

Course : (CISC3000) Introduction to Database system

Title : NBA statistics database design project

Professor : Prof Gong Zhi Guo

Info of Group Members :

1. WONG KAI YUAN (DC026157)
2. LAO CHON IP (DB826633)

**Table of Contents :**

1. Description of our chosen domain
2. Description to the database
3. ER diagram of the database
4. EER diagram of the database
5. DDL of tables, views, function and procedure
6. 10 ~ 15 Records of all tables (with screenshot)
7. 10 Queries of SQL

**A) Description of our chosen domain**

The NBA, or National Basketball Association, is a men's professional basketball league in North America. The league consists of 30 teams, with 29 teams in the United States and one team in Canada. The NBA is one of the four major professional sports leagues in the United States and Canada, and is widely regarded as the premier men's professional basketball league in the world. The NBA season runs from October to June, and includes regular season games, playoffs, and a championship series known as the NBA Finals. The NBA has a rich history of great players, teams, and moments, and the league continues to be a popular source of entertainment and inspiration for basketball fans around the world.

The NBA statistics database is designed to store information about players, teams, games, and statistics related to the NBA.

**B) Description to the database**

1. **Description of entities sets and attribute**

The entity sets in this database include Players, Teams, Games, Stats, Coaches, Drafts, and Awards. Each player entity has attributes such as player ID, first name, last name, position, height, weight, birthdate, team ID. Each team entity has attributes such as team ID, name, city, arena, founded year, and coach ID. Each game entity has attributes such as game ID, date, home team ID, away team ID, home team score, away team score and winning team ID. Each stats entity has attributes such as stat ID, player ID, game ID, points, rebounds, assists, steals, blocks, and turnovers. Each coach entity has attributes such as coach ID, first name, last name and team ID. Each draft entity has attributes such as draft ID, player ID, draft year, pick, and team ID. Each award entity has attributes such as award ID, award name, player ID, and award year.

1. **Description of relationship sets**

The relationships between the entity sets are established through foreign keys. For example, each player can belong to only one team, but each team can have multiple players. Each game involves two teams, and each player can have multiple stats for multiple games. Each team can have only one coach, and each coach can only belong to one team. Each draft pick can only belong to one team, and each team can have multiple draft picks. Each award can only be won by one player, and each player can win multiple awards.

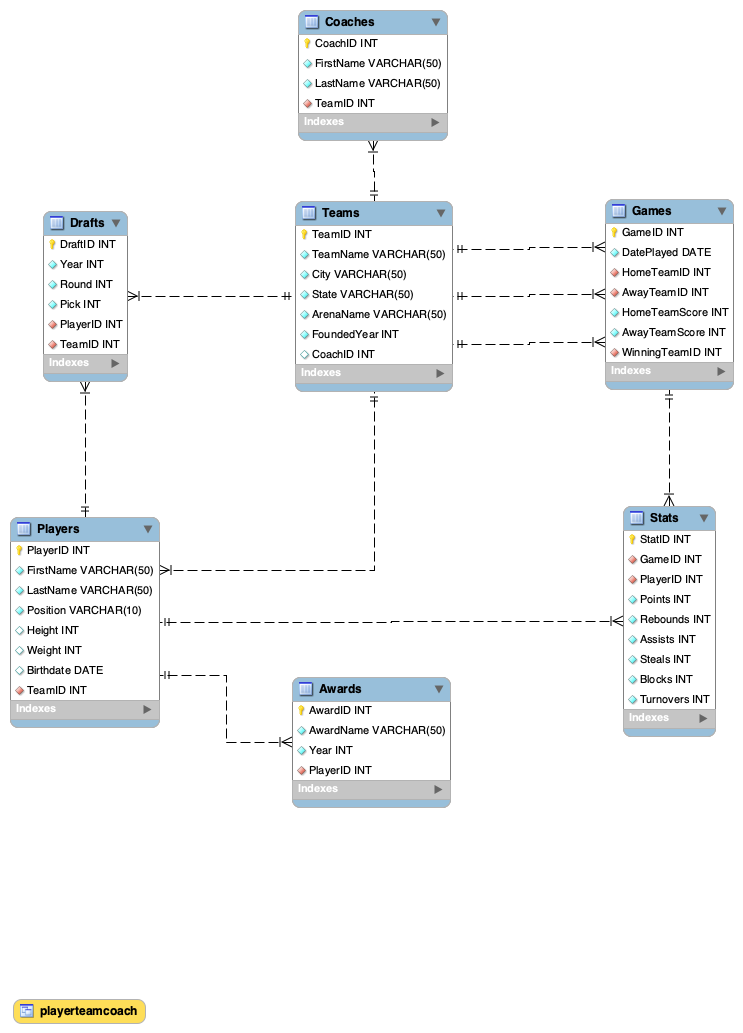
1. **Description of Keys and Constraints**

Constraints such as unique keys and foreign keys are used to ensure data integrity in the database. For example, player ID is a unique key for the Players entity set, and team ID is a foreign key in the Players entity set. Similarly, game ID is a unique key for the Games entity set, and home team ID and away team ID are foreign keys in the Games entity set.

**C) ER diagram of the database**

****

**D) EER diagram of the database**



**E) DDL of tables, views, function and procedure**

1. **DDL of tables**

create database NBA

use NBA;

-- Create the Players table

CREATE TABLE Players (

PlayerID INT PRIMARY KEY,

FirstName VARCHAR(50) NOT NULL,

LastName VARCHAR(50) NOT NULL,

Height INT NOT NULL,

Weight INT NOT NULL,

Position VARCHAR(10) NOT NULL,

Birthdate DATE NOT NULL,

College VARCHAR(50) NOT NULL

);

-- Create the Teams table

CREATE TABLE Teams (

TeamID INT PRIMARY KEY,

TeamName VARCHAR(50) NOT NULL,

City VARCHAR(50) NOT NULL,

State VARCHAR(50) NOT NULL,

ArenaName VARCHAR(50) NOT NULL,

FoundedYear INT NOT NULL,

CoachID INT

);

-- Create the Games table

CREATE TABLE Games (

GameID INT PRIMARY KEY,

DatePlayed DATE NOT NULL,

HomeTeamID INT NOT NULL,

AwayTeamID INT NOT NULL,

HomeTeamScore INT NOT NULL,

AwayTeamScore INT NOT NULL,

WinningTeamID INT NOT NULL,

FOREIGN KEY (HomeTeamID) REFERENCES Teams(TeamID),

FOREIGN KEY (AwayTeamID) REFERENCES Teams(TeamID),

FOREIGN KEY (WinningTeamID) REFERENCES Teams(TeamID)

);

-- Create the Stats table

CREATE TABLE Stats (

StatID INT PRIMARY KEY,

GameID INT NOT NULL,

PlayerID INT NOT NULL,

Points INT NOT NULL,

Rebounds INT NOT NULL,

Assists INT NOT NULL,

Steals INT NOT NULL,

Blocks INT NOT NULL,

Turnovers INT NOT NULL,

FOREIGN KEY (GameID) REFERENCES Games(GameID),

FOREIGN KEY (PlayerID) REFERENCES Players(PlayerID)

);

-- Create the Coaches table

CREATE TABLE Coaches (

CoachID INT PRIMARY KEY,

FirstName VARCHAR(50) NOT NULL,

LastName VARCHAR(50) NOT NULL,

TeamID INT NOT NULL,

FOREIGN KEY (TeamID) REFERENCES Teams(TeamID)

);

-- Create the Drafts table

CREATE TABLE Drafts (

DraftID INT PRIMARY KEY,

Year INT NOT NULL,

Round INT NOT NULL,

Pick INT NOT NULL,

PlayerID INT NOT NULL,

TeamID INT NOT NULL,

FOREIGN KEY (PlayerID) REFERENCES Players(PlayerID),

FOREIGN KEY (TeamID) REFERENCES Teams(TeamID)

);

-- Create the Awards table

CREATE TABLE Awards (

AwardID INT PRIMARY KEY,

AwardName VARCHAR(50) NOT NULL,

Year INT NOT NULL,

PlayerID INT NOT NULL,

FOREIGN KEY (PlayerID) REFERENCES Players(PlayerID)

);

1. **DDL of views**

-- View to display player information along with their team name and coach name

CREATE VIEW PlayerTeamCoach AS

SELECT Players.PlayerID, Players.FirstName, Players.LastName, Players.Position, Players.Height, Players.Weight, Teams.TeamName, Coaches.FirstName AS CoachFirstName, Coaches.LastName AS CoachLastName

FROM Players

INNER JOIN Teams ON Players.TeamID = Teams.TeamID

INNER JOIN Coaches ON Teams.CoachID = Coaches.CoachID;

1. **DDL of function :**

-- Natural Language : Function to get player's average points per game

DELIMITER //

CREATE FUNCTION AvgPointsPerGame(InputTeamID INT)

RETURNS INT

DETERMINISTIC

BEGIN

DECLARE AvgPoints INT;

SELECT AVG(Points) INTO AvgPoints FROM Stats WHERE PlayerID IN (SELECT PlayerID FROM Players WHERE TeamID = InputTeamID);

RETURN (AvgPoints);

END;

1. **DDL of procedure :**

-- Natural Language : Procedure to update a player's height and weight

DELIMITER //

-- Procedure to update a player's height and weight

CREATE PROCEDURE UpdatePlayerHeightAndWeight(IN PlayerID INT, IN NewHeight INT, IN NewWeight INT)

BEGIN

UPDATE Players SET Height = NewHeight, Weight = NewWeight WHERE PlayerID = PlayerID;

END

**F) 10 ~ 15 Records of all tables (with screenshot)**

1. Teams



1. Players



1. Games



1. Stats



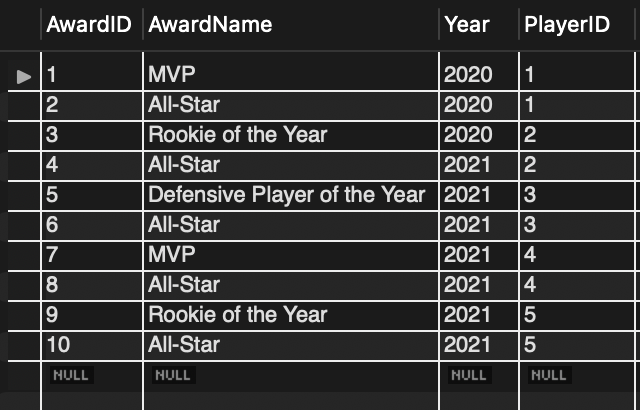
1. Coaches



1. Drafts



1. Awards



**G) 10 Queries of SQL**

1. Find the name of the coach for each team: (natural language)

SELECT TeamName, FirstName, LastName

FROM Teams

JOIN Coaches ON Teams.CoachID = Coaches.CoachID;



1. Find the number of games played by each team: (natural language)

SELECT TeamName, COUNT(GameID) AS NumGames

FROM Teams

JOIN Games ON Teams.TeamID = Games.HomeTeamID OR Teams.TeamID = Games.AwayTeamID

GROUP BY TeamName;



1. Find the average height of players on each team: (natural language)

SELECT TeamName, AVG(Height) AS AvgHeight\_cm

FROM Teams

JOIN Players ON Teams.TeamID = Players.TeamID

GROUP BY TeamName;



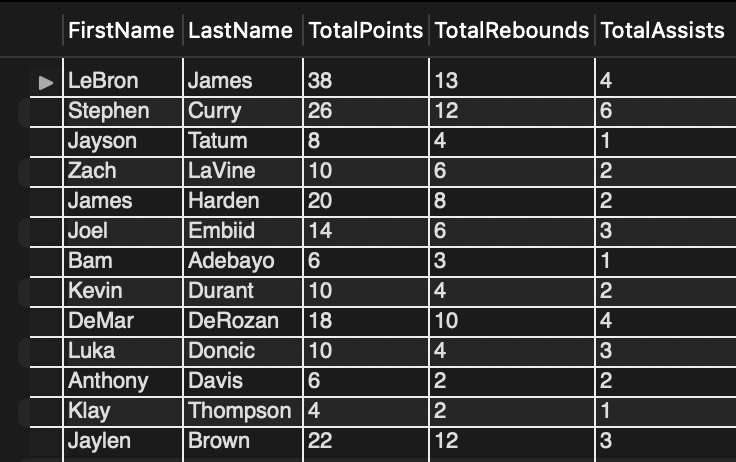
1. Find the total number of points, rebounds, and assists for each player: (natural language)

SELECT FirstName, LastName, SUM(Points) AS TotalPoints, SUM(Rebounds) AS TotalRebounds, SUM(Assists) AS TotalAssists

FROM Players

JOIN Stats ON Players.PlayerID = Stats.PlayerID

GROUP BY FirstName, LastName;



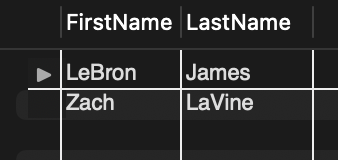
1. Find the names of players who have won the MVP award: (natural language)

SELECT FirstName, LastName

FROM Players

JOIN Awards ON Players.PlayerID = Awards.PlayerID

WHERE AwardName = 'MVP';



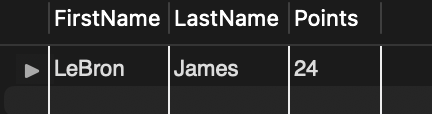
1. Find the name of the player who scored the most points in a game: (natural language)

SELECT FirstName, LastName, Points

FROM Players

JOIN Stats ON Players.PlayerID = Stats.PlayerID

WHERE Points = (SELECT MAX(Points) FROM Stats);

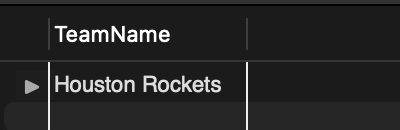


1. Find the teams that have not won a game yet: (natural language)

SELECT TeamName

FROM Teams

WHERE NOT EXISTS (SELECT \* FROM Games WHERE Teams.TeamID = Games.WinningTeamID);



1. Find the teams that have won more than half of their games: (natural language)

SELECT TeamName

FROM Teams

JOIN Games ON Teams.TeamID = Games.HomeTeamID OR Teams.TeamID = Games.AwayTeamID

GROUP BY TeamName

HAVING SUM(CASE WHEN Teams.TeamID = Games.WinningTeamID THEN 1 ELSE 0 END) > COUNT(\*)/2;



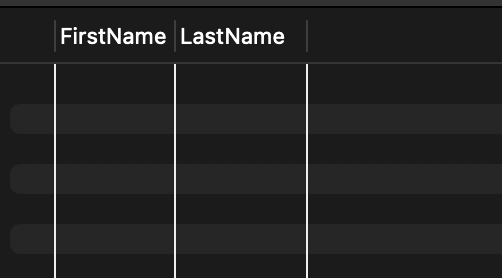
1. Find the players who have played for more than one team: (natural language)

SELECT FirstName, LastName

FROM Players

GROUP BY FirstName, LastName

HAVING COUNT(DISTINCT TeamID) > 1;

 (empty)

1. Find the players who were drafted in the first round: (natural language)

SELECT FirstName, LastName, Year, Round, Pick

FROM Players

JOIN Drafts ON Players.PlayerID = Drafts.PlayerID

WHERE Round = 1;

