Project-3 Financial Analysis

In [1]:
```
# Submitted by: Umang Parti
# Submitted to: Unified Mentor
```

In [2]:
```
# importing libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import statsmodels.api as sm
from IPython.display import FileLink
```

In [3]:
```
# importing dataset
finance_df = pd.read_csv(r"C:\Users\umang\Desktop\unified mentor\project - 3\Financial_data.c
print(finance_df.head())
```

```
   S.No.            Name  Mar Cap - Crore  Sales Qtr - Crore
0      1  Reliance Inds.        583436.72           99810.00
1      2             TCS        563709.84           30904.00
2      3       HDFC Bank        482953.59           20581.27
3      4             ITC        320985.27            9772.02
4      5         H D F C        289497.37           16840.51
```

In [4]:
```
# Cleaning the dataset
finance_df.isnull().sum()
```

Out[4]:
```
S.No.                 0
Name                  0
Mar Cap - Crore       9
Sales Qtr - Crore    30
dtype: int64
```

- We have missing values in 9 rows of Market Capital Column
- And missing values in 30 rows of Quarterly Sales

In [5]:
```
# Impute missing values (replace NaN with mean, median, or other strategies)
finance_df['Mar Cap - Crore'].fillna(finance_df['Mar Cap - Crore'].mean(), inplace=True)
```

In [6]:
```
# Impute missing values (replace NaN with mean, median, or other strategies)
finance_df['Sales Qtr - Crore'].fillna(finance_df['Sales Qtr - Crore'].mean(), inplace=True)
```

Exploratory Analysis

In [7]:
```
finance_df.shape
```

Out[7]:
```
(488, 4)
```

we can see that

- Our data of 488 companies
- Our data has four columns of serial number, company name, sales quarter and market capitalization

In [8]:
```
finance_df.columns
```

Out[8]:
```
Index(['S.No.', 'Name', 'Mar Cap - Crore', 'Sales Qtr - Crore'], dtype='object')
```

In [9]:
```
finance_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 488 entries, 0 to 487
Data columns (total 4 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   S.No.              488 non-null    int64
 1   Name               488 non-null    object
 2   Mar Cap - Crore    488 non-null    float64
 3   Sales Qtr - Crore  488 non-null    float64
dtypes: float64(2), int64(1), object(1)
memory usage: 15.4+ KB
```

In [10]:
```
finance_df.describe()
```

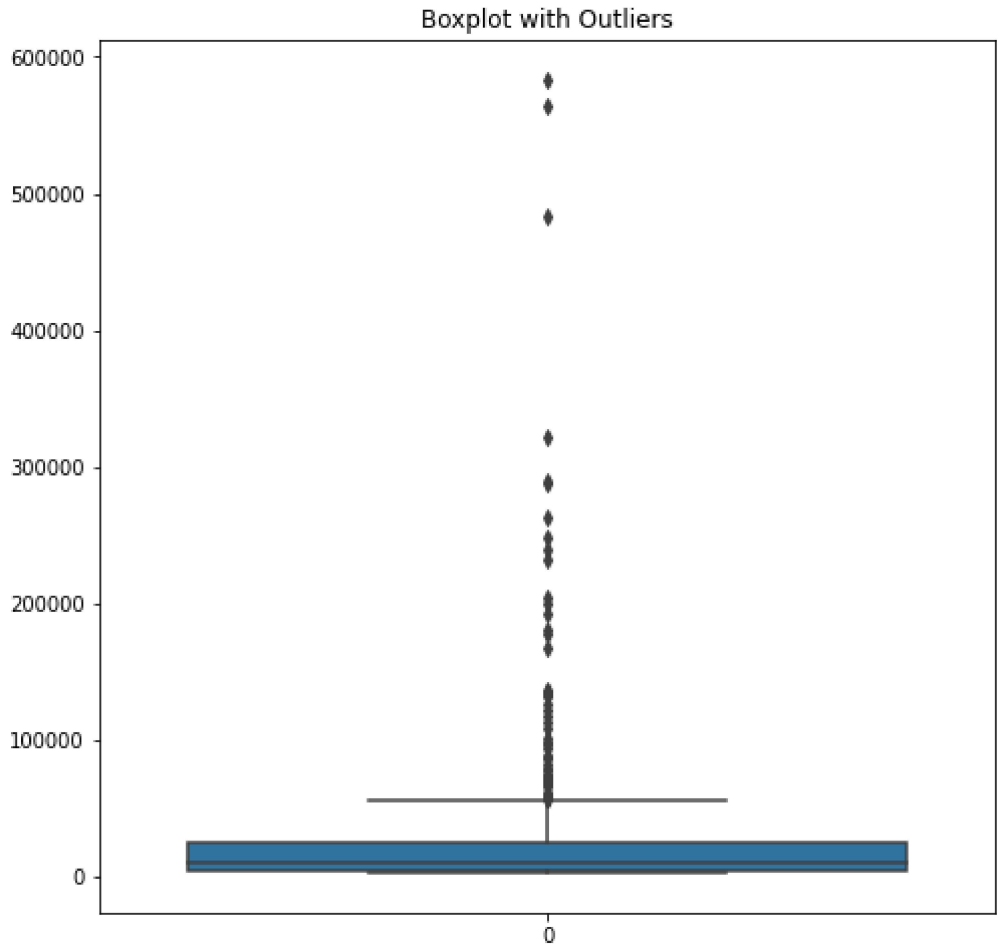
Out[10]:

	S.No.	Mar Cap - Crore	Sales Qtr - Crore
count	488.000000	488.000000	488.000000
mean	251.508197	28043.857119	3816.103362
std	145.884078	58912.585788	9685.917924
min	1.000000	3017.070000	19.420000
25%	122.750000	4879.612500	576.675000
50%	252.500000	10380.425000	1278.520000
75%	378.250000	25502.085000	3803.140841
max	500.000000	583436.720000	110666.930000

- Mean Market Capital is 28 thousand crores
- Maximum market capital is 583 thousand crore
- Minimum market capital is 3 thousand crore
- Mean Sales Quarter is 38 hundred crore
- Maximum Sales Quarter is 110 thousand crore
- Minimum Sales Quarter is 19 crore

Identifying Outliers

In [11]:
```
# Plotting boxplot for Market Capital
plt.figure(figsize=(8, 8))
sns.boxplot(finance_df['Mar Cap - Crore'])
plt.title('Boxplot with Outliers')
plt.savefig('my_chart.png')
plt.show()
```



In [12]:
```
FileLink(r'my_chart.png')
```

Out[12]: my_chart.png

In [13]:
```
# Function to get lower and upper critical values
def outlier_detection(df_marketcap):
    df_marketcap_cleaned = df_marketcap.dropna()
    if len(df_marketcap_cleaned) == 0:
        raise ValueError("No valid data after removing NaN values.")
    df_marketcap_sorted = sorted(df_marketcap_cleaned)
    Q1 = np.percentile(df_marketcap_sorted, 25)
    Q3 = np.percentile(df_marketcap_sorted, 75)
    IQR = Q3 - Q1
    lower_range = Q1 - (1.5 * IQR)
    upper_range = Q3 + (1.5 * IQR)
    return lower_range, upper_range
```

In [14]:
```
lower, upper = outlier_detection(finance_df['Mar Cap - Crore'])
print(lower, upper)
```

```
-26054.09625 56435.79375
```

In [15]:
```
# identifying outliers
outliers= (finance_df['Mar Cap - Crore'] >=upper) | (finance_df['Mar Cap - Crore'] <= lower)
outliers.value_counts()
```

Out[15]:
```
False    431
True      57
Name: Mar Cap - Crore, dtype: int64
```

- Out of 488 records our 57 records are considered as outliers

In [16]:
```
# Function to remove outliers based on IQR
def remove_outliers_iqr(df, column_name):
    df_no_outliers = df[(df[column_name] >= lower) & (df[column_name] <= upper)]
    return df_no_outliers
```

In [17]:
```
finance_df_no_outliers = remove_outliers_iqr(finance_df, 'Mar Cap - Crore')
print(finance_df_no_outliers)
```

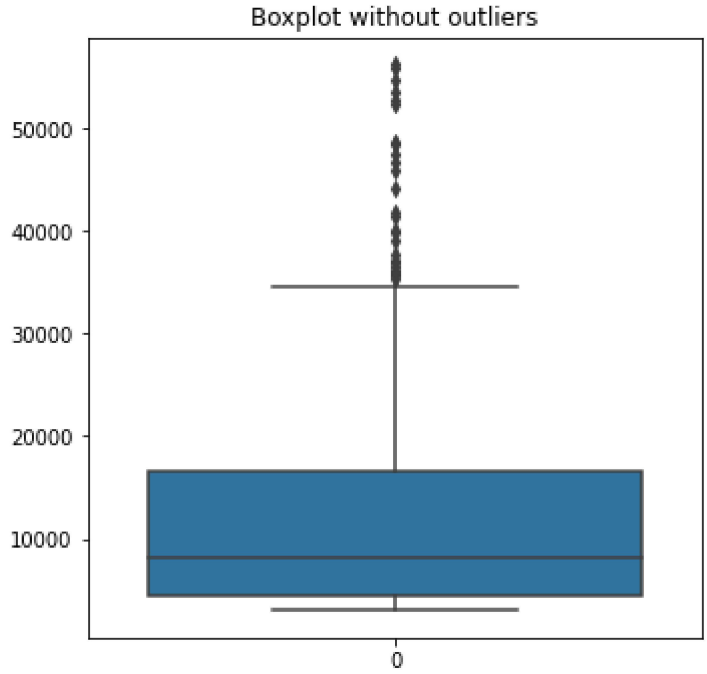
```
     S.No.              Name  Mar Cap - Crore  Sales Qtr - Crore
57      58     Tech Mahindra     56244.260000        7775.960000
58      59    Hindalco Inds.     55854.680000       11022.810000
59      60  Zee Entertainmen     54817.890000        1838.070000
60      61       Cairn India     53528.570000        2149.360000
61      62  Indiabulls Hous.     52781.670000        3115.890000
..     ...               ...              ...                ...
483    496   Lak. Vilas Bank      3029.570000         790.170000
484    497             NOCIL      3026.260000         249.270000
485    498     Orient Cement      3024.320000         511.530000
486    499   Natl.Fertilizer      3017.070000        2840.750000
487    500         L T Foods     28043.857119        3816.103362

[431 rows x 4 columns]
```

In [18]:
```
finance_df_no_outliers.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 431 entries, 57 to 487
Data columns (total 4 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   S.No.              431 non-null    int64  
 1   Name               431 non-null    object 
 2   Mar Cap - Crore    431 non-null    float64
 3   Sales Qtr - Crore  431 non-null    float64
dtypes: float64(2), int64(1), object(1)
memory usage: 16.8+ KB
```

In [19]:
```
finance_df_no_outliers_reset = finance_df_no_outliers.reset_index(drop=True)
plt.figure(figsize=(5.5,5.5))
sns.boxplot(finance_df_no_outliers_reset['Mar Cap - Crore'])
plt.title('Boxplot without outliers')
plt.savefig('my_chart2.png')
plt.show()
```



In [20]:
```
FileLink(r'my_chart2.png')
```

Out[20]: my_chart2.png

- We have successfully removed the Outliers.
- Earlier the maximum Market Capitalization was approx 6 lakhs crores
- Now Distribution is even and maximum market capitalization is 50 thousand crore

Market Capitalization Distribution

In [21]:
```
# Sort the Data by market capitalization in descending order
finance_df_sorted = finance_df.sort_values(by='Mar Cap - Crore', ascending=False)
print(finance_df_sorted)
```

```
     S.No.             Name  Mar Cap - Crore  Sales Qtr - Crore
0        1   Reliance Inds.        583436.72           99810.00
1        2              TCS        563709.84           30904.00
2        3        HDFC Bank        482953.59           20581.27
3        4              ITC        320985.27            9772.02
4        5          H D F C        289497.37           16840.51
..     ...              ...              ...                ...
482    495      Prime Focus          3031.50             609.61
483    496  Lak. Vilas Bank          3029.57             790.17
484    497            NOCIL          3026.26             249.27
485    498    Orient Cement          3024.32             511.53
486    499  Natl.Fertilizer          3017.07            2840.75

[488 rows x 4 columns]
```

In [22]:
```
# We will group the data in an interval of 50 each
bin_labels = ['500-450','450-400','400-350','350-300','300-250','250-200','200-150',
              '150-100','100-50','50-0','0']
bin_edges = np.arange(0, 551, 50)
```

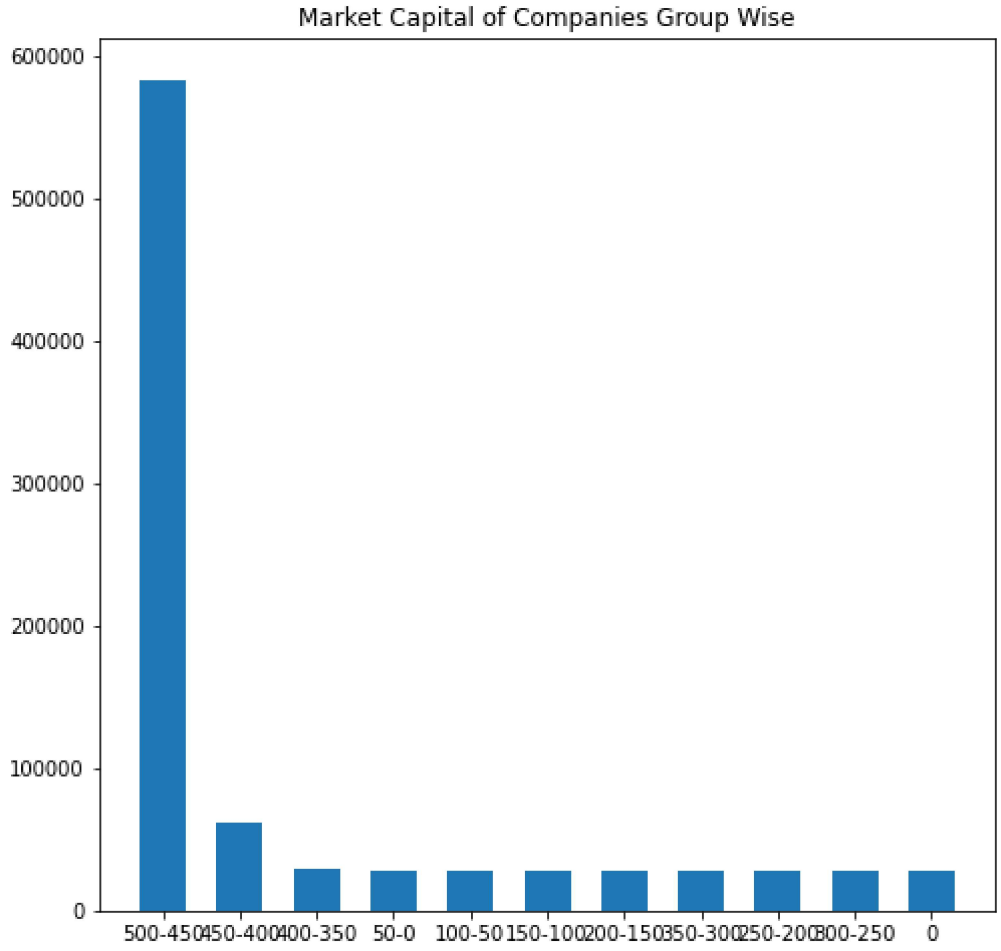
In [23]:
```
finance_df_sorted['Groups'] = pd.cut(finance_df_sorted['S.No.'],
                                     bins=bin_edges, labels=bin_labels, right=False)
finance_df_sorted['Groups'] = pd.Categorical(finance_df_sorted['Groups'], categories=bin_labe
                                             ordered=True)
finance_df_sorted
```

Out[23]:

	S.No.	Name	Mar Cap - Crore	Sales Qtr - Crore	Groups
0	1	Reliance Inds.	583436.72	99810.00	500-450
1	2	TCS	563709.84	30904.00	500-450
2	3	HDFC Bank	482953.59	20581.27	500-450
3	4	ITC	320985.27	9772.02	500-450
4	5	H D F C	289497.37	16840.51	500-450
...	...	...	...	...	...
482	495	Prime Focus	3031.50	609.61	50-0
483	496	Lak. Vilas Bank	3029.57	790.17	50-0
484	497	NOCIL	3026.26	249.27	50-0
485	498	Orient Cement	3024.32	511.53	50-0
486	499	Natl.Fertilizer	3017.07	2840.75	50-0

488 rows × 5 columns

In [24]:
```
# Creating bar chart and plotting the companies in agroup of 50 and their market capital
plt.figure(figsize=(8, 8))
plt.bar(finance_df_sorted['Groups'], finance_df_sorted['Mar Cap - Crore'], width=0.6)
plt.title('Market Capital of Companies Group Wise')
plt.xticks(finance_df_sorted['Groups'].cat.categories)
plt.savefig('my_chart3.png')
plt.show()
```



In [25]:
```
FileLink(r'my_chart3.png')
```

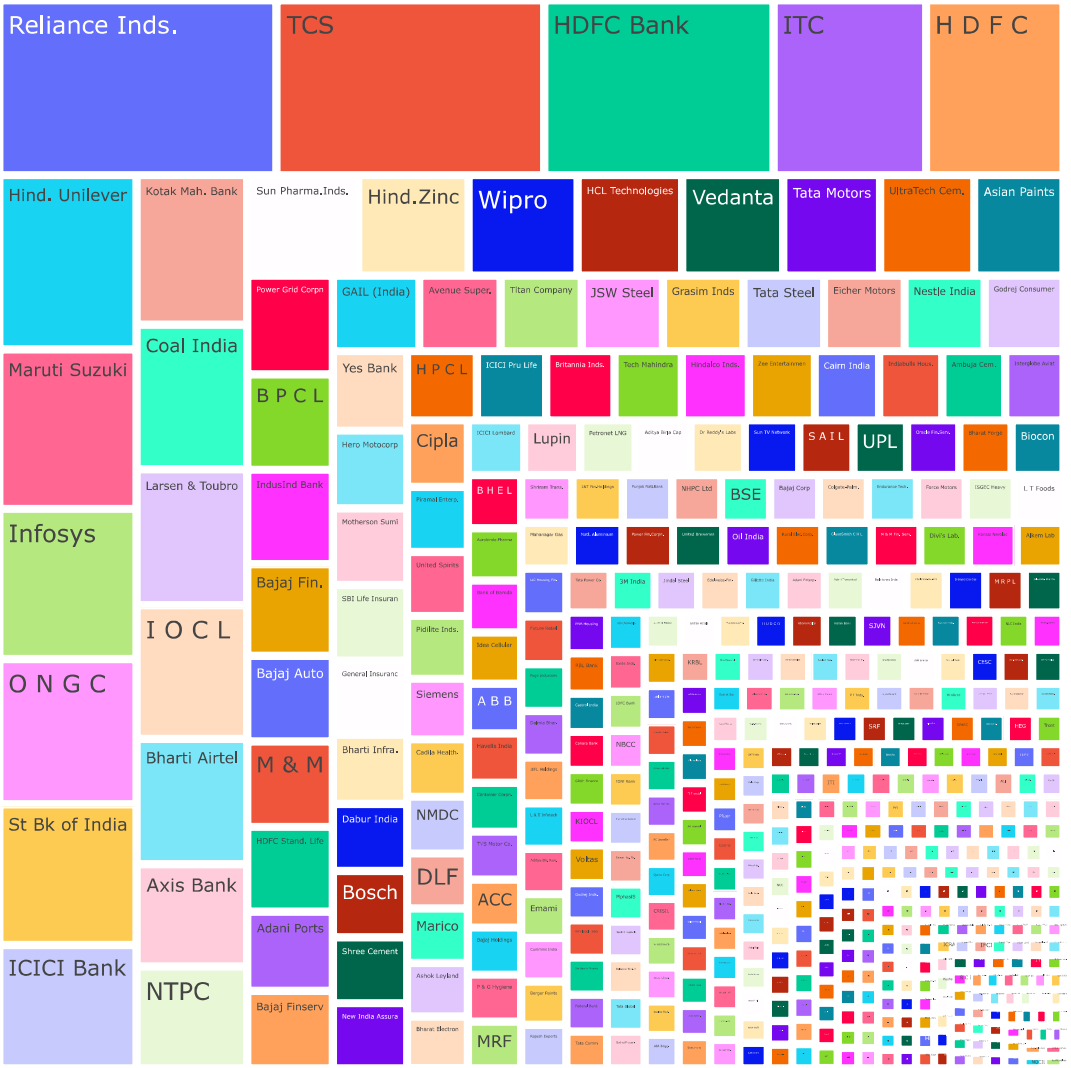
Out[25]: my_chart3.png

In [26]:
```
pip install -U kaleido
```

```
Requirement already satisfied: kaleido in c:\users\umang\anaconda3\envs\rstudio\lib\site-pack
ages (0.2.1)
Note: you may need to restart the kernel to use updated packages.
```

In [27]:
```
fig = px.treemap(finance_df_sorted, path=['Name'], values='Mar Cap - Crore')
fig.update_layout(width=700,height=700)
fig.update_layout(title_text='Treemap of Companies and Market Capital')
fig.show()
fig.write_image('treemap_chart.png')
```

Treemap of Companies and Market Capital



In [28]:
```
FileLink(r'treemap_chart.png')
```

Out[28]: treemap_chart.png

In [29]:
```
finance_df_sorted.drop(columns='Groups',inplace = True)
```

Top 50 Companies

In [30]:
```
# Select the top 50 companies
top_50_companies = finance_df_sorted.head(50)
print(top_50_companies.head())
```

```
   S.No.            Name  Mar Cap - Crore  Sales Qtr - Crore
0      1  Reliance Inds.        583436.72           99810.00
1      2             TCS        563709.84           30904.00
2      3       HDFC Bank        482953.59           20581.27
3      4             ITC        320985.27            9772.02
4      5         H D F C        289497.37           16840.51
```

In [31]:
```
# To get to 50 companies and their share in market capitalization
# Calculate the total market capitalization of the top 500 companies
```

```
total_market_cap_top_500 = finance_df['Mar Cap - Crore'].sum()
```

In [32]:
```
# Calculate the share of each company in the total market capitalization
top_50_companies['MarketCapShare'] = (top_50_companies['Mar Cap - Crore']/
                                      total_market_cap_top_500 * 100)
```

```
C:\Users\umang\anaconda3\envs\rstudio\lib\site-packages\ipykernel_launcher.py:3: SettingWithC
opyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guid
e/indexing.html#returning-a-view-versus-a-copy
```

In [33]:
```
# Print the top 50 companies
print(top_50_companies.head(20))
```

```
    S.No.              Name  Mar Cap - Crore  Sales Qtr - Crore  \
0       1    Reliance Inds.        583436.72           99810.00
1       2               TCS        563709.84           30904.00
2       3         HDFC Bank        482953.59           20581.27
3       4               ITC        320985.27            9772.02
4       5           H D F C        289497.37           16840.51
5       6    Hind. Unilever        288265.26            8590.00
6       7     Maruti Suzuki        263493.81           19283.20
7       8           Infosys        248320.35           17794.00
8       9           O N G C        239981.50           22995.88
9      10    St Bk of India        232763.33           57014.08
10     11        ICICI Bank        203802.35           13665.35
11     12   Kotak Mah. Bank        199253.77            6390.71
12     13        Coal India        192677.98           21643.28
13     14   Larsen & Toubro        180860.74           28747.45
14     15           I O C L        178017.48          110666.93
15     16     Bharti Airtel        167131.29           20318.60
16     17         Axis Bank        136380.76           11721.55
17     18              NTPC        135390.53           20774.37
18     19  Sun Pharma.Inds.        134241.36            6653.23
19     20         Hind.Zinc        133266.56            5922.00

    MarketCapShare
0         4.263205
1         4.119059
2         3.528969
3         2.345457
4         2.115373
5         2.106370
6         1.925364
7         1.814491
8         1.753558
9         1.700815
10        1.489195
11        1.455958
12        1.407909
13        1.321560
14        1.300784
15        1.221238
16        0.996542
17        0.989306
18        0.980909
19        0.973786
```

In [34]:
```
plt.figure(figsize=(6,6))
top_50_companies['MarketCapShare'].plot()
plt.savefig('my_chart5.png')
plt.title('Market Capital top 50 companies')
plt.show()
```

In [35]:
```
FileLink(r'my_chart5.png')
```

Out[35]: my_chart5.png

In [36]:
```
print(top_50_companies['MarketCapShare'].head(6).sum())
print(top_50_companies['MarketCapShare'].head(15).sum())
```

```
18.4784341691678
32.64806741169255
```

- We Notice that Top 6 companies contribute around 18.47% of Market Share
- Top 15 companies contribute to 32.64% of Market Share

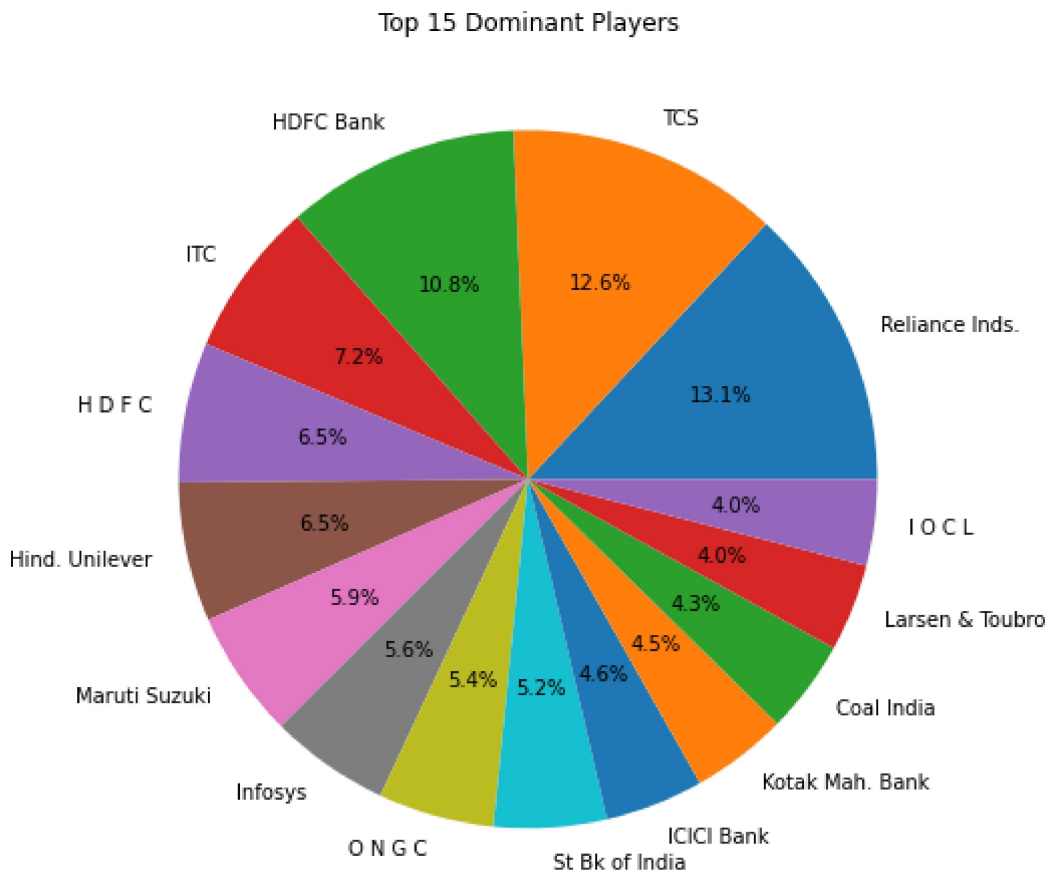
Top 15 Dominant Players in the Market

In [37]:
```
# Select the top 15 companies
top_15_companies = finance_df_sorted.head(15)
print(top_15_companies)
```

```
    S.No.             Name  Mar Cap - Crore  Sales Qtr - Crore
0       1   Reliance Inds.        583436.72           99810.00
1       2              TCS        563709.84           30904.00
2       3        HDFC Bank        482953.59           20581.27
3       4              ITC        320985.27            9772.02
4       5          H D F C        289497.37           16840.51
5       6   Hind. Unilever        288265.26            8590.00
6       7    Maruti Suzuki        263493.81           19283.20
7       8          Infosys        248320.35           17794.00
8       9          O N G C        239981.50           22995.88
9      10   St Bk of India        232763.33           57014.08
10     11       ICICI Bank        203802.35           13665.35
11     12  Kotak Mah. Bank        199253.77            6390.71
12     13       Coal India        192677.98           21643.28
13     14  Larsen & Toubro        180860.74           28747.45
14     15          I O C L        178017.48          110666.93
```

In [38]:
```
# creating a piechart
labels = top_15_companies['Name']
sizes = top_15_companies['Mar Cap - Crore']
```

In [39]:
```
plt.figure(figsize=(8, 8))
plt.pie(sizes, labels=labels, autopct='%1.1f%%')
plt.title('Top 15 Dominant Players')
plt.savefig('my_chart6.png')
plt.show()
```



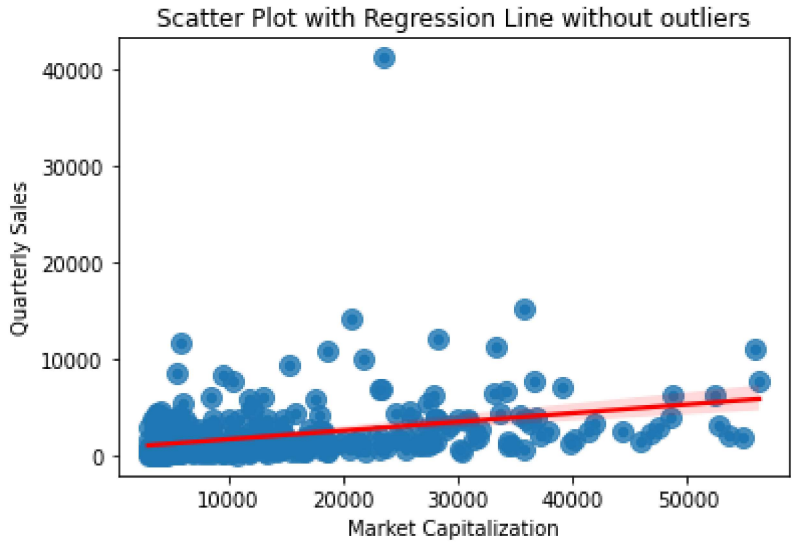
In [40]:
```
FileLink(r'my_chart6.png')
```

Out[40]: my_chart6.png

Market Capital and Quarterly Sales

In [41]:
```
# Scatter Plot
```

In [42]:
```
# Plotting the scatter plot with regression line
sns.scatterplot(x='Mar Cap - Crore', y='Sales Qtr - Crore', data=finance_df_no_outliers)
# for our analysis we will use the data without outliers
# Plot the regression line
sns.regplot(x='Mar Cap - Crore', y='Sales Qtr - Crore', data=finance_df_no_outliers,
            scatter_kws={'s': 100}, line_kws={'color': 'red'})
plt.xlabel('Market Capitalization')
plt.ylabel('Quarterly Sales')
plt.title('Scatter Plot with Regression Line without outliers')
plt.savefig('my_chart7.png')
plt.show()
```



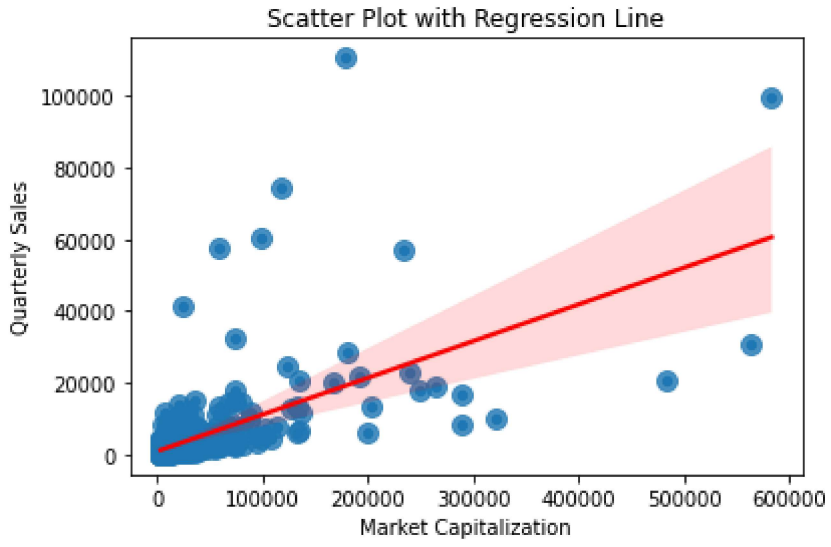
In [43]:
```
FileLink(r'my_chart7.png')
```

Out[43]: my_chart7.png

- We have Market Capital on X axis and Quaterly Sales on Y axis
- Mostly the points are clustered closed to 0 depicting that lower the market capital lower is the quaterly sales
- With the help of regression line we are able to see an upward positive trend between the variables more carefully
- The increase in Quarterly sales along with many other factors lead to increase in market capital

In [44]:
```
# Let us plot a scatter plot with Outliers as well
# Plotting the scatter plot with regression line
sns.scatterplot(x='Mar Cap - Crore', y='Sales Qtr - Crore', data=finance_df)

# Plot the regression line
sns.regplot(x='Mar Cap - Crore', y='Sales Qtr - Crore', data=finance_df, scatter_kws={'s': 10
            line_kws={'color': 'red'})
plt.xlabel('Market Capitalization')
plt.ylabel('Quarterly Sales')
plt.title('Scatter Plot with Regression Line')
plt.savefig('my_chart8.png')
plt.show()
```



In [45]:
```
FileLink(r'my_chart8.png')
```

Out[45]: my_chart8.png

- Now we see a positive upward slopping line more clearly and accurately

In [46]:
```
# Correlation Analysis
```

In [47]:
```
# To find correlation between the two variables
correlation_coefficient = finance_df['Mar Cap - Crore'].corr(finance_df['Sales Qtr - Crore'])
print("Correlation Coefficient:",correlation_coefficient)
```

```
Correlation Coefficient: 0.6221969818279508
```

- We find that we have a positive correlation between the variables of 0.62 which is fairly significant

In [48]:
```
# Regression Analysis
```

In [49]:
```
X = sm.add_constant(finance_df['Mar Cap - Crore'])
X
```

Out[49]:

	const	Mar Cap - Crore
0	1.0	583436.720000
1	1.0	563709.840000
2	1.0	482953.590000
3	1.0	320985.270000
4	1.0	289497.370000
...	...	...
483	1.0	3029.570000
484	1.0	3026.260000
485	1.0	3024.320000
486	1.0	3017.070000
487	1.0	28043.857119

488 rows × 2 columns

In [50]:
```
model = sm.OLS(finance_df['Sales Qtr - Crore'], X).fit()
```

In [51]:
```
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:      Sales Qtr - Crore   R-squared:                       0.387
Model:                            OLS   Adj. R-squared:                  0.386
Method:                 Least Squares   F-statistic:                     307.0
Date:                Mon, 15 Jan 2024   Prob (F-statistic):           1.24e-53
Time:                        12:24:30   Log-Likelihood:                -5051.6
No. Observations:                 488   AIC:                         1.011e+04
Df Residuals:                     486   BIC:                         1.012e+04
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                     coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const             947.3161    380.623      2.489      0.013     199.445    1695.187
Mar Cap - Crore     0.1023      0.006     17.521      0.000       0.091       0.114
==============================================================================
Omnibus:                      629.272   Durbin-Watson:                   1.757
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            81966.139
Skew:                           6.299   Prob(JB):                         0.00
Kurtosis:                      65.229   Cond. No.                     7.22e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 7.22e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

- F-statistic of 307 typically expect a very low p-value (close to zero), indicating that the overall model is statistically significant.
- R-squared of 0.387 means that approximately 38.7% of the variability in the dependent variable is explained by the independent variables included in your model.
- Adjusted R-squared of 0.386 is slightly lower than the R-squared. This indicates that the additional predictors in your model are not contributing much explanatory power.
- The coefficient for the variable "Mar Cap - Crore" is 0.1023.
- The standard error of the coefficient is 0.006, indicating the precision of the estimate.
- The t-statistic of 17.521 is highly significant, suggesting that the variable is strongly related to the dependent variable.
- The p-value (0.000) is less than the significance level, indicating the statistical significance of "Mar Cap - Crore."
- The 95% confidence interval for the coefficient is [0.091, 0.114], providing a range for the true population coefficient.

In [52]:
```
#.....................................................................................
```