**The Hoops Academy**

**Basketball League Database**

**Table Of Contents**

Part One..........................................................................................................................................3

Summary............................................................................................................................3

Stakeholders......................................................................................................................3

Business Rules....................................................................................................................3

Data Questions...................................................................................................................4

Conceptual Model..............................................................................................................4

Logical Model.....................................................................................................................5

Part Two..........................................................................................................................................3

Data Definition Language – Creating Tables and Constraints............................................6

Data Manipulation Language……….……….……………………………............................................8

Adding Data Using INSERT Statements…….……………..............................................8

Querying Data Using SELECT Statements…….…………….........................................11

Programming Objects………………….……….……………………………..........................................16

User Interface………….………………….……….……………………………..........................................19

Reflection………………..………………….……….……………………………..........................................22

**Part One**

**Summary:**

The Hoops Academy is hosting a youth basketball league this year. The academy would like to manage and track basic team and player statistics this season for the use of analysis for coaches. Additionally, individual awards will be presented to players from each league at the end of the season based on statistical performance. To assist with statistical team and player tracking for each game, the academy would like a database to be built.

**Stakeholders:**

* Company management
  + People who are in management positions at the Hoops Academy
* Coaches
  + Coaches of each basketball team

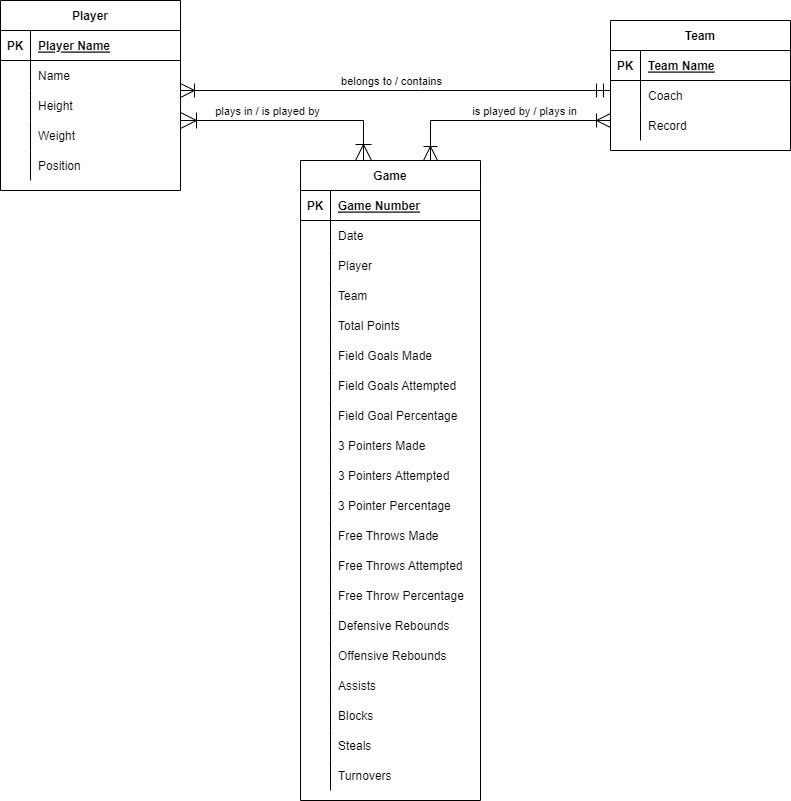
**Business Rules:**

* The league consists of 10 teams.
* The season will last 9 weeks.
* Each team will play one game every week against one other team from the league.
* Each team consists of 10 players.
* Each team may have two coaches – a head coach and an assistant coach. Each coach can only coach one team.
* A player must be assigned one of five positions – point guard, shooting guard, small forward, power forward, and center.
* Basic basketball statistics will be tracked and recorded each game for both teams and players.

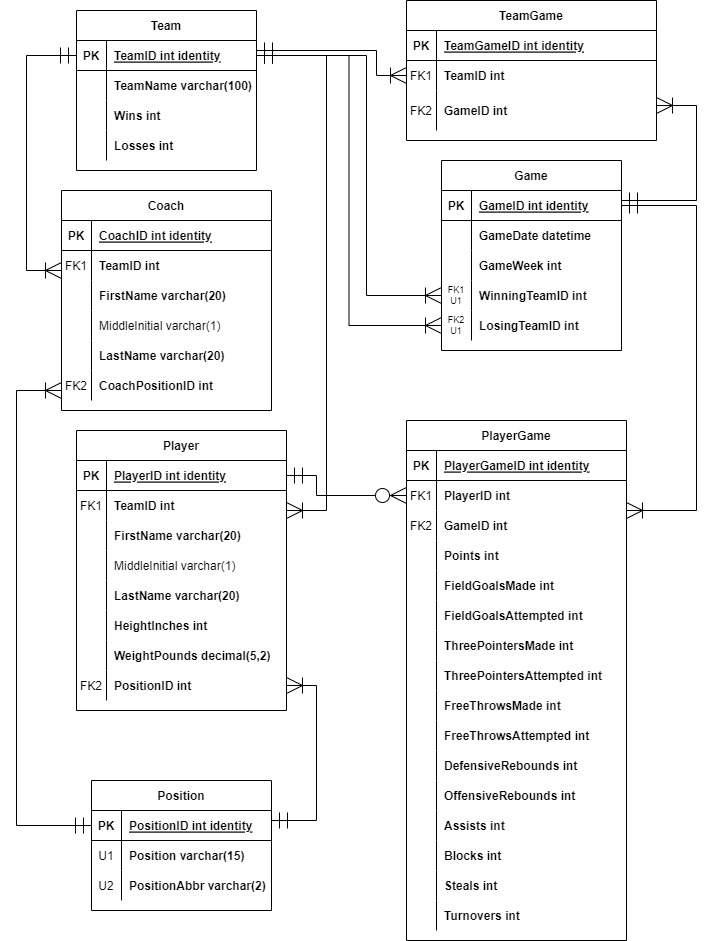
**Data Questions:**

* Which players and coaches belong to each team?
* What are the physical attributes of each player?
* What was the outcome of every game, and which teams were involved?
* Which players performed the best on average across the major statistical categories (points, field goal percentage, 3 point percentage, free throw percentage, assists, blocks, steals, and so forth)?
* Which teams performed the best on average across the major statistical categories?
* Which teams performed the best according to their wins and losses?

**Conceptual Model:**



**Logical Model:**



**Part Two**

**Data Definition Language**

**Creating Tables and Constraints**

IF EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'PlayerGame')

BEGIN

DROP TABLE PlayerGame

END

GO

IF EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'TeamGame')

BEGIN

DROP TABLE TeamGame

END

GO

IF EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'Game')

BEGIN

DROP TABLE Game

END

GO

IF EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'Coach')

BEGIN

DROP TABLE Coach

END

GO

IF EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'Player')

BEGIN

DROP TABLE Player

END

GO

IF EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'Team')

BEGIN

DROP TABLE Team

END

GO

IF EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'Position')

BEGIN

DROP TABLE Position

END

GO

-- Creating the Team table

CREATE TABLE Team (

TeamID int identity primary key,

TeamName varchar(100) not null,

Wins int not null,

Losses int not null

)

GO

--Creating the Position table

CREATE TABLE Position (

PositionID int identity primary key,

Position varchar(15) unique not null,

PositionAbbr varchar(2) unique not null)

GO

--Creating the Player table

CREATE TABLE Player (

PlayerID int identity primary key,

TeamID int FOREIGN KEY REFERENCES Team(TeamID) not null,

FirstName varchar(20) not null,

MiddleInitial varchar(1),

LastName varchar(20) not null,

HeightInches int not null,

WeightPounds decimal(5,2) not null,

PositionID int FOREIGN KEY REFERENCES Position(PositionID) not null

)

GO

--Creating the Coach table

CREATE TABLE Coach (

CoachID int identity primary key,

TeamID int FOREIGN KEY REFERENCES Team(TeamID) not null,

FirstName varchar(20) not null,

MiddleInitial varchar(1),

LastName varchar(20) not null,

CoachPositionID int FOREIGN KEY REFERENCES Position(PositionID) not null

)

GO

--Creating the Game table

CREATE TABLE Game (

GameID int identity primary key,

GameDate datetime not null,

GameWeek int not null,

WinningTeamID int FOREIGN KEY REFERENCES Team(TeamID) not null,

LosingTeamID int FOREIGN KEY REFERENCES Team(TeamID) not null

)

GO

--Creating the TeamGame table

CREATE TABLE TeamGame (

TeamGameID int identity primary key,

TeamID int FOREIGN KEY REFERENCES Team(TeamID) not null,

GameID int FOREIGN KEY REFERENCES Game(GameID) not null

)

GO

--Creating the PlayerGame table

CREATE TABLE PlayerGame (

PlayerGameID int identity primary key,

PlayerID int FOREIGN KEY REFERENCES Player(PlayerID) not null,

GameID int FOREIGN KEY REFERENCES Game(GameID) not null,

Points int not null,

FieldGoalsMade int not null,

FieldGoalsAttempted int not null,

ThreePointersMade int not null,

ThreePointersAttempted int not null,

FreeThrowsMade int not null,

FreeThrowsAttempted int not null,

DefensiveRebounds int not null,

OffensiveRebounds int not null,

Assists int not null,

Blocks int not null,

Steals int not null,

Turnovers int not null

)

GO

**Adding Data Using INSERT Statements**

--Populate Position table

INSERT INTO Position (Position, PositionAbbr) VALUES ('Point Guard', 'PG')

INSERT INTO Position (Position, PositionAbbr) VALUES ('Shooting Guard', 'SG')

INSERT INTO Position (Position, PositionAbbr) VALUES ('Small Forward', 'SF')

INSERT INTO Position (Position, PositionAbbr) VALUES ('Power Forward', 'PF')

INSERT INTO Position (Position, PositionAbbr) VALUES ('Center', 'C')

INSERT INTO Position (Position, PositionAbbr) VALUES ('Head Coach', 'HC')

INSERT INTO Position (Position, PositionAbbr) VALUES ('Assistant Coach', 'AC')

--Populate Team table

INSERT INTO Team (TeamName, Wins, Losses) VALUES ('That One Team',4,5)

INSERT INTO Team (TeamName, Wins, Losses) VALUES ('Ballers',7,2)

INSERT INTO Team (TeamName, Wins, Losses) VALUES ('Rudy Tomjanoviches',6,3)

INSERT INTO Team (TeamName, Wins, Losses) VALUES ('Floppers',3,6)

INSERT INTO Team (TeamName, Wins, Losses) VALUES ('Hand Check',4,5)

INSERT INTO Team (TeamName, Wins, Losses) VALUES ('Mediocre Team',5,4)

INSERT INTO Team (TeamName, Wins, Losses) VALUES ('Flamingos',4,5)

INSERT INTO Team (TeamName, Wins, Losses) VALUES ('Old Dudes',2,7)

INSERT INTO Team (TeamName, Wins, Losses) VALUES ('Young Bucks',4,5)

INSERT INTO Team (TeamName, Wins, Losses) VALUES ('Buzzer Beaters',6,3)

--Populate Player table (Only including first 10 inserts here)

INSERT INTO Player (TeamID, FirstName, MiddleInitial, LastName, HeightInches, WeightPounds, PositionID) VALUES (1, 'Neil', 'K', 'Skune', 68, 250, 1)

INSERT INTO Player (TeamID, FirstName, MiddleInitial, LastName, HeightInches, WeightPounds, PositionID) VALUES (1, 'Corny', 'O', 'Duckworth', 83, 179, 5)

INSERT INTO Player (TeamID, FirstName, MiddleInitial, LastName, HeightInches, WeightPounds, PositionID) VALUES (1, 'Aubert', 'O', 'Griffithe', 73, 210, 2)

INSERT INTO Player (TeamID, FirstName, MiddleInitial, LastName, HeightInches, WeightPounds, PositionID) VALUES (1, 'Kerk', 'F', 'Brandli', 81, 177, 5)

INSERT INTO Player (TeamID, FirstName, MiddleInitial, LastName, HeightInches, WeightPounds, PositionID) VALUES (1, 'Torrey', 'B', 'Adelberg', 80, 246, 4)

INSERT INTO Player (TeamID, FirstName, MiddleInitial, LastName, HeightInches, WeightPounds, PositionID) VALUES (1, 'Cleve', 'T', 'Goady', 71, 237, 2)

INSERT INTO Player (TeamID, FirstName, MiddleInitial, LastName, HeightInches, WeightPounds, PositionID) VALUES (1, 'Rutger', 'Y', 'Tossell', 74, 225, 3)

INSERT INTO Player (TeamID, FirstName, MiddleInitial, LastName, HeightInches, WeightPounds, PositionID) VALUES (1, 'Obed', 'T', 'Bownass', 78, 228, 4)

INSERT INTO Player (TeamID, FirstName, MiddleInitial, LastName, HeightInches, WeightPounds, PositionID) VALUES (1, 'Armstrong', 'B', 'Simioni', 69, 217, 1)

INSERT INTO Player (TeamID, FirstName, MiddleInitial, LastName, HeightInches, WeightPounds, PositionID) VALUES (1, 'Kev', 'S', 'Speller', 75, 221, 3)

--Populate Coach table

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (1,'Derril','J','Reston',6)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (2,'Basil','B','Shaughnessy',6)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (3,'Thacher','E','Antliff',6)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (4,'Daniela','K','Burdge',6)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (5,'Ritchie','R','Fife',6)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (6,'Jemmie','Y','Glenwright',6)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (7,'Bridgette','I','Yearnes',6)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (8,'Clarice','C','Furlong',6)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (9,'Corny','D','Kolin',6)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (10,'Evvy','B','Minors',6)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (1,'Carolann','D','Bloodworth',7)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (2,'Ludovico','V','Tunno',7)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (3,'Cordy','R','Chaudret',7)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (4,'Lidia','T','Fincke',7)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (5,'Bathsheba','H','Gentery',7)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (6,'Silvie','L','Catling',7)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (7,'Lucky','K','Sansome',7)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (8,'Alta','M','Mityushin',7)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (9,'Carlene','S','Smallcomb',7)

INSERT INTO Coach (TeamID, FirstName, MiddleInitial, LastName, CoachPositionID) VALUES (10,'Giacinta','H','Duigan',7)

--Populate Game table (Only including first 10 inserts here)

INSERT INTO Game (GameDate, GameWeek, WinningTeamID, LosingTeamID) VALUES ('5/5/2022',1,1,10)

INSERT INTO Game (GameDate, GameWeek, WinningTeamID, LosingTeamID) VALUES ('5/5/2022',1,2,9)

INSERT INTO Game (GameDate, GameWeek, WinningTeamID, LosingTeamID) VALUES ('5/5/2022',1,8,3)

INSERT INTO Game (GameDate, GameWeek, WinningTeamID, LosingTeamID) VALUES ('5/5/2022',1,7,4)

INSERT INTO Game (GameDate, GameWeek, WinningTeamID, LosingTeamID) VALUES ('5/5/2022',1,5,6)

INSERT INTO Game (GameDate, GameWeek, WinningTeamID, LosingTeamID) VALUES ('5/12/2022',2,2,7)

INSERT INTO Game (GameDate, GameWeek, WinningTeamID, LosingTeamID) VALUES ('5/12/2022',2,3,6)

INSERT INTO Game (GameDate, GameWeek, WinningTeamID, LosingTeamID) VALUES ('5/12/2022',2,4,5)

INSERT INTO Game (GameDate, GameWeek, WinningTeamID, LosingTeamID) VALUES ('5/12/2022',2,10,9)

INSERT INTO Game (GameDate, GameWeek, WinningTeamID, LosingTeamID) VALUES ('5/12/2022',2,8,1)

--Populate TeamGame table (Only including first 10 inserts here)

INSERT INTO TeamGame (TeamID, GameID) VALUES(1,1)

INSERT INTO TeamGame (TeamID, GameID) VALUES(10,1)

INSERT INTO TeamGame (TeamID, GameID) VALUES(2,2)

INSERT INTO TeamGame (TeamID, GameID) VALUES(9,2)

INSERT INTO TeamGame (TeamID, GameID) VALUES(3,3)

INSERT INTO TeamGame (TeamID, GameID) VALUES(8,3)

INSERT INTO TeamGame (TeamID, GameID) VALUES(4,4)

INSERT INTO TeamGame (TeamID, GameID) VALUES(7,4)

INSERT INTO TeamGame (TeamID, GameID) VALUES(5,5)

INSERT INTO TeamGame (TeamID, GameID) VALUES(6,5)

--Populate PlayerGame table (Only including first 10 inserts here)

INSERT INTO PlayerGame (PlayerID, GameID, Points, FieldGoalsMade, FieldGoalsAttempted, ThreePointersMade, ThreePointersAttempted, FreeThrowsMade, FreeThrowsAttempted, DefensiveRebounds, OffensiveRebounds, Assists, Blocks, Steals, Turnovers) VALUES (1,12,5,1,6,1,5,1,1,6,5,3,1,2,4)

INSERT INTO PlayerGame (PlayerID, GameID, Points, FieldGoalsMade, FieldGoalsAttempted, ThreePointersMade, ThreePointersAttempted, FreeThrowsMade, FreeThrowsAttempted, DefensiveRebounds, OffensiveRebounds, Assists, Blocks, Steals, Turnovers) VALUES (1,23,3,0,5,0,4,3,4,0,2,6,1,1,2)

INSERT INTO PlayerGame (PlayerID, GameID, Points, FieldGoalsMade, FieldGoalsAttempted, ThreePointersMade, ThreePointersAttempted, FreeThrowsMade, FreeThrowsAttempted, DefensiveRebounds, OffensiveRebounds, Assists, Blocks, Steals, Turnovers) VALUES (1,38,17,6,11,1,9,4,5,4,3,0,1,1,1)

INSERT INTO PlayerGame (PlayerID, GameID, Points, FieldGoalsMade, FieldGoalsAttempted, ThreePointersMade, ThreePointersAttempted, FreeThrowsMade, FreeThrowsAttempted, DefensiveRebounds, OffensiveRebounds, Assists, Blocks, Steals, Turnovers) VALUES (1,44,7,2,7,1,3,2,2,6,5,6,1,0,1)

INSERT INTO PlayerGame (PlayerID, GameID, Points, FieldGoalsMade, FieldGoalsAttempted, ThreePointersMade, ThreePointersAttempted, FreeThrowsMade, FreeThrowsAttempted, DefensiveRebounds, OffensiveRebounds, Assists, Blocks, Steals, Turnovers) VALUES (1,1,6,2,7,1,4,1,1,5,3,3,0,2,4)

INSERT INTO PlayerGame (PlayerID, GameID, Points, FieldGoalsMade, FieldGoalsAttempted, ThreePointersMade, ThreePointersAttempted, FreeThrowsMade, FreeThrowsAttempted, DefensiveRebounds, OffensiveRebounds, Assists, Blocks, Steals, Turnovers) VALUES (1,10,11,3,8,3,4,2,2,2,5,5,2,2,2)

INSERT INTO PlayerGame (PlayerID, GameID, Points, FieldGoalsMade, FieldGoalsAttempted, ThreePointersMade, ThreePointersAttempted, FreeThrowsMade, FreeThrowsAttempted, DefensiveRebounds, OffensiveRebounds, Assists, Blocks, Steals, Turnovers) VALUES (1,19,10,3,8,3,8,1,1,6,4,0,2,3,1)

INSERT INTO PlayerGame (PlayerID, GameID, Points, FieldGoalsMade, FieldGoalsAttempted, ThreePointersMade, ThreePointersAttempted, FreeThrowsMade, FreeThrowsAttempted, DefensiveRebounds, OffensiveRebounds, Assists, Blocks, Steals, Turnovers) VALUES (1,30,15,5,10,2,7,3,4,1,0,2,3,0,3)

INSERT INTO PlayerGame (PlayerID, GameID, Points, FieldGoalsMade, FieldGoalsAttempted, ThreePointersMade, ThreePointersAttempted, FreeThrowsMade, FreeThrowsAttempted, DefensiveRebounds, OffensiveRebounds, Assists, Blocks, Steals, Turnovers) VALUES (1,31,4,0,5,1,3,1,1,9,3,5,1,2,4)

INSERT INTO PlayerGame (PlayerID, GameID, Points, FieldGoalsMade, FieldGoalsAttempted, ThreePointersMade, ThreePointersAttempted, FreeThrowsMade, FreeThrowsAttempted, DefensiveRebounds, OffensiveRebounds, Assists, Blocks, Steals, Turnovers) VALUES (2,12,7,2,7,2,7,1,1,3,1,2,0,1,3)

**Querying Data Using SELECT Statements**

--Data Question 1: Which coaches and players belong to each team, and what are the physical attributes of each player?

CREATE OR ALTER VIEW vwTeamRoster

AS

SELECT

Team.TeamName,

Player.FirstName + ' ' + Player.LastName AS Player,

Position.Position,

Player.HeightInches,

Player.WeightPounds

FROM Player

JOIN Team ON Player.TeamID = Team.TeamID

JOIN Position ON Player.PositionID = Position.PositionID

UNION

SELECT

Team.TeamName,

Coach.FirstName + ' ' + Coach.LastName AS Player,

Position.Position,

NULL AS HeightInches,

NULL AS WeightPounds

FROM Coach

JOIN Team ON Coach.TeamID = Team.TeamID

JOIN Position ON Coach.CoachPositionID = Position.PositionID

GO

SELECT

\*

FROM vwTeamRoster

ORDER BY TeamName, Position

GO

**Results – First 12 Rows:**

Table

Description automatically generated

--Data question 2: What was the outcome of every game, and which teams were involved?

CREATE OR ALTER VIEW vwGameOutcome

AS

WITH TeamPoints AS (

SELECT

Player.TeamID,

PlayerGame.GameID,

SUM(PlayerGame.Points) AS Points

FROM PlayerGame

JOIN Player ON PlayerGame.PlayerID = Player.PlayerID

GROUP BY

Player.TeamID,

PlayerGame.GameID)

SELECT

Game.GameDate,

Game.GameWeek,

WinningTeam.TeamName AS WinningTeam,

LosingTeam.TeamName AS LosingTeam,

WinningPoints.Points AS WinningTeamPoints,

LosingPoints.Points AS LosingTeamPoints

FROM Game

JOIN Team AS WinningTeam ON Game.WinningTeamID = WinningTeam.TeamID

JOIN Team AS LosingTeam ON Game.LosingTeamID = LosingTeam.TeamID

JOIN TeamPoints AS WinningPoints ON Game.WinningTeamID = WinningPoints.TeamID

AND Game.GameID = WinningPoints.GameID

JOIN TeamPoints AS LosingPoints ON Game.LosingTeamID = LosingPoints.TeamID

AND Game.GameID = LosingPoints.GameID

GO

SELECT \*

FROM vwGameOutcome

GO

**Results – First 12 Rows:**

Graphical user interface, application

Description automatically generated

--Data Question 3: Which players performed the best on average across the major statistical categories?

CREATE OR ALTER VIEW vwPlayerAvgRanks

AS

WITH PlayerAvg AS (

SELECT

Player.FirstName + ' ' + Player.LastName AS Player,

ROUND(AVG(CONVERT(float, Points)), 2) AS PPG,

SUM(FieldGoalsMade) AS FGM,

SUM(FieldGoalsAttempted) AS FGA,

ROUND(SUM(CONVERT(float, FieldGoalsMade)) / SUM(CONVERT(float, FieldGoalsAttempted)), 2) AS [FG%],

SUM(ThreePointersMade) AS [3PM],

SUM(ThreePointersAttempted) AS [3PA],

ROUND(SUM(CONVERT(float, ThreePointersMade)) / SUM(CONVERT(float, ThreePointersAttempted)), 2) AS [3P%],

SUM(FreeThrowsMade) AS FTM,

SUM(FreeThrowsAttempted) AS FTA,

ROUND(SUM(CONVERT(float, FreeThrowsMade)) / SUM(CONVERT(float, FreeThrowsAttempted)), 2) AS [FT%],

ROUND(AVG(CONVERT(float, DefensiveRebounds)), 2) + ROUND(AVG(CONVERT(float, OffensiveRebounds)), 2) AS RPG,

ROUND(AVG(CONVERT(float, DefensiveRebounds)), 2) AS DRPG,

ROUND(AVG(CONVERT(float, OffensiveRebounds)), 2) AS ORPG,

ROUND(AVG(CONVERT(float, Assists)), 2) AS APG,

ROUND(AVG(CONVERT(float, Blocks)), 2) AS BPG,

ROUND(AVG(CONVERT(float, Steals)), 2) AS SPG,

ROUND(AVG(CONVERT(float, Turnovers)), 2) AS TOPG

FROM PlayerGame

JOIN Player ON PlayerGame.PlayerID = Player.PlayerID

GROUP BY Player.FirstName + ' ' + Player.LastName)

SELECT

Player,

PPG,

RANK() OVER(ORDER BY PPG DESC) AS PPGRank,

FGM,

RANK() OVER(ORDER BY FGM DESC) AS FGMRank,

FGA,

RANK() OVER(ORDER BY FGA DESC) AS FGARank,

[FG%],

RANK() OVER(ORDER BY [FG%] DESC) AS [FG%Rank],

[3PM],

RANK() OVER(ORDER BY [3PM] DESC) AS [3PMRank],

[3PA],

RANK() OVER(ORDER BY [3PA] DESC) AS [3PARank],

[3P%],

RANK() OVER(ORDER BY [3P%] DESC) AS [3P%Rank],

FTM,

RANK() OVER(ORDER BY FTM DESC) AS FTMRank,

FTA,

RANK() OVER(ORDER BY FTA DESC) AS FTARank,

[FT%],

RANK() OVER(ORDER BY [FT%] DESC) AS [FT%Rank],

RPG,

RANK() OVER(ORDER BY RPG DESC) AS RPGRank,

DRPG,

RANK() OVER(ORDER BY DRPG DESC) AS DRPGRank,

ORPG,

RANK() OVER(ORDER BY ORPG DESC) AS ORPGRank,

APG,

RANK() OVER(ORDER BY APG DESC) AS APGRank,

BPG,

RANK() OVER(ORDER BY BPG DESC) AS BPGRank,

SPG,

RANK() OVER(ORDER BY SPG DESC) AS SPGRank,

TOPG,

RANK() OVER(ORDER BY TOPG) AS TOPGRank

FROM PlayerAvg

GO

SELECT \*

FROM vwPlayerAvgRanks

GO

**Results – First 10 Rows:**

A screenshot of a computer

Description automatically generated with medium confidence

--Data Question 4: Which teams performed the best according to their wins and across the major statistical categories?

CREATE OR ALTER VIEW vwTeamAvgRanks

AS

WITH TeamTotals AS (

SELECT

Team.TeamName,

PlayerGame.GameID,

SUM(Points) AS Points,

SUM(FieldGoalsMade) AS FGM,

SUM(FieldGoalsAttempted) AS FGA,

SUM(ThreePointersMade) AS [3PM],

SUM(ThreePointersAttempted) AS [3PA],

SUM(FreeThrowsMade) AS FTM,

SUM(FreeThrowsAttempted) AS FTA,

SUM(DefensiveRebounds) + SUM(OffensiveRebounds) AS Rebounds,

SUM(DefensiveRebounds) AS DefensiveRebounds,

SUM(OffensiveRebounds) AS OffensiveRebounds,

SUM(Assists) AS Assists,

SUM(Blocks) AS Blocks,

SUM(Steals) AS Steals,

SUM(Turnovers) AS Turnovers

FROM PlayerGame

JOIN Player ON PlayerGame.PlayerID = Player.PlayerID

JOIN Team ON Player.TeamID = Team.TeamID

GROUP BY

Team.TeamName,

PlayerGame.GameID),

TeamAvg AS (

SELECT

TeamName,

ROUND(AVG(CONVERT(float, Points)), 2) AS PPG,

SUM(FGM) AS FGM,

SUM(FGA) AS FGA,

ROUND(SUM(CONVERT(float, FGM)) / SUM(CONVERT(float, FGA)), 2) AS [FG%],

SUM([3PM]) AS [3PM],

SUM([3PA]) AS [3PA],

ROUND(SUM(CONVERT(float, [3PM])) / SUM(CONVERT(float, [3PA])), 2) AS [3P%],

SUM(FTM) AS FTM,

SUM(FTA) AS FTA,

ROUND(SUM(CONVERT(float, FTM)) / SUM(CONVERT(float, FTA)), 2) AS [FT%],

ROUND(AVG(CONVERT(float, DefensiveRebounds)), 2) + ROUND(AVG(CONVERT(float, OffensiveRebounds)), 2) AS RPG,

ROUND(AVG(CONVERT(float, DefensiveRebounds)), 2) AS DRPG,

ROUND(AVG(CONVERT(float, OffensiveRebounds)), 2) AS ORPG,

ROUND(AVG(CONVERT(float, Assists)), 2) AS APG,

ROUND(AVG(CONVERT(float, Blocks)), 2) AS BPG,

ROUND(AVG(CONVERT(float, Steals)), 2) AS SPG,

ROUND(AVG(CONVERT(float, Turnovers)), 2) AS TOPG

FROM TeamTotals

GROUP BY TeamName)

SELECT

Team.TeamName,

Team.Wins,

Team.Losses,

PPG,

RANK() OVER(ORDER BY PPG DESC) AS PPGRank,

FGM,

RANK() OVER(ORDER BY FGM DESC) AS FGMRank,

FGA,

RANK() OVER(ORDER BY FGA DESC) AS FGARank,

[FG%],

RANK() OVER(ORDER BY [FG%] DESC) AS [FG%Rank],

[3PM],

RANK() OVER(ORDER BY [3PM] DESC) AS [3PMRank],

[3PA],

RANK() OVER(ORDER BY [3PA] DESC) AS [3PARank],

[3P%],

RANK() OVER(ORDER BY [3P%] DESC) AS [3P%Rank],

FTM,

RANK() OVER(ORDER BY FTM DESC) AS FTMRank,

FTA,

RANK() OVER(ORDER BY FTA DESC) AS FTARank,

[FT%],

RANK() OVER(ORDER BY [FT%] DESC) AS [FT%Rank],

RPG,

RANK() OVER(ORDER BY RPG DESC) AS RPGRank,

DRPG,

RANK() OVER(ORDER BY DRPG DESC) AS DRPGRank,

ORPG,

RANK() OVER(ORDER BY ORPG DESC) AS ORPGRank,

APG,

RANK() OVER(ORDER BY APG DESC) AS APGRank,

BPG,

RANK() OVER(ORDER BY BPG DESC) AS BPGRank,

SPG,

RANK() OVER(ORDER BY SPG DESC) AS SPGRank,

TOPG,

RANK() OVER(ORDER BY TOPG) AS TOPGRank

FROM TeamAvg JOIN Team

ON TeamAvg.TeamName = Team.TeamName

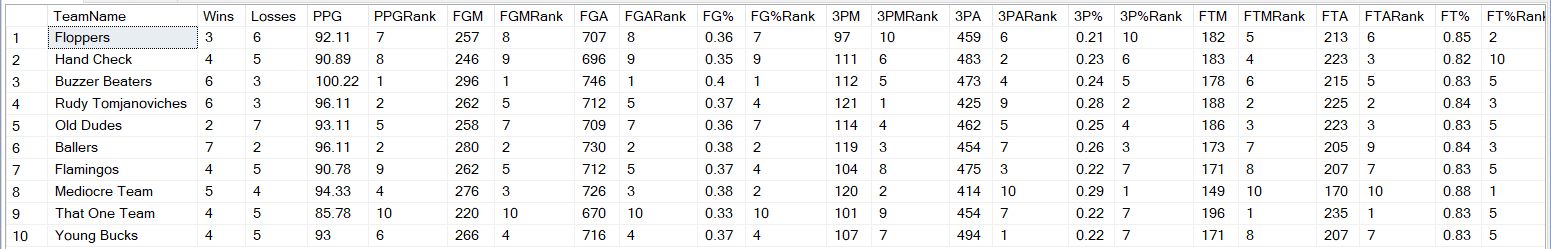
GO

SELECT \*

FROM vwTeamAvgRanks

GO

**Results – First 10 Rows:**



**Programming Objects**

**Stored Procedures**

--Create a proc that adds a row to the Position table

--Error out if the position already exists

CREATE OR ALTER PROCEDURE spAddPosition (@Position varchar(30), @Abbrv varchar(2))

AS

BEGIN

DECLARE @PositionID int

SELECT @PositionID = PositionID FROM Position

WHERE Position = @Position

--Add row using insert

INSERT INTO Position(Position, PositionAbbr)

VALUES(@Position, @Abbrv)

--Return SCOPE\_IDENTITY() So the calling code knows primary key of the row we just added

RETURN @@identity

END

GO

DECLARE @newPositionID int

EXEC @newPositionID = spAddPosition 'Water Boy’, ‘WB’

SELECT \* FROM Position WHERE PositionID = @newPositionID

**Results:** Water Boy position successfully inserted

Graphical user interface, application

Description automatically generated

--Water Boy is not an existing position specified from requirements.

--Stored procedure that deletes any record from Position that isn't in a specified position in the requirements

CREATE OR ALTER PROCEDURE spDeleteNonexistentPositions

AS

DELETE FROM Position

WHERE PositionAbbr NOT IN ('PG', 'SG', 'SF', 'PF', 'C', 'HC', 'AC')

GO

EXEC spDeleteNonexistentPositions

SELECT \* FROM Position

GO

**Results:** Water Boy position successfully deleted

Table, Excel

Description automatically generated

--Stored Procedure that updates the wins and losses on Team table to current wins and losses numbers

CREATE OR ALTER PROCEDURE spTeamRecordUpdate

AS

BEGIN

UPDATE Team

SET Team.Wins = Wins.Wins, Team.Losses = Losses.Losses

FROM Team LEFT JOIN (

SELECT

WinningTeamID,

COUNT(WinningTeamID) AS Wins

FROM Game

GROUP BY WinningTeamID) AS Wins

ON Team.TeamID = Wins.WinningTeamID LEFT JOIN (

SELECT

LosingTeamID,

COUNT(LosingTeamID) AS Losses

FROM Game

GROUP BY LosingTeamID) AS Losses

ON Team.TeamID = Losses.LosingTeamID

END

GO

EXEC spTeamRecordUpdate

SELECT \* FROM Team

GO

**Results:**

Table

Description automatically generated

**Functions**

--Function to convert HeightInches to feet and inches

CREATE OR ALTER FUNCTION dbo.fnHeightFeet(@PlayerID int)

RETURNS varchar(30) AS

BEGIN

DECLARE @returnFeet int

DECLARE @returnInches int

/\*

Get height in feet and inches and store in @returnFeet and @returnInches.

\*/

SELECT @returnFeet= HeightInches/12, @returnInches = HeightInches - ((HeightInches/12) \* 12) FROM Player

WHERE PlayerID = @PlayerID

--Return feet and inches combined in feet’inches” format

RETURN CONVERT(varchar(1), @returnFeet) + '''' + CONVERT(varchar(2), @returnInches) + '"'

END

GO

SELECT

FirstName + ' ' + LastName AS Player,

dbo.fnHeightFeet(PlayerID) AS HeightFeet

FROM Player

GO

**Results:**

Table

Description automatically generated

**User Interface**

Users will use Excel as their main interface to examine team and player performance. Various reports and interactive charts will be used for analysis.

**Establishing ODBC connection in Excel:**

Graphical user interface, text, application, email

Description automatically generated

**League Standings Report:**

This report shows the current league standings in an Excel pivot table.

Table

Description automatically generated

**Team Stat Interactive Dashboard:**

This interactive chart allows a user to select specific stats and compare them against each team.

Chart, bar chart

Description automatically generated

**League Leader Dashboard:**

This page shows pivot tables of the top 10 players for each statistical category.

Graphical user interface, application, table, Excel

Description automatically generated

**Player Stats Interactive Dashboard:**

This interactive chart allows a user to select a player to get a high-level view of their season averages.

Chart

Description automatically generated

**Reflection**

When I began with the conceptual model of this database, I had a lot of ideas that I wanted to implement. The more I dove into it, the more I realized a lot of those ideas would have required too much additional time and resources beyond the scope of this project. At that point I simplified the conceptual model to a point that felt realistic for the scope. The next time I go through the process of creating a database, the first thing I will do differently is to start with a basic conceptual model that covers all requirements, and then add to that model if I deem it necessary.

Another thing I might try differently, depending on the project, is to import test data from a reliable data source rather than generating random test data. While generating the data for my project, I found that there was a lot of numerical data that was dependent on other numerical data. Ensuring all the numbers and dependencies were correct took some time. It would have been helpful if there was already data available that would have eliminated that process.

Because I was able to go through the process of properly creating a database, I was able to see the advantages of having fully functional, normalized databases. This allowed me to answer all of the data questions that were posed.