Final Score (Team 105)
Shivam Syal, Satyam Singh, Sameer Komoravolu, Yash Mandavia
CS 411 SP24 Final Project Report

1. **Please list out changes in the directions of your project if the final project is different from your original proposal (based on your stage 1 proposal submission).**

Our final project has effectively accomplished our goal to "use [NBA L2M data] to identify biases in referees, stadiums, matchups, and more." Through our various advanced queries and tabular data visualizations, Final Score users are able to identify various refereeing trends for their favorite players and more easily parse through the currently unusable NBA L2M website. Although we didn't get the chance to implement all the creative components we intended on adding to Final Score, our final project still achieves our initial goal. As we discuss later in this report, we hope that future development can further improve Final Score to allow basketball fans to dive deeper into the rather untapped and overlooked world of refereeing in the NBA.

2. **Discuss what you think your application achieved or failed to achieve regarding its usefulness.**

One primary stand-out feature when compared to other basketball stat websites was that most other analytical platforms focus on player stats and efficiency metrics instead of officiating. To our knowledge, this is one of the first attempts to break down referee data and add elements of visualization and readability to it. This specific focus allows us to provide a more in-depth analysis than more general sports analytics tools. Not only that, but our efforts to implement our "fun facts" section on the home page allows for users to learn new, interesting quirks about the NBA that give a new perspective on the game for diehard fans and fanatics. From a personal standpoint, all of our team members were pleasantly surprised when we obtained the results to our SQL queries as we ourselves were able to learn more about the teams, players and game we love. One area of improvement for Final Score is definitely the visualizations; although a tabular method is definitely way more useful than the current NBA L2M website, implementing a more interactive and visually appealing data visualization method would've been ideal for the context of our website. This was another area of uniqueness we felt Final Score would have, but this wasn't fully present in the final version of our project due to constraint issues. Data visualization requires a more optimized query and state management system than the current version of our website, which would only be truly possible with an ORM.

3. **Discuss if you changed the schema or source of the data for your application.**

Although we ultimately changed some data types for a few variables across databases, we didn't change our schema or alter any critical components of our database or tables. We also didn't change the source of the data for our application; we ended up successfully using the NBA L2M Data.

4. **Discuss what you change to your ER diagram and/or your table implementations. What are some differences between the original design and the final design? Why? What do you think is a more suitable design?**

As mentioned earlier, we didn't change the ER diagram; however, the table implementations were a bit altered. Primarily we changed the team_id field's data type from an INT to VARCHAR as it made more sense for the context of our data to use the abbreviated team codes over random primary keys.

5. **Discuss what functionalities you added or removed. Why?**

As we built the initial prototype there were some functionalities we thought would be more useful to expand upon. We always intended on having a player search database, however we added a lot to it to make it the highlight of the project. In addition to allowing users to search for their favorite NBA players, we also added filters for different call types, referees, and more. Additionally, we added count features and filters for what specific season and the implications of a foul; so, whether it was a regular season or playoff game. To expand on this, we also provided a feature to manually add and remove players. The idea is if an NBA referee or user is using our app and finds an error, they can manually make changes as well. We made our trigger to expand on this to automatically replace null values or incorrect info with "PENDING" so admins of the app can go in and fix it laters.

A functionality we decided to remove was the data visualization aspect we initially planned on including. Since the player and referee search was by far the most important part of our program and due to the fact we expanded on it as well, we decided to leave out visualizations for these fouls. The functionality would have probably looked like a page with visuals and graphs based on the data we accumulated from the NBA L2M data.

6. **Explain how you think your advanced database programs complement your application.**

Our advanced database SQL queries deal with both the transactions and the fun fact section. The transactions ensure that every insertion and deletion operation on the Users table is atomic, rolling back if a user email is already present in the database. This is useful to ensure we don't get spam accounts associated with the same email. The fun facts section uses advanced SQL stored procedure queries to demonstrate features of the dataset that have likely never been seen given their relative obscurity, which can draw in more users. This is also a critical part of our project: providing these fun facts to users about critical insights on NBA refereeing is our main goal. Using stored procedures allows us to provide these obscure, unique fun facts efficiently and quickly so that multiple users can see the same data without much lag in server query times due to the high volume of data we currently maintain. Finally, our trigger provides a way for users to help moderate the website and improve its qualitative accuracy based on the Calls data we have at the moment. If there's a controversial ruling or an incorrect filing, users can remove a specific call's penalty ruling, causing the call to be marked as "PENDING" for all

pertinent data fields and allowing for an official or NBA employee with an admin email to edit the call's information with the right data. The trigger provides another layer of depth to the CRUD operations and introduces potential user/admin interaction.

7. **Each team member should describe one technical challenge that the team encountered. This should be sufficiently detailed such that another future team could use this as helpful advice if they were to start a similar project or where to maintain your project.**

Shivam - Our team had a relatively good idea of what tech stack we wanted to use when creating our project, but we didn't truly scope out the middleware connection issues we'd have with Next and Node. For me personally, I was one of the first team members to connect our backend and frontend which posed a real challenge to me as we weren't following the same method described in class due to personal preference and past experience with Next. One critical error we constantly ran into was proper state management and API endpoint creation within the application which would talk to our GCP database through Promise requests. Handling various random database callback errors throughout development really slowed us down when we could've spent more time writing more advanced queries or improving the frontend.

Satyam - Instead of using GCP, we decided to use Next.js and Node.js, which had many pros but also brought with them a small learning curve. We found ourselves spending more time on the front and backend of the application than on the actual SQL queries. One specific problem that arose frequently was tying the backend functionality to the SQL queries. In JavaScript, this was a bit awkward at times since there were many small things to take into consideration, such as using execute instead of query. From this struggle, we realized it would have been best to thoroughly review the documentation to ensure that integrating SQL queries into the application would be a whole lot smoother.

Yash - Going off what Satyam said, we decided to make a full-stack application using Next.js and Node.js which added complications to the project. Integrating the database we hosted on GCP into the app was tedious and took a while. At times, our application even stopped working due to high CPU usage or even forgetting to manually input the IP addresses into GCP. These were some general problems we had with the setup process. In terms of issues relating to my contributions, there was a bigger learning curve when it came to using SQL on VSCode than I originally expected. A lot of seemingly well-written SQL code had to be changed and edited to fix all the syntax errors. More specifically, when it came to the stored procedure and triggers, a lot of the working syntax was different from how we learned it in class.

Sameer - During our initial stages of design, we drafted our datasets from the two minute NBA reports, which detailed a lot of information about calls that referees made and about players. Shivam pulled and cleaned the data from 2015 to 2023, but since our schema required

information about users and the searches they made, we didn't know how to get this information without manually entering fake data at first. This process could be automated, so I developed a python script that utilized Numpy to generate fake user data at first. Since it would still take some work to modify this script for every dataset we needed to fabricate, we were able to generalize the script to take in column headers, header types, and number of entries desired and output the entire CSV file. This ended up working quite well, building the user and search information that remained consistent with real data.

8. **Are there other things that changed comparing the final application with the original proposal?**

As mentioned earlier, a big difference we had was bolstering the player search and filter features. However, this came at the expense of the data visualizations in the form of assorted graphs and patterns on a separate page of our app. Another change we made was not including the use of a neutral network in our project. We initially planned on using the XGBoost framework to add a machine-learning feature to use our data and make predictions. However, we decided against this because it was a big time commitment and also would not have added too much to the final result. The data filtering features we added already serve the purpose of helping find trends in addition to our advanced queries and fun facts page. Consequently,y we decided against including a machine learning aspect despite briefly mentioning it on our initial proposal.

9. **Describe future work that you think, other than the interface, that the application can improve on**

During development, we ensured that our roles were well defined. The CRUD operations and majority of the UI were handled by Shivam Syal and Satyam Singh. This included the front end and back end required to allow the user to interact with the database. We then gave the responsibility of using the advanced queries to create stored procedures, triggers, and transitions to Yash Mandavia and Skomo. We all had the responsibility of creating one of the advanced queries so that work was divided equally. All work was well distributed, and all team members contributed equally, and wherever someone needed help, another teammate was able to fill in the gaps.

10. **Describe the final division of labor and how well you managed teamwork.**

Going forward, we would go back and fulfill features that we were looking to implement in our proposal. This would include things like data visualization in the form of interactive graphs or the use of neural networks to better analyze data. While both of these features may be outside the scope of the class, they would work well with the purpose behind the website and help improve user experience. As mentioned in the proposal, we could have used the G3 data visualization library to translate our query results to a graph that makes the data easier to see.

The neural network part of our initial proposal would require more research before development but would drastically improve the functionality of the site.