Gautam Suresh Nambiar (gsn2012)
Sourabh Kumar Bhattacharjee (skb5275)

CS-GY 6083 – Final Project

Soccer Season Analysis System

Please find our application on this link: http://jedi.poly.edu:8660/

1. High-Level Application Description

We attempt to develop a simple and easy-to-use database system to manage and analyze data for a professional soccer league. This system can be used by sports analysts to analyze the performance of the Players, Teams, Coaches, Owners, etc., and how these change throughout the season. This system can be a useful insight-generation tool that can help these entities prepare better based on past data.
Our design is such that it enables any analyst to get answers to a variety of questions pertaining to (but not limited to) goals, wins/losses, bookings, opponent analysis, coach profile as well as referee allotments. This system will be focused on a single season of such a league but can be easily scalable to multiple seasons.

2. Entities, Relationship Sets, and Business Rules

Entities:

1. *Players* (id, name, age, nationality, foot, jersey_number, captain, appearances, substitutions, goals, penalties, yellow_cards, red_cards)
2. *Positions (*pos, pos_type*)*
3. *Teams* (id, name, establishement_year, city, titles, owner_id, owner_name, owner_age, owner_net_worth)
4. *Stadiums* (id, name, address)
5. *Managers* (id, name, age, nationality)
6. *Matches* (id, team1_id, team2_id, match_date, h_score, a_score, captain1_id, captain2_id)
7. *Referees* (id, name, nationality)
8. *Goals* (id, goal_time, winner, equalizer, own_goal, pen)
9. *Standings* (id, wins, draws, losses, points, pld, gf, ga). *This is a weak entity.*

Relationship Sets:

1. *Plays_in()* – Relationship set between *Players* and *Positions*.
2. *Plays_for()* – Relationship set between *Players* and *Teams*.
3. *Plays()* – Relationship set between *Teams*, and *Matches.* There's a recursive relationship between home and away Teams.
4. *Managed_by()* – Relationship set between *Teams* and *Managers*.

5. *Located_at()* – Relationship set between *Teams* and *Stadiums.*
6. *Held_at()* – Relationship set between *Matches* and *Stadiums*.
7. *Officiated_by()* – Relationship set between *Matches* and *Referees*.
8. *Score()* – Relationship set between *Players*, *Goals, and Matches.*
9. *Pertain_to* – Relationship set between weak entity *Standings* and *Teams*.

Business Rules:

1. Each Player plays in exactly one position, and each position has at least one player.
2. Each Player plays for exactly one Team, each Team has at least one player.
3. Each Team has exactly one Owner and each Owner owns exactly one Team.
4. Each Team is managed by exactly one Manager and each Manager manages at most one team.
5. Each Team has exactly one home Stadium and each stadium must be assigned to at most one Team.
6. Each Match is played by 2 teams, and all teams play some matches (Home/Away).
7. Every Match has some players playing in them.
8. Each Match must be played at exactly one Stadium.
9. Each Match must be officiated by at least one Referee.
10. Each goal must be scored by exactly one Player and must be associated with exactly one match.
11. Each Standing must be associated with exactly one Team, and each team must have some Standing. If a Team no longer exists, then we don't track it's standings.

3. Data Gathering

We have used the official *English Premier League* data for the season of 2021-22. We have data for 20 teams and 38 matches (for each team) played in the entire season. Data gathering was a bit challenging since all the data could not be found in one place/

The data has been sourced from the following sources:

1. Official Website of English Premier League Fantasy Football
2. Kaggle
3. Soccerstats Website

The tables that are formed are as follows:

1. Positions(pos, pos_type) - **16 tuples**
2. Referees(id, name, nationality) - **22 tuples**
3. Managers(id, name, age, nationality) - **20 tuples**
4. Stadiums(id, name, address) - **20 tuples**

5. Teams_Owner_Managed_Located(id,name,establishment_year, city, titles, owner_id, owner_name, owner_age, owner_net_worth, manager_id, stadium_id) - **20 tuples**
6. Standings_Pertain_to(id, pld, wins, draws, losses, gf, ga, points, t_id) - **20 tuples**
7. Players_Plays_In_Plays_for(id, name, age, nationality, jersey_number, foot, pos, captain, t_id, appearances, substitutions, goals, penalties, yellow_cards, red_cards) - **623 tuples**
8. Matches_Held_at(id, team1_id, team2_id, h_score, a_score, match_date, captain1_id, captain2_id, stadium_id) - **380 tuples**
9. Officiated_by(match_id, referee_id) - **380 tuples**
10. Teams_Play_Matches(match_id, team1_id, team2_id) - **380 tuples**
11. Goals_Scored(id, pen, goal_time, winner, equalizer, own_goal, player_id, match_id) - **1071 tuples**

4. Data Loading Procedure

Step 1: Data transfer to jedi.poly.edu server

a. Go to the root folder where we have the data folder in our local computer(this contains all the data in multiple csv files and also contains the schema.sql file)
b. Then in the terminal, type in the following command:

   "**scp -r ./data gsn2012@jedi.poly.edu:~/FinalProject/**"

c. Enter the password(this is the N number of the NYU ID).
d. This uploads the data folder inside the FinalProject folder of the jedi.poly.edu server.

Step 2: Create the tables

a. Once in the jedi server, type in the following command:

   "**psql -d gsn2012_db -a -f FinalProject/data/schema.sql**"

b. This creates the necessary tables.

Step 3: Populate the tables with data in the CSV files

a. Type in the following commands for each of the csv files present:

" **cat FinalProject/data/<file_name>.csv | psql -U gsn2012 -d gsn2012_db -c "COPY <file_name> from STDIN CSV HEADER"** "

b. Since there are 11 csv files, the above command has to be run 11 times for each different csv file.

5. Users Interacting with Database System

We have developed a StreamLit-based user interface that would enable the users/analysts to use a web client to access and query information from our database systems. StreamLit provides a lot of dropdown menu options which have been used by us to generate queries for our PostgreSQL Database using python. Python will then send this query to PostgreSQL Database via Psycopg2 and return the results accordingly.

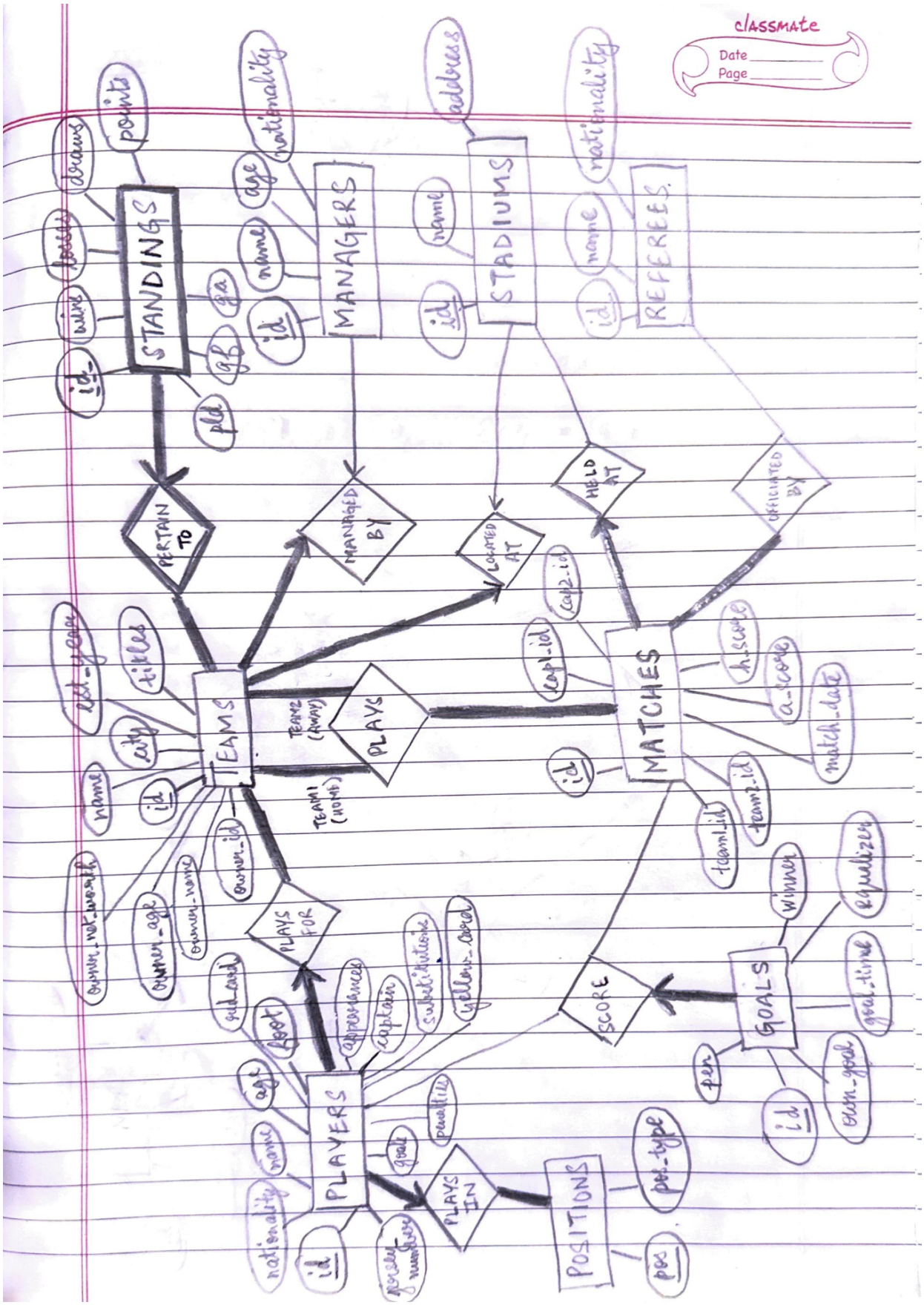
The analysts will be able to fetch answers to a variety of questions by selecting Entities and Relations for Projection and Join and by selecting values for Filtering and creating queries out of them without coding anything.

Some examples of the kinds of questions analysts will be able to fetch answers for are:

1. Find the Managers with the most home wins.
2. Who are the referees who have awarded the most penalties?
3. Find the highest goal scoring teams from London.

This is a non-exhaustive list of questions that our application can be used to answer, and these would require multiple joins and filtering that our code will be designed to handle.

ER Diagram:


STANDINGS
id
wins
losses
draws
points
gd
gf
ga
pld
PERTAIN TO
TEAMS
id
name
city
titles
est_year
owner_id
owner_name
owner_age
owner_nationality
TEAM1 (HOME)
TEAM2 (AWAY)
PLAYS
MANAGERS
id
name
age
nationality
MANAGED BY
STADIUMS
id
name
address
capacity
PLAYED AT
REFEREES
id
name
nationality
OFFICIATED BY
HELD AT
MATCHES
id
team1_id
team2_id
h_score
a_score
match_date
PLAYS FOR
PLAYERS
id
name
age
foot
nationality
jersey_number
goal
penalties
PLAYS IN
POSITIONS
pos
pos_type
SCORE
GOALS
id
per
own_goal
goal_time
team_id
winner
equalizer

Schema.sql

```
DROP TABLE IF EXISTS Managers CASCADE;
DROP TABLE IF EXISTS Referees CASCADE;
DROP TABLE IF EXISTS Positions CASCADE;
DROP TABLE IF EXISTS Stadiums CASCADE;
DROP TABLE IF EXISTS Teams_Owner_Managed_Located CASCADE;
DROP TABLE IF EXISTS Standings_Pertain_to CASCADE;
DROP TABLE IF EXISTS Players_Plays_In_Plays_for CASCADE;
DROP TABLE IF EXISTS Matches_Held_at CASCADE;
DROP TABLE IF EXISTS Officiated_by CASCADE;
DROP TABLE IF EXISTS Teams_Play_Matches CASCADE;
DROP TABLE IF EXISTS Goals_Scored CASCADE;


create table Managers (
    id integer primary key,
    name varchar(128),
    age integer,
    nationality varchar(128)
);

create table Referees (
    id integer primary key,
    name varchar(128),
    nationality varchar(128)
);

create table Positions (
    pos varchar(128) primary key,
    pos_type varchar(128)
);

create table Stadiums (
    id integer primary key,
    name varchar(128),
    address varchar(128)
);
```

```
create table Teams_Owner_Managed_Located (
    id integer primary key,
    name varchar(128),
    establishment_year integer,
    city varchar(128),
    titles integer,
    owner_id integer unique not null,
    owner_name varchar(128),
    owner_age integer,
    owner_net_worth decimal,
    manager_id integer unique not null,
    stadium_id integer unique not null,
    foreign key (manager_id) references Managers(id),
    foreign key (stadium_id) references Stadiums(id)
);

create table Standings_Pertain_to (
    id integer,
    pld integer,
    wins integer,
    draws integer,
    losses integer,
    gf integer,
    ga integer,
    points integer,
    T_id integer,
    primary key (T_id, id),
      foreign key (T_id) references Teams_Owner_Managed_Located(id) on
delete cascade
);

create table Players_Plays_In_Plays_for (
    id integer primary key,
    name varchar(128),
    age integer,
    nationality varchar(128),
    jersey_number integer,
    foot char(10),
    pos varchar(128) not null,
    captain boolean,
```

```
    T_id integer not null,
    appearances integer,
    substitutions integer,
    goals integer,
    penalties integer,
    yellow_cards integer,
    red_cards integer,
    foreign key (pos) references Positions(pos),
    foreign key (T_id) references Teams_Owner_Managed_Located(id)
);

create table Matches_Held_at (
    id integer primary key,
    team1_id integer,
    team2_id integer,
    h_score integer,
    a_score integer,
    match_date date,
    captain1_id integer not null,
    captain2_id integer not null,
    stadium_id integer not null,
    foreign key (stadium_id) references Stadiums(id),
    foreign key (captain1_id) references Players_Plays_In_Plays_for(id),
    foreign key (captain2_id) references Players_Plays_In_Plays_for(id)
);

create table Officiated_by (
    match_id integer,
    referee_id integer,
    primary key (match_id, referee_id),
    foreign key (match_id) references Matches_held_at(id),
    foreign key (referee_id) references Referees(id)
);

create table Teams_Play_Matches (
    match_id integer,
    team1_id integer,
    team2_id integer,
    primary key (match_id, team1_id, team2_id),
    foreign key (match_id) references Matches_Held_at(id),
```

```
    foreign key (team1_id) references Teams_Owner_Managed_Located(id),
    foreign key (team2_id) references Teams_Owner_Managed_Located(id)
);

create table Goals_Scored (
    id integer primary key,
    pen boolean,
    goal_time integer,
    winner boolean,
    equalizer boolean,
    own_goal boolean,
    player_id integer not null,
    match_id integer not null,
    foreign key (player_id) references Players_Plays_In_Plays_for(id),
    foreign key (match_id) references Matches_Held_at(id)
);
```