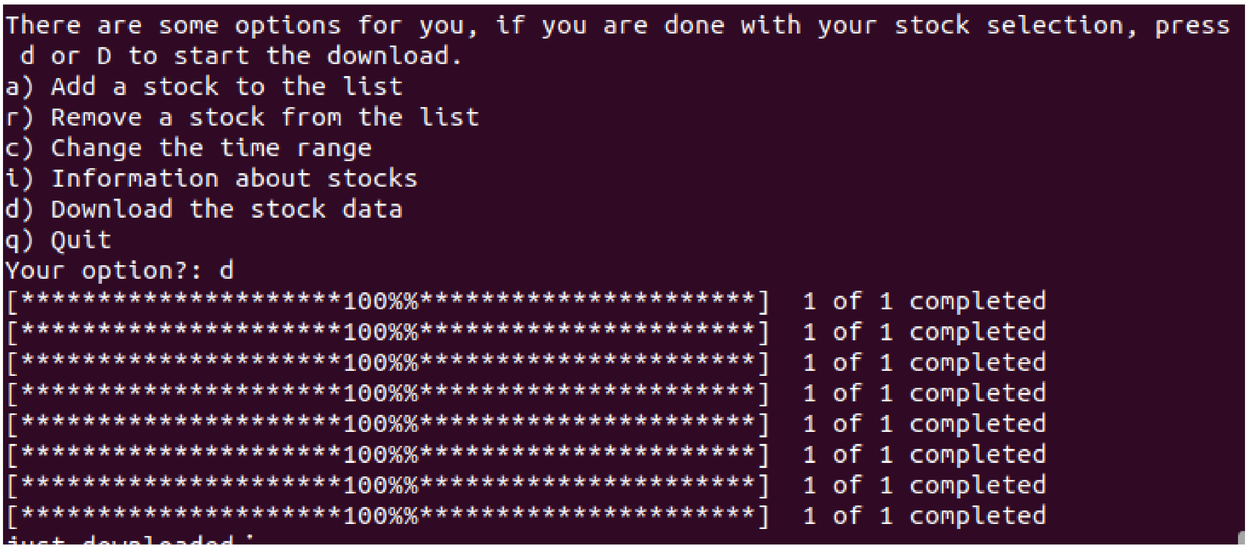
DSCI560-project

Python Environment: python = "^3.10" pandas = "2.0.3" duckdb = "^0.8.1" yfinance = "0.2.28"

Pulling the Data

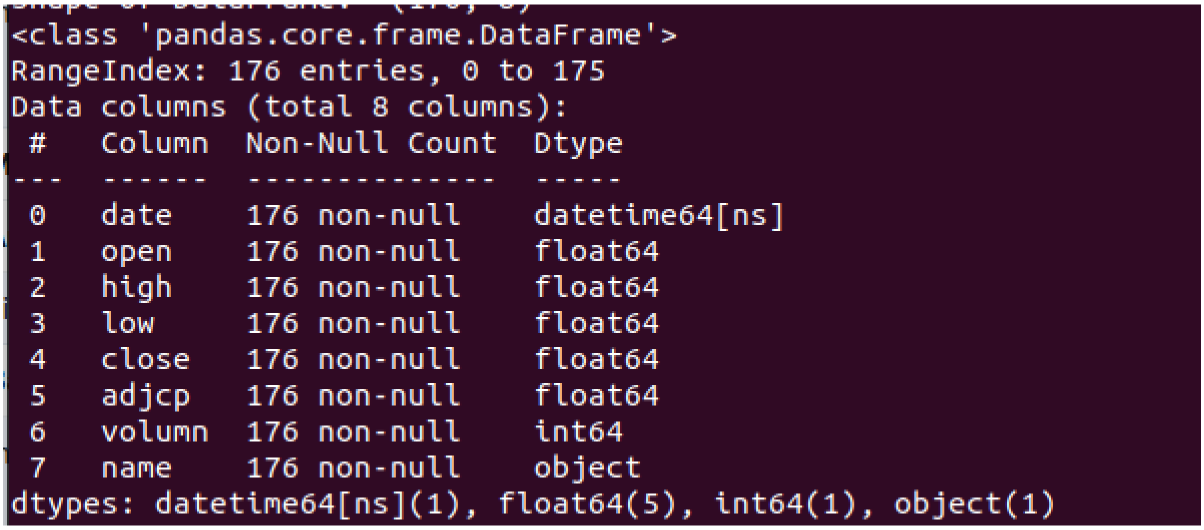
1. Go to the script storage location and enter "python3 fetch_data.py"
2. Press D or d to download what you need.



```
There are some options for you, if you are done with your stock selection, press
 d or D to start the download.
a) Add a stock to the list
r) Remove a stock from the list
c) Change the time range
i) Information about stocks
d) Download the stock data
q) Quit
Your option?: d
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
```

3. Or follow the prompts and press other buttons to add, delete or display the stocks or dates you selected.

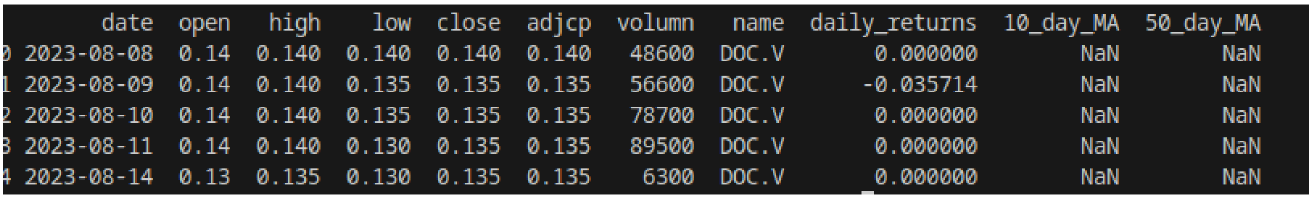
Data Structure



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 176 entries, 0 to 175
Data columns (total 8 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   date    176 non-null    datetime64[ns]
 1   open    176 non-null    float64
 2   high    176 non-null    float64
 3   low     176 non-null    float64
 4   close   176 non-null    float64
 5   adjcp   176 non-null    float64
 6   volumn  176 non-null    int64
 7   name    176 non-null    object
dtypes: datetime64[ns](1), float64(5), int64(1), object(1)
```

Preprocessed Stock Price Data

To Preprocess the stock price data please execute file : data_preprocessing.py



```
        date  open   high    low  close  adjcp  volumn    name  daily_returns  10_day_MA  50_day_MA
0 2023-08-08  0.14  0.140  0.140  0.140  0.140   48600  DOC.V       0.000000        NaN        NaN
1 2023-08-09  0.14  0.140  0.135  0.135  0.135   56600  DOC.V      -0.035714        NaN        NaN
2 2023-08-10  0.14  0.140  0.135  0.135  0.135   78700  DOC.V       0.000000        NaN        NaN
3 2023-08-11  0.14  0.140  0.130  0.135  0.135   89500  DOC.V       0.000000        NaN        NaN
4 2023-08-14  0.13  0.135  0.130  0.135  0.135    6300  DOC.V       0.000000        NaN        NaN
```

1. Handle all the missing values using 3 measures :
    1. Forward Filling
    2. Backward Filling
    3. Interpolation
2. Converting all the date attributes into datetime format.
3. Relevant Metrics Daily Return(Attributes) : It calculates the daily returns by taking the percentage change between each day's closing price and the closing price of the previous day. This is a common calculation used in financial analysis to measure the daily performance of a stock.

    The resulting 'daily_returns' column will contain the daily returns as decimal values, where a positive value indicates a gain (increase in price), and a negative value indicates a loss (decrease in price) compared to the previous day's closing price.
4. Moving Averages: Compute various moving averages (e.g., 10-day, 50-day, 200-day moving averages) to identify trends.

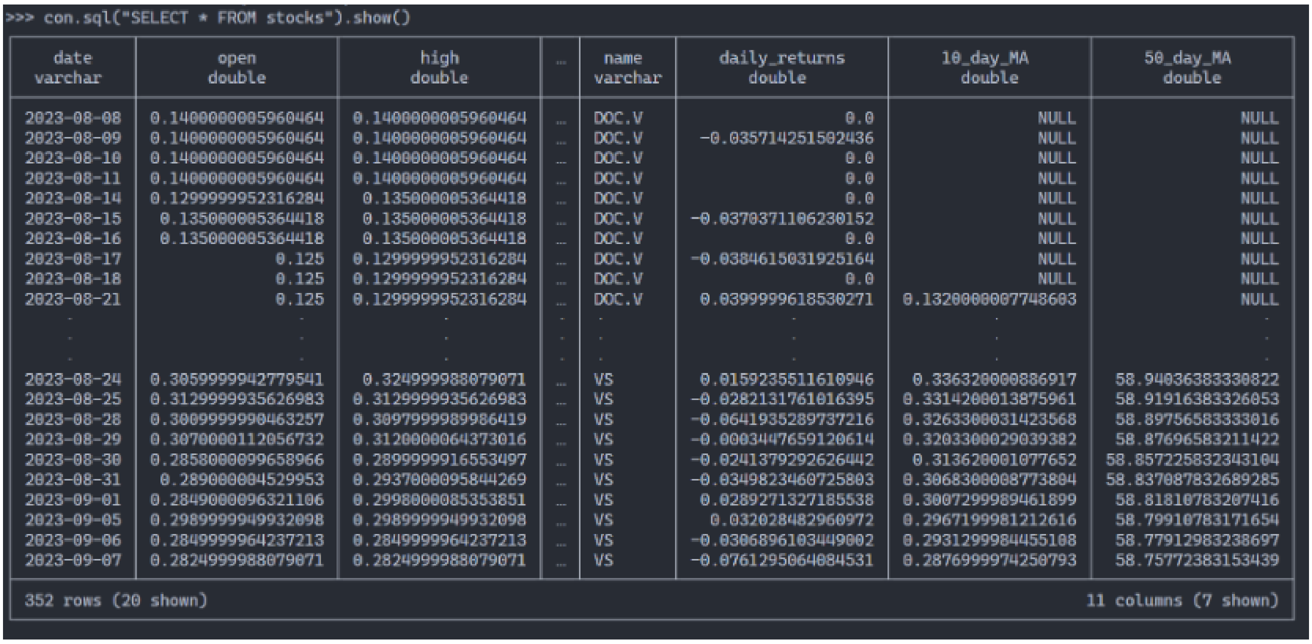
Storing the data

We decided to use duckdb(SQL OLAP) over mysql as it runs in-process, is optimized for analysis, and made for applications.Part 2 of this assignment might have other conditions so we may need to change this DB later.

1. The script that imports the CSV into our duckdb database is "to_db.py". Quick note for replication, must use pandas 2.0.3 as there is currently a bug when working with pandas and duckdb.

Table Output:

> con.sql("SELECT * FROM stocks").show()



```
>>> con.sql("SELECT * FROM stocks").show()
┌────────────┬────────────────────┬────────────────────┬───┬─────────┬─────────────────────┬────────────────────┬────────────────────┐
│    date    │        open        │        high        │ … │  name   │    daily_returns    │     10_day_MA      │     50_day_MA      │
│  varchar   │       double       │       double       │   │ varchar │       double        │       double       │       double       │
├────────────┼────────────────────┼────────────────────┼───┼─────────┼─────────────────────┼────────────────────┼────────────────────┤
│ 2023-08-08 │ 0.1400000005960464 │ 0.1400000005960464 │ … │ DOC.V   │                 0.0 │               NULL │               NULL │
│ 2023-08-09 │ 0.1400000005960464 │ 0.1400000005960464 │ … │ DOC.V   │  -0.035714251502436 │               NULL │               NULL │
│ 2023-08-10 │ 0.1400000005960464 │ 0.1400000005960464 │ … │ DOC.V   │                 0.0 │               NULL │               NULL │
│ 2023-08-11 │ 0.1400000005960464 │ 0.1400000005960464 │ … │ DOC.V   │                 0.0 │               NULL │               NULL │
│ 2023-08-14 │ 0.1299999952316284 │  0.135000005364418 │ … │ DOC.V   │                 0.0 │               NULL │               NULL │
│ 2023-08-15 │  0.135000005364418 │  0.135000005364418 │ … │ DOC.V   │ -0.0370371106230152 │               NULL │               NULL │
│ 2023-08-16 │  0.135000005364418 │  0.135000005364418 │ … │ DOC.V   │                 0.0 │               NULL │               NULL │
│ 2023-08-17 │              0.125 │ 0.1299999952316284 │ … │ DOC.V   │ -0.0384615031925164 │               NULL │               NULL │
│ 2023-08-18 │              0.125 │ 0.1299999952316284 │ … │ DOC.V   │                 0.0 │               NULL │               NULL │
│ 2023-08-21 │              0.125 │ 0.1299999952316284 │ … │ DOC.V   │  0.0399999618530271 │ 0.1320000007748603 │               NULL │
│     ·      │                  · │                  · │ · │ ·       │                   · │                  · │                  · │
│     ·      │                  · │                  · │ · │ ·       │                   · │                  · │                  · │
│     ·      │                  · │                  · │ · │ ·       │                   · │                  · │                  · │
│ 2023-08-24 │ 0.3059999942779541 │  0.324999988079071 │ … │ VS      │  0.0159235511610946 │  0.336320000886917 │  58.94036383330822 │
│ 2023-08-25 │ 0.3129999935626983 │ 0.3129999935626983 │ … │ VS      │ -0.0282131761016395 │ 0.3314200013875961 │  58.91916383326053 │
│ 2023-08-28 │ 0.3009999990463257 │ 0.3097999989986419 │ … │ VS      │ -0.0641935289737216 │ 0.3263300031423568 │  58.89756583333016 │
│ 2023-08-29 │ 0.3070000112056732 │ 0.3120000064373016 │ … │ VS      │ -0.0003447659120614 │ 0.3203300029039382 │  58.87696583211422 │
│ 2023-08-30 │ 0.2858000099658966 │ 0.2899999916553497 │ … │ VS      │ -0.0241379292626442 │  0.313620001077652 │ 58.857225832343104 │
│ 2023-08-31 │  0.289000004529953 │ 0.2937000095844269 │ … │ VS      │ -0.0349823460725803 │ 0.3068300008773804 │ 58.837087832689285 │
│ 2023-09-01 │ 0.2849000096321106 │ 0.2998000085353851 │ … │ VS      │  0.0289271327185538 │ 0.3007299989461899 │  58.81810783207416 │
│ 2023-09-05 │ 0.2989999949932098 │ 0.2989999949932098 │ … │ VS      │   0.032028482960972 │ 0.2967199981212616 │  58.79910783171654 │
│ 2023-09-06 │ 0.2849999964237213 │ 0.2849999964237213 │ … │ VS      │ -0.0306896103449002 │ 0.2931299984455108 │  58.77912983238697 │
│ 2023-09-07 │ 0.2824999988079071 │ 0.2824999988079071 │ … │ VS      │ -0.0761295064084531 │ 0.2876999974250793 │  58.75772383153439 │
├────────────┴────────────────────┴────────────────────┴───┴─────────┴─────────────────────┴────────────────────┴────────────────────┤
│ 352 rows (20 shown)                                                                                           11 columns (7 shown) │
└────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────┘
```