SEC Data V: Quarterly and Annual Values

In [1]:
```
import pandas as pd
import numpy  as np
import os
```

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
```

Get all reported R&D expense values for all firms from 2009 until last month:

In [3]:
```
tags = ['ResearchAndDevelopmentExpense']
items = get_items_from_SEC_files(tags) # items is a dictionary of tables.
```

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

R&D table:

In [5]:
```
item = items['ResearchAndDevelopmentExpense']
item
```

Out[5]:

		value_shortest	qtrs_shortest	value_longest	qtrs_longest
cik	**filed**				
883984	**2009-04-23**	738000.0	1	738000.0	1
884905	**2009-04-29**	18000000.0	1	18000000.0	1
1164727	**2009-04-30**	31000000.0	1	31000000.0	1
1080224	**2009-05-14**	558000.0	1	558000.0	1
38074	**2009-05-29**	661294000.0	4	661294000.0	4
...	...	...	...	...	...
1350102	**2021-01-29**	1300000.0	4	1300000.0	4
1385799	**2021-01-29**	182959.0	4	182959.0	4
1438943	**2021-01-29**	71975.0	1	338268.0	3
1551887	**2021-01-29**	14629.0	4	14629.0	4
1580149	**2021-01-29**	938101.0	1	1063112.0	2

68080 rows × 4 columns

Get the filing dates for all firms:

In [6]:
```
def get_all_filing_dates(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   = pd.DataFrame()                                          # Results will

    for filename in filenames:                                          # Loop over al
        data    = pd.read_csv(directory+filename, parse_dates=['filed','ddate'])
        results = results.append( data.groupby(['cik','filed'],as_index=False).f

    return results.sort_values(['cik','filed']).set_index('cik')
```

In [7]:
```
all_filing_dates = get_all_filing_dates()
all_filing_dates
```

Out[7]:

cik	filed
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

250421 rows × 1 columns

Save the the filing dates table:

In [8]:
```
all_filing_dates.to_csv('data/sec/dates/filing_dates.csv')
```

We will need these filing dates now to calculate quarterly/annual R&D expenses.

In [9]:
```
symbols = pd.read_json('https://www.sec.gov/files/company_tickers.json').transpo
symbols[:2]
```

Out[9]:

	ticker	title

cik_str	ticker	title

cik_str		
320193	AAPL	Apple Inc.
789019	MSFT	MICROSOFT CORP

Apple R&D:

In [10]:
```
cik = symbols[symbols.ticker=='AAPL'].index[0] # 320193
item.loc[cik] [:8]
```

Out[10]:

filed	value_shortest	qtrs_shortest	value_longest	qtrs_longest
2009-07-22	3.410000e+08	1	9.750000e+08	3
2009-10-27	1.333000e+09	4	1.333000e+09	4
2010-01-25	3.980000e+08	1	3.980000e+08	1
2010-04-21	4.260000e+08	1	8.240000e+08	2
2010-07-21	4.640000e+08	1	1.288000e+09	3
2010-10-27	1.782000e+09	4	1.782000e+09	4
2011-01-19	5.750000e+08	1	5.750000e+08	1
2011-04-21	5.810000e+08	1	1.156000e+09	2

Example:

In [11]:
```
date   = '2010-10-27'
values = item.loc[cik].value_shortest

# We need to subtract these values:
previous_values = values[:date][-4:-1]
previous_values
```

Out[11]:
```
filed
2010-01-25    398000000.0
2010-04-21    426000000.0
2010-07-21    464000000.0
Name: value_shortest, dtype: float64
```

Subtract the sum of these values from value on that date:

In [12]:
```
values.loc[date] - previous_values.sum()
```

Out[12]:
```
494000000.0
```

Loop over all quarters that are greater than 1:

In [13]:
```
qtrs = item.loc[cik].qtrs_shortest
qtrs[qtrs>1]
```

Out[13]:
```
filed
2009-10-27    4
2010-10-27    4
2011-10-26    4
2012-10-31    4
2013-10-30    4
2014-10-27    4
2015-10-28    4
2016-10-26    4
2017-11-03    4
2018-11-05    4
2019-10-31    4
2020-10-30    4
Name: qtrs_shortest, dtype: int64
```

In [14]:
```
for date,q in qtrs[qtrs>1].iteritems():
    print(date,q)
```

```
2009-10-27 00:00:00 4
2010-10-27 00:00:00 4
2011-10-26 00:00:00 4
2012-10-31 00:00:00 4
2013-10-30 00:00:00 4
2014-10-27 00:00:00 4
2015-10-28 00:00:00 4
2016-10-26 00:00:00 4
2017-11-03 00:00:00 4
2018-11-05 00:00:00 4
2019-10-31 00:00:00 4
2020-10-30 00:00:00 4
```

In [15]:
```
for date,q in qtrs[qtrs>1].iteritems():                    # Loop over all dates
    previous_values = values[:date][-q:-1]                 # Example: for q=3 we
    if len(previous_values) == q-1:                        # If all previous val
        print(date, 'subtract previous values')
    else:
        print('not enough data')
```

```
not enough data
2010-10-27 00:00:00 subtract previous values
2011-10-26 00:00:00 subtract previous values
2012-10-31 00:00:00 subtract previous values
2013-10-30 00:00:00 subtract previous values
2014-10-27 00:00:00 subtract previous values
2015-10-28 00:00:00 subtract previous values
2016-10-26 00:00:00 subtract previous values
2017-11-03 00:00:00 subtract previous values
2018-11-05 00:00:00 subtract previous values
2019-10-31 00:00:00 subtract previous values
2020-10-30 00:00:00 subtract previous values
```

Check R&D of Ford:

In [16]:
```
cik = symbols[symbols.ticker=='F'].index[0]
item.loc[cik]
```

Out[16]:

	value_shortest	qtrs_shortest	value_longest	qtrs_longest
filed				

filed	value_shortest	qtrs_shortest	value_longest	qtrs_longest
2011-02-28	5.000000e+09	4	5.000000e+09	4
2012-02-21	5.300000e+09	4	5.300000e+09	4
2013-02-19	5.500000e+09	4	5.500000e+09	4
2014-02-18	6.400000e+09	4	6.400000e+09	4
2015-02-13	6.900000e+09	4	6.900000e+09	4
2016-02-11	6.700000e+09	4	6.700000e+09	4
2017-02-09	7.300000e+09	4	7.300000e+09	4
2018-02-08	8.000000e+09	4	8.000000e+09	4
2019-02-21	8.200000e+09	4	8.200000e+09	4
2020-02-05	7.400000e+09	4	7.400000e+09	4

Note how Ford reports R&D only once per year (in their 10-K).

All filing dates of Ford:

In [17]:
```
all_filing_dates.loc[cik].filed
```

Out[17]:
```
cik
37996   2009-08-05
37996   2009-11-06
37996   2010-02-25
37996   2010-05-07
37996   2010-08-06
37996   2010-11-08
37996   2011-02-28
37996   2011-05-10
37996   2011-08-05
37996   2011-11-04
37996   2012-02-21
37996   2012-05-04
37996   2012-08-03
37996   2012-11-02
37996   2013-02-19
37996   2013-05-01
37996   2013-07-31
37996   2013-10-31
37996   2014-02-18
37996   2014-05-01
37996   2014-07-31
37996   2014-10-31
37996   2015-02-13
37996   2015-04-28
37996   2015-07-28
37996   2015-10-27
37996   2016-02-11
37996   2016-04-28
37996   2016-07-28
37996   2016-10-27
37996   2017-02-09
37996   2017-04-27
37996   2017-07-26
```

```
37996   2017-10-26
37996   2018-02-08
37996   2018-04-26
37996   2018-07-26
37996   2018-10-25
37996   2019-02-21
37996   2019-04-26
37996   2019-07-25
37996   2019-10-24
37996   2020-02-05
37996   2020-04-29
37996   2020-07-31
37996   2020-10-29
Name: filed, dtype: datetime64[ns]
```

Add these filing dates to Fords R&D table:

In [18]:
```
item.loc[cik].reindex(all_filing_dates.loc[cik].filed)
```

Out[18]:

filed	value_shortest	qtrs_shortest	value_longest	qtrs_longest
2009-08-05	NaN	NaN	NaN	NaN
2009-11-06	NaN	NaN	NaN	NaN
2010-02-25	NaN	NaN	NaN	NaN
2010-05-07	NaN	NaN	NaN	NaN
2010-08-06	NaN	NaN	NaN	NaN
2010-11-08	NaN	NaN	NaN	NaN
2011-02-28	5.000000e+09	4.0	5.000000e+09	4.0
2011-05-10	NaN	NaN	NaN	NaN
2011-08-05	NaN	NaN	NaN	NaN
2011-11-04	NaN	NaN	NaN	NaN
2012-02-21	5.300000e+09	4.0	5.300000e+09	4.0
2012-05-04	NaN	NaN	NaN	NaN
2012-08-03	NaN	NaN	NaN	NaN
2012-11-02	NaN	NaN	NaN	NaN
2013-02-19	5.500000e+09	4.0	5.500000e+09	4.0
2013-05-01	NaN	NaN	NaN	NaN
2013-07-31	NaN	NaN	NaN	NaN
2013-10-31	NaN	NaN	NaN	NaN
2014-02-18	6.400000e+09	4.0	6.400000e+09	4.0
2014-05-01	NaN	NaN	NaN	NaN
2014-07-31	NaN	NaN	NaN	NaN
2014-10-31	NaN	NaN	NaN	NaN
2015-02-13	6.900000e+09	4.0	6.900000e+09	4.0

filed	value_shortest	qtrs_shortest	value_longest	qtrs_longest
2015-04-28	NaN	NaN	NaN	NaN
2015-07-28	NaN	NaN	NaN	NaN
2015-10-27	NaN	NaN	NaN	NaN
2016-02-11	6.700000e+09	4.0	6.700000e+09	4.0
2016-04-28	NaN	NaN	NaN	NaN
2016-07-28	NaN	NaN	NaN	NaN
2016-10-27	NaN	NaN	NaN	NaN
2017-02-09	7.300000e+09	4.0	7.300000e+09	4.0
2017-04-27	NaN	NaN	NaN	NaN
2017-07-26	NaN	NaN	NaN	NaN
2017-10-26	NaN	NaN	NaN	NaN
2018-02-08	8.000000e+09	4.0	8.000000e+09	4.0
2018-04-26	NaN	NaN	NaN	NaN
2018-07-26	NaN	NaN	NaN	NaN
2018-10-25	NaN	NaN	NaN	NaN
2019-02-21	8.200000e+09	4.0	8.200000e+09	4.0
2019-04-26	NaN	NaN	NaN	NaN
2019-07-25	NaN	NaN	NaN	NaN
2019-10-24	NaN	NaN	NaN	NaN
2020-02-05	7.400000e+09	4.0	7.400000e+09	4.0
2020-04-29	NaN	NaN	NaN	NaN
2020-07-31	NaN	NaN	NaN	NaN
2020-10-29	NaN	NaN	NaN	NaN

Now put all this code into these functions:

In [19]:

```
def calculate_quarterly_values(item):                                  # item: dataF
    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 a
        filing_dates = pd.Series(all_filing_dates.loc[cik].filed)      # All filing
        values = item.loc[cik].value_shortest.reindex(filing_dates)
        qtrs   = item.loc[cik].qtrs_shortest.astype(int)
        for date,q in qtrs[qtrs>1].iteritems():                        # Loop over a
            previous_values = values[:date][-q:-1]                     # Example: fo
            if len(previous_values) == q-1:                            # If all prev
                values[date] -= previous_values.sum(skipna=False)      # Subtract pr
            else:
                values[date]  = np.nan
```

```
        result = result.append( pd.DataFrame({'cik':cik, 'filed':values.index, '

    return result.set_index(['cik','filed']).value


def calculate_annual_values(item, values_1Q):                          # item: dataF
    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 a
        filing_dates = pd.Series(all_filing_dates.loc[cik].filed)  # All filing
        valuesQ      = values_1Q.loc[cik].reindex(filing_dates)    # Use 1-quart
        values       = item     .loc[cik].value_longest.reindex(filing_dates)
        qtrs         = item     .loc[cik].qtrs_longest.astype(int)
        for date,q in qtrs[qtrs<4].iteritems():                        # Loop over a
            previous_values = valuesQ[:date][-4:-q]                    # Example: fo
            if len(previous_values) == 4-q:                            # If all prev
                values[date] += previous_values.sum(skipna=False)  # Add previou
            else:
                values[date]  = np.nan
        result = result.append( pd.DataFrame({'cik':cik, 'filed':values.index, '

    return result.set_index(['cik','filed']).value
```

In [20]:
```
item = items['ResearchAndDevelopmentExpense']
```

In [21]:
```
itemQ = calculate_quarterly_values(item)
itemQ
```

Out[21]:
```
cik      filed
883984   2009-04-23    738000.0
         2009-07-24    617000.0
         2009-10-22    661000.0
         2010-02-19    629000.0
         2010-04-23    918000.0
                          ...
1436229  2020-03-23       NaN
         2020-05-11       NaN
         2020-08-05       NaN
         2020-11-04       NaN
         2021-01-26       NaN
Name: value, Length: 93660, dtype: float64
```

In [22]:
```
itemA = calculate_annual_values(item, itemQ)
itemA
```

Out[22]:
```
cik      filed
883984   2009-04-23          NaN
         2009-07-24          NaN
         2009-10-22          NaN
         2010-02-19    2645000.0
         2010-04-23    2825000.0
                          ...
1436229  2020-03-23          NaN
         2020-05-11          NaN
         2020-08-05          NaN
         2020-11-04          NaN
```

```
          2021-01-26      45450.0
Name: value, Length: 93660, dtype: float64
```

Create a table to compare the reported values and the quarterly and annual values:

In [24]:

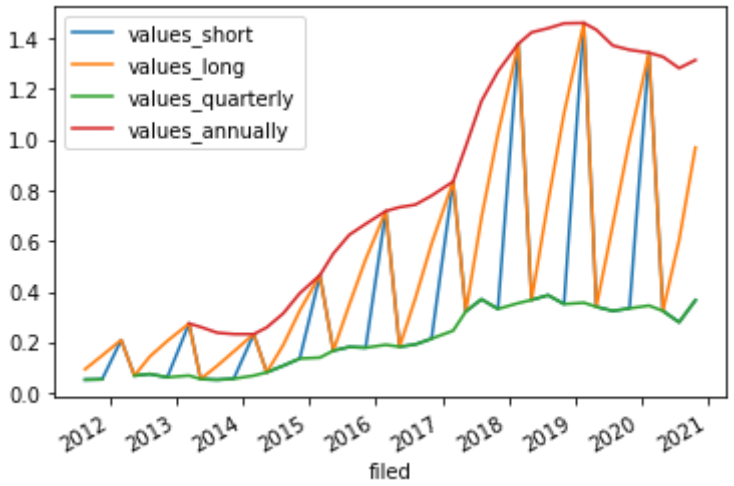
```
cik = symbols[symbols.ticker=='TSLA'].index[0]

t = pd.DataFrame()
t['values_short']     = item .loc[cik].value_shortest
t['values_long']      = item .loc[cik].value_longest
t['values_quarterly'] = itemQ.loc[cik]
t['values_annually']  = itemA.loc[cik]

t.div(10**9).plot()
```

Out[24]:

```
<AxesSubplot:xlabel='filed'>
```



In [ ]: