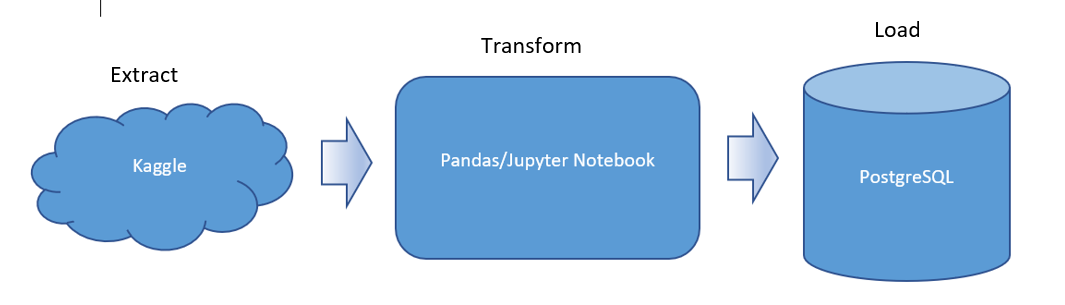
**ETL Project**

By Sherin Thomas, Therese Reed and Anjali Khare

**Introduction**

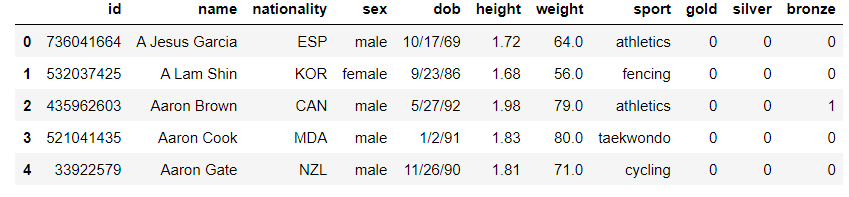
We extracted data from Kaggle, transformed using Pandas & Jupyter Notebook and loaded it in the form of tables in PostgreSQL.

****

**Extraction**

We used datasets from the public platform Kaggle and following csv files. All of our data is based on Olympics 2016 and 2021. These were the most recent ones we could find. Medals for 2021 contains the data till 27th July 2021. The sources for our dataset are as follows:

* 2016 athlete data
* 2016 country data
* 2021 athlete data
* 2021 medals data

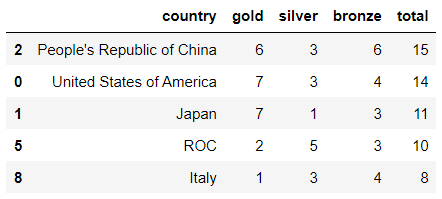
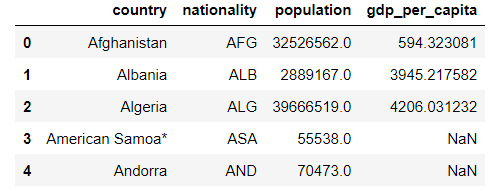


**Figure 1:** Raw data (2016 athlete data)

**Transformation**

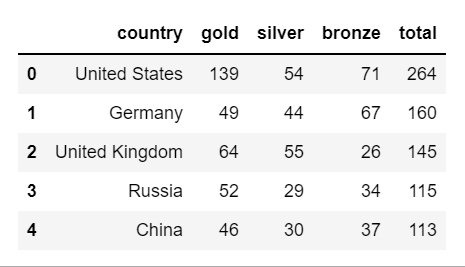
Our first step in cleaning up the datasets by using jupyter notebook and pandas involved figuring out which columns were not relevant.

* We transformed medals data of 2021, dropped “Rank” and “Rank by Total” columns and renamed “Team/NOC” column name to Country. **(Figure 2)**
* We renamed country “code” column in countries dataframe to “nationality” in order to merge the table with athlete data of 2016. **(Figure 3)**

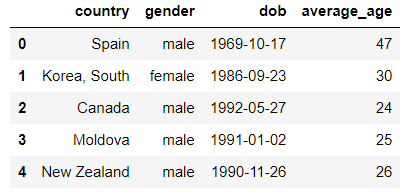
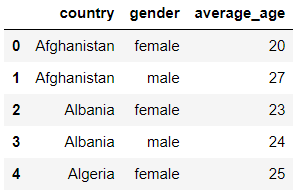
**Figure 2:** 2020 medals data **Figure 3:** 2016 country data

* We used group by function on country column of 2016 data to find the gold, silver and bronze medals for each country. These columns were added to find the total medals for each country. **(Figure 4)**



**Figure 4:** medals\_2016\_df

* We used country, gender and dob column from 2016 data to find average age of men and women for each country and saved in dataframe “age\_2016\_df”. **(Figure 5)**
* We used the dob column to find the age with pandas datetime object. There was a limitation with datetime index function when the dob is less than 1969. We created a user defined function to adjust the year when the dob is less than 1969.
* We used group by function with country and gender and calculated the mean to find the average age of players for each country per gender and saved in dataframe “avg\_age\_2016\_df”. **(Figure 6)**

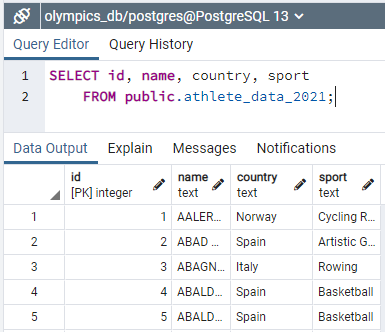
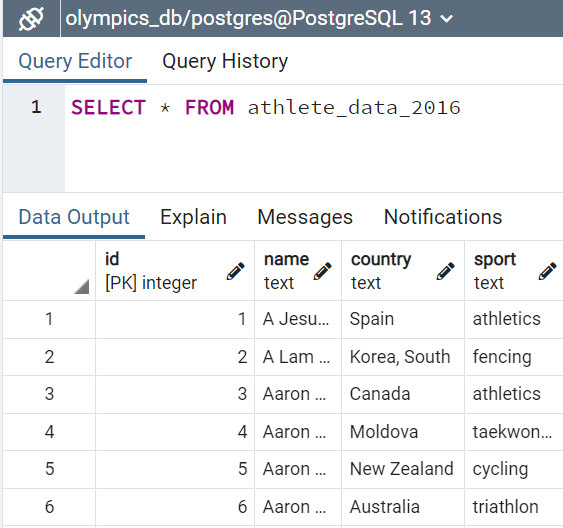
 

**Figure 5:** age\_2016\_df **Figure 6:** avg\_age\_2016\_df

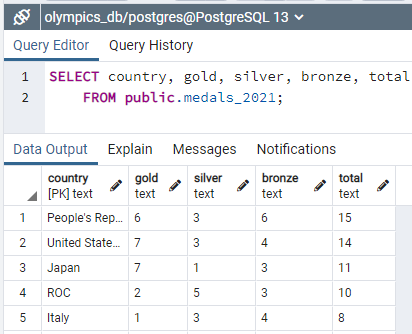
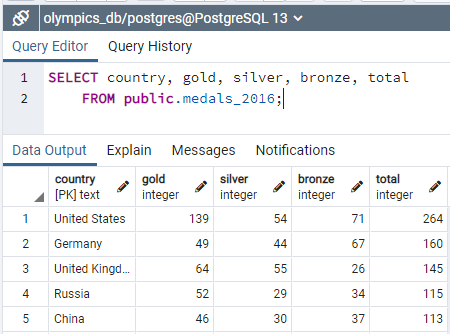
**Load**

The last step was to load our final dataframes into a database. The tables were created in PostgreSQL using SQL queries with the help of pgAdmin 4 tool. Using SQLAlchemy, we created a PostgreSQL database engine and uploaded dataframes into tables.

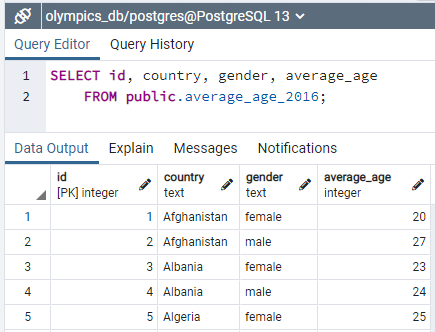
* Dataframe “clean\_ath\_2016\_df” loaded to “Olympics\_db.athlete\_data\_2016” **(Figure 6)**
* Dataframe “clean\_ath\_2021\_df” loaded to “Olympics\_db.athlete\_data\_2021” **(Figure 8)**
* Dataframe “medals\_2021\_df” loaded to “Olympics\_db.medals\_2021” **(Figure 9)**
* Dataframe “medals\_2016\_df” loaded to “Olympics\_db.medals\_2016” **(Figure 10)**
* Dataframe “avg\_age\_2016\_df” loaded to “Olympics\_db.average\_age\_2016” **(Figure 11)**

**Figure 7:** athlete\_data\_2021 **Figure 8:** athlete\_data\_2016

**Figure 9:** medals\_2021 **Figure 10:** medals\_2016



**Figure 11:** average\_age\_2016

**Resources**

**Links:** [2021 Olympics in Tokyo | Kaggle](https://www.kaggle.com/arjunprasadsarkhel/2021-olympics-in-tokyo)

[2016 Olympics in Rio de Janeiro | Kaggle](https://www.kaggle.com/rio2016/olympic-games)