Derek Watson
CSPB 3287: Fall 2023

Proposal: Soccer Club Explorer

A web application that allows user to explore 15 years of soccer match data for 383 European football clubs, covering nearly 68,000 matches.

Data Source and Database Design

My data was sourced from the website https://www.football-data.co.uk/ which has match data for multiple leagues of eleven countries. I chose to include only the largest, most well-known leagues in my dataset selecting data from the top two leagues in England, France, Germany, Italy and Spain and the top leagues in the Netherlands and Portugal. Originally, I wanted to do 20 years of data, but after downloading the ~250 CSVs comprising that data, I found that the data quality dropped off after 15 years, so I am including data for all seasons 2008-09 to 2022-23 for 12 leagues.

To collect and process the data I wrote a series of python scripts. One to download the 200+ CSVs from the website, one to process each and combine them, one to analyze the combined CSV to check data quality and one to create massive JSON files organized by database entity. I iterated through the combined CSV to create a record for what would be each instance of each entity in the database: Country, League, Season, Team, Bookie, Referee, Odds and TeamMatchStats. This created a JSON file that was about 400,000 lines and included tens of thousands of entity instances in JSON format.

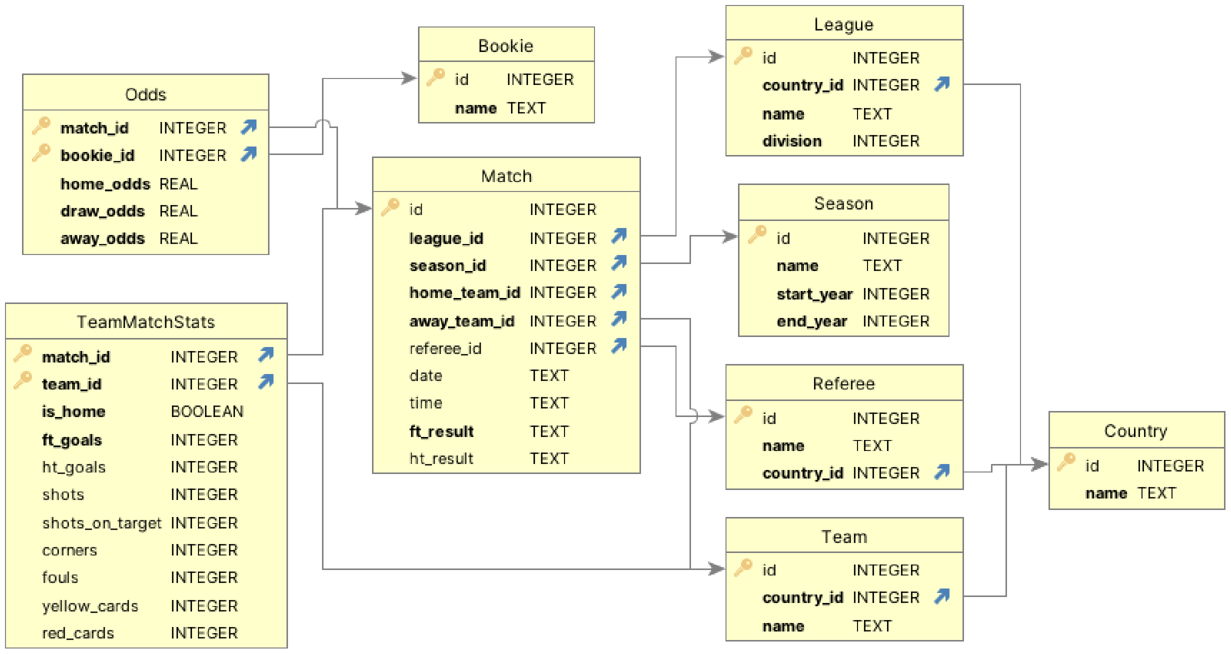

Figure 1: Final ERD for Database

Figure 1 shows the relationships between the data and how it is highly normalized. To get the full information about a match for example, one needs to JOIN with TeamMatchStats, Team, League and Season. The strong entities in the database are: Country, League, Season, Referee, Team, Bookie and Match. The weak instances are Odds and TeamMatchStats which both have composite primary keys comprising of two primary keys of other tables. There may be some minor adjustments to this table including adding indices and/or slight field requirement changes as I work through the product. Thanks to my scripts it is easy to change the schemas and reproduce the DB from scratch quickly. Currently the SQLite DB file is about 18MB.

Web Application

Currently I am planning to display this data to the user via a simple web application called Soccer Club Explorer. The idea of the application is that a user can query the database to discover predefined statistical information or trivia-like facts about the selected club via the user interface. The queries will all be prewritten, and the team will be injected into them based on the user's selection. Then the user will be able to query for things like:

- Head-to-head record with different opponents
- Most friendly/hostile referees
- Highest/lowest scoring matches
- Team performance over time
- Biggest upset wins/losses according to odds
- Possibly more…

Because all of the queries are read-only from the user, I am happy using a SQLite DB since no user will need exclusive locks on the data. I will also add filters for different queries, such that users can select only certain seasons, months, opponents etc. as appropriate. These will also be predefined on the front end and injected into the queries. This way I am only writing 10-20 complex queries, but they can be easily altered slightly by the user to get interesting data. The final component of the web application is visualization.

Figure 2: Rough application wireframe

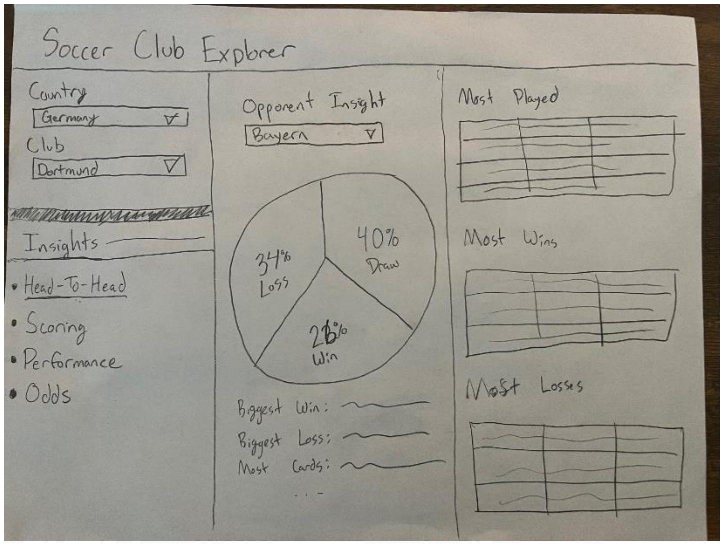

Figure 2 shows a rough wireframe of the web application. As you can see a team is selected in the top left corner below the header of the page. When a team is selected, various “Insight” categories will become available, each of which gives different information about the team selected. In this example, “Head-To-Head” is selected, which gives the user the ability to look at a direct statistical comparison between the team and a selected opponent. On the same screen is tabular data displaying the opponents that have been played against most, beaten most, and lost to most. This will be refined as I continue the project.

Web Application Technologies

I plan to use the following technologies to make the application functional:

- Streamlit: Python framework to create data web apps with lots of builtin tools
- SQLAlchemy: Python library that acts as an ORM and allows for SQL statement execution for different databases, “raw” queries will be my primary SQL interface, I will not use the ORM
- SQLite: Database that is suitable for this project because of its read-only nature
- Heroku: Free hosting for application and SQLite DB

Here is my GitHub repo for the project: https://github.com/derek-watson14/3287-Semester-Project

Learning Goals

I hope to reinforce my learning about database design, complex querying, database indexing and possibly also database caching and use of DBMS. I have already spent a lot more time on database design as part of this project, and I assume I will refine the design as I know the queries that I will be writing better. I expect I will have to write some very complex queries with lots of joins and aggregations as part of the front end development of this project as well, because the structure of the database is very normalized, experimenting I found myself writing dense queries like below:

```sql
Select
    Match.id,
    Country.name as "Country",
    League.name as "League",
    Season.name as "Season",
    Match.time,
    Match.date,
    HomeTeam.name as "Home",
    AwayTeam.name as "Away",
    Match.ft_result,
    HMS.ft_goals as "Home Goals",
    AMS.ft_goals as "Away Goals",
    Bookie.name,
    Odds.home_odds,
    Odds.draw_odds,
    Odds.away_odds
from Season
join Match on Match.season_id = Season.id
join League on League.id = Match.league_id
join Odds on Odds.match_id = Match.id
join Bookie on Bookie.id = Odds.bookie_id
join Country on Country.id = League.country_id
join Team HomeTeam on HomeTeam.id = Match.home_team_id
join Team AwayTeam on AwayTeam.id = Match.away_team_id
join TeamMatchStats HMS on HMS.match_id = Match.id and HMS.is_home = 1
join TeamMatchStats AMS on AMS.match_id = Match.id and AMS.is_home = 0
where (ft_result = "A" and away_odds > 10)
and AwayTeam.name = "Chelsea"
order by "Away Goals" DESC;
```

Finally, I suspect that I may eventually add some indices into the database to make queries more efficient, reinforcing my knowledge about that aspect of the course. Overall, I think there will be a lot to learn from diving into a project like this.