

Team Members: Jakob Danninger (JKD50), Beiming Lian (BXL606), Chan Kang (cxk481)

Database Name: Grand Slam Stats

Problem Statement:

Baseball, the old ball game, has an interesting quirk: statistics strongly matter! Modern baseball is dominated by statistics, batting averages, strikes, and ERA are just a few of the key metrics used to analyze the game. This fact was highlighted in the best selling book *Moneyball: The Art of Winning an Unfair Game*, which documented how the Oakland Athletics used statistics and empirical data to build a dominant baseball team on a small budget. Unfortunately data is not always available when it comes to amateur leagues . . . we intend to change that. Our team wants to make an intuitive tool, called Grand Slam Stats, for teams and fans alike to track smaller leagues and be able to record and retrieve statistics. This will be done by creating a UI tool that allows users to either enter new game information into our database or retrieve statistics about their favorite teams and players which is derived from our database. In conclusion, we hope that Grand Slam Stats can help baseball lovers better track their favorite minor league teams and players.

Table of Contents

1. Setup Guide
2. Database
 - a. ER Diagram
 - b. Explanation of ER Diagram
 - c. Normalization, Indexing, Procedures, and Committing
3. Use Cases
4. User Manual
5. Reflection

1. Setup Guide

Grand Slam Stats uses Microsoft SQL Server for the database, Java for the front end, Swing (the default Java GUI library) for the GUI, and JDBC to connect to the database.

If you are using the CSDS341 virtual machine you can connect to our version of the database using the following connection string (this is the default in our program and should natively work on the CSDS341 VM):

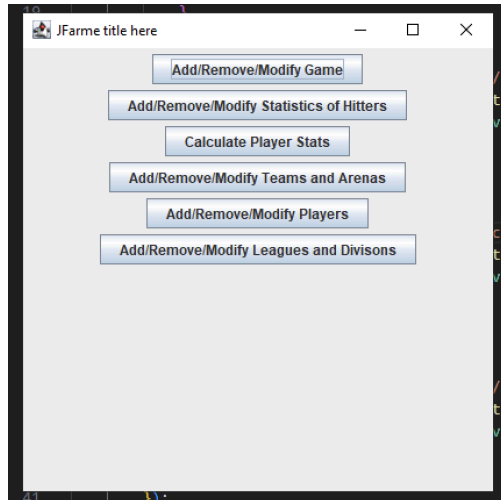
```
private String connectionString =  
    "jdbc:sqlserver://cxp-sql-02\\jkd50;"  
    + "database=GrandSlamStats;"  
    + "user=db_user;"  
    + "password=ThisIsANewPassword123@"  
    + "encrypt=true;"  
    + "trustServerCertificate=true;"  
    + "loginTimeout=15;"
```

Or you can create your own local version of the database. To create your own local version of the database, create a new database called GrandSlamStats and run the DDL.sql file. This file will set up the structure of the database, procedures, and the login details. Finally replace the connection URL in the connection String to your database for all connection strings.

In order to add mock data you can use our .bak file or you can run Populate Tables.sql.

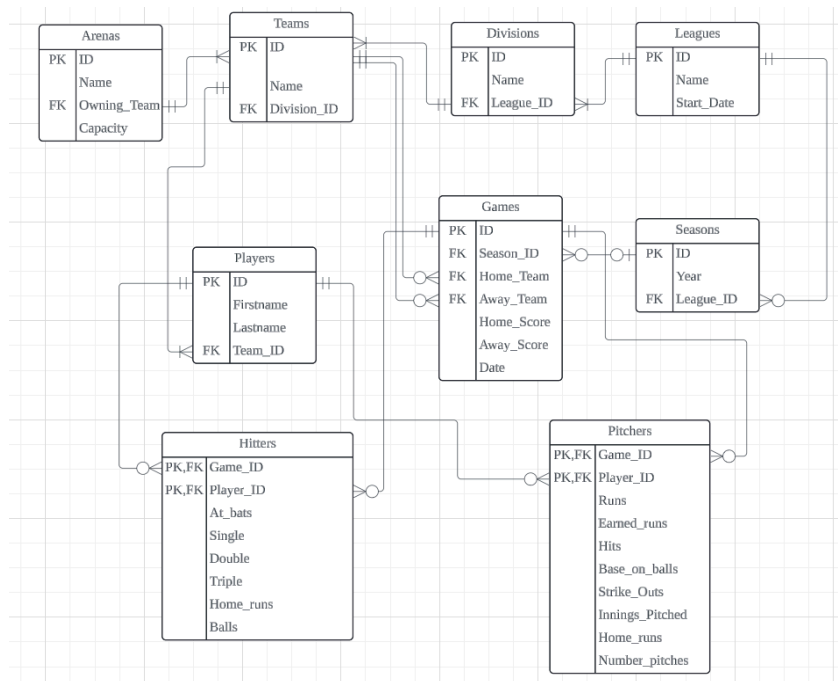
Finally to run the Java program run Main.java (this is where our UI is built).

If all works correctly the following UI should be launched



2. Database

i. ER-Diagram



ii. The explanation of the tables

The first table is the Leagues table which allows the user to have multiple leagues they are tracking. Each league can have zero to many seasons and each season can have zero to many

games. Games should be assigned to a unique season but does not always have to be assigned to a season since a game could be preseason or off season.

Leagues will also have at least one division since you cannot have a league without at least one division of teams. Divisions need to have teams so they have a one to many relationship with teams. Each team must be part of a division. Each team must have an Arena they play at but arenas can have multiple teams, for example in the 80s the Oakland Athletics and Oakland Raiders shared a MLB arena. Teams can face off in games which take on a home team and away team. Each team can play multiple games to no games.

Each player must be part of a team and teams need at least one but can have many players. Each time a player plays a game they can get a Hitter and or Pitcher stat which stores how well that player did during that game given the role of Hitter or Pitcher.

iii. Normalization, Indexing, and Committing

Indexing:

Indexing is used to be able to quickly reference a column of data. The following indexes are used in our table

```
CREATE INDEX fn_idx ON Players (Firstname);
CREATE INDEX ln_idx ON Players (Lastname);

CREATE INDEX hitter_idx ON Hitters (Player_ID);
CREATE INDEX pithcer_idx ON Pitchers (Player_ID);
```

First name and last name are commonly used in our program especially for drop downs and are thus indexed and the Player_ID is commonly searched for in order to calculate statistics and thus it is indexed.

```
]create index team_idx on Teams (Name);  
  create index division_idx on Divisions(Name, League_id);  
  create index arena_idx on Arenas (Name, Owning_Team);
```

For better reference, we also created indexes for teams, Divisions and Arenas for better visibility.

Normalization

Our table adheres to the first normal form since every column has a unique piece of information and cannot have multiple pieces of information. All of our tables except Hitter and Pitcher are second normal form since they have ID's for their primary key, but Hitter and Pitcher have composite keys. Hitter and Pitcher are fully functionally dependent and thus adheres to the second normal form.

Procedures

All queries are handled using procedures. If you look at the DDL.sql file you can see all the procedures we used.

Committing

After every insert statement in the front end that we want added to the database we use `connection.commit()` in order to finalize that change. We did not have use for rollbacks but they should work if we wanted them in the future. Here is an example of after an insert we commit:

```

try (Connection connection = DriverManager.getConnection(connectionUrl))
{
    String sql = "EXEC InsertPlayer @FirstName = ?, @LastName = ?, @TeamName = ?;";
    PreparedStatement preparedStatement = connection.prepareStatement(sql);
    preparedStatement.setString(1, fn);
    preparedStatement.setString(2, ln);
    preparedStatement.setString(3, team);

    int rowsInserted = preparedStatement.executeUpdate();
    connection.commit();
}

```

3. Use Cases (for all team members)

Use Case 1 (Beiming): Add/Remove/Modify a New Game to the Games Table

Description: "In managing game records, an administrative user can add, modify, or remove game entries in the 'GrandSlamStats' database. The addition of a new game record requires inputting details such as the season, the names of the home and away teams, scores, and date. This process involves resolving team names to their respective IDs, and the season description to its ID, using selection statements on the 'Teams' and 'Seasons' tables. Once the IDs are obtained, a game entry is created in the 'Games' table using an insert statement through a stored procedure, which ensures that team and season IDs are correctly associated with the respective scores and date. For modification, a similar procedure is followed, but it uses an update statement encapsulated in a stored procedure to alter the game record specified by the game ID. To delete a game, a remove operation is performed via a stored procedure that executes a delete statement. Confirmation of these operations can be validated by selecting and displaying the relevant game records post-modification or addition, ensuring that the database reflects the current and accurate state of game records."

Use Case 2 (Beiming): Add/Remove/Modify Player Hitter Statistics to a Game

Description: In this use case, an administrator or authorized user has the capability to track and update individual player statistics for each game within the 'GrandSlamStats' database. The user interface allows the selection of a specific game and player, along with the input of statistical data such as at-bats, singles, doubles, triples, home runs, and balls. Adding new statistics involves calling a stored procedure that inserts the provided data into the 'Hitters' table after the user inputs the information. This process includes the execution of an insert statement within the stored procedure, ensuring the statistics are correctly tied to both the game and player IDs. Modifying hitter statistics is accomplished via a stored procedure that updates the relevant records in the 'Hitters' table based on game and player ID, utilizing an update statement. Lastly, deleting statistics is done through a stored procedure that removes the specific entry from the 'Hitters' table with a delete statement, based on the selected game and player ID. After each operation—whether adding, updating, or deleting—transactions ensure data integrity and the final step is confirmed through a message to the user, verifying the action's success or providing an error if the operation failed.

Use Case 3 (Jakob): Calculate player statistics

Description: This is the ability to look up the most common statistics for each player for a given season. The statistics that can be calculated are as follows:

- Slugging Percentage (how many bases per at bat)
- On Base Percentage (how time a batter makes it to base per at bat)
- Strikeout per inning (how many strikeouts per inning pitched)
- WHIP (walks and hits per inning pitched)

To access this you can use the Calculate Player Stats window where you can select a player and stat and it will calculate it.

Use Case 4 (Chan): Add/remove/modify teams and their arena

Description: This use case describes the processes for adding, removing and modifying the teams in the league/divisions, and the home arena for the teams. When a baseball team is disbanded, we have to remove the team from the Team table. If a team changed the division or league, then we

have changed Division ID information. If a team changed the name, then we would like to change the name attribute of the team. Also, we will add home arenas for each team, and make sure we can update information about the arenas.

Use Case 5 (Jakob): Manage Players:

Description: We want to be able to add and remove players from the table. In order to do this, there is a drop down menu allowing the user to pick a team, then add player details and from there add the player. Or they can use a drop down menu to select a player to be removed from the database.

Use Case 6 (Chan): Add/Remove/Modify Leagues and Division

Description: We want to be able keep consistent updates on different Baseball leagues and the leagues' division information if the leagues have the divisions. If the division is removed from the league, we can remove that division tuple from the Division table. If a new division is added from the league, we can add a new division tuple to the table. Like this, we will track the changes of information in creation/updates/disband of leagues and divisions.

Use Case 7 (not implemented): Add/Remove/Modify Player Pitcher Statistics to a Game

Descriptions: This is the ability to add Pitcher performances for every game. For example if the Nationals play the Phillies and Stephen Strasburg pitches really well or poorly, we want to be able to record that for that game his performance in order to use it to calculate other stats later.

Use Case 8 (not implemented): Reorganize Divisions and Leagues

Descriptions

Use Case 9 (not implemented): Add/Remove/Modify Seasons

Descriptions: This is the ability to create new seasons, delete unnecessary seasons, and modify existing ones such as changing the Season year and what league it is assigned to.

Use Case 10 (not implemented): Calculate Team Statistics

Descriptions: This is the ability to calculate overall statistics for a team, such as that team's overall win/loss ratio, their batting average across all players, or their WHIP for all their pitchers. This allows the user to get a quick overview of how a team is performing. This involves querying the Players and Games table and from there referencing it against the Batters and Pitchers tables.

Use Case 11 (not implemented): Create a season overview statistics

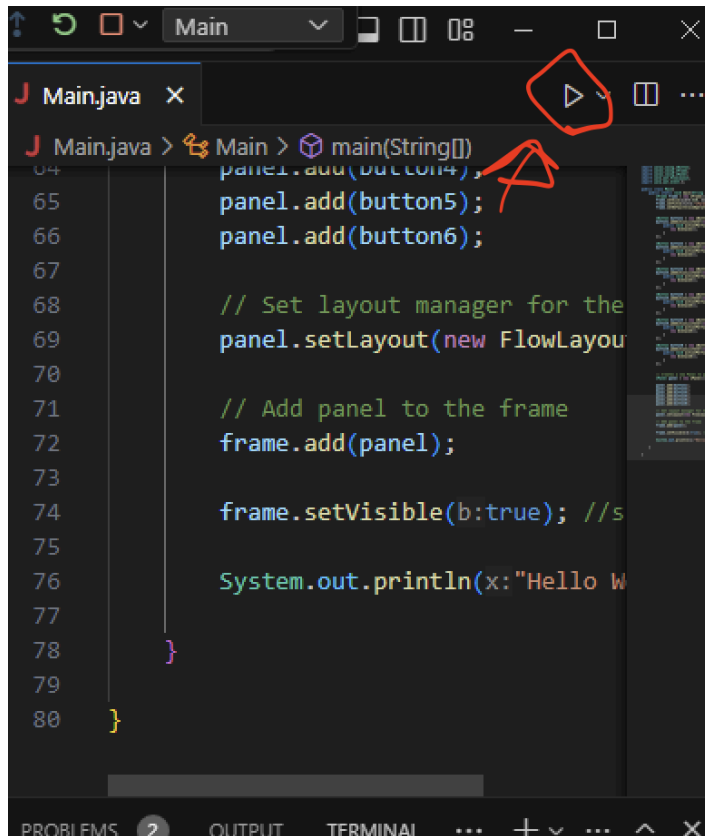
Descriptions: This is a menu that allows a user to calculate the overview of a season such as the overall win loss rate, the overall batting average and WHIP. This gives a high level summary of how a season went for all teams in general.

Use Case 12 (not implemented): Create a seasons player ranking

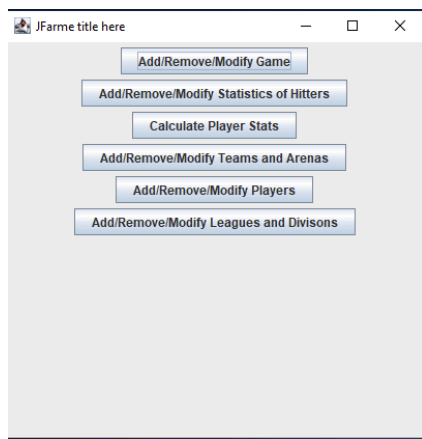
Descriptions: This is a new menu where the user can select a statistic/metric (such as WHIP) and receive a ranking of all the best players for that statistic, allowing them to figure out which players are doing the best.

4. User Manual

To launch the program run Main.java



After you run main the following home window should be displayed:



This menu allows you to access each one of our six test cases

- Add/Remove/Modify Game

Add/Remove/Modify Game

Add Game

Season: 2023 Home Team: Washington Nationals

Away Team: Washington Nationals Home Score:

Away Score: Date (YYYY-MM-DD):

Add Game

Modify Game

Game ID: 2 New Home Score: New Away Score:

New Date (YYYY-MM-DD):

Modify Game

Delete Game

Game ID: 2 Delete Game

- For adding a game, since the season and teams are foreign keys, you can't insert a new season or new teams here; instead, we select the seasons and teams from the existing data in our database. Besides that, you are able to edit scores and dates (they should be in the correct format and datatype).
- Since the game already exists, you can choose the game to modify or delete.
- Add/Remove/Modify Statistics of Hitters

Manage Hitters Statistics

Game ID: 2

Player ID: 1

At Bats:

Singles:

Doubles:

Triples:

Home Runs:

Balls:

Add
Modify
Delete

- This use case focuses on a specific player's statistics for a specific game. Since the game and players are foreign keys, you select them from our database, and then you can edit the values in the rest of the columns.

- Calculate Player Statistics

get statistics

Get Stats

Juan Soto WHIP

1.3333334 Calculate Stat

- Start by picking the player then the stat
- When you press calculate the bottom box will get the stat or if no data is available it will say not available

- Add/Remove/Modify Teams and Arenas

Add Teams

New Team Name
Divisions:
American League

submit

Update Team

Team:
Arizona Diamondbacks
New Team Name

Division Change:
American League

Update

Remove Teams

Which Team Do You Want to Delete?
Arizona Diamondbacks

Delete

Create New Arena

Team:
Arizona Diamondbacks
New Arena Name
Capacity:

Create

Change Arena Information

Choose Arena:
Citi Field
Make New Name:

New Team:
Arizona Diamondbacks

Modify Capacity:

Change Info

Delete Arenas

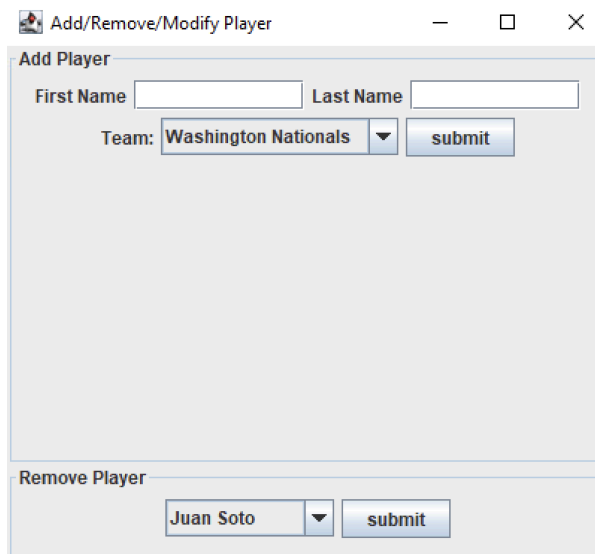
Arena to Delete
Citi Field

Delete

- If you want to add new teams, and go to add the team name in the New Team Name Section, select the division you want to add.If you want to change information about the existing teams, then go to the Update Teams Section. First, select a team you want to change, add a new name to the ‘New Team Name’ section, change the division that the team is belonging to, then click update. If you want to remove the team, then go to the Remove Teams section and select the team you want to remove from the box. Then click Delete.
- If you want to add a new arena, go to the ‘Create New Arena’ Section. Select the team that will own the arena, then add a name and capacity, and click ‘Create’. If you want to change the information of an arena, then go to ‘Change Arena

Information’. Choose the Arena you want to update, add a new name and select the new owning team of the arena (If the owning team doesn’t change, then select the team that is owning the arena currently). Finally, change the capacity then click ‘Change Info’. To delete Arena, just head to the bottom section, select the arena you want to delete, then click Delete.

- Add/Remove/Modify Players



The screenshot shows a web application window titled "Add/Remove/Modify Player". The window is divided into two main sections. The top section, titled "Add Player", contains two text input fields for "First Name" and "Last Name", a dropdown menu for "Team" currently showing "Washington Nationals", and a "submit" button. The bottom section, titled "Remove Player", contains a dropdown menu showing "Juan Soto" and a "submit" button.

- To add a player fill the first and last name boxes and select a team. After pressing submit the insert query gets run.
- To remove a player just pick them from the dropdown and press submit to delete that player.
- Add/Remove/Modify Leagues and Divisions

Add Leagues

New League
Starting Date(yyyy-mm-dd):
Create

Add Divisions

New Division Name
Leagues:
Major League Baseball

▼

submit

Remove Leagues

Which League Do You Want to Delete?
Major League Baseball

▼

Delete

Remove Division

Which Division Do You Want to Delete?
American League

▼

Delete

Update Leagues

Which League Do You Want to Update?
Major League Baseball

▼

New Name
New Date

Update

Update Division

Which Division Do You Want to Update?
American League

▼

New Name

Change the League If needed

Major League Baseball

▼

Update

- To add leagues, type in the new league's name, and the date the league is created.
To create a new division into a league, make a division name and select the league that division will belong to.
- To remove leagues and divisions select leagues or divisions, choose the league or division you want to delete from either box and click delete.
- To update, fill in the textfields and select the league/division you want to update from boxes.