Module 7: Data Wrangling with Pandas

CPE311 Computational Thinking with Python

Name: Fernandez, Don Eleazar
Section: CPE22S3

Performed on: 04/07/2025
Submitted on: 04/07/2025
Submitted to: Engr. Roman M. Richard

7.1 Supplementary Activity

Using the datasets provided, perform the following exercises:

Exercise 1

We want to look at data for the Facebook, Apple, Amazon, Netflix, and Google (FAANG) stocks, but we were given each as a separate CSV file. Combine them into a single file and store the dataframe of the FAANG data as faang for the rest of the exercises:

1. Read each file in.
2. Add a column to each dataframe, called ticker, indicating the ticker symbol it is for (Apple's is AAPL, for example). This is how you look up a stock. Each file's name is also the ticker symbol, so be sure to capitalize it.
3. Append them together into a single dataframe.
4. Save the result in a CSV file called faang.csv.

In [8]:

```python
import pandas as pd
```

In [9]:

```python
# 1. Read each file in.
facebook = pd.read_csv("fb.csv")
facebook = pd.DataFrame(facebook)
apple = pd.read_csv("aapl.csv")
apple = pd.DataFrame(apple)
amazon = pd.read_csv("amzn.csv")
amazon = pd.DataFrame(amazon)
netflix = pd.read_csv("nflx.csv")
netflix = pd.DataFrame(netflix)
google = pd.read_csv("goog.csv")
google = pd.DataFrame(google)
```

In [10]:

```python
# 2. Add a column to each dataframe, called ticker, indicating the ticker symbol it
facebook["ticker"] = "FB"
apple["ticker"] = "AAPL"
amazon["ticker"] = "AMZN"
netflix["ticker"] = "NFLX"
google["ticker"] = "GOOG"
```

In [11]:
```
# 3. Append them together into a single dataframe.
faang = pd.concat([facebook, apple, amazon, netflix, google])
faang
```

Out[11]:

	date	open	high	low	close	volume	ticker
0	2018-01-02	177.68	181.58	177.5500	181.42	18151903	FB
1	2018-01-03	181.88	184.78	181.3300	184.67	16886563	FB
2	2018-01-04	184.90	186.21	184.0996	184.33	13880896	FB
3	2018-01-05	185.59	186.90	184.9300	186.85	13574535	FB
4	2018-01-08	187.20	188.90	186.3300	188.28	17994726	FB
...	...	...	...	...	...	...	...
246	2018-12-24	973.90	1003.54	970.1100	976.22	1590328	GOOG
247	2018-12-26	989.01	1040.00	983.0000	1039.46	2373270	GOOG
248	2018-12-27	1017.15	1043.89	997.0000	1043.88	2109777	GOOG
249	2018-12-28	1049.62	1055.56	1033.1000	1037.08	1413772	GOOG
250	2018-12-31	1050.96	1052.70	1023.5900	1035.61	1493722	GOOG

1255 rows × 7 columns

In [12]:
```
# 4. Save the result in a CSV file called faang.csv.
faang = faang.to_csv("faang.csv")
faang
```

Exercise 2

- With faang, use type conversion to change the date column into a datetime and the volume column into integers. Then, sort by date and ticker.
- Find the seven rows with the highest value for volume.
- Right now, the data is somewhere between long and wide format. Use melt() to make it completely long format. Hint: date and ticker are our ID variables (they uniquely identify each row). We need to melt the rest so that we don't have separate columns for open, high, low, close, and volume.

In [14]:
```
faang = pd.read_csv("faang.csv")
faang = pd.DataFrame(faang)
```

In [15]:
```
# With faang, use type conversion to change the date column into a datetime and the
faang["date"] = faang["date"].apply(pd.to_datetime)
faang["volume"] = faang["volume"].apply(pd.to_numeric)
faang.dtypes
```

Out[15]:

```
Unnamed: 0             int64
date          datetime64[ns]
open                 float64
high                 float64
low                  float64
close                float64
volume                 int64
ticker                object
dtype: object
```

In [16]:

```
faang = faang.sort_values(["date", "ticker"], ascending = False)
faang
```

Out[16]:

	Unnamed: 0	date	open	high	low	close	volume	ticker
1003	250	2018-12-31	260.1600	270.1001	260.0000	267.6600	13508920	NFLX
1254	250	2018-12-31	1050.9600	1052.7000	1023.5900	1035.6100	1493722	GOOG
250	250	2018-12-31	134.4500	134.6400	129.9500	131.0900	24625308	FB
752	250	2018-12-31	1510.8000	1520.7600	1487.0000	1501.9700	6954507	AMZN
501	250	2018-12-31	157.8529	158.6794	155.8117	157.0663	35003466	AAPL
...	...	...	...	...	...	...	...	...
753	0	2018-01-02	196.1000	201.6500	195.4200	201.0700	10966889	NFLX
1004	0	2018-01-02	1048.3400	1066.9400	1045.2300	1065.0000	1237564	GOOG
0	0	2018-01-02	177.6800	181.5800	177.5500	181.4200	18151903	FB
502	0	2018-01-02	1172.0000	1190.0000	1170.5100	1189.0100	2694494	AMZN
251	0	2018-01-02	166.9271	169.0264	166.0442	168.9872	25555934	AAPL

1255 rows × 8 columns

In [17]:

```
# Find the seven rows with the highest value for volume.
faang = faang.sort_values("volume", ascending = False)
faang.head(7)
```

Out[17]:

	Unnamed: 0	date	open	high	low	close	volume	ticker
142	142	2018-07-26	174.8900	180.1300	173.7500	176.2600	169803668	FB
53	53	2018-03-20	167.4700	170.2000	161.9500	168.1500	129851768	FB
57	57	2018-03-26	160.8200	161.1000	149.0200	160.0600	126116634	FB
54	54	2018-03-21	164.8000	173.4000	163.3000	169.3900	106598834	FB
433	182	2018-09-21	219.0727	219.6482	215.6097	215.9768	96246748	AAPL
496	245	2018-12-21	156.1901	157.4845	148.9909	150.0862	95744384	AAPL
463	212	2018-11-02	207.9295	211.9978	203.8414	205.8755	91328654	AAPL

In [18]:

```
# Right now, the data is somewhere between long and wide format. Use melt() to make
faang_melt = pd.melt(faang, id_vars = ["date", "ticker"], value_vars = ["open", "hi
faang_melt
```

Out[18]:

	date	ticker	variable	value
0	2018-07-26	FB	open	174.8900
1	2018-03-20	FB	open	167.4700
2	2018-03-26	FB	open	160.8200
3	2018-03-21	FB	open	164.8000
4	2018-09-21	AAPL	open	219.0727
...	...	...	...	...
6270	2018-08-09	GOOG	volume	848601.0000
6271	2018-07-10	GOOG	volume	798412.0000
6272	2018-05-24	GOOG	volume	766773.0000
6273	2018-11-23	GOOG	volume	691462.0000
6274	2018-07-03	GOOG	volume	679034.0000

6275 rows × 4 columns

Exercise 3

- Using web scraping, search for the list of the hospitals, their address and contact information. Save the list in a new csv file, hospitals.csv.
- Using the generated hospitals.csv, convert the csv file into pandas dataframe. Prepare the data using the necessary preprocessing techniques.

In [20]:

```
import requests
```

In [21]:

```
# Using web scraping, search for the list of the hospitals, their address and conta

# To get a data from a website
url = "https://rted-web-external.citc.health.nsw.gov.au/api/GetHospitalsReport"

# To make a request
response = requests.get(url, params = {"$limit": 100_000})

# To check if response is ok
if response.ok:
# To download the dataset as csv
    with open("hospitals.csv", "wb") as file:
        file.write(response.content)
    print("File downloaded successfully.")
else:
    print(f'Request was not successful and returned code: {response.status_code}.')
```

```
File downloaded successfully.
```

In [22]:

```
# Using the generated hospitals.csv, convert the csv file into pandas dataframe. Pr

hospitals = pd.read_csv("hospitals.csv")
hospitals = pd.DataFrame(hospitals)
hospitals
```

Out[22]:

Name	Address	Suburb	Postcode	Phone	Email Address	Fax	LHD	Hospit Websi
Albury Wodonga Health	201 Borella Road	Albury	2640	02 6058 4444	NaN	NaN	Albury Wodonga Health	Na
Armidale Rural Referral Hospital	Rusden Street	Armidale	2350	02 6776 9500	NaN	02 6776 4774	Hunter New England Local Health District	Na
Auburn Hospital & Community Health Services	Hargrave Road	Auburn	2144	02 8759 3000	NaN	02 9563 9666	Western Sydney Local Health District	Na
Ballina District Hospital	Cherry Street	Ballina	2478	02 6686 2111	NaN	02 6686 6731	Northern NSW Local Health District	Na
...	...	...	...	...	...	...	...	
Woy Woy Public Hospital	Ocean Beach Road	Woy Woy	2256	02 4344 8444	NaN	02 4344 8555	Central Coast Local Health District	Na
Wyalong Hospital	70 Ungarie Road	West Wyalong	2671	02 6979 0000	NaN	02 6979 0006	Murrumbidgee Local Health District	Na
Wyong Public Hospital	Pacific Highway	Kanwal	2259	02 4394 8000	NaN	02 4393 8333	Central Coast Local Health District	Na
Yass District Hospital	Meehan Street	Yass	2582	(02) 6220 2000	NaN	02 6226 2944	Southern NSW Local Health District	Na
Young District Hospital	Allanan Street	Young	2594	02 6382 8888	NaN	02 6382 4398	Murrumbidgee Local Health District	Na

267 rows × 1 columns

In [23]:

```
import numpy as np

hospitals.replace([np.nan, "NaN"], "Not Available", inplace = True)
hospitals
```

Out[23]:

Name	Address	Suburb	Postcode	Phone	Email Address	Fax	LHD	Hospit Websi
Albury Wodonga Health	201 Borella Road	Albury	2640	02 6058 4444	NaN	NaN	Albury Wodonga Health	Na
Armidale Rural Referral Hospital	Rusden Street	Armidale	2350	02 6776 9500	NaN	02 6776 4774	Hunter New England Local Health District	Na
Auburn Hospital & Community Health Services	Hargrave Road	Auburn	2144	02 8759 3000	NaN	02 9563 9666	Western Sydney Local Health District	Na
Ballina District Hospital	Cherry Street	Ballina	2478	02 6686 2111	NaN	02 6686 6731	Northern NSW Local Health District	Na
...	...	...	...	...	...	...	...	
Woy Woy Public Hospital	Ocean Beach Road	Woy Woy	2256	02 4344 8444	NaN	02 4344 8555	Central Coast Local Health District	Na
Wyalong Hospital	70 Ungarie Road	West Wyalong	2671	02 6979 0000	NaN	02 6979 0006	Murrumbidgee Local Health District	Na
Wyong Public Hospital	Pacific Highway	Kanwal	2259	02 4394 8000	NaN	02 4393 8333	Central Coast Local Health District	Na
Yass District Hospital	Meehan Street	Yass	2582	(02) 6220 2000	NaN	02 6226 2944	Southern NSW Local Health District	Na
Young District Hospital	Allanan Street	Young	2594	02 6382 8888	NaN	02 6382 4398	Murrumbidgee Local Health District	Na

267 rows × 1 columns

7.2 Conclusion:

To conclude, the laboratory activity done helped me better understand what can be done with datasets. In exercise 1, I added a new column to identify each data and combined everything into one dataframe. In exercise 2, I learned how to use the melt() function, which was new to me. It helps rearrange the data based on the chosen columns. Finally, in exercise 3, I learned how to get, utilize, and clean up a dataset from a URL in Python.

In [ ]: