Extracting Stock Data Using a Python Library

A company's stock share is a piece of the company more precisely:

A stock (also known as equity) is a security that represents the ownership of a fraction of a corporation. This entitles the owner of the stock to a proportion of the corporation's assets and profits equal to how much stock they own. Units of stock are called "shares." [1]

An investor can buy a stock and sell it later. If the stock price increases, the investor profits, If it decreases,the investor with incur a loss. Determining the stock price is complex; it depends on the number of outstanding shares, the size of the company's future profits, and much more. People trade stocks throughout the day the stock ticker is a report of the price of a certain stock, updated continuously throughout the trading session by the various stock market exchanges.

You are a data scientist working for a hedge fund; it's your job to determine any suspicious stock activity. In this lab you will extract stock data using a Python library. We will use the yfinance library, it allows us to extract data for stocks returning data in a pandas dataframe. You will use the lab to extract.

Table of Contents

Estimated Time Needed: **30 min**

In [9]:
```
!pip install yfinance
!pip install matplotlib
# !pip install pandas==1.3.3
```

```
Collecting yfinance
  Downloading yfinance-0.2.54-py2.py3-none-any.whl.metadata (5.8 kB)
Collecting pandas>=1.3.0 (from yfinance)
  Downloading pandas-2.2.3-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_6
4.whl.metadata (89 kB)
Collecting numpy>=1.16.5 (from yfinance)
  Downloading numpy-2.2.3-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.
whl.metadata (62 kB)
Requirement already satisfied: requests>=2.31 in /opt/conda/lib/python3.12/site-p
ackages (from yfinance) (2.32.3)
Collecting multitasking>=0.0.7 (from yfinance)
  Downloading multitasking-0.0.11-py3-none-any.whl.metadata (5.5 kB)
Requirement already satisfied: platformdirs>=2.0.0 in /opt/conda/lib/python3.12/s
ite-packages (from yfinance) (4.3.6)
Requirement already satisfied: pytz>=2022.5 in /opt/conda/lib/python3.12/site-pac
kages (from yfinance) (2024.2)
Requirement already satisfied: frozendict>=2.3.4 in /opt/conda/lib/python3.12/sit
e-packages (from yfinance) (2.4.6)
Collecting peewee>=3.16.2 (from yfinance)
  Downloading peewee-3.17.9.tar.gz (3.0 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.0/3.0 MB 53.5 MB/s eta 0:00:00
  Installing build dependencies ... one
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Requirement already satisfied: beautifulsoup4>=4.11.1 in /opt/conda/lib/python3.1
2/site-packages (from yfinance) (4.12.3)
Requirement already satisfied: soupsieve>1.2 in /opt/conda/lib/python3.12/site-pa
ckages (from beautifulsoup4>=4.11.1->yfinance) (2.5)
Requirement already satisfied: python-dateutil>=2.8.2 in /opt/conda/lib/python3.1
2/site-packages (from pandas>=1.3.0->yfinance) (2.9.0.post0)
Collecting tzdata>=2022.7 (from pandas>=1.3.0->yfinance)
  Downloading tzdata-2025.1-py2.py3-none-any.whl.metadata (1.4 kB)
Requirement already satisfied: charset_normalizer<4,>=2 in /opt/conda/lib/python
3.12/site-packages (from requests>=2.31->yfinance) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/lib/python3.12/site-pac
kages (from requests>=2.31->yfinance) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /opt/conda/lib/python3.12/si
te-packages (from requests>=2.31->yfinance) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/lib/python3.12/si
te-packages (from requests>=2.31->yfinance) (2024.12.14)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.12/site-package
s (from python-dateutil>=2.8.2->pandas>=1.3.0->yfinance) (1.17.0)
Downloading yfinance-0.2.54-py2.py3-none-any.whl (108 kB)
Downloading multitasking-0.0.11-py3-none-any.whl (8.5 kB)
Downloading numpy-2.2.3-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.wh
l (16.1 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 16.1/16.1 MB 111.8 MB/s eta 0:00:00
Downloading pandas-2.2.3-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.w
hl (12.7 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.7/12.7 MB 114.5 MB/s eta 0:00:00
Downloading tzdata-2025.1-py2.py3-none-any.whl (346 kB)
Building wheels for collected packages: peewee
  Building wheel for peewee (pyproject.toml) ... one
  Created wheel for peewee: filename=peewee-3.17.9-cp312-cp312-linux_x86_64.whl s
ize=303802 sha256=e39395351a2511da2dd4d765f0efd8a03d0bb0d1ae3bc6ba08c9310f4092d97
e
  Stored in directory: /home/jupyterlab/.cache/pip/wheels/43/ef/2d/2c51d496bf0849
45ffdf838b4cc8767b8ba1cc20eb41588831
Successfully built peewee
Installing collected packages: peewee, multitasking, tzdata, numpy, pandas, yfina
```

```
nce
Successfully installed multitasking-0.0.11 numpy-2.2.3 pandas-2.2.3 peewee-3.17.9
tzdata-2025.1 yfinance-0.2.54
Collecting matplotlib
  Downloading matplotlib-3.10.0-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x
86_64.whl.metadata (11 kB)
Collecting contourpy>=1.0.1 (from matplotlib)
  Downloading contourpy-1.3.1-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86
_64.whl.metadata (5.4 kB)
Collecting cycler>=0.10 (from matplotlib)
  Downloading cycler-0.12.1-py3-none-any.whl.metadata (3.8 kB)
Collecting fonttools>=4.22.0 (from matplotlib)
  Downloading fonttools-4.56.0-cp312-cp312-manylinux_2_5_x86_64.manylinux1_x86_6
4.manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (101 kB)
Collecting kiwisolver>=1.3.1 (from matplotlib)
  Downloading kiwisolver-1.4.8-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x8
6_64.whl.metadata (6.2 kB)
Requirement already satisfied: numpy>=1.23 in /opt/conda/lib/python3.12/site-pack
ages (from matplotlib) (2.2.3)
Requirement already satisfied: packaging>=20.0 in /opt/conda/lib/python3.12/site-
packages (from matplotlib) (24.2)
Collecting pillow>=8 (from matplotlib)
  Downloading pillow-11.1.0-cp312-cp312-manylinux_2_28_x86_64.whl.metadata (9.1 k
B)
Collecting pyparsing>=2.3.1 (from matplotlib)
  Downloading pyparsing-3.2.1-py3-none-any.whl.metadata (5.0 kB)
Requirement already satisfied: python-dateutil>=2.7 in /opt/conda/lib/python3.12/
site-packages (from matplotlib) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.12/site-package
s (from python-dateutil>=2.7->matplotlib) (1.17.0)
Downloading matplotlib-3.10.0-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86
_64.whl (8.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.6/8.6 MB 77.4 MB/s eta 0:00:00
Downloading contourpy-1.3.1-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_6
4.whl (323 kB)
Downloading cycler-0.12.1-py3-none-any.whl (8.3 kB)
Downloading fonttools-4.56.0-cp312-cp312-manylinux_2_5_x86_64.manylinux1_x86_64.m
anylinux_2_17_x86_64.manylinux2014_x86_64.whl (4.9 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.9/4.9 MB 75.9 MB/s eta 0:00:00
Downloading kiwisolver-1.4.8-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_
64.whl (1.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.5/1.5 MB 80.0 MB/s eta 0:00:00
Downloading pillow-11.1.0-cp312-cp312-manylinux_2_28_x86_64.whl (4.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.5/4.5 MB 68.8 MB/s eta 0:00:00
Downloading pyparsing-3.2.1-py3-none-any.whl (107 kB)
Installing collected packages: pyparsing, pillow, kiwisolver, fonttools, cycler,
contourpy, matplotlib
Successfully installed contourpy-1.3.1 cycler-0.12.1 fonttools-4.56.0 kiwisolver-
1.4.8 matplotlib-3.10.0 pillow-11.1.0 pyparsing-3.2.1
```

In [13]:
```
import yfinance as yf
import pandas as pd
```

Using the yfinance Library to Extract Stock Data

Using the `Ticker` module we can create an object that will allow us to access functions to extract data. To do this we need to provide the ticker symbol for the stock, here the company is Apple and the ticker symbol is `AAPL`.

In [14]:
```
apple = yf.Ticker("AAPL")
```

Now we can access functions and variables to extract the type of data we need. You can view them and what they represent here https://aroussi.com/post/python-yahoo-finance.

In [15]:
```
!wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeve
```

```
--2025-02-27 00:50:24--  https://cf-courses-data.s3.us.cloud-object-storage.appdo
main.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/data/apple.json
Resolving cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-courses-
data.s3.us.cloud-object-storage.appdomain.cloud)... 169.63.118.104
Connecting to cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-cour
ses-data.s3.us.cloud-object-storage.appdomain.cloud)|169.63.118.104|:443... conne
cted.
200 OKequest sent, awaiting response...
Length: 5699 (5.6K) [application/json]
Saving to: ‘apple.json’

apple.json          100%[===================>]   5.57K  --.-KB/s    in 0s

2025-02-27 00:50:24 (484 MB/s) - ‘apple.json’ saved [5699/5699]
```

Stock Info

Using the attribute `info` we can extract information about the stock as a Python dictionary.

In [16]:
```
import json
with open('apple.json') as json_file:
    apple_info = json.load(json_file)
    # Print the type of data variable
    #print("Type:", type(apple_info))
apple_info
```

Out[16]:
{'zip': '95014',
 'sector': 'Technology',
 'fullTimeEmployees': 100000,
 'longBusinessSummary': 'Apple Inc. designs, manufactures, and markets smartpho
nes, personal computers, tablets, wearables, and accessories worldwide. It also
sells various related services. In addition, the company offers iPhone, a line
of smartphones; Mac, a line of personal computers; iPad, a line of multi-purpos
e tablets; AirPods Max, an over-ear wireless headphone; and wearables, home, an
d accessories comprising AirPods, Apple TV, Apple Watch, Beats products, HomePo
d, and iPod touch. Further, it provides AppleCare support services; cloud servi
ces store services; and operates various platforms, including the App Store tha
t allow customers to discover and download applications and digital content, su
ch as books, music, video, games, and podcasts. Additionally, the company offer
s various services, such as Apple Arcade, a game subscription service; Apple Mu
sic, which offers users a curated listening experience with on-demand radio sta
tions; Apple News+, a subscription news and magazine service; Apple TV+, which
offers exclusive original content; Apple Card, a co-branded credit card; and Ap
ple Pay, a cashless payment service, as well as licenses its intellectual prope
rty. The company serves consumers, and small and mid-sized businesses; and the
education, enterprise, and government markets. It distributes third-party appli
cations for its products through the App Store. The company also sells its prod
ucts through its retail and online stores, and direct sales force; and third-pa
rty cellular network carriers, wholesalers, retailers, and resellers. Apple In
c. was incorporated in 1977 and is headquartered in Cupertino, California.',
 'city': 'Cupertino',
 'phone': '408 996 1010',
 'state': 'CA',
 'country': 'United States',
 'companyOfficers': [],
 'website': 'https://www.apple.com',
 'maxAge': 1,
 'address1': 'One Apple Park Way',
 'industry': 'Consumer Electronics',
 'ebitdaMargins': 0.33890998,
 'profitMargins': 0.26579002,
 'grossMargins': 0.43019,
 'operatingCashflow': 112241000448,
 'revenueGrowth': 0.112,
 'operatingMargins': 0.309,
 'ebitda': 128217997312,
 'targetLowPrice': 160,
 'recommendationKey': 'buy',
 'grossProfits': 152836000000,
 'freeCashflow': 80153247744,
 'targetMedianPrice': 199.5,
 'currentPrice': 177.77,
 'earningsGrowth': 0.25,
 'currentRatio': 1.038,
 'returnOnAssets': 0.19875,
 'numberOfAnalystOpinions': 44,
 'targetMeanPrice': 193.53,
 'debtToEquity': 170.714,
 'returnOnEquity': 1.45567,
 'targetHighPrice': 215,
 'totalCash': 63913000960,
 'totalDebt': 122797998080,
 'totalRevenue': 378323009536,
 'totalCashPerShare': 3.916,
 'financialCurrency': 'USD',
 'revenuePerShare': 22.838,

```
'quickRatio': 0.875,
'recommendationMean': 1.8,
'exchange': 'NMS',
'shortName': 'Apple Inc.',
'longName': 'Apple Inc.',
'exchangeTimezoneName': 'America/New_York',
'exchangeTimezoneShortName': 'EDT',
'isEsgPopulated': False,
'gmtOffSetMilliseconds': '-14400000',
'quoteType': 'EQUITY',
'symbol': 'AAPL',
'messageBoardId': 'finmb_24937',
'market': 'us_market',
'annualHoldingsTurnover': None,
'enterpriseToRevenue': 7.824,
'beta3Year': None,
'enterpriseToEbitda': 23.086,
'52WeekChange': 0.4549594,
'morningStarRiskRating': None,
'forwardEps': 6.56,
'revenueQuarterlyGrowth': None,
'sharesOutstanding': 16319399936,
'fundInceptionDate': None,
'annualReportExpenseRatio': None,
'totalAssets': None,
'bookValue': 4.402,
'sharesShort': 111286790,
'sharesPercentSharesOut': 0.0068,
'fundFamily': None,
'lastFiscalYearEnd': 1632528000,
'heldPercentInstitutions': 0.59397,
'netIncomeToCommon': 100554997760,
'trailingEps': 6.015,
'lastDividendValue': 0.22,
'SandP52WeekChange': 0.15217662,
'priceToBook': 40.38392,
'heldPercentInsiders': 0.0007,
'nextFiscalYearEnd': 1695600000,
'yield': None,
'mostRecentQuarter': 1640390400,
'shortRatio': 1.21,
'sharesShortPreviousMonthDate': 1644883200,
'floatShares': 16302795170,
'beta': 1.185531,
'enterpriseValue': 2959991898112,
'priceHint': 2,
'threeYearAverageReturn': None,
'lastSplitDate': 1598832000,
'lastSplitFactor': '4:1',
'legalType': None,
'lastDividendDate': 1643932800,
'morningStarOverallRating': None,
'earningsQuarterlyGrowth': 0.204,
'priceToSalesTrailing12Months': 7.668314,
'dateShortInterest': 1647302400,
'pegRatio': 1.94,
'ytdReturn': None,
'forwardPE': 27.099087,
'lastCapGain': None,
'shortPercentOfFloat': 0.0068,
```

```
'sharesShortPriorMonth': 108944701,
'impliedSharesOutstanding': 0,
'category': None,
'fiveYearAverageReturn': None,
'previousClose': 178.96,
'regularMarketOpen': 178.55,
'twoHundredDayAverage': 156.03505,
'trailingAnnualDividendYield': 0.004833482,
'payoutRatio': 0.1434,
'volume24Hr': None,
'regularMarketDayHigh': 179.61,
'navPrice': None,
'averageDailyVolume10Day': 93823630,
'regularMarketPreviousClose': 178.96,
'fiftyDayAverage': 166.498,
'trailingAnnualDividendRate': 0.865,
'open': 178.55,
'toCurrency': None,
'averageVolume10days': 93823630,
'expireDate': None,
'algorithm': None,
'dividendRate': 0.88,
'exDividendDate': 1643932800,
'circulatingSupply': None,
'startDate': None,
'regularMarketDayLow': 176.7,
'currency': 'USD',
'trailingPE': 29.55445,
'regularMarketVolume': 92633154,
'lastMarket': None,
'maxSupply': None,
'openInterest': None,
'marketCap': 2901099675648,
'volumeAllCurrencies': None,
'strikePrice': None,
'averageVolume': 95342043,
'dayLow': 176.7,
'ask': 178.53,
'askSize': 800,
'volume': 92633154,
'fiftyTwoWeekHigh': 182.94,
'fromCurrency': None,
'fiveYearAvgDividendYield': 1.13,
'fiftyTwoWeekLow': 122.25,
'bid': 178.4,
'tradeable': False,
'dividendYield': 0.005,
'bidSize': 3200,
'dayHigh': 179.61,
'regularMarketPrice': 177.77,
'preMarketPrice': 178.38,
'logo_url': 'https://logo.clearbit.com/apple.com'}
```


We can get the `'country'` using the key country

In [17]:
```
apple_info['country']
```

Out[17]:
```
'United States'
```

Extracting Share Price

A share is the single smallest part of a company's stock that you can buy, the prices of these shares fluctuate over time. Using the `history()` method we can get the share price of the stock over a certain period of time. Using the `period` parameter we can set how far back from the present to get data. The options for `period` are 1 day (1d), 5d, 1 month (1mo) , 3mo, 6mo, 1 year (1y), 2y, 5y, 10y, ytd, and max.

In [18]:
```
apple_share_price_data = apple.history(period="max")
```

The format that the data is returned in is a Pandas DataFrame. With the `Date` as the index the share `Open` , `High` , `Low` , `Close` , `Volume` , and `Stock Splits` are given for each day.

In [19]:
```
apple_share_price_data.head()
```

Out[19]:

Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
1980-12-12 00:00:00-05:00	0.098726	0.099155	0.098726	0.098726	469033600	0.0	0.0
1980-12-15 00:00:00-05:00	0.094005	0.094005	0.093575	0.093575	175884800	0.0	0.0
1980-12-16 00:00:00-05:00	0.087136	0.087136	0.086707	0.086707	105728000	0.0	0.0
1980-12-17 00:00:00-05:00	0.088853	0.089282	0.088853	0.088853	86441600	0.0	0.0
1980-12-18 00:00:00-05:00	0.091429	0.091858	0.091429	0.091429	73449600	0.0	0.0

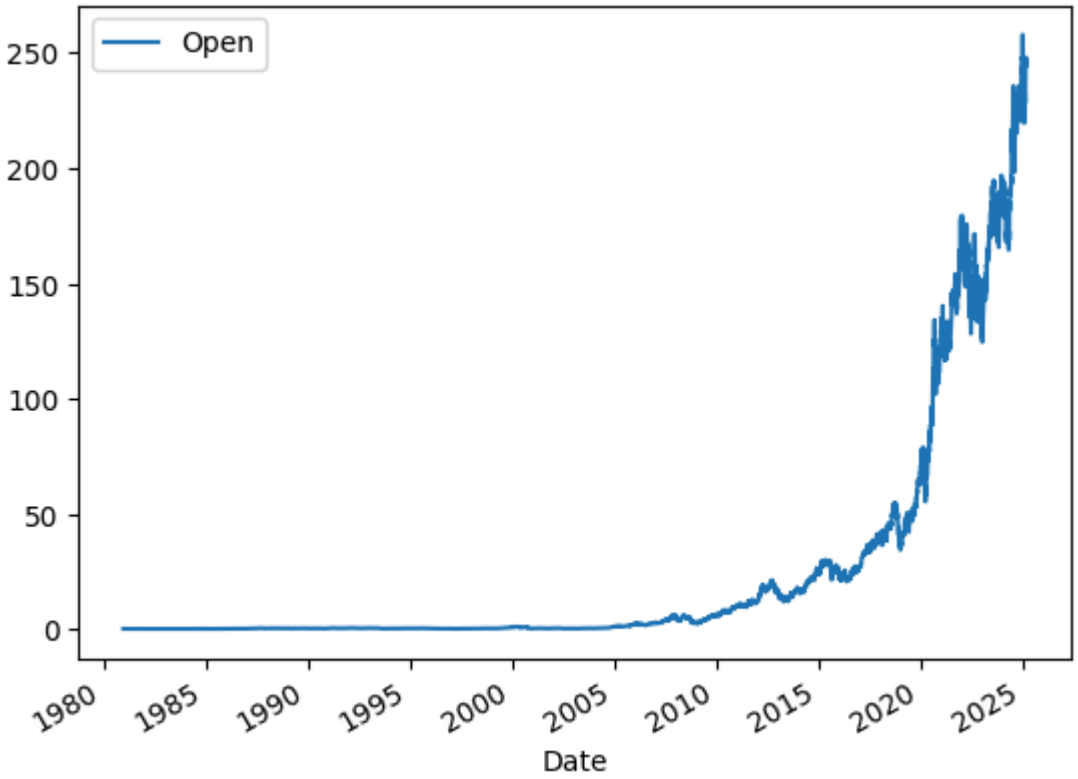
We can reset the index of the DataFrame with the `reset_index` function. We also set the `inplace` paramter to `True` so the change takes place to the DataFrame itself.

In [20]:
```
apple_share_price_data.reset_index(inplace=True)
```

We can plot the `Open` price against the `Date` :

In [21]:
```
apple_share_price_data.plot(x="Date", y="Open")
```

Out[21]:
```
<Axes: xlabel='Date'>
```



Extracting Dividends

Dividends are the distribution of a companys profits to shareholders. In this case they are defined as an amount of money returned per share an investor owns. Using the variable `dividends` we can get a dataframe of the data. The period of the data is given by the period defined in the 'history` function.

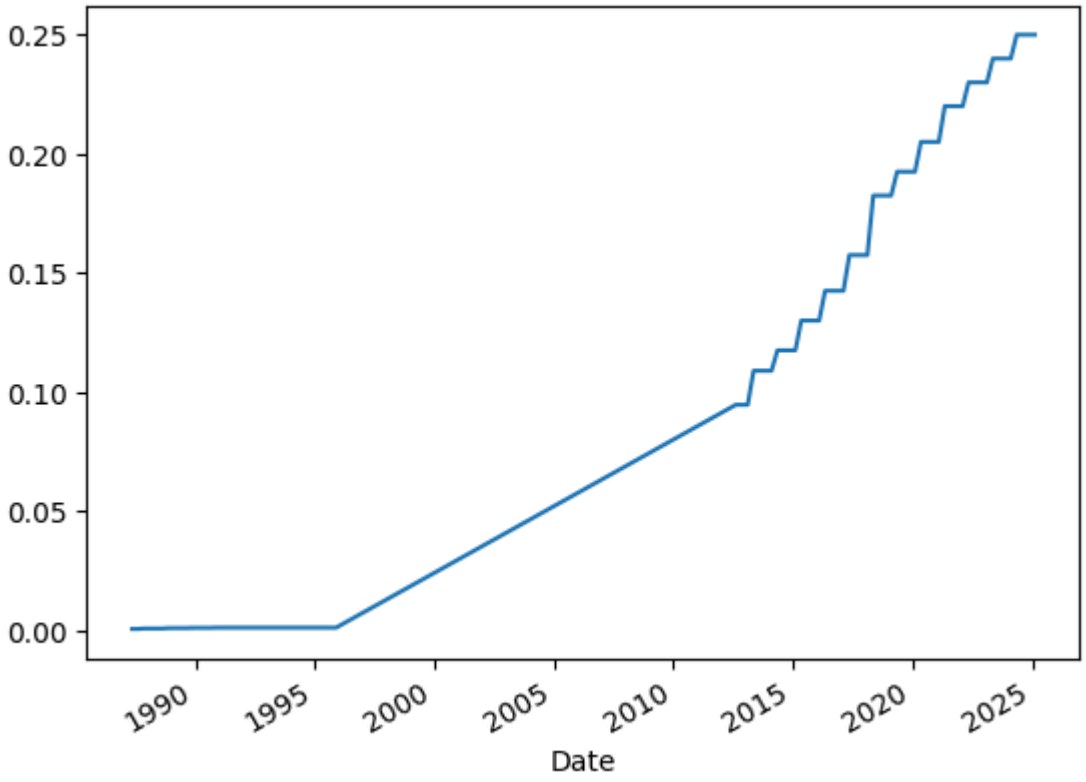
In [22]:
```
apple.dividends
```

Out[22]:
```
Date
1987-05-11 00:00:00-04:00    0.000536
1987-08-10 00:00:00-04:00    0.000536
1987-11-17 00:00:00-05:00    0.000714
1988-02-12 00:00:00-05:00    0.000714
1988-05-16 00:00:00-04:00    0.000714
                               ...
2024-02-09 00:00:00-05:00    0.240000
2024-05-10 00:00:00-04:00    0.250000
2024-08-12 00:00:00-04:00    0.250000
2024-11-08 00:00:00-05:00    0.250000
2025-02-10 00:00:00-05:00    0.250000
Name: Dividends, Length: 86, dtype: float64
```

We can plot the dividends overtime:

In [23]:
```
apple.dividends.plot()
```

Out[23]:
```
<Axes: xlabel='Date'>
```



Exercise

Now using the `Ticker` module create an object for AMD (Advanced Micro Devices) with the ticker symbol is `AMD` called; name the object `amd`.

In [24]:
```
amd = yf.Ticker("AMD")
```

In [4]:
```
!wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeve
```

```
--2025-02-27 00:45:47--  https://cf-courses-data.s3.us.cloud-object-storage.appdo
main.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/data/amd.json
Resolving cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-courses-
data.s3.us.cloud-object-storage.appdomain.cloud)... 169.63.118.104
Connecting to cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-cour
ses-data.s3.us.cloud-object-storage.appdomain.cloud)|169.63.118.104|:443... conne
cted.
200 OKequest sent, awaiting response...
Length: 5838 (5.7K) [application/json]
Saving to: ‘amd.json’

amd.json            100%[===================>]   5.70K  --.-KB/s    in 0s

2025-02-27 00:45:47 (504 MB/s) - ‘amd.json’ saved [5838/5838]
```

In [5]:
```
import json
with open('amd.json') as json_file:
    amd_info = json.load(json_file)
    # Print the type of data variable
    #print("Type:", type(apple_info))
amd_info
```

Out[5]:
{'zip': '95054',
 'sector': 'Technology',
 'fullTimeEmployees': 15500,
 'longBusinessSummary': 'Advanced Micro Devices, Inc. operates as a semiconductor company worldwide. The company operates in two segments, Computing and Graphics; and Enterprise, Embedded and Semi-Custom. Its products include x86 microprocessors as an accelerated processing unit, chipsets, discrete and integrated graphics processing units (GPUs), data center and professional GPUs, and development services; and server and embedded processors, and semi-custom System-on-Chip (SoC) products, development services, and technology for game consoles. The company provides processors for desktop and notebook personal computers under the AMD Ryzen, AMD Ryzen PRO, Ryzen Threadripper, Ryzen Threadripper PRO, AMD Athlon, AMD Athlon PRO, AMD FX, AMD A-Series, and AMD PRO A-Series processors brands; discrete GPUs for desktop and notebook PCs under the AMD Radeon graphics, AMD Embedded Radeon graphics brands; and professional graphics products under the AMD Radeon Pro and AMD FirePro graphics brands. It also offers Radeon Instinct, Radeon PRO V-series, and AMD Instinct accelerators for servers; chipsets under the AMD trademark; microprocessors for servers under the AMD EPYC; embedded processor solutions under the AMD Athlon, AMD Geode, AMD Ryzen, AMD EPYC, AMD R-Series, and G-Series processors brands; and customer-specific solutions based on AMD CPU, GPU, and multi-media technologies, as well as semi-custom SoC products. It serves original equipment manufacturers, public cloud service providers, original design manufacturers, system integrators, independent distributors, online retailers, and add-in-board manufacturers through its direct sales force, independent distributors, and sales representatives. The company was incorporated in 1969 and is headquartered in Santa Clara, California.',
 'city': 'Santa Clara',
 'phone': '408 749 4000',
 'state': 'CA',
 'country': 'United States',
 'companyOfficers': [],
 'website': 'https://www.amd.com',
 'maxAge': 1,
 'address1': '2485 Augustine Drive',
 'industry': 'Semiconductors',
 'ebitdaMargins': 0.24674,
 'profitMargins': 0.19240999,
 'grossMargins': 0.48248002,
 'operatingCashflow': 3520999936,
 'revenueGrowth': 0.488,
 'operatingMargins': 0.22198,
 'ebitda': 4055000064,
 'targetLowPrice': 107,
 'recommendationKey': 'buy',
 'grossProfits': 7929000000,
 'freeCashflow': 3122749952,
 'targetMedianPrice': 150,
 'currentPrice': 119.22,
 'earningsGrowth': -0.454,
 'currentRatio': 2.024,
 'returnOnAssets': 0.21327,
 'numberOfAnalystOpinions': 38,
 'targetMeanPrice': 152.02,
 'debtToEquity': 9.764,
 'returnOnEquity': 0.47428,
 'targetHighPrice': 200,
 'totalCash': 3608000000,
 'totalDebt': 732000000,
 'totalRevenue': 16433999872,
 'totalCashPerShare': 3.008,

```
'financialCurrency': 'USD',
'revenuePerShare': 13.548,
'quickRatio': 1.49,
'recommendationMean': 2.2,
'exchange': 'NMS',
'shortName': 'Advanced Micro Devices, Inc.',
'longName': 'Advanced Micro Devices, Inc.',
'exchangeTimezoneName': 'America/New_York',
'exchangeTimezoneShortName': 'EDT',
'isEsgPopulated': False,
'gmtOffSetMilliseconds': '-14400000',
'quoteType': 'EQUITY',
'symbol': 'AMD',
'messageBoardId': 'finmb_168864',
'market': 'us_market',
'annualHoldingsTurnover': None,
'enterpriseToRevenue': 8.525,
'beta3Year': None,
'enterpriseToEbitda': 34.551,
'52WeekChange': 0.51966953,
'morningStarRiskRating': None,
'forwardEps': 4.72,
'revenueQuarterlyGrowth': None,
'sharesOutstanding': 1627360000,
'fundInceptionDate': None,
'annualReportExpenseRatio': None,
'totalAssets': None,
'bookValue': 6.211,
'sharesShort': 27776129,
'sharesPercentSharesOut': 0.0171,
'fundFamily': None,
'lastFiscalYearEnd': 1640390400,
'heldPercentInstitutions': 0.52896,
'netIncomeToCommon': 3161999872,
'trailingEps': 2.57,
'lastDividendValue': 0.005,
'SandP52WeekChange': 0.15217662,
'priceToBook': 19.194977,
'heldPercentInsiders': 0.00328,
'nextFiscalYearEnd': 1703462400,
'yield': None,
'mostRecentQuarter': 1640390400,
'shortRatio': 0.24,
'sharesShortPreviousMonthDate': 1644883200,
'floatShares': 1193798619,
'beta': 1.848425,
'enterpriseValue': 140104957952,
'priceHint': 2,
'threeYearAverageReturn': None,
'lastSplitDate': 966902400,
'lastSplitFactor': '2:1',
'legalType': None,
'lastDividendDate': 798940800,
'morningStarOverallRating': None,
'earningsQuarterlyGrowth': -0.453,
'priceToSalesTrailing12Months': 11.805638,
'dateShortInterest': 1647302400,
'pegRatio': 0.99,
'ytdReturn': None,
'forwardPE': 25.258476,
```

```
'lastCapGain': None,
'shortPercentOfFloat': 0.0171,
'sharesShortPriorMonth': 88709340,
'impliedSharesOutstanding': 0,
'category': None,
'fiveYearAverageReturn': None,
'previousClose': 123.23,
'regularMarketOpen': 123.04,
'twoHundredDayAverage': 116.6998,
'trailingAnnualDividendYield': 0,
'payoutRatio': 0,
'volume24Hr': None,
'regularMarketDayHigh': 125.66,
'navPrice': None,
'averageDailyVolume10Day': 102167370,
'regularMarketPreviousClose': 123.23,
'fiftyDayAverage': 115.95,
'trailingAnnualDividendRate': 0,
'open': 123.04,
'toCurrency': None,
'averageVolume10days': 102167370,
'expireDate': None,
'algorithm': None,
'dividendRate': None,
'exDividendDate': 798940800,
'circulatingSupply': None,
'startDate': None,
'regularMarketDayLow': 118.59,
'currency': 'USD',
'trailingPE': 46.389107,
'regularMarketVolume': 99476946,
'lastMarket': None,
'maxSupply': None,
'openInterest': None,
'marketCap': 194013855744,
'volumeAllCurrencies': None,
'strikePrice': None,
'averageVolume': 102428813,
'dayLow': 118.59,
'ask': 117.24,
'askSize': 1100,
'volume': 99476946,
'fiftyTwoWeekHigh': 164.46,
'fromCurrency': None,
'fiveYearAvgDividendYield': None,
'fiftyTwoWeekLow': 72.5,
'bid': 117.24,
'tradeable': False,
'dividendYield': None,
'bidSize': 900,
'dayHigh': 125.66,
'regularMarketPrice': 119.22,
'preMarketPrice': 116.98,
'logo_url': 'https://logo.clearbit.com/amd.com'}
```


Question 1 Use the key `'country'` to find the country the stock belongs to, remember it as it will be a quiz question.

In [6]:
```
country = amd_info.get('country', 'Not Available')
print("Country:", country)
```

```
Country: United States
```

Question 2 Use the key `'sector'` to find the sector the stock belongs to, remember it as it will be a quiz question.

In [7]:
```
sector = amd_info.get('sector', 'Not Available')
print("Sector:", sector)
```

```
Sector: Technology
```

Question 3 Obtain stock data for AMD using the `history` function, set the `period` to max. Find the `Volume` traded on the first day (first row).

In [25]:
```
amd_history = amd.history(period="max")

# Find the Volume traded on the first day (first row)
first_day_volume = amd_history.iloc[0]['Volume']
print("Volume Traded on First Day:", first_day_volume)
```

```
Volume Traded on First Day: 219600.0
```

About the Authors:

Joseph Santarcangelo has a PhD in Electrical Engineering, his research focused on using machine learning, signal processing, and computer vision to determine how videos impact human cognition. Joseph has been working for IBM since he completed his PhD.

Azim Hirjani

In [ ]:
```
```