NBA Data Visualizations & User Platform

Wenbin Xuan wx2253 & Angel He ah3354

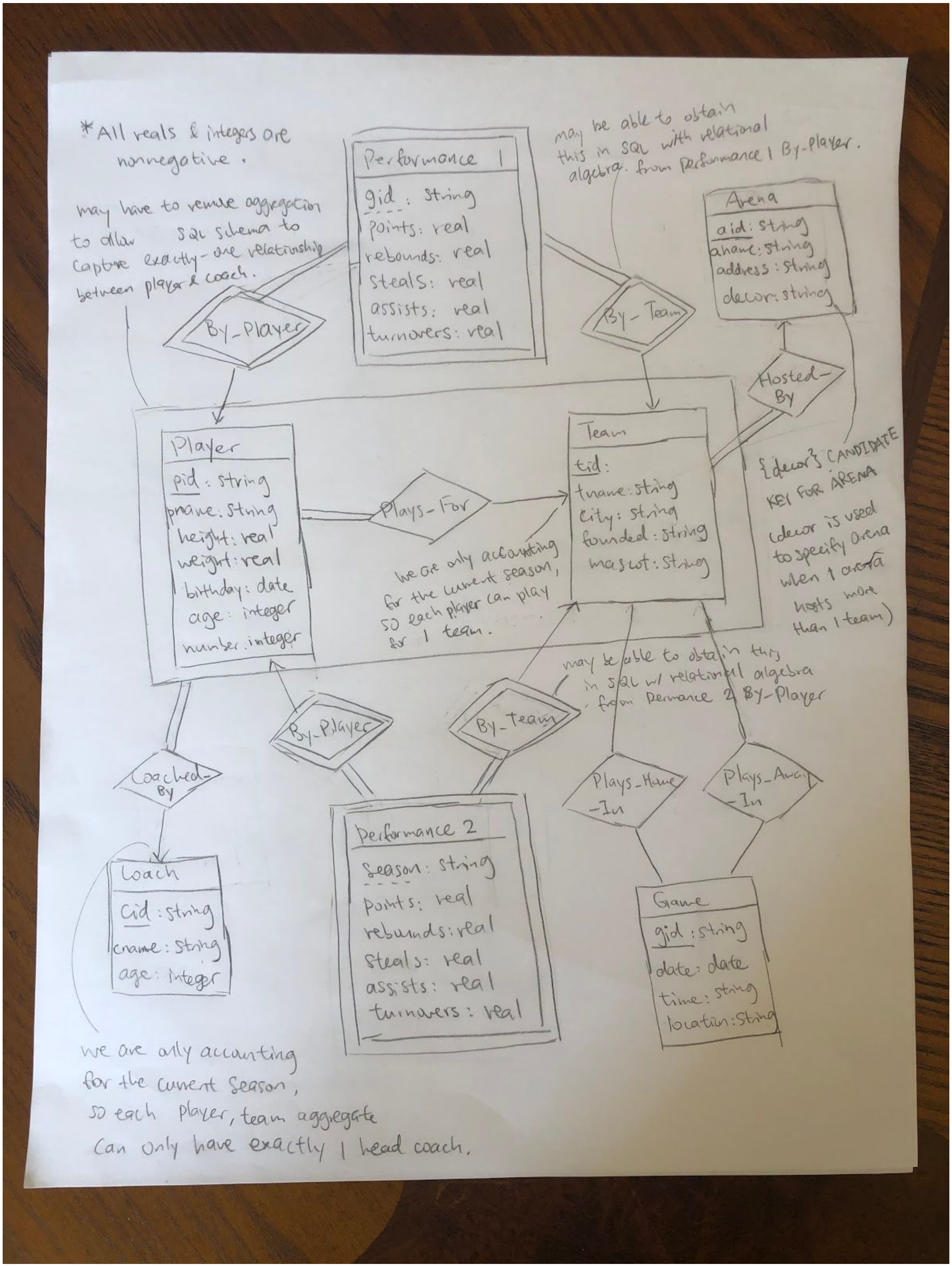
Description

We will design a database system for NBA statistics that would enable an application to turn these stats into data visualizations. We've noticed that NBA stats are usually presented in lists, but it would be helpful to have charts and graphs to accompany the data so some trends are easier to see. We would love to make a web application for this, but given our lack of experience with web design, we are leaning more towards the expanded design option. For part 3 we will expand our database design to allow for user interactions on the platform and incorporate links to relevant news and streaming sites so it will be a one-stop shop for NBA fans to track player/team performance and to easily watch and discuss games.

For our ER diagram we have strong entities for players, teams, games, coaches, and arenas. We also have weak entities for game and season performances because they are identified by the player or team they represent. Most of our relationships are binary, since there aren't many complex relationships between players and teams, teams and teams, and performance metrics and player/teams.

For part three, we will make our existing entities and relationships compatible with multiple seasons, so our users can look at data for the current season as well as past seasons. This will likely include adding relationship attributes, among other things. In addition, we will have user entities, so they can log in, favorite teams/players and follow their stats more easily. We also hope to incorporate links to videos and news articles alongside team/player/game/performance metrics so users can read more about a particular development or relive a particular moment. For news/video data we may have to do some web scraping from google searches for a given team/player/game/performance metric in order to compile a comprehensive list of article/video resources for each entry.

ER Diagram


*All reals & integers are nonnegative.
may have to remove aggregation to allow SQL schema to capture exactly-one relationship between player & coach.
Performance 1
gid: String
points: real
rebounds: real
steals: real
assists: real
turnovers: real
may be able to obtain this in SQL with relational algebra. from Performance 1 By-Player.
Arena
aid: string
aname: string
address: string
decor: string
By_Player
By - Team
Hosted-By
Player
pid: string
pname: String
height: real
weight: real
birthday: date
age: integer
number: integer
Plays_For
Team
tid:
tname: string
City: string
founded: string
mascot: String
we are only accounting for the current season, so each player can play for 1 team.
{decor} CANDIDATE KEY FOR ARENA
(decor is used to specify arena when 1 arena hosts more than 1 team)
may be able to obtain this in SQL w/ relational algebra from Permance 2 By-Player
Coached-By
By_Player
By_Team
Plays_Home-In
Plays_Away-In
Coach
Cid: string
cname: string
age: integer
Performance 2
Season: string
points: real
rebounds: real
steals: real
assists: real
turnovers: real
Game
gid: string
date: date
time: string
location: String
we are only accounting for the current season, so each player, team aggregate can only have exactly 1 head coach.

SQL

* All reals & integers should be nonnegative.

```
CREATE TABLE Player
        (pid CHAR(20),
         pname CHAR(20),
         height REAL,
         weight REAL,
         birthday DATE,
         Age INTEGER,
         number INTEGER,
         PRIMARY KEY (pid))

CREATE TABLE Team
        (tid CHAR(20),
         tname CHAR(20),
         city CHAR(20),
         founded CHAR(20),
         mascot CHAR(20),
         PRIMARY KEY (tid))


CREATE TABLE Arena
        (aid CHAR(20),
         aname CHAR(20),
         address CHAR(20),
         decor CHAR(20),
         PRIMARY KEY (aid),
         UNIQUE (decor))

CREATE TABLE Coach
        (cid CHAR(20),
         cname CHAR(20),
         age INTEGER,
         PRIMARY KEY (cid))
```

```
CREATE TABLE Game
        (gid CHAR(20),
         date DATE,
         time CHAR(20),
         location CHAR(20),
         PRIMARY KEY (gid))
```

```
CREATE TABLE Plays_For
        (pid CHAR(20),
         tid CHAR(20) NOT NULL,
         PRIMARY KEY (pid),
         FOREIGN KEY (pid) REFERENCES Player(pid),
         FOREIGN KEY (tid) REFERENCES Team(tid))
```

>We don't know yet how to capture the participation constraint of Player into Plays-For (In our ER-diagram, Plays_For has an 1 on 1 constraint, so we have to package Player and Plays_For into one table. However, we also need a single Plays_For table for aggregation. In this case, we don't know how to capture 1 on 1 constraint in the single Plays_For table yet).

```
CREATE TABLE Performance1-By_Player
        (gid CHAR(20),
         point REAL,
         rebounds REAL,
         steals REAL,
         asists REAL,
         turnovers REAL,
         pid CHAR(20),
         PRIMARY KEY (gid, pid),
         FOREIGN KEY (pid) REFERENCES Player (pid)
          ON DELETE CASCADE
          ON UPDATE CASCADE)
```

```
CREATE TABLE Performance1-By_Team
          (gid CHAR(20),
           point REAL,
           rebounds REAL,
           steals REAL,
           asists REAL,
           turnovers REAL,
           tid CHAR(20),
           PRIMARY KEY (gid, tid),
           FOREIGN KEY (tid) REFERENCES Team (tid)
            ON DELETE CASCADE
            ON UPDATE CASCADE)

CREATE TABLE Performance2-By_Player
          (season CHAR(20),
           point REAL,
           rebounds REAL,
           steals REAL,
           asists REAL,
           turnovers REAL,
           pid CHAR(20),
           PRIMARY KEY (season, pid),
           FOREIGN KEY (pid) REFERENCES Player (pid)
            ON DELETE CASCADE
            ON UPDATE CASCADE)


CREATE TABLE Performance2-By_Team
          (season CHAR(20),
           point REAL,
           rebounds REAL,
           steals REAL,
           asists REAL,
           turnovers REAL,
           tid CHAR(20),
           PRIMARY KEY (season, tid),
           FOREIGN KEY (tid) REFERENCES Team (tid)
            ON DELETE CASCADE
            ON UPDATE CASCADE)
```

```
CREATE TABLE Player-Plays_For
          (pid CHAR(20),
           pname CHAR(20),
           height REAL,
           weight REAL,
           birthday DATE,
           Age REAL,
           number REAL,
           tid CHAR(20) NOT NULL,
           PRIMARY KEY (pid),
           FOREIGN KEY (tid) REFERENCES Team (tid),
            ON DELETE NO ACTION
            ON UPDATE NO ACTION)
```

```
CREATE TABLE Team-Hosted_By
          (tid CHAR(20),
           tname CHAR(20),
           city CHAR(20),
           founded CHAR(20),
           mascot CHAR(20),
           aid CHAR(20) NOT NULL,
           PRIMARY KEY (tid),
           FOREIGN KEY (aid) REFERENCES Arena,
            ON DELETE NO ACTION
            ON UPDATE NO ACTION)
```

```
CREATE TABLE Plays_Home_In
          (gid CHAR(20),
           tid CHAR(20),
           PRIMARY KEY (gid, tid),
           FOREIGN KEY (gid) REFERENCES Game,
           FOREIGN KEY (tid) REFERENCES Team)
```

```
CREATE TABLE Plays_Away_In
        (gid CHAR(20),
         tid CHAR(20),
         PRIMARY KEY (gid, tid),
         FOREIGN KEY (gid) REFERENCES Game,
         FOREIGN KEY (tid) REFERENCES Team)
```

```
CREATE TABLE Plays_For-Coached_By
        (cid CHAR(20) NOT NULL,
         pid CHAR(20),
         tid CHAR(20) NOT NULL,
         PRIMARY KEY (pid, tid),
         FOREIGN KEY (cid) REFERENCES Coach (cid)
```

>Trouble on expressing aggregation when Plays_For and Coached_By are in the same table. Can't capture Foreign key references to Plays_For.

>While we need to combine Plays_For and Coached_By to construct an 1-1 constraint on Coached_By.

>One fix would be to remove aggregation of Player & Team and have Coach_By connect with just Player.

```
CREATE TABLE Coached_By
        (cid CHAR(20) NOT NULL,
         pid CHAR(20),
         tid CHAR(20) NOT NULL,
         PRIMARY KEY (pid, tid),
         FOREIGN KEY (cid) REFERENCES Coach (cid)
         FOREIGN KEY (pid, tid) REFERENCES Plays_For (pid, tid))
```

>Assuming Coached_By has an 1-1 constraint here….Since the previous case of aggregation doesn't really work.