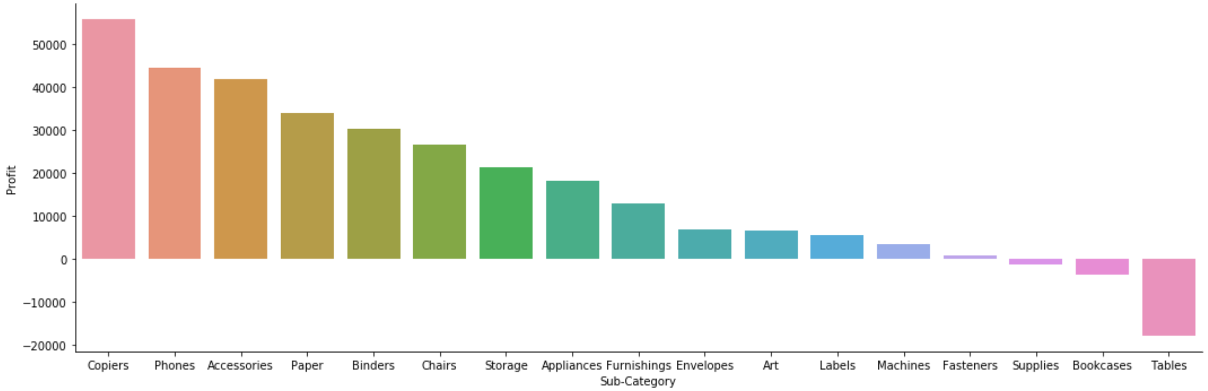
Profits by Sub-Categories

In [38]:

```
subcategory_profit=retail.groupby('Sub-Category')['Profit'].sum().reset_index().sort_values(by='Profit', ascending=False)
sns.catplot('Sub-Category', 'Profit', data=subcategory_profit, kind='bar', aspect=3, height=5)
```

Out[38]:

```
<seaborn.axisgrid.FacetGrid at 0x21e8232f1c8>
```



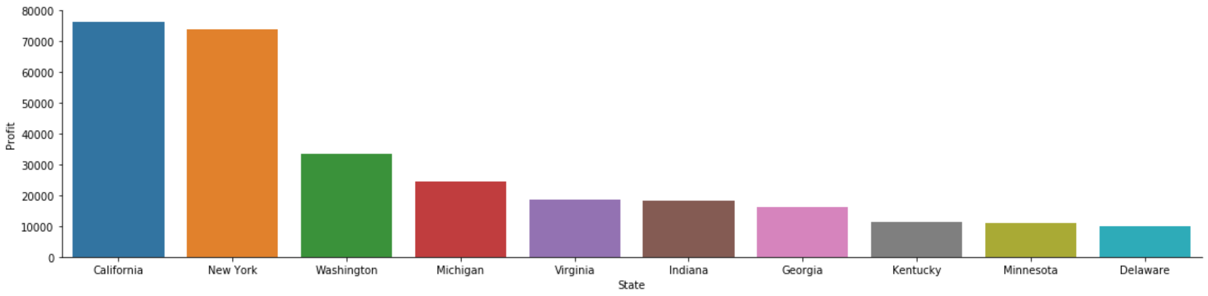
Profits of Top 10 States

In [39]:

```
states_profit=retail.groupby('State')['Profit'].sum().reset_index().sor
t_values(by='Profit', ascending=False)
top10_states_profit=states_profit.head(10)
sns.catplot('State', 'Profit', data=top10_states_profit, kind='bar', as
pect=4, height=4)
```

Out[39]:

```
<seaborn.axisgrid.FacetGrid at 0x21e832f7948>
```



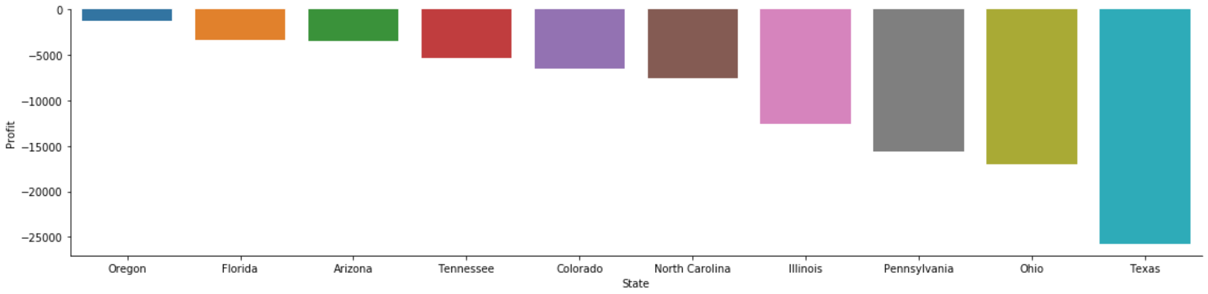
Profits of Bottom 10 States

In [40]:

```
states_profit=retail.groupby('State')['Profit'].sum().reset_index().sor
t_values(by='Profit', ascending=False)
bottom10_states_profit=states_profit.tail(10)
sns.catplot('State', 'Profit', data=bottom10_states_profit, kind='bar',
 aspect=4, height=4)
```

Out[40]:

```
<seaborn.axisgrid.FacetGrid at 0x21e83665c88>
```



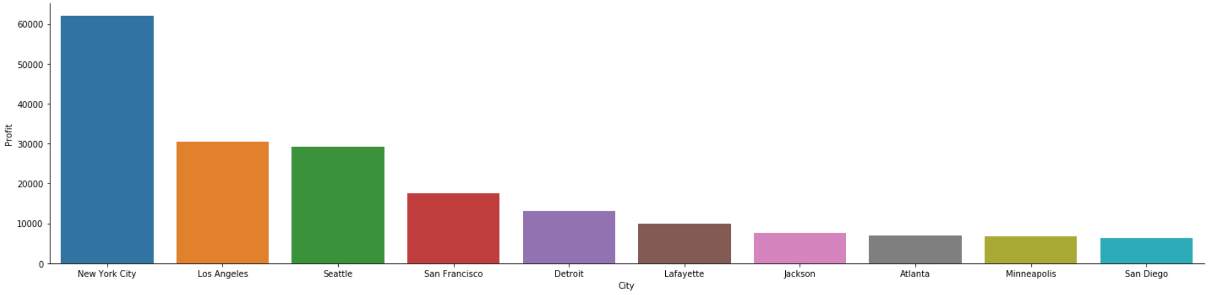
Profits of Top 10 Cities

In [41]:

```
city_profit=retail.groupby('City')['Profit'].sum().reset_index().sort_values(by='Profit', ascending=False)
top10_city_profit=city_profit.head(10)
sns.catplot('City', 'Profit', data=top10_city_profit, kind='bar', aspect=4, height=5)
```

Out[41]:

```
<seaborn.axisgrid.FacetGrid at 0x21e83665cc8>
```



Profits of Bottom 10 Cities

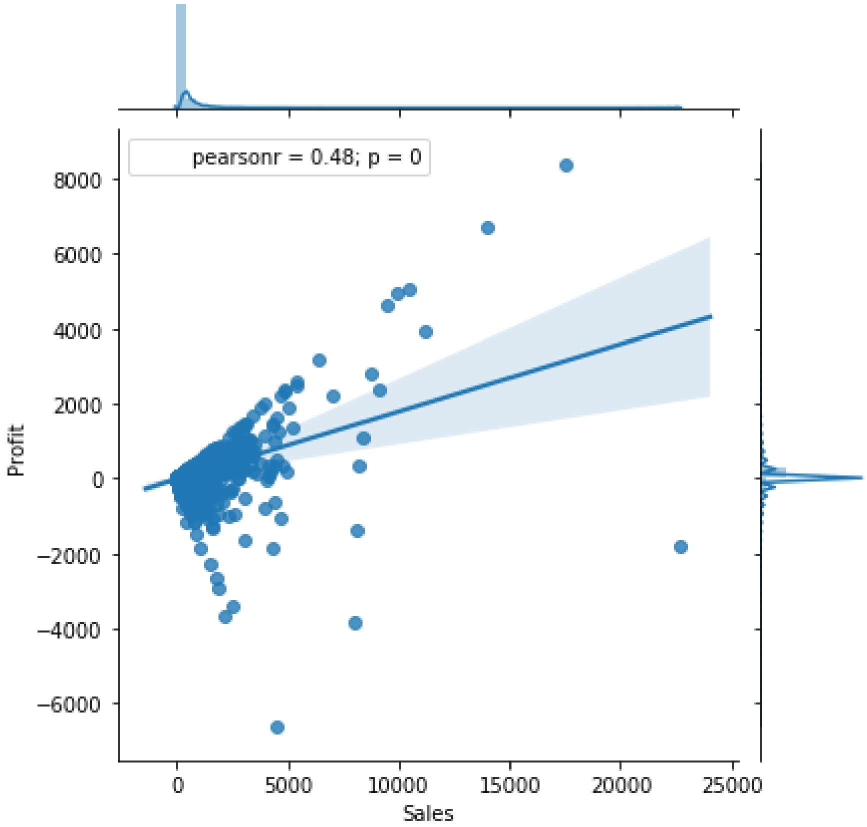
In [ ]:

```
city_profit=retail.groupby('City')['Profit'].sum().reset_index().sort_values(by='Profit', ascending=False)
bottom10_city_profit=city_profit.tail(10)
sns.catplot('City', 'Profit', data=bottom10_city_profit, kind='bar', aspect=4, height=5)
```

Sales vs Profit

In [42]:

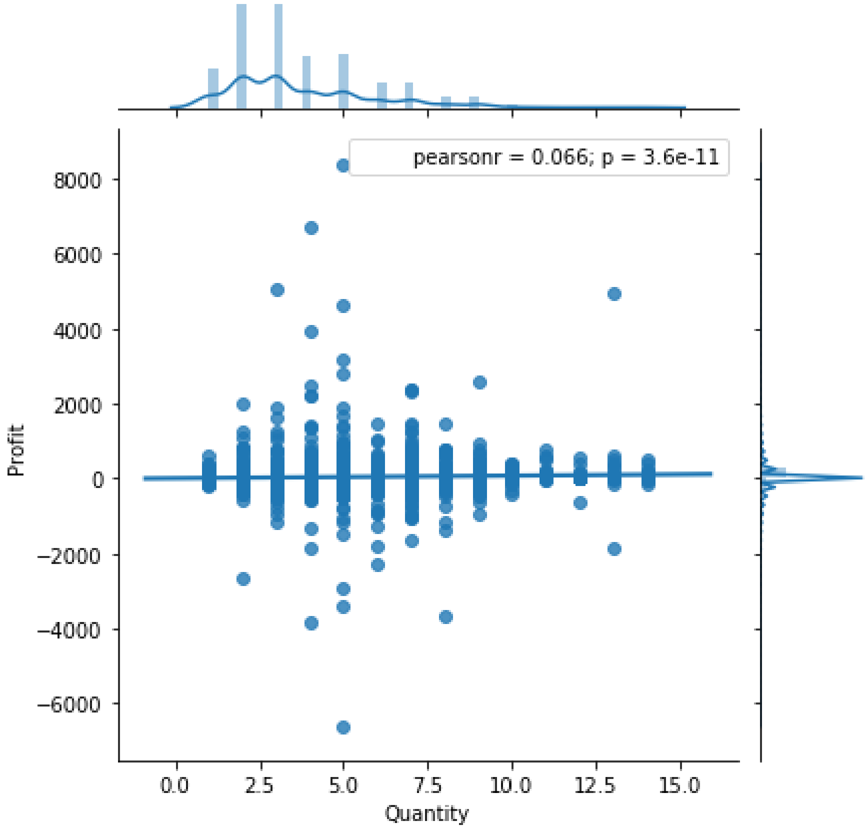
```
sns.jointplot(retail['Sales'], retail['Profit'], kind = "reg").annotate(stats.pearsonr)
plt.show()
```



Quantity vs Profit

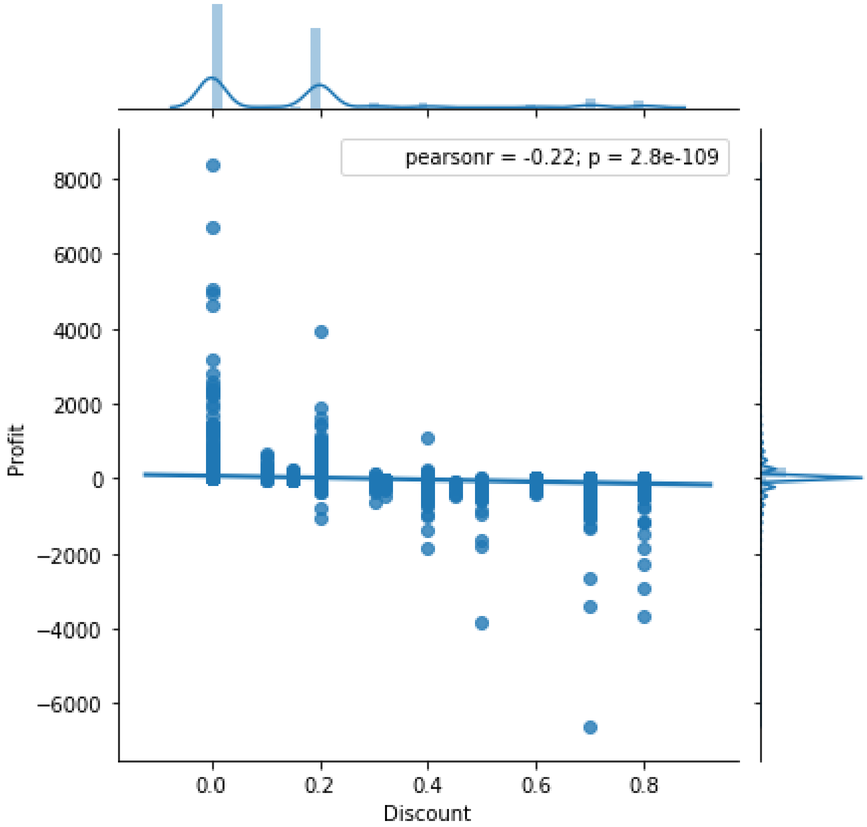
In [43]:

```
sns.jointplot(retail['Quantity'], retail['Profit'], kind = "reg").annot
ate(stats.pearsonr)
plt.show()
```



Discount vs Profit

```
In [44]: sns.jointplot(retail['Discount'], retail['Profit'], kind = "reg").annot
ate(stats.pearsonr)
plt.show()
```



Outcomes and Conclusion

Initially we have dropped the columns namely "Country" as entire dataset was of the United States only & "Postal Code" because we didn't find much need for keeping it.

It was observed that "Sales" positively affected the "Profit" to certain level while we did not find "Quantity" & "Discount" to be affecting the profit to maximum extent.

Sale of products were seen maximum in the "East" and "South" region of the country but Profit was seen maximum in "West" & "East" implying that although after selling maximum products in

Southern region the profit is minimum when compared to other region. Hence we need to rectify this issue and need to plan and exectue accordingly for geting the required profit.

"Standard Class" Shipmode generated maximum profits when compared to "Second Class" & "First Class". We have to look for generating moderate profit on the "Same Day".

Maximum profit was observed in "Technology" category while "Office Supplies" was not much behind while profit from "Furniture" category was really bad.

After checking out for profit in "Sub-Categories" we found out that "Tables" & "Bookcases" was showing negative as a result of which profit from "Furniture" category was the least.

It was observed that cities from where we received maximum orders, we also made significant amount of profit. For example, New York, Los Angeles, San Francisco, Seattle, San Diego etc.