3. Python Project for Data Engineering

PROYECTO DE WEB SCRAPING
- Ejemplos
- Project overview
- Scenario
- Project Tasks
- Resources
- Solution

EXTRACT API DATA

Watson Studio
- import glob
- The URL always points to the most recent version of the notebook. If you want the URL to point to your most recent changes, you must save a new version of the notebook with the File > Save Version command. Clicking File > Save only saves your changes; the command doesn't create a new version of the notebook and the share URL isn't updated. The URL still points to an older version of the notebook.

PROYECTO DE WEB SCRAPING

Ejemplos

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

list_csv = glob.glob('*.csv')
list_csv:['source1.csv', 'source2.csv', 'source3.csv']

list_json = glob.glob('*.json')
list_json:['source1.json', 'source2.json']

def extract_from_csv(file_to_process):
  dataframe = pd.read_csv(file_to_process)
  return dataframe

df = extract_from_csv('source1.csv')

def extract_from_json(file_to_process):
  dataframe = pd.read_json(file_to_process, lines=True)
  return dataframe

df = extract_from_json('source1.json')

def extract():
  #create an empty data frame to hold extracted data
  extracted_data = pd.DataFrame(columns = ['name', 'height', 'weight'])
  #process all csv files
  for csvfile in glob.glob("*.csv"):
    extracted_data = extracted_data.append(extract_from_csv(csvfile), ignore_index=True)
  #process all json files
  for jsonfile in glob.glob('*.json'):
    extracted_data = extracted_data.append(extract_from_json(jsonfile), ignore_index = True)
  return extracted_data


def transform():
  #convert height which is in inches to millimiter
  #convert inches to meters and round off to two decimals
  data['height'] = round(data.height * 0.0254, 2)
  #convert pounds to kilograms and round off to two decimals
  data['weight'] = round(data.weight * 0.045359237, 2)
  return data
```

```
def load(targetfile, data_to_load):
  data_to_load.to_csv(targetfile)

def log(message):
  timestamp_format = '%Y-%h-%d-%H:%M:%S'
  now = datetime.now()
  timestamp = now.strftime(timestamp_format)
  with open ("logfile.txt", "a") as f:
    f.write( timestamp + ',' + message + '\n')


log("ETL JOB STARTED")
log("extract")
extracted_data = extract()
log("transform")
transformed_data = transform()
#....
```

Project overview

Scenario

For this project, you will assume the role of data engineer working for an international financial analysis company. Your company tracks stock prices, commodities, forex rates, inflation rates. Your job is to extract financial data from various sources like websites, APIs and files provided by various financial analysis firms. After you collect the data, you extract the data of interest to your company and transform it based on the requirements given to you. Once the transformation is complete you load that data into a database.

Project Tasks

In this project you will:

- Collect data using APIs
- Collect data using webscraping.
- Download files to process.
- Read csv, xml and json file types.
- Extract data from the above file types.
- Transform data.
- Use the built in logging module.
- Save the transformed data in a ready-to-load format which data engineers can use to load the data.

Resources

How to create an empty DataFrame and append rows & columns to it in Pandas? - GeeksforGeeks

Let's discuss how to create an empty DataFrame and append rows & columns to it in Pandas. There are multiple ways in which we can do this task. Method #1: Create a complete empty DataFrame without any column name or indices and then appending columns one by one to it.

https://www.geeksforgeeks.org/how-to-create-an-empty-dataframe-and-append-rows-columns-to-it-in-pandas/

Solution

> The wikipedia webpage 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.

```
!pip install pandas
!pip install bs4
!pip install requests

from bs4 import BeautifulSoup
import requests
import pandas as pd
import json

html_data = requests.get("https://en.wikipedia.org/wiki/List_of_largest_banks").text

soup = BeautifulSoup(html_data, "html.parser")

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

for row in soup.find_all('tbody')[3].find_all('tr'):
    col = row.find_all('td')data = pd.DataFrame(columns=["Currency", "Rate"])
data.set_index("Currency")
rates = req_json["rates"]

for r in rates:
    data = data.append({"Currency": r, "Rate": rates[r]}, ignore_index = True)

data.head()
    if (len(col)):
        second_a_tag = col[1].find_all('a')[1]
        name = second_a_tag.get('title')
        marketcap = second_a_tag.find_next('td').text.strip()
        data = data.append({"Name": name, "Market Cap (US$ Billion)": marketcap}, ignore_index = True)

data.to_json(r'bank_market_cap.json') #export to json
```

EXTRACT API DATA

> Using ExchangeRate-API we will extract currency exchange rate data.
> Use the below steps to get the access key and to get the data.
> Open the url : https://exchangeratesapi.io/ and create a free account.
> Once the account is created. You will get the Get the Free API key option on the top

```
!pip install pandas
!pip install requests

import requests
import pandas as pd
import requests

url = "http://api.exchangeratesapi.io/v1/latest?base=EUR&access_key=bcc6a93f8d1d9fa8c78cc71b578e6fc0" #Make sure to cha
nge ******* to your API key.
req = requests.get(url)
req_text = req.text

#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.

data = pd.DataFrame(columns=["Currency", "Rate"])
data.set_index("Currency")
rates = req_json["rates"]

for r in rates:
    data = data.append({"Currency": r, "Rate": rates[r]}, ignore_index = True)
```

Asi como esta el codigo, si exporto a csv asi:

```
data.to_csv('exchange_rates_1.csv', index = False)
```

Obtengo esto:

	Currency	Rate
1	AED	4.304609
2	AFN	103.261224
3	ALL	121.394951
4	AMD	577.604074
5	ANG	2.106751
6	AOA	704.201627
7	ARS	115.518499
8	AUD	1.608366
9	AWG	2.110143
10	AZN	2.002667
11	BAM	1.956232
12	BBD	2.369787
13	BDT	100.039161
14	BGN	1.955791
15	BHD	0.441696
16	BIF	2332.059161
17	BMD	1.171976

Pero lo que me piden es esto:

	Rates
AED	4.398618
AFN	92.917693
ALL	123.099093
AMD	621.935674
ANG	2.149648

Watson Studio

In this lab, you will learn: Import the notebook in Watson Studio If you have not created a Watson service before proceed with Step 1, otherwise go to Step 2: For this project, you will use your IBM Watson Studio account from the previous chapter.

https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0101EN-SkillsNetwork/labs/FinalModule_Coursera/IBM_Cloud_and_Watson_Setup.md.html?origin=www.coursera.org

Mi assignment:

ETL_Engineer_Peer_Review_Assignment - IBM Cloud Pak for Data

https://eu-de.dataplatform.cloud.ibm.com/analytics/notebooks/v2/7517989b-19df-49c7-8035-62d29bc8fa3f/view?access_token=a39ec641c7bc31df275ca6c7549c8aa11bd64c8dbfe2c58ace97a6c15db5f861

import glob

Glob is a general term used to define techniques to match specified patterns according to rules related to Unix shell. Linux and Unix systems and shells also support glob and also provide function `glob()` in system libraries.

In Python, the glob module is used to retrieve **files/pathnames** matching a specified pattern. The pattern rules of glob follow standard Unix path expansion rules. It is also predicted that according to benchmarks it is faster than other methods to match pathnames in directories. With glob, we can also use wildcards `("*, ?, [ranges])` apart from exact string search to make path retrieval more simple and convenient.

The URL always points to the most recent version of the notebook. If you want the URL to point to your most recent changes, you must save a new version of the notebook with the File > Save Version command. Clicking File > Save only saves your changes; the command doesn't create a new version of the notebook and the share URL isn't updated. The URL still points to an older version of the notebook.