UNIVERSITY OF SOUTHERN CALIFORNIA

Final Report

“EmojiQL”

By Aaryan Akshay Shah
USC ID: 5532714539

Drive Link:
https://drive.google.com/drive/folders/1fiW3_ydTDIZW1JlhXDLCzPX0aCk8Sby-?usp=sharing

Date: 12/08/2023

TABLE OF CONTENTS

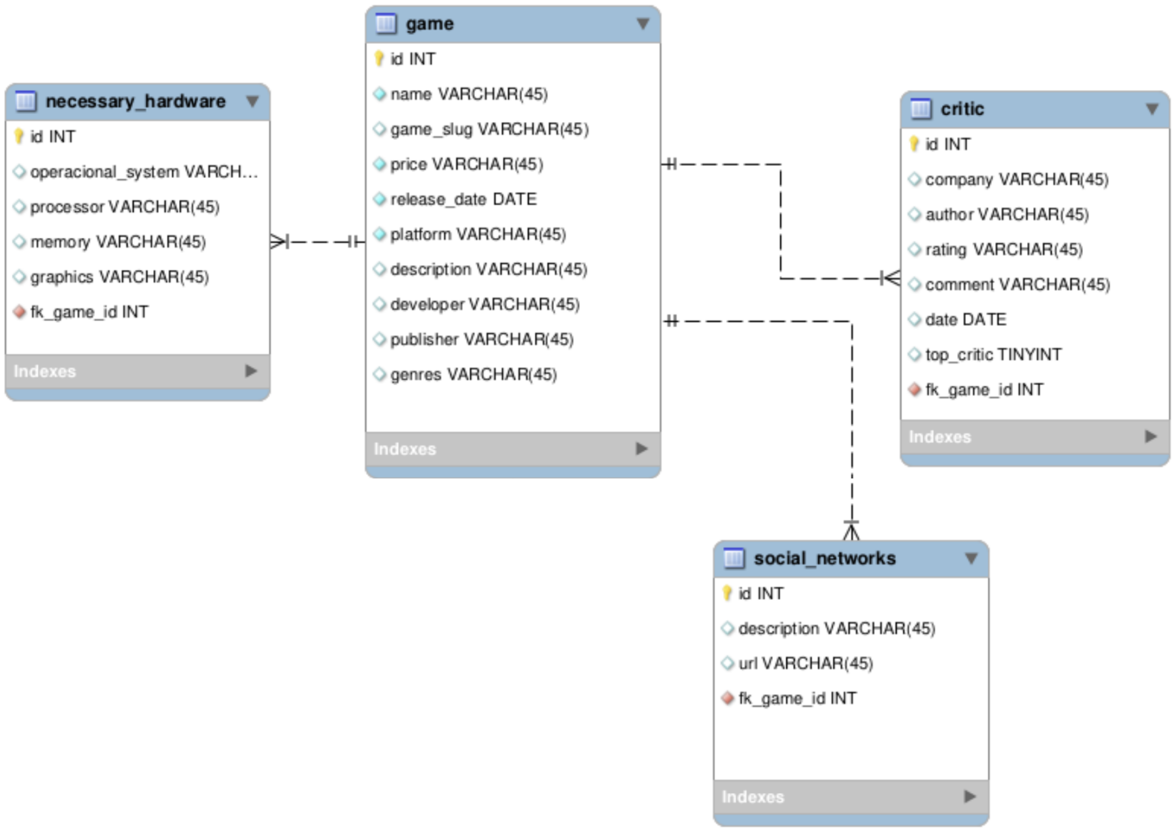
INTRODUCTION

- **Overview of EmojiQL:**

The EmojiQL project represents a novel approach to interacting with databases, aiming to simplify the process of constructing and executing SQL queries. It's a user-friendly interface that leverages the universal and intuitive nature of emojis to formulate SQL commands. This initiative addresses the often-intimidating appearance and complexity of traditional SQL syntax, making database queries more accessible and engaging for a broader audience, including those without extensive technical backgrounds in database management.

The core functionality of EmojiQL revolves around translating emoji-based inputs into standard SQL queries. For instance, users can construct queries using familiar emojis instead of typing out SQL syntax. This innovative method not only adds an element of fun to database interactions but also significantly lowers the entry barrier for beginners and non-technical users.

Dataset used for this project is EPIC GAMES DATASET extracted from Kaggle - link. A few redundant tables are deleted from the database to make the implementation simpler.



- **Motivation:**

The primary motivation behind EmojiQL is to demystify and simplify the process of database querying. SQL, while powerful, often presents a steep learning curve, especially for non-technical individuals or those new to database management. This complexity can hinder effective data manipulation and analysis, which are crucial in today's data-driven environment.

Moreover, the use of emojis as a medium for constructing queries introduces a visual and intuitive aspect to database interaction. Emojis are universally recognizable and can transcend language barriers, making them an excellent tool for visual communication. By harnessing their simplicity and universal appeal, EmojiQL aims to make SQL queries more approachable and less daunting.

- **Objectives (Implemented)**

1. **Simplify SQL Syntax:** Translate traditional SQL commands into emoji-based inputs to reduce complexity and enhance user-friendliness.

2. **Accessibility and Inclusivity:** Enhance database interaction accessibility, making it approachable for users with diverse backgrounds, including those with limited SQL knowledge.

3. **Educational Utility:** Utilize EmojiQL as a tool to teach database concepts and SQL in an engaging, less intimidating manner.

4. **Efficient Data Handling:**
    a. Implement data chunking to manage large datasets by processing smaller, manageable data segments.
    b. Use the glob module for efficient file path retrieval and management of chunked data files.
    c. Employ joblib for parallel processing, improving performance for data-intensive operations.

5. **Interactive GUI Development:** Develop a graphical user interface that allows users to construct queries using emoji buttons, enhancing interactivity and visual appeal.

6. **Comprehensive SQL Functionality:** Ensure crucial SQL operations (SELECT, JOIN, WHERE, ORDER BY, etc.) are effectively represented and executed via emoji-based inputs.

7. **Performance Optimization:** Leverage chunking and parallel processing to ensure EmojiQL's effectiveness and efficiency, even with extensive data.

8. **Scalability and Robustness:** Build EmojiQL to handle varying sizes of datasets and complexities of queries, ensuring scalability and robustness in real-world applications.

PLANNED IMPLEMENTATION

The planned implementation of the EmojiQL project, as outlined in your project proposal, involves developing an innovative system to simplify SQL queries using emojis. The key elements of the implementation are as follows:

- Use of Epic Games Store Dataset: The project utilizes a dataset from the Epic Games Store, comprising six CSV files, to demonstrate the functionality of EmojiQL.

- Relational Data Model Development: A relational data model will be constructed to effectively organize and relate the data within the dataset.

- EmojiQL Parser Creation: The project plans to develop an EmojiQL parser. This parser will interpret sequences of emojis and translate them into standard SQL queries, making database interactions more intuitive and less intimidating.

- Python for Backend Development: Python will be employed for backend processing, leveraging its robust libraries and frameworks to handle database operations and data manipulation.

- Efficient Data Handling through Chunking: To manage large datasets effectively, the project will implement data chunking techniques. This approach will divide large datasets into smaller, manageable pieces, allowing for more efficient processing and memory usage.

- Interactive Command Line Interface (CLI): An interactive CLI will be developed to facilitate user interaction with the database using emoji-based inputs. This interface aims to make SQL querying more accessible, especially for users with limited technical knowledge or experience with SQL.

- Accessibility and Engagement: The overarching goal of the project is to make database querying more accessible and engaging through the use of emojis. By replacing complex SQL syntax with intuitive emojis, the project seeks to lower the barrier to effective database querying and broaden the user base for database interaction tools.

This summary highlights the innovative approach of the EmojiQL project in transforming the way users interact with databases, making it a more user-friendly and accessible tool for a wide range of users.

- **Modification since the proposal**

Since the initial proposal of the EmojiQL project, notable modifications have been made to enhance its functionality and usability:

Optimized Data Handling: The project has advanced the data chunking process to handle large datasets more efficiently, reducing memory load and processing time.

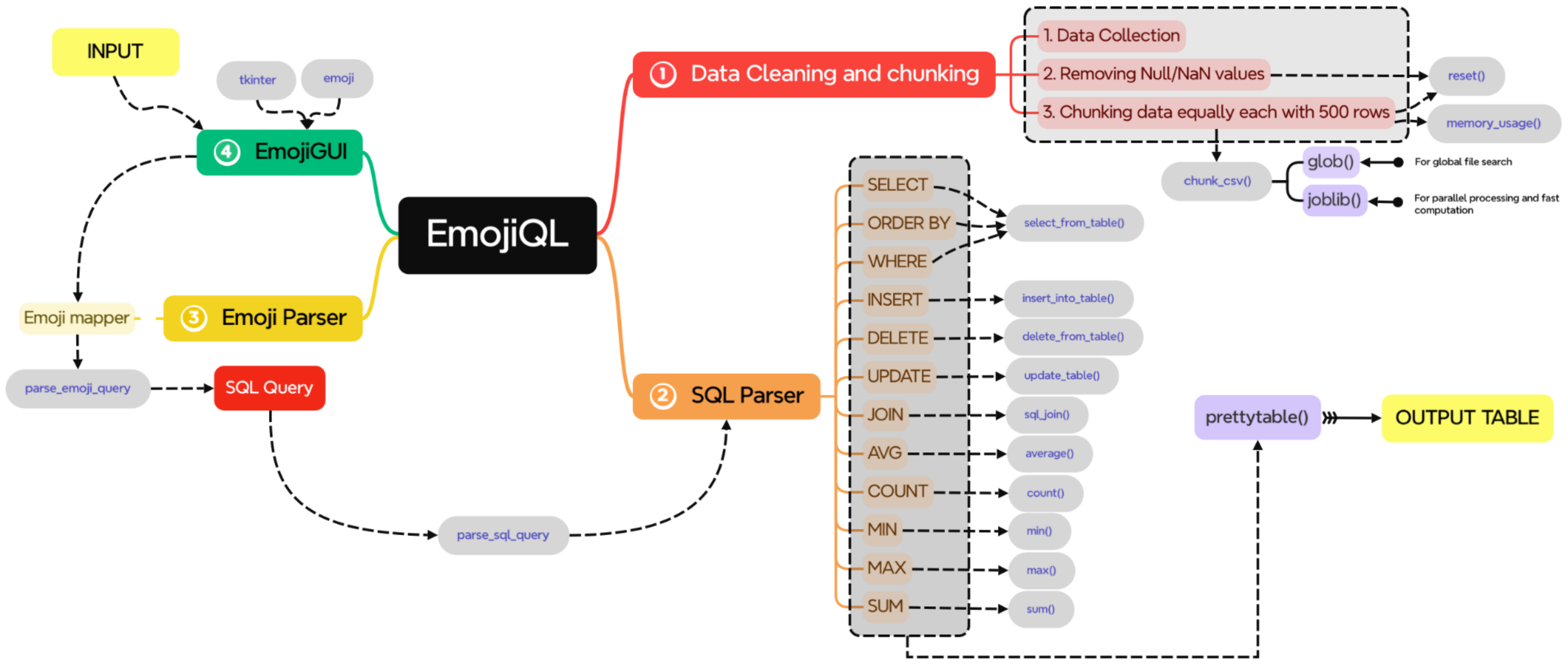
Refined EmojiQL Parser: Improvements have been made to the parser for more accurate and reliable translation of emoji inputs into SQL queries, enhancing the system's robustness.

Enhanced User Interface: The user interface has undergone improvements for better user interaction, focusing on ease of use and intuitive design. A GUI has been implemented for better interaction.

Expanded SQL Functionality: The range of SQL functionalities supported by EmojiQL has been broadened, ensuring a more comprehensive database interaction experience.

These modifications aim to further align the EmojiQL project with its core objective of making SQL querying accessible and engaging through an innovative, emoji-based approach.

ARCHITECTURE DESIGN



The provided flow diagram presents a comprehensive view of the EmojiQL project's architecture, illustrating the various components and their interactions. Here is a brief description of each component and its role within the system:

1. **Data Cleaning and Chunking:**
    a. Data Collection: The process starts with collecting the dataset, which in this case, is the Epic Games Store data.
    b. Removing Null/NaN values: The dataset undergoes a cleaning process to remove any null or 'Not a Number' (NaN) values, ensuring data quality.
    c. Chunking data equally with 500 rows: The cleaned data is then chunked into smaller datasets, each with 500 rows, to manage memory usage efficiently and facilitate parallel processing.
2. **SQL Parser:**
   The SQL Parser is the core engine that translates emoji-based queries into SQL syntax. It supports a variety of SQL operations such as SELECT, ORDER BY, WHERE, INSERT, DELETE, UPDATE, JOIN, and aggregation functions like AVG, COUNT, MIN, MAX, and SUM.
3. **Emoji Parser:**
   The Emoji Parser serves as an intermediary between the user interface and the SQL Parser. It receives emoji-based input from the user and maps each emoji to the corresponding SQL operation using an Emoji mapper.
4. **EmojiGUI:**
   The EmojiGUI is the graphical user interface that users interact with. It utilizes the Tkinter library for the GUI components and the emoji library to support emoji functionalities.
5. **Output Table:**
   The Output Table displays the results of the SQL queries. It is formatted using the PrettyTable library, which provides a way to output data in a table format that is easy to read and understand.
6. The diagram also highlights the use of additional libraries such as **glob** for file search and **joblib** for parallel processing, which optimize the handling and computation of data. The output is directed to the Output Table, where the end results of the user's queries are displayed clearly, allowing for easy interpretation and analysis.

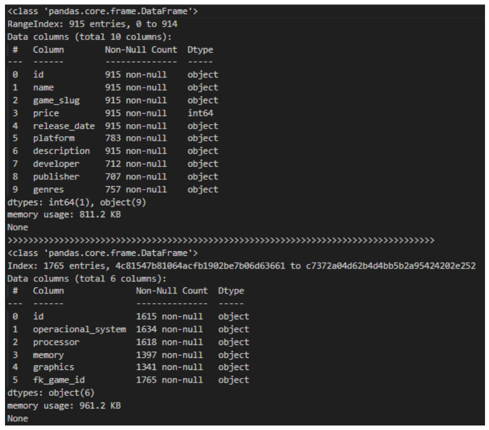
This flow diagram effectively communicates the logical sequence of operations from data intake to the presentation of query results, reflecting the system's capacity to translate user-friendly emoji inputs into complex SQL operations in a streamlined and efficient manner.

IMPLEMENTATION

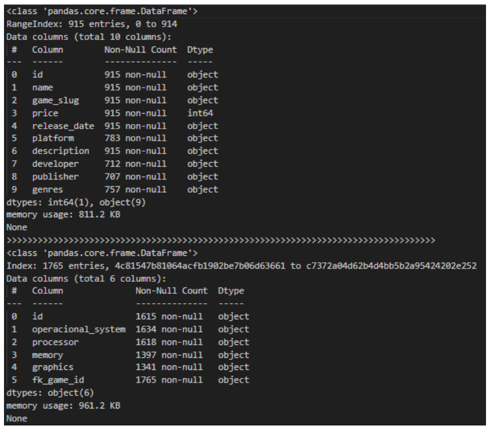
A. Functionalities:

The EmojiQL project offers a range of functionalities designed to simplify and enhance the SQL querying experience ranging from memory management to parallel processing:

a. <u>Memory Management and data chunking</u>:
Processing and loading datasets is always time consuming and inefficient when the datasets are large enough in terms of memory. So we first check the memory usage of each raw file and will break each raw csv file into smaller chunks each with 500 rows. Following this process makes the memory management efficient and faster. It also helps in accessing the data in chunks efficiently.



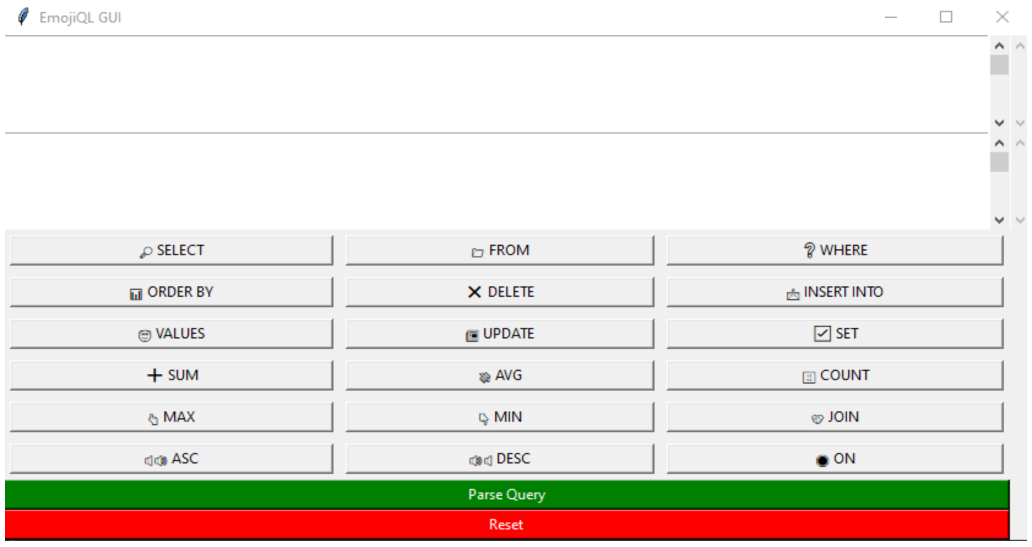
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 915 entries, 0 to 914
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   id            915 non-null    object
 1   name          915 non-null    object
 2   game_slug     915 non-null    object
 3   price         915 non-null    int64
 4   release_date  915 non-null    object
 5   platform      783 non-null    object
 6   description   915 non-null    object
 7   developer     712 non-null    object
 8   publisher     707 non-null    object
 9   genres        757 non-null    object
dtypes: int64(1), object(9)
memory usage: 811.2 KB
None
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
<class 'pandas.core.frame.DataFrame'>
Index: 1765 entries, 4c81547b81064acfb1902be7b06d63661 to c7372a04d62b4d4bb5b2a95424202e252
Data columns (total 6 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   id                  1615 non-null   object
 1   operacional_system  1634 non-null   object
 2   processor           1618 non-null   object
 3   memory              1397 non-null   object
 4   graphics            1341 non-null   object
 5   fk_game_id          1765 non-null   object
dtypes: object(6)
memory usage: 961.2 KB
None
```



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 915 entries, 0 to 914
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   id            915 non-null    object
 1   name          915 non-null    object
 2   game_slug     915 non-null    object
 3   price         915 non-null    int64
 4   release_date  915 non-null    object
 5   platform      783 non-null    object
 6   description   915 non-null    object
 7   developer     712 non-null    object
 8   publisher     707 non-null    object
 9   genres        757 non-null    object
dtypes: int64(1), object(9)
memory usage: 811.2 KB
None
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
<class 'pandas.core.frame.DataFrame'>
Index: 1765 entries, 4c81547b81064acfb1902be7b06d63661 to c7372a04d62b4d4bb5b2a95424202e252
Data columns (total 6 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   id                  1615 non-null   object
 1   operacional_system  1634 non-null   object
 2   processor           1618 non-null   object
 3   memory              1397 non-null   object
 4   graphics            1341 non-null   object
 5   fk_game_id          1765 non-null   object
dtypes: object(6)
memory usage: 961.2 KB
None
```

Memory usage of each RAW file is shown in the above images. A function called chunk_csv is created to custom chunk the large files.

b. <u>Interactive User Interface</u>: Provides a user-friendly interface using tkinter allowing for easy input and visualization of emoji-based queries.



c. **Parallel Data Processing**:
To parallelly process the data, the **joblib()** module is used in python. Joblib is optimized to be fast and robust on large data in particular. It basically easens the long running times.
- Avoid computing the same thing twice: code is often rerun again and again, for instance when prototyping computational-heavy jobs.
- Transparent and fast disk-caching of output value: save the computation to disk and rerun it only if necessary.

Furthermore, glob module is used, which is short for global, is a function that's used to search for files that match a specific file pattern or name. It can be used to search CSV files. This helps in checking each and every chunk file of each table which in turn makes sure that no data is left from any of the chunks created.

d. **Database Interaction and Management:**

Supports interactions with database systems for executing translated SQL queries and managing data retrieval and updates.

e. **Error Handling and Validation:**

Includes error handling mechanisms (such as user can use '*' as well as 'all' to select all the columns) to ensure the robustness of query translation and execution.

f. **Metadata handling:**
Extracted the metadata of each column of a particular table to access and parse the information that is a string in a csv file but needs to be parsed as an integer.

B. Tech Stack (tools and environment used)

1. Programming Languages:
Python: Chosen for its extensive libraries and ease of use, Python was the primary language used for backend development, including the Emoji Parser and database interaction.

2. Libraries and Frameworks:

   a. joblib: Utilized for parallel processing to enhance the efficiency of data operations.
   b. glob: Employed for file management, especially handling chunked data files.
   c. csv module: Used for reading and writing CSV files, particularly for chunked data handling.

    d. tabulate and prettytable: this libraries were used for printing the tables in a column-row format.
    e. Numpy and pandas: used just for data cleaning.

3. <u>Development Environment:</u>

    - <u>Integrated Development Environment (IDE):</u> Tools like PyCharm or Visual Studio Code were used to facilitate coding, debugging, and version control.
    - <u>Version Control</u>: Git, along with platforms like GitHub, was used for source code management, facilitating collaboration and version tracking.

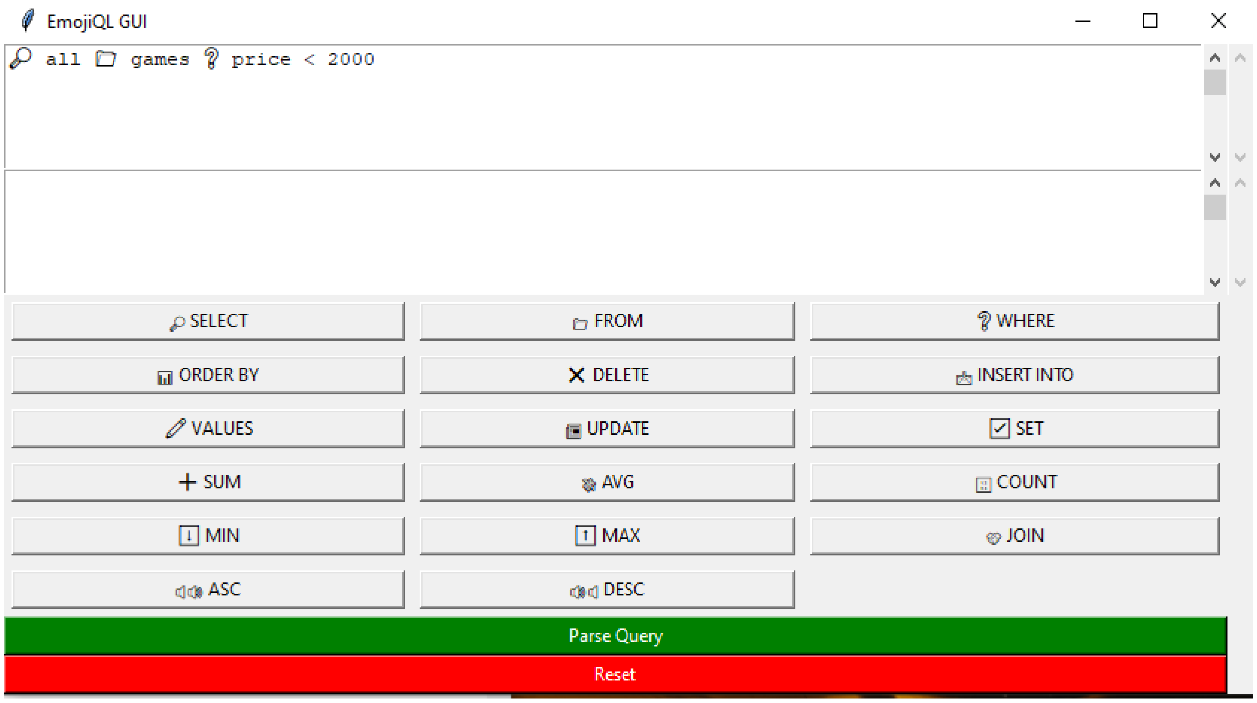
4. <u>Database:</u>
   Storing of the chunked csv is done using csv module and is stored in the csv format.

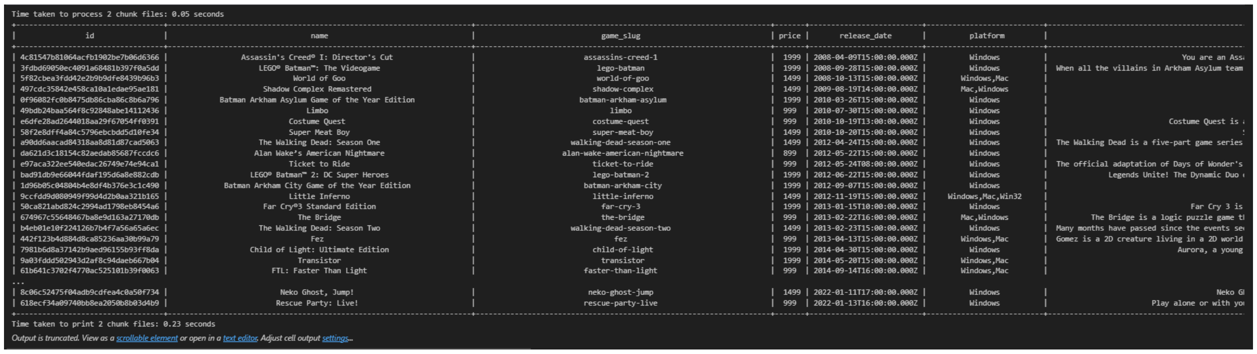
5. <u>User Interface Development:</u>
   GUI tools: tkinter is the library which is used in the project to create a graphical user interface which contains a text box, buttons representing emojis, Parser button and reset button.

C. Implementation of different queries

1. SELECT, FROM, and WHERE (🔍, 📁 and ❓):

emoji_query = "🔍 all 📁 games ❓ price < 2000"

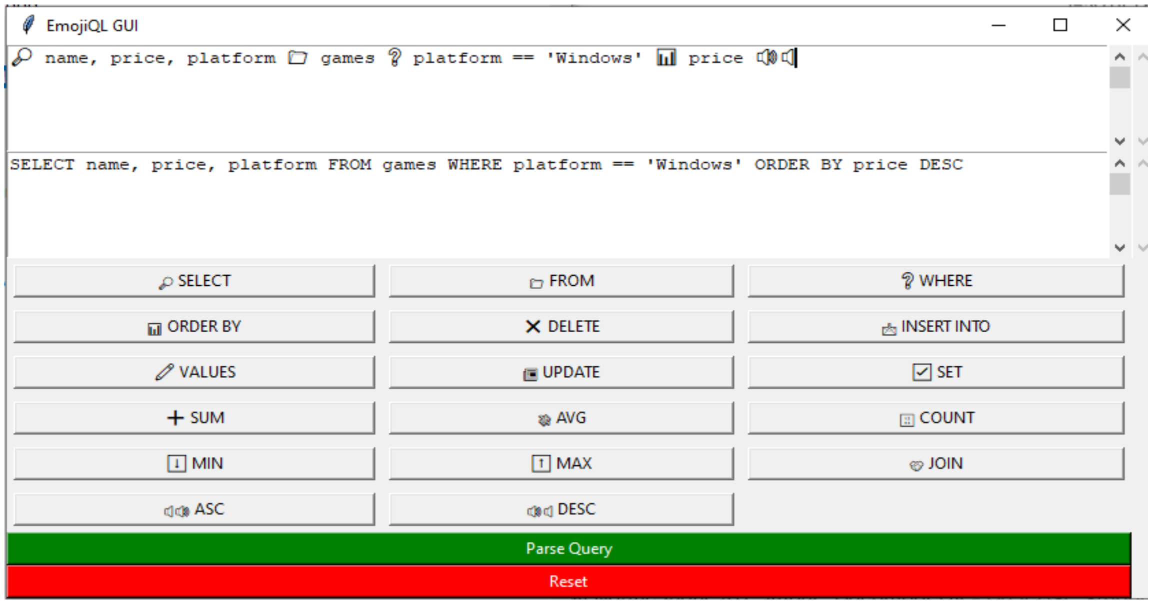




```
Time taken to process 2 chunk files: 0.05 seconds
+----------------------------------+-----------------------------------------------+----------------------------+-------+--------------------------+--------------------+----------------------------------------------
|                id                |                     name                      |         game_slug          | price |       release_date       |      platform      |
+----------------------------------+-----------------------------------------------+----------------------------+-------+--------------------------+--------------------+----------------------------------------------
| 4c81547b81064acfb1902be7b06d6366 |       Assassin's Creed® I: Director's Cut       |     assassins-creed-1      | 1999  | 2008-04-09T15:00:00.000Z |      Windows       |                              You are an Ass
| 3fdbd69050ec4091a68481b397f0a5dd |         LEGO® Batman™: The Videogame          |        lego-batman         | 1999  | 2008-09-28T15:00:00.000Z |      Windows       | When all the villains in Arkham Asylum team
| 5f82cbea3fdd42e2b9b9dfe8439b96b3 |                 World of Goo                  |        world-of-goo        | 1499  | 2008-10-13T15:00:00.000Z |    Windows,Mac     |
| 497cdc35842e458ca10a1edae95ae181 |          Shadow Complex Remastered            |       shadow-complex       | 1499  | 2009-08-19T14:00:00.000Z |    Mac,Windows     |
| 0f96082fc0b8475db86cba86c8b6a796 |  Batman Arkham Asylum Game of the Year Edition  |    batman-arkham-asylum    | 1999  | 2010-03-26T15:00:00.000Z |      Windows       |
| 49bdb24baa564f8c92848abe14112436 |                     Limbo                     |           limbo            |  999  | 2010-07-30T15:00:00.000Z |      Windows       |
| e6dfe28ad2644018aa29f67054ff0391 |                 Costume Quest                 |       costume-quest        |  999  | 2010-10-19T13:00:00.000Z |      Windows       |                              Costume Quest is
| 58f2e8dff4a84c5796ebcbdd5d10fe34 |                Super Meat Boy                 |       super-meat-boy       | 1499  | 2010-10-20T15:00:00.000Z |      Windows       |
| a90dd6aacad84318aa8d81d87cad5063 |         The Walking Dead: Season One          |  walking-dead-season-one   | 1499  | 2012-04-24T15:00:00.000Z |      Windows       | The Walking Dead is a five-part game series
| da621d3c18154c82aedab85687fccdc6 |         Alan Wake's American Nightmare         | alan-wake-american-nightmare |  899  | 2012-05-22T15:00:00.000Z |      Windows       |
| e97aca322ee540edac26749e74e94ca1 |                 Ticket to Ride                |       ticket-to-ride       |  999  | 2012-05-24T08:00:00.000Z |      Windows       | The official adaptation of Days of Wonder's
| bad91db9e66044fdaf195d6a8e882cdb |       LEGO® Batman™ 2: DC Super Heroes        |        lego-batman-2       | 1999  | 2012-06-22T15:00:00.000Z |      Windows       |         Legends Unite! The Dynamic Duo
| 1d96b05c04804b4e8df4b376e3c1c490 |  Batman Arkham City Game of the Year Edition  |     batman-arkham-city     | 1999  | 2012-09-07T15:00:00.000Z |      Windows       |
| 9ccfdd9d080949f99d4d2b0aa321b165 |                Little Inferno                 |       little-inferno       | 1499  | 2012-11-19T15:00:00.000Z | Windows,Mac,Win32  |
| 50ca821abd824c2994ad1798eb8454a6 |          Far Cry®3 Standard Edition           |         far-cry-3          | 1999  | 2013-01-15T10:00:00.000Z |      Windows       |                                  Far Cry 3 is
| 674967c55648467ba8e9d163a27170db |                  The Bridge                   |         the-bridge         |  999  | 2013-02-22T16:00:00.000Z |    Mac,Windows     |        The Bridge is a logic puzzle game t
| b4eb01e10f224126b7b4f7a56a65a6ec |         The Walking Dead: Season Two          |  walking-dead-season-two   | 1499  | 2013-02-23T15:00:00.000Z |      Windows       | Many months have passed since the events se
| 442f123b4d884d8ca85236aa30b99a79 |                      Fez                      |            fez             |  999  | 2013-04-13T15:00:00.000Z |    Windows,Mac     | Gomez is a 2D creature living in a 2D world
| 7981b6d8a37142b9aed96155b93ff8da |       Child of Light: Ultimate Edition        |       child-of-light       | 1999  | 2014-04-30T15:00:00.000Z |      Windows       |                              Aurora, a young
| 9a03fddd502943d2af8c94daeb667b04 |                  Transistor                   |         transistor         | 1999  | 2014-05-20T15:00:00.000Z |    Windows,Mac     |
| 61b641c3702f4770ac525101b39f0063 |            FTL: Faster Than Light             |     faster-than-light      |  999  | 2014-09-14T16:00:00.000Z |    Windows,Mac     |
...
| 8c06c52475f04adb9cdfea4c0a50f734 |              Neko Ghost, Jump!                |      neko-ghost-jump       | 1499  | 2022-01-11T17:00:00.000Z |      Windows       |                                    Neko G
| 618ecf34a09740bb8ea2050b8b03d4b9 |              Rescue Party: Live!              |     rescue-party-live      |  999  | 2022-01-13T16:00:00.000Z |      Windows       |                        Play alone or with yo
+----------------------------------+-----------------------------------------------+----------------------------+-------+--------------------------+--------------------+----------------------------------------------
Time taken to print 2 chunk files: 0.23 seconds
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

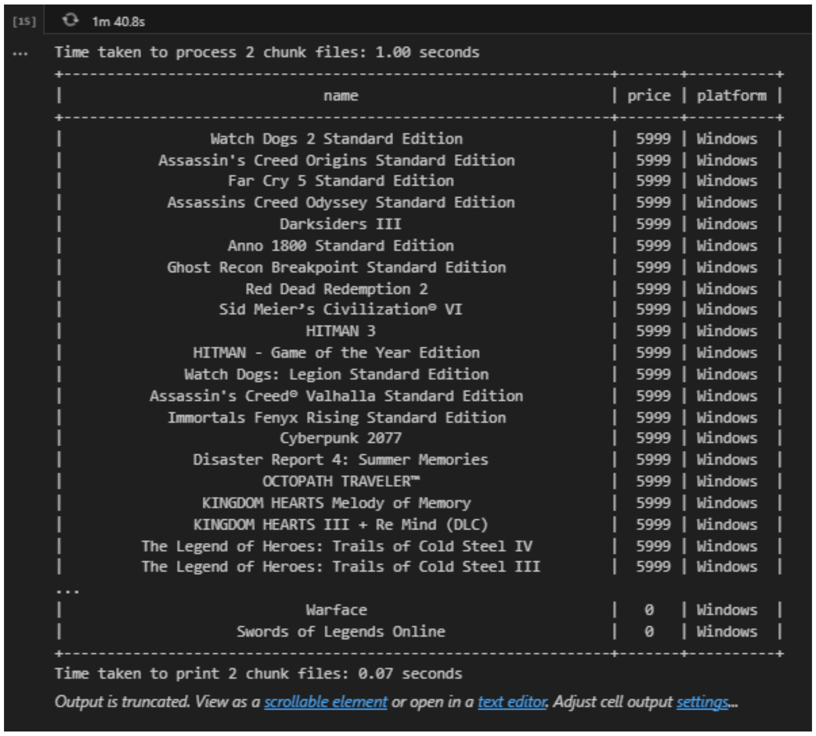
All the columns with price < 2000 are selected from the games table.

2. **ORDER BY, DESC, ASC (📊, 🔈🔊, 🔊🔈):**



```
🔎 name, price, platform 🗀 games ❓ platform == 'Windows' 📊 price 🔊🔈
```

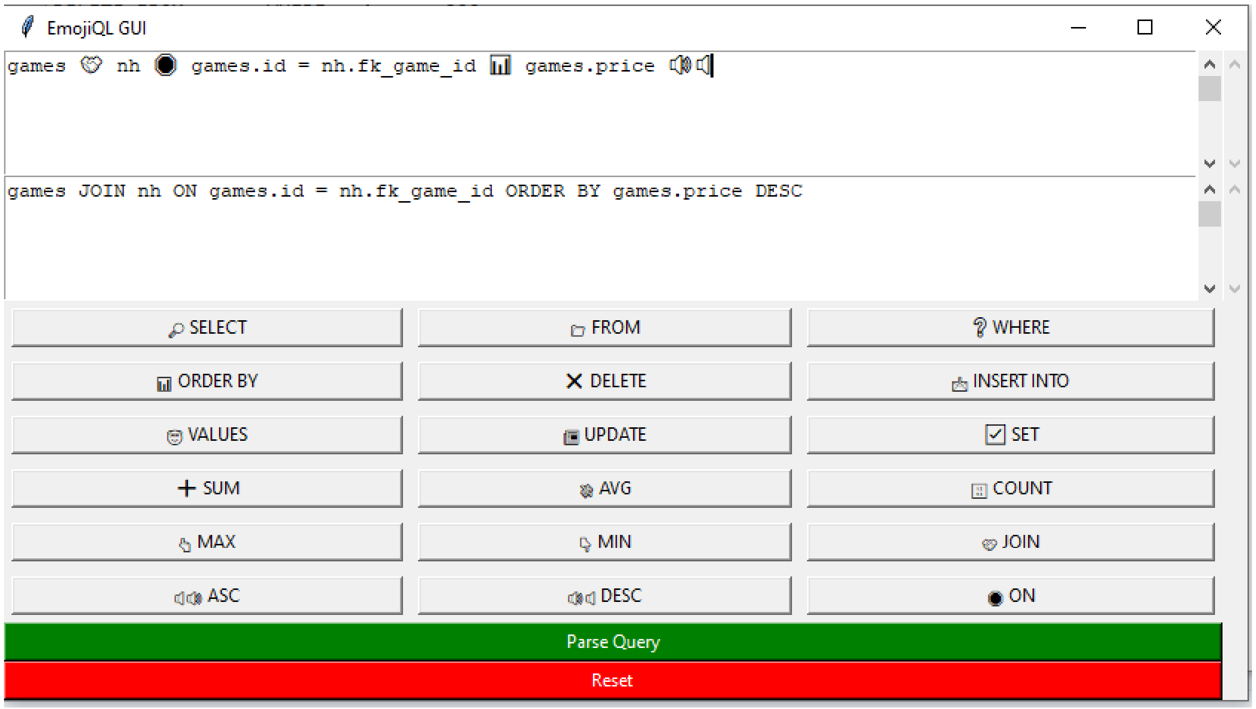
```
SELECT name, price, platform FROM games WHERE platform == 'Windows' ORDER BY price DESC
```



```
[15]   1m 40.8s
...  Time taken to process 2 chunk files: 1.00 seconds
     +----------------------------------------------------------------+-------+----------+
     |                              name                              | price | platform |
     +----------------------------------------------------------------+-------+----------+
     |                   Watch Dogs 2 Standard Edition                |  5999 | Windows  |
     |              Assassin's Creed Origins Standard Edition         |  5999 | Windows  |
     |                    Far Cry 5 Standard Edition                  |  5999 | Windows  |
     |              Assassins Creed Odyssey Standard Edition          |  5999 | Windows  |
     |                          Darksiders III                        |  5999 | Windows  |
     |                     Anno 1800 Standard Edition                 |  5999 | Windows  |
     |               Ghost Recon Breakpoint Standard Edition          |  5999 | Windows  |
     |                      Red Dead Redemption 2                     |  5999 | Windows  |
     |                    Sid Meier's Civilization® VI                |  5999 | Windows  |
     |                             HITMAN 3                           |  5999 | Windows  |
     |                 HITMAN - Game of the Year Edition              |  5999 | Windows  |
     |                 Watch Dogs: Legion Standard Edition            |  5999 | Windows  |
     |              Assassin's Creed® Valhalla Standard Edition       |  5999 | Windows  |
     |               Immortals Fenyx Rising Standard Edition          |  5999 | Windows  |
     |                          Cyberpunk 2077                        |  5999 | Windows  |
     |                 Disaster Report 4: Summer Memories             |  5999 | Windows  |
     |                        OCTOPATH TRAVELER™                      |  5999 | Windows  |
     |                   KINGDOM HEARTS Melody of Memory              |  5999 | Windows  |
     |                  KINGDOM HEARTS III + Re Mind (DLC)            |  5999 | Windows  |
     |             The Legend of Heroes: Trails of Cold Steel IV      |  5999 | Windows  |
     |             The Legend of Heroes: Trails of Cold Steel III     |  5999 | Windows  |
...
     |                             Warface                            |   0   | Windows  |
     |                     Swords of Legends Online                   |   0   | Windows  |
     +----------------------------------------------------------------+-------+----------+
     Time taken to print 2 chunk files: 0.07 seconds
     Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

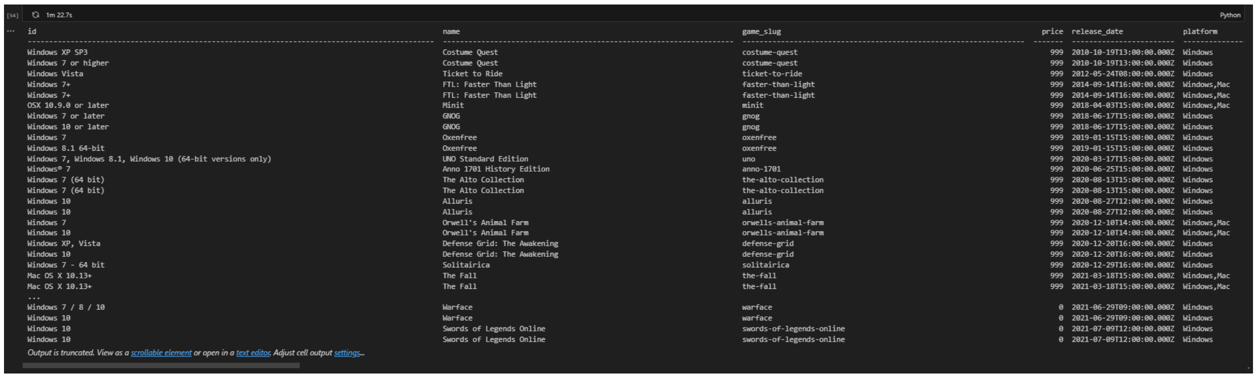
Name, price, and platform columns are selected and rows are ordered in descending order of price values.

3. **JOIN, ON (🤝, 🛑):**



```
games 🤝 nh 🛑 games.id = nh.fk_game_id 📊 games.price 🔊🔈
```

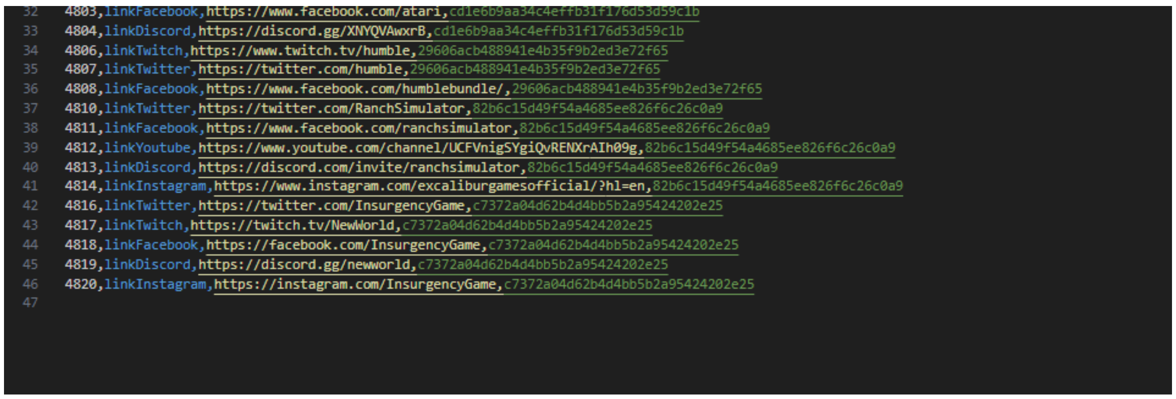
```
games JOIN nh ON games.id = nh.fk_game_id ORDER BY games.price DESC
```



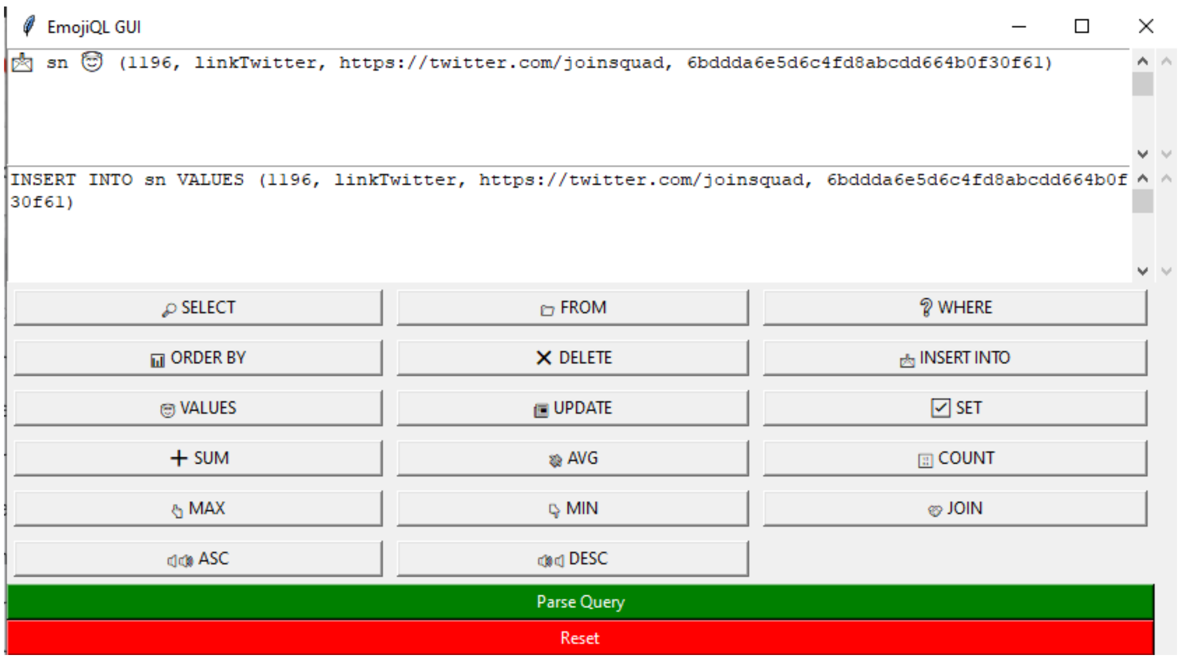
```
[54]   1m 22.7s                                                                                                   Python
id                                                      name                                   game_slug                             price  release_date              platform
------------------------------------------------------  -------------------------------------  ------------------------------------  -----  ------------------------  --------------
Windows XP SP3                                          Costume Quest                          costume-quest                           999  2010-10-19T13:00:00.000Z  Windows
Windows 7 or higher                                     Costume Quest                          costume-quest                           999  2010-10-19T13:00:00.000Z  Windows
Windows Vista                                           Ticket to Ride                         ticket-to-ride                          999  2012-05-24T08:00:00.000Z  Windows
Windows 7+                                              FTL: Faster Than Light                 faster-than-light                       999  2014-09-14T16:00:00.000Z  Windows,Mac
Windows 7+                                              FTL: Faster Than Light                 faster-than-light                       999  2014-09-14T16:00:00.000Z  Windows,Mac
OSX 10.9.0 or later                                     Minit                                  minit                                   999  2018-04-03T15:00:00.000Z  Windows,Mac
Windows 7 or later                                      GNOG                                   gnog                                    999  2018-06-17T15:00:00.000Z  Windows
Windows 10 or later                                     GNOG                                   gnog                                    999  2018-06-17T15:00:00.000Z  Windows
Windows 7                                               Oxenfree                               oxenfree                                999  2019-01-15T15:00:00.000Z  Windows
Windows 8.1 64-bit                                      Oxenfree                               oxenfree                                999  2019-01-15T15:00:00.000Z  Windows
Windows 7, Windows 8.1, Windows 10 (64-bit versions only) UNO Standard Edition                 uno                                     999  2020-03-17T15:00:00.000Z  Windows
Windows® 7                                              Anno 1701 History Edition              anno-1701                               999  2020-06-25T15:00:00.000Z  Windows
Windows 7 (64 bit)                                      The Alto Collection                    the-alto-collection                     999  2020-08-13T15:00:00.000Z  Windows
Windows 7 (64 bit)                                      The Alto Collection                    the-alto-collection                     999  2020-08-13T15:00:00.000Z  Windows
Windows 10                                              Alluris                                alluris                                 999  2020-08-27T12:00:00.000Z  Windows
Windows 10                                              Alluris                                alluris                                 999  2020-08-27T12:00:00.000Z  Windows
Windows 7                                               Orwell's Animal Farm                   orwells-animal-farm                     999  2020-12-10T14:00:00.000Z  Windows,Mac
Windows 10                                              Orwell's Animal Farm                   orwells-animal-farm                     999  2020-12-10T14:00:00.000Z  Windows,Mac
Windows XP, Vista                                       Defense Grid: The Awakening            defense-grid                            999  2020-12-20T16:00:00.000Z  Windows
Windows 10                                              Defense Grid: The Awakening            defense-grid                            999  2020-12-20T16:00:00.000Z  Windows
Windows 7 - 64 bit                                      Solitairica                            solitairica                             999  2020-12-29T16:00:00.000Z  Windows
Mac OS X 10.13+                                         The Fall                               the-fall                                999  2021-03-18T15:00:00.000Z  Windows,Mac
Mac OS X 10.13+                                         The Fall                               the-fall                                999  2021-03-18T15:00:00.000Z  Windows,Mac
...
Windows 7 / 8 / 10                                      Warface                                warface                                   0  2021-06-29T09:00:00.000Z  Windows
Windows 10                                              Warface                                warface                                   0  2021-06-29T09:00:00.000Z  Windows
Windows 10                                              Swords of Legends Online               swords-of-legends-online                  0  2021-07-09T12:00:00.000Z  Windows
Windows 10                                              Swords of Legends Online               swords-of-legends-online                  0  2021-07-09T12:00:00.000Z  Windows
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

Tables games and nh are joined on games id and nh games_id which is also ordered

4. **INSERT INTO, VALUES (📩, 😇):**

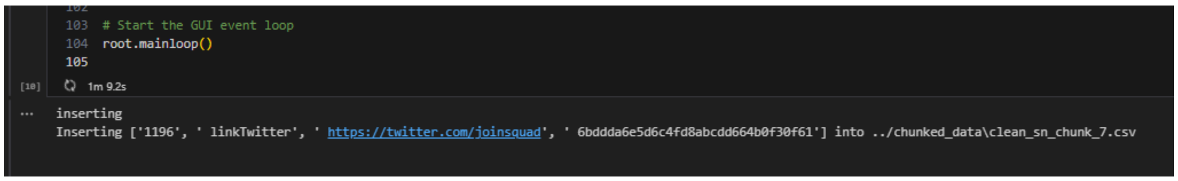


```
4803,linkFacebook,https://www.facebook.com/atari,cd1e6b9aa34c4effb31f176d53d59c1b
4804,linkDiscord,https://discord.gg/XNYQVAwxrB,cd1e6b9aa34c4effb31f176d53d59c1b
4806,linkTwitch,https://www.twitch.tv/humble,29606acb488941e4b35f9b2ed3e72f65
4807,linkTwitter,https://twitter.com/humble,29606acb488941e4b35f9b2ed3e72f65
4808,linkFacebook,https://www.facebook.com/humblebundle/,29606acb488941e4b35f9b2ed3e72f65
4810,linkTwitter,https://twitter.com/RanchSimulator,82b6c15d49f54a4685ee826f6c26c0a9
4811,linkFacebook,https://www.facebook.com/ranchsimulator,82b6c15d49f54a4685ee826f6c26c0a9
4812,linkYoutube,https://www.youtube.com/channel/UCFVnigSYgiQvRENXrAIh09g,82b6c15d49f54a4685ee826f6c26c0a9
4813,linkDiscord,https://discord.com/invite/ranchsimulator,82b6c15d49f54a4685ee826f6c26c0a9
4814,linkInstagram,https://www.instagram.com/excaliburgamesofficial/?hl=en,82b6c15d49f54a4685ee826f6c26c0a9
4816,linkTwitter,https://twitter.com/InsurgencyGame,c7372a04d62b4d4bb5b2a95424202e25
4817,linkTwitch,https://twitch.tv/NewWorld,c7372a04d62b4d4bb5b2a95424202e25
4818,linkFacebook,https://facebook.com/InsurgencyGame,c7372a04d62b4d4bb5b2a95424202e25
4819,linkDiscord,https://discord.gg/newworld,c7372a04d62b4d4bb5b2a95424202e25
4820,linkInstagram,https://instagram.com/InsurgencyGame,c7372a04d62b4d4bb5b2a95424202e25
```

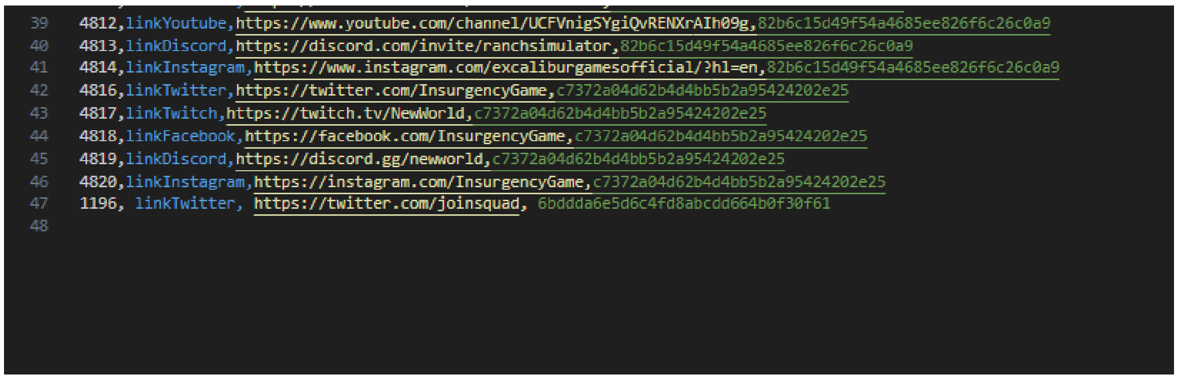


```
📩 sn 😇 (1196, linkTwitter, https://twitter.com/joinsquad, 6bddda6e5d6c4fd8abcdd664b0f30f61)
```

```
INSERT INTO sn VALUES (1196, linkTwitter, https://twitter.com/joinsquad, 6bddda6e5d6c4fd8abcdd664b0f
30f61)
```

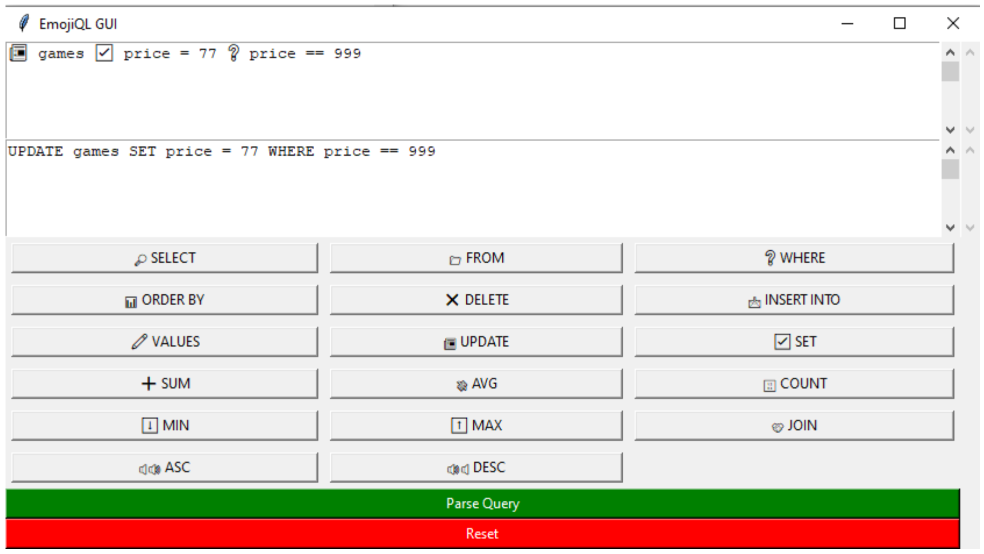


```
# Start the GUI event loop
root.mainloop()

[10]  1m 9.2s

inserting
Inserting ['1196', ' linkTwitter', ' https://twitter.com/joinsquad', ' 6bddda6e5d6c4fd8abcdd664b0f30f61'] into ../chunked_data\clean_sn_chunk_7.csv
```



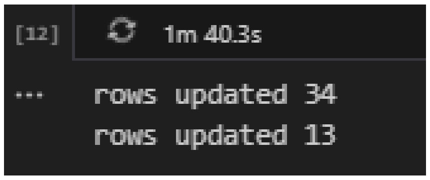
```
4812,linkYoutube,https://www.youtube.com/channel/UCFVnigSYgiQvRENXrAIh09g,82b6c15d49f54a4685ee826f6c26c0a9
4813,linkDiscord,https://discord.com/invite/ranchsimulator,82b6c15d49f54a4685ee826f6c26c0a9
4814,linkInstagram,https://www.instagram.com/excaliburgamesofficial/?hl=en,82b6c15d49f54a4685ee826f6c26c0a9
4816,linkTwitter,https://twitter.com/InsurgencyGame,c7372a04d62b4d4bb5b2a95424202e25
4817,linkTwitch,https://twitch.tv/NewWorld,c7372a04d62b4d4bb5b2a95424202e25
4818,linkFacebook,https://facebook.com/InsurgencyGame,c7372a04d62b4d4bb5b2a95424202e25
4819,linkDiscord,https://discord.gg/newworld,c7372a04d62b4d4bb5b2a95424202e25
4820,linkInstagram,https://instagram.com/InsurgencyGame,c7372a04d62b4d4bb5b2a95424202e25
1196, linkTwitter, https://twitter.com/joinsquad, 6bddda6e5d6c4fd8abcdd664b0f30f61
```

5. **UPDATE, SET (📰, ✅):**



```
📰 games ✅ price = 77 ❓ price == 999
```

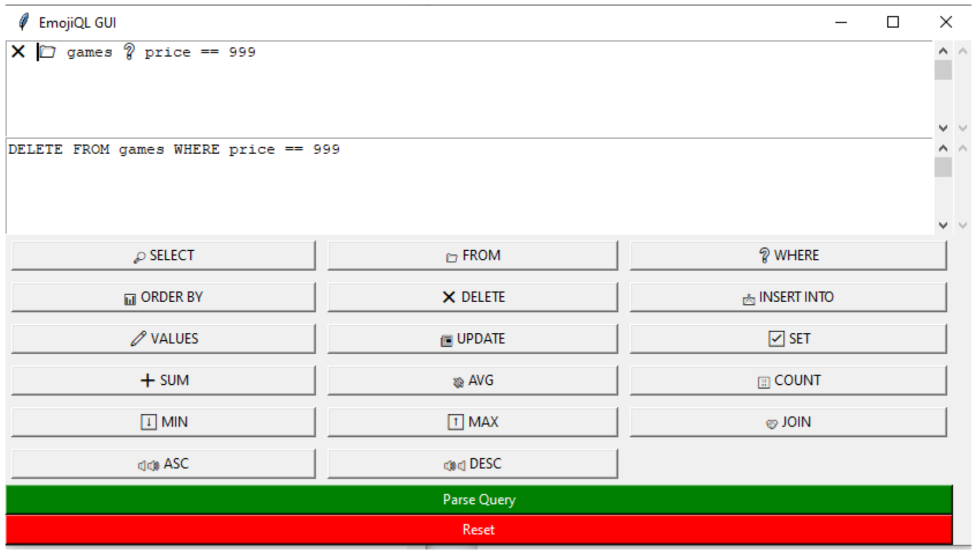
```
UPDATE games SET price = 77 WHERE price == 999
```



```
[12]   1m 40.3s

...    rows updated 34
       rows updated 13
```

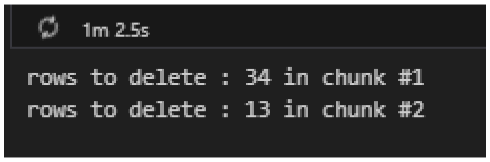
Shows how many rows are updated in each chunk file.

6. **DELETE (❌):**



```
❌ 📁 games ❓ price == 999
```

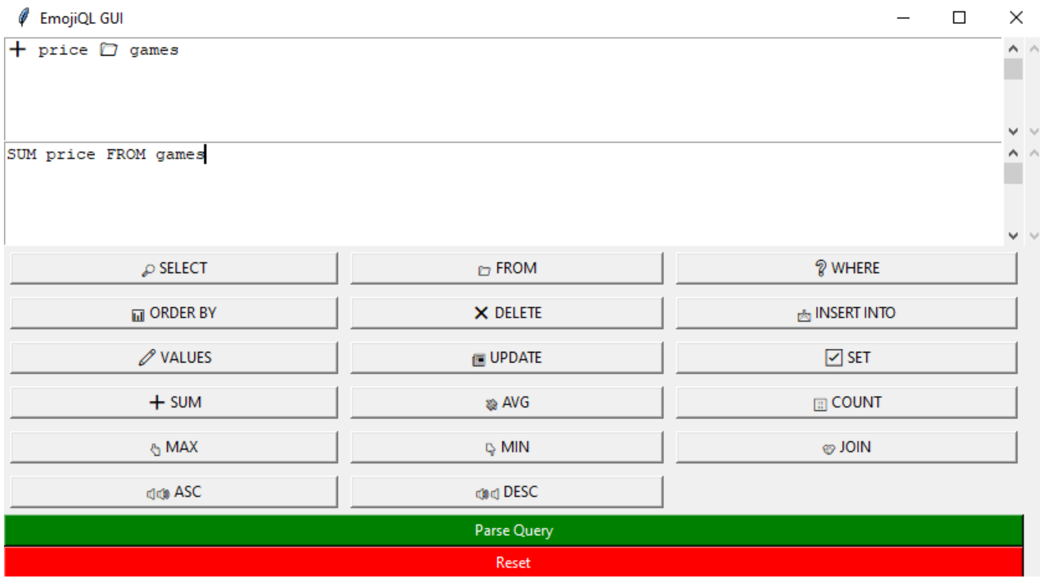
```
DELETE FROM games WHERE price == 999
```



```
1m 2.5s

rows to delete : 34 in chunk #1
rows to delete : 13 in chunk #2
```

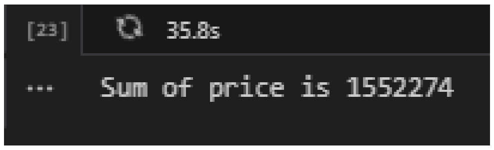
Shows how many rows are deleted in each chunk file.

7. **SUM (➕):**



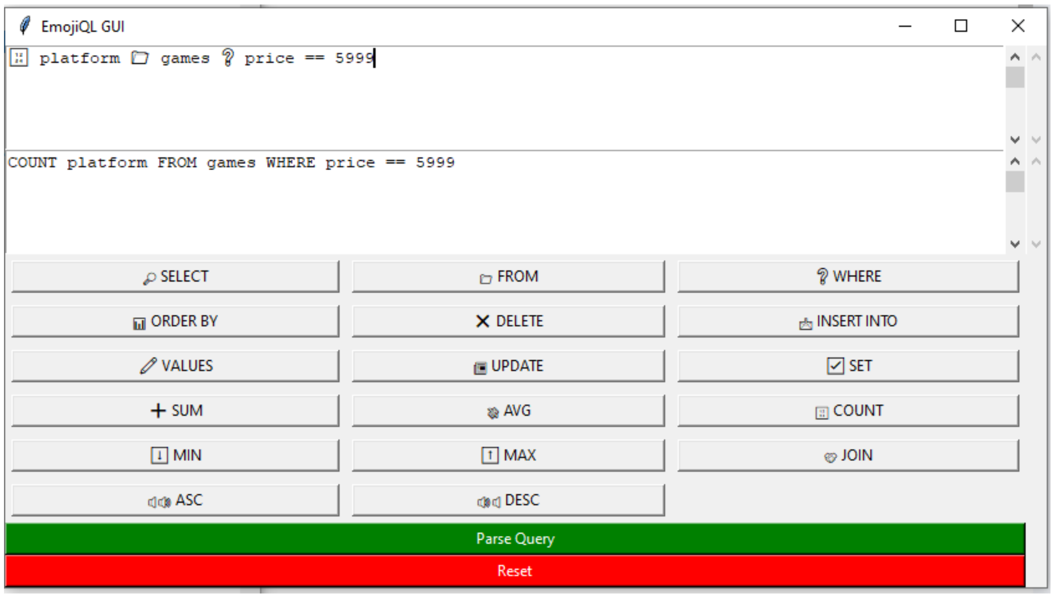
```
➕ price 🗀 games
```

```
SUM price FROM games
```



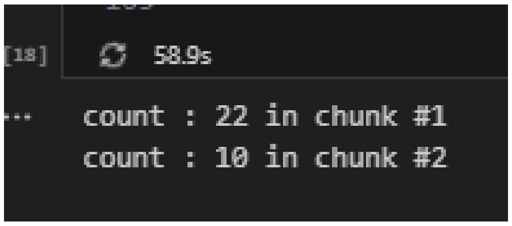
```
[23]  35.8s
...  Sum of price is 1552274
```

8. **COUNT (🔢):**



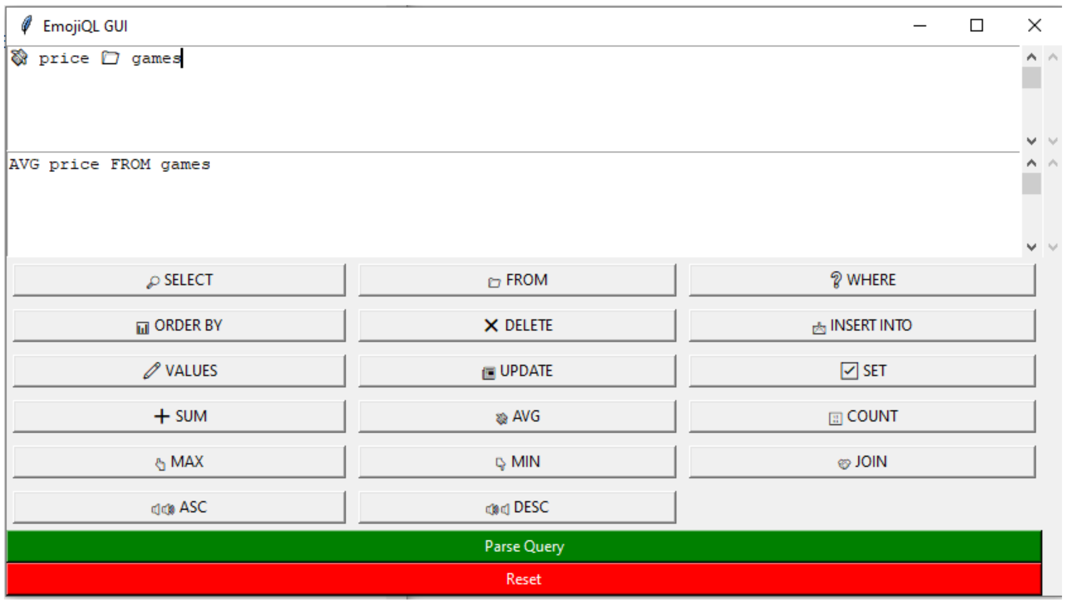
```
🔢 platform 🗀 games ❓ price == 5999
```

```
COUNT platform FROM games WHERE price == 5999
```



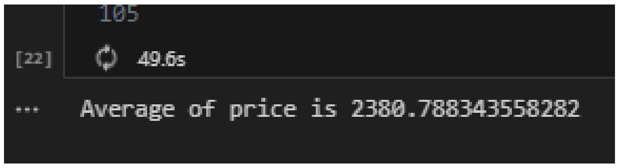
```
[18]  58.9s
...  count : 22 in chunk #1
     count : 10 in chunk #2
```

9. **AVERAGE (💸):**



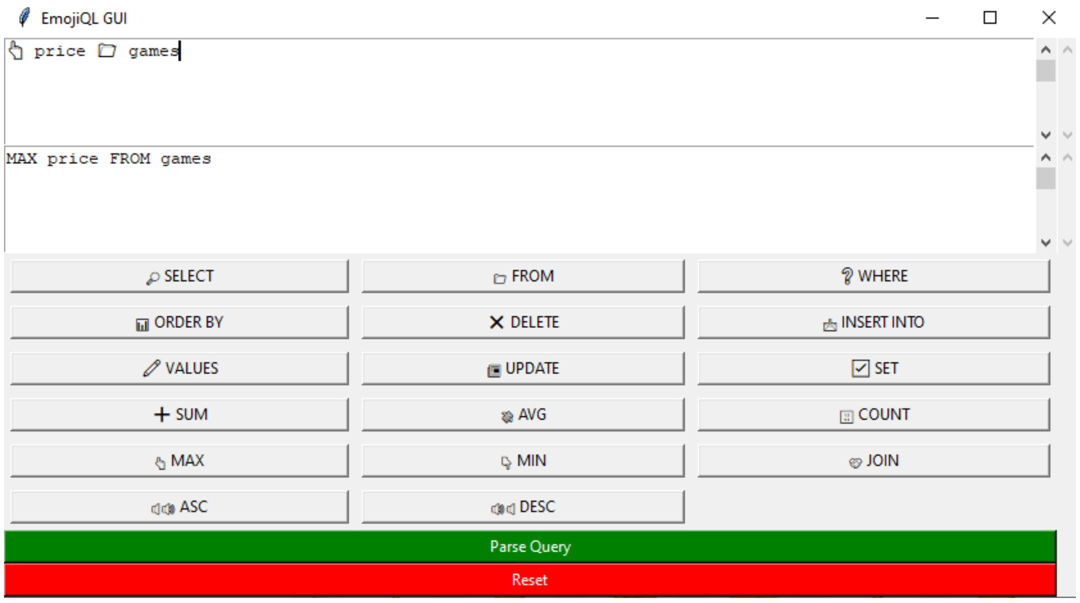
```
price 🗀 games
```

```
AVG price FROM games
```



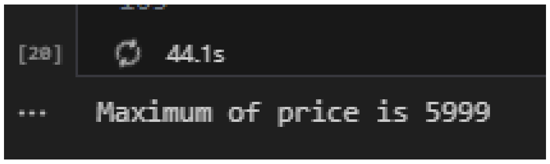
```
[22]  49.6s
...   Average of price is 2380.788343558282
```

10. **MAX (☝️):**



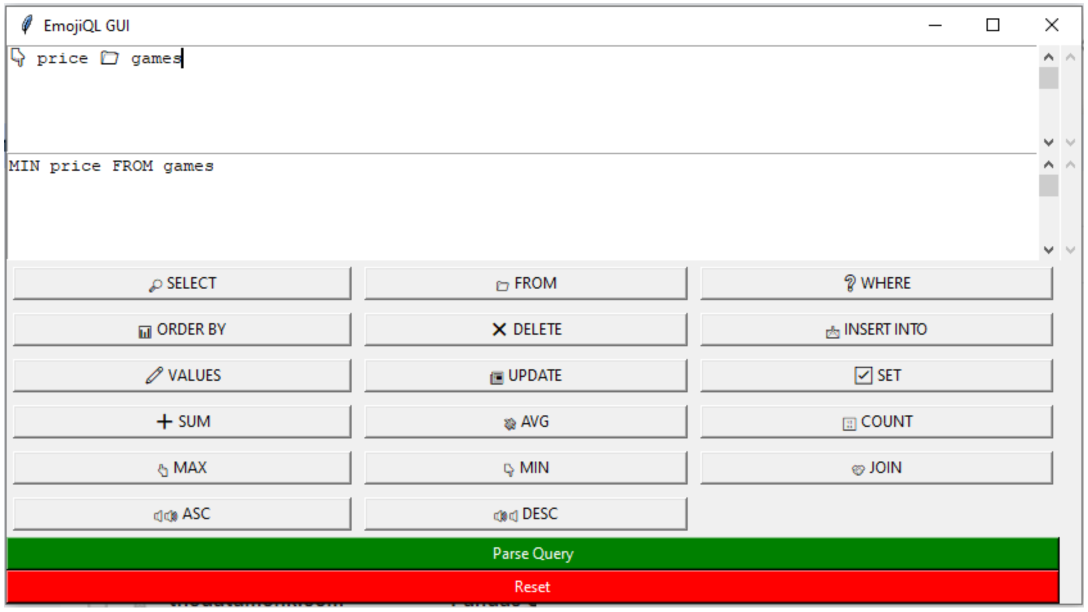
```
price 🗀 games
```

```
MAX price FROM games
```



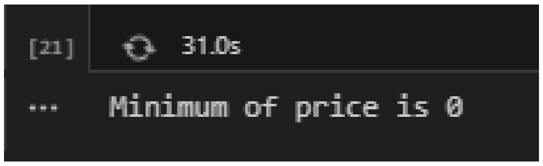
```
[28]  44.1s
...   Maximum of price is 5999
```

11. **MIN** (👇):



```
👇 price 🗀 games
```

```
MIN price FROM games
```



```
[21]   31.0s
...    Minimum of price is 0
```

LEARNINGS FROM THE PROJECT

- <u>Interdisciplinary Innovation:</u> Learned the importance of combining fields—user experience design with database management—to create innovative solutions.
- <u>Emoji as a Communication Tool:</u> Gained insight into the versatility of emojis as a means of simplifying complex technical concepts.
- <u>Python Proficiency:</u> Enhanced Python coding skills, especially in using libraries such as Tkinter for GUI development, glob for file handling, and joblib for parallel processing.
- <u>SQL Deep Dive:</u> Deepened understanding of SQL operations and how to effectively translate user inputs into executable queries.
- <u>Efficiency in Data Management:</u> Acquired knowledge on handling large datasets efficiently through chunking and parallel processing.
- <u>Designing User Interfaces:</u> Learned best practices for designing user-friendly interfaces that cater to a diverse user base.
- <u>Debugging and Testing:</u> Improved debugging and testing skills, learning to rigorously test each component to ensure a robust application.

Challenges faced during the project

- <u>Emoji-Query Mapping:</u> One of the initial challenges was to accurately map a wide range of emojis to corresponding SQL operations, ensuring intuitive associations.
- <u>Memory Management:</u> Faced issues with memory overload when dealing with large datasets, which led to the implementation of data chunking strategies. Finally wrote my own function to chunk the large csv files.
- <u>User Experience:</u> Balancing simplicity for the user while maintaining the functionality of complex SQL queries was a significant challenge.
- <u>Performance Optimization:</u> Ensuring the application ran efficiently, especially when scaling up to handle larger datasets and more complex queries.
- <u>Cross-Platform Compatibility:</u> Encountered difficulties in making the GUI responsive and consistent across different operating systems. (tkinter doesn't work properly on mac so stuck with windows only)
- <u>Community Feedback:</u> Incorporating user feedback into the development process was challenging but critical for the project's iterative improvement. TA's feedback on having improvements on running various emoji queries were addressed and accomplished.

CONCLUSIONS

- Innovative Approach: EmojiQL has successfully demonstrated an innovative approach to SQL querying, making it more accessible and engaging.
- User-Friendly Interface: The project has developed a user-friendly interface that simplifies the interaction with databases.
- Effective Translation: The Emoji Parser effectively translates emoji inputs into SQL queries, bridging the gap between non-technical users and database operations.
- Efficient Data Handling: Data chunking and parallel processing have proven effective in managing and querying large datasets.
- Educational Tool: EmojiQL has potential as an educational tool for introducing users to SQL concepts in a more intuitive manner.

FUTURE SCOPE

- Natural Language Processing (NLP): Incorporate NLP to further simplify query generation, allowing users to input queries in plain English alongside emojis.
- Wider Emoji Support: Expand the range of supported emojis to cover more SQL functionalities and operators.
- Advanced Database Operations: Introduce more complex database operations such as subqueries, transactions, and stored procedures.
- Cross-Platform GUI: Develop a cross-platform GUI to make EmojiQL accessible on various operating systems.
- Community Engagement: Open-source the project to encourage community contributions, leading to new features and improvements.
- Performance Optimization: Focus on optimizing the performance for handling even larger datasets and more concurrent users.
- Integration with Other Databases: Extend compatibility to various database systems, including NoSQL databases, for broader applicability.