Groupname: WangSHuangJ
Group members: Jiansong Huang, Shengdi Wang

README

Configure and run the application

First, **import the database schema** dump to re-create the database. As an example, you can take the following steps if you are on MacOS.

1. Open MySQL Workbench.
2. Click on “Sever” tab and choose “Data Import”.
3. Choose “Import from Self-contained File” and select the database dump.
4. Click “Start Import”.

Be sure to create a user on the MySQL server with the **username “java” and password “password.”** And grant all edit access to all parts of the imported database.

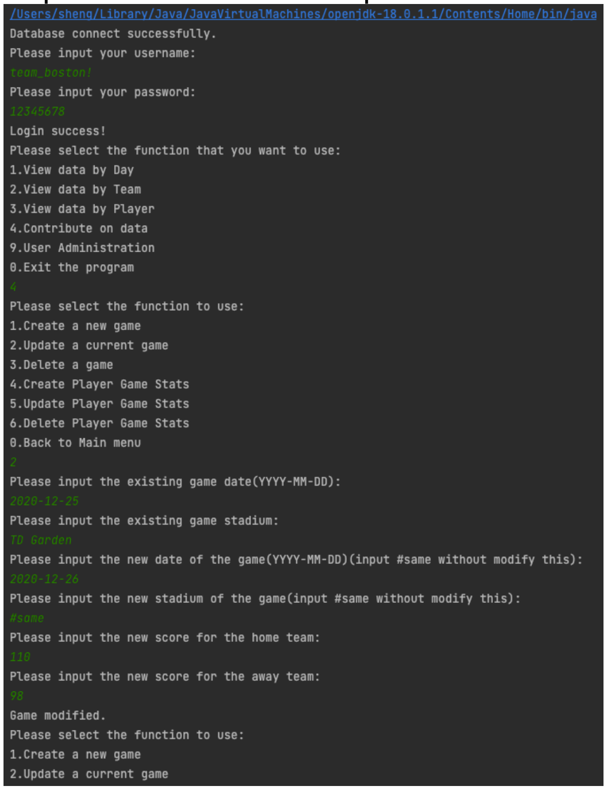
Then, navigate to the folder with finalProject.jar file in your terminal/command line prompt, and run “**java -jar finalProject.jar**” in the terminal to start the application.

Example user flow

We provide an example user interaction with the application.

1. First, provide the username and password. E.g. you can use the pre-configured username “team_boston!” and password “12345678”.
2. Choose the intended function to use by choosing relevant numeric options.
3. Enter detailed data as appropriate.

A screenshot of the example user transaction is provided.



```
/Users/sheng/Library/Java/JavaVirtualMachines/openjdk-18.0.1.1/Contents/Home/bin/java
Database connect successfully.
Please input your username:
team_boston!
Please input your password:
12345678
Login success!
Please select the function that you want to use:
1.View data by Day
2.View data by Team
3.View data by Player
4.Contribute on data
9.User Administration
0.Exit the program
4
Please select the function to use:
1.Create a new game
2.Update a current game
3.Delete a game
4.Create Player Game Stats
5.Update Player Game Stats
6.Delete Player Game Stats
0.Back to Main menu
2
Please input the existing game date(YYYY-MM-DD):
2020-12-25
Please input the existing game stadium:
TD Garden
Please input the new date of the game(YYYY-MM-DD)(input #same without modify this):
2020-12-26
Please input the new stadium of the game(input #same without modify this):
#same
Please input the new score for the home team:
110
Please input the new score for the away team:
98
Game modified.
Please select the function to use:
1.Create a new game
2.Update a current game
```

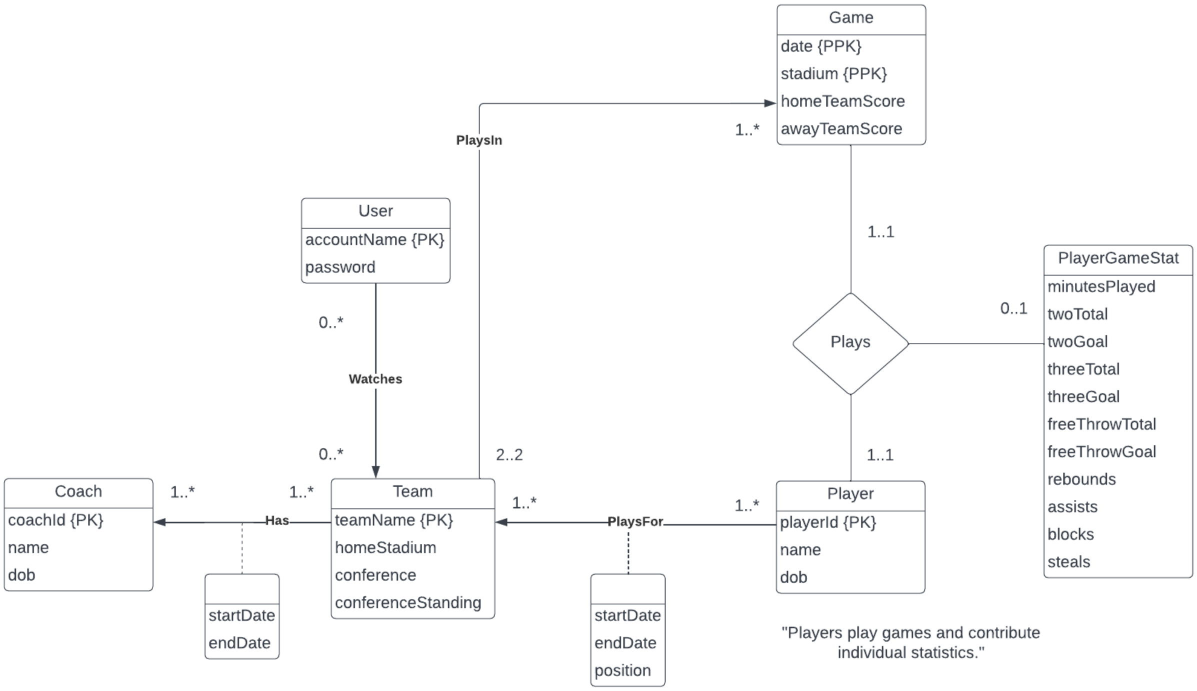
To perform other modifying transactions, be sure to use the view function to understand the existing data records first.

Technical Specification

Used the relational MySQL database as the DBMS for the project. Specifically, the server version used is MySQL Community Server 8.0.31.

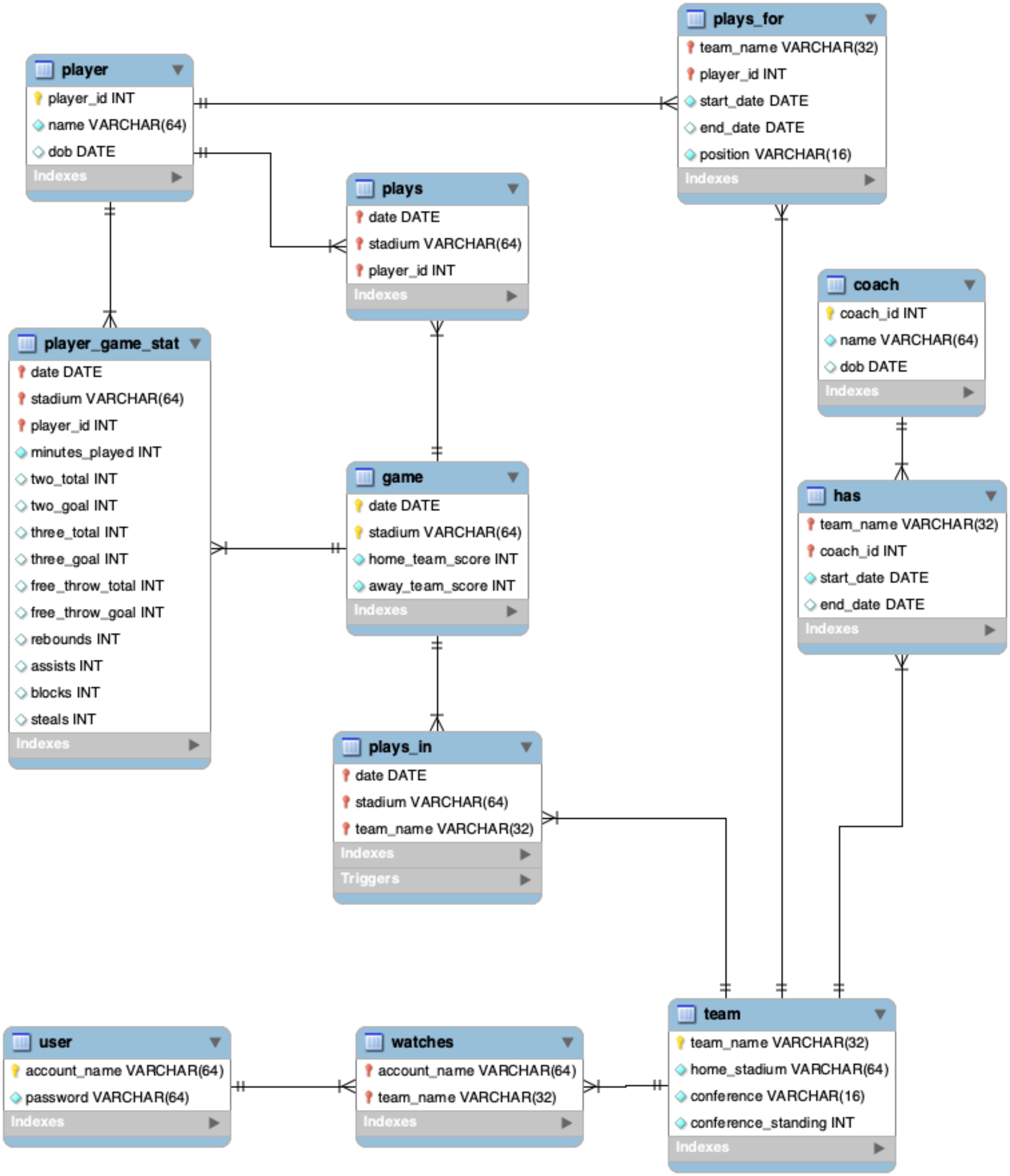
Used Java 17 to develop both the user interface and the back-end logics of the application and JDBC as the DBMS connector.

Final Conceptual Design

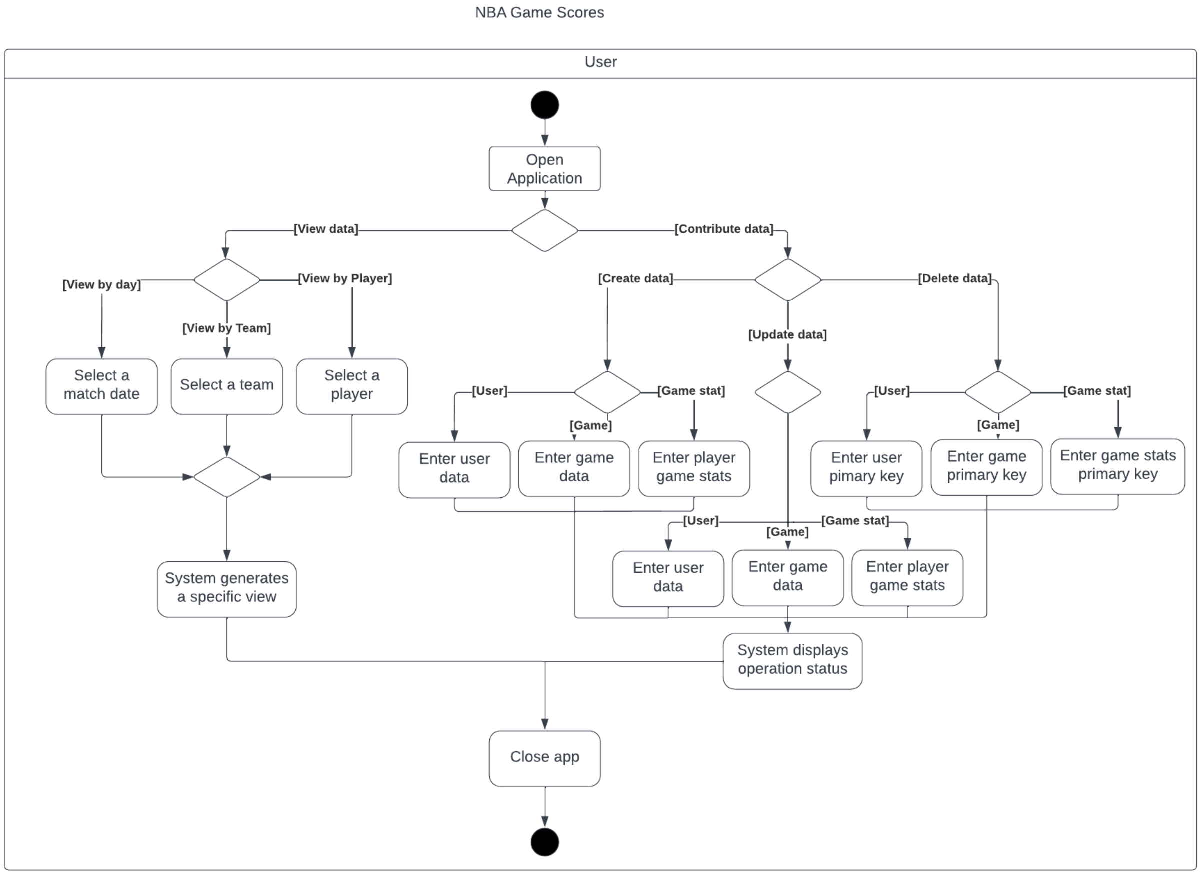


"Players play games and contribute individual statistics."

Final Logical Design



Final User Flow



Lessons Learned

Technical expertise gained

To support the CRUD operations for the application, the team dived deep into various aspects of MySQL, including Data Definition Language, procedures, functions, triggers, queries, etc. The project offers an opportunity for the team to apply and solidify our learnings from the coursework.

At the same time, the team built a substantial number of features, including models, views, controllers, Data Access Objects (DAOs) using the Java programming language. This was a great learning experience for both Java-related techniques and typical design patterns such as DAOs.

Insights, time management insights, data domain insights etc.

It was helpful that we started the project early in the semester as we prepared for the project proposal. This gave a more confidence in the direction we are taking, since the proposal received positive feedback from the grader. It also gave us a foundation to work with when developing the programs.

If we are to do this project again, we might want to start on the coding portion of the project earlier. Since the project deadline is close to Exam 3, as well as final project deadlines from other courses, we were under substantial time pressure toward the end. This could be avoided if we started a few weeks earlier.

It is interesting to see that seemingly simple queries with real-world significance, such as finding out the basketball player game statistics, can quickly become complex when multiple tables need to be built and normalized. We had to do multiple joins and take a long look at the schema to formulate such queries. We are now better able to appreciate the complexity and value of database programming.

Realized or contemplated alternative design / approaches to the project

One of the technical decisions is to decide whether to place a logic in the database procedures or Java programs. In most cases, we realized the higher likelihood to reuse codes if the logic is placed in database procedures. Therefore, for most operations, such as finding creating new game records and updating user information, we created SQL procedures, and called these procedures from within the Java programs.

We also decided to go with the traditional relational MySQL database, instead of NoSQL databases such as MongoDB. This is primarily because the application does not require a distributed database at the moment due to the small amount of data. Relational databases also allow us to have more consistent data which gives us confidence about the query results we produce.

Document any code not working in this section

All codes are working in this project.

Future Work

Planned uses of the database

Users can query for NBA game statistics, such as player game statistics, NBA team information and standing, as well as application users. This serves to satisfy the curiosity of some NBA fans, such as the project authors themselves.

In addition, the database is intended as a destination for crowdsourcing abovementioned data. Therefore, qualified users can be allowed to modify the game and player statistics to ensure they are up to date. For example, users can create/update/delete a player’s game statistics when such information is available.

Potential areas for added functionality

One area that can be improved is to add an authentication system for users. Since the users are given access to modify parts of the database, having an authentication system, or even an access control system, is valuable to ensure that only qualified individuals can modify the database.

Another way to maintain the health of the database is to add administrator roles. Admins will be given access to modify all areas of the database, and rectify errors as they occur during the database operations.

Also, additional fields in the database can be auto updated, such as the team's conference standings, termination of a player's contract status (as shown in the *plays_for* or *has* tables). This can be achieved by understanding relevant business rules and adding triggers to the database.