IMDB DATA MANAGEMENT SYSTEM

Viraj Chaudhari
vc6346@rit.edu

Sarang Narkhede
sn7330@rit.edu

Ashwani Kumar
ak8647@rit.edu

Prakash Mishra
pm1739@rit.edu

Devavrat Kalam
dk2792@rit.edu

ABSTRACT

In this report, we have described the tools, implementation details and project objectives which will be performed on the IMDB movies dataset.

Keywords

SQL, IMDB, GUI, Node JS, TSV

1. INTRODUCTION

In 2018, about approximately 50 movies were launched every day. It is not plausible to watch all of them, so people may waste their time watching movies/series which are not worthwhile. Our IMDB movie data management system will assist users in fetching information based on their personal choice.

2. IMPLEMENTATION DETAILS

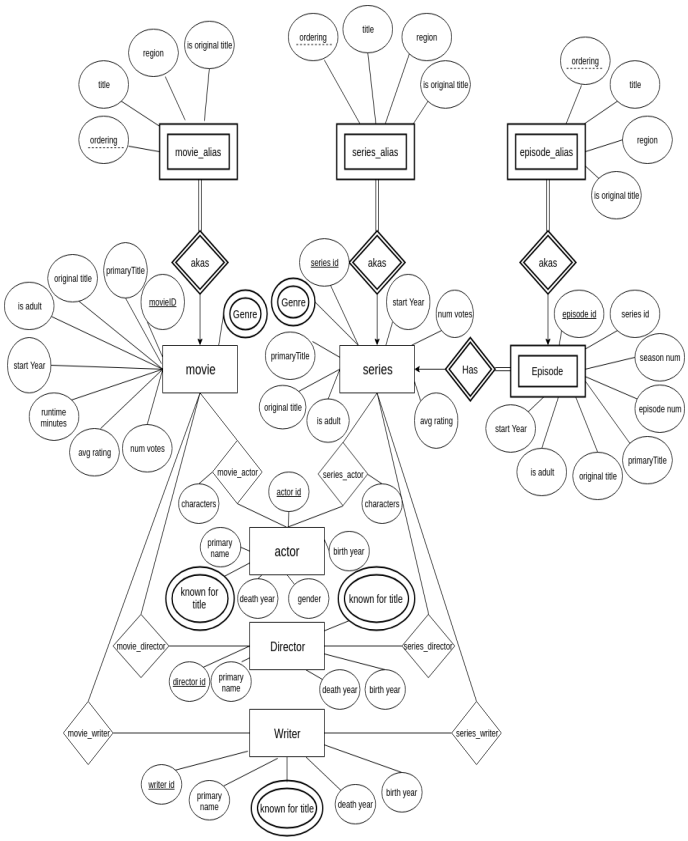
We have used IMDB data set (https://datasets.imdbws.com/), which has the largest collection of movies, TV series, information on cast, crew and other relevant details. We have built a single data retrieving page, which will parse through the data set to display, depending on the query, the corresponding result.

2.1 Dataset Implementation

The IMDB dataset had 7 tsv files -

- ***title.akas.tsv*** - Contains regional information of a particular title(which could be a movie, short movie, series, video, video game etc.)
- ***title.basics.tsv*** - Contains basic information including primary title name, original title name, genre, runtime duration, isadult, start year, end year information.
- ***title.crew.tsv*** - Contains the director and writer information for all the titles in IMDb
- ***title.episode.tsv*** - Contains the tv episode information.
- ***title.principals.tsv*** - Contains the principal cast/crew for titles i.e. maps a title to a person(which can be director, actor, cinematographer etc) involved in the title.
- ***title.ratings.tsv*** - Contains the IMDB rating and votes information for titles.
- ***name.basics.tsv*** - Contains the information for people involved in a title along with their top 3 profession and famous titles in which they have worked.

After doing careful analysis, we found inter-dependencies among the data in these dataset and came up with the below ER diagram -



We first started with the segregation of the entities from the original raw data present in the 7 datasets. For this, we

used python's pandas library and created entities from each dataset as follows :

- ***Title.basics.tsv*** - We first observed the type of titles present in this dataset and found some redundancies. There were 10 distinct values present in the title. They are as follows:

```
tvEpisode       3953453
short            671435
movie            514140
video            225403
tvSeries         161535
tvMovie          126081
tvMiniSeries      25283
videoGame         23157
tvSpecial         16843
tvShort            9111
Name: titleType, dtype: int64
```

  We clubbed these types together to remove data anomalies. For TV series and TV miniseries, which contains information of series. So, we clubbed them together under Series entity set using pandas. For tvshort, short, movie, TV movie are clubbed into an entity set named Movie. Lastly, we created an entity set for Episode for this dataset and dropped records related to video, video game and TV special. After this, we removed end year column from all of these entities as majority of them had null values in it. For Series and Episode, we dropped run-time minute attribute.

  After all this, we used title.ratings.tsv, which contained rating and votes count information of the titles, and merged these with movie, series and episode entities using merge function of pandas.

  For handling the multi-valued attribute genre feature, we split the cells with the delimiter "," and we normalized it into 1NF.

- ***Title.akas.tsv*** - This data set contains aliases of a title according to different regions and language. However, there was no way of knowing to which entity set a particular record in this data set belonged to. So using Title.basics.tsv, we merged it with the Title.akas.tsv so that we could get title type(like movie, series, episode etc) information of a particular title. Then we grouped this dataset according to title type and segregated the records into 3 entities - moviealias (containing akas info for movies), seriesalias (containing akas info for series) and episodealias (containing akas info for episodes).

- ***Title.crew.tsv*** - This dataset links writers and directors to a particular title. Both of these attributes are multi-valued. First we got the title type information from movie, series and episode entity and then, we split these attributes individually and created relationships viz. MovieDirectorRelation, MovieWriterRelation, SeriesDirectorRelation, SeriesWriterRelation and EpisodeDirectorRelation, EpisodeWriterRelation.

- ***Title.episode.tsv*** - This dataset contains information like season number and episode number of a particular episode. We already had episode information from title.basics.tsv data set. However, this data set was giving additional information of an episode, so we merged these 2 data sets using pandas merge function.

- ***Name.basics.tsv*** - This data set contains information of people and their profession in the movie. It also has multi value column named primary profession, we converted this dataset into 1NF by splitting multi valued cells into equivalent number of rows. Then, after that, we were able to get count occurrences of each profession.

- ***Title.principals.tsv*** - This dataset contains the principal cast/crew for titles i.e. it maps a title to a person(which can be director, actor, cinematographer etc) involved in the title. We first merged this dataset with datasets we created containing Movie, Series and Episode information which we got by removing anomalies. Then, we segregated this new dataset according to the title type(movie, series and episode) and got 3 datasets individually containing principal information of movies, series and episodes. Then, we worked on each dataset individually to get the relationship details amongst director, actor, writer and movies, series, episodes entities.

  Similarly, we found out relations between director, actor, writer and series.

2.2 Website Implementation

We created a website using Node JS and database as My SQL to provide a GUI interface for user to fetch results as per query. We have three pages associated with our website:

- ***WelcomePage*** - Here we have a search bar and a background page with mp4 video playing in background.

- ***AdvancedSearches*** - Here user can enter the queries based on Movie Name, Actor Name, Director Name, Writer Name and Series and get the complete results associated with that particular query.

- ***Resultspage*** - This Page will display all the results based on the query from the advanced search page.

3. CONCLUSION

We are successfully able to fetch the data from IMDB data set by importing data from tsv files into our MySQL database. We segregated the data further by redefining the relationships among entities to create a structured data which can fetch us useful results without data anomalies. We also successfully created a website using Node Js and MySQL database to provide a GUI at user end for fetching the results based on query.