Project 4- Financial Analytics

Objectives-

Without analyzing the competition, it is difficult for a business to survive. You are tasked to analyse the competition for the management to provide better results. This data set has information on the market capitalization of the top 500 companies in India. Serial NumberNameName of CompanyMar Cap – CroreMarket Capitalization in CroresSales Qtr – CroreQuarterly Sale in crores. Find key metrics and factors and show the meaningful relationships between attributes.

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

In [3]:
```
df_fin = pd.read_csv('Financial Analytics data.csv')

df_fin.head(3)
```

Out[3]:

	S.No.	Name	Mar Cap - Crore	Sales Qtr - Crore	Unnamed: 4
0	1	Reliance Inds.	583436.72	99810.00	NaN
1	2	TCS	563709.84	30904.00	NaN
2	3	HDFC Bank	482953.59	20581.27	NaN

In [5]:
```
df_fin.shape
```

Out[5]:
```
(488, 5)
```

In [35]:
```
df_fin.describe()
```

Out[35]:

	Mar Cap - Crore	Sales Qtr - Crore	Quaterly Sale- Crore
count	488.000000	488.000000	488.000000
mean	27708.961086	3610.168955	860.573443
std	58963.329098	9684.943920	850.831460
min	3017.070000	47.240000	0.000000
25%	4879.612500	725.732500	702.325000
50%	9885.050000	1278.300000	702.325000
75%	23400.815000	2084.097500	702.325000
max	583436.720000	110666.930000	7757.060000

Data Preprocessing/Cleaning

In [7]:
```
df_fin.isnull().any()
```

Out[7]:
```
S.No.                False
Name                 False
Mar Cap - Crore       True
Sales Qtr - Crore     True
Unnamed: 4            True
dtype: bool
```

In [13]:
```
print("The null values for Unnmaed coulmn are: ",df_fin['Unnamed: 4'].isnull().sum())

print("The null values for Mar cap column are: ",df_fin['Mar Cap - Crore'].isnull().su

print("The null values for Sales column are: ",df_fin['Sales Qtr - Crore'].isnull().su
```

```
The null values for Unnmaed coulmn are:  394
The null values for Mar cap column are:  9
The null values for Sales column are:  123
```

In [31]:
```
df_fin = df_fin.rename(columns={'Unnamed: 4': 'Quaterly Sale- Crore','Name' : 'Name of
```

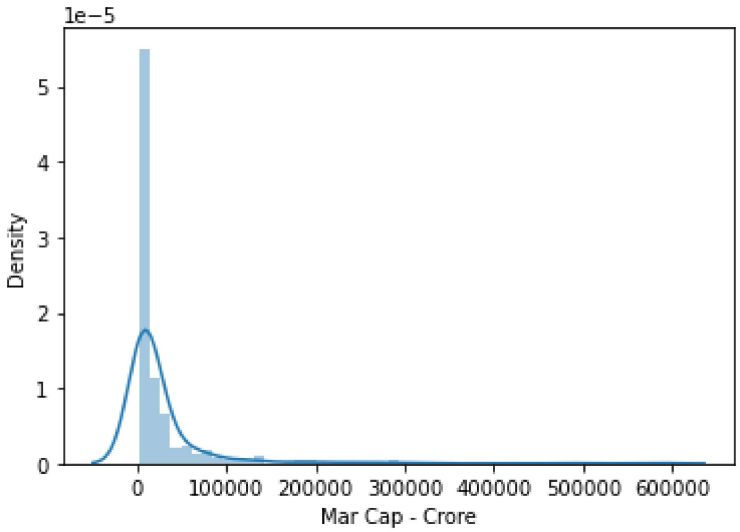
In [32]:
```
df_fin.columns
```

Out[32]:
```
Index(['Name of the Company', 'Mar Cap - Crore', 'Sales Qtr - Crore',
       'Quaterly Sale- Crore'],
      dtype='object')
```

In [18]:
```
sns.distplot(df_fin['Mar Cap - Crore'])
```

```
D:\Anaconda\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot
` is a deprecated function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar flexibility) or `h
istplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

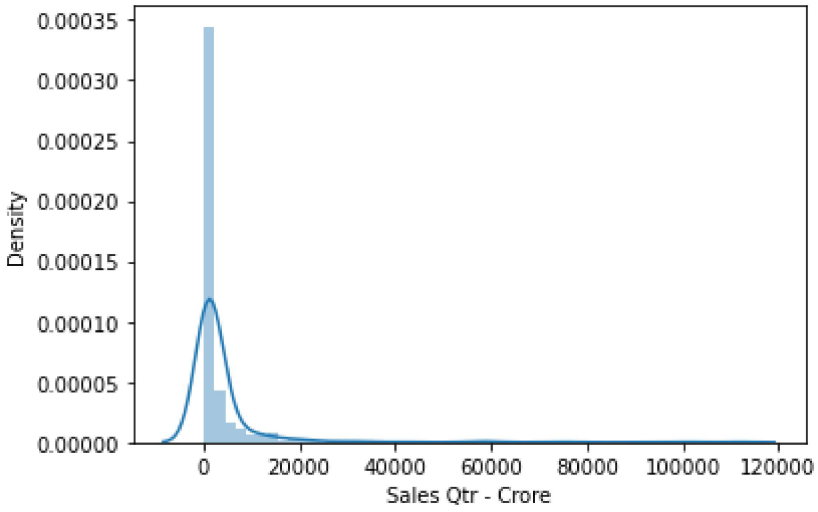
Out[18]:
```
<AxesSubplot:xlabel='Mar Cap - Crore', ylabel='Density'>
```



In [21]:
```
sns.distplot(df_fin['Sales Qtr - Crore'])
```

```
D:\Anaconda\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

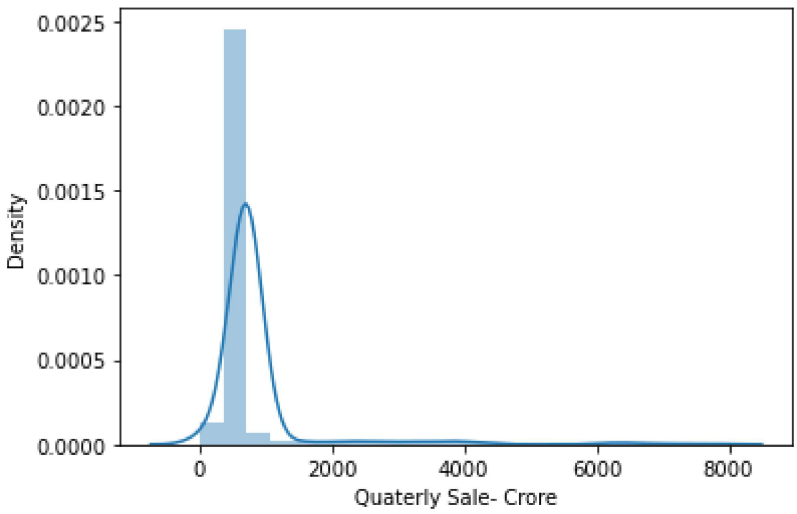
Out[21]:
```
<AxesSubplot:xlabel='Sales Qtr - Crore', ylabel='Density'>
```



In [22]:
```
sns.distplot(df_fin['Quaterly Sale- Crore'])
```

```
D:\Anaconda\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[22]:
```
<AxesSubplot:xlabel='Quaterly Sale- Crore', ylabel='Density'>
```



In [19]:
```
df_fin.fillna(df_fin.median(),inplace = True)
```

```
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\1659377371.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  df_fin.fillna(df_fin.median(),inplace = True)
```

In [20]:
```
df_fin.isnull().sum()
```

Out[20]:
```
S.No.                  0
Name                   0
Mar Cap - Crore        0
Sales Qtr - Crore      0
Quaterly Sale- Crore   0
dtype: int64
```

In [52]:
```
df_fin.head(3)
```

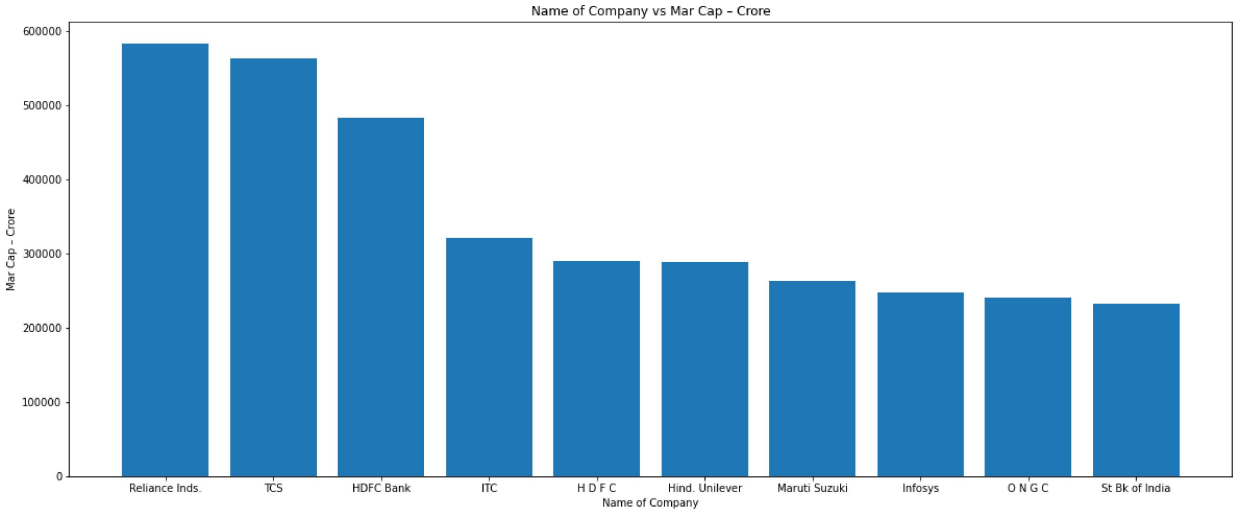
Out[52]:

S.No.	Name of the Company	Mar Cap - Crore	Sales Qtr - Crore	Quaterly Sale- Crore
1	Reliance Inds.	583436.72	99810.00	702.325
2	TCS	563709.84	30904.00	702.325
3	HDFC Bank	482953.59	20581.27	702.325

Data Visualization between Name of the company & Mar Cap- Crore and Name of the Company & Sales Qtr- Crore of top 10 Companies.

In [68]:
```
df= df_fin.sort_values(by='Mar Cap - Crore', ascending=False)
df_top10 = df.head(10)
```

In [69]:
```
plt.figure(figsize=(20,8))
plt.bar(df_top10['Name of the Company'], df_top10['Mar Cap - Crore'])
plt.xlabel('Name of Company')
plt.ylabel('Mar Cap - Crore')
plt.title('Name of Company vs Mar Cap - Crore')
plt.show()
```

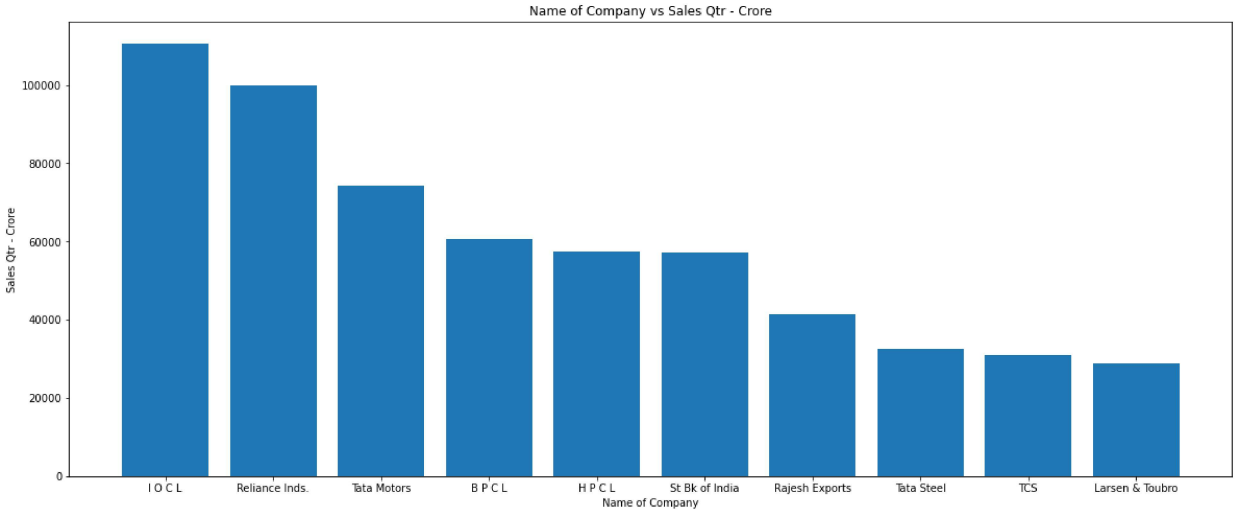


In [71]:
```
sales= df_fin.sort_values(by='Sales Qtr - Crore', ascending=False)
sales_top10 = sales.head(10)
sales_top10
```

Out[71]:

S.No.	Name of the Company	Mar Cap - Crore	Sales Qtr - Crore	Quaterly Sale- Crore
15	I O C L	178017.48	110666.93	702.325
1	Reliance Inds.	583436.72	99810.00	702.325
24	Tata Motors	117071.87	74156.07	702.325
28	B P C L	98278.00	60616.36	702.325
55	H P C L	58034.78	57474.25	702.325
10	St Bk of India	232763.33	57014.08	702.325
123	Rajesh Exports	23495.54	41304.84	702.325
41	Tata Steel	73376.14	32464.14	702.325
2	TCS	563709.84	30904.00	702.325
14	Larsen & Toubro	180860.74	28747.45	702.325

In [72]:
```
plt.figure(figsize=(20,8))
plt.bar(sales_top10['Name of the Company'], sales_top10['Sales Qtr - Crore'])
plt.xlabel('Name of Company')
plt.ylabel('Sales Qtr - Crore')
plt.title('Name of Company vs Sales Qtr - Crore')
plt.show()
```



In [73]:
```
## Finding the correlating between Mar Cap - Crore & Sales Qtr - Crore using the pears

# Calculate the correlation between Mar Cap - Crore and Sales Qtr - Crore
correlation = df_fin['Mar Cap - Crore'].corr(df_fin['Sales Qtr - Crore'])

print("The correlation between Mar Cap - Crore and Sales Qtr - Crore is:", correlatior
```

```
The correlation between Mar Cap - Crore and Sales Qtr - Crore is: 0.6204889718974049
```

In [74]:
```
# Calculate the correlation between Mar Cap - Crore and Sales Qtr - Crore using spearm
correlation_spearman = df['Mar Cap - Crore'].corr(df['Sales Qtr - Crore'], method='spe

print("The correlation between Mar Cap - Crore and Sales Qtr - Crore using spearman me
```

```
The correlation between Mar Cap - Crore and Sales Qtr - Crore using spearman method i
s: 0.5560465291681288
```

Analysis from the correlation can be -

The correlation coefficient between "Mar Cap - Crore" and "Sales Qtr - Crore" is 0.6204889718974049. This indicates a moderate positive correlation between the two variables. This means that as the market capitalization of a company increases, its quarterly sales also tend to increase. This makes sense as companies with a higher market capitalization likely have more resources to invest in sales and marketing efforts. The correlation using the spearman method also indicates the same i.e. this also indicates a moderate positive correlation between the two variables.

Company wise analysis: Breaking down the data by company and analyzing the performance of individual companies in terms of Market Capitalization and Sales.

In [155…

```python
def count_first_name(df, col_name):
    first_name_count = {}
    for name in df[col_name]:
        first_name = name.split()[0]
        if first_name in first_name_count:
            first_name_count[first_name] += 1
        else:
            first_name_count[first_name] = 1
    return first_name_count

count_first_name(df_fin, 'Name of the Company')
```

Out[155]:
```
{'Reliance': 6,
 'TCS': 1,
 'HDFC': 2,
 'ITC': 1,
 'H': 4,
 'Hind.': 1,
 'Maruti': 1,
 'Infosys': 1,
 'O': 1,
 'St': 2,
 'ICICI': 3,
 'Kotak': 1,
 'Coal': 1,
 'Larsen': 1,
 'I': 4,
 'Bharti': 2,
 'Axis': 1,
 'NTPC': 1,
 'Sun': 2,
 'Hind.Zinc': 1,
 'Wipro': 1,
 'HCL': 1,
 'Vedanta': 1,
 'Tata': 9,
 'UltraTech': 1,
 'Asian': 1,
 'Power': 2,
 'B': 2,
 'IndusInd': 1,
 'Bajaj': 6,
 'M': 3,
 'Adani': 4,
 'GAIL': 1,
 'Avenue': 1,
 'Titan': 1,
 'JSW': 2,
 'Grasim': 1,
 'Eicher': 1,
 'Nestle': 1,
 'Godrej': 4,
 'Yes': 1,
 'Hero': 1,
 'Motherson': 1,
 'SBI': 1,
 'General': 1,
 'Dabur': 1,
 'Bosch': 1,
 'Shree': 1,
 'New': 1,
 'Britannia': 1,
 'Tech': 1,
 'Hindalco': 1,
 'Zee': 1,
 'Cairn': 1,
 'Indiabulls': 2,
 'Ambuja': 1,
 'Interglobe': 1,
 'Cipla': 1,
 'Piramal': 1,
 'United': 2,
```

```
 'Pidilite': 1,
 'Siemens': 1,
 'Cadila': 1,
 'NMDC': 1,
 'DLF': 1,
 'Marico': 1,
 'Ashok': 1,
 'Bharat': 2,
 'Lupin': 1,
 'Petronet': 1,
 'Aditya': 3,
 'Dr': 2,
 'S': 4,
 'UPL': 1,
 'Oracle': 1,
 'Biocon': 1,
 'Aurobindo': 1,
 'Bank': 2,
 'Idea': 1,
 'A': 1,
 'Havells': 1,
 'Container': 1,
 'TVS': 1,
 'ACC': 1,
 'P': 2,
 'MRF': 1,
 'Shriram': 2,
 'Colgate-Palm.': 1,
 'L&T': 2,
 'Punjab': 1,
 'NHPC': 1,
 'Oil': 1,
 'Rural': 1,
 'GlaxoSmith': 1,
 "Divi's": 1,
 'Kansai': 1,
 'Alkem': 1,
 'LIC': 1,
 'Future': 3,
 'Page': 1,
 'Dalmia': 1,
 'IIFL': 1,
 'L': 2,
 'Emami': 1,
 'Cummins': 1,
 'Berger': 1,
 'Rajesh': 1,
 '3M': 1,
 'Jindal': 4,
 'Edelweiss.Fin.': 1,
 'Gillette': 1,
 'Balkrishna': 1,
 'Cholaman.Inv.&Fn': 1,
 'Indraprastha': 1,
 'Glaxosmi.': 1,
 'PNB': 1,
 'RBL': 1,
 'Castrol': 1,
 'Canara': 1,
 'GRUH': 1,
```

```
'KIOCL': 1,
'Voltas': 1,
'Whirlpool': 1,
'Sundaram': 2,
'Federal': 1,
'Endurance': 1,
'Exide': 1,
'IDFC': 1,
'NBCC': 1,
'IDBI': 1,
'The': 1,
'Dewan': 1,
'MphasiS': 1,
'Apollo': 1,
'AU': 1,
'Indian': 3,
'Motil.Oswal.Fin.': 1,
'Oberoi': 1,
'SJVN': 1,
'Supreme': 2,
'Muthoot': 1,
'NLC': 1,
'Info': 1,
'Glenmark': 1,
'Jubilant': 2,
'Crompton': 1,
'Honeywell': 1,
'Natco': 1,
'PC': 1,
'Quess': 1,
'CRISIL': 1,
'WABCO': 1,
'Amara': 1,
'Sterlite': 1,
'AIA': 1,
'KRBL': 1,
'Max': 1,
'Century': 2,
'Bayer': 1,
'Graphite': 1,
'Central': 2,
'Natl.': 1,
'CESC': 1,
'Shri.City': 1,
'Rain': 1,
'Torrent': 1,
'Dilip': 1,
'TI': 1,
'JM': 1,
'Syngene': 1,
'Abbott': 1,
'Hatsun': 1,
'Symphony': 1,
'Gujarat': 1,
'Ajanta': 1,
'City': 1,
'Varun': 1,
'Mindtree': 1,
'Prestige': 1,
'Sundram': 1,
```

```
'Sanofi': 1,
'Guj.St.Petronet': 1,
'Finolex': 2,
'Bombay': 2,
'SRF': 1,
'GE': 3,
'Alembic': 1,
'SPARC': 1,
'GMR': 1,
'HEG': 1,
'Trent': 1,
'Engineers': 1,
'Avanti': 1,
'Pfizer': 1,
'Escorts': 1,
'Blue': 2,
'Indbull.RealEst.': 1,
'ERIS': 1,
'Arvind': 1,
'Hexaware': 1,
'Mahanagar': 1,
'SKF': 1,
'Delta': 1,
'Union': 1,
'TV18': 1,
'Minda': 2,
'Solar': 1,
'Kajaria': 1,
'Astral': 1,
'Bata': 1,
'Phoenix': 1,
'BASF': 1,
'DCM': 1,
'Infibeam': 1,
'Aegis': 1,
'Mahindra': 3,
'Jet': 1,
'SpiceJet': 1,
'Thomas': 1,
'Guj': 3,
'Wockhardt': 1,
'Akzo': 1,
'Security': 1,
'Asahi': 1,
'TTK': 1,
'ITI': 1,
'Karur': 1,
'Vardhman': 1,
'Fortis': 1,
'Ipca': 1,
'Sheela': 1,
'IRB': 1,
'Atul': 1,
'Dish': 1,
'Rel.': 1,
'OCL': 1,
'NCC': 1,
'Relaxo': 1,
'J': 2,
'G': 3,
```

```
'Cochin': 1,
'Birla': 1,
'Kalpataru': 1,
'Suzlon': 1,
'Cyient': 1,
'Hind.Copper': 1,
'Carborundum': 1,
'Can': 1,
'Sadbhav': 2,
'Advanta': 1,
'Capital': 1,
'Lak.': 2,
'Himadri': 1,
'Coffee': 1,
'PVR': 1,
'Chambal': 1,
'Vijaya': 1,
'Welspun': 2,
'CEAT': 1,
'Strides': 1,
'Narayana': 1,
'Jyothy': 1,
'Johnson': 1,
'Prism': 1,
'Syndicate': 1,
'D': 1,
'Jain': 1,
'Persistent': 1,
'Redington': 1,
'Sunteck': 1,
'Raymond': 1,
'MOIL': 1,
'Grindwell': 1,
'EID': 1,
'Galaxy': 1,
'Laurus': 1,
'Timken': 1,
'C': 1,
'Dishman': 2,
'ISGEC': 1,
'MMTC': 1,
'IFB': 1,
'eClerx': 1,
'Sobha': 1,
'Kirloskar': 1,
'CG': 1,
'Westlife': 1,
'K': 1,
'Tube': 1,
'VST': 1,
'BEML': 1,
'FDC': 1,
'DCB': 1,
'Star': 2,
'Netwrk.18': 1,
'Gulf': 1,
'UCO': 1,
'Jagran': 1,
'Elgi': 1,
'JK': 2,
```

```
'Zydus': 1,
'Equitas': 1,
'India': 1,
'South': 1,
'Polaris': 1,
'V': 1,
'APL': 1,
'Swan': 1,
'NIIT': 1,
'Caplin': 1,
'Shoppers': 1,
'Godfrey': 1,
'R': 1,
'Rallis': 1,
'Stand.Chart.PLC': 1,
'Manpasand': 1,
'Essel': 1,
'Allcargo': 1,
'Radico': 1,
'Cera': 1,
'BSE': 1,
'Forbes': 1,
'KNR': 1,
'PNC': 1,
'Greenply': 1,
'Ujjivan': 1,
'Monsanto': 1,
'Vinati': 1,
'Lux': 1,
'Linde': 1,
'Ratnamani': 1,
'Cox': 1,
'Omaxe': 1,
'Ashoka': 1,
'Time': 1,
'Phillips': 1,
'Allahabad': 1,
'NESCO': 1,
'CARE': 1,
'JP': 2,
'Andhra': 1,
'Zensar': 1,
'Sintex': 1,
'SREI': 1,
'Techno': 1,
'HMT': 1,
'KPIT': 1,
'Triveni': 1,
'Shankara': 1,
'Multi': 1,
'Brigade': 1,
'Gayatri': 1,
'Magma': 1,
'VRL': 1,
'ICRA': 1,
'IFCI': 1,
'Suprajit': 1,
'Navin': 1,
'Karnataka': 1,
'Shilpa': 1,
```

```
 'Kushal': 1,
 "Venky's": 1,
 'Force': 1,
 'CCL': 1,
 'Excel': 1,
 'Trident': 1,
 'Corporation': 1,
 'Rane': 1,
 'Team': 1,
 'Oriental': 1,
 'Deepak': 2,
 'Heidelberg': 1,
 'Amber': 1,
 'Sharda': 1,
 'Dixon': 1,
 'Himatsing.': 1,
 'La': 1,
 'Rupa': 1,
 'Hind.Construct.': 1,
 'Ent.Network': 1,
 'MAS': 1,
 'Thyrocare': 1,
 'Prakash': 1,
 'Repco': 1,
 'Sonata': 1,
 'Puravankara': 1,
 'Tejas': 1,
 'ITD': 1,
 'Hathway': 1,
 'Dhanuka': 1,
 'Heritage': 1,
 'Mah.': 1,
 'Navneet': 1,
 'Firstsour.Solu.': 1,
 'Kaveri': 1,
 'Va': 1,
 'Prime': 1,
 'NOCIL': 1,
 'Orient': 1,
 'Natl.Fertilizer': 1}
```

In [130…

```
#Filter the rows that contain the keyword 'Tata' in the 'Name of the Company' column
tata_data = df_fin[df_fin['Name of the Company'].str.contains('Tata')]
reliance_data = df_fin[df_fin['Name of the Company'].str.contains('Reliance')]
```

In [137…

```
Tata_industries = tata_data.mean()
Tata_industries
```

```
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\2038304439.py:1: FutureWarning: Drop
ping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprec
ated; in a future version this will raise TypeError.  Select only valid columns befor
e calling the reduction.
  Tata_industries = tata_data.mean()
```

Out[137]:

```
Mar Cap - Crore         31923.200000
Sales Qtr - Crore       14598.062222
Quaterly Sale- Crore      629.513333
dtype: float64
```

In [139…
```
Reliance_industries=reliance_data.mean()
Reliance_industries
```

```
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\1858897557.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  Reliance_industries=reliance_data.mean()
```

Out[139]:
```
Mar Cap - Crore         106550.160000
Sales Qtr - Crore        18951.948333
Quaterly Sale- Crore       702.325000
dtype: float64
```

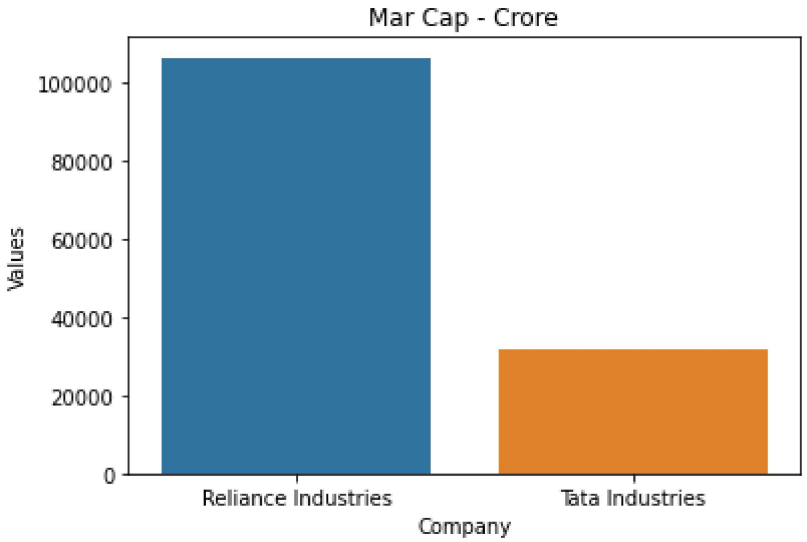
In [144…
```
# Create a dataframe from the provided data
industries = {'Company': ['Reliance Industries', 'Tata Industries'],
        'Mar Cap - Crore': [106550.16, 31923.2],
        'Sales Qtr - Crore': [18951.948333, 14598.062222],
        'Quaterly Sale- Crore' : [629.513333,702.325000]
        }
df = pd.DataFrame(industries, columns = ['Company', 'Mar Cap - Crore','Sales Qtr - Cr
df
```

Out[144]:

	Company	Mar Cap - Crore	Sales Qtr - Crore	Quaterly Sale- Crore
0	Reliance Industries	106550.16	18951.948333	629.513333
1	Tata Industries	31923.20	14598.062222	702.325000

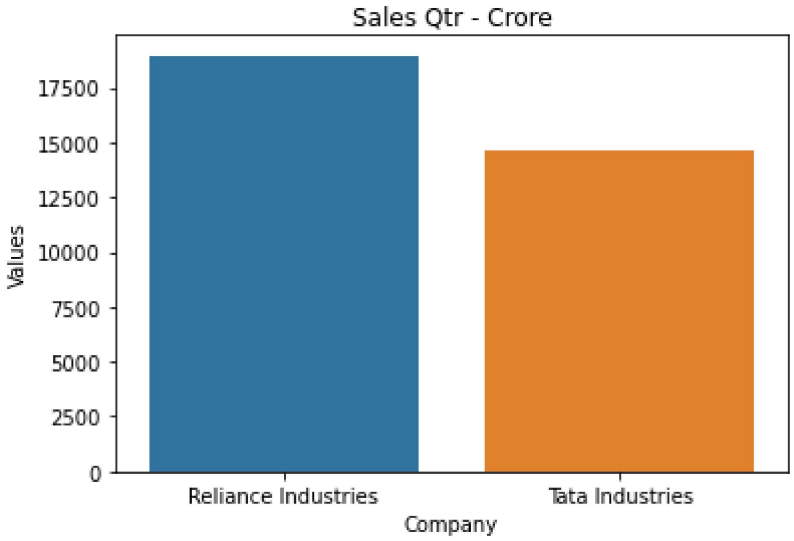
In [153…
```
# Create the bar chart
sns.barplot(x="Company", y="Mar Cap - Crore", data=df)

plt.xlabel('Company')
plt.ylabel('Values')
plt.title('Mar Cap - Crore')
plt.show()
```



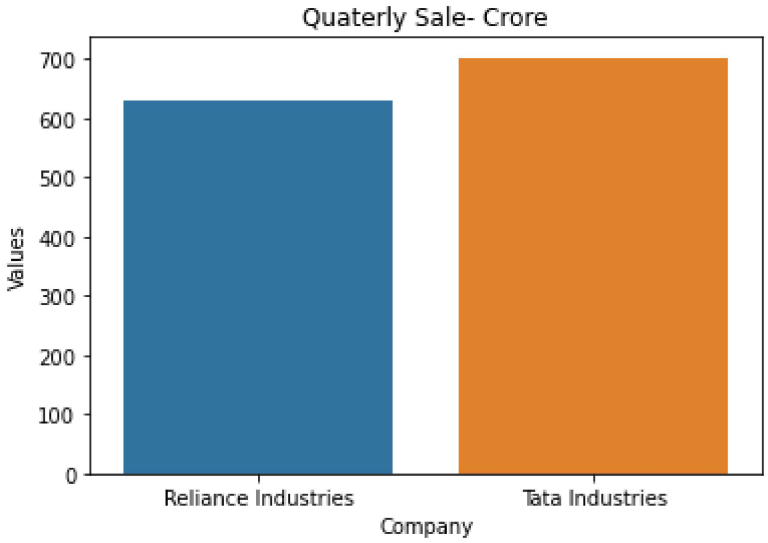
In [151…
```
sns.barplot(x="Company", y="Sales Qtr - Crore", data=df)
plt.xlabel('Company')
plt.ylabel('Values')
```

```
plt.title('Sales Qtr - Crore')
plt.show()
```



In [154…

```
sns.barplot(x="Company", y="Quaterly Sale- Crore", data=df)
plt.xlabel('Company')
plt.ylabel('Values')
plt.title('Quaterly Sale- Crore')
plt.show()
```



Comparison between the industries to identify the potential opportunties for growth.

In [179…

```
Bank = df_fin[df_fin['Name of the Company'].str.contains('Bank')]['Name of the Company
Automotive = df_fin[df_fin['Name of the Company'].str.contains('Motor')]['Name of the
Power= df_fin[df_fin['Name of the Company'].str.contains('Power')]['Name of the Compar
Steel = df_fin[df_fin['Name of the Company'].str.contains('Steel')]['Name of the Compa
Gas = df_fin[df_fin['Name of the Company'].str.contains('Gas')]['Name of the Company']
Pharma = df_fin[df_fin['Name of the Company'].str.contains('Pharma')]['Name of the Cor
Technology = df_fin[df_fin['Name of the Company'].str.contains('Tech')]['Name of the C
print("The number of Banks are: ",Bank)
print("The number of Automotive companies are:",Automotive)
print("The number of Power companies are:",Power)
```

```
print("The number of Steel companies are:",Steel)
print("The number of Gas companies are:",Gas)
print("The number of Pharmaceutical companies are:",Pharma)
print("The number of Tech companies are:",Technology)
```

```
The number of Banks are:  31
The number of Automotive companies are: 5
The number of Power companies are: 10
The number of Steel companies are: 3
The number of Gas companies are: 3
The number of Pharmaceutical companies are: 8
The number of Tech companies are: 15
```

In [181…

```
Bank_data= df_fin[df_fin['Name of the Company'].str.contains('Bank')]
Automotive_data = df_fin[df_fin['Name of the Company'].str.contains('Motor')]
Power_data = df_fin[df_fin['Name of the Company'].str.contains('Power')]
Steel_data = df_fin[df_fin['Name of the Company'].str.contains('Steel')]
Gas_data = df_fin[df_fin['Name of the Company'].str.contains('Gas')]
Pharma_data = df_fin[df_fin['Name of the Company'].str.contains('Pharma')]
Technology_data = df_fin[df_fin['Name of the Company'].str.contains('Tech')]
```

In [183…

```
## Getting the mean of all the industries.
Bank_data.mean()
```

```
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\3679764377.py:2: FutureWarning: Drop
ping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprec
ated; in a future version this will raise TypeError.  Select only valid columns befor
e calling the reduction.
  Bank_data.mean()
```

Out[183]:

```
Mar Cap - Crore         47457.725806
Sales Qtr - Crore        5605.658710
Quaterly Sale- Crore      707.584194
dtype: float64
```

In [184…

```
Automotive_data.mean()
```

```
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\3758817326.py:1: FutureWarning: Drop
ping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprec
ated; in a future version this will raise TypeError.  Select only valid columns befor
e calling the reduction.
  Automotive_data.mean()
```

Out[184]:

```
Mar Cap - Crore         48325.986
Sales Qtr - Crore       17350.270
Quaterly Sale- Crore     1298.850
dtype: float64
```

In [185…

```
Power_data.mean()
```

```
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\4046150404.py:1: FutureWarning: Drop
ping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprec
ated; in a future version this will raise TypeError.  Select only valid columns befor
e calling the reduction.
  Power_data.mean()
```

Out[185]:

```
Mar Cap - Crore         21121.285
Sales Qtr - Crore        3623.543
Quaterly Sale- Crore      702.325
dtype: float64
```

In [186…

```
Steel_data.mean()
```

```
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\1894152454.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  Steel_data.mean()
```

Out[186]:
```
Mar Cap - Crore         56780.263333
Sales Qtr - Crore       19105.900000
Quaterly Sale- Crore      702.325000
dtype: float64
```

In [187…
```
Gas_data.mean()
```

```
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\85756220.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  Gas_data.mean()
```

Out[187]:
```
Mar Cap - Crore         14141.446667
Sales Qtr - Crore        1344.510000
Quaterly Sale- Crore      702.325000
dtype: float64
```

In [188…
```
Pharma_data.mean()
```

```
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\2254106613.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  Pharma_data.mean()
```

Out[188]:
```
Mar Cap - Crore         30615.868750
Sales Qtr - Crore        1814.622500
Quaterly Sale- Crore     1156.244375
dtype: float64
```

In [189…
```
Technology_data.mean()
```

```
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\2750828425.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  Technology_data.mean()
```

Out[189]:
```
Mar Cap - Crore         24897.047333
Sales Qtr - Crore        2583.740667
Quaterly Sale- Crore      727.213333
dtype: float64
```

In [212…
```
a = Bank_data.mean()
b= Automotive_data.mean()
c = Power_data.mean()
d =Steel_data.mean()
e = Gas_data.mean()
f = Pharma_data.mean()
g = Technology_data.mean()
```

```
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\3118648952.py:1: FutureWarning: Drop
ping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprec
ated; in a future version this will raise TypeError.  Select only valid columns befor
e calling the reduction.
  a = Bank_data.mean()
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\3118648952.py:2: FutureWarning: Drop
ping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprec
ated; in a future version this will raise TypeError.  Select only valid columns befor
e calling the reduction.
  b= Automotive_data.mean()
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\3118648952.py:3: FutureWarning: Drop
ping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprec
ated; in a future version this will raise TypeError.  Select only valid columns befor
e calling the reduction.
  c = Power_data.mean()
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\3118648952.py:4: FutureWarning: Drop
ping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprec
ated; in a future version this will raise TypeError.  Select only valid columns befor
e calling the reduction.
  d =Steel_data.mean()
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\3118648952.py:5: FutureWarning: Drop
ping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprec
ated; in a future version this will raise TypeError.  Select only valid columns befor
e calling the reduction.
  e = Gas_data.mean()
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\3118648952.py:6: FutureWarning: Drop
ping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprec
ated; in a future version this will raise TypeError.  Select only valid columns befor
e calling the reduction.
  f = Pharma_data.mean()
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_8888\3118648952.py:7: FutureWarning: Drop
ping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprec
ated; in a future version this will raise TypeError.  Select only valid columns befor
e calling the reduction.
  g = Technology_data.mean()
```

In [222…

```
merged_df = pd.concat([a, b, c, d, e, f, g], axis=1)
merged_df.rename(columns={0: "Bank", 1:"Automotive", 2: "Power",
                          3: "Steel", 4: "Gas", 5: "Pharma", 6: "Technology"}, inplace
merged_df
```

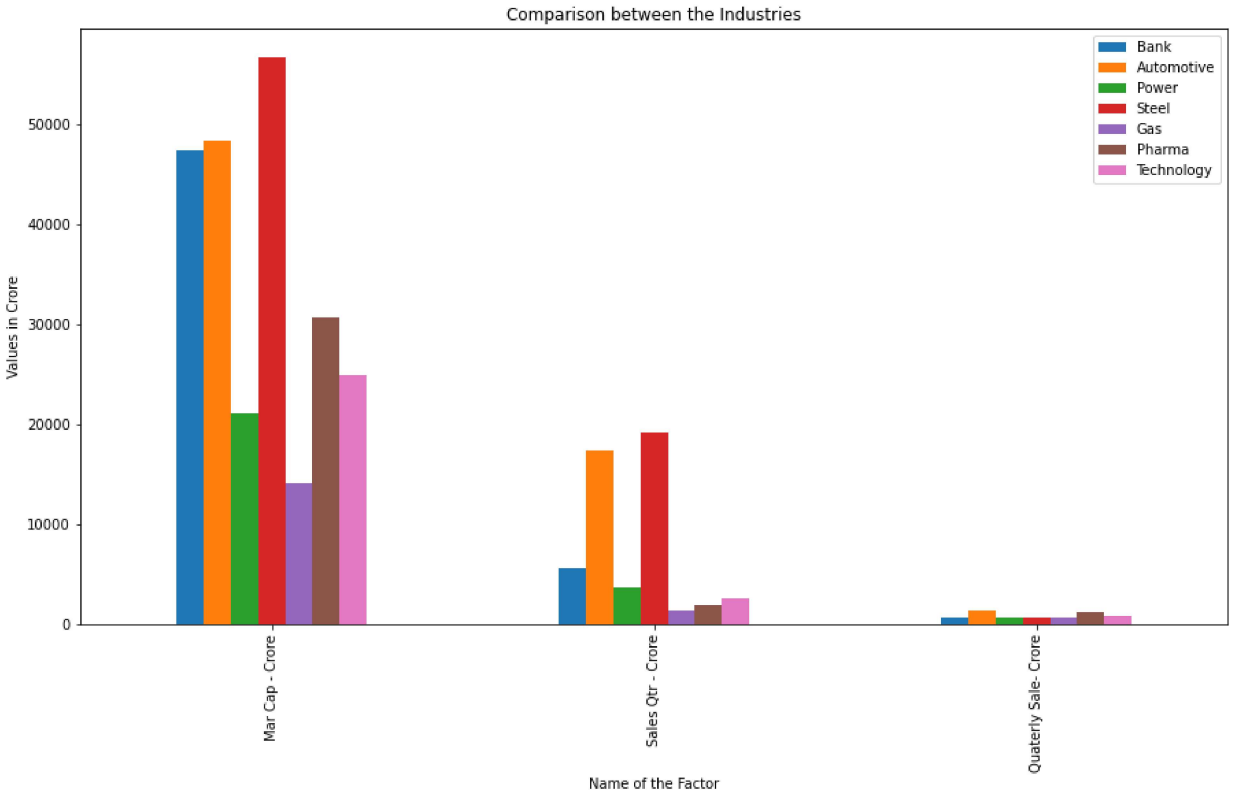
Out[222]:

	Bank	Automotive	Power	Steel	Gas	Pharma	Technolo
Mar Cap - Crore	47457.725806	48325.986	21121.285	56780.263333	14141.446667	30615.868750	24897.0473
Sales Qtr - Crore	5605.658710	17350.270	3623.543	19105.900000	1344.510000	1814.622500	2583.7406
Quaterly Sale-Crore	707.584194	1298.850	702.325	702.325000	702.325000	1156.244375	727.2133

In [227…

```
merged_df.plot(kind = 'bar', figsize=(15, 8))
plt.xlabel('Name of the Factor')
plt.ylabel('Values in Crore')
```

```
plt.title('Comparison between the Industries')
plt.show()
```



Summary

1. Understood the relationship between Market Capitalization and Sales and we can conclude from this that Reliance Inds has more market capitalization and IOCL has more Sales as compare to the other companies.
2. From the correlatin analysis we can understand that as the market capitalization of a company increases, its quarterly sales also tend to increase.
3. By comparing between the industries to identify the potential opportunties for growth we can conclude that Steel industry has more sales than the other industry.