ETL Project Report

Kim, Hunter, parmis, bita

# Extract

For this project, we decided to utilize open source movie data from Kaggle datasets. The main datasets we chose were:

* [The Movie Dataset](https://www.kaggle.com/rounakbanik/the-movies-dataset/downloads/the-movies-dataset.zip/7) – “Metadata on over 45,000 movies. 26 million ratings from over 270,000 users”
  + Data was downloaded as CSV files, with certain entries being JSON strings (lists of dictionaries). CSV files used were movie\_metadata.csv and keywords.csv.
  + All csv files included a movie id column that we used as our index value
* [The Academy Awards 1927-2015](https://www.kaggle.com/theacademy/academy-awards/downloads/academy-awards.zip/1) – “What actors and films have received the most Oscars?”
  + Data was downloaded as a single CSV file
* The Keywords Dataset – “What keywords is associated with each movie based on the movie ID as a part of the Movie Dataset.
  + Data was downloaded as a part of Movie Dataset.

# Transform

We primarily used PANDAS to clean and transform our datasets. Steps include:

* Loaded movie metadata, Academy Awards and keywords into pandas to create dataframe;
* We came to conclusion that it’s better for us to load our databases in pandas since in MySQL and Mongo the process would be much harder for merging these files.
* Dropped id columns that were loaded incorrectly (had a date string in the id column);
* Renamed Winner column to Result and replaced the nonvalue rows with nominee so we have a common area for merging, also to get rid of the nonvalue rows.
* We used a to\_JSON command in Pandas to transform a dataframe to JSON file format and then transposed them to columns of dataframe.
* Found the common areas to use an inner marge to combine our datasets.
* Even though we created our merged data set in JSON to be able to load it in Mongo, our data did not pull

Correctly and we had to do some trouble shooting with Pandas.

* Added id to AA table.
* Jsonifyied AA table.
* Merged keyword, meta data table.

# Load

the final database, tables/collections, and why this was chosen as well as your final data schema.

* We explored different platforms such as SQL and MangoDB to choose the most efficient platforms to create our schema.
* Finally, we chose Mango since our data had JSONified values which made utilizing SQL inefficient for this dataset.
* To be able to create tables in Mango we had to come up with a way to transform dataframe to Json file in a way that is readable in Mongo to create a nonrelational database.
* PyMongo was one of the libraries that we had to import.
* By using this format col.insert\_one({'type':'table','name': 'ourTable' 'data': dictData}) we thought we could over come the issue, but it still did not work.
* After several attempts, we found out that dictionaries are more relatable to Json files and that finally made this 3step transfer possible. We converted our data to dictionary and then transformed our dictionary to Mongo tables.

## Future WOrk

Create fun game of using keywords to guess movie title

* Kim also made a first attempt to create a challenge to search every movie in our data base with the word “Jealousy” and we found that we have a great find tool in our Mongo table which was pretty cool.