Top Spin

Project Milestone 2
CIS 5500: Database & Information Systems
University of Pennsylvania

Members (email; github):

- Akanksha Ashok (aashok@seas.upenn.edu; @aashok12)
- Noah Capp (@noahcapp@seas.upenn.edu; @noahmcapp)
- Bilal Ali (bilalali@seas.upenn.edu; @bilala45)
- Peter Akioyamen (peterai@seas.upenn.edu; @peter-ai)

Project Description

Top spin is a web application that allows both tennis fanatics and casual fans to gain insight into the most infamous question in any given sport - who is the greatest of all time (GOAT). It aggregates data about athletes, rankings, and matches dating back to 1968, and match odds from 2000 to 2019. The web app will display various dynamic summary views and more in-depth pages about historical tennis players and matches. Aligned with this, it will provide users the functionality to create matches with players from different eras of tennis, and show the user who was more likely to win given the user defined parameters about the match.

Pages

Player page

- Tabular overview of all player data, with individual views for each player and their associated data (name, DOB, height, dominant hand, career winning percentage, etc.).

Tournament Page

- Displays all tournaments and associated tournament data (location, tournament level, play surface), with the option for a user to select and explore an individual tournament. Selecting a tournament displays all matches played there, with individual views for each match that displays the match winner, score, match duration, etc.

Compare Page

- A user can select two arbitrary players across any year, level, or gender as part of a “head-to-head” matchup. Result will display information about each player’s individual career performance, historical betting odds, and the result of a simulated matchup.

Betting Overview and Strategy Page

- A user can select a player and receive betting history for that player and a potential future betting strategy.

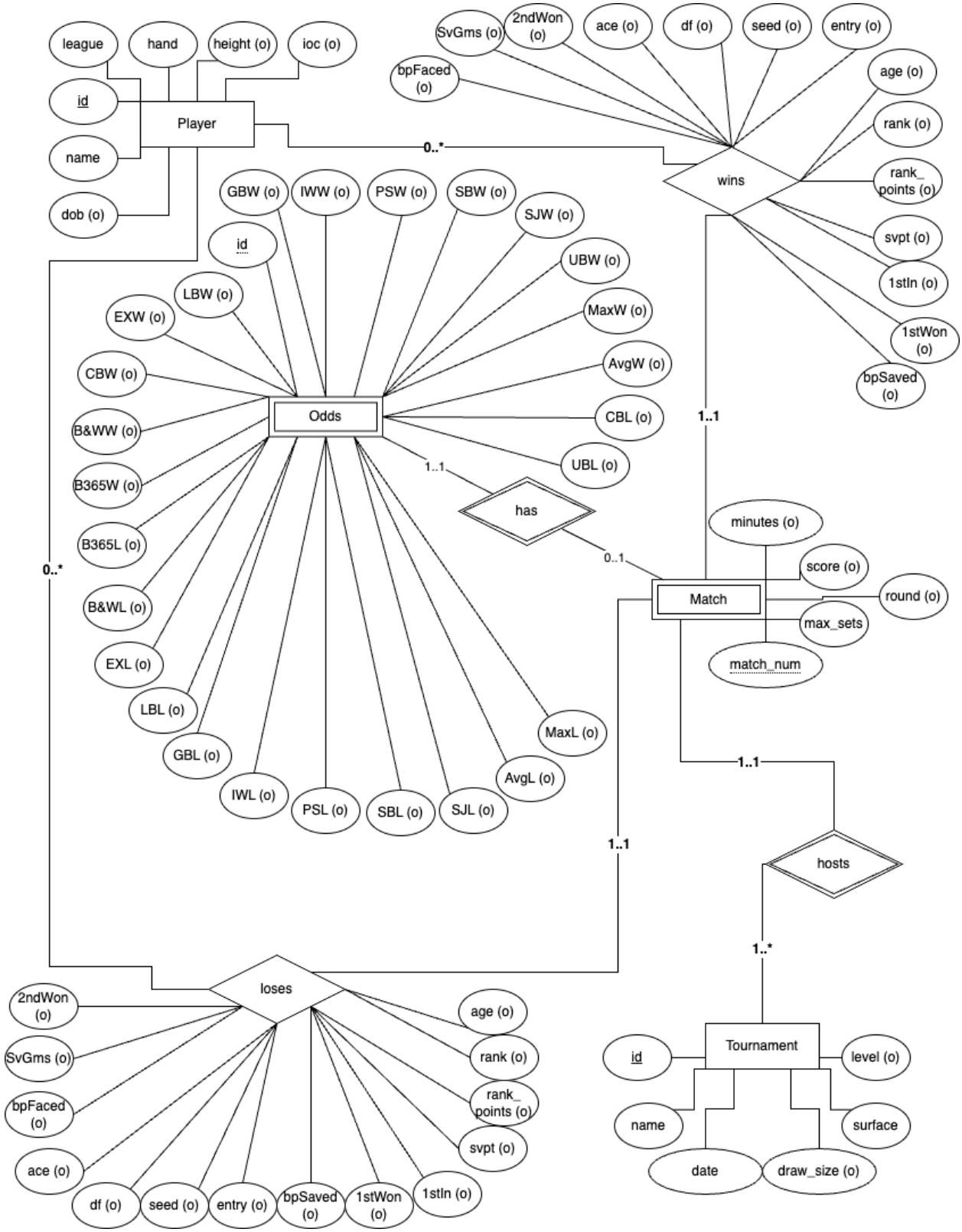
Must-Have Features

- Select a player to see player stats (within Player Page, and also accessible from any other page where a player is referenced)
- Select a tournament from a given list of tournaments to see tournament details.
- Select a match from a given tournament to see match details.
- Observe historical betting outcomes for a player.
- Compare any two players’ stats in a head-to-head

Nice-to-Have Features

- Integrate pictures of players into the web application by querying an API (Google Images?)
- Use ML to build a more sophisticated head-to-head matchup tool, allowing users to simulate an outcome.
- Player info only using data up to a certain year
    - Eg “Serena Williams in 2007”
- Implement a betting strategy that allows users to determine how much money they might make on certain bets. Using a ML model, we could potentially run the strategy against ‘the House’ to see if your bet paid off.

ER Diagram


league
hand
height (o)
ioc (o)
id
Player
name
dob (o)
SvGms (o)
2ndWon (o)
ace (o)
df (o)
seed (o)
entry (o)
bpFaced (o)
age (o)
rank (o)
rank_ points (o)
svpt (o)
1stIn (o)
1stWon (o)
bpSaved (o)
wins
0..*
1..1
GBW (o)
IWW (o)
PSW (o)
SBW (o)
SJW (o)
UBW (o)
MaxW (o)
AvgW (o)
CBL (o)
UBL (o)
id
LBW (o)
EXW (o)
CBW (o)
B&WW (o)
B365W (o)
B365L (o)
B&WL (o)
EXL (o)
LBL (o)
GBL (o)
IWL (o)
PSL (o)
SBL (o)
SJL (o)
AvgL (o)
MaxL (o)
Odds
1..1
has
0..1
0..*
minutes (o)
score (o)
round (o)
max_sets
match_num
Match
1..1
hosts
1..*
Tournament
id
level (o)
name
surface
date
draw_size (o)
loses
2ndWon (o)
SvGms (o)
bpFaced (o)
ace (o)
df (o)
seed (o)
entry (o)
bpSaved (o)
1stWon (o)
1stIn (o)
age (o)
rank (o)
rank_ points (o)
svpt (o)

SQL DDL

Player

(

id,
name NOT NULL,
league NOT NULL,
hand NOT NULL,
ioc,
height,
dob

)

PRIMARY KEY (id)

Tournament

(

id,
name NOT NULL,
date NOT NULL,
surface NOT NULL,
draw_size,
level

)

PRIMARY KEY (id)

Match

(

tourney_id,
match_num,
max_sets NOT NULL,
round,
score,
minutes,
w_id NOT NULL,
w_seed,
w_entry,
w_age,
w_rank,
w_rank_pts,
w_ace,
w_df,
w_svpt,
w_1stIn,
w_1stWon,

w_2ndWon,
w_SvGms,
w_bpSaved,
w_bpFaced,
l_id NOT NULL,
l_seed,
l_entry,
l_age,
l_rank,
l_rank_pts,
l_ace,
l_df,
l_svpt,
l_1stIn,
l_1stWon,
l_2ndWon,
l_SvGms,
l_bpSaved,
l_bpFaced
)
PRIMARY KEY (tourney_id, match_num)
FOREIGN KEY (tourney_id) REFERENCES Tournament(id)
FOREIGN KEY (w_id) REFERENCES Player(id)
FOREIGN KEY (l_id) REFERENCES Player(id)

Odds

(
id,
tourney_id,
match_num,
B365W,
B365L,
B&WW,
B&WL,
CBW,
CBL,
EXW,
EXL,
LBW,
LBL,
GBW,
GBL,
IWW,

```
        IWL,
        PSW,
        PSL,
        SBW,
        SBL,
        SJW,
        SJL,
        UBW,
        UBL,
        MaxW,
        MaxL,
        AvgW,
        AvgL,
)
        PRIMARY KEY ( id, tourney_id, match_num )
        FOREIGN KEY ( tourney_id ) REFERENCES Match( tourney_id )
        FOREIGN KEY ( match_num ) REFERENCES Match( match_num )
```

Data Cleaning & Preprocessing

The data will be cleaned and preprocessed using Python, along with several useful open sources packages such as numpy and pandas. The names of the players in the odds dataset are formatted as either {LAST_NAME FIRST_INITIAL.} or {LAST_NAME FIRST_INITIAL.MIDDLE_INITIAL.}. This differs from the players dataset which contains a column for both first and last name. Therefore, we will need to create temporary columns in the players datasets for the first initial, and parse out the last name and first initial from the odds dataset into their respective columns.

After cleaning up the player names, we will perform a string matching operation to ensure that each player found in the odds dataset has a match in the players dataset. In a perfect world, each player's last name and first initial will perfectly match, and a simple equality check would suffice. However it is likely that there are extra characters (such as hyphens, apostrophes, Jr., etc.) in one dataset and not in another. In this case, we will perform fuzzy matching based on string edit distance or a string similarity metric. Any unsuccessful matches after this operation will be manually edited or deleted from the dataset.

The tournament name in both the matches and odds datasets will be edited to be equivalent between the tables. The same will be performed with the round. With this information, we can join the matches and odds datasets together using the candidate key of {tournament, winner, loser, round} rather than the primary key of {tourney_id, match_num}. After this join is completed, we can add the match_num

column into the odds dataset and remove the other information so that the primary key of this table is also {tourney_id, match_num}.

Technologies

- React
- Node
- Express
- Python
- Pytorch
- Numpy
- Pandas
- MySQL

Team Member Responsibilities

- Akanksha Ashok
    - Implementing web app features, with an emphasis on full stack development.
- Noah Capp
    - Implementing web app features, with an emphasis on analysis of tennis odds, machine learning model and backend development.
- Bilal Ali
    - Implementing web app features across the stack, with an emphasis on backend development.
- Peter Akioyamen
    - Implementing web app features, with an emphasis on full stack development.
    - Database design and deployment.
    - Assisting with ML development.