JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

National Basketball Association (NBA) Financial Market Database

Course Section: CS605.641.81
Summer, 2025

Prepared by

Nicolas Escudero
08/02/2025

Table of Contents

1. Introduction

Basketball has and always will be a huge part of my life – I have been playing basketball ever since I was in the Little Kids' League when I was five years old. Although I never had any desire to go to the NBA myself, I invested so much time growing up watching the sport. Once I was older, I realized more that NBA as a league was more than just playing the game; there was an entire business aspect to the League that, although often overlooked, is the centerpiece of why the NBA is as big as it is today. Although fanbases associate the NBA with huge role models or super-teams, such as Michael Jordan and the 90's Chicago Bulls or Kobe Bryant of the 2000's Lakers, much of the popularity was made possible due to proper marketing of these Players and Teams at the right time. This Database reflects how we can view the NBA as less of a Basketball League and more of a Business. It will model seasonal financial data of many facets of the NBA, such as which Players, Teams, or Games are generating the most revenue. It will also track certain data that can be used to monitor how favorable it could be to market a certain Player or Team.

1.1. Scope and Purpose of Document

This document includes detailed discussions on the requirements and overall design of the NBA Financial Database. It will include the conceptual and logical design of the Database, as well as implementation in SQL language. The document also discusses system requirements and security aspects of the Database, including topics such as Hardware and Software requirements, Database backup and recovery methods, CASE Tool usage and a CRUD Matrix. The Database will focus on measuring the Financial and Marketing aspects of the NBA, with the end goal of allowing for better visualization of which areas are contributing more to the NBA's financial growth.

1.2. Project Objective

The Database project's objective is to measures two primary aspects of the market strategy: Accumulated Revenue, and Fan Popularity. For Accumulated Revenue, this aspect is applied across five different facets of the NBA: Players, Teams, Games, Stadiums, and Media Networks. For Fan Popularity, additional aspects are also measured, such as Player Statistics, Team Owner and Head Coach Popularity, and Team Seeding for the season.

Given this information, the next objective is to be able to pull any necessary financial information of the NBA for further marketing strategy purposes. We build a system that essentially archives this financial information for every season of the NBA, and allows retrieval of chunks of information to find trends in several factors all throughout the NBA. The data is also useful for comparative analysis between different factors, such as financial aspects between different seasons or from a Player in two different Teams.

2. System Requirements

We discuss the system requirements for the NBA Financial Market Database. This section covers topics such as hardware requirements, software requirements, functional requirements, and database requirements.

2.1 Hardware Requirements

The minimum hardware requirements are listed below:

- Processor: At least 1 GigaHertz x64
- RAM: At least 8 GigaBytes (recommended 32GB)
- Hard Disk Space: At least 500 GigaBytes
- Graphics Card: DirectX 9 or later
- Monitor: 800x600 or higher

2.2 Software Requirements

The minimum software requirements are listed below:

- Operating System: Windows 10 (20H2) or higher. Linux supported distributions (Red Hat Enterprise Linux, SUSE Enterprise Linux Server, Ubuntu)
- .NET Framework: Version 4.7.2 or higher

2.3 Functional Requirements

The Database supports the following functional requirements:

- A new Season starts around mid-October, where:
    - The Regular Season takes place from October-April
    - The NBA Playoffs take place from April-June
    - The Off-Season (period of break between Seasons) takes place from June-October.

  An exception to this can be made for a few seasons, such as during the COVID Pandemic.

- A Game can occur on any day of the week, and there can as many games as possible scheduled on the same day:
    - Some portions of the year may be blocked off from scheduling games. These include special yearly events such as:
        - All-Star Break (typically mid-February)
        - In-Season Tournament (typically end of November)
        - Thanksgiving Holiday
        - Election Days
    - Some portions of the year may be required to schedule a specific amount of games. These include:
        - Christmas Day Games: 5 games

        - Last day of Regular Season Games: All Teams play in a Game
- A Game must always have a Home Team and an Away Team.
    - The Home Team always plays at their designated Stadium. The number of times a Team plays at their Home Stadium is approximately half of the total games played in a Season.
    - The Away Team visits any Stadium that is not their own designated Stadium.
- For any given Season, a Team must play the following Games:
    - A Team that is NOT residing in the same Conference: 2x
    - A Team that is residing in the same Conference: 3-4x

- Players and Staff can be added to or dropped from any team throughout a season to signify a trade or restructure of the team and management.
    - Players can be traded at any point throughout a specified Trade Period. The Trade Period typically starts at the end of an NBA Season, through to the start of the subsequent NBA Season, and ends at the Trade Deadline in early-February.
    - Players and Staff can be dropped from a team, as well as picked up by a team with no associated trade, at any point throughout the Season and Off-Season.
- Players can have their game performances statistically tracked in any game in any season. These statistics are archived for future assessment, analysis, and marketing purposes.

- Player and Staff Popularity Scores are measured based on an external scoring system (not specified in this Database) of the critical acclaim and fan engagement directed towards that entity. Factors that are included in this external scoring system:
    - Fan Engagement
    - Merchandise Sales
    - Survey Feedback
    - Player and Staff Accolades (Championships won, MVP Awards, etc.)
    - Overall Likeability
- Popularity Scores are calculated and updated into the Database at the end of every month. A final score for the season is calculated at the end of the NBA Playoffs.

- Revenue that is tracked for a particular Season starts on the day of the first Game of the season and ends on the day of last Playoff Game of the Playoffs.
- Revenue amounts can be added to or deleted from at any point throughout the year for any Season, in the event of recounts or database audits.
    - New Revenue totals are calculated and added to the Database at the end of every month.

- A Stadium can accommodate any Game on any day of the week. A Stadium is also assumed to be able to host the game, with exceptions to factors such as Weather or unforeseen circumstances.

- A Media Network is any instance of a Network brand that shows a live broadcasting of a Game. Networks can range from those viewing on Cable TV, or those viewing from an associated streaming platform:
    - Examples of valid Networks: ESPN, NBC, NBA.TV, Regional Networks
    - Examples of valid Cable Providers: Spectrum, DirectTV, Verizon
    - Examples of valid Streaming Platforms: Amazon Prime, Hulu, YouTubeTV

2.4 Database Requirements

MySQL Version 8.0 was used for the Database implementation.

3. Database Design Description

This section describes the overall Database design, including aspects such as design rationale, the ER model, the Relational model, Database security and Database recovery.

3.1 Design Rationale

The design of the NBA Revenue Analytics Database was organized to reflect five principal areas of revenue streams: Players, Team Engagement, Game Engagement, Stadium Performance, and Media. Each revenue stream's performance is tracked throughout each Season of the NBA, with the Season typically running from October to June. The NBA Season proved to be a consistent timeline for tracking revenue data per year – therefore, the entity SEASON was created to contain and split all the rest of the Database's data into an individual season, allowing for consistency and clarity of the data.

From this, each revenue stream is then organized into their own respective "zones" in the database, and each stream at the very least contains one defining strong entity to track: PLAYER, TEAM, GAME, STADIUM, and MEDIA.

1. **PLAYER**

    Much of the NBA today relies heavily on player-based performances and player popularity. For example, if a player is highly acclaimed by fans and has many awards/accolades to their name, then that player will have higher merchandise, higher jersey sales, and more brand sponsorships.

    In the Database, three main entities define the overall performance and revenue of a Player in the NBA. The strong entity, PLAYER, contains the defining attributes of an NBA Player, such as the Player's name and current team. It also contains a popularity ranking for the Player, which is also necessary to setup a Player for better Marketing opportunities and fan engagement. Two weak entities also define how the Player contributes to the NBA. PLAYER_STAT tracks the Player's in-game statistics throughout a Season, which is necessary to understand how well a

Player is performing. PLAYER_REVENUE describes the monetary gain of the Player throughout the Season, which will be further discussed later.

2. TEAM

This area encompasses how an overall Team (sometimes referred to as a "Franchise") is performing within the organization. If a Team has a history of winning championships or acquiring star players that lead their team in impressive ways, then that Team has better marketability in terms of increased merchandise sales and fans coming to watch their games. Additionally, the Team's staffing also plays a significant role, as certain members of the staff pioneer the decision-making of the Team and determine how the Team functions as a unit, what players move in or out of the Team, and more. Lastly, Team performance can also be gauged on where the Team finished in the seasonal standings, or what seed the team finished in.

The main entities that reflect the overall Team performance are its strong entity TEAM and TEAM_REVENUE. Additional Staffing entities are also defined here, each with their own attributes of names and popularity scores: HEAD_COACH, OWNER. The Head Coach was chosen to track specifically due to their large role in guiding a Team throughout the season and their significant presence in Media as well. Furthermore, the Owner was chosen to track due to their large presence in the financial aspect of a Team, as they primarily fund much of the Team's successes.

3. GAME

This area encompasses the performance and revenue of each Game within an NBA Season. There are a daunting number of games throughout a single NBA Season, with only a handful of these games standing out to fans due to the popularity of the Teams, All-Star Player presence, brewing and ongoing rivalries, and more. These entities would help gain a better understanding of which of these games are more entertaining for fans to watch, and it would allow the NBA to prioritize marketing more highly anticipated games.

The main entities that fall under the Games category are the strong entity GAME and weak entity GAME_REVENUE. The GAME entity provides many significant details of a Game instance, such as which teams were playing in the game, who won the game, and how fans received the game.

4. STADIUM

This area encompasses the total performance and revenue gained within a Stadium throughout an NBA season. Stadium performance is crucial for the success of a Team's Marketability; if fans are treated to a very nicely organized stadium with plentiful amenities, then chances are that Stadium will be more

successful in the long run. Furthermore, since each Stadium offers its own promotions, food, merchandise, and space, this can help determine what fans love to engage with in the Stadium, and which Stadiums offer the best fan experience overall. Stadium location is also important, as fans must be more inclined to travel to watch a Game where it is not too inconvenient (see Oakland Raider's/Oakland A's).

The main entities that fall under the Stadium category are strong entity STADIUM and weak entity STADIUM_REVENUE. Included in describing each Stadium is it's associated team, location, maximum occupancy, and revenue aspects of in-Stadium sales.

5. **MEDIA**

The latest addition to this Database comes in the form of the Media and Television side of the NBA, which plays a substantial role in NBA Marketing and viewership for fans who, more normally, prefer to watch the games at home. Factors such as sponsorships, commercials, subscription services, and fan viewership are all topics of interest for this area, as all of these contribute heavily to how the NBA can approach certain marketing strategies while allowing for a seamless demonstration of the sport that everyone loves.

The main entities of this area is strong entity MEDIA and weak entity MEDIA_REVENUE. The MEDIA entity defines characteristics of a Sports Network, which are dedicated Network brands for the showing of NBA and other associated sports television. As these broadcasts are also shown across a multitude of providers and streaming services, these are also specified, which allows the NBA to analyze what method fans use to tune into the game.

Before going into further discussion about the Revenue entities, it is worth noting an important detail of why the Revenue entities are places where they are in the ERD. Prior to adding the Revenue entities, each strong entity was directly linked to the SEASON entity to reference the Season year. Doing so, however, resulted in the creation of several M:N relationships – for instance, a PLAYER instance can play in multiple SEASON instances, while a single SEASON instance can hold several PLAYER instances at once. These M:N relationships may be troublesome during implementation as it poses risk for duplicate entries and constraint violations. To fix this, a junction/intersectional entity must be introduced and inserted in between the M:N relationship to form less troubling M:1, and 1:N relationships with the junction entity.

This is the "hidden" secondary role of the Revenue entities: to mitigate the presence of M:N relationships to the SEASON entity while also tracking revenue data for its associated strong entity. The intersectional entity contains the revenue progress of the strong entity ("ENTITY"_REVENUE) across several attributes, specific to each strong entity. The intersectional revenue entity connects the ID of the specific instance of a strong entity to the specific season that is being tracked; because of this, it inherits the

primary keys of the strong entity and the SEASON entity. This proved to be a more efficient structure, due to its ability to be consistent across multiple areas of revenue streams, keep all revenue data into a single entity, and resolve the M:N relationships that would have been introduced if a relation was made directly from the strong entity to the SEASON entity.

Additional entities were added to encompass areas of revenue analytics that were deemed important for the overall production and marketability of the zone entities. The CONFERENCE and PLAYOFF_SEED entities help in defining characteristics of the TEAM entities by outlining where a Team placed in the Regular season, which is a helpful indication of how well a TEAM did that season.

3.2 ER Model

This section now discusses the general description of the ER Model and design. It will encompass entities, relationships, and the ERD associated with the Database.

3.2.1 Entities

SEASON

The SEASON can otherwise be referred to as the "backbone" of how the database is structured, as this entity helps track revenue profits of each of the areas of revenue streams per season. It defines the timetable of how much revenue should be tracked - in this case, performance of the NBA as a whole is graded per season.

The main attributes of SEASON include:

- Season_yr: Primary key, unique identifier of which season is currently being tracked.
- Champ_team (O): the champion of that particular season (optional if no champion yet)
- Gross_rev: the total revenue of that particular season.

PLAYER

The PLAYER entity defines the attributes of one player in the NBA, and includes the main characteristics that describe the player.

The main attributes of PLAYER include:

- Player_id: Primary Key, a unique identification number assigned to each player

- P_fname: First name of the player
- P_lname: Last name of the player
- P_suffix (O): Suffix, if needed. Ex: Michael Porter "Jr.", Jimmy Butler "III".
- P_team: Foreign Key, refers to Team_name of TEAM entity. Current team that the player plays for in the NBA.
- P_popscore: A measurement of how popular a player is in the NBA. Measured as a percentage rating, this is gauged based on fan interaction, merchandise sales of that particular player, and overall likeability of the player.

PLAYER_STAT

The PLAYER_STAT weak entity measures the statistical aspects of a player's performance throughout one season. This entity is needed to gauge marketability strategies of certain players - if a player is performing much better statistically, then the easier it would be to market the player and engage fans to that player.

The main attributes of PLAYER_STAT include:

- Player_id: Foreign Key, refers to the Player_id attribute of PLAYER entity.
- Season_yr: Foreign Key, refers to the Season_yr attribute of SEASON entity.
- Points: Total points scored, regardless of method of basket.
- Assists: Total assists, rewarded if a teammate scores off the main player's pass.
- Rebounds: Total rebounds, rewarded if a player possesses the ball after a missed shot attempt.
- Turnovers: Total turnovers, rewarded if a player inadvertently loses the ball, resulting in the other team gaining possession of the ball.
- Field_goal_pct: Field goal percentage. A measurement of how well a player is completing shots that are taken within the three-point line on the court.
- Three_pt_pct: Three-point percentage. A measurement of how well a player is completing shots that are taken outside of the three-point line on the court.
- Plus_minus: Plus/minus (+/-), a common unit of measurement in basketball, which gauges the positive or negative impact that a player has when they are currently playing on the court. This is measured by points scored by the player's team MINUS points scored by the opposing team, only during the duration of the player's time on the court.
- Minutes_played: Total minutes played throughout the season.

- Games_played: Total games played throughout the season.

PLAYER_REVENUE

The PLAYER_REVENUE weak entity measures the revenue aspect of the PLAYER area of interest. It encompasses all the aspects of a player that can generate revenue for the NBA.

The main attributes of PLAYER_REVENUE include:

- Player_id: Foreign Key, refers to the Player_id attribute of PLAYER entity.
- Season_yr: Foreign Key, refers to the Season_yr attribute of SEASON entity.
- P_jersey_rev: Total revenue of jersey sales associated to a specific player.
- P_apparel_rev: Total revenue of related apparel sales associated to a specific player, excluding jersey sales.
- P_misc_rev: Total revenue of miscellaneous product sales associated to a specific player. This encompasses non-apparel products, such as souvenirs or figurines.
- P_sponsor_rev: Total revenue of sponsorships associated to a specific player. Many players have brand deals and other sponsorships tied to them. Ex: Michael Jordan and Nike, Shaquille O'Neal and IcyHot.

TEAM

The TEAM entity defines the attributes of a single Team that partakes in the NBA, and includes the main characteristics that describe the Team.

The main attributes of TEAM include:

- Team_name: Primary Key, the unique name given to each Team in the NBA.
- Team_conference: Foreign Key, refers to Conference_name of CONFERENCE entity. The current conference that the team plays in.
- Team_stadium: Foreign Key, refers to the Stadium_id of STADIUM entity. The Team's home court, or the main Stadium that is located in the Team's city.
- Team_city: Current city location of the team.
- Team_state: Current state location of the team.
- Team_roster_size: Current roster size of the team, or how many players are playing in the team.

- Team_popscore: A measurement of how popular a team is in the NBA. Measured in a percentage rating, his is gauged based on fan interaction, merchandise sales of that particular team, and overall likeability of the team.

TEAM_REVENUE

The TEAM_REVENUE weak entity measures the revenue aspect of the TEAM area of interest. It encompasses all the aspects of a Team that can generate revenue for the NBA.

The main attributes of TEAM_REVENUE include:

- Team_name: Foreign Key, refers to the Team_name attribute of TEAM entity.
- Season_yr: Foreign Key, refers to the Season_yr attribute of SEASON entity.
- T_ticket_rev: Total revenue of ticket sales associated to a specific team.
- T_apparel_rev: Total revenue of related apparel sales associated to a specific team. Including jersey sales.
- T_misc_rev: Total revenue of miscellaneous product sales associated to a specific team. This encompasses non-apparel products, such as souvenirs.

CONFERENCE

The CONFERENCE entity defines the attributes of a conference, or a group of teams that compete against each other in the standings for a spot in the playoffs. These teams are generally located closer to each other, and play each other more frequently within the season.

The main attributes of CONFERENCE include:

- Conference_name: Primary Key, a unique name of the Conference. In a typical NBA season, this would simply be either "Eastern" or "Western" Conferences.
- Conference_size: The number of teams that make up the conference. In a typical NBA setting, this would be a size of 15 teams in one conference.

PLAYOFF_SEED

The PLAYOFF_SEED weak entity illustrates the standings in which teams compete against each other in, and are associated to one Conference. For instance, the top eight teams of a 15-team Conference move on to the Playoffs, and the names of each team are taken from 1 to 8, based on win/loss record. This helps gauge how well a team is doing in a season or across multiple seasons, which helps determine marketability.

The main attributes of PLAYOFF_SEED include:

- Conference_name: Foreign Key, refers to the Conference_name attribute of CONFERENCE entity.
- Season_yr: Foreign Key, refers to the Season_yr attribute of SEASON entity.
- Seed_one … Seed_eight: The full name of each team that places in a certain position in the standings at the end of the regular season.

HEAD_COACH

The HEAD_COACH entity defines all attributes of one Head Coach in the NBA. The Head Coach is an important figure of a Team, as they are responsible for making many of the coaching decisions of the team, both within and outside of a game.

The main attributes of HEAD_COACH include:

- Coach_id: A unique identification number for a Coach.
- Team_name (O): Foreign Key, references the Team_name attribute of TEAM. Describes current team that the Coach coaches or may not be coaching any team at the moment.
- Coach_fname: First name of the Coach.
- Coach_lname: Last name of the Coach.
- Coach_popscore: The measure of popularity and likeability of the Coach.

OWNER

The OWNER entity defines all attributes of an Owner of a Team/Franchise in the NBA. The Owner is known to be the financial support and representative of the Team that they own, hence being a primary figure in the measurement of NBA profitability.

The main attributes of OWNER include:

- Owner_id: A unique identification number of an Owner.
- Owner_fname: First name of the Owner.
- Owner_lname: Last name of the Owner.
- Owner_popscore: The measure of popularity and likeability of the Owner.

GAME

The GAME entity defines the attributes of a single Game that happens in the NBA and includes the main characteristics that describe a single Game instance.

The main attributes of GAME include:

- Game_id: Primary Key, a unique identification number assigned to each Game.
- Date: The date of which the game instance took place on.
- Stadium_location: Foreign Key, refers to Stadium_location attribute of STADIUM entity.
- Home_team: Foreign Key, refers to Team_name of TEAM entity. The team that is currently playing at their main stadium for this game.
- Away_team: Foreign Key, refers to Team_name of TEAM entity. The team that is currently visiting at another team's main stadium for this game.
- Home_fscore: The final score of the Home_team for this game.
- Away_fscore: The final score of the Away_team for this game.

GAME_REVENUE

The GAME_REVENUE weak entity measures the revenue aspect of the GAME area of interest. It encompasses all the aspects of a Game that can generate revenue for the NBA.

The main attributes of GAME_REVENUE include:

- Game_id: Foreign Key, refers to the Game_id attribute of GAME entity.
- Season_yr: Foreign Key, refers to the Season_yr attribute of SEASON entity.
- G_ticket_rev: Total revenue of ticket sales associated to a specific Game.
- G_concession_rev: Total revenue of concession stand sales associated to a specific game.
- G_store_rev: Total revenue of in-stadium merchandise sales associated to a specific game.
- G_sponsor_rev: Total revenue of sponsorship and brand deals presented during a specific game.
- G_media_rev: Total revenue of network and media involvement that took part in presenting and demonstrating the specific game.

STADIUM

The STADIUM entity defines the attributes of a single Stadium in the NBA, and includes the main characteristics that describe a Stadium.

The main attributes of STADIUM include:

- Stadium_id: Primary Key, a unique identification number assigned to each Stadium.
- Stadium_name: The full name of the stadium.
- Stadium_location: The location of a stadium.
- Max_capacity: The maximum occupancy of fans a stadium can seat at a time.

STADIUM_REVENUE

The STADIUM_REVENUE weak entity measures the revenue aspect of the STADIUM area of interest. It encompasses all the aspects of a Stadium that can generate revenue for the NBA.

The main attributes of STADIUM_REVENUE include:

- Stadium_id: Foreign Key, refers to the Stadium_id attribute of STADIUM entity.
- Season_yr: Foreign Key, refers to the Season_yr attribute of SEASON entity.
- S_ticket_rev: Total revenue of ticket sales associated to a specific Stadium.
- S_membership_rev: Total revenue of longer-term membership fees for attending the Stadium. This includes season passes and other multi-game package deals.
- S_concession_rev: Total revenue of concession stand sales associated to a specific Stadium
- S_store_rev: Total revenue of in-stadium merchandise sales associated to a specific Stadium

MEDIA

The MEDIA entity defines the attributes of a Network service that supports live broadcasts of the NBA and associated shows, such as talk shows, documentaries, and replays of games. It also describes several characteristics that make up one of the Network services.

The main attributes of MEDIA include:

- Network_id: Primary Key, a unique identification number assigned to each Media Network.
- Network_name: The full name of a Network service.

- Network_channel (O): The channel that the network is typically found on (may differ between cable providers, or may be optional if a fan is watching via streaming services).

- Network_stream (O): The type of streaming service that the fan is using to stream a specific Network broadcast (may be optional if fan is watching via cable TV).

- N_rating_pct: The television ratings of the Network, which measures how many fans engage with the Network at any given game or show.

MEDIA_REVENUE

The MEDIA_REVENUE weak entity measures the revenue aspect of the MEDIA area of interest. It encompasses all the aspects of a MEDIA Network service that can generate revenue for the NBA.

The main attributes of MEDIA_REVENUE include:

- Network_id: Foreign Key, refers to the Network_id attribute of MEDIA entity.

- Season_yr: Foreign Key, refers to the Season_yr attribute of SEASON entity.

- N_viewership_rev: Total revenue generated from viewership of an associated Network broadcast.

- N_sponsor_rev: Total revenue from sponsors and advertisements broadcasted within an associated Network service.

- N_subscription_rev: Total revenue generated from subscriptions to certain methods of watching the Network. This includes cable packages that require payment to watch, or memberships from streaming services.

3.2.2 Relationships

SEASON < - > PLAYER_REVENUE

- **Cardinality:** 1 Mandatory : N Optional

    - SEASON to PLAYER_REVENUE, one instance of a season can have multiple instances of player revenue, or it can have no revenue generated for that given season (extremely unlikely, but worth considering for the sake of inclusion).

    - PLAYER_REVENUE to SEASON, one instance of a player-generated revenue must be associated with a specific season, and it cannot exist without a season tied to it.

- **Relationship Identity**: Identifying
    - The PLAYER_REVENUE entity is a weak entity, and it cannot exist without inheriting the Primary Keys from SEASON and PLAYER entities. This is an intersection entity.

SEASON < - > TEAM_REVENUE

- **Cardinality:** 1 mandatory : N Optional
    - SEASON to TEAM_REVENUE, one instance of a season can have multiple instances of team revenue, or it can have no revenue generated for that given season.
    - TEAM_REVENUE to SEASON, one instance of a team-generated revenue must be associated with a specific season, and it cannot exist without a season tied to it.
- **Relationship Identity**: Identifying

    The TEAM_REVENUE entity is a weak entity, and it cannot exist without inheriting the Primary Keys from SEASON and TEAM entities. This is an intersection entity.

SEASON < - > GAME_REVENUE

- **Cardinality:** 1 mandatory : N Optional
    - SEASON to GAME_REVENUE, one instance of a season can have multiple instances of game revenue, or it can have no revenue generated for that given season.
    - GAME_REVENUE to SEASON, one instance of a game-generated revenue must be associated with a specific season, and it cannot exist without a season tied to it.
- **Relationship Identity:** Identifying

    The GAME_REVENUE entity is a weak entity, and it cannot exist without inheriting the Primary Keys from SEASON and GAME entities. This is an intersection entity.

SEASON < - > STADIUM_REVENUE

- **Cardinality:** 1 mandatory : N Optional
    - SEASON to STADIUM_REVENUE, one instance of a season can have multiple instances of stadium revenue, or it can have no revenue generated for that given season.

- STADIUM_REVENUE to SEASON, one instance of a stadium-generated revenue must be associated with a specific season, and it cannot exist without a season tied to it.

- **Relationship Identity:** Identifying

  The STADIUM_REVENUE entity is a weak entity, and it cannot exist without inheriting the Primary Keys from SEASON and STADIUM entities. This is an intersection entity.

SEASON < - > MEDIA_REVENUE

- **Cardinality:** 1 mandatory : N Optional

  - SEASON to MEDIA_REVENUE, one instance of a season can have multiple instances of media revenue, or it can have no revenue generated for that given season.

  - MEDIA_REVENUE to SEASON, one instance of a media-generated revenue must be associated with a specific season, and it cannot exist without a season tied to it

- **Relationship Identity:** Identifying

  The MEDIA_REVENUE entity is a weak entity, and it cannot exist without inheriting the Primary Keys from SEASON and MEDIA entities. This is an intersection entity.

SEASON < - > PLAYER_STAT

- **Cardinality:** 1 Mandatory : N Mandatory

  - SEASON to PLAYER_STAT, one instance of a season must have at least one statistical chart of a player, but can have as many player stats as needed.

  - PLAYER_STAT to SEASON, one instance of a player's statistical chart must be tied to one specific season.

- **Relationship Identity:** Identifying

  - The PLAYER_STAT entity is a weak entity, which cannot exist without inheriting the Primary Keys from SEASON and PLAYER entities. This is an intersection entity.

SEASON < - > PLAYOFF_SEED

- **Cardinality:** 1 Mandatory : 1 Mandatory

- SEASON to PLAYOFF_SEED, one instance of a season must contain one and exactly one playoff seeding chart, since the PLAYOFF_SEED entity already defines all eight playoff seeds in one instance.

- PLAYOFF_SEED to SEASON, one instance of a playoff seeding chart must be tied to one and exactly one season.

- **Relationship Identity:** Identifying

    - PLAYOFF_SEED entity is a weak entity, which cannot exist without inheriting the Primary Keys from SEASON and CONFERENCE. This is an intersection entity.

TEAM < - > TEAM_REVENUE

- **Cardinality:** 1 Mandatory : N Optional

    - TEAM to TEAM_REVENUE, one instance of a Team can contain several instances of the Team's generated revenue for a season, with the possible event that no revenue may be generated for a Team as well.

    - TEAM_REVENUE to TEAM, one instance of a Team's generated revenue must be associated with a specific Team.

- **Relationship Identity:** Identifying

    - TEAM_REVENUE is a weak entity that cannot exist without inheritance of the TEAM and SEASON primary keys. This is also an intersection entity that connects the M:N relationship between TEAM and SEASON.

TEAM < - > PLAYER

- **Cardinality:** 1 Optional : N Mandatory

    - TEAM to PLAYER, one Team can hold at least one Player, and up to N number of players at once.

    - PLAYER to TEAM, one Player can only play for at most one Team at a time. They also have the possibility of playing for no Team at any given time.

- **Relationship Identity:** Non-Identifying

    - The two entities are both strong entities, meaning that they do not depend entirely on each other's Primary Keys in order to exist. The PLAYER entity contains a Foreign Key to the Team_name attribute of the TEAM entity.

TEAM < - > CONFERENCE

- **Cardinality:** M Mandatory : 1 Mandatory
    - TEAM to CONFERENCE, one instance of a Team can be put into exactly one conference.
    - CONFERENCE to TEAM, one instance of a Conference can have at least one team but can also have multiple teams at a time.
- **Relationship Identity:** Non-Identifying
    - The two entities are both strong entities, meaning that they do not depend entirely on each other's Primary Keys in order to exist. The TEAM entity contains a Foreign Key to the Conference_name attribute of the CONFERENCE entity.

TEAM < - > GAME

- **Cardinality:** M Mandatory : 1 Mandatory
    - TEAM to GAME, one instance of a Team must play in at least one Game in a Season, but can play multiple games in one Season.
    - Game to TEAM, one instance of a Game can only contain one and only one team per Foreign Key attribute. This must be specified since GAME has two FKs to TEAM entity: one for Home_team, and one for Away_team.
- **Relationship Identity:** Non-Identifying
    - The two entities are both strong entities, meaning that they do not depend entirely on each other's Primary Keys in order to exist.

TEAM < - > STADIUM

- **Cardinality:** M Mandatory : 1 Mandatory
    - TEAM to STADIUM, one instance of a Team can be put into exactly one Stadium.
    - STADIUM to TEAM, one instance of a Stadium can have at least one team, but can also have multiple teams at a time.
- **Relationship Identity:** Non-Identifying
    - The two entities are both strong entities, meaning that they do not depend entirely on each other's Primary Keys in order to exist. The TEAM entity contains a Foreign Key to the Stadium_id attribute of the STADIUM entity.

TEAM < - > HEAD_COACH

- **Cardinality:** 1 Optional : 1 Mandatory
    - TEAM to HEAD_COACH, one instance of a Team must have one Head Coach leading the team at all times.
    - HEAD_COACH to TEAM, one instance of a Head Coach may coach up to one Team at a time. The Head Coach may also be currently coaching no team at the time.
- **Relationship Identity:** Non-Identifying
    - The two entities are both strong entities, meaning that they do not depend entirely on each other's Primary Keys in order to exist. The HEAD_COACH entity contains a Foreign Key to the Team_name attribute of the TEAM entity.

TEAM < - > OWNER

- **Cardinality:** 1 Optional : 1 Mandatory
    - TEAM to OWNER, one instance of a Team must have one Owner representing the Team at all times.
    - OWNER to TEAM, one instance of a Owner may represent up to one Team at a time. The Owner may also be currently representing no team at the time.
- **Relationship Identity:** Non-Identifying
    - The two entities are both strong entities, meaning that they do not depend entirely on each other's Primary Keys in order to exist. The OWNER entity contains a Foreign Key to the Team_name attribute of the TEAM entity.

PLAYER < - > PLAYER_REVENUE

- **Cardinality:** 1 Mandatory : N Optional
    - PLAYER to PLAYER_REVENUE, one instance of a Player may contain multiple data instances of a player's revenue. In other words, a player can have revenue data from multiple seasons.
    - PLAYER_REVENUE to PLAYER, one instance of the player's revenue has to be associated to one player, in order for this instance to exist.
- **Relationship Identity:** Identifying

- PLAYER_REVENUE is a weak entity that cannot exist without inheritance of the PLAYER and SEASON primary keys. This is also an intersection entity that connects the M:N relationship between PLAYER and SEASON

PLAYER < - > PLAYER_STAT

- **Cardinality:** 1 Mandatory : N Mandatory
    - PLAYER to PLAYER_STAT, one instance of a Player may contain at least one season of stats associated to the Player. The Player can also have no completed stats yet, in the event of first-season Rookies and long-term absences.
    - PLAYER_STAT to PLAYER, one instance of a player's stats must be associated to one player, in order for the instance to exist.
- **Relationship Identity:** Identifying
    - PLAYER_STAT is a weak entity that cannot exist without inheritance of the PLAYER and SEASON primary keys. This is also an intersection entity that connects the M:N relationship between PLAYER and SEASON

GAME < - > GAME_REVENUE

- **Cardinality:** 1 Mandatory : 1 Optional
    - GAME to GAME_REVENUE, one instance of a game can have up to one instance of the Game's Revenue. In other words, a Game revenue instance is unique to one particular game. The game may also generate no revenue for that particular instance.
    - GAME_REVENUE to GAME, one instance of a Game's Revenue must be associated to one Game, in order for this instance to exist.
- **Relationship Identity:** Identifying
    - GAME_REVENUE is a weak entity that cannot exist without inheritance of the GAME and SEASON primary keys. This is also an intersection entity that connects the M:N relationship between GAME and SEASON

GAME < - > STADIUM

- **Cardinality:** M Mandatory : 1 Mandatory
    - GAME to STADIUM, one instance of a Game must be taking place at one and exactly one Stadium location.

- STADIUM to GAME, one instance of a Stadium must be holding at least one Game within it. The Stadium can also hold many games within it, which is typical in a season.

- **Relationship Identity:** Non-Identifying

    - The two entities are both strong entities, meaning that they do not depend entirely on each other's Primary Keys in order to exist. The GAME entity contains a Foreign Key to the Stadium_location attribute of the STADIUM entity.

STADIUM < - > STADIUM_REVENUE

- **Cardinality:** 1 Mandatory : N Optional

    - STADIUM to STADIUM_REVENUE one instance of a Stadium must contain at least one instance of Revenue genereated from the Stadium. It may also contain no Revenue instances if none was generated for that season.

    - STADIUM_REVENUE to STADIUM, one instance of a stadium's generated revenue must be associated to a Stadium, in order for this instance to exist.

- **Relationship Identity:** Identifying

    - STADIUM_REVENUE is a weak entity that cannot exist without inheritance of the STADIUM and SEASON primary keys. This is also an intersection entity that connects the M:N relationship between STADIUM and SEASON.

MEDIA < - > MEDIA_REVENUE

- **Cardinality:** 1 Mandatory : N Optional

    - MEDIA to MEDIA_REVENUE one instance of Media must contain at least one instance of Revenue generated from the Media. It may also contain no Revenue instances if none was generated for that season.

    - MEDIA_REVENUE to MEDIA, one instance of a Media's generated revenue must be associated to a Media, in order for this instance to exist.

- **Relationship Identity:** Identifying

    - MEDIA_REVENUE is a weak entity that cannot exist without inheritance of the MEDIA and SEASON primary keys. This is also an intersection entity that connects the M:N relationship between MEDIA and SEASON.

CONFERENCE < - > PLAYOFF_SEED

1. **Cardinality:** 1 Mandatory : N Mandatory

    - CONFERENCE to PLAYOFF_SEED, one instance of a Conference must contain at least one instance of a playoff seed standings. It may hold up to N playoff seed standings, with each instance being a different season.

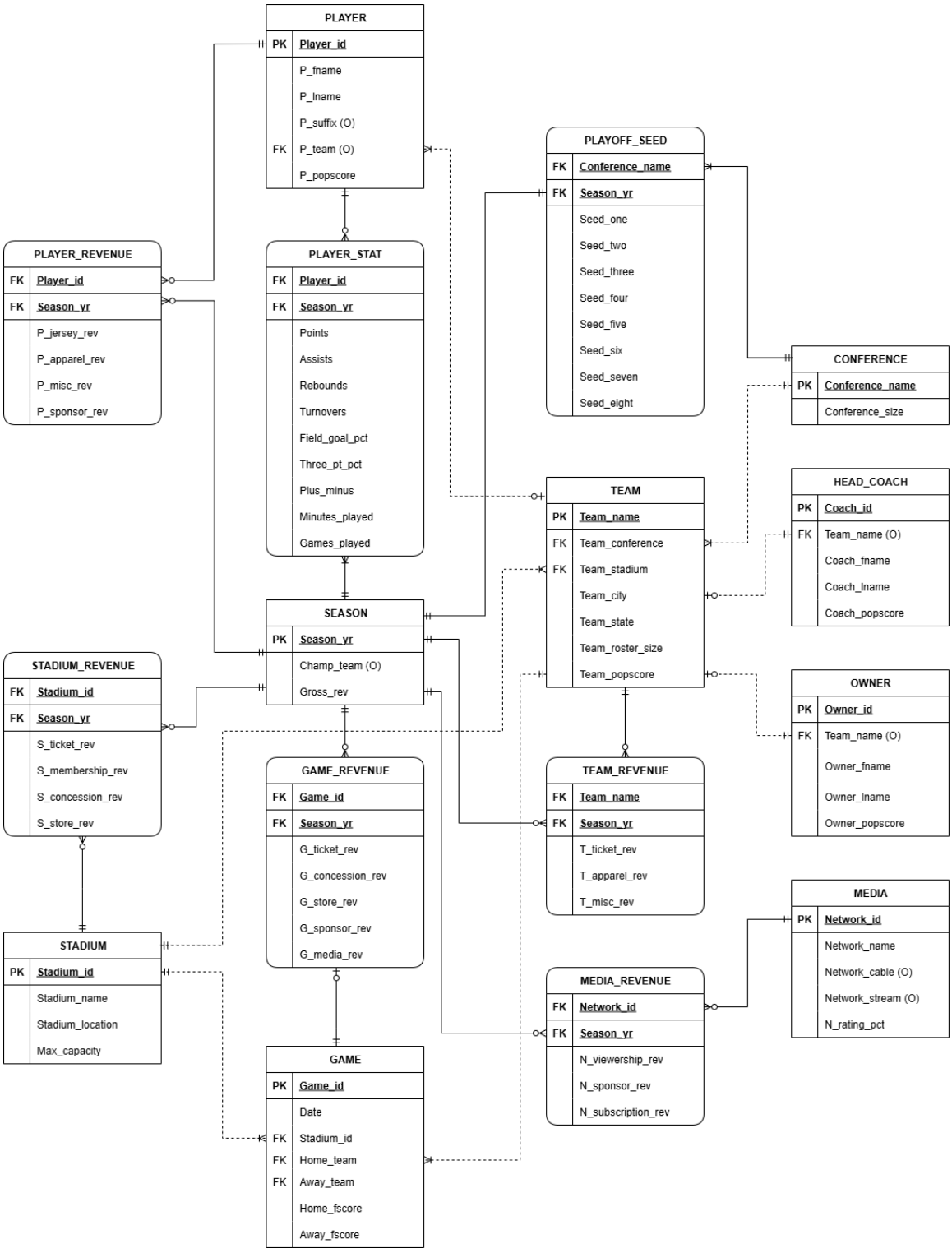
    - PLAYOFF_SEED to CONFERENCE, one instance of a playoff seeding must be associated to one and only one conference, in order for this instance to exist.

2. **Relationship Identity**: Identifying

    - PLAYOFF_SEED is a weak entity that cannot exist without inheritance of the CONFERENCE and SEASON primary keys. This acts as an intersection entity that connects and resolves the M:N relationship between CONFERENCE and SEASON.

3.2.3 ER Diagram

Along with the pasted diagram, please see attached file “NBA DATABASE – ERD.”


PLAYER
PK Player_id
P_fname
P_lname
P_suffix (O)
FK P_team (O)
P_popscore
PLAYOFF_SEED
FK Conference_name
FK Season_yr
Seed_one
Seed_two
Seed_three
Seed_four
Seed_five
Seed_six
Seed_seven
Seed_eight
PLAYER_REVENUE
FK Player_id
FK Season_yr
P_jersey_rev
P_apparel_rev
P_misc_rev
P_sponsor_rev
PLAYER_STAT
FK Player_id
FK Season_yr
Points
Assists
Rebounds
Turnovers
Field_goal_pct
Three_pt_pct
Plus_minus
Minutes_played
Games_played
CONFERENCE
PK Conference_name
Conference_size
TEAM
PK Team_name
FK Team_conference
FK Team_stadium
Team_city
Team_state
Team_roster_size
Team_popscore
HEAD_COACH
PK Coach_id
FK Team_name (O)
Coach_fname
Coach_lname
Coach_popscore
SEASON
PK Season_yr
Champ_team (O)
Gross_rev
STADIUM_REVENUE
FK Stadium_id
FK Season_yr
S_ticket_rev
S_membership_rev
S_concession_rev
S_store_rev
OWNER
PK Owner_id
FK Team_name (O)
Owner_fname
Owner_lname
Owner_popscore
GAME_REVENUE
FK Game_id
FK Season_yr
G_ticket_rev
G_concession_rev
G_store_rev
G_sponsor_rev
G_media_rev
TEAM_REVENUE
FK Team_name
FK Season_yr
T_ticket_rev
T_apparel_rev
T_misc_rev
MEDIA
PK Network_id
Network_name
Network_cable (O)
Network_stream (O)
N_rating_pct
STADIUM
PK Stadium_id
Stadium_name
Stadium_location
Max_capacity
MEDIA_REVENUE
FK Network_id
FK Season_yr
N_viewership_rev
N_sponsor_rev
N_subscription_rev
GAME
PK Game_id
Date
FK Stadium_id
FK Home_team
FK Away_team
Home_fscore
Away_fscore

3.3 Relational Model

This subsection discusses the Relational Model design of the NBA Financial Database. This includes Data Dictionary, integrity rules defined in the Database, and operational rules and operations that the user would expect to perform and follow.

3.3.1 Data Dictionary

SEASON

Column Name	Description	Data Type	Size	Constraint Type	Not Null?	Valid Values
Season_yr	Season Year	Char	9	Primary Key	Y	"YYYY-YYYY"
Champ_team	Name of Champion	VarChar	30		N	<= 30 Character strings
Gross_rev	Total Gross Revenue	Decimal	20	DEFAULT = 0.00	Y	<= 20 numeric digits

PLAYER

Column Name	Description	Data Type	Size	Constraint Type	Not Null?	Valid Values
Player_id	Player Identification Number	VarChar	9	Primary Key	Y	<= 9 numeric digits
P_fname	Player First name	VarChar	20		Y	<= 20 character string
P_lname	Player Last Name	VarChar	20		Y	<= 20 character string
P_suffix	Player suffix	VarChar	5		N	<= 5 character string
P_team	Player team	VarChar	30	Foreign Key	N	<= 30 character string

P_popscore	Player popularity score	Decimal	5	CHECK 0 <= x <= 100.00 DEFAULT = 0.00	Y	<= 5 numeric digits, two decimals

PLAYER_REVENUE

Column Name	Description	Data Type	Size	Constraint Type	Not Null?	Valid Values
Player_id	Player Identification Number	VarChar	9	Foreign Key	Y	<= 9 numeric digits
Season_yr	Season Year	Char	9	Foreign Key	Y	"YYYY-YYYY"
P_jersey_rev	Total Revenue of player jersey sales	Decimal	20	DEFAULT = 0.00	Y	<= 20 numeric digits
P_apparel_rev	Total revenue of other player apparel sales (no jerseys)	Decimal	20	DEFAULT = 0.00	Y	<= 20 numeric digits
P_misc_rev	Total revenue of player-associated miscellaneous products	Decimal	20	DEFAULT = 0.00	Y	<= 20 numeric digits
P_sponsor_rev	Total revenue of sponsorship deals with player	Decimal	20	DEFAULT = 0.00	Y	<= 20 numeric digits

PLAYER_STAT

Column Name	Description	Data Type	Size	Constraint Type	Not Null?	Valid Values
Player_id	Player Identification Number	VarChar	9	Foreign Key	Y	<= 9 numeric digits
Season_yr	Season Year	Char	9	Foreign Key	Y	"YYYY-YYYY"

Points	Total Points Scored	Int	4	DEFAULT = 0	Y	<= 4 numeric digits
Assists	Total Assists Made	Int	4	DEFAULT = 0	Y	<= 4 numeric digits
Rebounds	Total rebounds made	Int	4	DEFAULT = 0	Y	<= 4 numeric digits
Turnovers	Total turnovers made	Int	4	DEFAULT = 0	Y	<= 4 numeric digits
Field_goal_pct	Field Goal percentage	Decimal	5	CHECK 0 <= x <= 100.00 DEFAULT = 0.00	Y	<= 5 numeric digits, two decimals
Three_pt_pct	Three point percentage	Decimal	5	CHECK 0 <= x <= 100.00 DEFAULT = 0.00	Y	<= 5 numeric digits, two decimals
Plus_minus	Plus/Minus efficiency on court	Int	4	DEFAULT = 0	Y	<= 4 numeric digits
Minutes_played	Total minutes played	Int	4	DEFAULT = 0	Y	<= 4 numeric digits
Games_played	Total games played	Int	2	DEFAULT = 0	Y	<= 2 numeric digits

TEAM

Column Name	Description	Data Type	Size	Constraint Type	Not Null?	Valid Values
Team_name	Full Name of Team	VarChar	30	Primary Key	Y	<= 30 character string

Team_conference	Conference of Team	VarChar	30	Foreign Key	Y	<= 30 character string
Team_stadium	Home Stadium ID of Team	VarChar	3	Foreign Key	Y	<= 3 numeric digits
Team_city	City of Team	VarChar	30		Y	<= 30 character string
Team_state	State of Team	VarChar	30		Y	<= 30 character string
Team_roster_size	Current Roster Size of Team	Int	2	DEFAULT = 0	Y	<= 2 numeric digits
Team_popscore	Popularity score of Team	Decimal	5	CHECK 0 <= x <= 100.00 DEFAULT = 0.00	Y	<= 5 numeric digits, two decimals

TEAM_REVENUE

Column Name	Description	Data Type	Size	Constraint Type	Not Null?	Valid Values
Team_name	Full Name of Team	VarChar	30	Foreign Key	Y	<= 30 character string
Season_yr	Season Year	Char	9	Foreign Key	Y	"YYYY-YYYY"
T_ticket_rev	Total revenue of Team ticketing sales	Decimal	20	DEFAULT = 0.00	Y	<= 20 numeric digits
T_apparel_rev	Total revenue of Team apparel sales	Decimal	20	DEFAULT = 0.00	Y	<= 20 numeric digits

T_misc_rev	Total revenue of Team miscellaneous product sales	Decimal	20	DEFAULT = 0.00	Y	<= 20 numeric digits

HEAD_COACH

Column Name	Description	Data Type	Size	Constraint Type	Not Null?	Valid Values
Coach_id	Coach Identification Number	VarChar	9	Primary Key	Y	<= 9 Numeric digits
Team_name	Full Team name of Coach	VarChar	30	Foreign Key	N	<= 30 character string
Coach_fname	Coach first name	VarChar	20		Y	<= 20 character string
Coach_lname	Coach last name	VarChar	20		Y	<= 20 character string
Coach_popscore	Coach popularity score	Decimal	5	CHECK 0 <= x <= 100.00 DEFAULT = 0.00	Y	<= 5 numeric digits, two decimals

OWNER

Column Name	Description	Data Type	Size	Constraint Type	Not Null?	Valid Values
Owner_id	Owner identification number	VarChar	9	Primary Key	Y	<= 9 Numeric digits
Team_name	Full Team name of Owner	VarChar	30	Foreign Key	N	<= 30 character string

Owner_fname	Owner first name	VarChar	20		Y	<= 20 character string
Owner_lname	Owner last name	VarChar	20		Y	<= 20 character string
Owner_popscore	Owner popularity score	Decimal	5	CHECK 0 <= x <= 100.00 DEFAULT = 0.00	Y	<= 5 numeric digits, two decimals

GAME

Column Name	Description	Data Type	Size	Constraint Type	Not Null?	Valid Values
Game_id	Game Identification number	VarChar	9	Primary Key	Y	<= 9 numeric digits
Date	Date of Game instance	Date			Y	"YYYY-MM-DD"
Stadium_id	Identification number of Stadium	VarChar	30	Foreign Key	Y	<= 30 character string
Home_team	Name of Home team	VarChar	30	Foreign Key	Y	<= 30 character string
Away_team	Name of Away/Visiting team	VarChar	30	Foreign Key	Y	<= 30 character string
Home_fscore	Final score of Home team	Int	3	DEFAULT = 0	Y	<= 3 numeric digits
Away_fscore	Final Score of Away/Visiting team	Int	3	DEFAULT = 0	Y	<= 3 numeric digits

GAME_REVENUE

Column Name	Description	Data Type	Size	Constraint Type	Not Null?	Valid Values
Game_id	Game Identification number	VarChar	9	Foreign Key	Y	<= 9 numeric digits
Season_yr	Season Year	Char	9	Foreign Key	Y	"YYYY-YYYY"
G_ticket_rev	Total revenue of a game's ticketing sales	Decimal	20	DEFAULT = 0.00	y	<= 20 numeric digits
G_concession_rev	Total revenue of concession sales	Decimal	20	DEFAULT = 0.00	y	<= 20 numeric digits
G_store_rev	Total revenue of store and emrchandise sales	Decimal	20	DEFAULT = 0.00	y	<= 20 numeric digits
G_sponsor_rev	Total revenue from sponsorships and advertisement	Decimal	20	DEFAULT = 0.00	y	<= 20 numeric digits
G_media_rev	Total revenue from Media engagement	Decimal	20	DEFAULT = 0.00	y	<= 20 numeric digits

STADIUM

Column Name	Description	Data Type	Size	Constraint Type	Not Null?	Valid Values
Stadium_id	Stadium Identification Number	VarChar	3	Primary Key	Y	<= 3 Numerical digits
Stadium_name	Full name of Stadium	VarChar	30		Y	<= 30 character string

Stadium_location	Full name of Location of Stadium	VarChar	30		Y	<= 30 character string
Max_capacity	Maximum occupancy of Stadium	Int	5		Y	5 Numerical digits

STADIUM_REVENUE

Column Name	Description	Data Type	Size	Constraint Type	Not Null?	Valid Values
Stadium_id	Stadium Identification Number	VarChar	3	Foreign Key	Y	<= 3 Numerical digits
Season_yr	Season Year	Char	9	Foreign Key	Y	"YYYY-YYYY"
S_ticket_rev	Total revenue from ticketing sales	Decimal	20	DEFAULT = 0.00	Y	<= 20 numeric digits
S_membership_rev	Total revenue from memberships and passes to Stadium	Decimal	20	DEFAULT = 0.00	Y	<= 20 numeric digits
S_concession_rev	Total revenue from concession sales	Decimal	20	DEFAULT = 0.00	Y	<= 20 numeric digits
S_store_rev	Total revenue from store and merchandise sales	Decimal	20	DEFAULT = 0.00	Y	<= 20 numeric digits

MEDIA

Column Name	Description	Data Type	Size	Constraint Type	Not Null?	Valid Values

Network_id	Network Identification Number	VarChar	3	Primary Key	Y	<= 3 numeric digits
Network_name	Full name of Network Provider	VarChar	20		Y	<= 20 character string
N_rating_pct	Television/Streaming ratings of Network, Fan engagement	Decimal	5	CHECK 0 <= x <= 100.00 DEFAULT = 0.00	Y	<= 5 numeric digits, two decimals

MEDIA_REVENUE

Column Name	Description	Data Type	Size	Constraint Type	Not Null?	Valid Values
Network_id	Network Identification Number	VarChar	3	Foreign Key	Y	<= 3 numeric digits
Season_yr	Season Year	Char	9	Foreign Key	Y	"YYYY-YYYY"
N_viewership_rev	Total revenue from Network viewership	Decimal	20	DEFAULT = 0.00	Y	<= 20 numeric digits
N_sponsor_rev	Total revenue from sponsorship and advertisement	Decimal	20	DEFAULT = 0.00	Y	<= 20 numeric digits
N_subscription_rev	Total revenue from subscription packages	Decimal	20	DEFAULT = 0.00	Y	<= 20 numeric digits

CONFERENCE

Column Name	Description	Data Type	Size	Constraint Type	Not Null?	Valid Values

Conference_name	Full Name of conference	VarChar	20	Primary Key	Y	<= 20 Character string
Conference_size	Total size of conference	Int	2	DEFAULT = 0	Y	<= 2 numeric digits

PLAYOFF_SEED

Column Name	Description	Data Type	Size	Constraint Type	Not Null?	Valid Values
Conference_name	Full Name of conference	VarChar	20	Foreign Key	Y	<= 20 Character string
Season_yr	Season Year	Char	9	Foreign Key	Y	"YYYY-YYYY"
Seed_one	Full name of first seed team	VarChar	30		N	<= 30 character string
Seed_two	Full name of second seed team	VarChar	30		N	<= 30 character string
Seed_three	Full name of third seed team	VarChar	30		N	<= 30 character string
Seed_four	Full name of fourth seed team	VarChar	30		N	<= 30 character string
Seed_five	Full name of fifth seed team	VarChar	30		N	<= 30 character string
Seed_six	Full name of sixth seed team	VarChar	30		N	<= 30 character string
Seed_seven	Full name of seventh seed team	VarChar	30		N	<= 30 character string

Seed_eight	Full name of eighth seed team	VarChar	30		N	<= 30 character string

3.3.2 Integrity Rules

Several integrity rules and constraints were used to define and regulate the Database:

Domain Constraints:

Domain Constraints are a form of integrity constraint that ensures values that are entered into a column/attribute are valid values and meet the criteria that the column/attribute has established. Some constraints are inherent in SQL, such as constraints that are associated to a specified data type, and some constraints were introduced during database development. For attribute-specific constraints, criteria such as size and a range of valid input values were also provided for certain groups of similar attributes. Lastly, NULL constraints were also addressed throughout the design process to prevent some values from being NULL while allowing other values to be NULL, or optional.

The Data Types that were used throughout the Database were:

- **Char:** A value under this type must be in the form of a character or string of characters, where the format and length of the string DOES matter and must be specified.
    - This was solely used for the Season_yr attribute, where a user must enter a year in the format "YYYY-YYYY," such as the format "2024-2025" for the 24'-25' season. Entering any other format would generate an error.

- **VarChar:** A value under this type would be in the form of a character or string of characters, where the format and length DOES NOT matter, hence "VARiable CHARacter." Because of the flexibility of the string inputs under VarChar, this data type was used very frequently across attributes, including:
    - **Name attributes ("P_lname","Team_name", "Stadium_name", etc**.). Size of the name attributes were specified, with naming attributes in entities such as PLAYER, OWNER, and HEAD_COACH having a maximum size of 20 characters, while naming attributes in entities like TEAM, CONFERENCE, and STADIUM have a maximum size of 30 characters. For example, some naming constraints were specific as "<= 30 character string" to indicate a valid input must have less than or equal to 30 characters in it. Naming attributes are required to have a value in them, so these are NOT NULL.
    - **Location attributes ("Stadium_location", "Team_city", "Team_state" etc.).** Location attributes have a size of <= 30 characters associated to them. Location attributes are required to have a value in them, so these are NOT NULL.
    - **Identification attributes ("Game_id", "Owner_id", "Player_id", etc.).** Each non-revenue entity (minus TEAM entity) has an identification attribute associated with it. The size of the ID depends on the expected number of instances in the NBA – large numbers of instances such as PLAYER, GAME and

OWNER have a size of <= 9 numerical digits to encompass a large number of Players and Personnel. Entities that expect to have a smaller number of instances, such as STADIUM, TEAM, and MEDIA, have a size of <= 3 numerical digits. These are required to have, and in most cases are PRIMARY LEYS or FOREIGN KEYS of the entity, so these have to be NOT NULL.

- **Staffing Team attributes ("P_team", "OWNER.Team_name", "HEAD_COACH.Team_name").** For these Staff-related attributes, because staff are not required to be in a Team at any given time, the attributes for their respective Team Names allow NULL values and are optional. If a value is entered, then similar rules to the Name attributes apply.
- **Seeding Attributes ("Seed_one", "Seed_two", etc.)** These attributes allow NULL values, in the event that the season has yet to be completed. Otherwise, they behave the same way as Name Attributes.

- **Int:** A value under this type would be in the form of numerical digits. This would be in the form of an integer, which does not include significant figures or decimals. This was used in many of the Statistical attributes of the NBA, as well as wherever a total did not need to be tracked in decimal form:
  - **Many PLAYER_STAT attributes ("Points", "Assists", "Minutes_played", etc.).** These attributes were given a size of <= 4 numeric digits to account for data in the thousands, which is common in one season for an NBA player. Each of these attributes are required to have a data value inputted in it – however, in the event that a Player records no data for a given stat in a season, a DEFAULT = 0 constraint is in place if no data was recorded, thus preventing NULL values. These attributes are all NOT NULL.

  - **Size attributes ("Team_roster_size", "Max_capacity", Conference_size", etc.).** Size attributes were given a size definition specific to the expected range of the attribute's size. Team size and conference size were given sizes of <= 2 numeric digits as there are much less instances of these attributes. Max_capacity of the STADIUM entity was given a size of <= 5 numeric digits, since a Stadium can contain an abundance of fans within one Stadium. These attributes are NOT NULL with a DEFAULT = 0 constraint.

  - **Scoring attributes ("Home_score", "Away_score").** Scoring attributes are given a size of <= 3 numeric digits. These attributes are NOT NULL, with a DEFAULT = 0 constraint.

- **Decimal:** A value under this type would be in the form of numerical digits. A Decimal number includes decimal places and significant figures. This was therefore used for many of the percentage attributes, as well as all revenue attributes, where a total needed to be tracked in decimal form for better clarity. To ensure that inputted valued are read up to two decimal places, I also specified "two decimals" for each Decimal type valid value. Other constraints were also used to specify valid values:
  - **Popularity scoring attributes ("Team_popscore", "Player_popscore", etc.).** Popularity scoring for this Database is based on a percentage rating of the player's popularity. A size of 5 was given to the Popularity attributes, to encompass percentage ratings up to two decimal places "100.00". A CHECK constraint was put in place to ensure the value is a valid percentage format:

CHECK 0 <= x <= 100.00. These attributes are NOT NULL, with a DEFAULT = 0.00 constraint.

- **Percentage attributes ("Field_goal_pct", "Three_pt_pct", "N_rating_pct").** Percentage attributes were given a size of 5, to encompass percentage values up to two decimal places "100.00". A CHECK constraint was also initiated for all percentage attributes: CHECK 0 <= x <= 100.00. These values are NOT NULL, with a DEFAULT = 0.00 constraint.

- **Revenue attributes ("N_viewership_rev", "P_jersey_rev". etc.).** Revenue attributes were given a size of <= 20 numeric digits, to encompass a large range of data. The two decimal criteria was also established for these attributes. Revenue attributes are also NOT NULL, with a DEFAULT = 0.00 constraint.

- **Date:** A value under this Data Type must obey a strict formatting guideline for defining a date of an occurrence. For example, in MySQL the only acceptable format for Date value types is "YYYY-MM-DD."
  - **Only one attribute uses the Date DataType (Date).** Located in the GAME entity, this attribute specifies when a Game instance took place. This attribute is NOT NULL and is mandatory for a GAME instance to occur.

Referential Constraints:

The structure of the Database relies heavily on referential constraints to create relations across entities and tables. It helps ensure that references across different tables are valid references, and that only specific tables can reference other specific tables:

- **Primary Keys (PK):** These are attributes that must exist and be unique for every instance of an Entity. Every strong entity must have a Primary Key associated with it. The strong entities and associated Primary keys in this Database are:
  - **"SEASON.Season_yr"**
  - **"PLAYER.Player_id"**
  - **"TEAM.Team_name"**
  - **"HEAD_COACH.Coach_id"**
  - **"OWNER.Owner_id"**
  - **"GAME.Game_id"**
  - **"STADIUM.Stadium_id"**
  - **"MEDIA.Network._id"**
  - **"CONFERENCE.Conference_id"**

- **Foreign Keys (FK):** These are attributes that, in order to exist, must reference the Primary Key of another Entity/Table to obtain and inherit that value. Foreign Keys are mainly used in two scenarios for this Database:
  - **Intersectional Entity Relationships:** Several Entities are defined as "Weak entities," or entities that cannot uniquely exist on its own and must inherit an attribute from other entities in order to uniquely exist. For this particular Database, the weak entities are in the form of Intersectional Entities, which are necessary entities that are placed between two other entities to prevent M:N relationships between those two. These intersectional entities use the Primary Keys of the two entities that they are bridging as their own unique identifiers,

therefore using two Foreign Keys. These are used heavily to specify the Season that the data is currently tracking:

- **"PLAYER_REVENUE" FK to "SEASON.Season_yr" and "PLAYER.Player_id"**
- **"PLAYER_STAT" FK to "SEASON.Season_yr" and "PLAYER_Player_id"**
- **"TEAM_REVENUE" FK to "SEASON.Season_yr" and "TEAM.Team_name"**
- **"GAME_REVENUE" FK to "SEASON.Season_yr" and "GAME.Game_id"**
- **"STADIUM_REVENUE" FK to "SEASON.Season_yr" and "STADIUM.Stadium_id"**
- **"MEDIA_REVENUE" FK to "SEASON.Season_yr" and "MEDIA.Network_id"**
- **"PLAYOFF_SEED" FK to "SEASON.Season_yr" and "CONFERENCE.Conference_name"**

- **Non-Intersectional Entity Reference:** These Foreign Keys are not within an Intersectional Entity, but rather are their own instance of a reference to another table. These FKs are also generally not required to uniquely define the entity that it resides in:
  - **"PLAYER.P_team" FK to "TEAM.Team_name"**
  - **"TEAM.Team_conference" FK to "CONFERENCE.Conference_name"**
  - **"TEAM.Team_stadium" FK to "STADIUM.Stadium_id"**
  - **"HEAD_COACH.Team_name" FK to "TEAM.Team_name"**
  - **"OWNER.Team_name" FK to "TEAM.Team_name"**
  - **"GAME.Stadium_id" FK to "STADIUM.Stadium_id"**
  - **"GAME.Home_team" FK to "TEAM.Team_name"**
  - **"GAME.Away_team" FK to "TEAM.Team_name"**

3.3.3 Operational Rules

This section covers some operational constraints that the User and Database must follow:

- Certain privileges and limitations are defined for a User prior to a User interacting with the Database. These roles are specified in Section 3.4.
- Identification constraints across all Player and Staff entities allow for repeated names to be entered, with no restrictions.
- For any single instance of GAME, the Home_team attribute and Away_team attribute cannot be the same.
- For any single instance of PLAYOFF_SEED, each attribute name cannot be the same as any of the other attributes.
- Users are to enter Data in specified formats. For example, when specifying a PLAYER_REVENUE instance, User must be sure to specify both a Player_ID and Season_yr. Only specifying one will generate an error.

3.3.4 Operations

This section covers the operations that are required for Database use cases:

1. **Querying of Data for:**
    a. **Revenue Tracking Purposes:** User can use SELECT and JOIN features to query certain revenue attributes, such as all ticketing revenue data across the revenue streams. Users can also use SUM() and other aggregate functions to add revenue attributes together, such as achieving the total revenue from all of a Player's revenue attributes.
    b. **Accessing Seasonal Archives:** User can use SELECT and JOIN features to query certain non-revenue attributes for archive retrieval, such as retrieving total points scored by a Player.
2. **Insertion/Deletion of New Non-Revenue Data:**
    a. **For Insertion of New Players and Staff (Rookies, Newcomers, etc.):** Insertion instances using INSERT can be performed by certain Users to add new personnel into the Database. Similarly, Users can DELETE certain instances, for the purposes of storage cleanup or invalid entries. These actions can take place at any point throughout the year.
    b. **For Insertion of Game Instances:** A new Game instance can and should be Inserted throughout the Season by the User.
    c. **For Insertion of New Teams, Conferences, Stadiums, or Media**: In a rarer case, Users can INSERT or DELETE instances of these entities at any point throughout the year. No restrictions are applied to the number of these entities, so these actions are valid.
    d. **For Insertion of Seeding for Playoffs**: Once the Regular Season ends, the User can begin inserting Teams into a new instance of the PLAYOFF_SEED table.
3. **Updating Non-Revenue Data:**
    a. **For Trades and Management Reorganization**: User can perform UPDATE statements to update staffing attributes throughout the Database. If a Player gets traded to another Team, or a Coach gets fired, then the Database must be updated promptly to reflect this.
    b. **For Popularity Score Calculations**: User can UPDATE the Popularity Score of Player and Staff entities at any point in time throughout the year, given that these scores are to be calculated at the end of each month.
    c. **For Updating Attributes of Teams, Games, Conferences, Stadiums, or Media**: User can perform UPDATE statements to update information on these entities. These could be in the event of rebranding of Teams, relocation events, or general changes. For Games, these could be in the event of score adjustments.
4. **Insertion/Deletion of New Revenue Data:**
    a. **For Creation of New Seasonal Tracking**: Users can INSERT new instances of Revenue entities for all associated entities at the beginning of each Season. Users can also DELETE Revenue instances in the event of invalid entries or storage cleanup.
5. **Updating Revenue Data:**

a. **For Seasonal Tracking**: Users can UPDATE any Revenue Attribute with new amounts and any point throughout the Season, given that Revenue is typically calculated at the end of every month.

6. **Inserting a New Season**:
    a. **For the Start of a New Tracking Season**: Limited to only the highest User Role, this User can initiate the INSERT of a new Season to essentially create a clean Database tracker for that Seasonal year. A DELETE function is also granted to this User, however these are almost never performed, and restrictions are applied to prevent accidental deletions.

3.4 Security

Security of the Database is ensured through both User-specific job roles and implementation of extra security measures through stored procedures and prepared statements.

User-specific job roles were specified through the use fo roles and privileges. These are divided into 4 Tiers of specializations:

- Tier 1: Basic access to Database. User can only view previous records and design simple queries to SELECT information. No access to INSERT, DELETE, or UPDATE actions.
- Tier 2a: Users are able to perform INSERT, DELETE, and UPDATE actions on Non-Revenue entities only, with a restriction from SEASON entity and Revenue entities.
- Tier 2b: Users are able to perform INSERT, DELETE, and UPDATE actions on Revenue entities only, with a restriction from SEASON entity and Non-Revenue entities.
- Tier 3: Users are able to perform INSERT, DELETE, and UPDATE actions on all entities, except for the SEASON entity.
- Tier 4: Users are able to perform INSERT, DELETE, and UPDATE actions on all entities, including SEASON entity for addition of new seasonal data.

Additionally, stored procedures were implemented in the Database to prevent the threat of SQL injection. These stored procedures would place a check constraint to ensure that specific data types were entered into the imputed parameters.

3.5 Database Backup and Recovery

Database backup is executed by using the command mysqldump in the Windows terminal command line. The command mysqldump essentially returns all SQL code from the Database for the user, including all created DDLs and DMLs, as well as triggers and stored procedures.

3.6 Using Database Design or CASE Tool

The ERD drawing application Draw.io was used to generate the ER model shown in an earlier section. This model was then translated into the MySQL Workbench 8.0 application. MySQL Workbench GUI was then utilized for the implementation of the Database and writing of all SQL statements.

3.7 Other Possible ER Relationships

There were several designs that were made before creating the final ERD. For instance, the initial design of this Database surprisingly had no specific revenue entities at all; the first idea was to create a more general SALES entity for each entity, which tracked the number of sales made for a specific item, as well as the total profit made from those sales. For instance, a Player's Jersey that has 5 sales in a given day would be given each a separate instance ("sale_no: 1, sale_no: 2, etc.), and each instance would track the Jersey's attributes and how much profit was made all in the same entity. This ended up being extremely complicated and redundant in tracking both sale number and profit; therefore, I chose to track only the revenue aspect of the sales, as this amount could be added to and deleted from much more easily, and proves to be a more unified tracker amongst factors of the NBA.

From this, the initial design that included the Revenue aspect had Revenue entities as a strong entity. Although this could still technically work with little flaw, the main issue was more due to the other Non-Revenue entities and how to relate them together. Many of these Non-Revenue entities existed as M:N relationships that still required a form of junction to resolve. This led to the idea that the Revenue attributes could act as those junction, as this would solve both the M:N relationship issues, as well as provide the database with a much better technique of tracking revenue by Season.

Additionally, some smaller alternatives were also considered for a few entities. For Stadium, I could also include average attendance to gauge another aspect of Stadium performance. I could also extend this to utilize the max_occupancy attribute to scale the average attendance amount; since each stadium has a cap on how many fans can be in the building at once, a ratio of average attendance to maximum occupancy could give us a more accurate representation of how many fans attend the stadium.

Lastly, I heavily considered implementing a running total Revenue counter across all five of the Revenue streams, instead of having the User specify queries to obtain totals. There were two ideas that were attempted:

- Each revenue stream would have its own total Revenue attribute (Ex. PLAYER_REVENUE has attribute "P_total_rev"). This attribute would be automatically updated after the addition of Revenue to any of the rest of that entity's attributes. If an UPDATE was performed to add money to a "P_jersey_rev" or "P_misc_rev" attribute of PLAYER_REVENUE, then that amount of money would also automatically be added to the P_total_rev attribute. I attempted to implement this as a trigger to add to the total Revenue attribute after

every addition of Revenue, using AFTER UPDATE, however this was not possible as it would use another instance of UPDATE within the trigger – the one used to update the total amount. This generated an error, as we cannot update the same entity within a trigger of the same entity.

- From this, I considered creating another table, called SEASON_TOTAL, that would house the totals for all five revenue streams. Any addition to Revenue attributes in any of the revenue streams would automatically update the SEASON_TOTAL associated attributes. This, however, also proved to be troublesome to implement.

4. Implementation Description

This section discusses general implementation requirements that are used for the Database, including the Data Dictionary, advanced features used, and Queries for primary Database usage.

4.1 Data Dictionary

The Data Dictionary can be found through an attached PDF entitled "NBA Database – Data Dictionary." These definitions were obtained by using the DEFINE statements specified below:

```
214 •  DESCRIBE season;
215 •  DESCRIBE conference;
216 •  DESCRIBE stadium;
217 •  DESCRIBE team;
218 •  DESCRIBE player;
219 •  DESCRIBE game;
220 •  DESCRIBE head_coach;
221 •  DESCRIBE owner;
222 •  DESCRIBE media;
223 •  DESCRIBE player_revenue;
224 •  DESCRIBE team_revenue;
225 •  DESCRIBE stadium_revenue;
226 •  DESCRIBE media_revenue;
227 •  DESCRIBE game_revenue;
228 •  DESCRIBE player_stat;
229 •  DESCRIBE playoff_seed;
```

4.2 Advanced Features

One notable Stored Function was used for query purposes. This function is described more in detail in the following Section 4.3.1.

```
#QUERY1
CREATE FUNCTION PerformScore (PTS INT, AST INT, REB INT, TURN INT)
RETURNS DECIMAL(10,2)
DETERMINISTIC
RETURN PTS + (AST * 1.2) + (REB * 1.5) + (TURN * -0.5);
```

Triggers were also used throughout the database to prevent inadvertent updates to Revenue totals.

- A BEFORE UPDATE trigger was used, prompting the User with a message to ensure if the transaction is meant to occur.
- An AFTER UPDATE trigger was used, prompting the User with a confirmation message that an UPDATE transaction was made, and no further action is needed.
- A BEFORE DELETE trigger was used, prompting the User with a password prompt to ensure that the action is intentional and made by a valid employee.

4.3 Queries

Querying is and should be heavily used for operation and maintenance of this NBA Financial Database. Below are 8 query situations that are used often for general procedures of the User.

1. Revenue summation of a single entity
2. Revenue summation of a like revenue attribute across multiple entities
3. Revenue query to obtain totals across all entities
4. Querying a specific revenue entity for subsequent update of data
5. Attribute query for Players in a team
6. Another attribute query for player stats in multiple teams
7. Attribute query to obtain popularity scores of each player, sorted
8. Attribute query to find highest grossing game played for LA clippers

4.3.1 Obtain High-Performance Players

For this query, we obtain and sort the Player names (first, last, and suffix if needed) of those that exhibited the best performances across the 2024-2025 season. The performances, in this case, are measured by adding the total points, rebounds, and assists of a player, assigning each statistic a weight to factor into the overall score. This is useful for determining All-Star caliber Players for the Season, as well as which players are ideal marketing targets for the subsequent off-season. The weighted function outline and stored function in MySQL are shown below:

performance_score = Points + (Assists * 1.2) + (Rebounds * 1.5) + (Turnovers * -0.5)

```
#QUERY1
CREATE FUNCTION PerformScore (PTS INT, AST INT, REB INT, TURN INT)
RETURNS DECIMAL(10,2)
DETERMINISTIC
RETURN PTS + (AST * 1.2) + (REB * 1.5) + (TURN * -0.5);
```

And the Original data is found here:

PLAYER

Player_id	P_fname	P_lname	P_suffix	P_team	P_popscore
001	James	Lebron	Jr.	NULL	99.99
002	Lebron	James	NULL	Los Angeles Lakers	98.00
003	Stephen	Curry	NULL	Golden State Warriors	0.00
004	James	Harden	NULL	Los Angeles Clippers	0.00
005	Devin	Booker	NULL	Phoenix Suns	0.00
006	Jayson	Tatum	NULL	Boston Celtics	0.00
007	Giannis	Antetokounmpo	NULL	Milwaukee Bucks	0.00
008	Kyrie	Irving	NULL	Brooklyn Nets	0.00
009	Jimmy	Butler	III	Miami Heat	0.00
010	Nikola	Jokic	NULL	Denver Nuggets	0.00
023	Michael	Jordan	NULL	NULL	100.00
NULL	NULL	NULL	NULL	NULL	NULL

PLAYER_STAT

Player_id	Season_yr	Points	Assists	Rebounds	Turnovers	Field_goal_pct	Three_pt_pct	Plus_minus	Minutes_played	Games_played
003	2024-2025	1700	420	300	200	44.80	40.70	60	2240	70
004	2024-2025	1500	470	320	250	42.60	37.70	30	2010	60
009	2023-2024	1200	130	100	90	34.60	30.70	-23	1650	50
001	2024-2025	10	0	5	10	40.20	20.00	-30	90	3
005	2024-2025	1500	200	170	240	45.00	41.50	75	2345	72
006	2024-2025	2500	240	180	232	39.00	34.00	85	2424	80
007	2024-2025	2400	60	300	203	49.00	30.00	23	2001	72

The stored function was then called in this statement below, and the output was stored as performance_score. Then, JOIN was used to obtain a table of Names along with the score, and an ORDER BY was used to sort in ascending order:

```
SELECT P_fname, P_lname, P_suffix,
PerformScore(Points, Assists, Rebounds, Turnovers) AS performance_score
FROM player_stat
JOIN player ON player_stat.Player_id = player.Player_id
WHERE Season_yr = '2024-2025'
ORDER BY performance_score DESC;
```

FINAL_RESULT

P_fname	P_lname	P_suffix	performance_score
Jayson	Tatum	NULL	2942.00
Giannis	Antetokounmpo	NULL	2820.50
Stephen	Curry	NULL	2554.00
James	Harden	NULL	2419.00
Devin	Booker	NULL	1875.00
James	Lebron	Jr.	12.50

4.3.2 Obtain Highest Grossing Games

For this query, we are interested in obtaining the highest grossing games of the 2024-2025 season that was played by the most recent NBA Champion, the Oklahoma City Thunder. This can be used for scheduling and marketing purposes in the subsequent season, where the NBA can purposefully schedule highly acclaimed games on big Game instances like Opening Tip-Off or Christmas Day games.

GAME

Game_id	Date	Stadium_id	Home_team	Away_team	Home_fscore	Away_fscore
001	2025-11-12	9	Milwaukee Bucks	Boston Celtics	111	121
002	2025-11-12	6	Oklahoma City Thunder	Denver Nuggets	103	100
003	2025-11-13	1	Los Angeles Clippers	Los Angeles Lakers	120	98
004	2025-11-13	12	Chicago Bulls	Miami Heat	121	122
005	2025-11-15	6	Oklahoma City Thunder	Phoenix Suns	92	97
006	2025-11-15	9	Milwaukee Bucks	Philadelphia 76ers	132	125
007	2025-11-15	8	Boston Celtics	Los Angeles Clippers	113	115
008	2025-11-16	2	Los Angeles Lakers	Golden State Warriors	111	121
009	2025-11-18	15	Indiana Pacers	Oklahoma City Thunder	114	101
010	2025-11-21	3	Sacramento Kings	Oklahoma City Thunder	112	127
NULL	NULL	NULL	NULL	NULL	NULL	NULL

GAME_REVENUE

Game_id	Season_yr	G_ticket_rev	G_concession_rev	G_store_rev	G_sponsor_rev	G_media_rev
002	2024-2025	1800.00	3200.00	1700.00	1900.00	1300.00
003	2024-2025	5000.00	3000.00	3200.00	1700.00	1500.00
005	2024-2025	2600.00	2600.00	1600.00	1800.00	1200.00
006	2023-2024	4000.00	2000.00	2500.00	500.00	1900.00
009	2022-2023	3000.00	3400.00	2200.00	3000.00	1000.00
010	2024-2025	2000.00	3000.00	1400.00	900.00	900.00
NULL	NULL	NULL	NULL	NULL	NULL	NULL

The statement used is here, where we selected for the summation of all GAME Revenue attributes. We then used a JOIN to link the Home and Away teams together, with a WHERE clause to ensure that only games involving OKC Thunder in the most recent Season was obtained:

```
# QUERY 2
SELECT game.Home_team, game.Away_team, (G_ticket_rev + G_concession_rev + G_store_rev + G_sponsor_rev + G_media_rev) AS total
FROM game_revenue
JOIN game ON game_revenue.Game_id = game.Game_id
WHERE (Home_team = "Oklahoma City Thunder" OR Away_team = "Oklahoma City Thunder") AND (game_revenue.Season_yr = "2024-2025");
```

FINAL RESULT

Home_team	Away_team	total
Oklahoma City Thunder	Denver Nuggets	9900.00
Oklahoma City Thunder	Phoenix Suns	9800.00
Sacramento Kings	Oklahoma City Thunder	8200.00

4.3.3 Ranking Popularity Scores

This query involves obtaining and sorting all currently rostered Player Popularity Scores from ascending to descending order. This can easily be done by using SELECT on PLAYER.Player_popscore, WHERE P_team is NOT NULL:

PLAYER

Player_id	P_fname	P_lname	P_suffix	P_team	P_popscore
001	James	Lebron	Jr.	NULL	99.99
002	Lebron	James	NULL	Los Angeles Lakers	98.00
003	Stephen	Curry	NULL	Golden State Warriors	99.50
004	James	Harden	NULL	Los Angeles Clippers	98.70
005	Devin	Booker	NULL	Phoenix Suns	92.70
006	Jayson	Tatum	NULL	Boston Celtics	95.30
007	Giannis	Antetokounmpo	NULL	Milwaukee Bucks	96.50
008	Kyrie	Irving	NULL	Brooklyn Nets	90.10
009	Jimmy	Butler	III	Miami Heat	91.90
010	Nikola	Jokic	NULL	Denver Nuggets	98.00
023	Michael	Jordan	NULL	NULL	100.00
NULL	NULL	NULL	NULL	NULL	NULL

```
SELECT P_fname, P_lname, P_popscore
FROM player
WHERE P_team IS NOT NULL
ORDER BY P_popscore DESC;
```

FINAL_RESULT

P_fname	P_lname	P_popscore
Stephen	Curry	99.50
James	Harden	98.70
Nikola	Jokic	98.00
Lebron	James	98.00
Giannis	Antetokounmpo	96.50
Jayson	Tatum	95.30
Devin	Booker	92.70
Jimmy	Butler	91.90
Kyrie	Irving	90.10

4.3.4 Finding Best Overall Team

Now using the CONFERENCE and PLAYOFF_SEED entities, we would like to obtain trending data on the top overall performance teams from the last 3 years. This can be

mimicked by obtaining all previous PLAYOFF_SEED standings into separate query methods.

An Example Result is shown here for instances of Seed_one:

	Seed_one	seedcount1
▶	Boston Celtics	3
	Oklahoma City Thunder	1
	Denver Nuggets	1
	Los Angeles Clippers	1

The following statements are for each Seed in the table:

```
# QUERY 4
(SELECT Seed_one,
COUNT(*) AS seedcount1
FROM playoff_seed
WHERE Season_yr = "2022-2023" OR Season_yr = "2023-2024" OR Season_yr = "2024-2025'
GROUP BY Seed_one
ORDER BY seedcount1 DESC);

SELECT Seed_two,
COUNT(*) AS seedcount2
FROM playoff_seed
WHERE Season_yr = "2022-2023" OR Season_yr = "2023-2024" OR Season_yr = "2024-2025"
GROUP BY Seed_two
ORDER BY seedcount2 DESC ;

(SELECT Seed_three,
COUNT(*) AS seedcount3
FROM playoff_seed
WHERE Season_yr = "2022-2023" OR Season_yr = "2023-2024" OR Season_yr = "2024-2025"
GROUP BY Seed_three
ORDER BY seedcount3 DESC);
```

```
(SELECT Seed_four,
COUNT(*) AS seedcount4
FROM playoff_seed
WHERE Season_yr = "2022-2023" OR Season_yr = "2023-2024" OR Season_yr = "2024-2025"
GROUP BY Seed_four
ORDER BY seedcount4 DESC);

(SELECT Seed_five,
COUNT(*) AS seedcount5
FROM playoff_seed
WHERE Season_yr = "2022-2023" OR Season_yr = "2023-2024" OR Season_yr = "2024-2025"
GROUP BY Seed_five
ORDER BY seedcount5 DESC);

(SELECT Seed_six,
COUNT(*) AS seedcount6
FROM playoff_seed
WHERE Season_yr = "2022-2023" OR Season_yr = "2023-2024" OR Season_yr = "2024-2025"
GROUP BY Seed_six
ORDER BY seedcount6 DESC);

(SELECT Seed_seven,
COUNT(*) AS seedcount7
FROM playoff_seed
WHERE Season_yr = "2022-2023" OR Season_yr = "2023-2024" OR Season_yr = "2024-2025"
GROUP BY Seed_seven
ORDER BY seedcount7 DESC );
```

4.3.5 Summation of Revenue for an Entity

Now for a very common Revenue query, we would like to obtain the total amount of Revenue that was assessed for one Entity during a specific Season. For example, we can use the PLAYER_REVENUE entity and addition operations to output the total of its attributes:

PLAYER_REVENUE

Player_id	Season_yr	P_jersey_rev	P_apparel_rev	P_sponsor_rev	P_misc_rev
002	2024-2025	30000.00	50000.00	35000.00	30000.00
003	2024-2025	33000.00	40000.00	37000.00	35000.00
006	2024-2025	31000.50	30000.00	30000.00	29000.00
010	2024-2025	30000.00	44000.00	27000.00	25000.00
NULL	NULL	NULL	NULL	NULL	NULL

Statement:

```
SELECT player.P_fname, player.P_lname, player.P_suffix,
(P_jersey_rev +P_apparel_rev+ P_sponsor_rev+ P_misc_rev) AS P_total
FROM player_revenue
JOIN player ON player_revenue.Player_id = player.Player_id
WHERE P_team IS NOT NULL;
```

FINAL_RESULT

	P_fname	P_lname	P_suffix	P_total
▶	Lebron	James	NULL	145000.00
	Stephen	Curry	NULL	145000.00
	Jayson	Tatum	NULL	120000.50
	Nikola	Jokic	NULL	126000.00

4.3.6 Summation of a Revenue Type Across Entities

This query involves another summation of Revenue, this time for a specific Revenue type across entities that use it as a valid attribute. Keep in mind that this should only be performed for two mutually exclusive revenue attributes, or two attributes that do not share revenue into its total. An example of this is calculating sponsorship money collected across entities for a specific season; sponsorship revenue that goes to a Player are different revenue streams from sponsorships shown exclusively in Games, which are also different from sponsorship revenue that goes towards a Media Network. An example of something that is not mutually exclusive is ticket sales from Stadiums and Games, as money obtained from ticket sales in a Game may be the same money obtained from the Game's Stadium.

We built a statement based on a series of SUM() operations, which holds all the totals of each sponsorship revenue outlet, as well as several SELECT and UNION ALL clauses. Within each SELECT, the statement calls for the Season_yr of the sponsorship, as well as one of the three sponsorship revenue outlets. The UNION ALL clause helps join all of these SELECT statements together without deleting the unifying columns (Season_yr), which are needed throughout the query. Finally, the SUM() totals were grouped by Season_yr, giving the output

```sql
# QUERY 6
SELECT Season_yr, SUM(P_spons) AS P_total, SUM(G_spons) AS G_total, SUM(M_spons) AS M_total,
SUM(P_spons + G_spons + M_spons) AS ful_total
FROM(
    SELECT Season_yr, P_sponsor_rev AS P_spons, 0 AS G_spons, 0 AS M_spons
    FROM player_revenue

  UNION ALL

  SELECT Season_yr, 0 AS P_spons, G_sponsor_rev AS G_spons, 0 AS M_spons
  FROM game_revenue

  UNION ALL

  SELECT Season_yr, 0 AS P_spons, 0 AS G_spons, N_sponsor_rev AS M_spons
  FROM media_revenue
) AS combined
GROUP BY Season_yr
```

FINAL_RESULT

Season_yr	P_total	G_total	M_total	ful_total
2022-2023	15000.00	3000.00	80000.00	98000.00
2023-2024	15000.00	500.00	65000.00	80500.00
2024-2025	129000.00	6300.00	235000.00	370300.00

4.3.7 Preparing Revenue Query for Update

This simple but effective query is primarily used for Users with Financial roles to edit Revenue entities and attributes. If we wanted to update the TEAM_REVENUE attributes with new quantities obtained from and end of month calculation, we perform the following:

TEAM_REVENUE (Original Data)

Team_name	Season_yr	T_ticket_rev	T_apparel_rev	T_misc_rev
Boston Celtics	2024-2025	30000.00	40000.00	35000.00
Golden State Warriors	2024-2025	35000.00	30000.00	45000.00
Los Angeles Clippers	2023-2024	30000.00	20000.00	20000.00
Los Angeles Clippers	2024-2025	10000.00	10000.00	20000.00
Los Angeles Lakers	2024-2025	20000.00	30000.00	50000.00
New York Knicks	2024-2025	20000.00	20000.00	20000.00
NULL	NULL	NULL	NULL	NULL

First update statement initialized on the “Los Angeles Clippers” for Season “2024-2025.”

```
# QUERY 7
SELECT * FROM team_revenue;

UPDATE team_revenue
SET T_ticket_rev = T_ticket_rev + 5000.00
WHERE Team_name = "Los Angeles Clippers" AND Season_yr = "2024-2025";
```

TEAM_REVENUE (Clippers Ticket Revenue Updated)

Team_name	Season_yr	T_ticket_rev	T_apparel_rev	T_misc_rev
Boston Celtics	2024-2025	30000.00	40000.00	35000.00
Golden State Warriors	2024-2025	35000.00	30000.00	45000.00
Los Angeles Clippers	2023-2024	30000.00	20000.00	20000.00
Los Angeles Clippers	2024-2025	15000.00	10000.00	20000.00
Los Angeles Lakers	2024-2025	20000.00	30000.00	50000.00

We can even do a series of UPDATE clauses on the tables, which is generally recommended. This one is performed on the “Boston Celtics” for all attributes:

-
```
UPDATE team_revenue
SET T_ticket_rev = T_ticket_rev + 5000.00
WHERE Team_name = "Boston Celtics" AND Season_yr = "2024-2025";
```

-
```
UPDATE team_revenue
SET T_apparel_rev = T_apparel_rev + 7000.00
WHERE Team_name = "Boston Celtics" AND Season_yr = "2024-2025";
```

-
```
UPDATE team_revenue
SET T_misc_rev = T_misc_rev + 10000.00
WHERE Team_name = "Boston Celtics" AND Season_yr = "2024-2025";
```

TEAM_REVENUE (Celtics All-Updated)

Team_name	Season_yr	T_ticket_rev	T_apparel_rev	T_misc_rev
Boston Celtics	2024-2025	35000.00	47000.00	45000.00
Golden State Warriors	2024-2025	35000.00	30000.00	45000.00
Los Angeles Clippers	2023-2024	30000.00	20000.00	20000.00
Los Angeles Clippers	2024-2025	15000.00	10000.00	20000.00
Los Angeles Lakers	2024-2025	20000.00	30000.00	50000.00
New York Knicks	2024-2025	20000.00	20000.00	20000.00
NULL	NULL	NULL	NULL	NULL

4.3.8 Determining Best Revenue Attribute for an Entity

This query involves gathering Revenue data for a single entity across several seasons, then adding the data from each season together. We can do an example of this on the PLAYER_REVENUE entity.

The statement below involves another series of SUM() statements, each enclosing one of the four PLAYER_REVENUE attributes. Similarly to a previous example, the rest of the statement is a series of SELECT and UNION ALL clauses that each store a total amount of revenue for an attribute:

```
SELECT 'P_jersey_rev' AS revenue_type, SUM(P_jersey_rev) AS total FROM player_revenue
UNION ALL
SELECT 'P_apparel_rev', SUM(P_apparel_rev) FROM player_revenue
UNION ALL
SELECT 'P_sponsor_rev', SUM(P_sponsor_rev) FROM player_revenue
UNION ALL
SELECT 'P_misc_rev', SUM(P_misc_rev) FROM player_revenue
ORDER BY total DESC;
```

FINAL_RESULT

revenue_type	total
P_apparel_rev	231000.00
P_jersey_rev	166000.50
P_misc_rev	164000.00
P_sponsor_rev	159000.00

5. CRUD Matrix

This section demonstrates the construction of the CRUD Matrix for the NBA Financial Database.

5.1 List of Entity Types

E1: season
E2: conference
E3: stadium
E4: team
E5: player
E6: game
E7: head_coach
E8: owner
E9: media
E10: player_revenue
E11: team_revenue
E12: stadium_revenue
E13: media_revenue
E14: game_revenue
E15: player_stat
E16: playoff_seed

5.2 List of Functions

F1: Obtaining High Performance Players
- Read PLAYER_STAT
- Read PLAYER
- Read SEASON

F2: Obtain Highest Grossing Games
- Read GAME
- Read GAME_REVENUE
- Read SEASON

F3: Ranking Popularity Scores
- Read PLAYER
- Update PLAYER
- Read HEAD_COACH
- Update HEAD_COACH
- Read OWNER
- Update OWNER

F4: Finding Best Overall Team
- Read CONFERENCE
- Read PLAYOFF_SEED
- Read TEAM
- Read SEASON
- Update PLAYOFF_SEED

F5: Summation of Revenue for Entity

- Read REVENUE Entities
- Read SEASON
- Read NON_REVENUE Entities

F6: Summation Across Entities
- Read REVENUE Entities
- Read SEASON
- Read NON_REVENUE Entities

F7: Preparing and Updating Query
- Read ALL entities
- Update REVENUE Entities
- Insert REVENUE Entity Instances if needed

F8: Determine Best Revenue Attribute
- Read REVENUE Entities
- Read SEASON
- Read NON_REVENUE Entities

CRUD MATRIX:

	F1	F2	F3	F4	F5	F6	F7	F8
E1	R	R		R	R	R		R
E2				R			R	
E3					R	R	R	R
E4				R	R	R	R	R
E5	R		R,U		R	R	R	R
E6		R			R	R	R	R
E7			R,U				R	
E8			R,U				R	
E9					R	R	R	R
E10					R	R	C,R,U	R
E11					R	R	C,R,U	R
E12					R	R	C,R,U	R
E13					R	R	C,R,U	R
E14		R			R	R	C,R,U	R
E15	R						R	
E16				R,U			R	

6. Concluding Remarks

This Database course was, in fact, my very first introduction into Databases and the SQL Language, and overall I enjoyed learning so much about it throughout the course of the short summer semester. Prior to this course, I always heard of SQL as a highly used platform, especially when it came to seeing it on many job postings and research laboratory postings. I was always intimidated to learn about it though, which is why I

can’t thank this class enough for breaking down those barriers and allowing me to learn enough to generate a database of my own.

This project overall was a great experience, and although I do wish I had a bit more time to be able to implement more constraints and triggers to the Database, I do feel inspired to continue working on this even after this class, just for my own practice. One of the biggest lessons I learned in this project is to really force yourself to strictly meet the recommended Module checkpoints; I do admit towards the second half of this semester, I did tend to lag on completion of these deadlines, and it definitely took a toll on me. Regardless, I believe I did a substantial job in turning this into a strength in terms of learning to time manage better, as well as deeply learn and understand the design and implementation of the Database and why certain aspects work, while others do not.

I think a strength of the project itself is the flexibility that the chosen model has for Revenue tracking. I do believe that tracking the Revenue by Season and keeping the Revenue attributes into separate entities made the overall design and implementation process much less complicated. Additionally, once I got more acquainted with how to add Check constraints, FK constraints, and get more acquainted with MySQL in general, it also made the project easier.

A weakness of my project is the lack of certain constraints that may have been left unchecked. For instance, very late into the project, I discovered that the GAME_REVENUE entity inadvertently generated multiple tuples of the same Game_id with different seasons. I then realized that my definition of the Game_id Primary Key for the GAME Entity was not unique; for instance, I assumed that Game_id “009” would correspond to single OKC-IND game that occurred in 2024. However, if I called GAME_REVENUE for the Game_id “009”, but for 2023, then this would generate revenue from the exact same OKC-IND Game_id “009”, but as if it took place in a different year. Unless Revenue data and an OKC-IND game occurrence somehow were all the same on that same exact day in 2023, this creates a constraint violation. I hopefully fixed this issue from further specifications, and I also ensured that this does not apply to other strong entities as well.

Lastly, the biggest factor that I wish I would have implemented with more time is the addition of a running total attribute into the database. This would have saved much time and script length in querying data for total revenue calculations. However after several attempts at trying to get an efficient running total calculator working, no solution ended up making the final cut. It will, however, be a goal of mine to perform soon as I continue the design of this Database!

Appendices

Appendix A - DDL, INSERT, SELECT Statements

For DDL Statements, please see Attached File "NBA DATABASE – DDL Statements."

The SELECT statement used to display each entity is here:

```
1  • SELECT * FROM season;
2  • SELECT * FROM conference;
3  • SELECT * FROM stadium;
4  • SELECT * FROM team;
5  • SELECT * FROM player;
6  • SELECT * FROM game;
7  • SELECT * FROM head_coach;
8  • SELECT * FROM owner;
9  • SELECT * FROM media;
10 • SELECT * FROM player_revenue;
11 • SELECT * FROM team_revenue;
12 • SELECT * FROM stadium_revenue;
13 • SELECT * FROM media_revenue;
14 • SELECT * FROM game_revenue;
15 • SELECT * FROM player_stat;
16 • SELECT * FROM playoff_seed;
```

Some examples of INSERT statements are shown here:

```
INSERT INTO player (Player_id, P_fname, P_lname, P_team)
VALUES
("003", "Stephen", "Curry", "Golden State Warriors"),
("004", "James", "Harden", "Los Angeles Clippers"),
("005", "Devin", "Booker", "Phoenix Suns"),
("006", "Jayson", "Tatum", "Boston Celtics"),
("007", "Giannis", "Antetokounmpo", "Milwaukee Bucks"),
("008", "Kyrie", "Irving", "Brooklyn Nets"),
("009", "Jimmy", "Butler", "Miami Heat"),
("010", "Nikola", "Jokic", "Denver Nuggets");
```

```
INSERT INTO game (Game_id, Date, Stadium_id, Home_team, Away_team, Home_fscore, Away_fscore)
VALUES ("010", "2025-11-21", 3, "Sacramento Kings", "Oklahoma City Thunder", 112, 127),
("009", "2025-11-18", 15, "Indiana Pacers", "Oklahoma City Thunder", 114, 101),
("001", "2025-11-12", 9, "Milwaukee Bucks", "Boston Celtics", 111, 121),
("002", "2025-11-12", 6, "Oklahoma City Thunder", "Denver Nuggets", 103, 100),
("003", "2025-11-13", 1, "Los Angeles Clippers", "Los Angeles Lakers", 120, 98),
("004", "2025-11-13", 12, "Chicago Bulls", "Miami Heat", 121, 122),
("005", "2025-11-15", 6, "Oklahoma City Thunder", "Phoenix Suns", 92, 97),
("006", "2025-11-15", 9, "Milwaukee Bucks", "Philadelphia 76ers", 132, 125),
("007", "2025-11-15", 8, "Boston Celtics", "Los Angeles Clippers", 113, 115),
("008", "2025-11-16", 2, "Los Angeles Lakers", "Golden State Warriors", 111, 121);
```

```
INSERT INTO team_revenue (Team_name, Season_yr, T_ticket_rev, T_apparel_rev, T_misc_rev)
VALUES
("Los Angeles Clippers", "2023-2024", 30000.00, 20000.00, 20000.00),
("Los Angeles Lakers", "2024-2025", 20000.00, 30000.00, 50000.00),
("Golden State Warriors", "2024-2025", 35000.00, 30000.00, 45000.00),
("Boston Celtics", "2024-2025", 30000.00, 40000.00, 35000.00),
("New York Knicks", "2024-2025", 20000.00, 20000.00, 20000.00);

INSERT INTO player_revenue (Player_id, Season_yr, P_jersey_rev, P_apparel_rev, P_sponsor_rev, P_misc_rev)
VALUES
("003", "2024-2025", 33000.00, 40000.00, 37000.00, 35000),
("010", "2024-2025", 30000.00, 44000.00, 27000.00, 25000),
("006", "2024-2025", 31000.5, 30000.00, 30000.00, 29000);

INSERT INTO game_revenue (Game_id, Season_yr, G_ticket_rev, G_concession_rev, G_store_rev, G_sponsor_rev, G_media_rev)
VALUE("002", "2024-2025", 3000, 2200, 2300, 1500, 1700),
("003", "2024-2025", 5000, 3000, 3200, 1700, 1500),
```

Appendix B - Data Dictionary Index

Please see Attachment titled “NBA DATABASE – Data Dictionary."

References

- Elmasri, R. and Navathe, S.B. (2016) Fundamentals of Database Systems. 7th Edition, - Addison-Wesley, Boston.
- MySQL 8.0 Reference Manual