U.S. Stock Market I: Matching SEC Data to Trading Days and Ticker Symbols

In [1]:
```
import pandas as pd
import numpy as np
import requests, zipfile, io
import os
from pathlib import Path

from tiingo import TiingoClient
tiingo = TiingoClient({'api_key':'XXX'})
```

In [2]:
```
def get_items_from_SEC_files(tags, filename=None):                  # Function inp

    directory = 'data/sec/merged/'                                  # Read data fr
    filenames = [filename] if filename else os.listdir(directory) # Supplied fil
    filenames = [f for f in filenames if not f.startswith(".")]   # Exclude hidd

    results   = {t:pd.DataFrame() for t in tags}                    # Dictionary o

    for filename in filenames:                                      # Loop over al
        print(filename)
        data = pd.read_csv(directory+filename, parse_dates=['filed','ddate'])  #

        for t in tags:                                              # Loop over al
            item  = data[data.tag==t]                               # Select all d
            short = item.sort_values(['cik','filed','ddate','qtrs'], ascending=[
            long  = item.sort_values(['cik','filed','ddate','qtrs'], ascending=[
            short = short.groupby(['cik','filed']).last()[['value','qtrs']]
            long  = long .groupby(['cik','filed']).last()[['value','qtrs']]
            short_long = short.join(long, lsuffix='_shortest', rsuffix='_longest
            results[t] = results[t].append( short_long )

    for t in tags:                                                  # Now sort all
        if not results[t].empty: results[t] = results[t].sort_index(level='filed

    return results



def calculate_quarterly_annual_values(item):                          # item: tabl
    result           = pd.DataFrame()                                 # Results go
    all_firms        = item.index.get_level_values('cik').unique()  # All CIKs.
    all_filing_dates = pd.read_csv('data/sec/dates/filing_dates.csv', index_col=

    for cik in all_firms:                                             # Loop over
        filing_dates = pd.Series(all_filing_dates.filed[cik])       # All filing

        # Quarterly values:
        valuesQ = item.loc[cik].value_shortest.reindex(filing_dates) # Values wi
        qtrsQ   = item.loc[cik].qtrs_shortest.astype(int)           # Number of
        for date,q in qtrsQ[qtrsQ>1].iteritems():                   # Loop over
            previous_values = valuesQ[:date][-q:-1]                 # Example: f
            if len(previous_values) == q-1:                         # If all pre
                valuesQ[date] -= previous_values.sum(skipna=False)  # Subtract p
```

```
            else:
                valuesQ[date]  = np.nan

        # Annual values:
        valuesA = item.loc[cik].value_longest.reindex(filing_dates) # Values wit
        qtrsA   = item.loc[cik].qtrs_longest.astype(int)            # Number of
        for date,q in qtrsA[qtrsA<4].iteritems():                   # Loop over
            previous_values = valuesQ[:date][-4:-q]                 # Example: f
            if len(previous_values) == 4-q:                         # If all pre
                valuesA[date] += previous_values.sum(skipna=False)  # Add previo
            else:
                valuesA[date]  = np.nan

        result = result.append( pd.DataFrame({'cik':cik, 'filed':filing_dates, '

    return result.set_index(['cik','filed'])                        # Return a t
```

Get R&D values:

In [3]:

```
tags  = ['ResearchAndDevelopmentExpense']
items = get_items_from_SEC_files(tags)
item  = items[tags[0]]

rnd = calculate_quarterly_annual_values(item)
rnd
```

```
2018q4.csv
2018q3.csv
2018q2.csv
2021_01.csv
2018q1.csv
2020q2.csv
2020q3.csv
2020q1.csv
2019q4.csv
2019q1.csv
2019q3.csv
2019q2.csv
2013q4.csv
2015q2.csv
2015q3.csv
2017q1.csv
2020_12.csv
2020_10.csv
2017q3.csv
2015q1.csv
2017q2.csv
2011q4.csv
2020_11.csv
2013q2.csv
2015q4.csv
2011q1.csv
2013q3.csv
2013q1.csv
2011q3.csv
2017q4.csv
2011q2.csv
2009q4.csv
2014q1.csv
2016q3.csv
2010q4.csv
2016q2.csv
```

```
2014q2.csv
2012q4.csv
2016q1.csv
2014q3.csv
2009q2.csv
2010q3.csv
2012q1.csv
2010q2.csv
2016q4.csv
2009q3.csv
2009q1.csv
2014q4.csv
2012q2.csv
2012q3.csv
2010q1.csv
```

Out[3]:

		valueQ	valueA
cik	**filed**		
883984	**2009-04-23**	738000.0	NaN
	2009-07-24	617000.0	NaN
	2009-10-22	661000.0	NaN
	2010-02-19	629000.0	2645000.0
	2010-04-23	918000.0	2825000.0
...	**...**	...	...
1436229	**2020-03-23**	NaN	NaN
	2020-05-11	NaN	NaN
	2020-08-05	NaN	NaN
	2020-11-04	NaN	NaN
	2021-01-26	NaN	45450.0

93660 rows × 2 columns

Save this table:

In [4]:
```
rnd.to_csv('data/sec/items/RnD.csv')
```

And now we can read the file like this:

In [5]:
```
rnd = pd.read_csv('data/sec/items/RnD.csv', parse_dates=['filed'], index_col=['c
rnd
```

Out[5]:

		valueQ	valueA
cik	**filed**		
883984	**2009-04-23**	738000.0	NaN
	2009-07-24	617000.0	NaN
	2009-10-22	661000.0	NaN
	2010-02-19	629000.0	2645000.0

		valueQ	valueA
cik	filed		
	2010-04-23	918000.0	2825000.0
...	...	...	...
1436229	2020-03-23	NaN	NaN
	2020-05-11	NaN	NaN
	2020-08-05	NaN	NaN
	2020-11-04	NaN	NaN
	2021-01-26	NaN	45450.0

93660 rows × 2 columns

Unstack the quarterly table (put cik as column):

In [6]:
```
rndQ = rnd.valueQ.unstack('cik')
rndQ
```

Out[6]:

cik	1800	2034	2098	2186	2488	2491	2969	3116	3570	4127	...	1809
filed												
2009-04-23	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	
2009-04-29	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	
2009-04-30	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	
2009-05-07	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	
2009-05-14	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	
...	...	...	...	...	...	...	...	...	...	...	...	
2021-01-25	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	
2021-01-26	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	
2021-01-27	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	
2021-01-28	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	
2021-01-29	NaN	NaN	NaN	NaN	573000000.0	NaN	NaN	NaN	NaN	121600000.0	...	

2712 rows × 4038 columns

Get data for SPY:

In [7]:
```
tiingo.get_dataframe('SPY','2009-04-1')
```

Out[7]:

date	close	high	low	open	volume	adjClose	
2009-04-01 00:00:00+00:00	81.059998	81.419998	78.330002	78.529999	377018300	64.099701	64
2009-04-02 00:00:00+00:00	83.430000	84.610001	81.129997	83.080002	476230700	65.973824	66
2009-04-03 00:00:00+00:00	84.260002	84.279999	82.669998	83.489998	284646300	66.630163	66
2009-04-06 00:00:00+00:00	83.599998	84.279999	82.290001	83.339996	264866600	66.108253	66
2009-04-07 00:00:00+00:00	81.650002	82.650002	81.510002	82.250000	258947800	64.566257	65
...	...	...	...	...	...	...	
2021-02-24 00:00:00+00:00	391.770000	392.230000	385.270000	386.330000	72433946	391.770000	392
2021-02-25 00:00:00+00:00	382.330000	391.880000	380.778900	390.410000	144712701	382.330000	391
2021-02-26 00:00:00+00:00	380.360000	385.580000	378.230000	384.350000	149530614	380.360000	385
2021-03-01 00:00:00+00:00	389.580000	390.920000	380.572000	385.590000	105348798	389.580000	390
2021-03-02 00:00:00+00:00	386.540000	390.070000	386.000000	389.820000	77930773	386.540000	390

3000 rows × 12 columns

Use these dates as "trading days":

In [8]:
```
trading_days = tiingo.get_dataframe('SPY','2009-04-1').index.tz_convert(None)
trading_days
```

Out[8]:
```
DatetimeIndex(['2009-04-01', '2009-04-02', '2009-04-03', '2009-04-06',
               '2009-04-07', '2009-04-08', '2009-04-09', '2009-04-13',
               '2009-04-14', '2009-04-15',
               ...
               '2021-02-17', '2021-02-18', '2021-02-19', '2021-02-22',
               '2021-02-23', '2021-02-24', '2021-02-25', '2021-02-26',
               '2021-03-01', '2021-03-02'],
              dtype='datetime64[ns]', name='date', length=3000, freq=None)
```

Combine the trading days and the dates from the R&D table:

In [9]:
```
trading_days.union(rndQ.index)
```

Out[9]:
```
DatetimeIndex(['2009-04-01', '2009-04-02', '2009-04-03', '2009-04-06',
               '2009-04-07', '2009-04-08', '2009-04-09', '2009-04-13',
               '2009-04-14', '2009-04-15',
```

```
               ...
               '2021-02-17', '2021-02-18', '2021-02-19', '2021-02-22',
               '2021-02-23', '2021-02-24', '2021-02-25', '2021-02-26',
               '2021-03-01', '2021-03-02'],
              dtype='datetime64[ns]', length=3014, freq=None)
```

Add these dates to the R&D table:

In [10]:
```
rndQ = rndQ.reindex( trading_days.union(rndQ.index) ).sort_index()
rndQ
```

Out[10]:

cik	1800	2034	2098	2186	2488	2491	2969	3116	3570	4127	...	1809519	1810182
2009-04-01	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
2009-04-02	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
2009-04-03	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
2009-04-06	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
2009-04-07	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...
2021-02-24	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
2021-02-25	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
2021-02-26	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
2021-03-01	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN
2021-03-02	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN

3014 rows × 4038 columns

Get all filing dates (we saved this file previously):

In [11]:
```
filing_dates = pd.read_csv('data/sec/dates/filing_dates.csv', index_col='cik', p
filing_dates
```

Out[11]:
```
cik
1750       2010-09-23
1750       2010-12-21
1750       2011-03-22
1750       2011-07-13
1750       2011-09-23
              ...
1824920    2020-12-18
1824963    2020-12-10
1825024    2020-12-04
1825042    2020-12-29
```

```
1825079   2021-01-15
Name: filed, Length: 250421, dtype: datetime64[ns]
```

Last filing date for our data:

In [12]:
```
last_filing_date_all_firms = filing_dates.max()
last_filing_date_all_firms
```

Out[12]:
```
Timestamp('2021-01-29 00:00:00')
```

When was the last filing date for Red Hat?

In [13]:
```
symbols = pd.read_json('https://www.sec.gov/files/company_tickers.json').transpo

cik = symbols[symbols.ticker=='RHT'].index[0]    # Red Hat

last_filing_date = filing_dates[cik].iloc[-1]
last_filing_date
```

Out[13]:
```
Timestamp('2019-06-28 00:00:00')
```

How many days since Red Hat last filed?

In [14]:
```
days_since_last_filed = (last_filing_date_all_firms - last_filing_date).days
days_since_last_filed
```

Out[14]:
```
581
```

Assumption: if firm filed within last 120 days, then firm still active:

In [15]:
```
last_date_this_firm = trading_days[-1] if days_since_last_filed < 120 else last_
last_date_this_firm
```

Out[15]:
```
Timestamp('2019-06-28 00:00:00')
```

Now fill missing values for Red Hat with previously reported value (ffill) until last day of Red Hat:

In [16]:
```
rndQ.loc[:last_date_this_firm, cik].ffill()#.plot()
```

Out[16]:
```
2009-04-01            NaN
2009-04-02            NaN
2009-04-03            NaN
2009-04-06            NaN
2009-04-07            NaN
                   ...
2019-06-24    171461000.0
2019-06-25    171461000.0
2019-06-26    171461000.0
2019-06-27    171461000.0
2019-06-28    182961000.0
Name: 1087423, Length: 2592, dtype: float64
```

Last filing date for Microsoft:

In [17]:
```
cik = symbols[symbols.ticker=='MSFT'].index[0]
```

```
last_filing_date = filing_dates[cik].iloc[-1]
last_filing_date
```

Out[17]:
```
Timestamp('2021-01-26 00:00:00')
```

How many days sincce Microsoft last filed?

In [18]:
```
days_since_last_filed = (last_filing_date_all_firms - last_filing_date).days
days_since_last_filed
```

Out[18]:
```
3
```

Last date for Microsoft:

In [19]:
```
last_date_this_firm = trading_days[-1] if days_since_last_filed < 120 else last_
last_date_this_firm
```

Out[19]:
```
Timestamp('2021-03-02 00:00:00')
```

Forward fill R&D values:

In [20]:
```
rndQ.loc[:last_date_this_firm, cik].ffill()
```

Out[20]:
```
2009-04-01             NaN
2009-04-02             NaN
2009-04-03             NaN
2009-04-06             NaN
2009-04-07             NaN
                   ...
2021-02-24    4.899000e+09
2021-02-25    4.899000e+09
2021-02-26    4.899000e+09
2021-03-01    4.899000e+09
2021-03-02    4.899000e+09
Name: 789019, Length: 3014, dtype: float64
```

Put this into a function:

In [21]:
```
def ffill_values(item, dates):
    data = item.unstack('cik')
    data = data.reindex(dates.union(data.index)).sort_index()           # Add sp
    filing_dates = pd.read_csv('data/sec/dates/filing_dates.csv', index_col='cik
    last_filing_date_all_firms = filing_dates.max()                     # Most r

    for cik in data.columns:                                            # Loop o
        last_filing_date      = pd.Series(filing_dates[cik]).iloc[-1]   # Last d
        days_since_last_filed = (last_filing_date_all_firms - last_filing_date).
        last_date_this_firm   = dates[-1] if days_since_last_filed < 120 else la
        data.loc[:last_date_this_firm, cik].ffill(inplace=True)         # Forwar

    return data.loc[dates]                                              # Return
```

Use function like this:

In [22]:
```
rndQ = ffill_values(rnd.valueQ, trading_days)
```

```
rndQ
```

Out[22]:

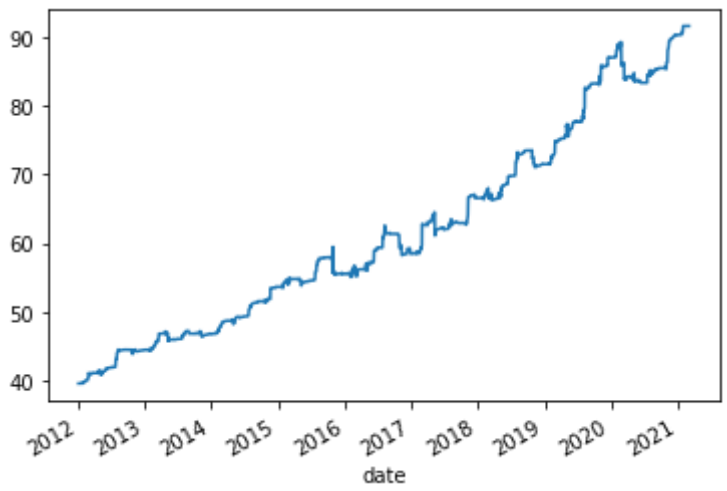
cik	1800	2034	2098	2186	2488	2491	2969	3116	3570	41
date										
2009-04-01	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
2009-04-02	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
2009-04-03	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
2009-04-06	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
2009-04-07	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
...	...	...	...	...	...	...	...	...	...	
2021-02-24	580000000.0	NaN	NaN	NaN	573000000.0	NaN	27100000.0	NaN	0.0	12160000
2021-02-25	580000000.0	NaN	NaN	NaN	573000000.0	NaN	27100000.0	NaN	0.0	12160000
2021-02-26	580000000.0	NaN	NaN	NaN	573000000.0	NaN	27100000.0	NaN	0.0	12160000
2021-03-01	580000000.0	NaN	NaN	NaN	573000000.0	NaN	27100000.0	NaN	0.0	12160000
2021-03-02	580000000.0	NaN	NaN	NaN	573000000.0	NaN	27100000.0	NaN	0.0	12160000

3000 rows × 4038 columns

Total R&D for U.S. stock market:

In [23]:
```
rndQ.sum('columns').div(10**9)['2012':].plot()
```

Out[23]:
```
<AxesSubplot:xlabel='date'>
```



Match CIKs to ticker symbols

Ticker symbol file from SEC:

In [24]:
```
sec = pd.read_json('https://www.sec.gov/files/company_tickers.json').transpose()
sec = sec.rename(columns={'cik_str':'cik'})
sec
```

Out[24]:

	cik	ticker	title
0	320193	AAPL	Apple Inc.
1	789019	MSFT	MICROSOFT CORP
2	1018724	AMZN	AMAZON COM INC
3	1652044	GOOG	Alphabet Inc.
4	1293451	TCEHY	Tencent Holdings Ltd
...	...	...	...
11185	1830081	CFVIU	CF Acquisition Corp. VI
11186	1830210	STPC-UN	Star Peak Corp II
11187	1830188	DHHCU	DiamondHead Holdings Corp.
11188	1830531	PAQCU	Provident Acquisition Corp.
11189	1830795	QFTA-UN	Quantum FinTech Acquisition Corp

11190 rows × 3 columns

Get Google CIK:

In [25]:
```
sec[sec.ticker=='GOOG']
```

Out[25]:

	cik	ticker	title
3	1652044	GOOG	Alphabet Inc.

Get all rows for this CIK:

In [26]:
```
sec[sec.cik==1652044]
```

Out[26]:

	cik	ticker	title
3	1652044	GOOG	Alphabet Inc.
8583	1652044	GOOGL	Alphabet Inc.

So Google has 2 share classes. Which one should we use? We need more information for this decision.

Get ticker symbol file from tiingo:

In [27]:
```
r = requests.get('https://apimedia.tiingo.com/docs/tiingo/daily/supported_ticker
```

```
z = zipfile.ZipFile(io.BytesIO(r.content))

z.namelist()
```

Out[27]: ['supported_tickers.csv']

Open this file:

In [28]:
```
tngo = pd.read_csv(z.open('supported_tickers.csv'))
tngo
```

Out[28]:

	ticker	exchange	assetType	priceCurrency	startDate	endDate
0	000001	SHE	Stock	CNY	2007-08-30	2021-03-01
1	000002	SHE	Stock	CNY	2000-01-04	2021-03-01
2	000003	SHE	Stock	CNY	NaN	NaN
3	000004	SHE	Stock	CNY	2007-08-31	2021-03-01
4	000005	SHE	Stock	CNY	2001-01-02	2021-03-01
...	...	...	...	...	...	...
92268	ZZK	NYSE ARCA	Stock	USD	2020-07-22	2021-03-01
92269	ZZLL	OTCQB	Stock	USD	2017-09-26	2021-03-01
92270	ZZLLD	OTCBB	Stock	USD	2017-10-06	2017-10-30
92271	ZZZ	NYSE ARCA	Stock	USD	2014-10-31	2021-03-01
92272	ZZZOF	PINK	Stock	USD	2017-09-22	2021-03-01

92273 rows × 6 columns

Merge the SEC and the tiingo table:

In [29]:
```
all_shares = sec.merge(tngo, on='ticker', how='outer')
all_shares
```

Out[29]:

	cik	ticker	title	exchange	assetType	priceCurrency	startDate	endDate
0	320193	AAPL	Apple Inc.	NASDAQ	Stock	USD	1980-12-12	2021-03-01
1	789019	MSFT	MICROSOFT CORP	NASDAQ	Stock	USD	1986-03-13	2021-03-01
2	1018724	AMZN	AMAZON COM INC	NASDAQ	Stock	USD	1997-05-15	2021-03-01
3	1652044	GOOG	Alphabet Inc.	NASDAQ	Stock	USD	2014-03-27	2021-03-01
4	1293451	TCEHY	Tencent Holdings Ltd	PINK	Stock	USD	2008-11-03	2021-03-01
...	...	...	...	...	...	...	...	...

	cik	ticker	title	exchange	assetType	priceCurrency	startDate	endDate
93397	NaN	ZZHGY	NaN	PINK	Stock	USD	NaN	NaN
93398	NaN	ZZK	NaN	NYSE ARCA	Stock	USD	2020-07-22	2021-03-01
93399	NaN	ZZLLD	NaN	OTCBB	Stock	USD	2017-10-06	2017-10-30
93400	NaN	ZZZ	NaN	NYSE ARCA	Stock	USD	2014-10-31	2021-03-01
93401	NaN	ZZZOF	NaN	PINK	Stock	USD	2017-09-22	2021-03-01

93402 rows × 8 columns

Check SPY:

In [30]:
```
all_shares[all_shares.ticker=='SPY']
```

Out[30]:

	cik	ticker	title	exchange	assetType	priceCurrency	startDate	endDate
15	884394	SPY	SPDR S&P 500 ETF TRUST	NYSE ARCA	ETF	USD	1993-01-29	2021-03-01

Exclude ETFs:

In [31]:
```
all_shares = all_shares[all_shares.assetType!='ETF']
```

Check Alphabet again:

In [32]:
```
all_shares[all_shares.cik==1652044]  # CIK Alphabet
```

Out[32]:

	cik	ticker	title	exchange	assetType	priceCurrency	startDate	endDate
3	1652044	GOOG	Alphabet Inc.	NASDAQ	Stock	USD	2014-03-27	2021-03-01
8879	1652044	GOOGL	Alphabet Inc.	NASDAQ	Stock	USD	2004-08-19	2021-03-01

We will assume that the shares first issued are the "primary shares".

Lets select the first share for each firm:

In [33]:
```
symbols = all_shares.sort_values(['cik','startDate']).groupby('cik',as_index=Fal
symbols
```

Out[33]:

cik	ticker	title	exchange	assetType	priceCurrency	startDate	endDate
1750	AIR	AAR CORP	NYSE	Stock	USD	1984-07-19	2021-03-01

cik	ticker	title	exchange	assetType	priceCurrency	startDate	endDate
1800	ABT	ABBOTT LABORATORIES	NYSE	Stock	USD	1983-04-06	2021-03-01
1961	WDDD	WORLDS INC	OTCQB	Stock	USD	1998-10-20	2021-03-01
2098	ACU	ACME UNITED CORP	NYSE MKT	Stock	USD	1984-09-07	2021-03-01
2178	AE	ADAMS RESOURCES & ENERGY, INC.	NYSE MKT	Stock	USD	1984-09-07	2021-03-01
...	...	...	...	...	...	...	...
1846163	TIOA	Tio Tech A	None	None	None	None	None
1846189	JAAC	Just Another Acquisition Corp.	NASDAQ	Stock	USD	None	None
1846996	MSDA	MSD ACQUISITION CORP. / NEW	None	None	None	None	None
1847090	TPBA	TPB Acquisition Corp I	None	None	None	None	None
1847127	GGPI	Gores Guggenheim, Inc.	None	None	None	None	None

8307 rows × 7 columns

Check Alphabet:

In [34]:
```
symbols.loc[[1652044]]
```

Out[34]:

cik	ticker	title	exchange	assetType	priceCurrency	startDate	endDate
1652044	GOOGL	Alphabet Inc.	NASDAQ	Stock	USD	2004-08-19	2021-03-01

Note now we have the shares the Alphabe originally issued (in 2004).

Save the symbols table:

In [35]:
```
Path('data/ticker_symbols/').mkdir(parents=True, exist_ok=True)  # Generate the

symbols.to_csv('data/ticker_symbols/symbols.csv')
```

What are the top 10 firms with higest most recent R&D?

In [36]:
```
rndQ.iloc[-1].nlargest(10)
```

Out[36]:
```
cik
1652044    6.856000e+09
1326801    5.207000e+09
```

```
320193     5.163000e+09
789019     4.899000e+09
50863      3.655000e+09
310158     3.390000e+09
14272      2.499000e+09
858877     1.612000e+09
1341439    1.601000e+09
804328     1.581000e+09
Name: 2021-03-02 00:00:00, dtype: float64
```

Put this into a dataFrame:

In [37]:
```
top_10 = rndQ.iloc[-1].nlargest(10).to_frame('Value')
top_10
```

Out[37]:

cik	Value
1652044	6.856000e+09
1326801	5.207000e+09
320193	5.163000e+09
789019	4.899000e+09
50863	3.655000e+09
310158	3.390000e+09
14272	2.499000e+09
858877	1.612000e+09
1341439	1.601000e+09
804328	1.581000e+09

Which firms are these?

In [38]:
```
top_10.join(symbols)
```

Out[38]:

cik	Value	ticker	title	exchange	assetType	priceCurrency	startDate	er
1652044	6.856000e+09	GOOGL	Alphabet Inc.	NASDAQ	Stock	USD	2004-08-19	
1326801	5.207000e+09	FB	Facebook Inc	NASDAQ	Stock	USD	2012-05-18	
320193	5.163000e+09	AAPL	Apple Inc.	NASDAQ	Stock	USD	1980-12-12	
789019	4.899000e+09	MSFT	MICROSOFT CORP	NASDAQ	Stock	USD	1986-03-13	
50863	3.655000e+09	INTC	INTEL CORP	NASDAQ	Stock	USD	1980-03-17	
310158	3.390000e+09	MRK	Merck & Co., Inc.	NYSE	Stock	USD	1970-01-02	

cik	Value	ticker	title	exchange	assetType	priceCurrency	startDate	en
14272	2.499000e+09	BMY	BRISTOL MYERS SQUIBB CO	NYSE	Stock	USD	1972-06-01	
858877	1.612000e+09	CSCO	CISCO SYSTEMS, INC.	NASDAQ	Stock	USD	1990-03-26	
1341439	1.601000e+09	ORCL	ORACLE CORP	NYSE	Stock	USD	1986-03-12	
804328	1.581000e+09	QCOM	QUALCOMM INC/DE	NASDAQ	Stock	USD	1991-12-16	