Stock Analysis

In [69]:
```
import pandas as pd
import os
import glob
import numpy as np
```

In [2]:
```
path=r'C:\Users\dell\Downloads\individual_stocks_5yr'
```

In [3]:
```
file=os.listdir(path)
```

In [5]:
```
os.chdir(r'C:\Users\dell\Downloads\individual_stocks_5yr')
a=[]
for i in file:
    d=pd.read_csv(i,index_col=None, header=0)
    a.append(d)
```

In [6]:
```
df=pd.concat(a,axis=0, ignore_index=True)
```

In [7]:
```
df
```

Out[7]:

	date	open	high	low	close	volume	Name
0	2013-02-08	15.07	15.12	14.63	14.75	8407500	AAL
1	2013-02-11	14.89	15.01	14.26	14.46	8882000	AAL
2	2013-02-12	14.45	14.51	14.10	14.27	8126000	AAL
3	2013-02-13	14.30	14.94	14.25	14.66	10259500	AAL
4	2013-02-14	14.94	14.96	13.16	13.99	31879900	AAL
...	...	...	...	...	...	...	...
624071	2018-02-01	76.84	78.27	76.69	77.82	2982259	ZTS
624072	2018-02-02	77.53	78.12	76.73	76.78	2595187	ZTS
624073	2018-02-05	76.64	76.92	73.18	73.83	2962031	ZTS
624074	2018-02-06	72.74	74.56	72.13	73.27	4924323	ZTS
624075	2018-02-07	72.70	75.00	72.69	73.86	4534912	ZTS

624076 rows × 7 columns

In [8]:
```
df.dtypes
```

Out[8]:
```
date       object
open      float64
high      float64
low       float64
close     float64
volume      int64
Name       object
dtype: object
```

In [9]:
```
df['date']=pd.to_datetime(df['date'])
```

In [10]:
```
df['date'][0]
```

Out[10]:
```
Timestamp('2013-02-08 00:00:00')
```

In [11]:
```
df.dtypes
```

Out[11]:
```
date      datetime64[ns]
open             float64
high             float64
low              float64
close            float64
volume             int64
Name              object
dtype: object
```

In [12]:
```
df.describe()
```

Out[12]:

	open	high	low	close	volume
count	624065.000000	624068.00000	624068.000000	624076.000000	6.240760e+05
mean	82.785214	83.53762	82.020783	82.805679	4.364546e+06
std	97.038541	97.86450	96.170141	97.049487	8.730019e+06
min	1.620000	1.69000	1.500000	1.590000	0.000000e+00
25%	40.130000	40.52000	39.730000	40.150000	1.075820e+06
50%	62.440000	63.00000	61.870000	62.470000	2.096811e+06
75%	94.050000	94.86000	93.240000	94.090000	4.322817e+06
max	2044.000000	2067.99000	2035.110000	2049.000000	6.182376e+08

In [13]:
```
nan_rows  = df.loc[df.isna().any(axis=1)]
```

In [14]:
```
nan_rows
```

Out[14]:

	date	open	high	low	close	volume	Name
82949	2017-07-26	NaN	NaN	NaN	69.0842	3	BHF
165734	2015-07-17	NaN	88.76	88.24	88.7200	2056819	DHR
165857	2016-01-12	NaN	NaN	NaN	88.5500	0	DHR
205076	2015-07-17	NaN	48.49	47.85	47.9200	1246786	ES
239832	2016-07-01	NaN	NaN	NaN	49.5400	0	FTV
434379	2015-07-17	NaN	47.31	46.83	46.9900	1229513	O
434502	2016-01-12	NaN	NaN	NaN	52.4300	0	O
478594	2015-06-09	NaN	NaN	NaN	526.0900	12135	REGN
563249	2016-04-07	NaN	NaN	NaN	41.5600	0	UA
586942	2015-05-12	NaN	NaN	NaN	124.0800	569747	VRTX
603272	2015-06-26	NaN	NaN	NaN	61.9000	100	WRK

In [15]:
```
df1=df.dropna()
```

In [16]:
```
df1
```

Out[16]:

	date	open	high	low	close	volume	Name
0	2013-02-08	15.07	15.12	14.63	14.75	8407500	AAL
1	2013-02-11	14.89	15.01	14.26	14.46	8882000	AAL
2	2013-02-12	14.45	14.51	14.10	14.27	8126000	AAL
3	2013-02-13	14.30	14.94	14.25	14.66	10259500	AAL
4	2013-02-14	14.94	14.96	13.16	13.99	31879900	AAL
...	...	...	...	...	...	...	...
624071	2018-02-01	76.84	78.27	76.69	77.82	2982259	ZTS
624072	2018-02-02	77.53	78.12	76.73	76.78	2595187	ZTS
624073	2018-02-05	76.64	76.92	73.18	73.83	2962031	ZTS
624074	2018-02-06	72.74	74.56	72.13	73.27	4924323	ZTS
624075	2018-02-07	72.70	75.00	72.69	73.86	4534912	ZTS

624065 rows × 7 columns

In [17]:
```
nan_check  = df1.loc[df1.isna().any(axis=1)]
```

In [18]:
```
nan_check
```

Out[18]:

date	open	high	low	close	volume	Name

In [19]:
```
df1.describe()
```

Out[19]:

	open	high	low	close	volume
count	624065.000000	624065.000000	624065.000000	624065.000000	6.240650e+05
mean	82.785214	83.537726	82.020884	82.805221	4.364615e+06
std	97.038541	97.864714	96.170352	97.048648	8.730080e+06
min	1.620000	1.690000	1.500000	1.590000	1.010000e+02
25%	40.130000	40.520000	39.730000	40.150000	1.075840e+06
50%	62.440000	63.000000	61.870000	62.470000	2.096857e+06
75%	94.050000	94.860000	93.240000	94.090000	4.322947e+06
max	2044.000000	2067.990000	2035.110000	2049.000000	6.182376e+08

In [20]:
```
df1
```

Out[20]:

	date	open	high	low	close	volume	Name
0	2013-02-08	15.07	15.12	14.63	14.75	8407500	AAL
1	2013-02-11	14.89	15.01	14.26	14.46	8882000	AAL
2	2013-02-12	14.45	14.51	14.10	14.27	8126000	AAL
3	2013-02-13	14.30	14.94	14.25	14.66	10259500	AAL
4	2013-02-14	14.94	14.96	13.16	13.99	31879900	AAL
...	...	...	...	...	...	...	...
624071	2018-02-01	76.84	78.27	76.69	77.82	2982259	ZTS
624072	2018-02-02	77.53	78.12	76.73	76.78	2595187	ZTS
624073	2018-02-05	76.64	76.92	73.18	73.83	2962031	ZTS
624074	2018-02-06	72.74	74.56	72.13	73.27	4924323	ZTS
624075	2018-02-07	72.70	75.00	72.69	73.86	4534912	ZTS

624065 rows × 7 columns

In [21]:
```
AAL=df1[df1['Name']=='AAL']
```

In [22]:
```
AAL.head()
```

Out[22]:

	date	open	high	low	close	volume	Name
0	2013-02-08	15.07	15.12	14.63	14.75	8407500	AAL
1	2013-02-11	14.89	15.01	14.26	14.46	8882000	AAL
2	2013-02-12	14.45	14.51	14.10	14.27	8126000	AAL
3	2013-02-13	14.30	14.94	14.25	14.66	10259500	AAL
4	2013-02-14	14.94	14.96	13.16	13.99	31879900	AAL

In [23]:
```
len(pd.unique(df1['Name']))
```

Out[23]:
```
505
```

In [24]:
```
df1.nunique()
```

Out[24]:
```
date        1259
open       49715
high       81499
low        82354
close      51151
volume    586434
Name         505
dtype: int64
```

In [25]:
```
stock_list=df1['Name'].unique()
```

In [26]:
```
stock_list
```

Out[26]:
```
array(['AAL', 'AAPL', 'AAP', 'ABBV', 'ABC', 'ABT', 'ACN', 'ADBE', 'ADI',
       'ADM', 'ADP', 'ADSK', 'ADS', 'AEE', 'AEP', 'AES', 'AET', 'AFL',
       'AGN', 'AIG', 'AIV', 'AIZ', 'AJG', 'AKAM', 'ALB', 'ALGN', 'ALK',
       'ALLE', 'ALL', 'ALXN', 'AMAT', 'AMD', 'AME', 'AMGN', 'AMG', 'AMP',
       'AMT', 'AMZN', 'ANDV', 'ANSS', 'ANTM', 'AON', 'AOS', 'APA', 'APC',
       'APD', 'APH', 'APTV', 'ARE', 'ARNC', 'ATVI', 'AVB', 'AVGO', 'AVY',
       'AWK', 'AXP', 'AYI', 'AZO', 'A', 'BAC', 'BAX', 'BA', 'BBT', 'BBY',
       'BDX', 'BEN', 'BF.B', 'BHF', 'BHGE', 'BIIB', 'BK', 'BLK', 'BLL',
       'BMY', 'BRK.B', 'BSX', 'BWA', 'BXP', 'CAG', 'CAH', 'CAT', 'CA',
       'CBG', 'CBOE', 'CBS', 'CB', 'CCI', 'CCL', 'CDNS', 'CELG', 'CERN',
       'CFG', 'CF', 'CHD', 'CHK', 'CHRW', 'CHTR', 'CINF', 'CI', 'CLX',
       'CL', 'CMA', 'CMCSA', 'CME', 'CMG', 'CMI', 'CMS', 'CNC', 'CNP',
       'COF', 'COG', 'COL', 'COO', 'COP', 'COST', 'COTY', 'CPB', 'CRM',
       'CSCO', 'CSRA', 'CSX', 'CTAS', 'CTL', 'CTSH', 'CTXS', 'CVS', 'CVX',
       'CXO', 'C', 'DAL', 'DE', 'DFS', 'DGX', 'DG', 'DHI', 'DHR', 'DISCA',
       'DISCK', 'DISH', 'DIS', 'DLR', 'DLTR', 'DOV', 'DPS', 'DRE', 'DRI',
       'DTE', 'DUK', 'DVA', 'DVN', 'DWDP', 'DXC', 'D', 'EA', 'EBAY',
       'ECL', 'ED', 'EFX', 'EIX', 'EL', 'EMN', 'EMR', 'EOG', 'EQIX',
       'EQR', 'EQT', 'ESRX', 'ESS', 'ES', 'ETFC', 'ETN', 'ETR', 'EVHC',
       'EW', 'EXC', 'EXPD', 'EXPE', 'EXR', 'FAST', 'FBHS', 'FB', 'FCX',
       'FDX', 'FE', 'FFIV', 'FISV', 'FIS', 'FITB', 'FLIR', 'FLR', 'FLS',
       'FL', 'FMC', 'FOXA', 'FOX', 'FRT', 'FTI', 'FTV', 'F', 'GD', 'GE',
       'GGP', 'GILD', 'GIS', 'GLW', 'GM', 'GOOGL', 'GOOG', 'GPC', 'GPN',
       'GPS', 'GRMN', 'GS', 'GT', 'GWW', 'HAL', 'HAS', 'HBAN', 'HBI',
       'HCA', 'HCN', 'HCP', 'HD', 'HES', 'HIG', 'HII', 'HLT', 'HOG',
       'HOLX', 'HON', 'HPE', 'HPQ', 'HP', 'HRB', 'HRL', 'HRS', 'HSIC',
       'HST', 'HSY', 'HUM', 'IBM', 'ICE', 'IDXX', 'IFF', 'ILMN', 'INCY',
       'INFO', 'INTC', 'INTU', 'IPG', 'IP', 'IQV', 'IRM', 'IR', 'ISRG',
       'ITW', 'IT', 'IVZ', 'JBHT', 'JCI', 'JEC', 'JNJ', 'JNPR', 'JPM',
       'JWN', 'KEY', 'KHC', 'KIM', 'KLAC', 'KMB', 'KMI', 'KMX', 'KORS',
       'KO', 'KR', 'KSS', 'KSU', 'K', 'LB', 'LEG', 'LEN', 'LH', 'LKQ',
       'LLL', 'LLY', 'LMT', 'LNC', 'LNT', 'LOW', 'LRCX', 'LUK', 'LUV',
       'LYB', 'L', 'MAA', 'MAC', 'MAR', 'MAS', 'MAT', 'MA', 'MCD', 'MCHP',
       'MCK', 'MCO', 'MDLZ', 'MDT', 'MET', 'MGM', 'MHK', 'MKC', 'MLM',
       'MMC', 'MMM', 'MNST', 'MON', 'MOS', 'MO', 'MPC', 'MRK', 'MRO',
       'MSFT', 'MSI', 'MS', 'MTB', 'MTD', 'MU', 'MYL', 'M', 'NAVI', 'NBL',
       'NCLH', 'NDAQ', 'NEE', 'NEM', 'NFLX', 'NFX', 'NI', 'NKE', 'NLSN',
       'NOC', 'NOV', 'NRG', 'NSC', 'NTAP', 'NTRS', 'NUE', 'NVDA', 'NWL',
       'NWSA', 'NWS', 'OKE', 'OMC', 'ORCL', 'ORLY', 'OXY', 'O', 'PAYX',
       'PBCT', 'PCAR', 'PCG', 'PCLN', 'PDCO', 'PEG', 'PEP', 'PFE', 'PFG',
       'PGR', 'PG', 'PHM', 'PH', 'PKG', 'PKI', 'PLD', 'PM', 'PNC', 'PNR',
       'PNW', 'PPG', 'PPL', 'PRGO', 'PRU', 'PSA', 'PSX', 'PVH', 'PWR',
       'PXD', 'PX', 'PYPL', 'QCOM', 'QRVO', 'RCL', 'REGN', 'REG', 'RE',
       'RF', 'RHI', 'RHT', 'RJF', 'RL', 'RMD', 'ROK', 'ROP', 'ROST',
       'RRC', 'RSG', 'RTN', 'SBAC', 'SBUX', 'SCG', 'SCHW', 'SEE', 'SHW',
       'SIG', 'SJM', 'SLB', 'SLG', 'SNA', 'SNI', 'SNPS', 'SO', 'SPGI',
       'SPG', 'SRCL', 'SRE', 'STI', 'STT', 'STX', 'STZ', 'SWKS', 'SWK',
       'SYF', 'SYK', 'SYMC', 'SYY', 'TAP', 'TDG', 'TEL', 'TGT', 'TIF',
       'TJX', 'TMK', 'TMO', 'TPR', 'TRIP', 'TROW', 'TRV', 'TSCO', 'TSN',
       'TSS', 'TWX', 'TXN', 'TXT', 'T', 'UAA', 'UAL', 'UA', 'UDR', 'UHS',
       'ULTA', 'UNH', 'UNM', 'UNP', 'UPS', 'URI', 'USB', 'UTX', 'VAR',
       'VFC', 'VIAB', 'VLO', 'VMC', 'VNO', 'VRSK', 'VRSN', 'VRTX', 'VTR',
       'VZ', 'V', 'WAT', 'WBA', 'WDC', 'WEC', 'WFC', 'WHR', 'WLTW', 'WMB',
       'WMT', 'WM', 'WRK', 'WU', 'WYNN', 'WYN', 'WY', 'XEC', 'XEL',
       'XLNX', 'XL', 'XOM', 'XRAY', 'XRX', 'XYL', 'YUM', 'ZBH', 'ZION',
       'ZTS'], dtype=object)
```

In [27]:
```
len(stock_list)
```

Out[27]:
```
505
```

In [28]:
```
df1['Month']=pd.DataFrame(df1['date'].dt.month)
df1['Year']=pd.DataFrame(df1['date'].dt.year)
```

```
C:\Users\dell\AppData\Local\Temp\ipykernel_996\373712656.py:1: SettingWithCopyWarning:
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_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df1['Month']=pd.DataFrame(df1['date'].dt.month)
C:\Users\dell\AppData\Local\Temp\ipykernel_996\373712656.py:2: SettingWithCopyWarning:
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_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df1['Year']=pd.DataFrame(df1['date'].dt.year)
```

In [29]:
```
df1.head()
```

Out[29]:

	date	open	high	low	close	volume	Name	Month	Year
0	2013-02-08	15.07	15.12	14.63	14.75	8407500	AAL	2	2013
1	2013-02-11	14.89	15.01	14.26	14.46	8882000	AAL	2	2013
2	2013-02-12	14.45	14.51	14.10	14.27	8126000	AAL	2	2013
3	2013-02-13	14.30	14.94	14.25	14.66	10259500	AAL	2	2013
4	2013-02-14	14.94	14.96	13.16	13.99	31879900	AAL	2	2013

In [30]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

closing price Vs Time

In [31]:
```
df1.sample(1001,ignore_index=True)
```

Out[31]:

	date	open	high	low	close	volume	Name	Month	Year
0	2013-05-08	68.83	69.15	68.670	69.10	649916	SNI	5	2013
1	2013-11-21	64.34	64.92	63.490	64.19	16382646	TGT	11	2013
2	2017-01-19	103.62	103.62	101.770	102.08	740200	SRE	1	2017
3	2013-10-08	74.40	74.45	72.800	73.17	3148141	LYB	10	2013
4	2017-09-25	186.46	186.55	177.700	178.55	9367231	NFLX	9	2017
...	...	...	...	...	...	...	...	...	...
996	2016-11-03	71.71	72.30	71.240	71.29	1308751	MSI	11	2016
997	2015-10-14	100.19	100.47	98.339	98.99	2146508	ANDV	10	2015
998	2015-06-12	681.28	686.14	680.010	682.08	192626	AZO	6	2015
999	2015-03-18	17.32	17.84	17.300	17.76	15151248	BSX	3	2015
1000	2017-08-17	64.60	64.76	64.020	64.06	1500524	WEC	8	2017

1001 rows × 9 columns

In [32]:
```
AAL=df1[df1['Name']=='AAL']
```

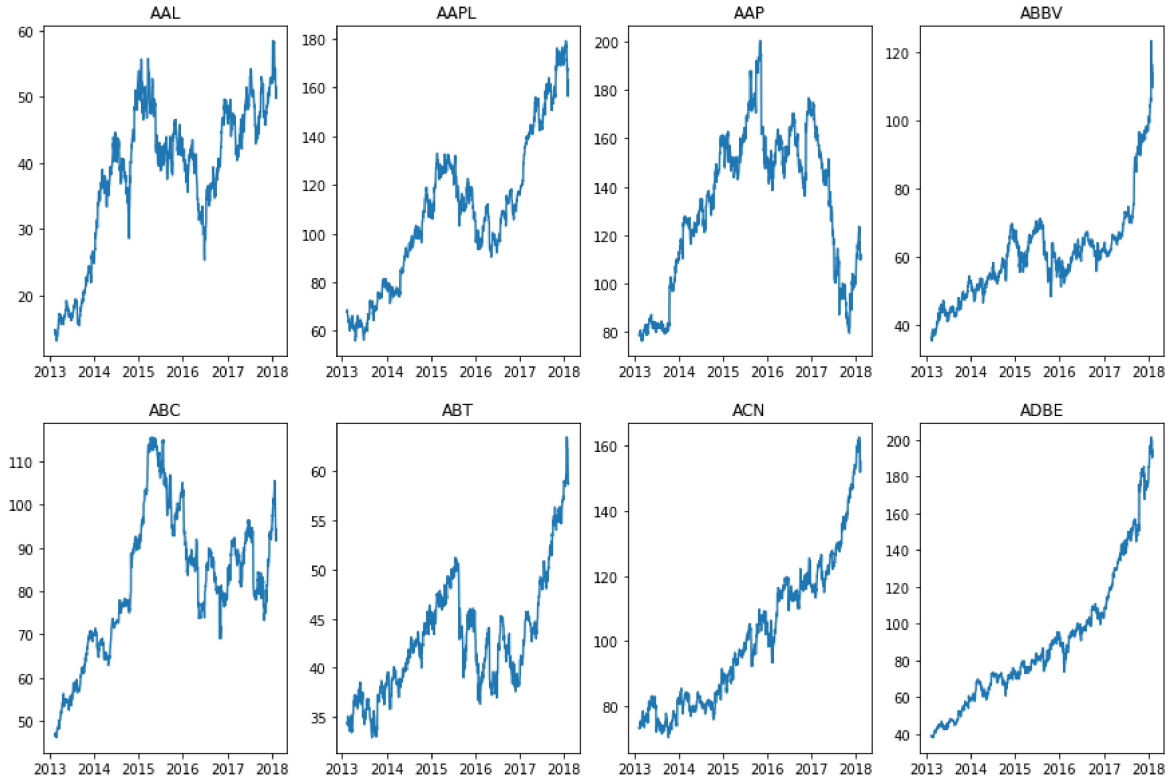
In [33]:
```
AAL
```

Out[33]:

	date	open	high	low	close	volume	Name	Month	Year
0	2013-02-08	15.07	15.12	14.63	14.75	8407500	AAL	2	2013
1	2013-02-11	14.89	15.01	14.26	14.46	8882000	AAL	2	2013
2	2013-02-12	14.45	14.51	14.10	14.27	8126000	AAL	2	2013
3	2013-02-13	14.30	14.94	14.25	14.66	10259500	AAL	2	2013
4	2013-02-14	14.94	14.96	13.16	13.99	31879900	AAL	2	2013
...	...	...	...	...	...	...	...	...	...
1254	2018-02-01	54.00	54.64	53.59	53.88	3623078	AAL	2	2018
1255	2018-02-02	53.49	53.99	52.03	52.10	5109361	AAL	2	2018
1256	2018-02-05	51.99	52.39	49.75	49.76	6878284	AAL	2	2018
1257	2018-02-06	49.32	51.50	48.79	51.18	6782480	AAL	2	2018
1258	2018-02-07	50.91	51.98	50.89	51.40	4845831	AAL	2	2018

1259 rows × 9 columns

In [34]:
```
plt.figure(figsize=(15,10))
for i, comp in enumerate(stock_list,1):
    if i < 9:
        plt.subplot(2,4,i)
        f=df1[df1['Name']==comp]
        plt.plot(f['date'],f['close'])
        plt.title(comp)
```



Quantity Volume Traded vs Time

In [35]:
```
#selecting some of the particular stock on the basis of the sector they belong to
stock_list1=['AAPL_data.csv','CAT_data.csv','GOOGL_data.csv','SO_data.csv']
```

In [36]:
```
vol_df=pd.DataFrame()
for i in stock_list1:
    itr_df=pd.read_csv(path+'/'+i)
    vol_df=pd.concat([itr_df,vol_df])
```

In [37]:
```
vol_df
```

Out[37]:

	date	open	high	low	close	volume	Name
0	2013-02-08	43.890	43.89	43.6300	43.85	2355131	SO
1	2013-02-11	43.850	44.24	43.7900	44.06	2311614	SO
2	2013-02-12	44.070	44.17	43.9100	44.16	1956351	SO
3	2013-02-13	44.120	44.29	44.0600	44.17	2331939	SO
4	2013-02-14	44.110	44.22	43.8000	43.91	3142488	SO
...	...	...	...	...	...	...	...
1254	2018-02-01	167.165	168.62	166.7600	167.78	47230787	AAPL
1255	2018-02-02	166.000	166.80	160.1000	160.50	86593825	AAPL
1256	2018-02-05	159.100	163.88	156.0000	156.49	72738522	AAPL
1257	2018-02-06	154.830	163.72	154.0000	163.03	68243838	AAPL
1258	2018-02-07	163.085	163.40	159.0685	159.54	51608580	AAPL

5036 rows × 7 columns

In [38]:
```
st_ls=vol_df.Name.unique()
```

In [39]:
```
st_ls
```

Out[39]:
```
array(['SO', 'GOOGL', 'CAT', 'AAPL'], dtype=object)
```

In [40]:
```
vol_df.sample(4)
```

Out[40]:

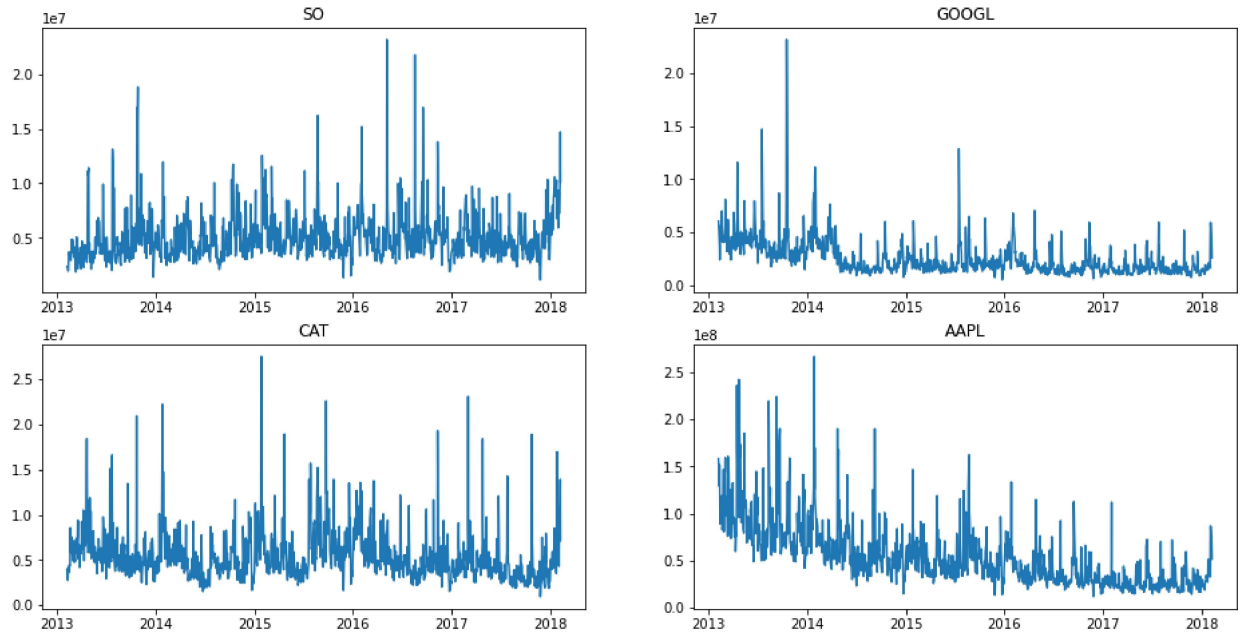
	date	open	high	low	close	volume	Name
627	2015-08-06	76.98	77.65	76.183	77.46	5609516	CAT
748	2016-01-29	94.79	97.34	94.350	97.34	64416504	AAPL
921	2016-10-05	113.40	113.66	112.690	113.05	21453089	AAPL
948	2016-11-11	93.39	94.34	91.820	93.01	7369906	CAT

In [41]:
```
vol_df['date']=pd.to_datetime(vol_df['date'])
```

In [42]:
```
vol_df.dtypes
```

Out[42]:
```
date      datetime64[ns]
open             float64
high             float64
low              float64
close            float64
volume             int64
Name              object
dtype: object
```

In [43]:
```
plt.figure(figsize=(16,8))
for i,comp in enumerate(st_ls,1):
    plt.subplot(2,2,i)
    a=vol_df[vol_df['Name']==comp]
    plt.plot(a['date'],a['volume'])
    plt.title(comp)
```

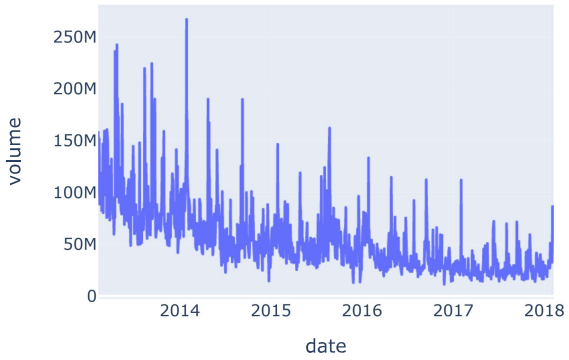


In [44]:
```
import plotly.express as px
```

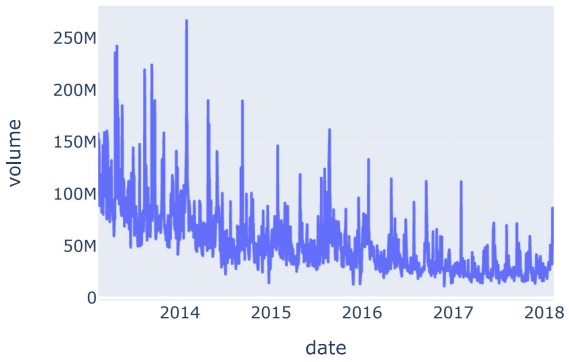
In [45]:
```
import plotly.graph_objects as go
from plotly.subplots import make_subplots
figure = make_subplots(rows=2, cols=2)
```

In [195]:
```
for company in st_ls:
    a=vol_df[vol_df['Name']==comp]
    figure=px.line(a,x='date',y='volume',title=company,width=500,height=400)
    figure.show()
```

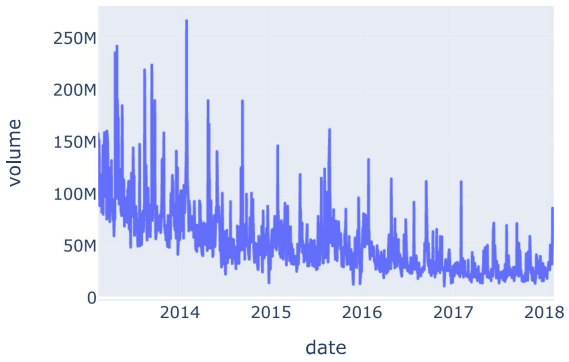
SO

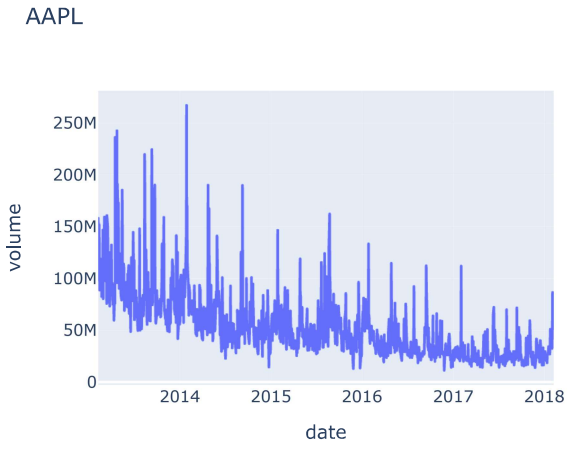


GOOGL



CAT





Daily Percentage change of Particular stock

In [47]:
```
df3=pd.read_csv(r'C:\Users\dell\Downloads\individual_stocks_5yr\AMZN_data.csv')
```

In [48]:
```
df3
```

Out[48]:

	date	open	high	low	close	volume	Name
0	2013-02-08	261.40	265.25	260.555	261.95	3879078	AMZN
1	2013-02-11	263.20	263.25	256.600	257.21	3403403	AMZN
2	2013-02-12	259.19	260.16	257.000	258.70	2938660	AMZN
3	2013-02-13	261.53	269.96	260.300	269.47	5292996	AMZN
4	2013-02-14	267.37	270.65	265.400	269.24	3462780	AMZN
...	...	...	...	...	...	...	...
1254	2018-02-01	1445.00	1459.88	1385.140	1390.00	9113808	AMZN
1255	2018-02-02	1477.39	1498.00	1414.000	1429.95	11125722	AMZN
1256	2018-02-05	1402.62	1458.98	1320.720	1390.00	11494985	AMZN
1257	2018-02-06	1361.46	1443.99	1351.790	1442.84	11066819	AMZN
1258	2018-02-07	1449.00	1460.99	1415.150	1416.78	7162741	AMZN

1259 rows × 7 columns

In [49]:
```
df3['date']=pd.to_datetime(df3['date'])
```

In [50]:
```
df3.dtypes
```

Out[50]:
```
date      datetime64[ns]
open             float64
high             float64
low              float64
close            float64
volume             int64
Name              object
dtype: object
```

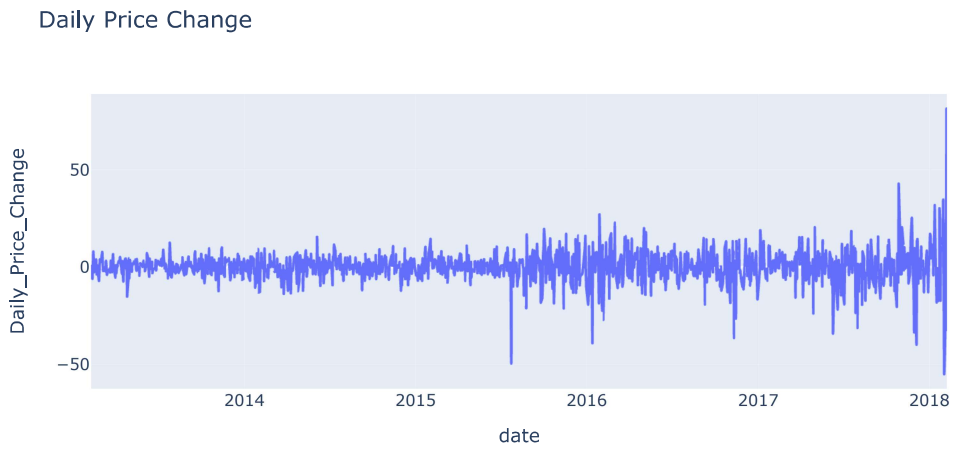
In [51]:
```
df3['Daily_Price_Change']=df3['close']-df3['open']
```

In [52]:
```
df3.sample()
```

Out[52]:

	date	open	high	low	close	volume	Name	Daily_Price_Change
565	2015-05-08	430.75	435.2	430.174	433.69	2908838	AMZN	2.94

In [196]:
```
figure=px.line(df3,x='date',y='Daily_Price_Change',title='Daily Price Change',width=800,height=400)
figure.show()
```



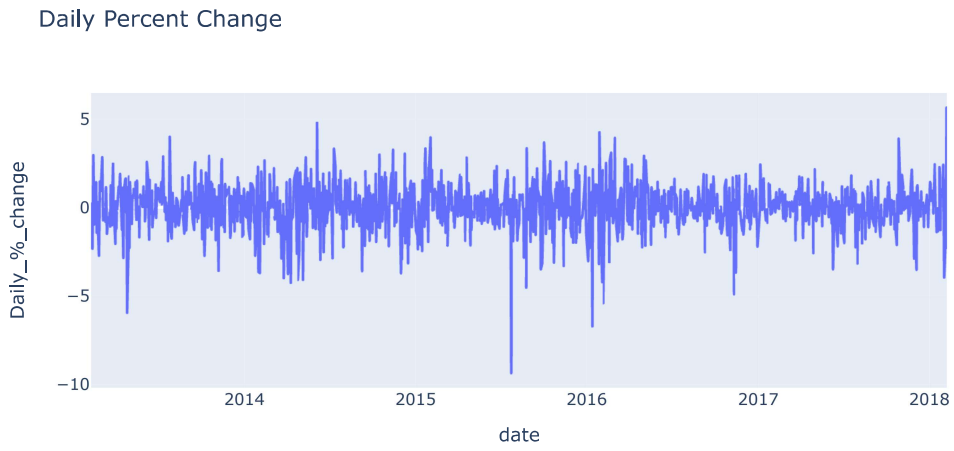
In [54]:
```
#Daily percentage return for a Amazon stock
df3['Daily_%_change']=(df3['Daily_Price_Change']/df3['close'])*100
```

In [55]:
```
df3.sample()
```

Out[55]:

	date	open	high	low	close	volume	Name	Daily_Price_Change	Daily_%_change
704	2015-11-24	674.14	675.8	661.2125	671.15	4543417	AMZN	-2.99	-0.445504

In [56]:
```
figure=px.line(df3,x='date',y='Daily_%_change',title='Daily Percent Change',width=800,height=400)
figure.show()
```



Monthly mean of closing Price

In [62]:
```
df4=df3.copy()
```

In [65]:
```
df4.dtypes
```

Out[65]:
```
date                  datetime64[ns]
open                         float64
high                         float64
low                          float64
close                        float64
volume                         int64
Name                          object
Daily_Price_Change           float64
Daily_%_change               float64
dtype: object
```

In [66]:
```
df4.set_index('date',inplace=True)
```

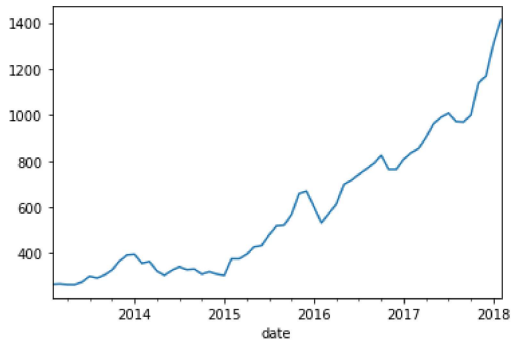
In [68]:
```
df4.head(3)
```

Out[68]:

date	open	high	low	close	volume	Name	Daily_Price_Change	Daily_%_change
2013-02-08	261.40	265.25	260.555	261.95	3879078	AMZN	0.55	0.209964
2013-02-11	263.20	263.25	256.600	257.21	3403403	AMZN	-5.99	-2.328836
2013-02-12	259.19	260.16	257.000	258.70	2938660	AMZN	-0.49	-0.189409

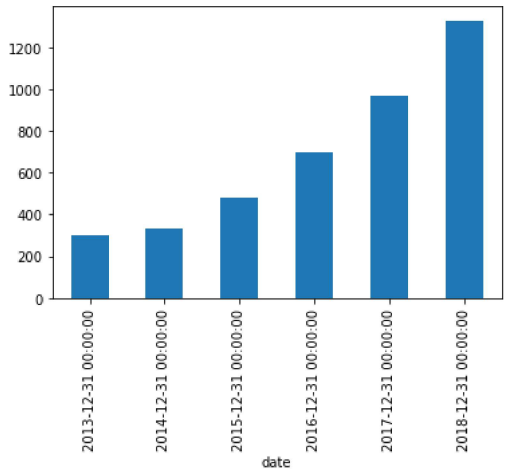
In [71]:
```
df4['close'].resample('M').mean().plot()
```

Out[71]:
```
<AxesSubplot:xlabel='date'>
```



In [74]:
```
df4['close'].resample('Y').mean().plot(kind='bar')
```

Out[74]:
```
<AxesSubplot:xlabel='date'>
```



Checking correlation

In [130]:
```
ls=vol_df['Name'].unique()
```

In [131]:
```
ls
```

Out[131]:
```
array(['SO', 'GOOGL', 'CAT', 'AAPL'], dtype=object)
```

In [82]:
```
df2=vol_df.copy()
```

In [183]:
```
df2.head(2)
```

Out[183]:

	date	open	high	low	close	volume	Name
0	2013-02-08	43.89	43.89	43.63	43.85	2355131	SO
1	2013-02-11	43.85	44.24	43.79	44.06	2311614	SO

In [186]:
```
closing_data=pd.DataFrame()
for i in df2['Name']:
    closing_data[i]=df2.loc[df2['Name']==i,'close']
```

In [187]:
```
closing_data
```

Out[187]:

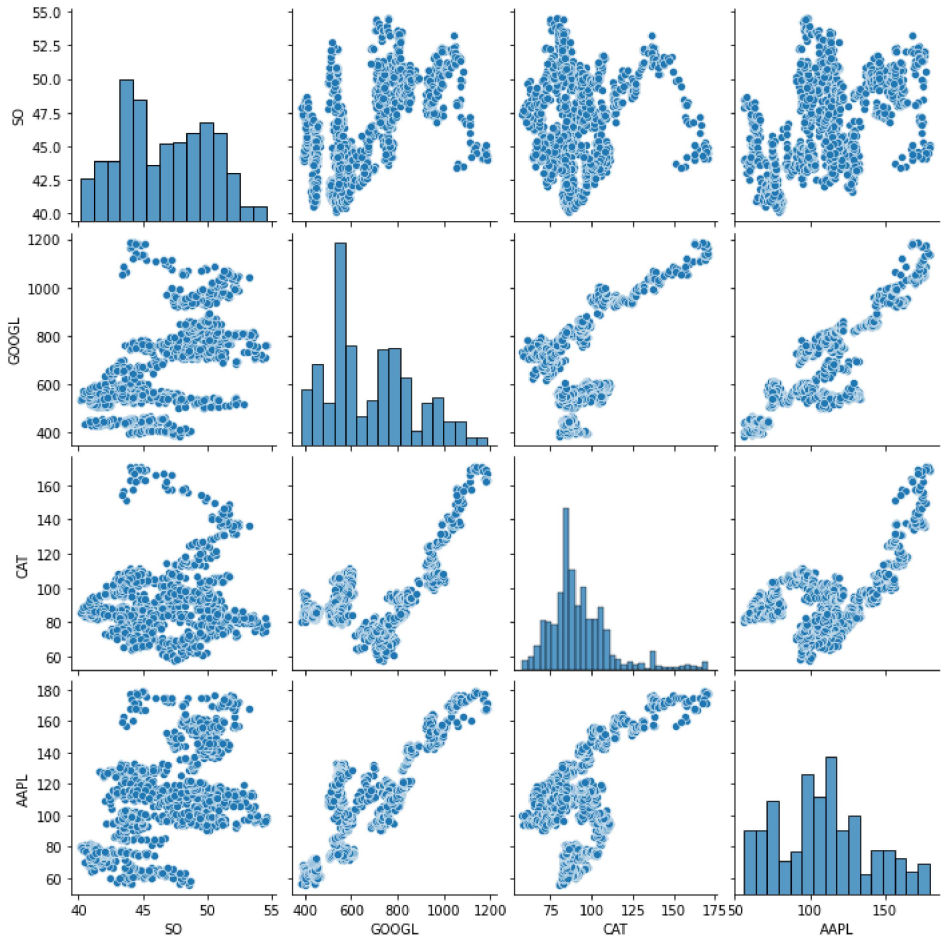
	SO	GOOGL	CAT	AAPL
0	43.85	393.0777	96.85	67.8542
1	44.06	391.6012	96.60	68.5614
2	44.16	390.7403	97.22	66.8428
3	44.17	391.8214	96.38	66.7156
4	43.91	394.3039	96.07	66.6556
...	...	...	...	...
1254	44.38	1181.5900	162.24	167.7800
1255	44.17	1119.2000	157.49	160.5000
1256	43.72	1062.3900	151.08	156.4900
1257	43.49	1084.4300	156.41	163.0300
1258	43.34	1055.4100	154.34	159.5400

1259 rows × 4 columns

In [189]:
```
import seaborn as sns
```

In [190]:
```
sns.pairplot(data=closing_data)
```

Out[190]:
```
<seaborn.axisgrid.PairGrid at 0x1fa0e335b80>
```

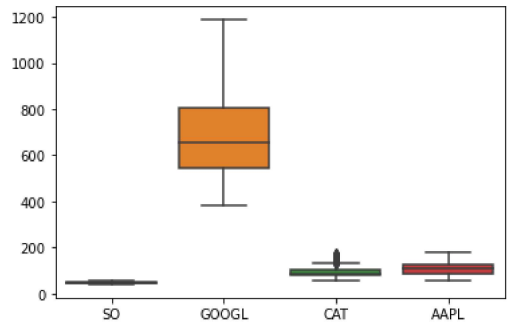


In [193]:
```
sns.heatmap(closing_data.corr(),annot=True)
```

Out[193]: <AxesSubplot:>

In [194]:
```
sns.boxplot(data=closing_data)
```

Out[194]: <AxesSubplot:>



In [ ]: