{

"cells": [

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"<p style=\"text-align:center\">\n",

" <a href=\"https://skills.network\" target=\"\_blank\">\n",

" <img src=\"https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/assets/logos/SN\_web\_lightmode.png\" width=\"200\" alt=\"Skills Network Logo\">\n",

" </a>\n",

"</p>\n"

]

},

{

"cell\_type": "markdown",

"metadata": {

"tags": []

},

"source": [

"# Peer Review Assignment - Data Engineer - Webscraping\n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"Estimated time needed: \*\*20\*\* minutes\n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"## Objectives\n",

"\n",

"In this part you will:\n",

"\n",

"- Use webscraping to get bank information\n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"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.\n"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": [

"#!mamba install pandas==1.3.3 -y\n",

"#!mamba install requests==2.26.0 -y\n",

"!mamba install bs4==4.10.0 -y\n",

"!mamba install html5lib==1.1 -y"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"## Imports\n",

"\n",

"Import any additional libraries you may need here.\n"

]

},

{

"cell\_type": "code",

"execution\_count": 16,

"metadata": {

"tags": []

},

"outputs": [],

"source": [

"import requests\n",

"from bs4 import BeautifulSoup\n",

"import pandas as pd"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"## Extract Data Using Web Scraping\n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"The wikipedia webpage <https://web.archive.org/web/20200318083015/https://en.wikipedia.org/wiki/List_of_largest_banks> provides information about largest banks in the world by various parameters. Scrape the data from the table 'By market capitalization' and store it in a JSON file.\n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"### Webpage Contents\n",

"\n",

"Gather the contents of the webpage in text format using the `requests` library and assign it to the variable <code>html\_data</code>\n"

]

},

{

"cell\_type": "code",

"execution\_count": 17,

"metadata": {

"tags": []

},

"outputs": [],

"source": [

"URL = \"https://en.wikipedia.org/wiki/List\_of\_largest\_banks\"\n",

"html\_data = requests.get(URL).text"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"<b>Question 1</b> Print out the output of the following line, and remember it as it will be a quiz question:\n"

]

},

{

"cell\_type": "code",

"execution\_count": 18,

"metadata": {

"tags": []

},

"outputs": [

{

"data": {

"text/plain": [

"'guage-in-main-page-head'"

]

},

"execution\_count": 18,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"html\_data[101:124]"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"### Scraping the Data\n",

"\n",

"<b> Question 2</b> Using the contents and `beautiful soup` load the data from the `By market capitalization` table into a `pandas` dataframe. The dataframe should have the bank `Name` and `Market Cap (US$ Billion)` as column names. Display the first five rows using head. \n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"Using BeautifulSoup parse the contents of the webpage.\n"

]

},

{

"cell\_type": "code",

"execution\_count": 20,

"metadata": {

"tags": []

},

"outputs": [],

"source": [

"soup=BeautifulSoup(html\_data,'html.parser')"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"Load the data from the `By market capitalization` table into a pandas dataframe. The dataframe should have the bank `Name` and `Market Cap (US$ Billion)` as column names. Using the empty dataframe `data` and the given loop extract the necessary data from each row and append it to the empty dataframe.\n"

]

},

{

"cell\_type": "code",

"execution\_count": 21,

"metadata": {

"tags": []

},

"outputs": [

{

"ename": "IndexError",

"evalue": "list index out of range",

"output\_type": "error",

"traceback": [

"\u001b[0;31m---------------------------------------------------------------------------\u001b[0m",

"\u001b[0;31mIndexError\u001b[0m Traceback (most recent call last)",

"\u001b[0;32m/tmp/ipykernel\_68/864684765.py\u001b[0m in \u001b[0;36m<module>\u001b[0;34m\u001b[0m\n\u001b[1;32m 2\u001b[0m \u001b[0mdata\u001b[0m \u001b[0;34m=\u001b[0m \u001b[0mpd\u001b[0m\u001b[0;34m.\u001b[0m\u001b[0mDataFrame\u001b[0m\u001b[0;34m(\u001b[0m\u001b[0mcolumns\u001b[0m\u001b[0;34m=\u001b[0m\u001b[0;34m[\u001b[0m\u001b[0;34m\"Name\"\u001b[0m\u001b[0;34m,\u001b[0m \u001b[0;34m\"Market Cap (US$ Billion)\"\u001b[0m\u001b[0;34m]\u001b[0m\u001b[0;34m)\u001b[0m\u001b[0;34m\u001b[0m\u001b[0;34m\u001b[0m\u001b[0m\n\u001b[1;32m 3\u001b[0m \u001b[0;34m\u001b[0m\u001b[0m\n\u001b[0;32m----> 4\u001b[0;31m \u001b[0;32mfor\u001b[0m \u001b[0mrow\u001b[0m \u001b[0;32min\u001b[0m \u001b[0msoup\u001b[0m\u001b[0;34m.\u001b[0m\u001b[0mfind\_all\u001b[0m\u001b[0;34m(\u001b[0m\u001b[0;34m'tbody'\u001b[0m\u001b[0;34m)\u001b[0m\u001b[0;34m[\u001b[0m\u001b[0;36m3\u001b[0m\u001b[0;34m]\u001b[0m\u001b[0;34m.\u001b[0m\u001b[0mfind\_all\u001b[0m\u001b[0;34m(\u001b[0m\u001b[0;34m'tr'\u001b[0m\u001b[0;34m)\u001b[0m\u001b[0;34m:\u001b[0m\u001b[0;34m\u001b[0m\u001b[0;34m\u001b[0m\u001b[0m\n\u001b[0m\u001b[1;32m 5\u001b[0m \u001b[0mcol\u001b[0m \u001b[0;34m=\u001b[0m \u001b[0mrow\u001b[0m\u001b[0;34m.\u001b[0m\u001b[0mfind\_all\u001b[0m\u001b[0;34m(\u001b[0m\u001b[0;34m\"td\"\u001b[0m\u001b[0;34m)\u001b[0m\u001b[0;34m\u001b[0m\u001b[0;34m\u001b[0m\u001b[0m\n\u001b[1;32m 6\u001b[0m \u001b[0;31m#have to make sure there is no empty tr\u001b[0m\u001b[0;34m\u001b[0m\u001b[0;34m\u001b[0m\u001b[0;34m\u001b[0m\u001b[0m\n",

"\u001b[0;31mIndexError\u001b[0m: list index out of range"

]

}

],

"source": [

" #Write your code here\n",

"data = pd.DataFrame(columns=[\"Name\", \"Market Cap (US$ Billion)\"])\n",

"\n",

"for row in soup.find\_all('tbody')[3].find\_all('tr'):\n",

" col = row.find\_all(\"td\")\n",

" #have to make sure there is no empty tr\n",

" if len(col)==3:\n",

" Bank\_Name = col[1].text.strip()\n",

" Market\_Cap = col[2].text.strip()\n",

" #finally we append the data of each row to the talbe\n",

" data = data.append({\"Name\":Bank\_Name,\"Market Cap (US$ Billion)\":Market\_Cap},ignore\_index=Tru)"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"\*\*Question 3\*\* Display the first five rows using the `head` function.\n"

]

},

{

"cell\_type": "code",

"execution\_count": 22,

"metadata": {

"tags": []

},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>Name</th>\n",

" <th>Market Cap (US$ Billion)</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

"Empty DataFrame\n",

"Columns: [Name, Market Cap (US$ Billion)]\n",

"Index: []"

]

},

"execution\_count": 22,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"#Write your code here\n",

"data.head(5)"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"\n",

"### Loading the Data\n",

"\n",

"Load the `pandas` dataframe created above into a JSON named `bank\_market\_cap.json` using the `to\_json()` function.\n"

]

},

{

"cell\_type": "code",

"execution\_count": 14,

"metadata": {

"collapsed": true,

"jupyter": {

"outputs\_hidden": true

},

"tags": []

},

"outputs": [

{

"ename": "NameError",

"evalue": "name 'data' is not defined",

"output\_type": "error",

"traceback": [

"\u001b[0;31m---------------------------------------------------------------------------\u001b[0m",

"\u001b[0;31mNameError\u001b[0m Traceback (most recent call last)",

"\u001b[0;32m/tmp/ipykernel\_68/3367239104.py\u001b[0m in \u001b[0;36m<module>\u001b[0;34m\u001b[0m\n\u001b[1;32m 1\u001b[0m \u001b[0;31m#Write your code here\u001b[0m\u001b[0;34m\u001b[0m\u001b[0;34m\u001b[0m\u001b[0;34m\u001b[0m\u001b[0m\n\u001b[0;32m----> 2\u001b[0;31m \u001b[0mdata\u001b[0m\u001b[0;34m.\u001b[0m\u001b[0mhead\u001b[0m\u001b[0;34m(\u001b[0m\u001b[0;34m)\u001b[0m\u001b[0;34m\u001b[0m\u001b[0;34m\u001b[0m\u001b[0m\n\u001b[0m",

"\u001b[0;31mNameError\u001b[0m: name 'data' is not defined"

]

}

],

"source": [

"#Write your code here\n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"## Authors\n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"Ramesh Sannareddy, Joseph Santarcangelo and Azim Hirjani\n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"### Other Contributors\n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"Rav Ahuja\n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"## Change Log\n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"| Date (YYYY-MM-DD) | Version | Changed By | Change Description |\n",

"| ----------------- | ------- | ----------------- | ---------------------------------- |\n",

"| 2022-07-12 | 0.2 | Appalabhaktula Hema | Corrected the code and markdown |\n",

"| 2020-11-25 | 0.1 | Ramesh Sannareddy | Created initial version of the lab |\n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"Copyright Â© 2020 IBM Corporation.\n"

]

}

],

"metadata": {

"kernelspec": {

"display\_name": "Python",

"language": "python",

"name": "conda-env-python-py"

},

"language\_info": {

"codemirror\_mode": {

"name": "ipython",

"version": 3

},

"file\_extension": ".py",

"mimetype": "text/x-python",

"name": "python",

"nbconvert\_exporter": "python",

"pygments\_lexer": "ipython3",

"version": "3.7.12"

},

"toc-autonumbering": true,

"toc-showcode": true,

"toc-showtags": false

},

"nbformat": 4,

"nbformat\_minor": 4

}