

Relational Database
For The National Basketball Association

Anthony Dushaj

Database Management

November 25, 2019

Table of Contents

Overall Purpose & Business Rules.....	3
Chen's Entity Relationship Model.....	5
SQL Table Statements & Table Descriptions.....	6
Normalization Justification.....	17
SQL Query Statements & Results.....	18

Overall Purpose & Business Rules

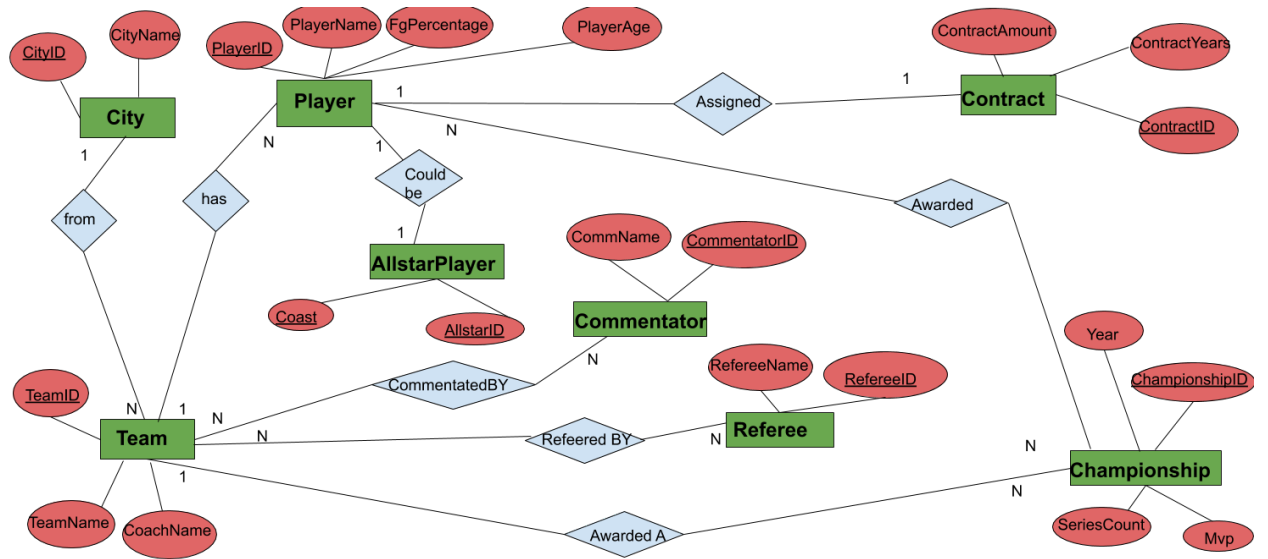
This Database for the National Basketball Association will consist of a platform of data that will allow an individual to find a wide variety of relevant information pertaining to different teams and players in the NBA. For example, if a fan wanted to know how many championship rings a player or team has, they can use the database to find that data. Another example is, someone like a scout wanting to know information about a players contract so they can see when the player that they want will become a free agent and the range that their salary is in. Lastly, even a player may want to use the database to see statistics like their field goal percentage and see where they might want to prove on the next following offseason.

The database will consist of several different entities. Starting with the entity Player, many different players can have one team. Each player must be assigned to a team. Every single player receives one contract that is distinct from everyone elses. Also each player could be an all star player, this a supertype relationship as an allstar is a player and carries all the same attributes. Moreover, allstar is a player but a special instance of one. There should only be 24 all star players, 12 being from the west coast, and 12 being from the east coast. No player can be more than one allstar, or have more than one contract, nor be on more than one team at a time. Many players can win and receive many different championships. Some players may have championship wins with several different teams in the past. Only one player can be a championship MVP per championship, however a player can be a MVP multiple times.

A single championship is also awarded to one team only, moreover one team can win many different championships. There will be 18 different teams. Each team will consist of 10 different players. Also there will be multiple referees that will referee one or more different

teams. There will be a total of 5 referees. Multiple teams can be from one city. An example is in the NBA today there is a team called the Clippers that is from Los Angeles and a team called the Lakers that is also from Los Angeles. Each team will also have one or more commentators that commentate them. There will be a total of 3 commentators that will commentate the teams in the NBA.

Entity Relationship Diagram



SQL Table Statements & Descriptions

Player Table

```
create table xplayer(

playerID INT,
playerName VARCHAR2(45),
playerAge INT,
fgPercentage INT,
contractID INT,
teamID INT,

CONSTRAINT PK_PLAYERID PRIMARY KEY (PLAYERID),
CONSTRAINT FK_TEAMID FOREIGN KEY (TEAMID) REFERENCES XTEAM (TEAMID),
CONSTRAINT FK_CONTRACTID FOREIGN KEY (CONTRACTID) REFERENCES
XCONTRACT (CONTRACTID)
);
```

This entity stores data about each player and the specific characteristics, and statistics related to this player. The first attribute which is the playerID, is the primary key. Players will have a contract ID and team ID. The attribute contract ID will be a foreign key that references the primary key contract ID in the table Contract, each player is assigned a contract ID that will tell us information about their contract. This is a one to one relationship as one player may only receive one contract. The attribute team ID will also be a foreign key references the primary key team ID in the team table this lets us know which team the player is associated with. This is a many to one relationship as many players can consist of one team.

AllstarPlayer Table

```
create table xallstarPlayer(  
  
allstarID INT,  
coast VARCHAR2(15),  
  
CONSTRAINT PK_ALLSTARCOAST PRIMARY KEY (ALLSTARID, COAST),  
CONSTRAINT FK_ALLSTARID FOREIGN KEY (ALLSTARID) REFERENCES XPLAYER  
(PLAYERID)  
);
```

The Allstarplayer table is a supertype relation of the player table, the primary key is both the allstarID and the coast. As a player can't be an allstar from both the east and west coast (in one particular season). The foreign key allstar ID will reference the primary key player ID in the player table which will tell us all the details of that specific player that is an allstar. This is a one to one relationship as one player can be only be one allstar.

Team Table

```
create table xteam(

teamID INT,
teamName VARCHAR2(30),
coachName VARCHAR2(30),
cityID INT,

CONSTRAINT PK_TEAMID PRIMARY KEY (TEAMID),
CONSTRAINT FK_TEAMCITYID FOREIGN KEY (CITYID) REFERENCES XCITY (CITYID)

);
```

The team table consists of a primary key called the team ID, which will be referenced by many other tables in this database. The foreign key city ID will reference the primary key city ID in the city table to give us the details on which city the team is from. This is a one to many relationship in that multiple teams can be from one city.

City Table

```
create table xcity(  
  
cityID INT,  
cityName VARCHAR2(40),  
  
CONSTRAINT PK_CITYID PRIMARY KEY (CITYID)  
  
);
```

The City table has a primary key of City ID, and an attribute of city Name. City ID is the unique identifier that each team will have and the city ID determines the name of the city in this table.

Contract Table

```
create table xcontract(  
  
contractID INT,  
contractAmount LONG,  
contractYears INT,  
  
CONSTRAINT PK_CONTRACTID PRIMARY KEY (CONTRACTID)  
  
);
```

The primary key in the contract table is the contract ID, the contract ID will determine the contract amount along with the contract years. As stated before, each player is only given one of these contracts.

PlayerChampionship Table

```
create table xplayerChampionship(  
  
    playerID INT,  
    championshipID INT,  
  
    CONSTRAINT PK_PLAYERIDCHAMPID PRIMARY KEY (PLAYERID, CHAMPIONSHIPID),  
    CONSTRAINT FK_PLAYERID FOREIGN KEY (PLAYERID) REFERENCES XPLAYER  
    (PLAYERID),  
    CONSTRAINT FK_CHAMPID FOREIGN KEY (CHAMPIONSHIPID) REFERENCES  
    XCHAMPIONSHIP (CHAMPIONSHIPID)  
  
);
```

The Primary key in the player championship table is a candidate key of both a championship ID, along with a player ID. The foreign key player ID will reference the player ID in the player table, while the foreign key championship ID will reference the championshipID in the championship table. This table will tell us which players won which championship. This is a many to many relationship as many players may win many championships.

Championship Table

```
create table xchampionship(  
  
championshipID INT,  
championshipYear NUMBER,  
seriesCount VARCHAR2(15),  
mvpName VARCHAR2(45),  
teamID INT,  
  
CONSTRAINT PK_CHAMPID PRIMARY KEY (CHAMPIONSHIPID),  
CONSTRAINT FK_TEAMCHIPID FOREIGN KEY (TEAMID) REFERENCES XTEAM  
(TEAMID)  
  
);
```

The primary key in this table will be championship ID, which will tell us what year, the series count, the mvp name, and the team ID of that championship. The foreign key team ID references the team table which will tell us what specific team won the championship. This is a many to one relationship as many championships can be won by one team.

Commentator Table

```
create table xcommentator(  
  
commentatorID INT,  
commentatorName VARCHAR2(25),  
  
CONSTRAINT PK_COMMID PRIMARY KEY (COMMENTATORID)  
  
);
```

The primary key is the commentator ID which will determine the name of the commentator.

CommentatedBy Table

```
create table xcommentatedBy(  
  
commentatorID INT,  
teamID INT,  
  
CONSTRAINT PK_TEAMIDCOMMID PRIMARY KEY (COMMENTATORID, TEAMID),  
CONSTRAINT FK_TEAMCOMMID FOREIGN KEY (COMMENTATORID) REFERENCES  
XCOMMENTATOR (COMMENTATORID),  
CONSTRAINT FK_COMM_TEAMID FOREIGN KEY (TEAMID) REFERENCES XTEAM  
(TEAMID)  
  
);
```

The commentated by table has a primary key that is a candidate key of both commentator ID, and team ID. While the foreign key team ID references team ID in the table team, and the foreign key commentator ID will reference the commentator ID in the commentator table. This table will let us know which teams are commentated by which commentators. This is a many to many commentators may commentate many teams.

Referee Table

```
create table xreferee(

refereeID INT,
refereeName VARCHAR2(35),
CONSTRAINT PK_REFFID PRIMARY KEY (REFereeID)

);
```

The primary key is the referee ID which will determine the name of the referee.

RefereedBy Table

```
create table xrefereedBy(

refereeID INT,
teamID INT,

CONSTRAINT PK_REFFIDTEAMID PRIMARY KEY (REFEREEID, TEAMID),
CONSTRAINT FK_TEAMREFID FOREIGN KEY (TEAMID) REFERENCES XTEAM
(TeamID),
CONSTRAINT FK_REFFIDREFF FOREIGN KEY (REFEREEID) REFERENCES XREFEREE
(REFEREEID)

);
```

The commented by table has a primary key that is a candidate key of both referee ID, and team ID. While the foreign key team ID references team ID in the table team, and the foreign key referee ID will reference the referee ID in the referee table. This table will let us know which teams are refereed by which referees. This is a many to many relationship as many referees may referee many teams.

Justification of Normalization

A database is in first normal form if it has no repeating groups and it has a primary key.

A database is in second normal if it is in first normal form, and has no partial dependencies.

Partial dependency means that a nonprime attribute is functionally dependent on part of a candidate key. A nonprime attribute is an attribute that's not part of any candidate key. A

database is in third normal form if it is in second normal form and there are no transitive dependencies. Furthermore, Transitive dependencies are when a nonkey attribute determines another attribute. With all of that being said, this NBA Database satisfies all of the previous requirements of being in third normal form.

SQL Query Statements

1. Get playername, allstarid, championshipid for players who are an allstar and an nba champion, as well as those for either, as well as those for neither

SQL:

```
select playername, xallstarplayer.allstarID, xchampionship.championshipID
from xplayer
full outer join xallstarplayer on (xallstarplayer.allstarID = xplayer.playerid)
full outer join xplayerchampionship on
(xplayerchampionship.playerid = xplayer.playerid)
full outer join xchampionship on
(xplayerchampionship.championshipid = xchampionship.championshipid)
```

PLAYERNAME	ALLSTARID	CHAMPIONSHIPID
1 Adams, Steven	(null)	(null)
2 Adebayo, Bam	(null)	(null)
3 Aldridge, LaMarcus	(null)	(null)
4 Alexander-Walker, Nickeil	(null)	(null)
5 Barea, J.J.	5	1
6 Barnes, Harrison	(null)	5
7 Barnes, Harrison	(null)	2
8 Barrett, RJ	(null)	(null)
9 Barton, Will	(null)	(null)
10 Steven, Codd.	(null)	(null)
11 Barnes, Harrison	(null)	(null)
12 Barrett, Warren	(null)	(null)
13 Barton, Smith	(null)	(null)
14 Curry, Seth	13	(null)
15 Curry, Stephen	(null)	18
16 Curry, Stephen	(null)	10
17 Curry, Stephen	(null)	3
18 Chriss, Marquese	(null)	(null)
19 Clark, Gary	(null)	(null)
20 James, Lebron	17	19
21 James, Lebron	17	11
22 Clarkson, Jordan	(null)	(null)

Cardinality: 213

2. Get all star numbers for players who are all stars and played on a team that won a championship

SQL:

```
select distinct xallstarplayer.allstarID
from xallstarplayer
where xallstarplayer.allstarID in
      (select xplayer.playerID
       from xplayer
       where xplayer.teamID IN
            (select xteam.teamID
             from xteam
             where xteam.teamID In
                  (select xchampionship.teamID
                   from xchampionship, xplayer)))
```

	ALLSTARID
1	70
2	59
3	86
4	158
5	79
6	60
7	17
8	5
9	58
10	72
11	27
12	67
13	151
14	24
15	53
16	93
17	13
18	90

Cardinality: 18

3. Get team name for teams that are commented by every commentator

SQL:

```
select teamname
from xteam
where not exists(
    select *
    from xcommentator xc
    where not exists(
        select *
        from xcommentatedby
        where xc.COMMENTATORID = xcommentatedby.COMMENTATORID
        and xteam.teamid = xcommentatedby.teamID))
```

	TEAMNAME
1	Bulls
2	Blazers
3	Magic

Cardinality: 3

4. Get MVPname, team name , team id for players who were an mvp more than once on two or more different teams

SQL:

```
select distinct teamname, champ1.teamID, champ1.mvpname
from xteam, xchampionship champ1, xchampionship champ2
where champ1.mvpname = champ2.mvpname
and champ1.teamID <> champ2.teamID
and xteam.teamid = champ1.teamID
```

	TEAMNAME	TEAMID	MVPNAME
1	Timberwolves	11	Bryant, Kobe.
2	Nets	3	James, LeBron
3	Hornets	4	James, LeBron
4	Mavericks	6	Bryant, Kobe.

Cardinality: 4

5. get city name, team name, along with team ID for those who won a championship including for those who didn't win a championship

SQL:

```
select cityname, teamname, xchampionship.teamID
```

```
from xteam
```

```
join xcity on (xcity.cityid = xteam.cityid)
```

```
left outer join xchampionship on (xteam.teamID = xchampionship.teamID)
```

	CITYNAME	TEAMNAME	TEAMID
1	Chicago	Bulls	5
2	Boston	Celtics	(null)
3	Los Angeles	Kings	14
4	Miami	Magic	17
5	Detroit	Pelicans	(null)
6	Miami	Heat	(null)
7	New Orleans	Super Sonic	18
8	Dallas	Mavericks	6
9	Minnesota	Timberwolves	11
10	Denver	Nuggets	(null)
11	New Orleans	Hornets	4
12	Detroit	Pistons	8
13	Denver	Pacers	(null)
14	Los Angeles	Lakers	9
15	Los Angeles	Clippers	10
16	Atlanta	Hawks	1
17	Portland	Blazers	(null)
18	Brooklyn	Nets	3

Cardinality: 18

6. get city name and team name for teams in which none of the players have a contract for 10 years or more

SQL:

```
select cityname, teamname
from xcity, xteam
where xcity.cityid = xteam.cityid
and xteam.teamid not in
    (select xplayer.teamid
     from xplayer, xcontract
     where xplayer.contractid = xcontract.contractid
     and contractyears >= 10)
```

	CITYNAME	TEAMNAME
1	Atlanta	Hawks
2	Boston	Celtics
3	Brooklyn	Nets
4	New Orleans	Super Sonic
5	Chicago	Bulls
6	Dallas	Mavericks
7	Denver	Pacers
8	Denver	Nuggets
9	Detroit	Pelicans
10	Los Angeles	Kings
11	Los Angeles	Clippers
12	Los Angeles	Lakers
13	Minnesota	Timberwolves
14	Miami	Heat

Cardinality: 14

7. get team names for teams that are only referred by Maur Ken

SQL:

```
select teamname
from xteam
where xteam.teamID not in
      (select XREFEREEDBY.teamID
       from XREFEREEDBY
       where xrefereedby.refereeid not in
            (select xreferee.refereeid
             from XREFEREE
             where xreferee.refereename = 'Mauer, Ken'))
```

	TEAMNAME
1	Celtics
2	Bulls
3	Pelicans

Cardinality: 3

8. get playername, playerage, allstarid for players over the age 30 who and are an allstar including for those are not an allstar

SQL:

select playername, playerage, allstarid

from xallstarplayer

right outer join xplayer on (xallstarplayer.allstarID = xplayer.playerid)

where playerage > 30

order by playerage

	PLAYERNAME	PLAYERAGE	ALLSTARID
1	Barton, Will	31	(null)
2	Villanueva, Charlie	31	(null)
3	Scheffler, Tom	31	(null)
4	Shamet, Landry	31	(null)
5	Kenon, Larry	31	(null)
6	Turpin, Mel	31	(null)
7	Nene	31	(null)
8	Johnson, JaJuan	32	(null)
9	Tyra, Charlie	32	(null)
0	Scott, Mike	32	(null)
1	Larese, York	32	(null)
2	LaVine, Zach	32	27
3	Chriss, Marquese	32	(null)
4	Schlueter, Dale	32	(null)
5	Wanamaker, Brad	32	(null)
6	Trier, Allonzo	33	(null)
7	Schofield, Admiral	33	(null)
8	Dudley, Jared	33	(null)
9	Kerner, Jonathan	33	(null)
0	Ingram, Brandon	33	(null)
1	Iverson, Allen	33	59
2	Sitton, Charlie	33	(null)
3	Pippen, Scotty	34	174

Cardinality: 62

9. get player name, the amount of their contract, along with team and city id for players whose contract end in 2024 and play on a team whose team name starts with the letter 'D' and is refereered by Barnaky, Brent

SQL:

```
select playername, contractamount, xteam.teamID, xcity.cityID
from xteam, xplayer, xcontract, xreferee, xrefereedby, xcity
where xplayer.contractid = xcontract.contractid
and xcontract.contractyears = 5
and xplayer.playerid = xteam.teamid
and xteam.cityid = xcity.cityid
and xcity.cityname like 'D%'
and xteam.teamid = xrefereedby.teamid
and xrefereedby.refereeid = xreferee.refereeid
and xreferee.refereename = 'Barnaky, Brent'
```

	PLAYERNAME	CONTRACTAMOUNT	TEAMID	CITYID
1	Barrett, RJ	3100000	7	7

Cardinality: 1

10. get team name and average of fg percentage for all the players on the team of the team with the highest average of fg percentage for all their players

SQL:

```
select teamname, avg(fgpercentage)as highestFGPercentage
from xteam, xplayer
where xplayer.teamid = xteam.teamid
group by teamname having avg(fgpercentage) in
(select (max(avg(fgpercentage)))
from xteam, xplayer
where xplayer.teamid = xteam.teamid
group by teamname)
```

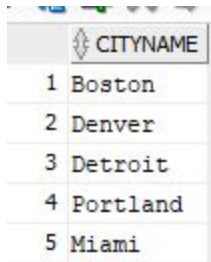
	TEAMNAME	HIGHESTFGPERCENTAGE
1	Pelicans	34.2

Cardinality: 1

11. get city names of teams that never won a championship

SQL:

```
select cityname
from xcity
where xcity.cityID in
      (select xteam.cityid
       from xteam
       where xteam.teamid not in(
         select xchampionship.teamid
         from xchampionship
       ))
```



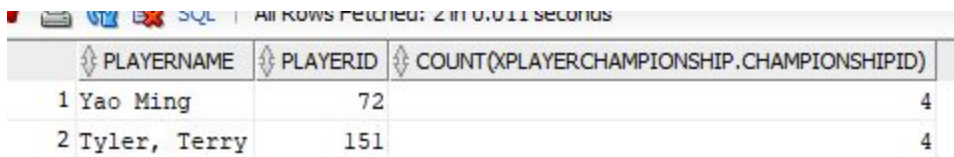
	CITYNAME
1	Boston
2	Denver
3	Detroit
4	Portland
5	Miami

Cardinality: 5

12. Get playername, player id, and number of championships they won for the players that won the maximum amount of championships

SQL:

```
select distinct playername, xplayerchampionship.playerid, count(
xplayerchampionship.championshipid)
from xplayer
join xplayerchampionship on (xplayer.playerid = xplayerchampionship.playerid)
group by playername, xplayerchampionship.playerid
having count( xplayerchampionship.championshipid) =
(
    select max(count(distinct x1.championshipid))
    from xplayerchampionship x1
    group by playerid
)
```



	PLAYERNAME	PLAYERID	COUNT(XPLAYERCHAMPIONSHIP.CHAMPIONSHIPID)
1	Yao Ming	72	4
2	Tyler, Terry	151	4

Cardinality:2

13. Get team names for teams that are refereed by every ref, or are commentated by every commentator

SQL:

```
select teamname
from xteam
where not exists(
    select *
    from xcommentator xc
    where not exists(
        select *
        from xcommentatedby
        where xc.COMMENTATORID = xcommentatedby.COMMENTATORID
        and xteam.teamid = xcommentatedby.teamID))
or not exists(
    select *
    from xreferee
    where not exists(
        select *
        from xrefereedby
        where xreferee.refereeid = xrefereedby.refereeid
        and xteam.teamid = xrefereedby.teamid))
```

	TEAMNAME
1	Bulls
2	Nuggets
3	Timberwolves
4	Blazers
5	Magic

Cardinality:5

14. Get mvp name, teamid, and series count for the team that won the most recent championship

SQL:

```
Select teamid, seriescount, mvpname  
from xchampionship  
where Rowid in  
(  
    select max(Rowid) from xchampionship  
);
```

	TEAMID	SERIESCOUNT	MVPNAME
1	184	to 2	Turner, Myles

Cardinality:1