{

"cells": [

{

"cell\_type": "markdown",

"metadata": {},

"source": [

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

" <a href=\"https://skills.network/?utm\_medium=Exinfluencer&utm\_source=Exinfluencer&utm\_content=000026UJ&utm\_term=10006555&utm\_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkPY0221ENSkillsNetwork899-2023-01-01\">\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": {},

"source": [

"# Peer Review Assignment - Data Engineer - Extract API Data\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",

"- Collect exchange rate data using an API\n",

"- Store the data as a CSV\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": 1,

"metadata": {

"tags": []

},

"outputs": [],

"source": [

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

"#!mamba install requests==2.26.0 -y"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"## Imports\n",

"\n",

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

]

},

{

"cell\_type": "code",

"execution\_count": 2,

"metadata": {

"tags": []

},

"outputs": [],

"source": [

"import requests\n",

"import pandas as pd"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"## Extract Data Using an API\n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"Using ExchangeRate-API we will extract currency exchange rate data. Use the below steps to get the access key and to get the data. \n",

"1. Open the url : <https://exchangeratesapi.io/> and click on \*\*Get Free API Key\*\*. \n",

"2. Subscribe for Free plan and Sign-in with the Google Account. \n",

"3. Once the account is created you will be redirected to [https://apilayer.com](https://apilayer.com/) website.\n",

"2. Now, click on the \*\*user icon\*\* and click \*\*Account\*\* as shown below:\n",

"\n",

"<img src=\"https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0221EN-SkillsNetwork/labs/module%206/Final%20Assignment/Images/account.png\">\n",

"\n",

"3. Scroll down and you will get the API Key section. Copy the API key and use in the url in Question 1.\n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"### Call the API\n",

"\n",

" <b> Question 1</b> Using the `requests` library call the endpoint given above and save the text, remember the first few characters of the output: \n"

]

},

{

"cell\_type": "code",

"execution\_count": 33,

"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>success</th>\n",

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

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

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

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

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

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

" <td>True</td>\n",

" <td>1697024824</td>\n",

" <td>EUR</td>\n",

" <td>2023-10-11</td>\n",

" <td>3.893479</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>True</td>\n",

" <td>1697024824</td>\n",

" <td>EUR</td>\n",

" <td>2023-10-11</td>\n",

" <td>79.956803</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>True</td>\n",

" <td>1697024824</td>\n",

" <td>EUR</td>\n",

" <td>2023-10-11</td>\n",

" <td>105.847765</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>True</td>\n",

" <td>1697024824</td>\n",

" <td>EUR</td>\n",

" <td>2023-10-11</td>\n",

" <td>421.491099</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>True</td>\n",

" <td>1697024824</td>\n",

" <td>EUR</td>\n",

" <td>2023-10-11</td>\n",

" <td>1.910660</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>True</td>\n",

" <td>1697024824</td>\n",

" <td>EUR</td>\n",

" <td>2023-10-11</td>\n",

" <td>265.374085</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>True</td>\n",

" <td>1697024824</td>\n",

" <td>EUR</td>\n",

" <td>2023-10-11</td>\n",

" <td>20.010335</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>True</td>\n",

" <td>1697024824</td>\n",

" <td>EUR</td>\n",

" <td>2023-10-11</td>\n",

" <td>9541.384015</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>True</td>\n",

" <td>1697024824</td>\n",

" <td>EUR</td>\n",

" <td>2023-10-11</td>\n",

" <td>22.820518</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>True</td>\n",

" <td>1697024824</td>\n",

" <td>EUR</td>\n",

" <td>2023-10-11</td>\n",

" <td>341.323629</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"<p>170 rows Ã— 5 columns</p>\n",

"</div>"

],

"text/plain": [

" success timestamp base date rates\n",

"AED True 1697024824 EUR 2023-10-11 3.893479\n",

"AFN True 1697024824 EUR 2023-10-11 79.956803\n",

"ALL True 1697024824 EUR 2023-10-11 105.847765\n",

"AMD True 1697024824 EUR 2023-10-11 421.491099\n",

"ANG True 1697024824 EUR 2023-10-11 1.910660\n",

".. ... ... ... ... ...\n",

"YER True 1697024824 EUR 2023-10-11 265.374085\n",

"ZAR True 1697024824 EUR 2023-10-11 20.010335\n",

"ZMK True 1697024824 EUR 2023-10-11 9541.384015\n",

"ZMW True 1697024824 EUR 2023-10-11 22.820518\n",

"ZWL True 1697024824 EUR 2023-10-11 341.323629\n",

"\n",

"[170 rows x 5 columns]"

]

},

"execution\_count": 33,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"# Write your code here\n",

"url = url =\"http://api.exchangeratesapi.io/v1/latest?base=EUR&access\_key=5373d3a867921d39fd914dec6ae1b10c\"\n",

"data1 = requests.get (url). text\n",

"\n",

"type(data1)\n",

"\n",

"data2 = requests.get (url)\n",

"type(data2)\n",

"\n",

"\n",

"data = data2.json()\n",

"type(data)\n",

"\n",

"df = pd.DataFrame(data)\n",

"df\n",

"\n",

"\n",

"\n",

"\n",

"\n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"### Save as DataFrame\n",

"\n",

" <b> Question 2</b> Using the data gathered turn it into a `pandas` dataframe. The dataframe should have the Currency as the index and `Rate` as their columns. Make sure to drop unnecessary columns.\n"

]

},

{

"cell\_type": "code",

"execution\_count": 3,

"metadata": {

"tags": []

},

"outputs": [

{

"ename": "NameError",

"evalue": "name 'pd' 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\_620/2050433470.py\u001b[0m in \u001b[0;36m<module>\u001b[0;34m\u001b[0m\n\u001b[1;32m 1\u001b[0m \u001b[0;31m# Turn the data into a dataframe\u001b[0m\u001b[0;34m\u001b[0m\u001b[0;34m\u001b[0m\u001b[0;34m\u001b[0m\u001b[0m\n\u001b[1;32m 2\u001b[0m \u001b[0;34m\u001b[0m\u001b[0m\n\u001b[0;32m----> 3\u001b[0;31m \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\"Currency\"\u001b[0m\u001b[0;34m,\u001b[0m\u001b[0;34m\"Rates\"\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 4\u001b[0m \u001b[0;34m\u001b[0m\u001b[0m\n\u001b[1;32m 5\u001b[0m \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;36m2\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[0;31mNameError\u001b[0m: name 'pd' is not defined"

]

}

],

"source": [

"# Turn the data into a dataframe\n",

"\n",

"data = pd.DataFrame(columns = [\"Currency\",\"Rates\"])\n",

"\n",

"for row in soup.find\_all('tbody')[2].find\_all('tr'):\n",

"\n",

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

"\n",

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

"\n",

" if len(col)!=0:\n",

"\n",

" Bank\_Name = col[0].text.strip()\n",

"\n",

" Market\_Cap = col[1].text.strip()\n",

"\n",

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

"\n",

" data = data.append({\"Name\":Bank\_Name,\n",

"\n",

" \"Market Cap (US$ Billion)\":Market\_Cap},\n",

"\n",

" ignore\_index=True)"

]

},

{

"cell\_type": "code",

"execution\_count": 11,

"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>success</th>\n",

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

" </tr>\n",

" <tr>\n",

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

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

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

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

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

" <td>False</td>\n",

" <td>101</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>False</td>\n",

" <td>You have not supplied an API Access Key. [Requ...</td>\n",

" </tr>\n",

" <tr>\n",

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

" <td>False</td>\n",

" <td>missing\_access\_key</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" success error\n",

"Currency \n",

"code False 101\n",

"info False You have not supplied an API Access Key. [Requ...\n",

"type False missing\_access\_key"

]

},

"execution\_count": 11,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"# Drop unnescessary columns\n",

"df.index.name = 'Currency'\n",

"df\n"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"### Load the Data\n",

"\n",

"Using the dataframe save it as a CSV names `exchange\_rates\_1.csv`.\n"

]

},

{

"cell\_type": "code",

"execution\_count": 1,

"metadata": {

"tags": []

},

"outputs": [

{

"ename": "NameError",

"evalue": "name 'df' 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\_620/1231286398.py\u001b[0m in \u001b[0;36m<module>\u001b[0;34m\u001b[0m\n\u001b[1;32m 1\u001b[0m \u001b[0;31m# Save the Dataframe\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[0mdf\u001b[0m\u001b[0;34m.\u001b[0m\u001b[0mto\_csv\u001b[0m\u001b[0;34m(\u001b[0m\u001b[0;34mr'exchange\_rates\_1.csv'\u001b[0m\u001b[0;34m,\u001b[0m \u001b[0mindex\u001b[0m \u001b[0;34m=\u001b[0m \u001b[0;32mFalse\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 'df' is not defined"

]

}

],

"source": [

"# Save the Dataframe\n",

"df.to\_csv(r'exchange\_rates\_1.csv', index = False)"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"Your CSV should be in this format with more currencies\n",

"\n",

"| | Rates |\n",

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

"| AED| 4.398618 |\n",

"| AFN| 92.917693 | \n",

"| ALL| 123.099093 |\n",

"| AMD| 621.935674 |\n",

"| ANG| 2.149648 | \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",

"| 2022-05-06 | 0.3 | Malika | Updated instructions to get the API and the url|\n",

"| 2021-04-15 | 0.2 | Malika | Updated the lab from USD to EUR |\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. This notebook and its source code are released under the terms of the [MIT License](https://cognitiveclass.ai/mit-license?utm\_medium=Exinfluencer&utm\_source=Exinfluencer&utm\_content=000026UJ&utm\_term=10006555&utm\_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkPY0221ENSkillsNetwork899-2023-01-01&cm\_mmc=Email\_Newsletter-\_-Developer\_Ed%2BTech-\_-WW\_WW-\_-SkillsNetwork-Courses-IBM-DA0321EN-SkillsNetwork-21426264&cm\_mmca1=000026UJ&cm\_mmca2=10006555&cm\_mmca3=M12345678&cvosrc=email.Newsletter.M12345678&cvo\_campaign=000026UJ).\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"

}

},

"nbformat": 4,

"nbformat\_minor": 4

}