Skills
Network

Peer Review Assignment - Data Engineer - ETL

Estimated time needed: **20** minutes

Objectives

In this final part you will:

- Run the ETL process
- Extract bank and market cap data from the JSON file `bank_market_cap.json`
- Transform the market cap currency using the exchange rate data
- Load the transformed data into a seperate CSV

For this lab, we are going to be using Python and several Python libraries. Some of these libraries might be installed in your lab environment or in SN Labs. Others may need to be installed by you. The cells below will install these libraries when executed.

In [ ]:

```
#!mamba install pandas==1.3.3 -y
#!mamba install requests==2.26.0 -y
```

Imports

Import any additional libraries you may need here.

In [2]:

```
import glob
import pandas as pd
from datetime import datetime
```

As the exchange rate fluctuates, we will download the same dataset to make marking simpler. This will be in the same format as the dataset you used in the last section

In [125…

```
!wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkil
!wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkil
!wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkil
```

```
--2022-06-13 21:30:54--  https://cf-courses-data.s3.us.cloud-object-storage.appdomain.clou
d/IBMDeveloperSkillsNetwork-PY0221EN-SkillsNetwork/labs/module%206/Lab%20-%20Extract%20Tra
nsform%20Load/data/bank_market_cap_1.json
--2022-06-13 21:30:54--  https://cf-courses-data.s3.us.cloud-object-storage.appdomain.clou
d/IBMDeveloperSkillsNetwork-PY0221EN-SkillsNetwork/labs/module%206/Lab%20-%20Extract%20Tra
nsform%20Load/data/bank_market_cap_2.json
--2022-06-13 21:30:55--  https://cf-courses-data.s3.us.cloud-object-storage.appdomain.clou
d/IBMDeveloperSkillsNetwork-PY0221EN-SkillsNetwork/labs/module%206/Final%20Assignment/exch
ange_rates.csv
```

Extract

JSON Extract Function

This function will extract JSON files.

In [102…

```
# Set path
target_file = "bank_market_cap_gbp.csv"
```

In [113…

```
# first making sure can read the file without using a function

xp = pd.read_json('bank_market_cap_1.json')
print(xp)
```

```
                                       Name  Market Cap (US$ Billion)
0                            JPMorgan Chase                   390.934
1   Industrial and Commercial Bank of China                   345.214
2                           Bank of America                   325.331
3                               Wells Fargo                   308.013
4                   China Construction Bank                   257.399
..                                      ...                       ...
65                             Ping An Bank                    37.993
66                        Standard Chartered                   37.319
67                     United Overseas Bank                    35.128
68                                QNB Group                    33.560
69                              Bank Rakyat                    33.081

[70 rows x 2 columns]
```

In [6]:

```
# Then using a function to extract the file

def extract_from_json(file_to_process):
    # if using the suggested parameter of "lines=True" it adds the data as a dictionary a
    #dataframe = pd.read_json(file_to_process, lines=True)
    dataframe = pd.read_json(file_to_process)
    return dataframe
```

In [115…

```
# Confirmation that the function extracts the data properly

extract_from_json('bank_market_cap_1.json')
```

Out[115…

	Name	Market Cap (US$ Billion)
0	JPMorgan Chase	390.934
1	Industrial and Commercial Bank of China	345.214
2	Bank of America	325.331
3	Wells Fargo	308.013
4	China Construction Bank	257.399
...	...	...
65	Ping An Bank	37.993
66	Standard Chartered	37.319
67	United Overseas Bank	35.128
68	QNB Group	33.560
69	Bank Rakyat	33.081

70 rows × 2 columns

Extract Function

Define the extract function that finds JSON file `bank_market_cap_1.json` and calls the function created above to extract data from them. Store the data in a `pandas` dataframe. Use the following list for the columns.

In [7]:

```
# This is the columns that we target for the json files on Market Cap topic

columns=['Name','Market Cap (US$ Billion)']
```

In [8]:

```
# Now using an additional extract function that was originally intended to merge differen
# no longer applicable as we are now just pulling from 1 file but still want to make sure

def extract():
    # Write your code here
    extracted_data = pd.DataFrame(columns = columns)

    jsonfile = 'bank_market_cap_1.json'

    extracted_data = extracted_data.append(extract_from_json(jsonfile), ignore_index=True)

    return extracted_data
```

In [9]:

```
# ------------ Will place together with Logs to call the function

my_extracted_data = extract()
my_extracted_data
```

Out[9]:

	Name	Market Cap (US$ Billion)
0	JPMorgan Chase	390.934
1	Industrial and Commercial Bank of China	345.214
2	Bank of America	325.331
3	Wells Fargo	308.013
4	China Construction Bank	257.399
...	...	...
65	Ping An Bank	37.993
66	Standard Chartered	37.319
67	United Overseas Bank	35.128
68	QNB Group	33.560
69	Bank Rakyat	33.081

70 rows × 2 columns

Question 1 Load the file `exchange_rates.csv` as a dataframe and find the exchange rate for British pounds with the symbol `GBP`, store it in the variable `exchange_rate`, you will be asked for the number. Hint: set the parameter `index_col` to 0.

In [82]:

```
# Write your code here

# Task 1: Load the file exchange_rates.csv as a dataframe
```

```
x = pd.read_csv('exchange_rates.csv', index_col=0)

x2 = pd.DataFrame(x)
x2.index.name='Name'
x2.head()
```

Out[82]:

	Rates
Name	
AUD	1.297088
BGN	1.608653
BRL	5.409196
CAD	1.271426
CHF	0.886083

In [83]:

```
# Task 2: find the exchange rate for British pounds with the symbol GBP, store it in the v

exchange_rate = x2.loc['GBP',"Rates"]
print(exchange_rate)
```

```
0.7323984208000001
```

Transform

Using `exchange_rate` and the `exchange_rates.csv` file find the exchange rate of USD to GBP. Write a transform function that

1. Changes the `Market Cap (US$ Billion)` column from USD to GBP
2. Rounds the Market Cap (US$ Billion)` column to 3 decimal places
3. Rename `Market Cap (US$ Billion)` to `Market Cap (GBP$ Billion)`

In [109…

```
# Something is somehow not working on the original function that I learned on this course

#def transform(data):
    # Write your code here
 #    data['Market Cap (US$ Billion)'] = round(data['Market Cap (US$ Billion)']*0.768568, 3
  #  data.rename(columns={"Market Cap (US$ Billion)": "Market Cap (GPB$ Billion)"}, inplac
   # return transformed_data
```

In [119…

```
# Using the commands by themselves do work!
# Here the transformation of GPB currency instead of US:

my_extracted_data['Market Cap (US$ Billion)'] = round(my_extracted_data['Market Cap (US$ B
my_extracted_data
```

Out[119…

	Name	Market Cap (US$ Billion)
0	JPMorgan Chase	300.459
1	Industrial and Commercial Bank of China	265.320
2	Bank of America	250.039
3	Wells Fargo	236.729

	Name	Market Cap (US$ Billion)
4	China Construction Bank	197.829
...	...	...
65	Ping An Bank	29.200
66	Standard Chartered	28.682
67	United Overseas Bank	26.998
68	QNB Group	25.793
69	Bank Rakyat	25.425

70 rows × 2 columns

In [120…

```
# Using the commands by themselves do work!
# Here the transformation of the name of the column with the now GPB currency:

my_extracted_data.rename(columns={"Market Cap (US$ Billion)": "Market Cap (GPB$ Billion)"
my_extracted_data
```

Out[120…

	Name	Market Cap (GPB$ Billion)
0	JPMorgan Chase	300.459
1	Industrial and Commercial Bank of China	265.320
2	Bank of America	250.039
3	Wells Fargo	236.729
4	China Construction Bank	197.829
...	...	...
65	Ping An Bank	29.200
66	Standard Chartered	28.682
67	United Overseas Bank	26.998
68	QNB Group	25.793
69	Bank Rakyat	25.425

70 rows × 2 columns

In [121…

```
#my_transformed_data = transform(my_extracted_data)
#my_transformed_data

#transform(my_extracted_data)
```

In [138…

```
# Trying writing a new function from scratch:

def my_own_transformation():
    my_extracted_data['Market Cap (US$ Billion)'] = round(my_extracted_data['Market Cap (U
    my_extracted_data.rename(columns={"Market Cap (US$ Billion)": "Market Cap (GPB$ Billio
    return my_extracted_data
```

In [139…

```
my_own_transformation()
```

Out[139…

	Name	Market Cap (GPB$ Billion)
0	JPMorgan Chase	300.459
1	Industrial and Commercial Bank of China	265.320
2	Bank of America	250.039
3	Wells Fargo	236.729
4	China Construction Bank	197.829
...	...	...
65	Ping An Bank	29.200
66	Standard Chartered	28.682
67	United Overseas Bank	26.998
68	QNB Group	25.793
69	Bank Rakyat	25.425

70 rows × 2 columns

In [10]:

```
# Now that know it works, same but with parameter to add argument:

def my_own_new_transformation(data):
    data['Market Cap (US$ Billion)'] = round(data['Market Cap (US$ Billion)']*0.768568, 3)
    data.rename(columns={"Market Cap (US$ Billion)": "Market Cap (GPB$ Billion)"}, inplace
    return my_extracted_data
```

In [11]:

```
# ------------ Will place together with Logs to call the function

transformed_data = my_own_new_transformation(my_extracted_data)
transformed_data
```

Out[11]:

	Name	Market Cap (GPB$ Billion)
0	JPMorgan Chase	300.459
1	Industrial and Commercial Bank of China	265.320
2	Bank of America	250.039
3	Wells Fargo	236.729
4	China Construction Bank	197.829
...	...	...
65	Ping An Bank	29.200
66	Standard Chartered	28.682
67	United Overseas Bank	26.998
68	QNB Group	25.793
69	Bank Rakyat	25.425

70 rows × 2 columns

> ### Now that I know it works as it should can send all the data for the functions to the last cells to work at once

Load

Create a function that takes a dataframe and load it to a csv named `bank_market_cap_gbp.csv` . Make sure to set `index` to `False` .

In [22]:

```
def load(target_file, data_to_load):
    # Write your code here
    data_to_load.to_csv(target_file)
```

In [23]:

```
target_file = "bank_market_cap_gbp_rod.csv"
```

In [24]:

```
# ------------ Will place together with Logs to call the function

load(target_file, transformed_data)
```

In [18]:

```
# Getting a lot of errors like NameError: name 'target_file' is not defined
# Thus making sure the loading process works well

transformed_data.to_csv('testing_loading_613.csv', index=False)

# Worked! File is at local folder
```

In [19]:

```
# Pulling it up to double-check:

w = pd.read_csv("testing_loading_613.csv")
w

# Worked!
```

Out[19]:

	Name	Market Cap (GPB$ Billion)
0	JPMorgan Chase	300.459
1	Industrial and Commercial Bank of China	265.320
2	Bank of America	250.039
3	Wells Fargo	236.729
4	China Construction Bank	197.829
...	...	...
65	Ping An Bank	29.200
66	Standard Chartered	28.682
67	United Overseas Bank	26.998
68	QNB Group	25.793
69	Bank Rakyat	25.425

70 rows × 2 columns

Logging Function

Write the logging function `log` to log your data:

In [25]:

```
# Creating a log file where to store the data

log_file_etl = "rodlogfile.txt"
```

In [32]:

```
def log(my_msg):
    # Write your code here

    timestamp_format = '%Y-%h-%d-%H:%M:%S'

    now = datetime.now()

    timestamp = now.strftime(timestamp_format)

    with open("rodlogfile.txt", "a") as f:
        f.write(timestamp + ',' + my_msg + '\n' + '\n')
        print("")
```

Running the ETL Process

Log the process accordingly using the following `"ETL Job Started"` and `"Extract phase Started"`

In [34]:

```
# Write your code here

log("ETl job starts")

log("Extract phase starts")

# Calling the Extract function
my_extracted_data = extract()

log("Extract phase ends")


log("Transform phase starts")

# Calling the Transform function
transformed_data = my_own_new_transformation(my_extracted_data)

log("Transform phase ends")


log("Load phase starts")

# Calling the load function
load(target_file, transformed_data)

log("Load phase ends")


log("ETl job ends")
```

Extract

`Question 2` Use the function `extract`, and print the first 5 rows, take a screen shot:

In [1]:

```
# Call the function here

# Print the rows here
```

Log the data as `"Extract phase Ended"`

In [ ]:

```
# Write your code here
```

Transform

Log the following `"Transform phase Started"`

In [ ]:

```
# Write your code here
```

`Question 3` Use the function `transform` and print the first 5 rows of the output, take a screen shot:

In [ ]:

```
# Call the function here

# Print the first 5 rows here
```

Log your data `"Transform phase Ended"`

In [ ]:

```
# Write your code here
```

Load

Log the following `"Load phase Started"`.

In [ ]:

```
# Write your code here
```

Call the load function

In [ ]:

```
# Write your code here
```

Log the following `"Load phase Ended"`.

In [ ]:

```
# Write your code here
```

Authors

Ramesh Sannareddy, Joseph Santrcangelo and Azim Hirjani

Other Contributors

Rav Ahuja

Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2020-11-25	0.1	Ramesh Sannareddy	Created initial version of the lab