Data Engineering: **Pipelines on the Cloud**

I started a Data science Boot-camp almost two months ago and it has all been about Data. After analysing Data for quite some time, we were given the Task to Engineering Data for a company called Gans. This will help the company to expand his idea on a sustainable mobility and promote their services in spite of the bigger concurrence in the current market. in order for me to accomplish the task, some key steps were followed:

- **gathering necessary data from different sources**

Figure 1: Gathering Data

At the beginning of the project, some useful Data were collected from different sites on the Web. After collecting the Data, the next think was Data cleaning. Several approaches were considered in order to gather those Data.

first of all, Web scraping using beautiful soup. This allowed Data scraping from Wikipedia and different sources with free data access. One of the used codes bellowed.

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

url = "https://en.wikipedia.org/wiki/List_of_municipalities_of_Cameroon"

lapage = requests.get(url)
lapage.status_code

code = BeautifulSoup(lapage.content, "html.parser")

ville= []

for v in code.select("td:nth-child(1) > a"):
    ville.append(v.get_text())
len(ville)

region = []

for r in code.select(" td:nth-child(2)"):
    region.append(r.get_text())

len(region)

population =[]

i =1
while i!=88:
    for p in code.select("td:nth-child(3)"):

        population.append(p.get_text())


len(population)
```

Figure 2:Gathering Data using Python

Due to the fact that bike riding is mostly on days with no rain, checking the weather some days ahead is necessary. This helps the company to make more bikes available and accessible for the clients that will like to use them on those days.

Regarding the goal ahead, Apis is an appropriate source for the collection of this type of Data. In addition, it has been demonstrated that travellers mostly use bikes to move from one place to another reason why it was also important to collect flight Data particularly the "arrivals". And for this purpose, RapidAPI were used since it can deliver flights and airport information in advance. After data collection and data cleaning, the next was to store the Data

- **Transferring collected Data**

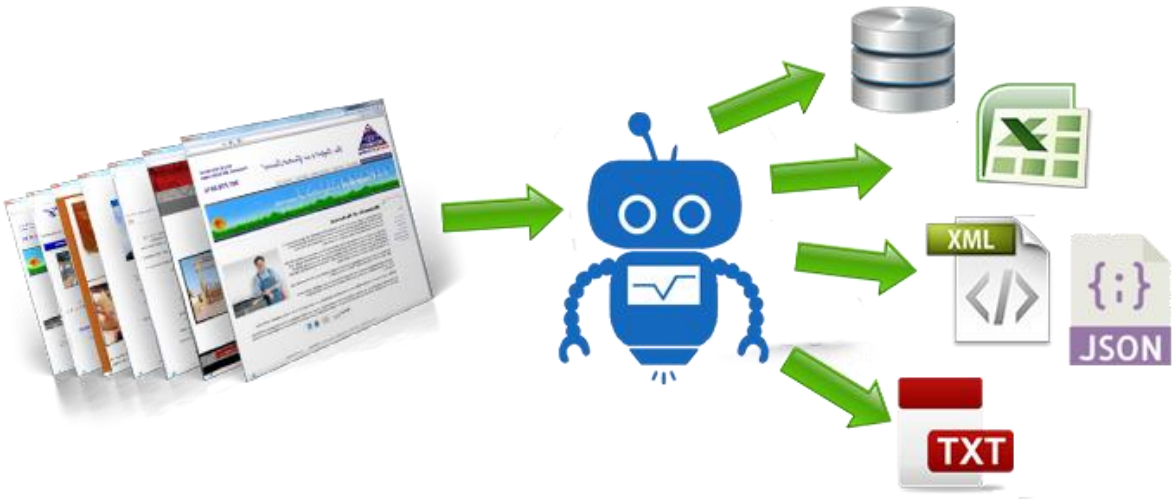

Figure 3: Data tranfer in process

For this purpose, sqlalchemy is the best tool to save all the necessary information into mysql workbench. In mysql where workbench tables were created with appropriate datatype and constraints, relationships were crafted between tables and the diagram also shows the connection between those tables.

At this point the Data are ready to be used by the company. But there were some important questions remaining: what will happen if the local storage gets destroyed? Or what if another type of disaster occurs and of the Data are loss? Losing the Data will be costly of time wasted in the previous work and time also means money. Even some backup might not be able of help if such a disaster occurs. In order to minimize disasters, cloud using was the best option to store the Data

```python
import json
import requests
import pandas as pd
import sqlalchemy

weather_0 = pd.read_csv('weather_data_info_WFC.csv')

weather = weather_0[['id', 'Country', 'DateTime', 'Main', 'Temperatur', 'Description', 'Longitude'

schema="project3"
host="project3-wbs.cty4wakvv8pl.us-east-1.rds.amazonaws.com"#"127.0.0.1"
user="admin"#"root"
password="Blessingnate1"
port=3306
con = f'mysql+pymysql://{user}:{password}@{host}:{port}/{schema}'

weather.to_sql('weather',
               if_exists='append',
               con=con,
               index=False)
```

Figure 4: From Csv_file to sql table

- **Transferring Data to the cloud**

Figure 5: Process of transfering Data to the cloud

In order to store different tables saved in mysql workbench, an account should be created on the cloud. So doing, the data would never be lost as long as there is a functioning connection with the cloud.

Some of the information mention above like flight and weather information might not be the same every day and in this case, they will need to be updated often. It can be exhausting to update tables saved in the database manually all the time, the time to update can sometimes be forgotten. To avoid forgetting doing it at the appointed time,

best practice could be to create a lambda function. The lambda function would contain a trigger where the time is being set to when the data should be updated and also the amount of data the user might want to receive.

Figure 6: Manualy uptating data

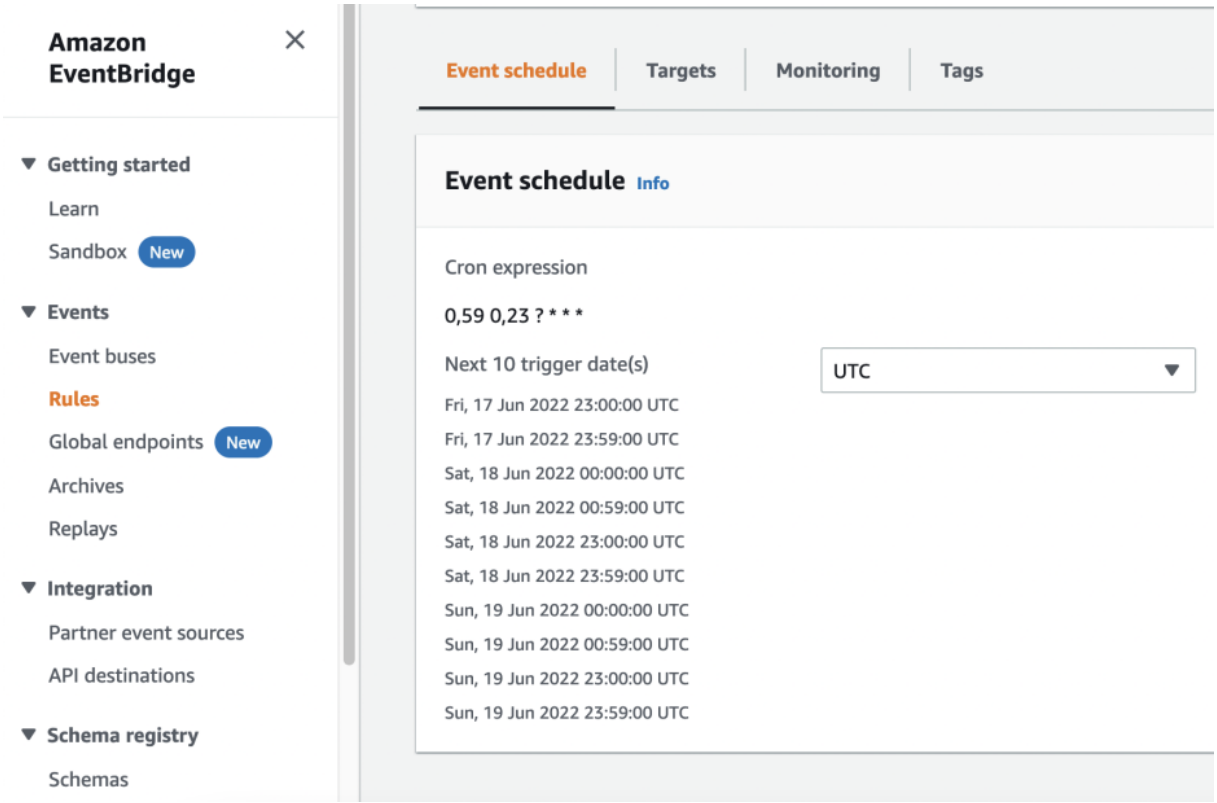

Figure 7: Trigger scheduled for the updating Task

Finally:

Now that data have been connected and store into a running system, it is important not neglect the fact that everything is not just picture perfect as it might sounds like. Collecting Data from the web is not always easy because some pages are not accessible and some accessible pages are costly. Even after scraping data and saving into the cloud, data that are triggered may generate some costs that will increase along the line depending on the amount of information update. If the trigger also stops working due to one unknown reason, the information will also not be updated.

Nevertheless, going through all these processes help me learn a lot about a job as a data scientist and the task ahead. Just excited for what is in store for me in the future. And i am very grateful for have been through this process.

C/Users/steph/OneDrive/Desktop