

# **PROJECT PROPOSAL REPORT**

## **“EmojiQL”**

**DSCI-551**

By Aaryan Akshay Shah,  
USC ID: 5532714539

# **TABLE OF CONTENTS**

- DATASET
- DATA MODEL DESIGN
- DESIGN AND IMPLEMENTATION OF CUSTOM QUERY
- PROGRAMMING LANGUAGE TO BE USED
- TESTING AND VALIDATION:
- RESULTS AND CONCLUSIONS
- GROUP MEMBERS

## DATASET

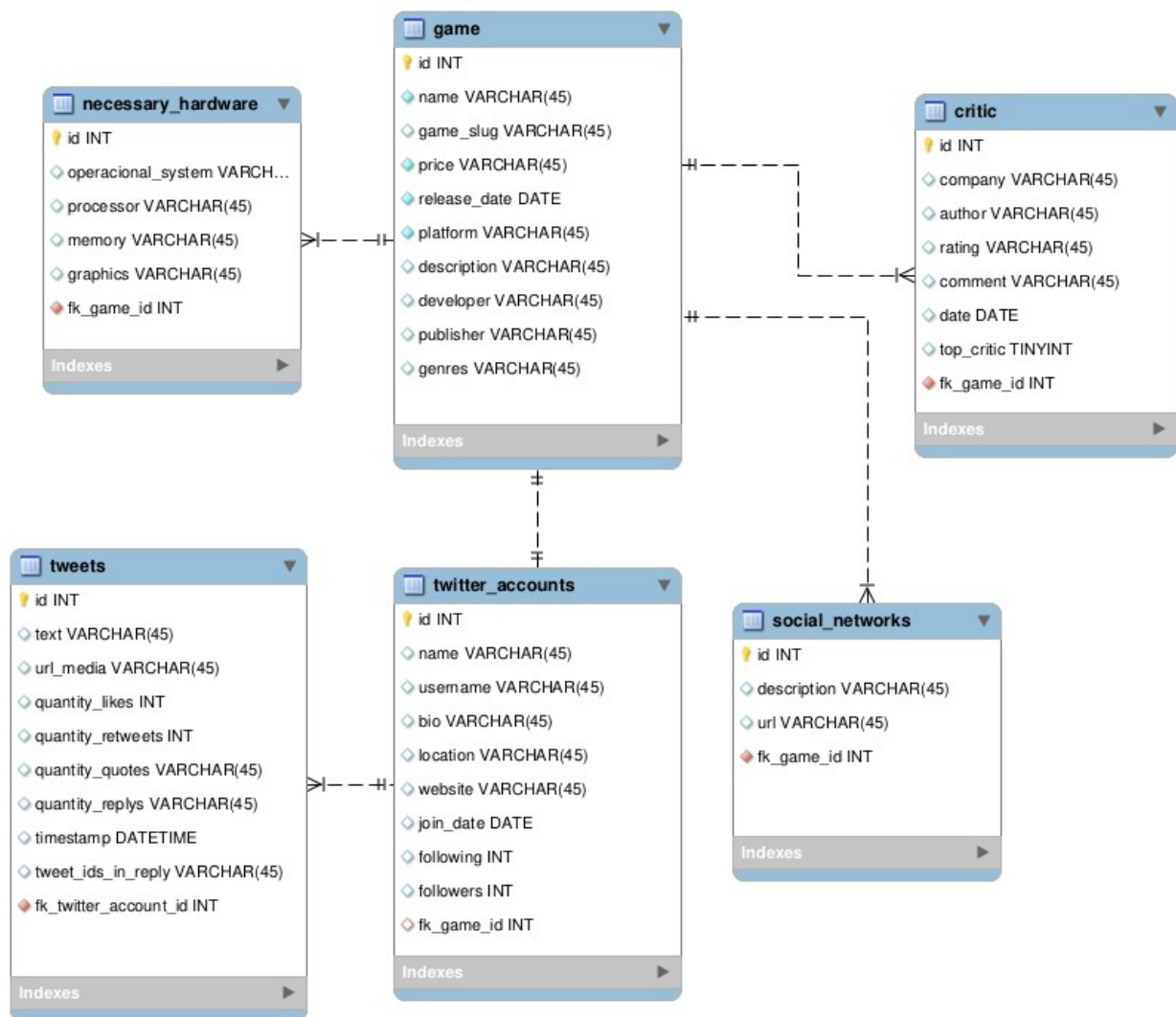
I am going to use the 'Epic Games Store dataset' obtained from kaggle.

[Kaggle](#):

Year: 2022

The relational dataset contains 6 csv(comma separated value) files

Following is the ERM (entity relationship model) for the dataset:



The relational dataset contains 6 csv(comma separated value) files/tables and their details are as follows:

### **TABLES:**

#### **1. Games table ("games.csv"):**

Description of games file columns:

- ID: identification of game.
- Name: name of game.
- Game slug: short name of game.
- Price: price of game.
- Release date: release date of game.

- Platform: platforms that the game is available.
- Description: description of game.
- Developer: company that developed the game.
- Publisher: company that published the game.
- Genres: genres of game

## **2. Necessary Hardware table ("necessary\_hardware.csv"):**

All the games has a minimum and recommended necessary hardware.

Description of necessary hardware file columns:

- ID: identification of necessary hardware.
- Operational system: operational system the game supports.
- Processor: processor minimum or recommended that game needs to run.
- Memory: RAM memory minimum or recommended that game needs to run.
- Graphics: graphics minimum or recommended that game needs to run.
- Game ID: identification of game.

## **3. Social networks table ("social\_networks.csv"):**

Description of social networks file columns:

- ID: identification of social network.
- Description: name of social network (such as "linkTwitter").
- URL: URL of social network of the game.
- Game ID: identification of game.

## **4. Twitter accounts table ("twitter\_accounts.csv"):**

Description of twitter accounts file columns:

- ID: anonymized identification of twitter account.
- Name: profile name of twitter account.
- Username: username of twitter account.
- Bio: biography of twitter account, such as a little description.
- Location: location of twitter account owner.
- Website: in case that user has a website, contain the link.
- Joined date: date that user created your twitter account.
- Following: quantity of users that profile is following.
- Followers: quantity of users that follow the profile.
- Game ID: identification of game.

## **5. Tweets table ("tweets.csv"):**

Description of tweets file columns:

- ID: anonymized identification of tweet.
- Text: text of tweet.
- URL Media: image or video link of tweet.
- Quantity likes: quantity likes of tweet.
- Quantity retweets: quantity retweets of tweet.
- Quantity quotes: quantity quotes of tweet.
- Quantity replies: quantity replies of tweet.
- In Reply To User ID: user id that posted parent tweet.
- Timestamp: date and hour that tweet was posted.

- Twitter Account ID: anonymized identification of twitter account.

## 6. Open Critic table ("open\_critic.csv")

Description of open critic file columns:

- ID: identification of critic.
- Rating: rating of game (example: 80).
- Comment: author comment about the game.
- Company: company name that rated the game.
- Author: author name that rated the game.
- Date: date of critic.
- Description: description of game.
- Top critic: verify if is a top critic (authors with verdict).
- Game ID: identification of game.







The relationship between the tables, primary key and foreign key will be implemented according to the ER diagram.

## DATA MODEL DESIGN

*(refer ERM diagram above)*

I will be using a relational data model just like MySQL in this project. Dataset is in csv format and hence I have planned to store it in the text file as of now to operate the data. (may change)

PC > New Volume (D:) > dataset > Epic Games

| Name   | Date modified    | Type                 | Size       |
|--|------------------|----------------------|------------|
|  games              | 9/6/2023 8:31 AM | Microsoft Excel C... | 329 KB     |
|  necessary_hardware | 9/6/2023 8:31 AM | Microsoft Excel C... | 306 KB     |
|  open_critic        | 9/6/2023 8:31 AM | Microsoft Excel C... | 7,144 KB   |
|  social_networks    | 9/6/2023 8:31 AM | Microsoft Excel C... | 264 KB     |
|  tweets             | 9/6/2023 8:31 AM | Microsoft Excel C... | 181,880 KB |
|  twitter_accounts   | 9/6/2023 8:31 AM | Microsoft Excel C... | 140 KB     |

The will be stored just like above where all the txt files will be present in a file/folder. Data will be stored using a relational database system, ensuring efficient storage and retrieval.

## DESIGN AND IMPLEMENTATION OF CUSTOM QUERY

Emoji representation - A set of emojis will be chosen to represent database operations.

Creating an Emoji SQL database with MySQL keywords represented by emojis can be a playful and unique project. Here are some MySQL keywords and their corresponding emojis:

SELECT: 🔍  
 FROM: 📁  
 WHERE: 🎯  
 INSERT INTO: ➕  
 VALUES: 📝  
 UPDATE: 🔄  
 SET: 🛠️  
 DELETE: 🗑️

CREATE TABLE: 🏗️  
 ALTER TABLE: 🔧  
 DROP TABLE: ❌  
 PRIMARY KEY: 🔑  
 FOREIGN KEY: ➡️🔑  
 INNER JOIN: 🤝  
 LEFT JOIN: 👉🤝  
 RIGHT JOIN: 🤝👉  
 OUTER JOIN: 👉🤝👉  
 GROUP BY: 🔍👥  
 HAVING: 🕵️  
 ORDER BY: 📄  
 ASC: ⬆️  
 DESC: ⬇️  
 LIMIT: 📏  
 OFFSET: ⏭️

In a similar way, we can even map other emojis for various other keywords.

Some of the sample emoji query can be

- 🔍 📁 **games** *(Select all games)*
- 🔍 📁 **name, rating** 🎯 **rating > 4.5** *(Select game names and ratings for games with ratings greater than 4.5)*
- 🔄 📁 **games** 🔧 **rating = 4.9** 🎯 **name = 'Specific Game'** *(Update the rating of a specific game)*
- 🔍 📁 \* 📄 ⬇️ **release\_date** *(Retrieve games sorted by release date in descending order)*

A mapping between the chosen emojis and the corresponding database operations will be established.

EmojiQL Parser - A parser will be developed to interpret and translate sequences of emojis into corresponding database operations.

The logic for processing and executing queries based on the parsed emojis will be implemented using python. Creating a Command Line Interface (CLI) for your EmojiQL project using Python can be done using libraries like argparse or click.

## PROGRAMMING LANGUAGE TO BE USED

Python will be used as the backend language to develop execution logic supporting common operations. It will also be used to provide an interactive command line interface for user interaction.

Tentative libraries that will be used- (other libraries may or may not be added)

- emoji
- (a library for parser)
- argparse or click (for building the interactive CLI)
- pandas
- unittest or pytest

Pandas library will be majorly used to manage the data in the txt file. It will be used to implement various queries which will be asked to perform by the user in the backend.

## **TESTING AND VALIDATION:**

### **Unit Testing**

Unit tests will be conducted to validate the functionality of individual components.

### **Integration Testing**

Integration tests will be performed to ensure all components work together seamlessly.

## **RESULTS AND CONCLUSIONS**

### **Achievements**

The EmojiQL project aims to deliver a unique and user-friendly approach to querying the Epic Games Store dataset. The accomplishments will be summarised in further reports.

### **Limitations and Future Improvements**

Any limitations encountered during the project will be discussed, and potential areas for future enhancements will be suggested.

### **Conclusion**

The EmojiQL project represents an innovative and engaging way to interact with relational datasets. By utilising emojis as a query language, it aims to make database systems more accessible and enjoyable for a wider audience. Our other objective is to learn to store, manage, and manipulate the data using custom query and without any help of the already existing database model.

## **GROUP MEMBERS**

**Name :** Aaryan Akshay Shah

**USC ID:** 5532714539

**Undergraduate in:** Btech in Electronics and telecommunication

**Skills:** Python (numpy, pandas, Matplotlib, Seaborn, Scikit-learn, TensorFlow, OpenCV, Pickle), MySQL, MongoDB, Firebase,

**Responsibilities:** Complete and execute what I claim before the submission deadline.