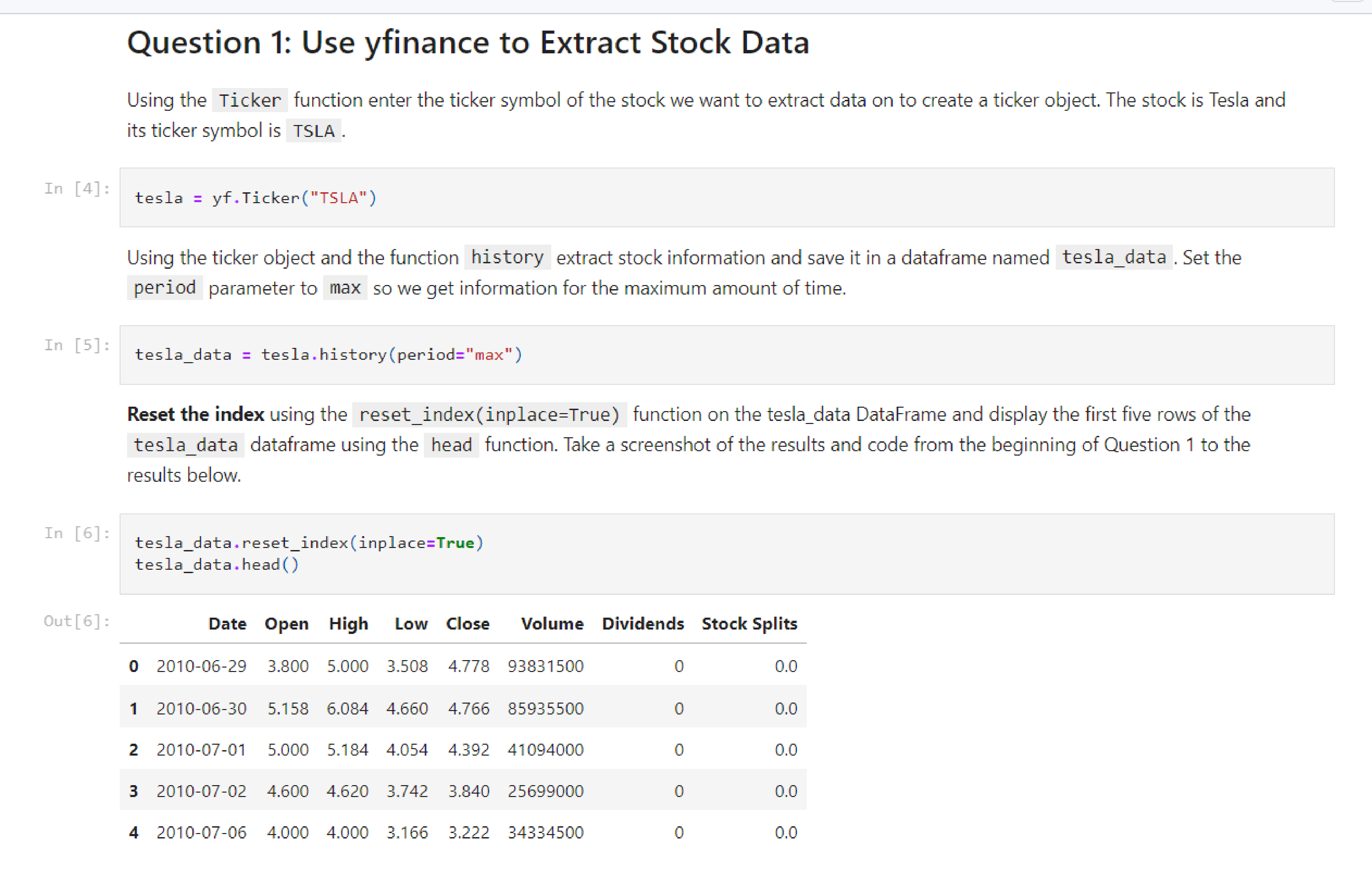


Question 1: Use yfinance to Extract Stock Data

Using the `Ticker` function enter the ticker symbol of the stock we want to extract data on to create a ticker object. The stock is Tesla and its ticker symbol is `TSLA`.

In [4]:

```
tesla = yf.Ticker("TSLA")
```

Using the ticker object and the function `history` extract stock information and save it in a dataframe named `tesla_data`. Set the `period` parameter to `max` so we get information for the maximum amount of time.

In [5]:

```
tesla_data = tesla.history(period="max")
```

Reset the index using the `reset_index(inplace=True)` function on the tesla_data DataFrame and display the first five rows of the `tesla_data` dataframe using the `head` function. Take a screenshot of the results and code from the beginning of Question 1 to the results below.

In [6]:

```
tesla_data.reset_index(inplace=True)
tesla_data.head()
```

Out[6]:

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	2010-06-29	3.800	5.000	3.508	4.778	93831500	0	0.0
1	2010-06-30	5.158	6.084	4.660	4.766	85935500	0	0.0
2	2010-07-01	5.000	5.184	4.054	4.392	41094000	0	0.0
3	2010-07-02	4.600	4.620	3.742	3.840	25699000	0	0.0
4	2010-07-06	4.000	4.000	3.166	3.222	34334500	0	0.0

Click here if you need help removing the dollar sign and comma

```
If you parsed the HTML table by row and column you can use the replace function on the string

    revenue = col[1].text.replace("$", "").replace(",", "")

If you use the read_html function you can use the replace function on the string representation of the
column

    tesla_revenue["Revenue"] = tesla_revenue["Revenue"].str.replace("$", "").str.replace(",", "")
```

Remove the rows in the dataframe that are empty strings or are NaN in the Revenue column. Print the entire `tesla_revenue` DataFrame to see if you have any.

In [10]:

```
tesla_revenue
```

Out[10]:

	Date	Revenue
0	2020-12-31	10744
1	2020-09-30	8771
2	2020-06-30	6036
3	2020-03-31	5985
4	2019-12-31	7384
5	2019-09-30	6303
6	2019-06-30	6350
7	2019-03-31	4541

33	2012-09-30	50
34	2012-06-30	27
35	2012-03-31	30
36	2011-12-31	39
37	2011-09-30	58
38	2011-06-30	58
39	2011-03-31	49
40	2010-12-31	36
41	2010-09-30	31
42	2010-06-30	28
43	2010-03-31	21
44	2009-12-31	NaN
45	2009-09-30	46
46	2009-06-30	27
47	2008-12-31	NaN

Click here if you need help removing the Nan or empty strings

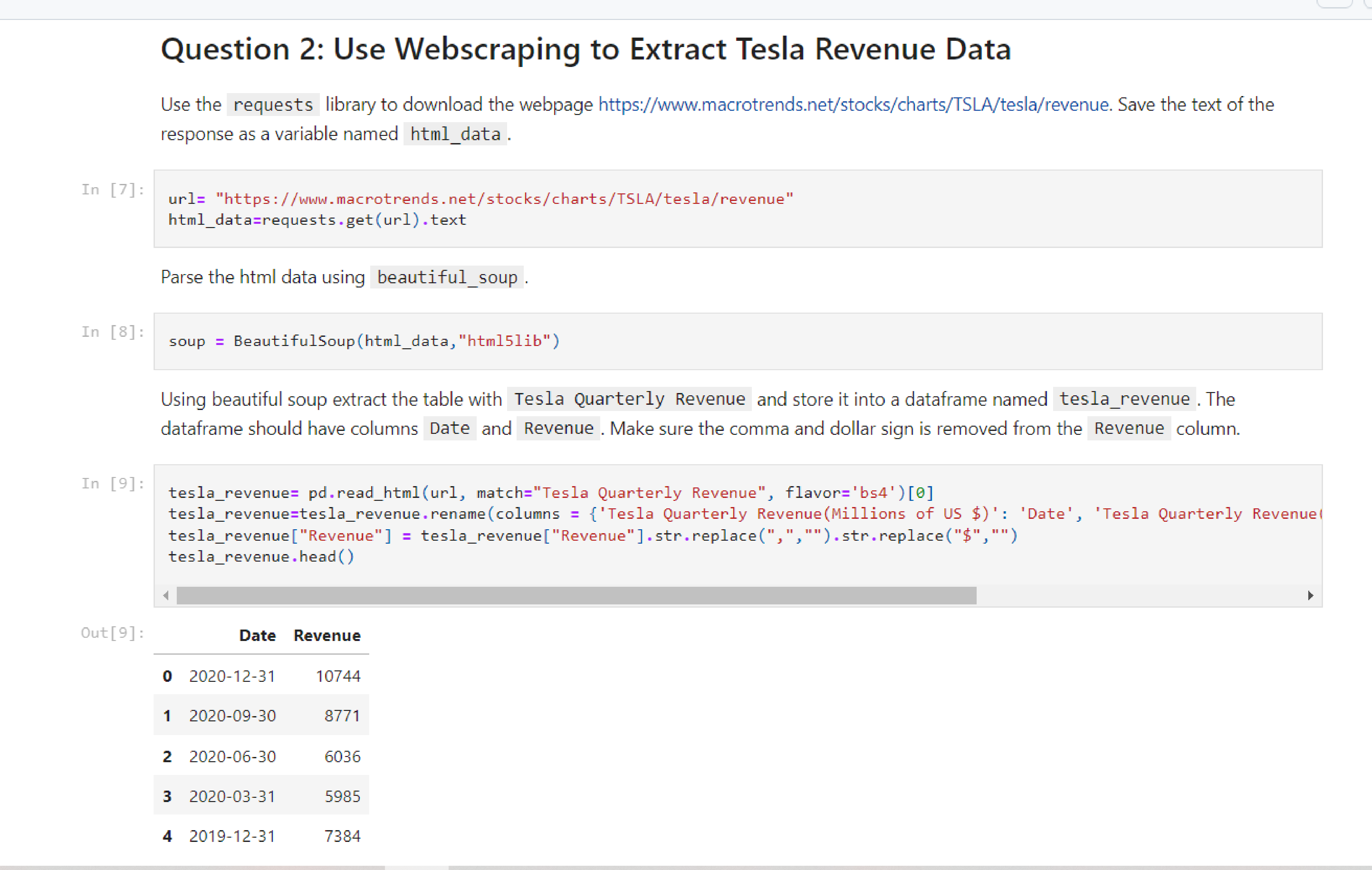
```
If you have NaN in the Revenue column

    tesla_revenue.dropna(inplace=True)

If you have emtpty string in the Revenue column
```

```
tesla_revenue = tesla_revenue[tesla_revenue['Revenue'] != ""]
```

Display the last 5 row of the `tesla_revenue` dataframe using the `tail` function. Take a screenshot of the results.

In [11]:
```
tesla_revenue.dropna(inplace=True)
tesla_revenue.tail()
```

Out[11]:
```
          Date  Revenue
41  2010-09-30       31
42  2010-06-30       28
43  2010-03-31       21
45  2009-09-30       46
46  2009-06-30       27
```



Question 2: Use Webscraping to Extract Tesla Revenue Data

Use the `requests` library to download the webpage https://www.macrotrends.net/stocks/charts/TSLA/tesla/revenue. Save the text of the response as a variable named `html_data`.

In [7]:
```
url= "https://www.macrotrends.net/stocks/charts/TSLA/tesla/revenue"
html_data=requests.get(url).text
```

Parse the html data using `beautiful_soup`.

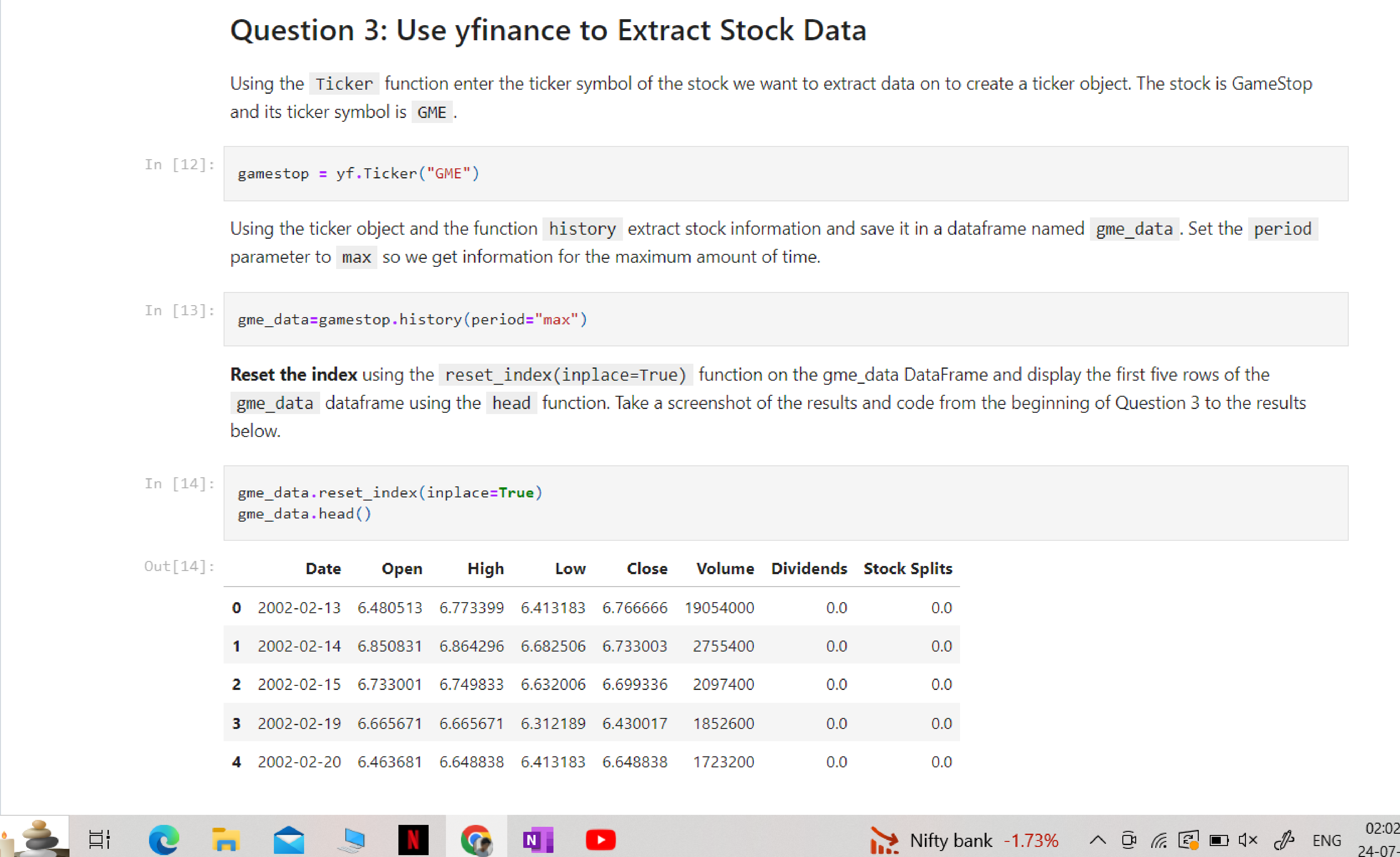
In [8]:
```
soup = BeautifulSoup(html_data,"html5lib")
```

Using beautiful soup extract the table with `Tesla Quarterly Revenue` and store it into a dataframe named `tesla_revenue`. The dataframe should have columns `Date` and `Revenue`. Make sure the comma and dollar sign is removed from the `Revenue` column.

In [9]:
```
tesla_revenue= pd.read_html(url, match="Tesla Quarterly Revenue", flavor='bs4')[0]
tesla_revenue=tesla_revenue.rename(columns = {'Tesla Quarterly Revenue(Millions of US $)': 'Date', 'Tesla Quarterly Revenue(
tesla_revenue["Revenue"] = tesla_revenue["Revenue"].str.replace(",","").str.replace("$","")
tesla_revenue.head()
```

Out[9]:

	Date	Revenue
0	2020-12-31	10744
1	2020-09-30	8771
2	2020-06-30	6036
3	2020-03-31	5985
4	2019-12-31	7384



Question 3: Use yfinance to Extract Stock Data

Using the `Ticker` function enter the ticker symbol of the stock we want to extract data on to create a ticker object. The stock is GameStop and its ticker symbol is `GME`.

In [12]:
```
gamestop = yf.Ticker("GME")
```

Using the ticker object and the function `history` extract stock information and save it in a dataframe named `gme_data`. Set the `period` parameter to `max` so we get information for the maximum amount of time.

In [13]:
```
gme_data=gamestop.history(period="max")
```

Reset the index using the `reset_index(inplace=True)` function on the gme_data DataFrame and display the first five rows of the `gme_data` dataframe using the `head` function. Take a screenshot of the results and code from the beginning of Question 3 to the results below.

In [14]:
```
gme_data.reset_index(inplace=True)
gme_data.head()
```

Out[14]:

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	2002-02-13	6.480513	6.773399	6.413183	6.766666	19054000	0.0	0.0
1	2002-02-14	6.850831	6.864296	6.682506	6.733003	2755400	0.0	0.0
2	2002-02-15	6.733001	6.749833	6.632006	6.699336	2097400	0.0	0.0
3	2002-02-19	6.665671	6.665671	6.312189	6.430017	1852600	0.0	0.0
4	2002-02-20	6.463681	6.648838	6.413183	6.648838	1723200	0.0	0.0

Display the last five rows of the `gme_revenue` dataframe using the `tail` function. Take a screenshot of the results.

In [18]:
```
gme_revenue.dropna(inplace=True)
gme_revenue.tail()
```

Out[18]:
```
          Date  Revenue
59  2006-01-31     1667
60  2005-10-31      534
61  2005-07-31      416
62  2005-04-30      475
63  2005-01-31      709
```

Question 4: Use Webscraping to Extract GME Revenue Data

Use the `requests` library to download the webpage https://www.macrotrends.net/stocks/charts/GME/gamestop/revenue. Save the text of the response as a variable named `html_data`.

In [15]:
```
url="https://www.macrotrends.net/stocks/charts/GME/gamestop/revenue"
html_data=requests.get(url).text
```

Parse the html data using `beautiful_soup`.

In [16]:
```
soup = BeautifulSoup(html_data,"html5lib")
```

Using beautiful soup extract the table with `GameStop Quarterly Revenue` and store it into a dataframe named `gme_revenue`. The dataframe should have columns `Date` and `Revenue`. Make sure the comma and dollar sign is removed from the `Revenue` column using a method similar to what you did in Question 2.

In [17]:
```
gme_revenue= pd.read_html(url, match="GameStop Quarterly Revenue", flavor='bs4')[0]
gme_revenue=gme_revenue.rename(columns = {'GameStop Quarterly Revenue(Millions of US $)': 'Date', 'GameStop Quarterly Revenu
gme_revenue["Revenue"] = gme_revenue["Revenue"].str.replace(",","").str.replace("$","")
```

Display the last five rows of the `gme_revenue` dataframe using the `tail` function. Take a screenshot of the results.

In [18]:
```
gme_revenue.dropna(inplace=True)
gme_revenue.tail()
```

Out[18]:

	Date	Revenue
59	2006-01-31	1667
60	2005-10-31	534

Question 5: Plot Tesla Stock Graph

Use the `make_graph` function to graph the Tesla Stock Data, also provide a title for the graph. The structure to call the `make_graph` function is `make_graph(tesla_data, tesla_revenue, 'Tesla')`

In [19]:

```
make_graph(tesla_data, tesla_revenue, 'Tesla Stock Data Graph')
```

Question 6: Plot GameStop Stock Graph

Use the `make_graph` function to graph the GameStop Stock Data, also provide a title for the graph. The structure to call the `make_graph` function is `make_graph(gme_data, gme_revenue, 'GameStop')`.

In [64]:

```
make_graph(gme_data, gme_revenue, 'GameStop Stock Data Graph')
```