DSC540 Final

January 9, 2021

0.0.1 Michael Hotaling DSC540: Term Project 12/24/2020

0.1 Milestone 1 (Weeks 3 & 4)

0.1.1 Identify Datasets

For my project, I'll be gathering information on insider trading among publically traded companies. Illegal insider trading is one of many types of financial frauds where insiders, defined as board members, executives and 10% share owners, use non-public information to buy or sell securities.

I indend to find relationships between all three datasets to determine if these shareholders may be using insider information to make financial decisions, something that is illegal to do, but is still very common.

I will first find the transactions by these members and determine how much money the transfer between these stocks. Then, using historical data, find out whether or not these transactions were beneficial. Then, I'll analyze the quarterly reports for the following quarter to determine whether or not the company showed any financial loss or gain, which would drive the price. This will be my primary method of collecting the originally non-public information.

0.1.2 Web Scraping

The SEC requires all executives, board members, and 10% share owners to disclose their security transactions via Form 4. These forms are available on the SEC's Electronic Data Gathering, Analysis, and Retrieval system (EDGAR) and are added to the system in real time. _____

SEC Form 4

FORM 4

UNITED STATES SECURITIES AND EXCHANGE COMMISSION
Washington, D.C. 20549

OMB APPROVAL	
OMB Number:	3235-0287
Estimated average burden hours per response:	0.5

☐ Check this box if no longer subject to Section 16. Form 4 or Form 5 obligations may continue. See Instruction 1(b).

STATEMENT OF CHANGES IN BENEFICIAL OWNERSHIP

Filed pursuant to Section 16(a) of the Securities Exchange Act of 1934 or Section 30(h) of the Investment Company Act of 1940

1. Name and Address of Reporting Person*	2. Issuer Name and Ticker or Trading Symbol	5. Relationship of Reporting Person(s) to Issuer (Check all applicable)
BEZOS JEFFREY P	AMAZON COM INC [AMZN]	X Director; X 10% Owner; X Officer (give title below); Other (specify below)
(Last) (First) (Middle): P.O. BOX 81226	3. Date of Earliest Transaction (Month/Day/Year): 11/02/2020	Chairman, CEO and President
(Street): SEATTLE WA 98108-1226	4. If Amendment, Date of Original Filed (Month/Day/Year)	6. Individual or Joint/Group Filing (Check Applicable Line): X Form filed by One Reporting Person; Form filed by More than One Reporting Person
(City) (State) (Zip)		

Table I - Non-Derivative Securities Acquired, Disposed of, or Beneficially Owned

1. Title of Security (Instr. 3)	2. Transaction Date (Month/Day/Year)	2A. Deemed Execution Date, if any (Month/Day/Year)	3. Transaction Code (Instr. 8)		4. Securities Acquired (A) or Disposed Of (D) (Instr. 3, 4 and 5)			5. Amount of Securities Beneficially Owned Following Reported Transaction(s) (Instr. 3 and 4)	6. Ownership Form: Direct (D) or Indirect (I) (Instr. 4)	7. Nature of Indirect Beneficial Ownership (Instr. 4)
			Code	V	Amount	(A) or (D)	Price			
Common Stock, par value $.01 per share	11/02/2020		S[1]		3,312	D	$3,053.7284[2]	53,997,103	D	
Common Stock, par value $.01 per share	11/02/2020		S[1]		1,187	D	$3,055.1306[3]	53,995,916	D	
Common Stock, par value $.01 per share	11/02/2020		S[1]		698	D	$3,056.3361[4]	53,995,218	D	
Common Stock, par value $.01 per share	11/02/2020		S[1]		2,479	D	$3,056.8272[5]	53,992,739	D	
Common Stock, par value $.01 per share	11/02/2020		S[1]		1,100	D	$3,058.0118[6]	53,991,639	D	
Common Stock, par value $.01 per share	11/02/2020		S[1]		5,909	D	$3,059.8118[7]	53,985,730	D	
Common Stock, par value $.01 per share	11/02/2020		S[1]		12,487	D	$3,061.7927[8]	53,973,243	D	
Common Stock, par value $.01 per share	11/02/2020		S[1]		2,809	D	$3,062.9528[9]	53,970,434	D	
Common Stock, par value $.01 per share	11/02/2020		S[1]		1,603	D	$3,064.1062[10]	53,968,831	D	
Common Stock, par value $.01 per share	11/02/2020		S[1]		5,500	D	$3,065.5282[11]	53,963,331	D	
Common Stock, par value $.01 per share	11/02/2020		S[1]		1,700	D	$3,067.57[12]	53,961,631	D	
Common Stock, par value $.01 per share	11/02/2020		S[1]		7,102	D	$3,069.1949[13]	53,954,529	D	
Common Stock, par value $.01 per share	11/02/2020		S[1]		1,644	D	$3,069.8093[14]	53,952,885	D	
Common Stock, par value $.01 per share	11/02/2020		S[1]		3,200	D	$3,071.6756[15]	53,949,685	D	
Common Stock, par value $.01 per share	11/02/2020		S[1]		1,957	D	$3,073.1263[16]	53,947,728	D	
Common Stock, par value $.01 per share	11/02/2020		S[1]		900	D	$3,075.1944[17]	53,946,828	D	
Common Stock, par value $.01 per share	11/02/2020		S[1]		400	D	$3,079.05	53,946,428	D	
Common Stock, par value $.01 per share	11/03/2020		S[1]		8,404	D	$2,983.3668[18]	53,938,024	D	
Common Stock, par value $.01 per share	11/03/2020		S[1]		3,443	D	$2,984.4844[19]	53,934,581	D	
Common Stock, par value $.01 per share	11/03/2020		S[1]		7,121	D	$2,985.6959[20]	53,927,460	D	

The SEC provides this data in HTML and TXT formats and allows web scraping for these files. Many websites also have this information available in an easier to access format, but with a delay. I'll be attempting to gather the data directly from the SEC, but if this is not possible within the scope of this class, I'll be scraping the data from Openinsider

[1]:
```
import pandas as pd

ticker = "AMZN"

df = pd.DataFrame()
url = 'http://openinsider.com/screener?
 ↪s={}&o=&pl=&ph=&ll=&lh=&fd=730&fdr=&td=0&tdr=&fdlyl=&fdlyh=&daysago=&xp=1&xs=1&vl=&vh=&ocl=
 ↪format(ticker)
```

```
for i in range(1,100):
    tab = pd.read_html(url+str(i))
    if tab[11].columns[1] == 1:
        break
    df = pd.concat([df,tab[11]])

df = df.reset_index()
df
```

[1]:

	index	X	Filing Date	Trade Date	Ticker	\
0	0	NaN	2021-01-07 16:11:50	2021-01-05	AMZN	
1	1	NaN	2020-12-03 16:48:47	2020-12-01	AMZN	
2	2	NaN	2020-11-25 16:51:21	2020-11-23	AMZN	
3	3	NaN	2020-11-24 17:04:23	2020-11-20	AMZN	
4	4	D	2020-11-17 17:29:49	2020-11-16	AMZN	
..	…	…	…	…	…	
104	4	D	2019-02-20 18:08:54	2019-02-15	AMZN	
105	5	D	2019-02-20 18:08:16	2019-02-15	AMZN	
106	6	D	2019-02-20 18:07:39	2019-02-15	AMZN	
107	7	D	2019-02-20 18:06:55	2019-02-15	AMZN	
108	8	D	2019-02-20 18:05:41	2019-02-15	AMZN	

	Insider Name	Title	Trade Type	Price	\
0	Rubinstein Jonathan	Dir	S - Sale	$3,166.01	
1	Wilke Jeffrey A	CEO Worldwide Consumer	S - Sale	$3,200.59	
2	Wilke Jeffrey A	CEO Worldwide Consumer	S - Sale	$3,091.15	
3	Huttenlocher Daniel P	Dir	S - Sale	$3,114.79	
4	Wilke Jeffrey A	CEO Worldwide Consumer	S - Sale+OE	$3,114.36	
..	…	…	…	…	
104	Blackburn Jeffrey M	SVP	S - Sale+OE	$1,616.34	
105	Jassy Andrew R	CEO Amazon Web Services	S - Sale+OE	$1,618.79	
106	Olsavsky Brian T	SVP, CFO	S - Sale+OE	$1,627.86	
107	Zapolsky David	SVP	S - Sale+OE	$1,619.37	
108	Reynolds Shelley	VP	S - Sale+OE	$1,621.42	

	Qty	Owned	ΔOwn	Value	1d	1w	1m	6m
0	-314	6758	-4%	-$994,127	NaN	NaN	NaN	NaN
1	-2000	48858	-4%	-$6,401,175	NaN	NaN	NaN	NaN
2	-750	50858	-1%	-$2,318,360	NaN	NaN	NaN	NaN
3	-172	950	-15%	-$535,744	NaN	NaN	NaN	NaN
4	-1200	51608	-2%	-$3,737,229	NaN	NaN	NaN	NaN
..	…	…	…	…	..	..	..	..
104	-2055	67459	-3%	-$3,321,570	NaN	NaN	NaN	NaN
105	-1644	95568	-2%	-$2,661,295	NaN	NaN	NaN	NaN
106	-2030	1068	-66%	-$3,304,556	NaN	NaN	NaN	NaN
107	-1015	2851	-26%	-$1,643,665	NaN	NaN	NaN	NaN

```
108  -437   6122   -7%    -$708,561 NaN NaN NaN NaN

[109 rows x 17 columns]
```

0.1.3 CSV Files

Another important side of detecting insider information is evaluating whether or not these transactions were fiscally beneficial for these members. Since our Form 4 data only shows a snapshot of how many shares and the total value of the transactions, we will need an outside data source to provide the share price information before and after the transactions were made.

Yahoo Finance offers downloadable CSV files on their website and lets us pick our time ranges to select how much data we would like to download. We can directly import this CSV file into pandas using `pd.read_csv(url)`

[2]:
```
import time
import datetime

def date_to_stamp(s):
    return round(time.mktime(datetime.datetime.strptime(s, "%m/%d/%Y").
 ↪timetuple()))
```

[3]:
```
ticker = "AMZN"
start_date = '1/10/2010'
end_date = datetime.date.today().strftime("%m/%d/%Y")

url = "https://query1.finance.yahoo.com/v7/finance/download/{}?
 ↪period1={}&period2={}&interval=1d&events=history&includeAdjustedClose=true".
 ↪format(ticker,date_to_stamp(start_date), date_to_stamp(end_date))

df = pd.read_csv(url)
df
```

[3]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2010-01-11	132.619995	132.800003	129.210007	130.309998	130.309998	8779400
1	2010-01-12	128.990005	129.820007	126.550003	127.349998		
2	2010-01-13	127.900002	129.710007	125.750000	129.110001		
3	2010-01-14	129.139999	130.380005	126.400002	127.349998		
4	2010-01-15	129.179993	129.649994	127.059998	127.139999		
…	…	…	…	…	…		
2764	2021-01-04	3270.000000	3272.000000	3144.020020	3186.629883		
2765	2021-01-05	3166.010010	3223.379883	3165.060059	3218.510010		
2766	2021-01-06	3146.479980	3197.510010	3131.159912	3138.379883		
2767	2021-01-07	3157.000000	3208.540039	3155.000000	3162.159912		
2768	2021-01-08	3180.000000	3190.639893	3142.199951	3182.699951		

```
1       127.349998   9096300
2       129.110001  10723200
3       127.349998   9774900
4       127.139999  15376500
…              …         …
2764   3186.629883   4411400
2765   3218.510010   2655500
2766   3138.379883   4394800
2767   3162.159912   3514500
2768   3182.699951   3534300

[2769 rows x 7 columns]
```

We can add some common 'technical indicators' such as 50 day moving averages and 200 day moving averages to our dataset to help us determine the general momentum of the stock as well as add Bollinger Bands which are commonly used to evaluate the volatility of a stock

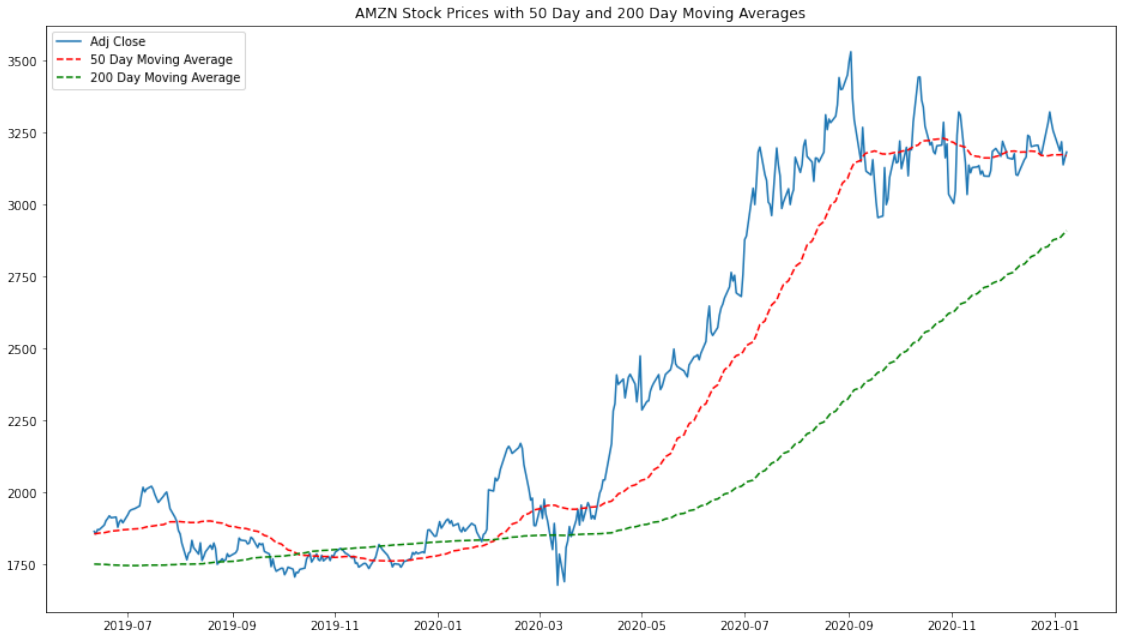
[4]:
```
# Adding Moving Averages
df['50MA'] = df['Adj Close'].rolling(window=50).mean()
df['200MA'] = df['Adj Close'].rolling(window=200).mean()

# Adding Bollinger Bands
df['middle_band'] = df['Adj Close'].rolling(window=20).mean()
df['upper_band'] = df['Adj Close'].rolling(window=20).mean() + df['Adj Close'].
 ↪rolling(window=20).std()*2
df['lower_band'] = df['Adj Close'].rolling(window=20).mean() - df['Adj Close'].
 ↪rolling(window=20).std()*2

df
```

[4]:
```
            Date         Open         High          Low        Close  \
0     2010-01-11   132.619995   132.800003   129.210007   130.309998
1     2010-01-12   128.990005   129.820007   126.550003   127.349998
2     2010-01-13   127.900002   129.710007   125.750000   129.110001
3     2010-01-14   129.139999   130.380005   126.400002   127.349998
4     2010-01-15   129.179993   129.649994   127.059998   127.139999
…            …            …            …            …            …
2764  2021-01-04  3270.000000  3272.000000  3144.020020  3186.629883
2765  2021-01-05  3166.010010  3223.379883  3165.060059  3218.510010
2766  2021-01-06  3146.479980  3197.510010  3131.159912  3138.379883
2767  2021-01-07  3157.000000  3208.540039  3155.000000  3162.159912
2768  2021-01-08  3180.000000  3190.639893  3142.199951  3182.699951

        Adj Close    Volume         50MA        200MA  middle_band  \
0      130.309998   8779400          NaN          NaN          NaN
1      127.349998   9096300          NaN          NaN          NaN
2      129.110001  10723200          NaN          NaN          NaN
```

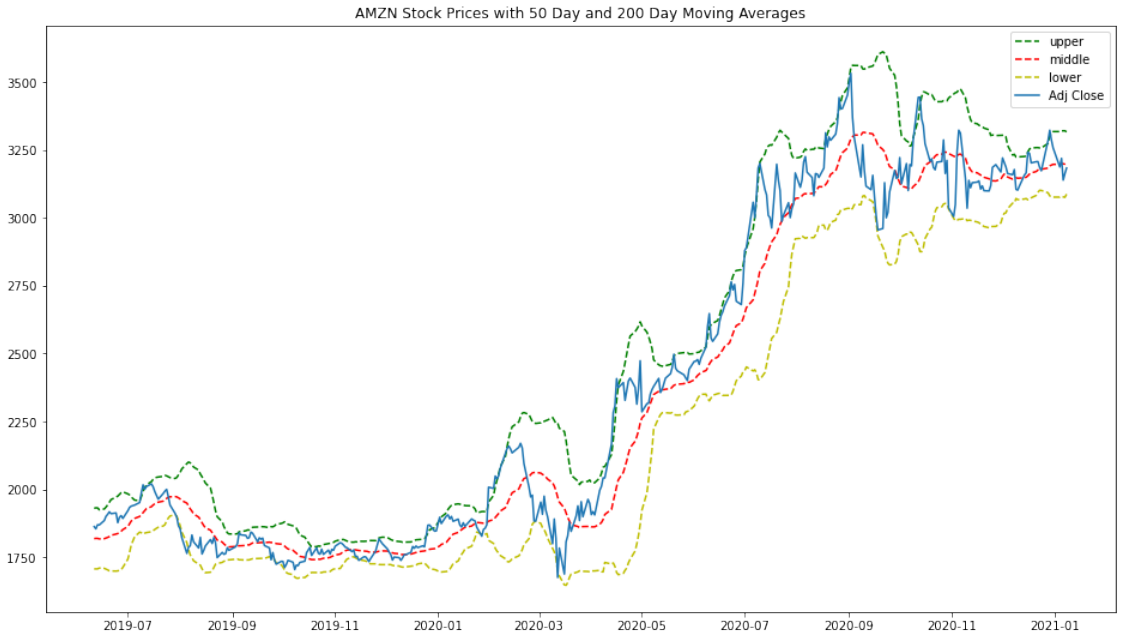
```
3      127.349998   9774900          NaN          NaN          NaN
4      127.139999  15376500          NaN          NaN          NaN
…           …       …          …           …           …
2764  3186.629883   4411400  3173.147993  2883.849697  3196.339490
2765  3218.510010   2655500  3173.990195  2890.711797  3199.135986
2766  3138.379883   4394800  3172.669795  2896.889546  3198.154981
2767  3162.159912   3514500  3171.772192  2902.999846  3197.398474
2768  3182.699951   3534300  3169.699590  2909.484146  3201.323474

       upper_band   lower_band
0             NaN          NaN
1             NaN          NaN
2             NaN          NaN
3             NaN          NaN
4             NaN          NaN
…           …          …
2764  3316.949095  3075.729885
2765  3319.041336  3079.230637
2766  3319.786026  3076.523935
2767  3319.761937  3075.035011
2768  3315.887040  3086.759908

[2769 rows x 12 columns]
```

[5]:
```
import matplotlib.pyplot as plt
```

[6]:
```
plt.figure(figsize=(16,9))
plt.title("{} Stock Prices with 50 Day and 200 Day Moving Averages".
 ↪format(ticker))
plt.plot(pd.to_datetime(df['Date']).iloc[-400:], df['Adj Close'].iloc[-400:],␣
 ↪label="Adj Close")
plt.plot(pd.to_datetime(df['Date']).iloc[-400:], df['50MA'].iloc[-400:], 'r--',␣
 ↪label = "50 Day Moving Average" )
plt.plot(pd.to_datetime(df['Date']).iloc[-400:], df['200MA'].iloc[-400:],␣
 ↪'g--', label = "200 Day Moving Average" );
plt.legend()
plt.show()
```



[7]:
```
plt.figure(figsize=(16,9))
plt.title("{} Stock Prices with 50 Day and 200 Day Moving Averages".
 ↪format(ticker))
plt.plot(pd.to_datetime(df['Date']).iloc[-400:], df['upper_band'].iloc[-400:],␣
 ↪'g--', label="upper")
plt.plot(pd.to_datetime(df['Date']).iloc[-400:], df['middle_band'].iloc[-400:],␣
 ↪'r--', label="middle")
plt.plot(pd.to_datetime(df['Date']).iloc[-400:], df['lower_band'].iloc[-400:],␣
 ↪'y--', label="lower")
plt.plot(pd.to_datetime(df['Date']).iloc[-400:], df['Adj Close'].iloc[-400:],␣
 ↪label="Adj Close")
plt.legend()
plt.show()
```



0.1.4 API Source

Finally, we will need to determine whether or not non-public information gave these insiders an advantage. Each fiscal year and quarter, publically traded companies must disclose their finances to the SEC. These yearly and quarterly reports (10-K and 10-Q) contain information about revenue, cash flows, liabilities, and several other metrics that are extremely impactful to the value of a company, and therefore, the stock prices. If time allows, I'd also like to utilize news articles to determine whether or not the company had any recent scandels. I can pull some URLs off of a website like Twitter using a list of known news outlet handles and then processing the website text for sentiment analysis.

I originally wanted to scrape the financial data from the SEC by accessing the `master.idx` file, which contains all the filings and locations where they are stored. This ended up being a lot more difficult than I imagined. I left my work in here because it took me over 100 hrs to figure out, so I got a little attached to it

[8]:
```
import os

years = range(2000,2021)
quarters = ['QTR1', 'QTR2', 'QTR3', 'QTR4']
base_path = 'C:\\Users\\hotal\\Documents\\DSC540\\Finals'
current_dirs = os.listdir(path=base_path)

for yr in years:
    if str(yr) not in current_dirs:
        os.mkdir('/'.join([base_path, str(yr)]))
    current_files = os.listdir('/'.join([base_path, str(yr)]))
```

```
    for qtr in quarters:
        local_filename =  f'xbrl-index-{yr}-{qtr}.txt'
        local_file_path = '/'.join([base_path, str(yr), local_filename])
        if local_filename in current_files:
            continue
        url = f'https://www.sec.gov/Archives/edgar/full-index/{yr}/{qtr}/xbrl.
↪idx'
        r = requests.get(url, stream=True)
        with open(local_file_path, 'wb') as f:
            for chunk in r.iter_content(chunk_size=10240):
                f.write(chunk)
        time.sleep(1)
```

Before continuing, we need to convert our ticker into a CIK value, which is unique to each security. CIK values are always consistant, even if a companies ticker value changes. A dictionary of ticker values is available on the SEC website, were we can download it and extract the CIK value.

[9]:
```
def ticker_downloader():
    tickers = pd.read_table("https://www.sec.gov/include/ticker.txt", names =␣
↪['Ticker', "CIK"])
    tickers.to_csv("ticker.csv")

def ticker_to_cik(ticker):
    """Returns the CIK value for a given ticker"""
    tickers = pd.read_csv("ticker.csv")
    ticker = ticker.replace(".","-")
    return int(tickers[tickers['Ticker'] == ticker.lower()]["CIK"])

ticker_downloader()
```

[10]:
```
ticker_to_cik(ticker)
```

[10]: 1018724

[11]:
```
super_frame = pd.DataFrame(columns = ["CIK", "Company Name", "Form Type","Date␣
↪Filed","Filename"])

for i in range(2000,2021):
    for j in range(1,5):
        df = pd.read_table("{}/xbrl-index-{}-QTR{}.txt".format(i,i,j),sep="|",␣
↪skiprows=10,
                           names = ["CIK", "Company Name", "Form Type","Date␣
↪Filed","Filename"])
        super_frame = super_frame.append(df, ignore_index=True)
```

```
df = super_frame[((super_frame['Form Type'] == "10-K")|(super_frame['Form␣
 ↪Type'] == "10-Q")) & (super_frame['CIK'] == ticker_to_cik(ticker))]
df.tail(10)
```

[11]:

	CIK	Company Name	Form Type	Date Filed	Filename
303928	1018724	AMAZON COM INC	10-Q	2018-07-27	edgar/data/1018724/0001018724-18-000108.txt
311616	1018724	AMAZON COM INC	10-Q	2018-10-26	edgar/data/1018724/0001018724-18-000159.txt
319365	1018724	AMAZON COM INC	10-K	2019-02-01	edgar/data/1018724/0001018724-19-000004.txt
326782	1018724	AMAZON COM INC	10-Q	2019-04-26	edgar/data/1018724/0001018724-19-000043.txt
335637	1018724	AMAZON COM INC	10-Q	2019-07-26	edgar/data/1018724/0001018724-19-000071.txt
346417	1018724	AMAZON COM INC	10-Q	2019-10-25	edgar/data/1018724/0001018724-19-000089.txt
360739	1018724	AMAZON COM INC	10-K	2020-01-31	edgar/data/1018724/0001018724-20-000004.txt
376325	1018724	AMAZON COM INC	10-Q	2020-05-01	edgar/data/1018724/0001018724-20-000010.txt
395062	1018724	AMAZON COM INC	10-Q	2020-07-31	edgar/data/1018724/0001018724-20-000021.txt
411570	1018724	AMAZON COM INC	10-Q	2020-10-30	edgar/data/1018724/0001018724-20-000030.txt

Now, we have the current directory of each filing. We can then parse through the information to extract the data we need

[12]:
```
urls = "https://www.sec.gov/Archives/" + df['Filename']
for ind, i in enumerate(urls):
    print(i)
    if ind > 4:
        break
```

```
https://www.sec.gov/Archives/edgar/data/1018724/0001193125-09-154174.txt
https://www.sec.gov/Archives/edgar/data/1018724/0001193125-09-212134.txt
https://www.sec.gov/Archives/edgar/data/1018724/0001193125-10-016098.txt
https://www.sec.gov/Archives/edgar/data/1018724/0001193125-10-090367.txt
https://www.sec.gov/Archives/edgar/data/1018724/0001193125-10-164083.txt
https://www.sec.gov/Archives/edgar/data/1018724/0001193125-10-233883.txt
```

[13]:
```
urls_json = []

for i in urls:
    urls_json.append(i.replace("-","").replace(".txt",'/index.json'))
```

```
for ind, i in enumerate(urls_json):
    print(i)
    if ind > 4:
        break
```

```
https://www.sec.gov/Archives/edgar/data/1018724/000119312509154174/index.json
https://www.sec.gov/Archives/edgar/data/1018724/000119312509212134/index.json
https://www.sec.gov/Archives/edgar/data/1018724/000119312510016098/index.json
https://www.sec.gov/Archives/edgar/data/1018724/000119312510090367/index.json
https://www.sec.gov/Archives/edgar/data/1018724/000119312510164083/index.json
https://www.sec.gov/Archives/edgar/data/1018724/000119312510233883/index.json
```

[14]:
```
import requests

content = requests.get(urls_json[-1]).json()

for file in content['directory']['item']:
    if file['name'] == 'FilingSummary.xml':
        xml_summary = "https://www.sec.gov" + content['directory']['name'] + "/
 ↪" + file['name']
        print(xml_summary)
```

```
https://www.sec.gov/Archives/edgar/data/1018724/000101872420000030/FilingSummary
.xml
```

[15]:
```
from bs4 import BeautifulSoup

base_url = xml_summary.replace('FilingSummary.xml',"")
content = requests.get(xml_summary).content
soup = BeautifulSoup(content, 'lxml')
reports = soup.find('myreports')
```

[16]:
```
master_reports = []

for report in reports.find_all('report')[:-1]:
    report_dict = {}
    report_dict['shortname'] = report.shortname.text
    report_dict['longname'] = report.longname.text
    report_dict['position'] = report.position.text
    report_dict['menucategory'] = report.menucategory.text
    report_dict['filename'] = base_url + report.htmlfilename.text

    master_reports.append(report_dict)
```

[17]:
```
testdf = pd.DataFrame(master_reports)
testdf[testdf['menucategory'] == 'Statements']['shortname']
```

[17]:
```
1                Consolidated Statements of Cash Flows
2                Consolidated Statements of Operations
3      Consolidated Statements of Comprehensive Income
4    Consolidated Statements of Comprehensive Incom…
5                          Consolidated Balance Sheets
6          Consolidated Balance Sheets (Parenthetical)
Name: shortname, dtype: object
```

[18]:
```
statements_url = []

for report_dict in master_reports:
    """Everything is named differently so I'll need to do some fancy regex work␣
 ↪to get this to work"""

    item1 = "Consolidated Balance Sheets".upper()
    item2 = "Consolidated Statements of Comprehensive Income".upper()
    item3 = "Consolidated Statements of Cash Flows".upper()

    report_list = [item1,item2,item3]

    if report_dict['shortname'].upper() in report_list:
        print(report_dict['shortname'])
        print(report_dict['filename'])
        statements_url.append(report_dict['filename'])
```

```
Consolidated Statements of Cash Flows
https://www.sec.gov/Archives/edgar/data/1018724/000101872420000030/R2.htm
Consolidated Statements of Comprehensive Income
https://www.sec.gov/Archives/edgar/data/1018724/000101872420000030/R4.htm
Consolidated Balance Sheets
https://www.sec.gov/Archives/edgar/data/1018724/000101872420000030/R6.htm
```

[19]:
```
statements_data = []

for statement in statements_url:
    statement_data = {}
    statement_data['header'] = []
    statement_data['section'] = []
    statement_data['data'] = []
    content = requests.get(statement).content
    report_soup = BeautifulSoup(content,"html")
    for index, row in enumerate(report_soup.table.find_all('tr')):
        cols = row.find_all('td')
        if (len(row.find_all('th'))) == 0 and (len(row.find_all('strong')) ==␣
 ↪0):
            reg_row = [ele.text.strip() for ele in cols]
            statement_data['data'].append(reg_row)
```

```
        elif (len(row.find_all('th')) == 0 and len(row.find_all('strong')) !=␣
↪0):
            sec_row = cols[0].text.strip()
            statement_data['section'].append(sec_row)
        elif (len(row.find_all('th'))!= 0):
            hed_row = [ele.text.strip() for ele in row.find_all('th')]
            statement_data['header'].append(hed_row)
    statements_data.append(statement_data)

pd.DataFrame(statements_data)
```

[19]:

```
                                                header  \
0  [[Consolidated Statements of Cash Flows - USD …
1  [[Consolidated Statements of Comprehensive Inc…
2  [[Consolidated Balance Sheets - USD ($) $ in M…

                                               section  \
0  [Statement of Cash Flows [Abstract], OPERATING…
1  [Statement of Comprehensive Income [Abstract],…
2  [Current assets:, Current liabilities:, Stockh…

                                                  data
0  [[CASH, CASH EQUIVALENTS, AND RESTRICTED CASH,…
1  [[Net income, $ 6,331, $ 2,134, $ 14,109, $ 8,…
2  [[Cash and cash equivalents, $ 29,930, $ 36,09…
```

[20]:

```
income_header = statements_data[1]['header'][1]
income_data = statements_data[1]['data']

income_df = pd.DataFrame(income_data)

income_df.index = income_df[0]
income_df.index.name = 'Category'
income_df = income_df.drop(0,axis=1)
income_df = income_df.replace('[\$,)]',"", regex=True)
income_df = income_df.replace('[(]',"-", regex = True)
income_df = income_df.replace('', 'NaN', regex = True)
income_df = income_df.astype(float)
income_df.columns = income_header

income_df
```

[20]:

```
                                                Sep. 30, 2020  \
Category
Net income                                                6331.0
Foreign currency translation adjustments, net o…           408.0
Reclassification adjustment for foreign currenc…             0.0
```

```
Net foreign currency translation adjustments                408.0
Unrealized gains (losses), net of tax of $(2), …            35.0
Reclassification adjustment for losses (gains) …           -17.0
Net unrealized gains (losses) on available-for-…            18.0
Other comprehensive income (loss)                           426.0
Comprehensive income                                       6757.0

                                                      Sep. 30, 2019  \
Category
Net income                                                 2134.0
Foreign currency translation adjustments, net o…          -368.0
Reclassification adjustment for foreign currenc…          -108.0
Net foreign currency translation adjustments               -476.0
Unrealized gains (losses), net of tax of $(2), …            9.0
Reclassification adjustment for losses (gains) …           -2.0
Net unrealized gains (losses) on available-for-…            7.0
Other comprehensive income (loss)                          -469.0
Comprehensive income                                       1665.0

                                                      Sep. 30, 2020  \
Category
Net income                                                14109.0
Foreign currency translation adjustments, net o…          -260.0
Reclassification adjustment for foreign currenc…             0.0
Net foreign currency translation adjustments              -260.0
Unrealized gains (losses), net of tax of $(2), …           239.0
Reclassification adjustment for losses (gains) …           -22.0
Net unrealized gains (losses) on available-for-…           217.0
Other comprehensive income (loss)                           -43.0
Comprehensive income                                       14066.0

                                                      Sep. 30, 2019
Category
Net income                                                 8320.0
Foreign currency translation adjustments, net o…          -369.0
Reclassification adjustment for foreign currenc…          -108.0
Net foreign currency translation adjustments               -477.0
Unrealized gains (losses), net of tax of $(2), …            85.0
Reclassification adjustment for losses (gains) …           -2.0
Net unrealized gains (losses) on available-for-…            83.0
Other comprehensive income (loss)                          -394.0
Comprehensive income                                       7926.0
```

[21]:
```
income_header = statements_data[0]['header']
income_data = statements_data[0]['data']

income_df = pd.DataFrame(income_data)
```

```
income_df.index = income_df[0]
income_df.index.name = 'Category'
income_df = income_df.drop(0,axis=1)
income_df = income_df.replace('[\$,)]',"", regex=True)
income_df = income_df.replace('[(]',"-", regex = True)
income_df = income_df.replace('', 'NaN', regex = True)
income_df = income_df.astype(float)
income_df.columns = income_header[1]

income_df
```

[21]:

Category	Sep. 30, 2020
CASH, CASH EQUIVALENTS, AND RESTRICTED CASH, BE…	37842.0
Net income	6331.0
Depreciation and amortization of property and e…	6523.0
Stock-based compensation	2288.0
Other operating expense (income), net	67.0
Other expense (income), net	-1051.0
Deferred income taxes	295.0
Inventories	-3899.0
Accounts receivable, net and other	-2016.0
Accounts payable	3658.0
Accrued expenses and other	-310.0
Unearned revenue	78.0
Net cash provided by (used in) operating activi…	11964.0
Purchases of property and equipment	-11063.0
Proceeds from property and equipment sales and …	1255.0
Acquisitions, net of cash acquired, and other	-1735.0
Sales and maturities of marketable securities	13135.0
Purchases of marketable securities	-17468.0
Net cash provided by (used in) investing activi…	-15876.0
Proceeds from short-term debt, and other	1311.0
Repayments of short-term debt, and other	-1349.0
Proceeds from long-term debt	0.0
Repayments of long-term debt	-1198.0
Principal repayments of finance leases	-2857.0
Principal repayments of financing obligations	-12.0
Net cash provided by (used in) financing activi…	-4105.0
Foreign currency effect on cash, cash equivalen…	377.0
Net increase (decrease) in cash, cash equivalen…	-7640.0
CASH, CASH EQUIVALENTS, AND RESTRICTED CASH, EN…	30202.0

Category	Sep. 30, 2019
CASH, CASH EQUIVALENTS, AND RESTRICTED CASH, BE…	22965.0

Net income	2134.0
Depreciation and amortization of property and e…	5563.0
Stock-based compensation	1779.0
Other operating expense (income), net	47.0
Other expense (income), net	388.0
Deferred income taxes	92.0
Inventories	-381.0
Accounts receivable, net and other	-1181.0
Accounts payable	226.0
Accrued expenses and other	-722.0
Unearned revenue	-53.0
Net cash provided by (used in) operating activi…	7892.0
Purchases of property and equipment	-4697.0
Proceeds from property and equipment sales and …	1312.0
Acquisitions, net of cash acquired, and other	-398.0
Sales and maturities of marketable securities	7251.0
Purchases of marketable securities	-8542.0
Net cash provided by (used in) investing activi…	-5074.0
Proceeds from short-term debt, and other	415.0
Repayments of short-term debt, and other	-341.0
Proceeds from long-term debt	287.0
Repayments of long-term debt	-14.0
Principal repayments of finance leases	-2307.0
Principal repayments of financing obligations	0.0
Net cash provided by (used in) financing activi…	-1960.0
Foreign currency effect on cash, cash equivalen…	-269.0
Net increase (decrease) in cash, cash equivalen…	589.0
CASH, CASH EQUIVALENTS, AND RESTRICTED CASH, EN…	23554.0

Category	Sep. 30, 2020 \
CASH, CASH EQUIVALENTS, AND RESTRICTED CASH, BE…	36410.0
Net income	14109.0
Depreciation and amortization of property and e…	17633.0
Stock-based compensation	6646.0
Other operating expense (income), net	416.0
Other expense (income), net	-1255.0
Deferred income taxes	1082.0
Inventories	-3178.0
Accounts receivable, net and other	-3608.0
Accounts payable	4231.0
Accrued expenses and other	-1375.0
Unearned revenue	932.0
Net cash provided by (used in) operating activi…	35633.0
Purchases of property and equipment	-25317.0
Proceeds from property and equipment sales and …	3467.0
Acquisitions, net of cash acquired, and other	-1945.0

Sales and maturities of marketable securities	32899.0
Purchases of marketable securities	-51678.0
Net cash provided by (used in) investing activi…	-42574.0
Proceeds from short-term debt, and other	4361.0
Repayments of short-term debt, and other	-3886.0
Proceeds from long-term debt	9994.0
Repayments of long-term debt	-1439.0
Principal repayments of finance leases	-8274.0
Principal repayments of financing obligations	-44.0
Net cash provided by (used in) financing activi…	712.0
Foreign currency effect on cash, cash equivalen…	21.0
Net increase (decrease) in cash, cash equivalen…	-6208.0
CASH, CASH EQUIVALENTS, AND RESTRICTED CASH, EN…	30202.0

Category	Sep. 30, 2019 \
CASH, CASH EQUIVALENTS, AND RESTRICTED CASH, BE…	32173.0
Net income	8320.0
Depreciation and amortization of property and e…	15619.0
Stock-based compensation	5024.0
Other operating expense (income), net	114.0
Other expense (income), net	246.0
Deferred income taxes	612.0
Inventories	-1762.0
Accounts receivable, net and other	-3776.0
Accounts payable	-2490.0
Accrued expenses and other	-4277.0
Unearned revenue	1225.0
Net cash provided by (used in) operating activi…	18855.0
Purchases of property and equipment	-11549.0
Proceeds from property and equipment sales and …	2800.0
Acquisitions, net of cash acquired, and other	-1684.0
Sales and maturities of marketable securities	15056.0
Purchases of marketable securities	-25368.0
Net cash provided by (used in) investing activi…	-20745.0
Proceeds from short-term debt, and other	722.0
Repayments of short-term debt, and other	-704.0
Proceeds from long-term debt	453.0
Repayments of long-term debt	-115.0
Principal repayments of finance leases	-6848.0
Principal repayments of financing obligations	-3.0
Net cash provided by (used in) financing activi…	-6495.0
Foreign currency effect on cash, cash equivalen…	-234.0
Net increase (decrease) in cash, cash equivalen…	-8619.0
CASH, CASH EQUIVALENTS, AND RESTRICTED CASH, EN…	23554.0

	Sep. 30, 2020 \

Category	
CASH, CASH EQUIVALENTS, AND RESTRICTED CASH, BE…	23554.0
Net income	17377.0
Depreciation and amortization of property and e…	23803.0
Stock-based compensation	8486.0
Other operating expense (income), net	466.0
Other expense (income), net	-1749.0
Deferred income taxes	1267.0
Inventories	-4694.0
Accounts receivable, net and other	-7515.0
Accounts payable	14914.0
Accrued expenses and other	1520.0
Unearned revenue	1417.0
Net cash provided by (used in) operating activi…	55292.0
Purchases of property and equipment	-30629.0
Proceeds from property and equipment sales and …	4838.0
Acquisitions, net of cash acquired, and other	-2722.0
Sales and maturities of marketable securities	40525.0
Purchases of marketable securities	-58122.0
Net cash provided by (used in) investing activi…	-46110.0
Proceeds from short-term debt, and other	5042.0
Repayments of short-term debt, and other	-4701.0
Proceeds from long-term debt	10412.0
Repayments of long-term debt	-2490.0
Principal repayments of finance leases	-11054.0
Principal repayments of financing obligations	-68.0
Net cash provided by (used in) financing activi…	-2859.0
Foreign currency effect on cash, cash equivalen…	325.0
Net increase (decrease) in cash, cash equivalen…	6648.0
CASH, CASH EQUIVALENTS, AND RESTRICTED CASH, EN…	30202.0

	Sep. 30, 2019
Category	
CASH, CASH EQUIVALENTS, AND RESTRICTED CASH, BE…	21032.0
Net income	11347.0
Depreciation and amortization of property and e…	19881.0
Stock-based compensation	6441.0
Other operating expense (income), net	186.0
Other expense (income), net	443.0
Deferred income taxes	784.0
Inventories	-3112.0
Accounts receivable, net and other	-5172.0
Accounts payable	4393.0
Accrued expenses and other	-1612.0
Unearned revenue	1753.0
Net cash provided by (used in) operating activi…	35332.0
Purchases of property and equipment	-15282.0

```
Proceeds from property and equipment sales and …         3414.0
Acquisitions, net of cash acquired, and other            -2015.0
Sales and maturities of marketable securities            16994.0
Purchases of marketable securities                       -27428.0
Net cash provided by (used in) investing activi…        -24317.0
Proceeds from short-term debt, and other                  1292.0
Repayments of short-term debt, and other                 -1129.0
Proceeds from long-term debt                              589.0
Repayments of long-term debt                             -124.0
Principal repayments of finance leases                   -8754.0
Principal repayments of financing obligations            -129.0
Net cash provided by (used in) financing activi…         -8255.0
Foreign currency effect on cash, cash equivalen…         -238.0
Net increase (decrease) in cash, cash equivalen…         2522.0
CASH, CASH EQUIVALENTS, AND RESTRICTED CASH, EN…         23554.0
```

[22]:
```
income_header = statements_data[2]['header'][0][1::]
income_data = statements_data[2]['data']

income_df = pd.DataFrame(income_data)

income_df.index = income_df[0]
income_df.index.name = 'Category'
income_df = income_df.drop(0,axis=1)
income_df = income_df.replace('[\$,)]',"", regex=True)
income_df = income_df.replace('[(]',"-", regex = True)
income_df = income_df.replace('', 'NaN', regex = True)
income_df = income_df.astype(float)
income_df.columns = income_header
income_df
```

[22]:
```
                                                Sep. 30, 2020  \
Category
Cash and cash equivalents                                29930.0
Marketable securities                                    38472.0
Inventories                                              23735.0
Accounts receivable, net and other                       20832.0
Total current assets                                    112969.0
Property and equipment, net                              99981.0
Operating leases                                         34119.0
Goodwill                                                 14960.0
Other assets                                             20150.0
Total assets                                            282179.0
Accounts payable                                         58334.0
Accrued expenses and other                               34327.0
Unearned revenue                                          9251.0
Total current liabilities                               101912.0
```

Long-term lease liabilities	48589.0
Long-term debt	32929.0
Other long-term liabilities	15974.0
Commitments and contingencies (Note 4)	NaN
Preferred stock, $0.01 par value: Authorized sh…	0.0
Common stock, $0.01 par value: Authorized share…	5.0
Treasury stock, at cost	-1837.0
Additional paid-in capital	40307.0
Accumulated other comprehensive income (loss)	-1029.0
Retained earnings	45329.0
Total stockholders' equity	82775.0
Total liabilities and stockholders' equity	282179.0

	Dec. 31, 2019
Category	
Cash and cash equivalents	36092.0
Marketable securities	18929.0
Inventories	20497.0
Accounts receivable, net and other	20816.0
Total current assets	96334.0
Property and equipment, net	72705.0
Operating leases	25141.0
Goodwill	14754.0
Other assets	16314.0
Total assets	225248.0
Accounts payable	47183.0
Accrued expenses and other	32439.0
Unearned revenue	8190.0
Total current liabilities	87812.0
Long-term lease liabilities	39791.0
Long-term debt	23414.0
Other long-term liabilities	12171.0
Commitments and contingencies (Note 4)	NaN
Preferred stock, $0.01 par value: Authorized sh…	0.0
Common stock, $0.01 par value: Authorized share…	5.0
Treasury stock, at cost	-1837.0
Additional paid-in capital	33658.0
Accumulated other comprehensive income (loss)	-986.0
Retained earnings	31220.0
Total stockholders' equity	62060.0
Total liabilities and stockholders' equity	225248.0

In the end, I was able to extract all three tables using the `xbrl` formatting provided by the SEC. Unfortunately, each company has a different structure to their data, so extracting multiple data for multiple companies is very difficult without manual intervention.

I decided to go with an API for this portion to save me the headache. I'll be using the SimFim

API, which will allow me to pull basic financial records for each company.

[23]:
```
with open("apikey.txt","r") as f:
    api_key = f.readline()

periods = ["q1", "q2", "q3", "q4"]
year_start = 2012
year_end = 2020

# request url for all financial statements
request_url = 'https://simfin.com/api/v2/companies/statements'

# variable to store the names of the columns
columns = []
# variable to store our data
output = []

for year in range(year_start, year_end + 1):
    # loop through periods
    for period in periods:
        # define the parameters for the query
        parameters = {"statement": "all", "ticker": ticker, "period": period,␣
↪"fyear": year, "api-key": api_key}
        # make the request
        request = requests.get(request_url, parameters)

        # convert response to json and take 0th index as we only requested one␣
↪ticker (if more than one ticker is requested, the data for the nth ticker␣
↪will be at the nth position in the result returned from the API)
        data = request.json()[0]

        # make sure that data was found
        if data['found'] and len(data['data']) > 0:
            # add the column descriptions once only
            if len(columns) == 0:
                columns = data['columns']
            # add the data
            output += data['data']

df = pd.DataFrame(output, columns=columns)
df
```

[23]:

	SimFinId	Ticker	Fiscal Period	Fiscal Year	Report Date	Publish Date	\
0	62747	AMZN	Q1	2012	2012-03-31	2012-04-26	
1	62747	AMZN	Q2	2012	2012-06-30	2012-07-26	
2	62747	AMZN	Q3	2012	2012-09-30	2012-10-25	
3	62747	AMZN	Q4	2012	2012-12-31	2013-01-26	

4	62747	AMZN	Q1	2013	2013-03-31	2013-04-25
5	62747	AMZN	Q2	2013	2013-06-30	2013-07-25
6	62747	AMZN	Q3	2013	2013-09-30	2013-10-24
7	62747	AMZN	Q4	2013	2013-12-31	2014-01-25
8	62747	AMZN	Q1	2014	2014-03-31	2014-04-24
9	62747	AMZN	Q2	2014	2014-06-30	2014-07-24
10	62747	AMZN	Q3	2014	2014-09-30	2014-10-23
11	62747	AMZN	Q4	2014	2014-12-31	2015-01-24
12	62747	AMZN	Q1	2015	2015-03-31	2015-04-24
13	62747	AMZN	Q2	2015	2015-06-30	2015-07-24
14	62747	AMZN	Q3	2015	2015-09-30	2015-10-23
15	62747	AMZN	Q4	2015	2015-12-31	2016-01-29
16	62747	AMZN	Q1	2016	2016-03-31	2016-04-28
17	62747	AMZN	Q2	2016	2016-06-30	2016-07-28
18	62747	AMZN	Q3	2016	2016-09-30	2016-10-27
19	62747	AMZN	Q4	2016	2016-12-31	2017-01-28
20	62747	AMZN	Q1	2017	2017-03-31	2017-04-27
21	62747	AMZN	Q2	2017	2017-06-30	2017-07-27
22	62747	AMZN	Q3	2017	2017-09-30	2017-10-26
23	62747	AMZN	Q4	2017	2017-12-31	2018-01-27
24	62747	AMZN	Q1	2018	2018-03-31	2018-04-26
25	62747	AMZN	Q2	2018	2018-06-30	2018-07-26
26	62747	AMZN	Q3	2018	2018-09-30	2018-10-25
27	62747	AMZN	Q4	2018	2018-12-31	2019-01-26
28	62747	AMZN	Q1	2019	2019-03-31	2019-04-26
29	62747	AMZN	Q2	2019	2019-06-30	2019-07-26
30	62747	AMZN	Q3	2019	2019-09-30	2019-10-25
31	62747	AMZN	Q4	2019	2019-12-31	2020-01-31
32	62747	AMZN	Q1	2020	2020-03-31	2020-05-01
33	62747	AMZN	Q2	2020	2020-06-30	2020-07-31
34	62747	AMZN	Q3	2020	2020-09-30	2020-10-30

	Restated Date	Source	TTM	\
0	2013-04-26	https://www.sec.gov/Archives/edgar/data/101872…	False	
1	2013-07-26	https://www.sec.gov/Archives/edgar/data/101872…	False	
2	2013-10-25	https://www.sec.gov/Archives/edgar/data/101872…	False	
3	2015-01-30	https://www.sec.gov/Archives/edgar/data/101872…	False	
4	2014-04-25	https://www.sec.gov/Archives/edgar/data/101872…	False	
5	2014-07-25	https://www.sec.gov/Archives/edgar/data/101872…	False	
6	2014-10-24	https://www.sec.gov/Archives/edgar/data/101872…	False	
7	2016-01-29	https://www.sec.gov/Archives/edgar/data/101872…	False	
8	2015-04-24	https://www.sec.gov/Archives/edgar/data/101872…	False	
9	2015-07-24	https://www.sec.gov/Archives/edgar/data/101872…	False	
10	2015-10-23	https://www.sec.gov/Archives/edgar/data/101872…	False	
11	2017-02-10	https://www.sec.gov/Archives/edgar/data/101872…	False	
12	2016-04-29	https://www.sec.gov/Archives/edgar/data/101872…	False	
13	2016-07-29	https://www.sec.gov/Archives/edgar/data/101872…	False	

14	2016-10-28	https://www.sec.gov/Archives/edgar/data/101872…	False
15	2018-02-02	https://www.sec.gov/Archives/edgar/data/101872…	False
16	2017-04-28	https://www.sec.gov/Archives/edgar/data/101872…	False
17	2017-07-28	https://www.sec.gov/Archives/edgar/data/101872…	False
18	2017-10-27	https://www.sec.gov/Archives/edgar/data/101872…	False
19	2019-02-01	https://www.sec.gov/Archives/edgar/data/101872…	False
20	2018-04-27	https://www.sec.gov/Archives/edgar/data/101872…	False
21	2018-07-27	https://www.sec.gov/Archives/edgar/data/101872…	False
22	2018-10-26	https://www.sec.gov/Archives/edgar/data/101872…	False
23	2020-01-31	https://www.sec.gov/Archives/edgar/data/101872…	False
24	2019-04-26	https://www.sec.gov/Archives/edgar/data/101872…	False
25	2019-07-26	https://www.sec.gov/Archives/edgar/data/101872…	False
26	2019-10-25	https://www.sec.gov/Archives/edgar/data/101872…	False
27	2020-01-31	https://www.sec.gov/Archives/edgar/data/101872…	False
28	2020-05-01	https://www.sec.gov/Archives/edgar/data/101872…	False
29	2020-07-31	https://www.sec.gov/Archives/edgar/data/101872…	False
30	2019-10-25	https://www.sec.gov/Archives/edgar/data/101872…	False
31	2020-01-31	https://www.sec.gov/Archives/edgar/data/101872…	False
32	2020-05-01	https://www.sec.gov/Archives/edgar/data/101872…	False
33	2020-07-31	https://www.sec.gov/Archives/edgar/data/101872…	False
34	2020-10-30	https://www.sec.gov/Archives/edgar/data/101872…	False

	Value Check	…	Current Ratio	Liabilities to Equity Ratio	Debt Ratio	\
0	True	…	1.16142	1.79728	0.00000	
1	True	…	1.07798	1.80107	0.00000	
2	True	…	1.04371	2.02317	0.00000	
3	True	…	1.12072	2.97400	0.09473	
4	True	…	1.14680	2.36539	0.10713	
5	True	…	1.11388	2.39208	0.10269	
6	True	…	1.08224	2.50622	0.09551	
7	True	…	1.07158	3.12056	0.07946	
8	True	…	1.05284	2.52091	0.08654	
9	True	…	1.00021	2.57461	0.08230	
10	True	…	0.89109	2.91051	0.07667	
11	True	…	1.11528	4.07448	0.15164	
12	True	…	1.11844	3.60544	0.16489	
13	True	…	1.10367	3.45615	0.15732	
14	True	…	1.08223	3.52373	0.14659	
15	True	…	1.05365	3.83764	0.12706	
16	True	…	1.08252	3.14259	0.13446	
17	True	…	1.08899	2.93494	0.12619	
18	True	…	1.06302	2.98701	0.11573	
19	True	…	1.04485	3.32471	0.09225	
20	True	…	1.05511	2.73577	0.09499	
21	True	…	1.01202	2.78138	0.08752	
22	True	…	1.03199	3.67463	0.21437	
23	True	…	1.03998	3.73889	0.18843	

24	True	…	1.05795	3.01621	0.19500
25	True	…	1.07244	2.83198	0.18373
26	True	…	1.08244	2.67272	0.17178
27	True	…	1.09811	2.73483	0.14445
28	True	…	1.09005	2.67903	0.13095
29	True	…	1.10207	2.60625	0.12192
30	True	…	1.09590	2.52338	0.11287
31	True	…	1.09705	2.62952	0.10395
32	True	…	1.07871	2.38948	0.10594
33	True	…	1.18118	2.50361	0.12825
34	True	…	1.10850	2.40899	0.11670

	Earnings Per Share, Basic	Earnings Per Share, Diluted	Sales Per Share	\
0	0.28698	0.28261	29.10596	
1	0.01552	0.01528	28.45676	
2	-0.60619	-0.60619	30.54425	
3	0.21491	0.22172	46.64035	
4	0.18022	0.17711	35.31868	
5	-0.01535	-0.01535	34.43860	
6	-0.08972	-0.08972	37.40044	
7	0.52174	0.49587	55.62174	
8	0.23478	0.23077	42.91522	
9	-0.27332	-0.27332	41.95228	
10	-0.94384	-0.94384	44.44708	
11	0.46121	0.46930	63.20690	
12	-0.12258	-0.12258	48.85376	
13	0.19700	0.19328	49.64668	
14	0.16880	0.16527	54.18376	
15	1.02991	0.98569	76.38034	
16	1.08917	1.06653	61.84289	
17	1.81184	1.77433	64.27907	
18	0.53165	0.51959	69.01688	
19	1.56695	1.53799	91.50837	
20	1.51782	1.47755	74.87212	
21	0.41127	0.40041	79.23800	
22	0.53222	0.51822	90.94387	
23	3.84265	3.74194	125.16149	
24	3.36570	3.27108	105.45868	
25	5.21399	5.06800	108.81893	
26	5.90779	5.75449	115.93443	
27	6.17755	6.04192	147.72041	
28	7.25255	7.09363	121.58859	
29	5.32454	5.21869	128.60852	
30	4.31111	4.23413	141.37576	
31	6.57545	6.44576	175.92958	
32	5.09036	5.00988	151.51004	
33	10.48600	10.30059	177.82400	

```
34                 12.63673                    12.36523       191.90619

   Equity Per Share Free Cash Flow Per Share  Dividends Per Share  \
0          16.05077                 -5.71965                 None
1          16.64080                  3.15078                 None
2          16.71018                  1.37389                 None
3          17.96491                  6.73465                 None
4          18.53187                 -6.04176                 None
5          19.15132                  2.99781                 None
6          19.88403                  1.30197                 None
7          21.18696                  9.91522                 None
8          22.45217                 -5.95435                 None
9          22.99783                  6.39913                 None
10         22.32397                  1.53348                 None
11         23.14871                  8.49784                 None
12         23.38280                 -2.88817                 None
13         25.19914                 16.84797                 None
14         26.55983                  5.97863                 None
15         28.59829                 -5.74573                 None
16         31.32909                 -1.14013                 None
17         34.96406                 26.04228                 None
18         37.51477                 28.60549                 None
19         40.34519                -32.57322                 None
20         45.43816                 -8.44654                 None
21         48.46347                  7.52192                 None
22         51.26403                  1.85863                 None
23         57.36853                  7.63561                 None
24         65.00620                 -1.96901                 None
25         72.00617                  2.00206                 None
26         80.17418                  6.57172                 None
27         88.87551                 22.98367                 None
28         98.59470                 14.85947                 None
29        107.62880                 12.19878                 None
30        114.15758                  4.86061                 None
31        124.86922                  7.42254                 None
32        131.06827                 -8.52209                 None
33        147.45600                 22.85800                 None
34        165.21956                 -0.53094                 None

   Pietroski F-Score
0                  3
1                  5
2                  4
3                  5
4                  2
5                  5
6                  5
```

```
7                   6
8                   4
9                   4
10                  4
11                  6
12                  3
13                  6
14                  6
15                  6
16                  4
17                  7
18                  6
19                  5
20                  4
21                  5
22                  4
23                  3
24                  6
25                  6
26                  7
27                  5
28                  7
29                  6
30                  5
31                  3
32                  4
33                  6
34                  5

[35 rows x 224 columns]
```

[24]:
```
list(df.columns)
```

[24]:
```
['SimFinId',
 'Ticker',
 'Fiscal Period',
 'Fiscal Year',
 'Report Date',
 'Publish Date',
 'Restated Date',
 'Source',
 'TTM',
 'Value Check',
 'Revenue',
 'Sales & Services Revenue',
 'Financing Revenue',
 'Other Revenue',
```

```
'Cost of Revenue',
'Cost of Goods & Services',
'Cost of Financing Revenue',
'Cost of Other Revenue',
'Gross Profit',
'Other Operating Income',
'Operating Expenses',
'Selling, General & Administrative',
'Selling & Marketing',
'General & Administrative',
'Research & Development',
'Depreciation & Amortization',
'Provision for Doubtful Accounts',
'Other Operating Expenses',
'Operating Income (Loss)',
'Non-Operating Income (Loss)',
'Interest Expense, Net',
'Interest Expense',
'Interest Income',
'Other Investment Income (Loss)',
'Foreign Exchange Gain (Loss)',
'Income (Loss) from Affiliates',
'Other Non-Operating Income (Loss)',
'Pretax Income (Loss), Adj.',
'Abnormal Gains (Losses)',
'Acquired In-Process R&D',
'Merger & Acquisition Expense',
'Abnormal Derivatives',
'Disposal of Assets',
'Early Extinguishment of Debt',
'Asset Write-Down',
'Impairment of Goodwill & Intangibles',
'Sale of Business',
'Legal Settlement',
'Restructuring Charges',
'Sale of Investments & Unrealized Investments',
'Insurance Settlement',
'Other Abnormal Items',
'Pretax Income (Loss)',
'Income Tax (Expense) Benefit, Net',
'Current Income Tax',
'Deferred Income Tax',
'Tax Allowance/Credit',
'Income (Loss) from Affiliates, Net of Taxes',
'Income (Loss) from Continuing Operations',
'Net Extraordinary Gains (Losses)',
'Discontinued Operations',
```

```
'Accounting Charges & Other',
'Income (Loss) Incl. Minority Interest',
'Minority Interest',
'Net Income',
'Preferred Dividends',
'Other Adjustments',
'Net Income (Common)',
'Cash, Cash Equivalents & Short Term Investments',
'Cash & Cash Equivalents',
'Short Term Investments',
'Accounts & Notes Receivable',
'Accounts Receivable, Net',
'Notes Receivable, Net',
'Unbilled Revenues',
'Inventories',
'Raw Materials',
'Work In Process',
'Finished Goods',
'Other Inventory',
'Other Short Term Assets',
'Prepaid Expenses',
'Derivative & Hedging Assets (Short Term)',
'Assets Held-for-Sale',
'Deferred Tax Assets (Short Term)',
'Income Taxes Receivable',
'Discontinued Operations (Short Term)',
'Misc. Short Term Assets',
'Total Current Assets',
'Property, Plant & Equipment, Net',
'Property, Plant & Equipment',
'Accumulated Depreciation',
'Long Term Investments & Receivables',
'Long Term Investments',
'Long Term Marketable Securities',
'Long Term Receivables',
'Other Long Term Assets',
'Intangible Assets',
'Goodwill',
'Other Intangible Assets',
'Prepaid Expense',
'Deferred Tax Assets (Long Term)',
'Derivative & Hedging Assets (Long Term)',
'Prepaid Pension Costs',
'Discontinued Operations (Long Term)',
'Investments in Affiliates',
'Misc. Long Term Assets',
'Total Noncurrent Assets',
```

```
'Total Assets',
'Payables & Accruals',
'Accounts Payable',
'Accrued Taxes',
'Interest & Dividends Payable',
'Other Payables & Accruals',
'Short Term Debt',
'Short Term Borrowings',
'Short Term Capital Leases',
'Current Portion of Long Term Debt',
'Other Short Term Liabilities',
'Deferred Revenue (Short Term)',
'Liabilities from Derivatives & Hedging (Short Term)',
'Deferred Tax Liabilities (Short Term)',
'Liabilities from Discontinued Operations (Short Term)',
'Misc. Short Term Liabilities',
'Total Current Liabilities',
'Long Term Debt',
'Long Term Borrowings',
'Long Term Capital Leases',
'Other Long Term Liabilities',
'Accrued Liabilities',
'Pension Liabilities',
'Pensions',
'Other Post-Retirement Benefits',
'Deferred Compensation',
'Deferred Revenue (Long Term)',
'Deferred Tax Liabilities (Long Term)',
'Liabilities from Derivatives & Hedging (Long Term)',
'Liabilities from Discontinued Operations (Long Term)',
'Misc. Long Term Liabilities',
'Total Noncurrent Liabilities',
'Total Liabilities',
'Preferred Equity',
'Share Capital & Additional Paid-In Capital',
'Common Stock',
'Additional Paid in Capital',
'Other Share Capital',
'Treasury Stock',
'Retained Earnings',
'Other Equity',
'Equity Before Minority Interest',
'Minority Interest',
'Total Equity',
'Total Liabilities & Equity',
'Net Income/Starting Line',
'Net Income',
```

```
'Net Income from Discontinued Operations',
'Other Adjustments',
'Depreciation & Amortization',
'Non-Cash Items',
'Stock-Based Compensation',
'Deferred Income Taxes',
'Other Non-Cash Adjustments',
'Change in Working Capital',
'Change in Accounts Receivable',
'Change in Inventories',
'Change in Accounts Payable',
'Change in Other',
'Net Cash from Discontinued Operations (Operating)',
'Net Cash from Operating Activities',
'Change in Fixed Assets & Intangibles',
'Disposition of Fixed Assets & Intangibles',
'Disposition of Fixed Assets',
'Disposition of Intangible Assets',
'Acquisition of Fixed Assets & Intangibles',
'Purchase of Fixed Assets',
'Acquisition of Intangible Assets',
'Other Change in Fixed Assets & Intangibles',
'Net Change in Long Term Investment',
'Decrease in Long Term Investment',
'Increase in Long Term Investment',
'Net Cash from Acquisitions & Divestitures',
'Net Cash from Divestitures',
'Cash for Acquisition of Subsidiaries',
'Cash for Joint Ventures',
'Net Cash from Other Acquisitions',
'Other Investing Activities',
'Net Cash from Discontinued Operations (Investing)',
'Net Cash from Investing Activities',
'Dividends Paid',
'Cash from (Repayment of) Debt',
'Cash from (Repayment of) Short Term Debt, Net',
'Cash from (Repayment of) Long Term Debt, Net',
'Repayments of Long Term Debt',
'Cash from Long Term Debt',
'Cash from (Repurchase of) Equity',
'Increase in Capital Stock',
'Decrease in Capital Stock',
'Other Financing Activities',
'Net Cash from Discontinued Operations (Financing)',
'Net Cash from Financing Activities',
'Net Cash Before Disc. Operations and FX',
'Change in Cash from Disc. Operations and Other',
```

```
'Net Cash Before FX',
'Effect of Foreign Exchange Rates',
'Net Change in Cash',
'EBITDA',
'Total Debt',
'Free Cash Flow',
'Gross Profit Margin',
'Operating Margin',
'Net Profit Margin',
'Return on Equity',
'Return on Assets',
'Free Cash Flow to Net Income',
'Current Ratio',
'Liabilities to Equity Ratio',
'Debt Ratio',
'Earnings Per Share, Basic',
'Earnings Per Share, Diluted',
'Sales Per Share',
'Equity Per Share',
'Free Cash Flow Per Share',
'Dividends Per Share',
'Pietroski F-Score']
```