Скачивание параллельными api запросами исторических цен за 1 мин и 15 мин с alphavantage.co и формирование аггрегированных данных по каждому тикеру

In [5]:
```
import csv
import requests
import numpy as np
import pandas as pd
import json
import os
import time
import threading
from tqdm import tqdm
from datetime import datetime
from requests.adapters import HTTPAdapter
from requests.packages.urllib3.util.retry import Retry
```

Формирование списка тикеров:

Необходимо для начала узнать какие акции произвели делистинг за последние два года и исключить из списка. IPO наоборот включается в список.

In [6]:
```
# delisted
CSV_URL = 'https://www.alphavantage.co/query?function=LISTING_STATUS&date=2021-07-31&state=delisted&apikey=F9IT

with requests.Session() as s:
    download = s.get(CSV_URL)
    decoded_content = download.content.decode('utf-8')
    cr = csv.reader(decoded_content.splitlines(), delimiter=',')
    delisted_stocks_list = list(cr)
```

In [7]:
```
my_stocks_delisted_list=[]
for i in np.arange(1,len(delisted_stocks_list)):
    delisting_date_string = datetime.fromisoformat(delisted_stocks_list[i][4])
    date_limit = datetime.fromisoformat('2019-07-30')
    if delisted_stocks_list[i][3]=='Stock' and delisting_date_string > date_limit:
        my_stocks_delisted_list.append([delisted_stocks_list[i][0], delisted_stocks_list[i][1], delisted_stocks

my_stocks_delisted_df = pd.DataFrame(my_stocks_delisted_list,columns=['symbol','name','ipoDate', 'delistingDate
my_stocks_delisted_df
```

Out[7]:

	symbol	name	ipoDate	delistingDate
0	AACQU	Origin Materials Inc - Units (1 Ord Share Clas...	2020-07-14	2021-06-24
1	AAN-W	Aarons Holdings Company Inc When Issued	2020-11-25	2020-11-30
2	ACACU	PLAYSTUDIOS Inc - Units (1 Ord Share Class A &...	2020-10-23	2021-06-21
3	ACEL-WS		2019-11-21	2020-07-15
4	ACND-U	Marketwise Inc - Units (1 Ord Cls A & 0.5 Red...	2020-07-24	2021-07-21
...	...	...	...	...
344	XBITV	XBiotech Inc	2020-02-14	2020-02-19
345	XL-WS	XL Fleet Corp Wt Exp 06012025	2020-12-22	2020-12-22
346	XL-WS	XL Fleet Corporation - Warrants (01/06/2025)	2019-09-03	2021-02-26
347	XPO-W	XPO Logistics Inc ExDistribution Whenissued	2021-07-23	2021-07-27
348	ZNOGW	Zion Oil & Gas Inc - Warrants (31/01/2023)	2020-09-02	2020-11-05

349 rows × 4 columns

In [8]:
```
my_stocks_delisted_df.ipoDate.min(), my_stocks_delisted_df.ipoDate.max()
```

Out[8]:
```
('2019-08-01', '2021-07-29')
```

In [14]:
```
my_stocks_delisted_list_symbols = [delisted_stock[0] for delisted_stock in my_stocks_delisted_list]
my_stocks_delisted_list_symbols[:5]
```

Out[14]:
```
['AACQU', 'AAN-W', 'ACACU', 'ACEL-WS', 'ACND-U']
```

In [10]:
```
with open("my_stocks_delisted_list_symbols.txt", "w") as fp:
    json.dump(my_stocks_delisted_list_symbols, fp)
```

В качестве образца для рабочих дат применим даты цен общего рынка:

In [12]:
```
url_SPY = 'https://www.alphavantage.co/query?function=TIME_SERIES_DAILY_ADJUSTED&symbol=SPY&outputsize=full&ap
r_SPY = requests.get(url_SPY)
data_SPY = r_SPY.json()
list_SPY = list(data_SPY['Time Series (Daily)'].keys())
dates = list_SPY[:list_SPY.index('2019-07-31')+1]
dates[:5]
```

Out[12]:
```
['2021-10-19', '2021-10-18', '2021-10-15', '2021-10-14', '2021-10-13']
```

In [ ]:
```
my_stocks_list_per_date =[]
for date in dates:
    CSV_URL = 'https://www.alphavantage.co/query?function=LISTING_STATUS&date=' + date + '&state=active&apikey=
    with requests.Session() as s:
        download = s.get(CSV_URL)
        decoded_content = download.content.decode('utf-8')
        cr = csv.reader(decoded_content.splitlines(), delimiter=',')
        my_list = list(cr)
        my_stocks_list=[]
        for ticker in np.arange(1,len(my_list)):
            if my_list[ticker][3]=='Stock' and my_list[ticker][0] not in my_stocks_delisted_list_symbols:
                my_stocks_list.append(my_list[ticker])
    my_stocks_list_per_date.append(my_stocks_list)
```

In [ ]:
```
with open("tickers_list_total.txt", "w") as fp:
    json.dump(my_stocks_list_per_date, fp)
```

Скачивание по сформированному списку тикеров:

In [2]:
```
with open("tickers_list_total.txt", "r") as fp:
    tickers_list_total = json.load(fp)
len(tickers_list_total[0])
```

Out[2]:
```
8248
```

In [3]:
```
all_tickers = tickers_list_total[0]
```

In [4]:
```
periods = ['yearlmonth1',
'yearlmonth2',
'yearlmonth3',
'yearlmonth4',
'yearlmonth5',
'yearlmonth6',
'yearlmonth7',
'yearlmonth8',
'yearlmonth9',
'yearlmonth10',
'yearlmonth11',
'yearlmonth12',
'year2month1',
'year2month2',
'year2month3',
'year2month4',
'year2month5',
'year2month6',
'year2month7',
'year2month8',
'year2month9',
'year2month10',
'year2month11',
'year2month12']
```

Для 1 min исторических данных за последние два года:

In [ ]:
```
n = 150
new_list = [all_tickers[i:i + n] for i in range(0, len(all_tickers), n)]
```

In [ ]:
```
for ticker in tickers_list_total[0]:
    directory = ticker
    parent_dir = 'C:/Users/Kuanysh/Downloads/pump_and_dump/all_tickers_1m'
    path = os.path.join(parent_dir, directory)
    os.makedirs(path)
```

In [ ]:
```
def get_ticker_series(ticker, period):
    CSV_URL = 'https://www.alphavantage.co/query?function=TIME_SERIES_INTRADAY_EXTENDED&symbol=' + ticker + '&i
    with requests.Session() as s:
        retry = Retry(connect=3, backoff_factor=0.5)
        adapter = HTTPAdapter(max_retries=retry)
        s.mount('http://', adapter)
        s.mount('https://', adapter)
        download = s.get(CSV_URL)
        decoded_content = download.content.decode('utf-8')
        cr = csv.reader(decoded_content.splitlines(), delimiter=',')
        ticker_quotes = list(cr)

    df = pd.DataFrame(ticker_quotes)
    header_row=0
    df.columns = df.iloc[header_row]
    df = df.drop(header_row)
    df.set_index('time', inplace=True)

    directory = ticker
    parent_dir = 'C:/Users/Kuanysh/Downloads/pump_and_dump/all_tickers_1m'
    path = os.path.join(parent_dir, directory)
    df.to_csv(os.path.join(path, ticker  + '_' + period + '.csv'))

def get_tickers_parallel(tickers, period):
    threads = list()
    for ticker in tickers:
        ticker_thread = threading.Thread(target=get_ticker_series, args=(ticker, period))
        threads.append(ticker_thread)
        ticker_thread.start()
    for tick_thread in threads:
        tick_thread.join()
```

In [ ]:
```
limit = 7.7
for p in tqdm(range(len(periods))):
    for t in tqdm(range(len(new_list))):
        start_time = time.perf_counter()
        get_tickers_parallel(new_list[t], periods[p])
        end_time = time.perf_counter()
        execution_time = end_time - start_time
        if execution_time < limit:
            delay = limit - execution_time
            print ('delay',delay)
            time.sleep(delay)
```

In [ ]:
```
for i in tqdm(range(len(all_tickers))):
    directory = all_tickers[i]
    parent_dir = 'C:/Users/Kuanysh/Downloads/pump_and_dump/all_tickers_1m'
    path = os.path.join(parent_dir, directory)
    files = glob.glob(path + '/*')
    df = pd.concat(map(pd.read_csv, files), ignore_index=True)
    df = df.drop_duplicates()
    df = df.sort_values('time',ascending=False)
    df = df.set_index('time', inplace=False)
    next_dir = 'C:/Users/Kuanysh/Downloads/pump_and_dump/agg_tickers_1m'
    df.to_csv(os.path.join(next_dir, all_tickers[i] + '.csv'))
```

Для 15 min исторических данных за последние два года:

In [ ]:
```
n = 24
new_list = [all_tickers[i:i + n] for i in range(0, len(all_tickers), n)]
```

In [ ]:
```
for ticker in tickers_list_total[0]:
    directory = ticker
    parent_dir = 'C:/Users/Kuanysh/Downloads/pump_and_dump/all_tickers_15m'
    path = os.path.join(parent_dir, directory)
    os.makedirs(path)
```

In [ ]:
```
def get_ticker_series(ticker, period):
    CSV_URL = 'https://www.alphavantage.co/query?function=TIME_SERIES_INTRADAY_EXTENDED&symbol=' + ticker + '&i
    with requests.Session() as s:
        retry = Retry(connect=3, backoff_factor=0.5)
        adapter = HTTPAdapter(max_retries=retry)
        s.mount('http://', adapter)
        s.mount('https://', adapter)
        download = s.get(CSV_URL)
        decoded_content = download.content.decode('utf-8')
        cr = csv.reader(decoded_content.splitlines(), delimiter=',')
        ticker_quotes = list(cr)

    df = pd.DataFrame(ticker_quotes)
    header_row=0
    df.columns = df.iloc[header_row]
    df = df.drop(header_row)
    df.set_index('time', inplace=True)

    directory = ticker
    parent_dir = 'C:/Users/Kuanysh/Downloads/pump_and_dump/all_tickers_15m'
    path = os.path.join(parent_dir, directory)
    df.to_csv(os.path.join(path, ticker  + '_' + period + '.csv'))

def get_tickers_parallel(tickers, period):
    threads = list()
    for ticker in tickers:
        ticker_thread = threading.Thread(target=get_ticker_series, args=(ticker, period))
        threads.append(ticker_thread)
        ticker_thread.start()
    for tick_thread in threads:
        tick_thread.join()
```

In [ ]:
```
limit = 6
for p in tqdm(range(len(periods))):
    for t in tqdm(range(len(new_list))):
        start_time = time.perf_counter()
        get_tickers_parallel(new_list[t], periods[p])
        end_time = time.perf_counter()
        execution_time = end_time - start_time
        if execution_time < limit:
            delay = limit - execution_time
            print ('delay',delay)
            time.sleep(delay)
```

In [ ]:
```
for i in tqdm(range(len(all_tickers))):
    directory = all_tickers[i]
    parent_dir = 'C:/Users/Kuanysh/Downloads/pump_and_dump/all_tickers_15m'
    path = os.path.join(parent_dir, directory)
    files = glob.glob(path + '/*')
    df = pd.concat(map(pd.read_csv, files), ignore_index=True)
    df = df.drop_duplicates()
    df = df.sort_values('time',ascending=False)
    df = df.set_index('time', inplace=False)
    next_dir = 'C:/Users/Kuanysh/Downloads/pump_and_dump/agg_tickers_15m'
    df.to_csv(os.path.join(next_dir, all_tickers[i] + '.csv'))
```