ETL Project – MUBI Review and Database

# Introduction

We have selected data from <https://www.kaggle.com/datasets/clementmsika/mubi-sqlite-database-for-movie-lovers?resource=download> for the basis of our project.

From the the abovementioned link, we will be using 2 tables being:

1. Mubi\_movie\_data.csv: Contains data from all movies registered on MUBI.
2. Mubi\_ratings\_data.csv: Contains data from ratings on MUBI for users who did not set their profile in private mode. This table goes back to 2008 and has about 15 million rows.

# Objective

The objective of this analysis is, for the 2019 calendar year:

1. Calculate the average rating per movie based on the mubi\_ratings\_data.csv dataset.
2. Calculate the number of ratings received per movie (this gives an indication of how many times a movie is watched).
3. Determine the top 100 rated movies in the analysis period.
4. Convert the top 100 views list into a database and load into PostgressSQL
5. Review data and record at least 2 observations.

# Record Of Work Completed

## 1. Extract

1. Through the site <https://www.kaggle.com/datasets/clementmsika/mubi-sqlite-database-for-movie-lovers?resource=download>, download the following csv files:
2. Mubi\_movie\_data.csv
3. Mubi\_ratings\_data.csv
4. Load the csv files into Jupyter notebook.

## 2. Transform

1. Cleanse Mubi\_movie\_data.csv data:
   1. Convert floats to integer;
   2. Replace “NaN” to “0”;
   3. Extract the following columns to a new dataframe named “movie\_list\_df”:
      1. “movie\_id”
      2. “movie\_title”
      3. “release\_year”
2. Cleanse Mubi\_ratings\_data.csv data:
   1. Convert the “rating\_timestamp\_utc” column to show the date only;
   2. Filter data to show only ratings made in the 2019 calendar year
   3. Drop the “NaN” values. Note: “NaN” values are dropped as this will skew the average calculations.
3. Apply analysis over the Mubi\_ratings\_data.csv data:
   1. Calculate the average and count values on “rating score” per movie by using group by and aggregate;
   2. Calculate the weighted average rating per movie;
   3. Sort the table by the weighted average descending.
   4. Rename columns from ‘mean’ to ‘avg\_rating’.
   5. Take the first 100 entries into a new table named “rating\_top100\_df”

## 3. Load

1. Create database “Project-ETL\_db” in Postgress PGAdmin.
2. Create 2 tables in the Project-ETL\_db database:
   1. movie\_list:
      1. movie\_id (Primary Key)
      2. movie\_title
      3. release\_year
   2. rating\_top100:
      1. movie\_id (Primary Key, Foreign Key)
      2. avg\_rating
      3. count
      4. weight\_average
3. Load the following dataframes from Jupyter Notebook (refer to Jupyter notebook for code used):
   1. movie\_list\_df; and
   2. rating\_top100\_df.
4. Join table by “movie\_id” column.

# Observations

1. The top 10 highest rated movies (on average) are as follows:
   1. “The Gleaners and I”: 4.58
   2. “The Beaches of Agnès”: 4.56
   3. “Paris, Texas”: 4.55
   4. “Modern Times”: 4.51
   5. “Pixote: The Law of the Weakest”: 4.48
   6. “The Godfather: Part II”: 4.47
   7. “Kirikou and the Sorceress”: 4.47
   8. “Apocalypse Now”: 4.46
   9. “Funeral Parade of Roses”: 4.43
   10. “The White Ribbon”: 4.39
2. Movies released in 2009 appeared the most in the top 100 listing (10 qty).
3. The top 10 most rated movies (indication of most watched movies) are as follows:
   1. “The Gleaners and I”: 1952
   2. “Les mistons”: 1732
   3. “Paris, Texas”: 1704
   4. “Children of Men”: 1672
   5. “Mauvais Sang”: 1652
   6. “They Live”: 1578
   7. “The Seventh Continent”: 1559
   8. “The Beaches of Agnès”: 1556
   9. “Funeral Parade of Roses”: 1515
   10. “Pixote: The Law of the Weakest”: 1434