**ETL Project - Team 1**

**Allison Patnoe, Michael Duszynski**

**Extract**

**Data Sources:**

We chose two datasets that would be interesting to pair together –1) the history of Oscar winners across categories, and 2) the top 1000 movies on IMDB. Using these two data sets together, joined on film title, a user could identify if high IMDB ratings (i.e., fan favorites) translates to winning awards, among other relationships.

* Kaggle - The Oscar Award, 1927 - 2020 *(*[*https://www.kaggle.com/unanimad/the-oscar-award?select=the\_oscar\_award.csv*](https://www.kaggle.com/unanimad/the-oscar-award?select=the_oscar_award.csv)*)*
* Kaggle - IMDB Movies Dataset, Top 1000 Movies ([*https://www.kaggle.com/harshitshankhdhar/imdb-dataset-of-top-1000-movies-and-tv-shows*](https://www.kaggle.com/harshitshankhdhar/imdb-dataset-of-top-1000-movies-and-tv-shows)*)*

**Transform**

* Oscars Dataset
  + Uploaded data from CSV file
  + Created two tables for SQL upload:
    - **oscars\_data**
      1. Filtered data to remove Oscar nominees that did not win (i.e., ‘winner’ = True)
      2. Renamed the ‘name’ column to ‘name\_of\_winner’
      3. Set index to ‘film’ to set film as the primary key for SQL
    - **total\_oscars\_by\_film**
      1. Starting from the dataframe created above, selected only the ‘film’ and ‘category’ columns
      2. Grouped by ‘film’ and created a count of categories won, sorted in descending order to show the most awarded film
* IMDB Dataset
  + Uploaded data from CSV file
  + Created table for SQL upload:
    - **imdb**
      1. Renamed relevant columns to correspond with Oscars dataset
      2. Removed unnecessary rows that were not needed (i.e. Poster Link, Star 1, Star 2, etc.)
      3. Set index to ‘film’ to set film as the primary key for SQL
* Final tables for upload:
  + **oscars\_data**
    - Includes film (title) ***as the primary key***, year\_film (release year), year\_ceremony (year ceremony took place), ceremony (ceremony number, e.g., the 92nd annual), category (e.g., Actor), name\_of\_winner (award recipient), winner (T/F)
  + **total\_oscars\_by\_film**
    - Includes film (title) ***as the primary key***, and categories\_won (number of categories won)
  + **imdb** 
    - Includes film (title) ***as the primary key***, year\_film (release year), Runtime, Genre, IMDB Rating, Meta Score, Number of Votes, Film Gross

**Load**

All tables were loaded into a SQL database – movies\_db. We chose a relational database to set up for additional analysis – joins, merges, etc. We also wanted to have the ability to query multiple tables in one place for more in-depth or specific analyses.

For example, could perform the following queries:

* To get the IMDB ratings for Oscar winners:

SELECT i.film, i."IMDB\_Rating", t.categories\_won

FROM imdb i

JOIN total\_oscars\_by\_film t

ON (i.film = t.film)

* To get the Gross Revenue for Oscar winners:

SELECT i.film, i."Film\_Gross", t.categories\_won

FROM imdb i

JOIN total\_oscars\_by\_film t

ON (i.film = t.film)

Where i."Film\_Gross" IS not NULL

ORDER BY i."Film\_Gross" DESC

By selecting a relational database, there are many options for future analyses!