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`.

[13]:
```
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.

[14]:
```
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.

[15]:
```
tesla_data.reset_index(inplace=True)
tesla_data.head()
```

[15]:

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	2010-06-29 00:00:00-04:00	1.266667	1.666667	1.169333	1.592667	281494500	0.0	0.0
1	2010-06-30 00:00:00-04:00	1.719333	2.028000	1.553333	1.588667	257806500	0.0	0.0
2	2010-07-01 00:00:00-04:00	1.666667	1.728000	1.351333	1.464000	123282000	0.0	0.0
3	2010-07-02 00:00:00-04:00	1.533333	1.540000	1.247333	1.280000	77097000	0.0	0.0
4	2010-07-06 00:00:00-04:00	1.333333	1.333333	1.055333	1.074000	103003500	0.0	0.0

Question 2: Use Webscraping to Extract Tesla Revenue Data

Use the `requests` library to download the webpage https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/revenue.htm Save the text of the response as a variable named `html_data`.

[143]:
```
url="https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/r
html_data=requests.get(url).text
```

Parse the html data using `beautiful_soup` using parser i.e `html5lib` or `html.parser`.

[144]:
```
soup=BeautifulSoup(html_data,'html5lib')
```

Using `BeautifulSoup` or the `read_html` function extract the table with `Tesla Revenue` and store it into a dataframe named `tesla_revenue`. The dataframe should have columns `Date` and `Revenue`.

[146]:
```

# Initialize an empty DataFrame
tesla_revenue = pd.DataFrame(columns=["Date", "Revenue"])

# Find all rows in the tbody of the table
for row in soup.find("tbody").find_all('tr'):
    col = row.find_all("td")
    if len(col) > 1:  # Ensure there are at least two columns
        date = col[0].text.strip()  # Clean up the date string (e.g., remove leading/trailing spaces)

        # Clean up the revenue string: remove commas and dollar signs
        revenue = col[1].text.strip().replace(",", "").replace("$", "")

        # Convert the cleaned-up revenue to a float
        try:
            revenue = int(revenue)
        except ValueError:
            revenue = None  # Handle cases where revenue cannot be converted to float

        # Append the row using pd.concat for better performance
        new_row = pd.DataFrame({"Date": [date], "Revenue": [revenue]})
        tesla_revenue = pd.concat([tesla_revenue, new_row], ignore_index=True)
```

Execute the following line to remove the comma and dollar sign from the `Revenue` column.

[134]:
```
tesla_revenue["Revenue"] = tesla_revenue['Revenue'].str.replace(',|\$',"")
```

Execute the following lines to remove an null or empty strings in the Revenue column.

[147]:
```
tesla_revenue.dropna(inplace=True)

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.

[148]:
```
tesla_revenue.tail()
```

[148]:

	Date	Revenue
8	2013	2013
9	2012	413
10	2011	204
11	2010	117
12	2009	112

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`.

[107]:
```python
game = 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.

[108]:
```python
gme_data = game.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.

[109]:
```python
gme_data.reset_index(inplace=True)
gme_data.head()
```

[109]:

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	2002-02-13 00:00:00-05:00	1.620129	1.693350	1.603296	1.691667	76216000	0.0	0.0
1	2002-02-14 00:00:00-05:00	1.712707	1.716074	1.670626	1.683251	11021600	0.0	0.0
2	2002-02-15 00:00:00-05:00	1.683250	1.687458	1.658002	1.674834	8389600	0.0	0.0
3	2002-02-19 00:00:00-05:00	1.666418	1.666418	1.578047	1.607504	7410400	0.0	0.0
4	2002-02-20 00:00:00-05:00	1.615920	1.662210	1.603296	1.662210	6892800	0.0	0.0

Question 4: Use Webscraping to Extract GME Revenue Data

Use the `requests` library to download the webpage https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/stock.html. Save the text of the response as a variable named `html_data_2`.

[116]:
```python
url="https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/stock.html"
html_data_2=requests.get(url).text
```

Parse the html data using `beautiful_soup` using parser i.e `html5lib` or `html.parser`.

[117]:
```python
soup=BeautifulSoup(html_data_2,'html.parser')
```

Using `BeautifulSoup` or the `read_html` function extract the table with `GameStop 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.

> **Note: Use the method similar to what you did in question 2.**

▶ Click here if you need help locating the table

[150]:
```python
gme_revenue = pd.DataFrame(columns=["Date", "Revenue"])

# Find the table containing GameStop revenue
table = soup.find("table")  # Find the first <table> tag or specify an id/class if necessary
tbody = table.find("tbody")  # Assuming the data is within a <tbody> tag

# Loop through each row in the table
for row in tbody.find_all('tr'):
    col = row.find_all('td')

    if len(col) > 1:  # Make sure there are at least two columns (Date, Revenue)
        # Extract Date and Revenue
        date = col[0].text.strip()
        revenue = col[1].text.strip()

        # Clean the Revenue string: remove dollar sign and commas, then convert to a numeric value
        revenue = revenue.replace('$', '').replace(',', '')

        # Store the data into the DataFrame using pd.concat for better performance
        new_row = pd.DataFrame({"Date": [date], "Revenue": [revenue]})
        gme_revenue = pd.concat([gme_revenue, new_row], ignore_index=True)

# Optional: Convert the 'Revenue' column to numeric type (float or int)
gme_revenue['Revenue'] = pd.to_numeric(gme_revenue['Revenue'], errors='coerce')
```

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

[119]:
```python
gme_revenue.tail()
```

[119]:

	Date	Revenue
11	2009	8806
12	2008	7094
13	2007	5319
14	2006	3092
15	2005	1843

Question 5: Plot Tesla Stock Graph

Use the `make_graph` function to graph the Tesla Stock Data, also provide a title for the graph. Note the graph will only show data upto June 2021.

▼ Hint

```
You just need to invoke the make_graph function with the required parameter to print the graphs.The structure to call the `make_graph` function is `make_graph(tesla_data, tesla_revenue, 'Tesla')`.
```

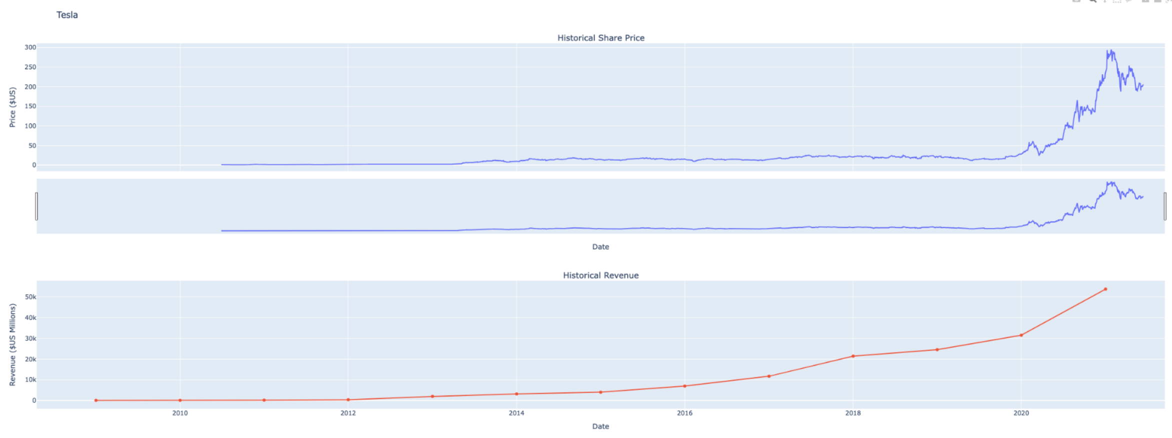
[149]:
```
make_graph(tesla_data, tesla_revenue, 'Tesla')
```

```
/tmp/ipykernel_1498/2068038883.py:5: UserWarning:

The argument 'infer_datetime_format' is deprecated and will be removed in a future version. A strict version of it is now the default, see https://pandas.pydata.org/pdeps/0004-consistent-to-datetime-parsing.html. You can safely remove this argument.

/tmp/ipykernel_1498/2068038883.py:6: UserWarning:

The argument 'infer_datetime_format' is deprecated and will be removed in a future version. A strict version of it is now the default, see https://pandas.pydata.org/pdeps/0004-consistent-to-datetime-parsing.html. You can safely remove this argument.
```



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')`. Note the graph will only show data upto June 2021.

▶ Hint

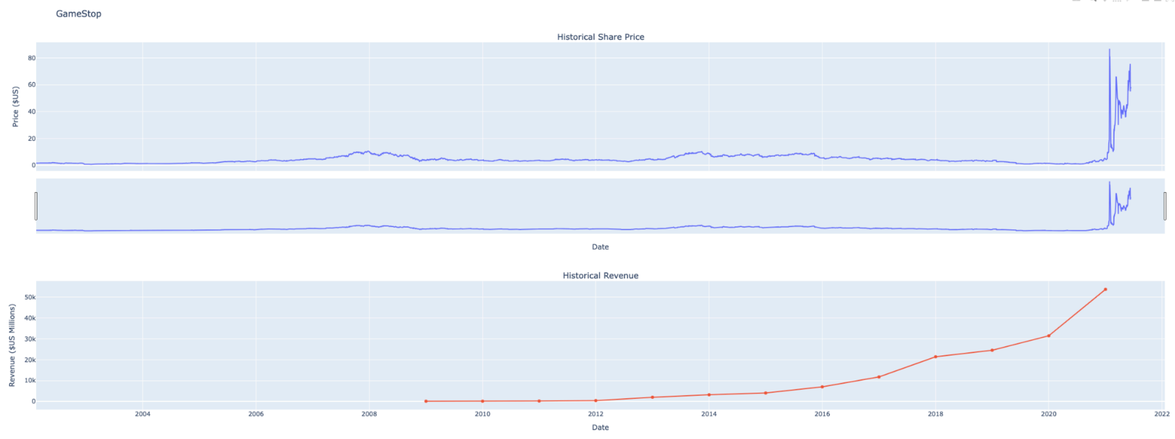
[151]:
```
make_graph(gme_data, gme_revenue, "GameStop")
```

```
/tmp/ipykernel_1498/2068038883.py:5: UserWarning:

The argument 'infer_datetime_format' is deprecated and will be removed in a future version. A strict version of it is now the default, see https://pandas.pydata.org/pdeps/0004-consistent-to-datetime-parsing.html. You can safely remove this argument.

/tmp/ipykernel_1498/2068038883.py:6: UserWarning:

The argument 'infer_datetime_format' is deprecated and will be removed in a future version. A strict version of it is now the default, see https://pandas.pydata.org/pdeps/0004-consistent-to-datetime-parsing.html. You can safely remove this argument.
```


GameStop
Historical Share Price
Price ($US)
Date
Historical Revenue
Revenue ($US Millions)
50k
40k
30k
20k
10k
Date