Explain your assumptions for each entity and relationship in your model. Discuss why you've modeled something as an entity rather than an attribute of another entity. Describe the cardinality of relationships, like why a student is linked to only one advisor. These assumptions might come from customer requirements or application constraints. Please clarify them.

- **Game:** Most essential entity that contains information about each NBA game. This is modeled as an entity because each NBA game contains several important attributes that provide information about a given game. In terms of cardinality we know that each game can have 0 to many calls hence the first relation. Furthermore, each game has exactly 1 home and away team. Lastly, each game can be searched 0 to many times, and this describes all the relationships from the Game.
    - **game_id (primary key):** Specifies each games and is referenced by the other tables
    - **date:** Keeps track of the date of each match
    - **home_team (foreign key → Team):** Keeps track of the home team of the game
    - **away_team (foreign key → Team):** Keeps track of the away team of the game
    - **attendance:** Size of the crowd at each game
    - **box_score:** Link to end results/outcome of the game
    - **season:** Year game took place
    - **playoff:** Boolean value of whether the game was a playoff game or not

---

- **User**: This entity that represents user data who tries to access our database. This is modeled as an entity since user information is stored as the attributes, thus creating a hierarchy of organization. The cardinality for users is relatively simplistic: each user can make 0 to many searches. The primary purpose of users is to be able to sort and see visualizations of NBA foul data so it only has to map to searching.
    - **user_id (primary key):** Unique identifier for each user
    - **first_name:** User's first name
    - **last_name:** User's last name
    - **email:** User's email address linked with account
    - **password:** User account's password

- **Call**: This entity represents a foul call made in a given NBA game. This is modeled as an entity as every call in a game is unique, occurs at different times, and has various comments/attributes associated with it. Alongside that, a game can have multiple different calls (either correct or incorrect) which have unique data points. The cardinality for calls is implied: a unique, specific call can only be attributed to one game but a game could have multiple calls. Furthermore, exactly one team receives a call, one player receives a foul call, and each referee either made the call or did not make the call. The calls entity is the driving source of our visualizations and is integral to our core functionality.
    - **call_id:** Unique identifier for each call
    - **game_id (foreign key → Game):** References the game this call is attributed to

- **call_type:** Type of referee call (correct call, correct no call, incorrect call, incorrect no call)
- **committing (foreign key → Player):** References the player who the call was attributed to
- **disadvantaged (foreign key → Player):** References the player who was disadvantaged by the call
- **decision:** the outcome of the call
- **comments:** extra notes/details
- **home_score:** the home score at time of call
- **away_score:** the away score at time of call
- **time_left:** the time left in the game at time of call
- **period:** the time period of call (Q4, Q5, etc)
- **video_link:** contains a clip of the moment of the game when the call was made if available
- **ref1 (foreign key → Referee):** 1st referee id
- **ref2 (foreign key → Referee):** 2nd referee id
- **ref3 (foreign key → Referee):** 3rd referee id

- **Search:** This entity represents a search made by a user in the NBA database. Each search is unique and is associated with various attributes such as the game, player, and user involved. The cardinality for searches is implied: a unique, specific search can only be attributed to one user but a user could make multiple searches. Furthermore, a search can be associated with either one game or not associated with any game, and similarly, it can

be associated with one player or not associated with any player. The Search entity is crucial for understanding user behavior and preferences.

  - **search_id (primary key):** Unique identifier for a given search
  - **query:** Full user query
  - **game_id (foreign key → Game):** References the game a user searched for
  - **player_id (foreign key → Player):** References the player a user searched for
  - **user_id (foreign key → User):** User who made the query
  - **search_date:** Day the query was made

- **Referee:** This entity represents a referee in an NBA game. Each referee is unique and has various attributes such as their name and the number of calls they've made. In terms of cardinality, each referee can make zero to many calls in a given NBA game, but each call is made by exactly one referee. This allows us to track the performance and decision-making of each referee. The Referee entity is crucial for understanding the decision-making process in the game.
  - **ref_id (primary key):** Unique identifier for a referee
  - **first_name:** referee's first name
  - **last_name:** referee's last name
  - **call_count:** number of foul calls made
  - **i_call_count:** count of calls distinguished by referee

---

- **Player:** This entity represents a player in an NBA game. Each player is unique and has various attributes such as their name and the team they play for. Each player plays for at

most one team but a team can have many players. Also, each player can commit zero to many fouls but only one player can commit a foul at a time. There are zero to many player_ids based on how many players are in the game, but at most one id per player. Also, each player plays for at most one team but a team can have many players. This allows us to track the performance and behavior of each player in the context of their team.. The Player entity is crucial for understanding the dynamics of the game.

    - **player_id (primary key):** unique identifier for a player in a game
    - **first_name:** player's first name
    - **last_name:** player's last name
    - **team_id (foreign key → Team):** unique identifier for team the player plays on

- **Team:** This entity represents a team in the NBA. Each team is unique and has various attributes such as their name and the city they are based in. In terms of cardinality, each team can have many user_ids and team game_ids since they play many games but exactly one game id per team. Also, a team can play many home and away games in a season and can have many disadvantaged calls in a given game. This allows us to track the performance and behavior of each team in the context of their games and calls. The Team entity is crucial for understanding the dynamics of the game and the performance of the teams.

    - **team_id (primary key):** unique identifier for a team
    - **name:** team name
    - **city:** location team is based in

Normalize your database. Apply BCNF or 3NF to your schema or show that your schema adheres to one of these normal forms. Describe why you choose to use BCNF vs 3NF.

We opted to use 3NF (Third Normal Form) to effectively normalize our database. Our choice of 3NF stems from its ability to preserve all functional dependencies, ensuring the coherence of our logical design. Additionally, 3NF offers a relatively straightforward normalization process without the need to sacrifice essential dependencies. In our case, this is important because we need to ensure that attributes for each entity are preset so that we can display all the corresponding data.

The first step in applying 3NF was to ensure that our data was in 2NF. This was a quick check, as we ensured that each attribute in an entity was not partially dependent on some key. 3NF normalization was then applied to check for transitive dependencies, where attributes depended on other attributes that were not keys. In our case, this did not occur, so our database was 3NF normalized.

Convert your conceptual database design (ER/UML) to the logical design (relational schema). Note that a relational schema is NOT an SQL DDL command.

Table Name: Game(

Column 1: game_id:INT [PK]

Column 2: date:INT

Column 3: home_team:INT [FK to Team.team_id]

Column 4: away_team:INT [FK to to Team.team_id]

Column 5: attendance:INT

Column 6: box_score:VARCHAR(255)

Column 7: season:INT

Column 8: playoff:BIT

)

Table Name: Team(

Column 1: team_id:INT [PK]

Column 2: name:VARCHAR(50)

Column 3: city:VARCHAR(255)

)

Table Name: User(

Column 1: user_id:INT [PK]

Column 2: first_name:VARCHAR(50)

Column 3: last_name:VARCHAR(50)

Column 4: email:VARCHAR(50)

Column 5: password:VARCHAR(50)

)

Table Name: Search(

Column 1: search_id:INT [PK]

Column 2: query:VARCHAR(225)

Column 3: game_id:INT [FK to Game.game_id]

Column 4: player_id:INT [FK to Player.player_id]

Column 5: user_id:INT [FK to User.user_id]

Column 6: search_date:DATE

)

Table Name: Player(

Column 1: player_id:INT [PK]

Column 2: first_name:VARCHAR(50)

Column 3: last_name:VARCHAR(50)

Column 3: team_id:INT [FK to Team.team_id]

)

Table Name: Referee(

Column 1: red_id:INT [PK]

Column 2: first_name:VARCHAR(50)

Column 3: last_name:VARCHAR(50)

Column 4: call_count:INT

Column 5: i_call_count:INT

)

Table Name: Call(

Column 1: call_id:INT [PK]

Column 1: game_id:INT [FK to Game.game_id]

Column 3: call_type:VARCHAR(225)

Column 4: committing:INT [FK to Player.player_id]

Column 5: disadvantaged:INT [FK to Player.player_id]

Column 6: decision:VARCHAR(225)

Column 7: comments:VARCHAR(225)

Column 8: home_score:INT

Column 9: away_score:INT

Column 10: time_left:INT

Column 11: period:VARCHAR(225)

Column 12: video_link:VARCHAR(225)

Column 13: ref1:INT [FK to Referee.ref_id]

Column 14: ref2:INT [FK to Referee.ref_id]

Column 15: ref3:INT [FK to Referee.ref_id]

)