National Basketball Association (NBA) Mini-Database Proposal

Group Members: Peter Mah, Jason Paul, Thomas Yee

Abstract

The main goal of this project is to create a mini-database for the National Basketball Association (NBA) league which includes the statistics (stats) for teams, players, and games. Millions of people watch the NBA daily with a big percentage interested in stats of teams and players. A user will be able to search the database for their favorite teams or players in order to see their stats. An entity relationship diagram will be created to display the relationship between the entities (teams, players, etc.) in the database. Next, a relational model will be created to represent the database as a collection of relations. This model will showcase how the data will be stored in the database and how the tables relate to each other. Finally, a functional model will be created to show the functions of our system. The frontend will be created using PHP and the database with MySQL. This project will further improve our understanding of database design, query writing, and API development.

Scenario and Goals

The NBA mini-database will store basketball sports statistics for all players and teams in the league over the period of October 2017 - March 2020.
Users of the NBA mini-database will be able to interact with the PHP based user interface (UI) to perform simple operations such as:

- Look up statistics for all players in a single team
- Look up all statistics for all players in a past game
- Look up all statistics for a single player over their career
- Filter results by season
- Compare multiple players’ statistics on the same table
- Identify minimum/maximum values for all players in the league for a given statistic

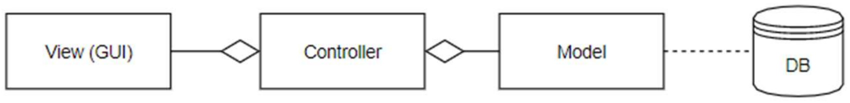
See Appendix A for an example use-case scenario of a user searching the database for all players on a particular team, filtered by one season, sorted by player position.

Note, the number of different types of statistics available to make a comprehensive basketball statistics table may lead to potential size and speed constraints. Therefore, the mini-database in this project will include only a few common statistics for brevity.

Design Strategy

The NBA mini-database will follow the Model View Controller (MVC) design pattern. The main components of this pattern are:

- Model: Provides the means to make queries from the database.
- View: The interface to the user, in our case a graphical User Interface (GUI). It is an interactive display that the users will use to make their requests to the database so that they can view the statistics.
- Controller: Contains the control logic. It translates the user-generated events from the GUI into database queries sent to the model.
- Database: This will contain all of the data pertaining to the basketball teams, players, games, and statistics. Only the Model component interacts with the database.



MVC Design Pattern

The Model and Controller may be implemented using Java due to the team's familiarity with this programming language, but we would like to explore the possibility of using PHP for the GUI. We could also use Java for all of the components, although we understand this approach may not be acceptable to the instructor. We are open to recommendations from the course instructor/TA's on how best to approach this due to our limited experience with choosing the best technologies for an application.

The design may also implement a client-server architecture so that multiple clients can connect to the database at the same time, although the scope of this project may not require this addition. If we decide to forgo the client-server approach, we may be able to use SQLite rather than MySQL in order to simplify the database.

Design Unknowns/Risks

A few aspects of the project are difficult to estimate time/effort due to inexperience/little experience working with the required programming tools. Data will be scraped from the web however many of the tables readily available on the web may contain a large quantity of irrelevant data to this project. Filtering and cleaning through the scraped data for relevant data may require extended effort. The final area of concern in this project is the front-end design using PHP. Due to unfamiliarity with the scripting language, the team may experience a steep learning curve to begin working on the front-end design.

Implementation Plan & Schedule

The first step will be to acquire the data, which will include web scraping or downloading the data and cleaning it. Next, we will finalize the Entity Relation Diagram (ERD). We will also complete draft versions of the Relational Model and Functional Model shortly thereafter in order to provide enough project definition to proceed with creating the database and starting the coding of the other components. The database will be created next since it only requires having the data. After creating the database, we will complete the Model and Controller components and finalize the Relational model. The Front-end (GUI) and Functional Model will be completed last.

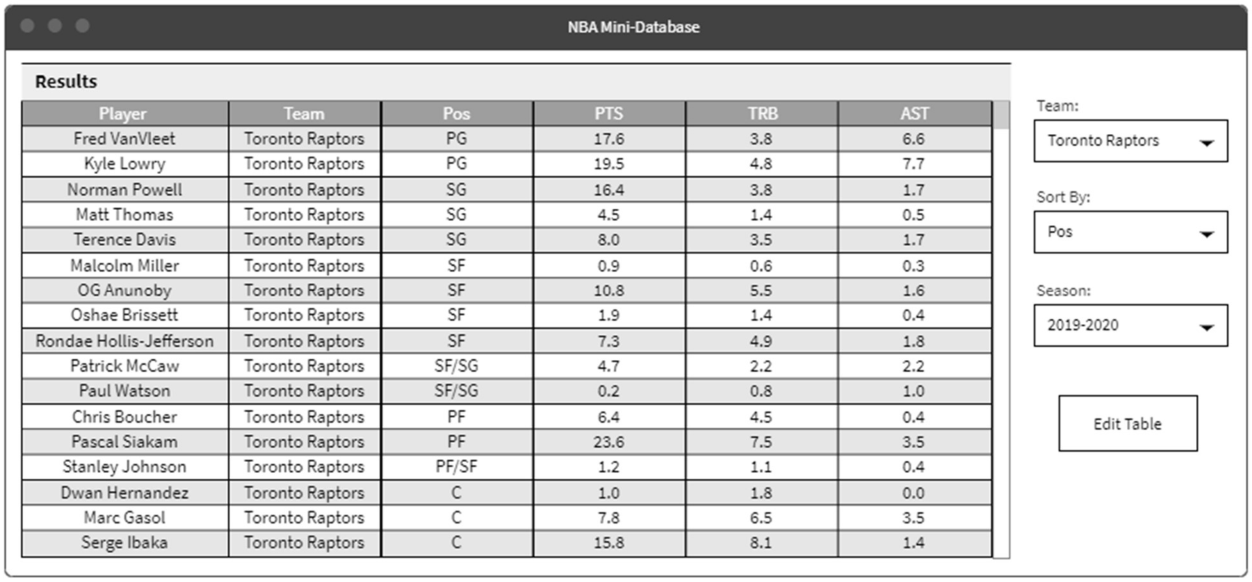
It is also likely that we’ll use an iterative approach to development, whereby we start with a smaller subset of the data just to get going on building a working version with minimal functionality in order to test the feasibility of our architecture. In this approach, we will each likely start working on all coding components simultaneously. Once we have a working system, we would then build on the complexity with the full dataset and functionality. Testing will be conducted extensively with the iterations. GIT will also be used for version control.

Milestones will be completed per the tentative deadlines proposed in Appendix B.

Evaluation

Our success will be evaluated based on our progress reports and final product. If we can follow the schedule outlined in our implementation plan outlined in Appendix B, we will be able to finish before the final deadline. Our program will also undergo tests and whether they are passed will be another factor for success. A fully functional program with no major complaints from the user will satisfy our objectives for this project. The following tradeoff will be evaluated: PHP GUI compared to a Java GUI. More trade-offs will likely be evaluated as the project progresses.

Appendix A: User Interface Example



Player	Team	Pos	PTS	TRB	AST
Fred VanVleet	Toronto Raptors	PG	17.6	3.8	6.6
Kyle Lowry	Toronto Raptors	PG	19.5	4.8	7.7
Norman Powell	Toronto Raptors	SG	16.4	3.8	1.7
Matt Thomas	Toronto Raptors	SG	4.5	1.4	0.5
Terence Davis	Toronto Raptors	SG	8.0	3.5	1.7
Malcolm Miller	Toronto Raptors	SF	0.9	0.6	0.3
OG Anunoby	Toronto Raptors	SF	10.8	5.5	1.6
Oshae Brissett	Toronto Raptors	SF	1.9	1.4	0.4
Rondae Hollis-Jefferson	Toronto Raptors	SF	7.3	4.9	1.8
Patrick McCaw	Toronto Raptors	SF/SG	4.7	2.2	2.2
Paul Watson	Toronto Raptors	SF/SG	0.2	0.8	1.0
Chris Boucher	Toronto Raptors	PF	6.4	4.5	0.4
Pascal Siakam	Toronto Raptors	PF	23.6	7.5	3.5
Stanley Johnson	Toronto Raptors	PF/SF	1.2	1.1	0.4
Dwan Hernandez	Toronto Raptors	C	1.0	1.8	0.0
Marc Gasol	Toronto Raptors	C	7.8	6.5	3.5
Serge Ibaka	Toronto Raptors	C	15.8	8.1	1.4

Team: Toronto Raptors

Sort By: Pos

Season: 2019-2020

Appendix B: Proposed Project Schedule

Task	Deadline	Responsible
Scrape data from web	March 13, 2020	Peter, Jason, Thomas
Entity Relationship Diagram	March 15, 2020	Peter, Jason, Thomas
Clean raw data	March 18, 2020	Thomas
Create functional database	March 20, 2020	Jason
Backend completed	March 23, 2020	Peter
Relational Model	March 29, 2020	Thomas, Jason
Frontend completed	April 10, 2020	Peter, Jason, Thomas
Functional Model	April 12, 2020	Peter, Jason
Final Report	April 15, 2020	Peter, Jason, Thomas