GoT ETL Documentation

# Extract:

We used [www.kaggle.com](https://www.kaggle.com/mylesoneill/game-of-thrones/version/1) to extract our Game of Thrones data. Jupyter notebook was used to load three CSV files: “Battles.csv”, “Character-Deaths.csv”, and “Character-Predictions.csv”.

# Transform:

During the transformation process, various steps were taken to extract the information needed. For the ‘df2\_character\_deaths’ table, we renamed the “Name” column to “name” so that it could be merged with the ‘df3\_character\_predictions’ table. After the merger, we verified that there were no duplicate values between both tables. With our first table, we wanted to display all of the members per each house in Game of Thrones, so in our first table we grouped by the column ‘Allegiance’ and ‘name’, this returned each member that belonged to each house. To make the dataframe transferrable to SQL we had to reset the index of each column. After resetting the indexes we created a new dataframe which is the dataframe that we load into the appropriate SQL table.

For the second table, we had to calculate the number of battles that each house was in and the number of battles that each house won. After finding the number of battles won and the total number of battles each house was a part of we then calculated the win ratio. We created a dataframe that included the name of the house, the total number of battles, and their win ratio.

Finally, for the third table, we wanted to list out the members of each house that remain alive. We had to remove multiple columns from the original dataframe so that we could focus on the data that we wanted. In this dataframe there is a column named ‘isAlive’ with 0 representing dead and 1 representing alive. We removed columns that contained 0. After this data extraction, we had to reset the indexes and rename the columns so that we could load the table into SQL. We then transformed the dataframe. The final dataframe included each characters name and the house that they belong to.

# Load:

In MySQL, we created a script that would create a database named “GoT\_db”, then it would create three empty schemas which would later hold the transformed data from the dataframes.

The first schema is named “members\_Of\_Each\_House” with the following columns: “id INT PRIMARY KEY”, “allegiance TEXT”, and “character\_name TEXT”. The second schema is named “ratios” with the following columns: “id INT PRIMARY KEY”, “allegiance TEXT”, “total\_battles INT”, and “win\_ratio FLOAT”. The final schema is “members\_alive’ with the following columns: “id INT PRIMARY KEY”, “allegiance TEXT”, and “character\_name TEXT”.

Once the above script is run, we use Python in Jupyter to create a direct connection to MySQL and store the three dataframes we transformed into the three schemas we created. The user is asked to enter their MySQL password so that they can access their local server and load the data into the schemas.

# Steps to reproduce ETL

To reproduce the ETL process please see the instructions below:

1. Open MySQL Workbench and open the schema.sql file
2. Run schema.sql in MySQL workbench
   1. This will create the database and the corresponding tables
3. Open the GoT\_ETL.ipynb Jupyter notebook in jupyter
4. Run the code in the notebook
   1. The SQL tables will be loaded with data after running the notebook
5. To view the tables open MySQL Workbench and open the query.sql, this will load the tables with the appropriate data.

# Resources

<https://www.kaggle.com/mylesoneill/game-of-thrones/version/1>