Aqib Ahsan, Andrew Perrone, Sri Sakticharan

Professor Houwei Cao

Database Management

Final Project Report

December 13, 2023

Fantasy Premier League Database

Abstract:

Fantasy Premier League (FPL) is a website that hosts data from the real-life football/soccer Premier League, allows users to create virtual teams of players using that data, and compete in leagues with other users. There is a large amount of data available on the website which lends itself to the relational database model and there are user interactions that can be represented and recorded in the same model.

Our database stores real-life data from the FPL website including: season data, league data, teams and their statistics, players and their statistics, and match data (fixtures). Currently all real-life data is fixed and does not update according to changes in FPL’s current dataset. Our database also has user generated data including: users and their teams (as one relation), users’ teams statistics, players on a user’s team, user generated leagues, which users are members of which league and fixtures between users’ teams. Currently this data has been input manually and is only a fraction of the full database, but we have plans to extract the relevant data from the FPL website’s API and input it into the database. As for the user generated data, we have plans to create randomized dummy data using a script.

We have given some example SQL queries for data retrieval and three views that aggregate each team’s, player’s, and user’s team statistics from each season to create their “all time” statistics. We have written the necessary scripts that run a fixture between two users and update the stats of each user after the fixture is created. We have plans to make a transaction out of this process because, as it is currently written, it requires the most recent fixture id. If another fixture were to be created during this process, incorrect data could be added to the user’s statistics.

This report also includes an entity relation (ER) diagram representing the schema of the database. The schema (and the database) has been normalized into third normal form (3NF).

Introduction:

Millions of people around the world play and enjoy watching football/soccer (as well as many other sports). Fantasy football websites provide a unique way to engage with the sport that would otherwise not be possible. It’s a common feeling when watching sports that if you were in charge of the team, you could lead them to victory. Fantasy football websites allow users to take charge and imagine a team their way. Users can create custom teams using the statistics of real players and compete with each other to see if their skills really match up to reality. Centralizing the data allows users to make informed decisions on how to construct their teams and including accurate live data, such as upcoming matches, can be valuable to fans.

The Fantasy Premier League (FPL) is one of these websites. FPL provides live statistics relevant to the real-life Premier League (a football league in England). These include team, player, and fixture (match) statistics. Users can then add players from the premier league (according to a budget and price for the player). After creating their team users can then join leagues of other users and compete in fantasy fixtures. Performance is informed by players’ real-life statistics. Good performance in fantasy fixtures awards users with points towards their budget. Their budget can then be used to draft/swap new players to potentially improve their team’s performance.

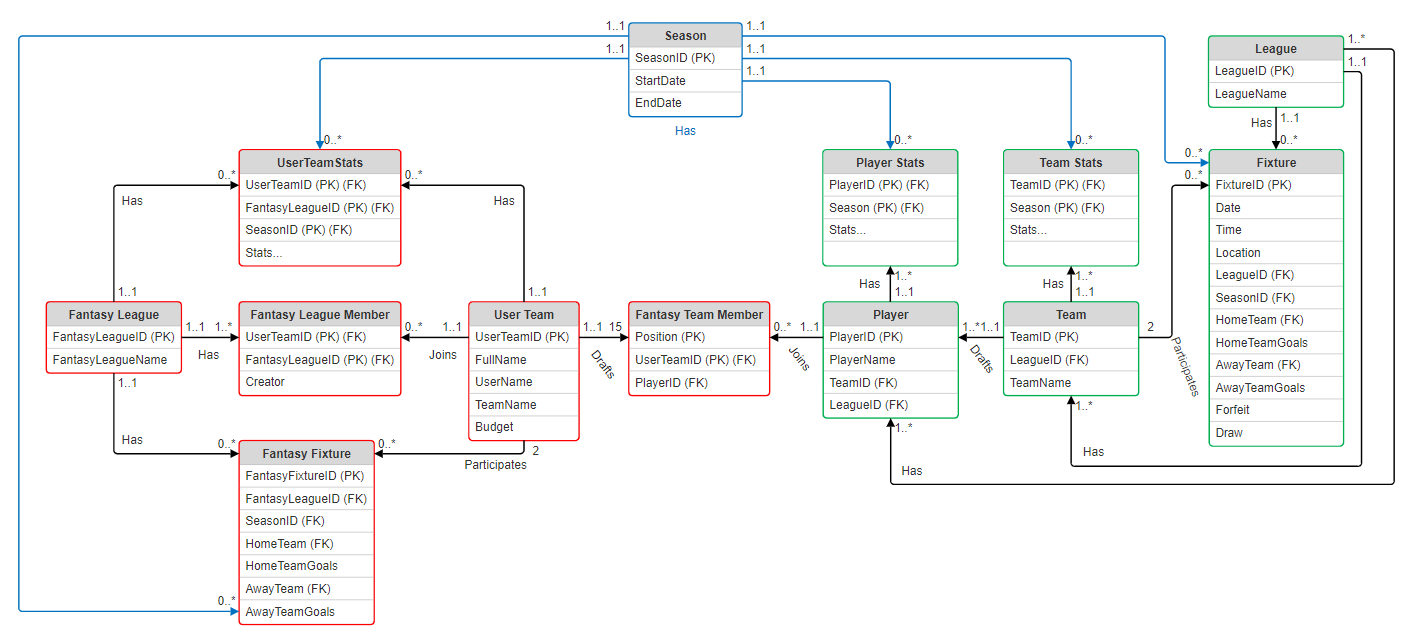
Our goal was to replicate the systems in the FPL website using a Structure Query Language (SQL) database. There are two main systems that needed to be replicated: the statistics (real life data) and the fantasy game (users’ information and interactions). SQL databases provide simple data storage and retrieval which is exactly what is needed to mimic the FPL’s player and team statistics search function. SQL is also well suited to recording users’ fantasy teams, leagues and fixtures.

To achieve the desired result for the database we would need to complete a series of subtasks listed below:

* Decide what specific information and functions are required of the database
* Define the database schema and create a 3NF ER diagram
* Write SQL queries to construct the database
* Collect relevant player, team and fixture statistics from the FPL website (preferably by using a script but manual data entry is acceptable)
* Create relevant user data (again a script to generate data (dummy data) is preferred but manual entry is acceptable)
* Input data
* Add functionalities included in the FPL website:
  + Ability for users to draft/trade players to their teams with limitations (players must stay within budget)
  + Ability for users to create and join leagues
  + Ability for users to run fixtures within a league
  + An automated method for updating a user’s team statistics after a fixture is run
* Create views where applicable

Database Functions:

* Store and retrieve real life statistics:
  + Season
  + League
  + Team
  + Player
  + Fixture
* Store user data and preform user related functions:
  + Add players to team
  + Join a fantasy league
  + Run a fantasy fixture
  + Update Users’ team statistics

Proposed Database (Entity Relation Diagram):

Details:

The database has two main functions which can be described as the “real” side (consisting of real-life data) and the “fantasy” side (consisting of user data and interactions. These pieces are separate except through the “fantasy team member” relation. Some effort was put into devising a structure that would combine these two sides into one, but that turned out not to be feasible (this will be discussed further in the conclusion). Each relation is touched upon in further detail on the following page. Primary keys are underlined, foreign keys are labeled.

Real Side:

1. Season: seasonID, startDate, endDate
   1. The Premier League has a season that starts around the August of one year and continues until around May of the next year.
   2. Statistics and fixtures are tracked during a season (regardless of whether they are real or fantasy)
2. League: leagueID, leagueName
   1. There is currently only one league (the Premier League) but we wanted the database to be expandable
3. Team: teamID, leagueID (FK), teamName
4. TeamStats: teamID (FK), seasonID (FK), statistics…
   1. Team statistics are recorded for each season
   2. A view has been created (total\_teamstats) which represents a team’s statistics for all time
5. Player: playerID, playerName, teamID (FK), leagueID (FK)
   1. Players reference both their team and their league to prevent chasm traps
6. PlayerStats: playerID (FK), seasonID (FK), statistics…
   1. Player statistics are recorded for each season
   2. A view has been created (total\_playerstats) which represents a player’s statistics for all time
7. Fixture: fixtureID, leagueID (FK), seasonID (FK), homeTeamID (FK), awayTeamID (FK), …
   1. Fixtures occur during a season
   2. Fixtures are associated with a league
   3. Two teams must compete in a fixture (home and away)

Fantasy Side:

1. FantasyLeague: fantasyLeagueID, fantasyLeagueName
   1. User made leagues
2. UserTeam: userTeamID, fullName, userName, teamName, budget
   1. The concept of a user and their team are combined into one relation because in the FPL website users only have one team associated with their account
3. FantasyLeagueMember: fantasyLeagueID (FK), userTeamID (FK), creator
   1. This relation connects users to the league or leagues they are members of
4. UserTeamStats: userTeamID (FK), seasonID (FK), statistics…
   1. UserTeam statistics are recorded for each season AND for each league
   2. A view has been created (total\_userteamstats) which represents a userteam’s statistics for all time, across all fantasy leagues
5. FantasyFixture: fantasyFixtureID, fantasyLeagueID (FK), seasonID (FK), homeTeamID (FK), awayTeamID (FK), …
   1. Fantasy fixtures occur during a season
   2. Fantasy fixtures are associated with a fantasy league
   3. Two userteams must compete in a fantasy fixture (home and away)
   4. FantasyFixture result should be dependent on the stats of the players on a userTeam but currently are completely random
6. FantasyTeamMember: position, userTeamID (FK), playerID (FK)
   1. When a userTeam is created fifteen (15) FantasyTeamMembers are made with it, playerID’s default to null
   2. One for each position (goal keeper, striker, etc.) totaling to eleven (11)
   3. Four (4) reserve members
   4. Position is required to be part of the primary key because playerID starts as null
   5. Multiple users may have the same player on their team but one user may not duplicates of the same player, this currently is not the case. A custom constraint to prevent this still needs to be made
   6. Currently adding players to a team does not take into account the userTeam’s budget, a custom constraint still needs to be made for this.

Conclusion:

The Fantasy Premier League database project has been successfully been designed and implemented. The ER diagram served as a foundational guide, facilitating the creation of a well-formed database. The created database fulfilled the project objectives allowing for real statistics and user information to be stored and retrieved. The baseline for user interactions is created as well. The database currently is partially filled with data. The data was collected and inputted manually but we have plans to collect and implement more from the website using scripts. Three views that aggregate statistic have been created and are functional. The ability to run fixtures between two users works and we have a method to update user statistics after they are run.

Somethings were harder than expected, such as data collection, constraints, and creating functions out of SQL queries. Without these functions and constraints, adding user data could cause a fault in data integrity. We intend to continue to work on the database to fix these issues. Regardless, the project has completed all objectives and meets expectations.

Contributions:

Aqib Ahsan: Project inspiration, database design, data collection and input, SQL queries, testing/debugging

Andrew Perrone: Database design, database implementation, data collection and input, SQL queries, SQL views, testing/debugging

Sri Sakticharan: Database design, data collection and input, SQL queries, testing/debugging, creation of presentations