# **ETL Project Final Report**

## The Team:

Matt Clewis

Nathan Cruz

Clint Jansson

## Extract:

**CSV Source - IMDB data from 2006 to 2016 located on Kaggle.com**

With our ETL project for the first leg we went with an IMDB data set pulled from Kaggle in CSV format. It included the top 1000 most popular movies and the top 4 actors in script. This was given to us in a not so friendly format. The actor column was not broken up and housed all 4 actors per movie.

**Web Source – Wikipedia.org**

We searched Wikipedia.org for notable people from North Carolina. The entry included North Carolinians in media. We scraped the webpage for this list using chromedriver, splinter and beautifulsoup.

## Transform:

**CSV Source - IMDB data from 2006 to 2016 located on Kaggle.com**

Using excel and pandas this was broken up and ordered. Using excel to break up the Actor column on the comma present. Then pulling the CSV into jupyter notebook and the use of pandas started.

* pd.melt, was used to transform and transpose the actors column into one column.
* Then reorganizing the order of the columns took place. After the columns were reorganized ".sort\_value" was used to order the "Title" column. This was to group the 4 actors together in the data set and the only common point of data was the "Title" column.
* After sorting the data, we had to reset the index and drop the newly formed index column due to resetting the original index.
* Once the data was cleaned up and transformed a ".to\_csv" function was used to create a new CSV, named “IMDBACTOR” to be uploaded.

**Web Source – Wikipedia.org**

The list of media personalities from North Carolina scraped from Wikipedia.org was broken into two sections on the website: A to L and M to Z. These were combined and assigned to a list: actors\_list. The list included performers from other professions. The list was filtered by the string “act” in order to capture actors and actresses. This info was then stored in the data frame: actors\_db. The original list was a collection of strings showing names, dates of birth/death, profession(s) and city of birth. For example: Sidney Blackmer (1895-1973), actor (Salisbury). A comma separated name and date of birth/death from the other information. The string was split on the comma and the actors’ names and birth/death years were stored in a new variable: actor\_name. The actors’ year of birth/death was stripped using string replace to leave only the name. The actors’ place of birth was pulled from the actors\_db using string split from the last character of the string until the opening parenthesis in front of the city name. The closing parenthesis was removed using string replace. In a single instance, for the actress Allyn King, a city of birth was not listed. Instead, “actress” was listed. This was removed using string replace. This data was stored in the data frame birth\_city. A new data frame, actor\_info\_db, was created by joining actor\_name and birth\_city. The index was reset and the column names were changed to Actor Name and City of Birth.

## Load:

With the completed datasets above each was loaded into MySQL using the SQL Alchemy package in python. The data loads were directed to 2 tables (imbdactors and wikiactors). The 2 tables were then inner joined to isolate the movies in the database that were the product of North Carolina actors.

The MySQL application was chosen for the ease of loading and joining together of datasets on a singular dimension. The SQLAlchemy was selected as the data was being processed in the Jupyter notebook and it made for easy transfer of data into the MySQL database.

The resultant was 21 matches for Movie Titles from Actors born in NC.

