Game Of Thrones Wiki

Nikhil Chaturvedi
2013CS50291

Harman Kumar
2013CS10224

March 15, 2017

1 Overall Design

We present a complete database for the most popular TV series, Game of Thrones. We have over 1000 character stats and prediction about their survival based on popularity, number of dead relatives they have and how good their house is doing.

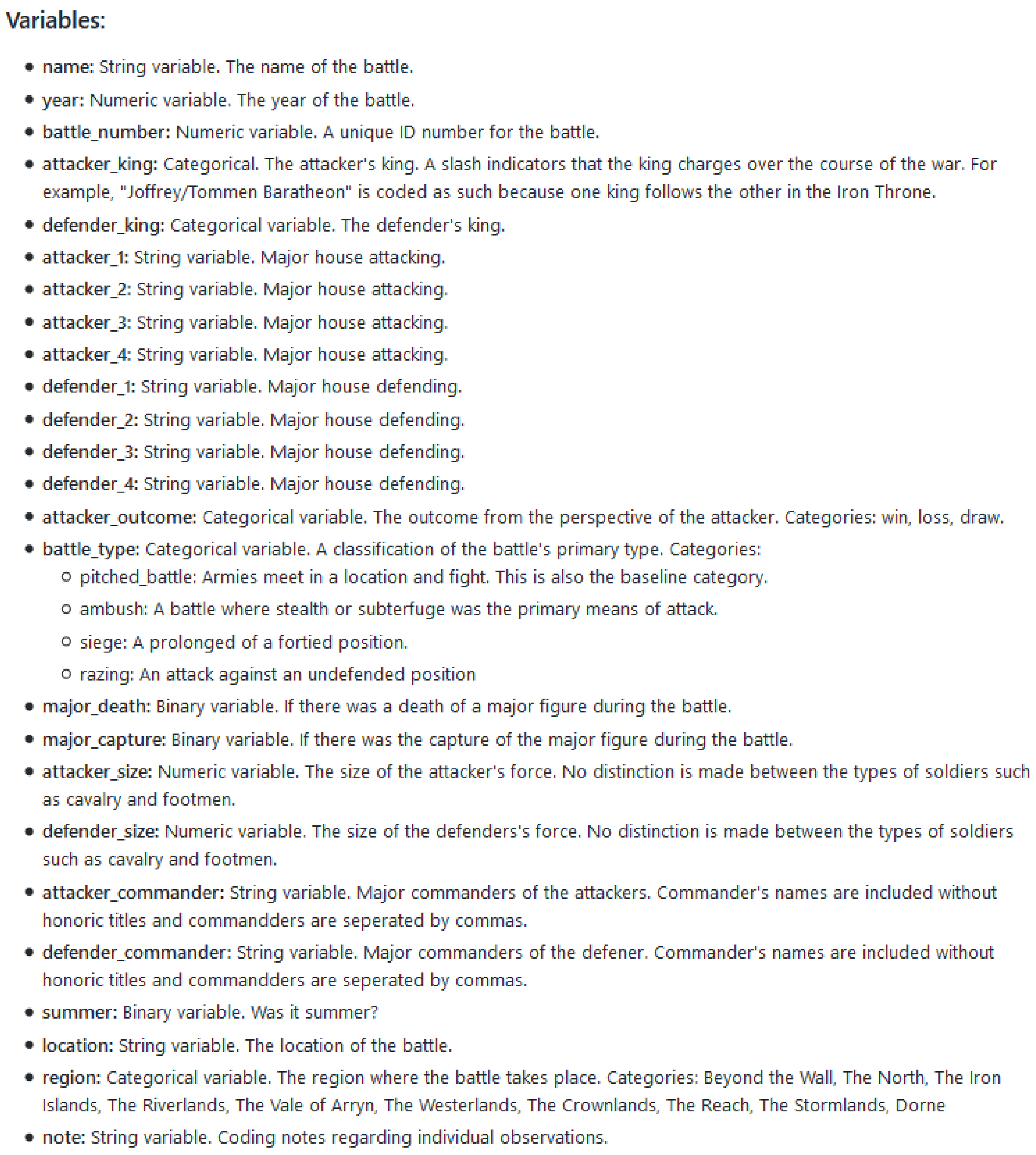
The system allows users to create password secured accounts in order in order to access the GOT Wiki. We also have a separate admin login which allows us to add new information to our databases using a form.
You can find the ER diagram of the project at the end of the document.

2 Data Source

We took our data from three sources:

- A collection of all battles in game of thrones at
  https://github.com/chrisalbon/war_of_the_five_kings_dataset
- Information about characters, when they died and a brief boi about the characters.
  http://allendowney.blogspot.in/2015/03/bayesian-survival-analysis-for-game-of.html
- A collection of all the character predictions. This dataset contains an expanded view on character deaths, including predictions of how likely they are to die. (Prediction's come from A Song of Ice and Data's algorithm)
  https://got.show/

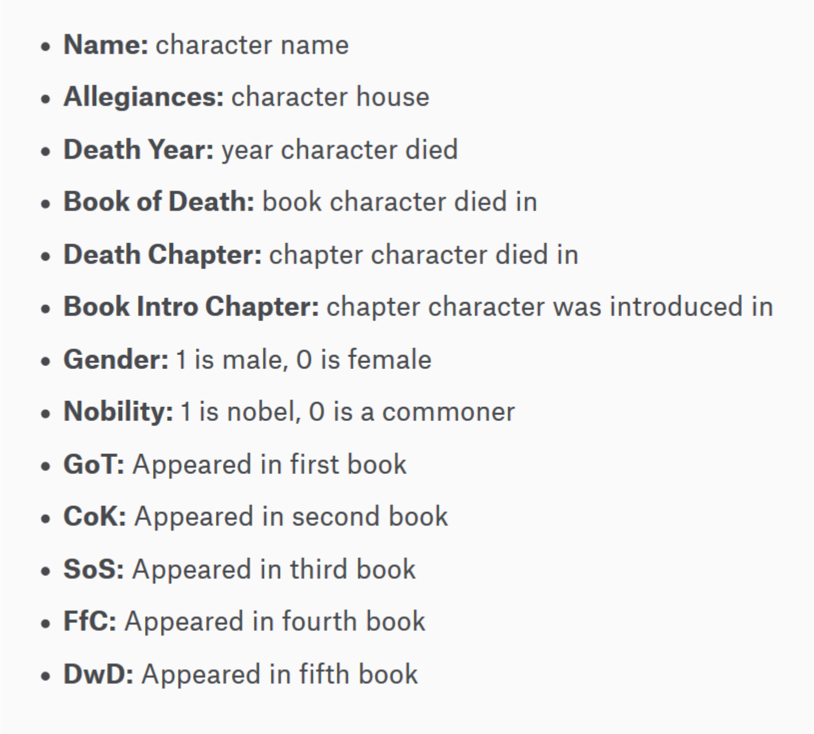
  Following is the schema of the data



Variables:

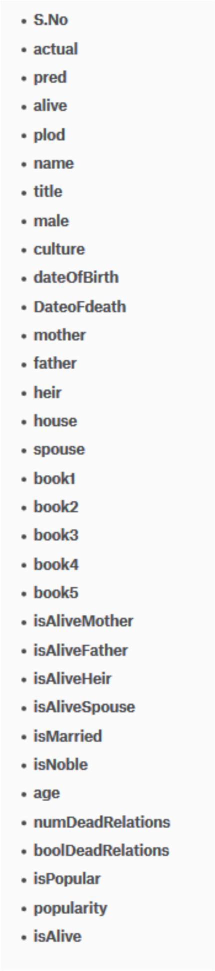
- **name:** String variable. The name of the battle.
- **year:** Numeric variable. The year of the battle.
- **battle_number:** Numeric variable. A unique ID number for the battle.
- **attacker_king:** Categorical. The attacker's king. A slash indicators that the king charges over the course of the war. For example, "Joffrey/Tommen Baratheon" is coded as such because one king follows the other in the Iron Throne.
- **defender_king:** Categorical variable. The defender's king.
- **attacker_1:** String variable. Major house attacking.
- **attacker_2:** String variable. Major house attacking.
- **attacker_3:** String variable. Major house attacking.
- **attacker_4:** String variable. Major house attacking.
- **defender_1:** String variable. Major house defending.
- **defender_2:** String variable. Major house defending.
- **defender_3:** String variable. Major house defending.
- **defender_4:** String variable. Major house defending.
- **attacker_outcome:** Categorical variable. The outcome from the perspective of the attacker. Categories: win, loss, draw.
- **battle_type:** Categorical variable. A classification of the battle's primary type. Categories:
  - pitched_battle: Armies meet in a location and fight. This is also the baseline category.
  - ambush: A battle where stealth or subterfuge was the primary means of attack.
  - siege: A prolonged of a fortied position.
  - razing: An attack against an undefended position
- **major_death:** Binary variable. If there was a death of a major figure during the battle.
- **major_capture:** Binary variable. If there was the capture of the major figure during the battle.
- **attacker_size:** Numeric variable. The size of the attacker's force. No distinction is made between the types of soldiers such as cavalry and footmen.
- **defender_size:** Numeric variable. The size of the defenders's force. No distinction is made between the types of soldiers such as cavalry and footmen.
- **attacker_commander:** String variable. Major commanders of the attackers. Commander's names are included without honoric titles and commandders are seperated by commas.
- **defender_commander:** String variable. Major commanders of the defener. Commander's names are included without honoric titles and commandders are seperated by commas.
- **summer:** Binary variable. Was it summer?
- **location:** String variable. The location of the battle.
- **region:** Categorical variable. The region where the battle takes place. Categories: Beyond the Wall, The North, The Iron Islands, The Riverlands, The Vale of Arryn, The Westerlands, The Crownlands, The Reach, The Stormlands, Dorne
- **note:** String variable. Coding notes regarding individual observations.

Figure 1: Schema of the battles dataset



- **Name:** character name
- **Allegiances:** character house
- **Death Year:** year character died
- **Book of Death:** book character died in
- **Death Chapter:** chapter character died in
- **Book Intro Chapter:** chapter character was introduced in
- **Gender:** 1 is male, 0 is female
- **Nobility:** 1 is nobel, 0 is a commoner
- **GoT:** Appeared in first book
- **CoK:** Appeared in second book
- **SoS:** Appeared in third book
- **FfC:** Appeared in fourth book
- **DwD:** Appeared in fifth book

Figure 2: Schema of the character deaths dataset



- S.No
- actual
- pred
- alive
- plod
- name
- title
- male
- culture
- dateOfBirth
- DateoFdeath
- mother
- father
- heir
- house
- spouse
- book1
- book2
- book3
- book4
- book5
- isAliveMother
- isAliveFather
- isAliveHeir
- isAliveSpouse
- isMarried
- isNoble
- age
- numDeadRelations
- boolDeadRelations
- isPopular
- popularity
- isAlive

Figure 3: Schema of the prediction Dataset

A python script was written to cleanup the data, create the database, specify the schema of tables, various constraints, triggers, do indexing on the data, create virtual and materialized views and load the data into the tables.

3 Functionality and Components

- **View Interesting Characters:**
  Our dataset has a field that species whether a character is interesting or not. Using this the give the user an option to filter out the main characters.
- **Find a character:**
  In our frontend we have the search bar functionality using which we can search for a house or a character. This is a much required functionality as there are over 1000 characters in GOT.
- **View Interesting Characters:**
  Our dataset has a field that species wheteher a character is interesting or not. Using this the give the user an option to filter out the main characters.
- **Sort by Most Likely to Die:**
  Our predictions dataset has a field that predicts the likelihood of a character dying. In order to view spoilers, one can sort the characters by this field and find out which characters would die next.
- **Sort By Popularity:**
  Sort characters by popularity to see the main charcters and then the side characters.