

English Premier League Data Visualization and Statistical Analysis

Zoe Olson and Kyle Holmberg

A documented journey in exploring database design, data-driven web design, and interactive data visualizations utilizing statistics from the English Premier League.



<https://github.com/kylemh/FPL-DataVisualization>

<http://ix.cs.uoregon.edu/~zoeo/>



Table of Contents

| Page | Topic |
|------------|------------------------------------|
| 3 ... | Project Summary |
| 4 ... | Logical Design (ER Diagram) |
| 4 ... | Physical Design and Table Contents |
| 5 ... | Database Applications |
| 6 | User's Guide |
| 6 ... | Implementation Code |
| 6 ... | Project Conclusion |

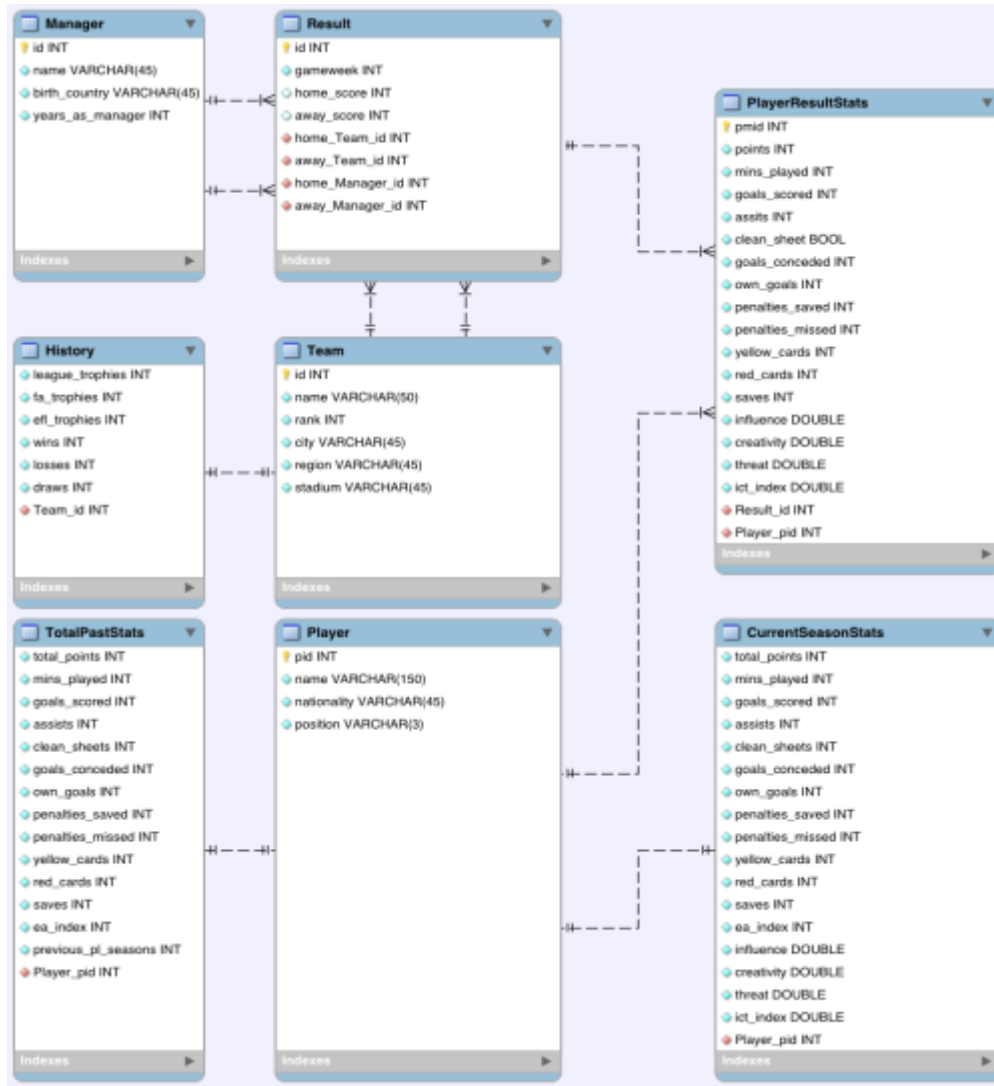


Project Summary

At a high level, this database will shape the general design of the English Premier League, the top, professional soccer league in the United Kingdom. There will be central tables for records of **Team** and **Player**, interacting via a **Result** table – the scoreline of a match. There are exactly 20 teams in the Premier League (PL) and a variable number of players per team (usually around 40). Each team has a manager stored in the **Manager** table. Teams also have their own corresponding **History** table to keep track of silverware and overall performance record. Players have both **CurrentSeasonStats** and **TotalPastStats** to track relevant performance data for both the current season and the sum of the player's past seasons in the PL, if there are any. Teams plan the matches and keep track of the score, but players play the matches and have stats kept for each match – those stats are kept in the **PlayerResultStats** table. The schema is designed to be able to provide database users the ability to draw relevant conclusions about the effects of certain variables upon a player's or team's performance. As a final note, Chris Wilson approved of Zoe Olson and Kyle Holmberg working together for this project. He also gave us his blessings to utilize this work in another class.



Logical Design



Physical Design / Table Contents

MySQL Dump:

https://github.com/kylemh/FPL-Data-Visualization/blob/master/fpl_mysql_dump.sql



Database Applications

Utilized Application:

d3.js Data Visualization –

Which countries have the most representative players in the English Premier League?

Potential Applications:

Various Data Visualizations –

Create a difficulty rating outlined by color codes and based upon the average ICT Index that a team possesses. For example, the team with the highest average ICT Index may be dark red; the lowest, light green.

Recreate the utilized visualization, but have the nation with the players that have the highest average EA Games Index be colored darker.

Queries –

Does Arsenal get more yellow cards when they play Spurs?

Which nationality proportionately gets the most red cards?

Which midfielder had the highest median threat rating while playing away in the first five gameweeks?

Overall application usage:

The shared purpose of every application example is to make the best choices for betting and/or for playing Fantasy Premier League – visit fantasy.premierleague.com for more info.



User's Guide

Go to ix.cs.uoregon.edu/~zoeo and enjoy our data visualization and data query field.

To examine the database, please use guest identified by guest on port 3640.

Implementation Code

<https://github.com/kylemh/FPL-DataVisualization>

Conclusion

This project was very enlightening towards the end as the database seemed to... highlight its own issues and fast-track our normalization efforts. The hardest aspect of this project was perfecting our database design, but the plethora of query types and data visualizations we *planned* on doing never came to fruition because the scraper's completion was hinging upon the completion of the database design. Working with Chris, reading out of the Head First PHP & MySQL book, and getting assistance from Database Designers in the DevChat Open Slack channel proved invaluable. Although, our project lacks many queries to the database, the design is such that many advanced queries could run without issue. The only aspect of our database design that we were not able to figure out is how to properly give teams ownership of players, but we are able to make statements that connect players to their correct team.

