Database Re-Design
for
Spaceman "Hangman" Game

Andrew Dillard

07/21/2024

OVERVIEW

I will be using PostgreSQL for data related to my Spaceman game web application. I had originally opted for SQLite, but ultimately went with PostgreSQL since it's more user friendly and for its relational model – which bests suits the structure for the game data. This choice allows for efficient and simplistic querying and management of game states, scores, and user data without the need for complex relationships between tables. Additionally, PostgreSQL offers strong support for transactions, as it follows ACID compliance – ensuring that each score submission completes successfully without leaving the database in an inconsistent state.

The web application will interact with PostgreSQL through the backend service layer, handling all database operations. This approach separates the database concerns from the frontend, adhering to best practices in web application architecture.

DATA SPECIFICATIONS

For data validation and schema management, I will utilize an ORM (Object-Relational Mapping) library for Node.js, which provides a simple way to model PostgreSQL tables and execute queries.

Table: "User Scores"

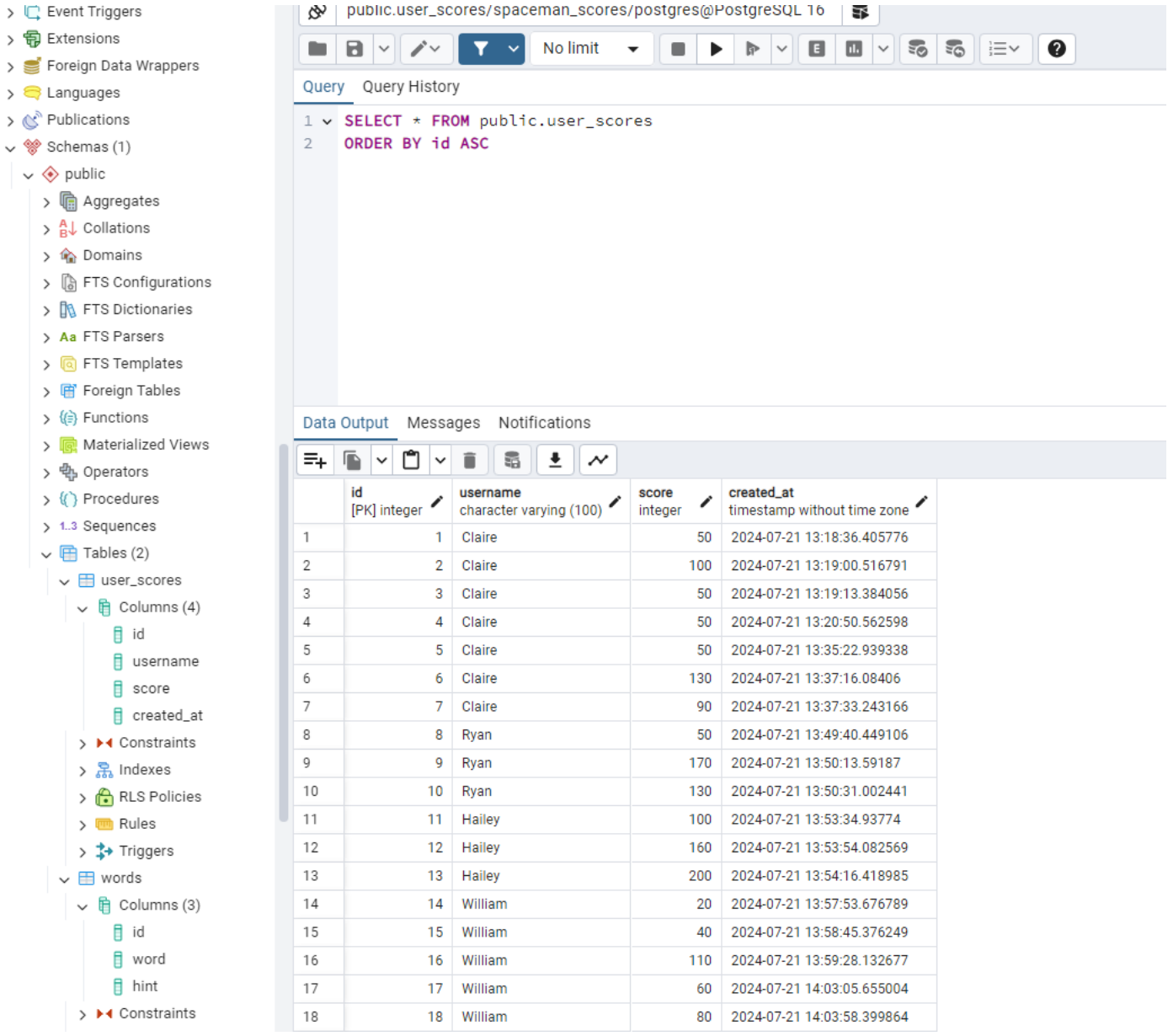
This table will store high scores achieved by players (like an arcade style version of scoring). The front end will display the current top 5 user scores. User's names are not unique, nor is there a password/login type system.

Example High Scores (as seen on the front-end):

Alice 180
Billy 160
Alice 150
Claire: 120
William: 110

The table (User_Scores) is used to record game sessions.

- id: unique ID for the data row.
- username: allows multiple game sessions to be recorded under the same name.
- score: records the score achieved by the player during each game session.
- created_at: timestamps when each game session record is created.



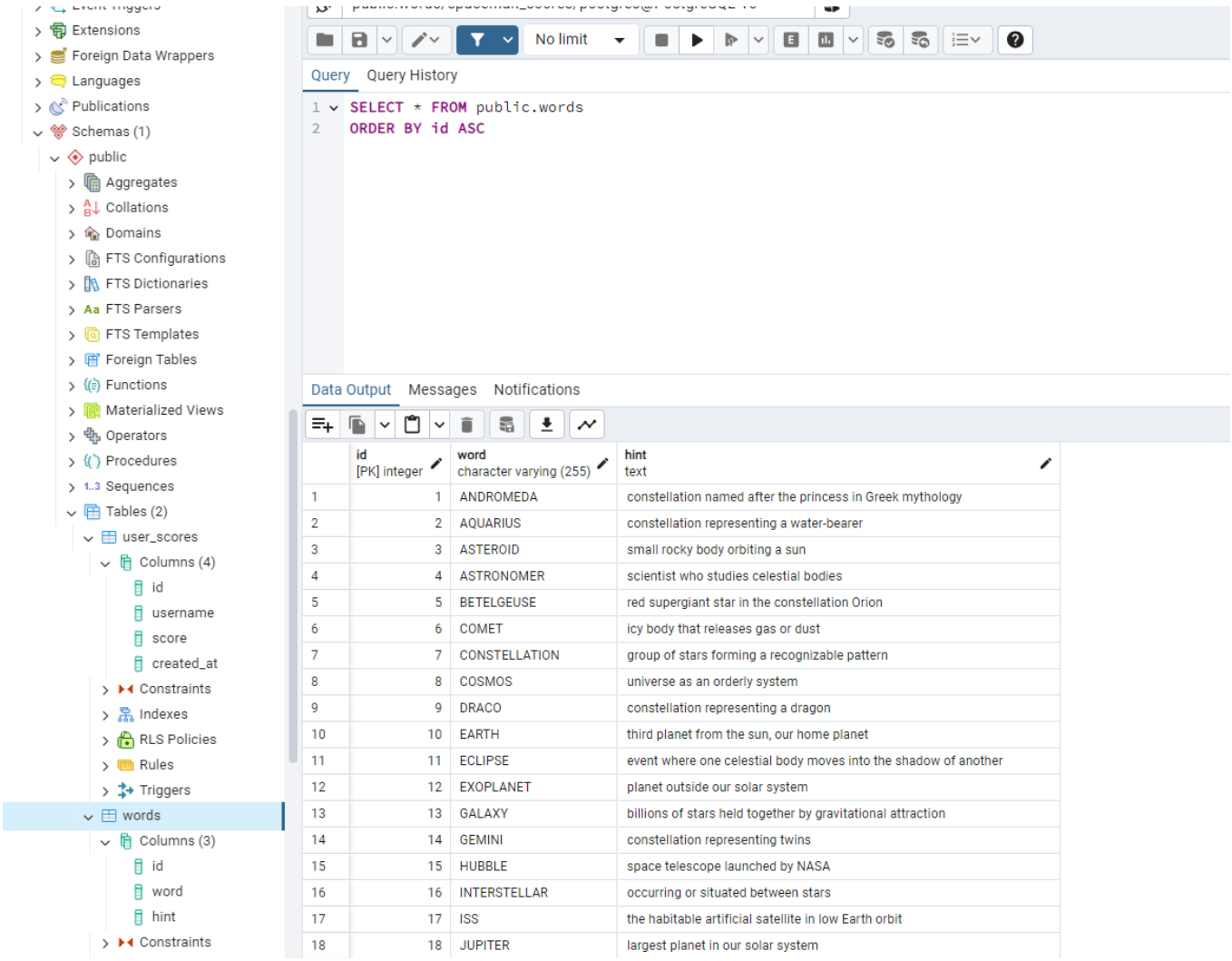
	id [PK] integer	username character varying (100)	score integer	created_at timestamp without time zone
1	1	Claire	50	2024-07-21 13:18:36.405776
2	2	Claire	100	2024-07-21 13:19:00.516791
3	3	Claire	50	2024-07-21 13:19:13.384056
4	4	Claire	50	2024-07-21 13:20:50.562598
5	5	Claire	50	2024-07-21 13:35:22.939338
6	6	Claire	130	2024-07-21 13:37:16.08406
7	7	Claire	90	2024-07-21 13:37:33.243166
8	8	Ryan	50	2024-07-21 13:49:40.449106
9	9	Ryan	170	2024-07-21 13:50:13.59187
10	10	Ryan	130	2024-07-21 13:50:31.002441
11	11	Hailey	100	2024-07-21 13:53:34.93774
12	12	Hailey	160	2024-07-21 13:53:54.082569
13	13	Hailey	200	2024-07-21 13:54:16.418985
14	14	William	20	2024-07-21 13:57:53.676789
15	15	William	40	2024-07-21 13:58:45.376249
16	16	William	110	2024-07-21 13:59:28.132677
17	17	William	60	2024-07-21 14:03:05.655004
18	18	William	80	2024-07-21 14:03:58.399864

Table: “Words”

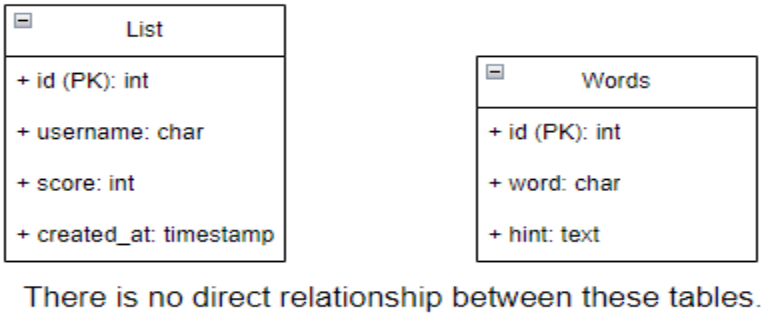
This table will store the database of words the game will use. The game will pull and cycle through this list at the start of each game and after each completed word.

The table (Words) is used to record game sessions.

- id: unique ID for that row
- word: the word that’s used in the game.
- hint: the hint the player sees to help guess the word.



Entity-Relationship



The ERD design is simple as the requirements of the game are not overly-complex. In arcade fashion, player names are recorded before playing and scores are saved at the end of each game session – without the need for storing unique user-specific data.

PURPOSE, IMPLEMENTATION, AND INTERACTIONS

Purpose:

The scores achieved by players are recorded – in much the same fashion as the old arcade games like Donkey Kong. No login, email or password is required, and the score is saved per session and tagged by the name entered at the start of the game. This promotes competition between users and themselves.

All words used in this are stored in a separate database with the purpose of being interchangeable when needed.

Implementation:

Each time a player successfully completes a game session with a high score, a new record is inserted into this table. Scores are sorted in descending order to display the highest scores prominently.

All the words stored in the database will be pulled randomly. No new words can be added from the front end.

Interaction:

The ultimate idea is that players will be able to view the high scores on a leaderboard displayed in the front-end interface, providing motivation to achieve higher scores with each gameplay session.