Code to scrape data for company information & get price data from yfinnace API

In [1]:

```
import pandas as pd
import numpy as np

#Data Scraping Packages
import requests
from bs4 import BeautifulSoup

#historical price downloader package
import yfinance as yf
```

In [2]:

```
#Data Scraper Code

website_text = requests.get('https://en.wikipedia.org/wiki/Dow_Jones_Industrial_Average#2010s').te
xt
soup = BeautifulSoup(website_text,'xml')

table = soup.find('table',{'class':'wikitable sortable'})
table_rows = table.find_all('tr')

data = []
for row in table_rows:
    data.append([t.text.strip() for t in row.find_all('td')])

df = pd.DataFrame(data, columns=['name', 'exchange_id', 'ticker', 'sector', 'Date Added','d'])
df.drop(['d','Date Added'], axis=1, inplace=True)
df.drop([0], axis=0, inplace=True)
df['ticker'] = df['ticker'].str.replace('NYSE:', '')
df['exchange_id'] = df['exchange_id'].str.replace('NYSE','1')
df['exchange_id'] = df['exchange_id'].str.replace('NASDAQ','2')
df['exchange_id'] = df['exchange_id'].astype(int)
df.insert(4, 'created_date', pd.datetime.now().replace(microsecond=0))
df.insert(5, 'last_updated_date', pd.datetime.now().replace(microsecond=0))

def rearrange_list(input_list, input_item_to_move, input_item_insert_here):
    '''
    Helper function to re-arrange the order of items in a list.
    Useful for moving column in pandas dataframe.

    Inputs:
        input_list - list
        input_item_to_move - item in list to move
        input_item_insert_here - item in list, insert before

    returns:
        output_list
    '''
    # make copy for output, make sure it's a list
    output_list = list(input_list)

    # index of item to move
    idx_move = output_list.index(input_item_to_move)

    # pop off the item to move
    itm_move = output_list.pop(idx_move)

    # index of item to insert here
    idx_insert = output_list.index(input_item_insert_here)

    # insert item to move into here
    output_list.insert(idx_insert, itm_move)

    return output_list

ls_cols = df.columns
ls_cols = rearrange_list(ls_cols, 'exchange_id', 'name')
ls_cols = rearrange_list(ls_cols, 'ticker', 'name')
df=df[ls_cols]
df.to_csv(r'C:\Users\Rahul Kalubowila\Desktop\Year 3\2nd Sem\Database\Stock\symbol.csv',index=True
)
df
```

Out[2]:

	exchange_id	ticker	name	sector	created_date	last_updated_date
1	1	MMM	3M	Conglomerate	2020-02-03 21:21:51	2020-02-03 21:21:51
2	1	AXP	American Express	Financial services	2020-02-03 21:21:51	2020-02-03 21:21:51
3	2	AAPL	Apple Inc.	Information technology	2020-02-03 21:21:51	2020-02-03 21:21:51
4	1	BA	Boeing	Aerospace manufacturer and Arms industry	2020-02-03 21:21:51	2020-02-03 21:21:51
5	1	CAT	Caterpillar Inc.	Construction and Mining	2020-02-03 21:21:51	2020-02-03 21:21:51
6	1	CVX	Chevron Corporation	Petroleum industry	2020-02-03 21:21:51	2020-02-03 21:21:51
7	2	CSCO	Cisco Systems	Information technology	2020-02-03 21:21:51	2020-02-03 21:21:51
8	1	KO	The Coca-Cola Company	Food industry	2020-02-03 21:21:51	2020-02-03 21:21:51
9	1	DOW	Dow Inc.	Chemical industry	2020-02-03 21:21:51	2020-02-03 21:21:51
10	1	XOM	ExxonMobil	Petroleum industry	2020-02-03 21:21:51	2020-02-03 21:21:51
11	1	GS	Goldman Sachs	Financial services	2020-02-03 21:21:51	2020-02-03 21:21:51
12	1	HD	The Home Depot	Retailing	2020-02-03 21:21:51	2020-02-03 21:21:51
13	1	IBM	IBM	Information technology	2020-02-03 21:21:51	2020-02-03 21:21:51
14	2	INTC	Intel	Information technology	2020-02-03 21:21:51	2020-02-03 21:21:51
15	1	JNJ	Johnson & Johnson	Pharmaceutical industry	2020-02-03 21:21:51	2020-02-03 21:21:51
16	1	JPM	JPMorgan Chase	Financial services	2020-02-03 21:21:51	2020-02-03 21:21:51
17	1	MCD	McDonald's	Food industry	2020-02-03 21:21:51	2020-02-03 21:21:51
18	1	MRK	Merck & Co.	Pharmaceutical industry	2020-02-03 21:21:51	2020-02-03 21:21:51
19	2	MSFT	Microsoft	Information technology	2020-02-03 21:21:51	2020-02-03 21:21:51
20	1	NKE	Nike	Apparel	2020-02-03 21:21:51	2020-02-03 21:21:51
21	1	PFE	Pfizer	Pharmaceutical industry	2020-02-03 21:21:51	2020-02-03 21:21:51
22	1	PG	Procter & Gamble	Fast moving consumer goods	2020-02-03 21:21:51	2020-02-03 21:21:51
23	1	TRV	The Travelers Companies	Financial services	2020-02-03 21:21:51	2020-02-03 21:21:51
24	1	UNH	UnitedHealth Group	Managed health care	2020-02-03 21:21:51	2020-02-03 21:21:51
25	1	UTX	United Technologies	Conglomerate	2020-02-03 21:21:51	2020-02-03 21:21:51
26	1	VZ	Verizon	Telecommunication	2020-02-03 21:21:51	2020-02-03 21:21:51
27	1	V	Visa Inc.	Financial services	2020-02-03 21:21:51	2020-02-03 21:21:51
28	1	WMT	Walmart	Retailing	2020-02-03 21:21:51	2020-02-03 21:21:51
29	2	WBA	Walgreens Boots Alliance	Retailing	2020-02-03 21:21:51	2020-02-03 21:21:51
30	1	DIS	The Walt Disney Company	Broadcasting and entertainment	2020-02-03 21:21:51	2020-02-03 21:21:51

In [3]:

```
#Code to download Historical Prices using yfinance API

tickers = df['ticker']
tickers.drop(tickers.index[8], inplace=True)
tickers.reset_index(drop=True, inplace=True)
tickers.index = np.arange(1, len(tickers) + 1)

data = pd.DataFrame ()

for ticker in tickers:
# Get the data for the stock Apple by specifying the stock ticker, start date, and end date
    data = data.append(yf.download(ticker,'2019-02-28','2019-02-28'))

sample = data.copy()
```

```
sample['price_date'] = sample.index
sample.reset_index(drop=True, inplace=True)
sample.index = np.arange(1, len(sample) + 1)
sample['ticker'] = tickers
sample['data_vendor_id'] = 1

sample.to_csv(r'C:\Users\Rahul Kalubowila\Desktop\Year 3\2nd Sem\Database\Stock\daily_price.csv',i
ndex=True)
sample
```

```
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
```

Out[3]:

	Open	High	Low	Close	Adj Close	Volume	price_date	ticker	data_vendor_id
1	209.050003	209.050003	206.960007	207.389999	202.017090	2121700	2019-02-28	MMM	1
2	107.220001	108.449997	107.080002	107.739998	106.249207	3295800	2019-02-28	AXP	1
3	174.320007	174.910004	172.919998	173.149994	171.318542	28215400	2019-02-28	AAPL	1
4	438.700012	441.420013	437.079987	439.959991	432.211761	5063100	2019-02-28	BA	1
5	138.550003	139.000000	137.139999	137.339996	133.477356	4024000	2019-02-28	CAT	1
6	120.000000	120.160004	118.489998	119.580002	116.066391	6406500	2019-02-28	CVX	1
7	51.549999	51.959999	51.349998	51.770000	50.367107	30708500	2019-02-28	CSCO	1
8	45.119999	45.610001	45.110001	45.340000	43.947472	22533900	2019-02-28	KO	1
9	79.449997	79.750000	78.709999	79.029999	76.255180	14597900	2019-02-28	XOM	1
10	198.039993	198.399994	196.009995	196.699997	193.505081	2964300	2019-02-28	GS	1
11	183.600006	185.190002	183.110001	185.139999	180.231155	8060600	2019-02-28	HD	1
12	138.770004	139.059998	137.720001	138.130005	133.343857	3457800	2019-02-28	IBM	1
13	52.919998	53.180000	52.810001	52.959999	51.998444	18388800	2019-02-28	INTC	1
14	135.949997	137.949997	135.690002	136.639999	133.775223	10133200	2019-02-28	JNJ	1
15	105.010002	105.209999	104.180000	104.360001	101.371384	15156900	2019-02-28	JPM	1
16	182.130005	183.979996	182.009995	183.839996	180.650620	3297900	2019-02-28	MCD	1
17	80.529999	81.760002	80.529999	81.290001	79.134270	11057600	2019-02-28	MRK	1
18	112.040001	112.879997	111.730003	112.029999	110.869133	29083900	2019-02-28	MSFT	1
19	86.080002	86.400002	85.680000	85.730003	84.831573	4988100	2019-02-28	NKE	1
20	42.950001	43.779999	42.869999	43.349998	41.736805	37004200	2019-02-28	PFE	1
21	98.910004	99.199997	98.080002	98.550003	96.041931	10575400	2019-02-28	PG	1

	Open	High	Low	Close	Adj Close	Volume	price_date	ticker	data_vendor_id
22	132.580002	133.070007	132.220001	132.910004	129.909866	1739700	2019-02-28	TRV	1
23	249.699997	251.949997	239.149994	242.220001	238.191849	11034300	2019-02-28	UNH	1
24	126.690002	127.330002	125.570000	125.669998	123.625832	4416200	2019-02-28	UTX	1
25	56.740002	57.610001	56.720001	56.919998	54.610676	15743000	2019-02-28	VZ	1
26	147.259995	148.820007	147.229996	148.119995	147.432495	6250200	2019-02-28	V	1
27	98.110001	99.470001	97.769997	98.989998	97.024994	11375900	2019-02-28	WMT	1
28	71.410004	71.879997	70.970001	71.190002	69.435623	7775000	2019-02-28	WBA	1
29	112.900002	113.430000	112.750000	112.839996	111.477608	6716700	2019-02-28	DIS	1

In [4]:

```
sample.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 29 entries, 1 to 29
Data columns (total 9 columns):
Open              29 non-null float64
High              29 non-null float64
Low               29 non-null float64
Close             29 non-null float64
Adj Close         29 non-null float64
Volume            29 non-null int64
price_date        29 non-null datetime64[ns]
ticker            29 non-null object
data_vendor_id    29 non-null int64
dtypes: datetime64[ns](1), float64(5), int64(2), object(1)
memory usage: 2.3+ KB
```