Final Project: NBA Database

CSS 475

Team: G10

Member: Alexander Carswell
Peter Pitojo
Peter Van
Tien Huynh

Table of Contents

Introduction

For our project we are creating a NBA statistical database. This database will hold data about the performance of individual players, and teams. Some of the people who might use this data include sports analysis, the teams themselves, NBA fantasy players and many others. This data can help teams identify what they need to do to get better. Or it may be used as something interesting for sportscasters to talk about when they are about to have an awkward silence on air.

Database Application

This will be a sports team database containing information of the statistics of individual players and teams in the National Basketball Association (NBA). Fans can use the database to look for game related statistics and use them for their own purposes.

Entities

The database will contain individual player's statistics as well as overall team statistics. The data will be containing the specific information as follows:

CITY

City_Id	City Name	State

STADIUM

Stadium_Id	Stadium_Name	Seats	Year_Built	S_CityId

PLAYER_CONTRACT

Player ID	Team	Year_Start	Year_End	Salary

PLAYER_STATS

Player_ID	FG	3P_percent	PPG	RPG	APG	TOPG

PLAYER_PROFILE

Player_ID	Player Name	DOB	College	Height	Weight	Position

DIVISION

Div_Id	Div_Name	Conference_Name

SPONSOR

Sponsor_Id	Sponsor_Name	Sponsor_Business

SPONSOR_CONTRACT

Sponsor_Id	Player_Id	Amount	Start_Date	End_Date

TEAM

Div_Id	Team_Name	City	Away_percentage	Home_Win_Percentage	Championship_ Wins

COACH

Coach_Name	Team	Total_Championship_Wins

OWNER

ID_Number	Name	Team_Purchase_Value	Team

TEAM_WORTH

t_Name	t_Value

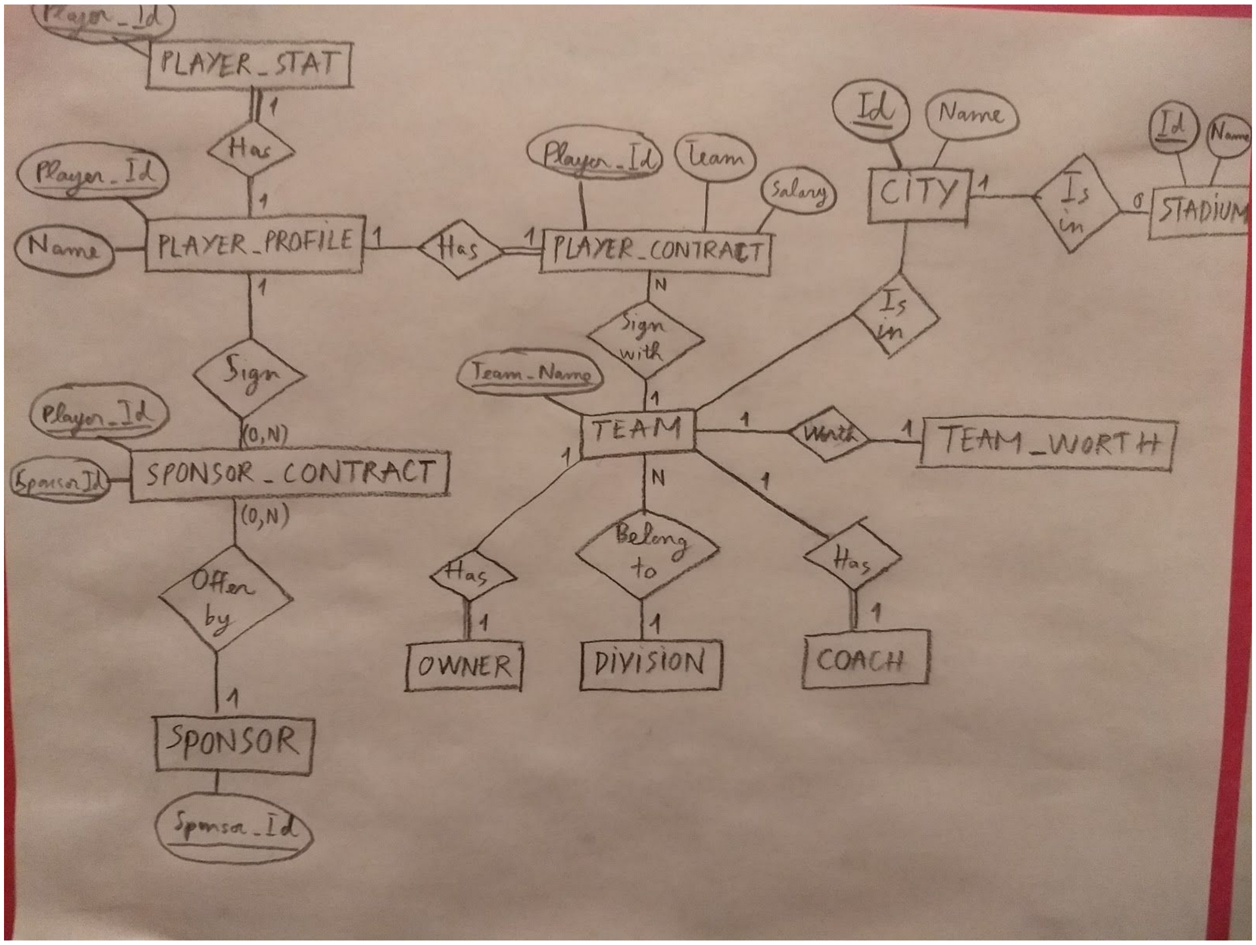
Designs

Assumptions

- Fans are only interested in data that we have selected and no more
- The statistics for players and coaches are only relevant for the current team they are on
- Teams only have one owner (The one with the most shares)
- Only holds present information, as opposed to past and present

Entity Relation Diagram

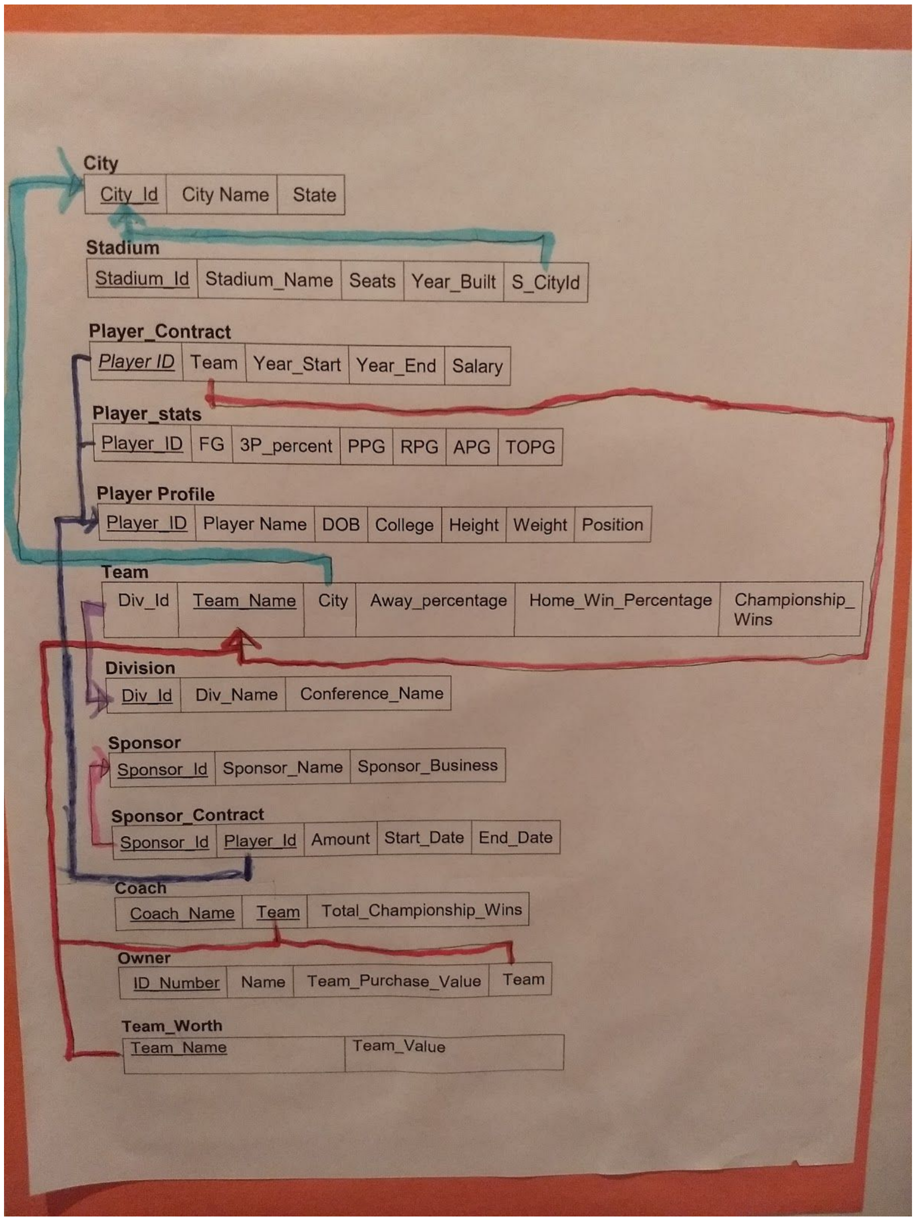
The entity relationship diagram demonstrates the key entities that will exist in our NBA statistical database as well as the interactions that exist between them. One important item to note in our ER diagram is that we chose to assign a unique ID to the Owner entity in order to fulfill its key constraint. We chose to do this instead of using a combination of existing attributes because any combination of existing attributes will not be able to consistently produce a unique key for owners.



Relational Data Model

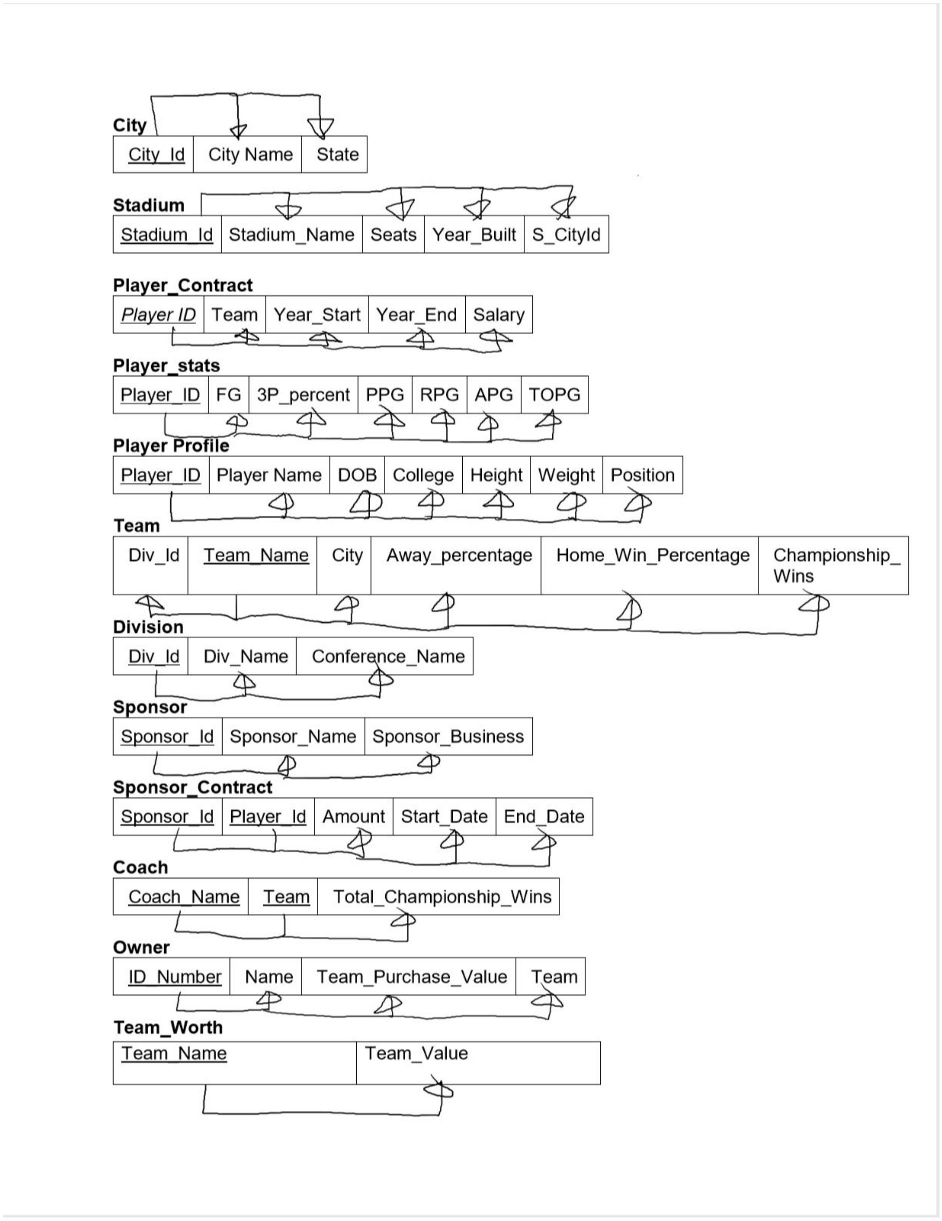
Relational data model is a model we used to illustrate our data. The relational data model represent data as a collection of table. The NBA Database's relational data model is mapped from our NBA database entity relation diagram, which means each table are representative of entity in our diagram.

The figure below is a result of using relational model to visualize the NBA Database's. Each table are constructed from entities that exist in our entity relation diagram, as such each table will be given a name based on the entity it represents.


City
City_Id | City Name | State
Stadium
Stadium_Id | Stadium_Name | Seats | Year_Built | S_CityId
Player_Contract
Player ID | Team | Year_Start | Year_End | Salary
Player_stats
Player_ID | FG | 3P_percent | PPG | RPG | APG | TOPG
Player Profile
Player_ID | Player Name | DOB | College | Height | Weight | Position
Team
Div_Id | Team_Name | City | Away_percentage | Home_Win_Percentage | Championship_ Wins
Division
Div_Id | Div_Name | Conference_Name
Sponsor
Sponsor_Id | Sponsor_Name | Sponsor_Business
Sponsor_Contract
Sponsor_Id | Player_Id | Amount | Start_Date | End_Date
Coach
Coach_Name | Team | Total_Championship_Wins
Owner
ID_Number | Name | Team_Purchase_Value | Team
Team_Worth
Team_Name | Team_Value

Normalization

All of our tables are normalized to 3NF. The following displays the functional dependencies that exist in each table.



City

City_Id	City Name	State

Stadium

Stadium_Id	Stadium_Name	Seats	Year_Built	S_CityId

Player_Contract

Player ID	Team	Year_Start	Year_End	Salary

Player_stats

Player_ID	FG	3P_percent	PPG	RPG	APG	TOPG

Player Profile

Player_ID	Player Name	DOB	College	Height	Weight	Position

Team

Div_Id	Team_Name	City	Away_percentage	Home_Win_Percentage	Championship_ Wins

Division

Div_Id	Div_Name	Conference_Name

Sponsor

Sponsor_Id	Sponsor_Name	Sponsor_Business

Sponsor_Contract

Sponsor_Id	Player_Id	Amount	Start_Date	End_Date

Coach

Coach_Name	Team	Total_Championship_Wins

Owner

ID_Number	Name	Team_Purchase_Value	Team

Team_Worth

Team_Name	Team_Value

Tooling Assessment

Database Software

SQL server is a relational database management system built on top of SQL. In general SQL server has great tooling support (e.g. support for ORM framework which has to do with mapping objects to tables). In addition it has good integration if you want to use the database to a web or mobile app. It also has support from visual studios making it more easy and convenient to use.

UI Tools

Our group chose to use a windows form application written in C# for our database UI. We decided on a windows form application because it has the potential to fulfill the tasks we need it to as a UI while allowing us a little creativity. Additionally, windows forms provide an acceptable level of difficulty in terms of coding. We chose to use the C# language to code it because it supports a good library for windows forms. Additionally, C# will allow us to work with Visual Studio which is well supported by both SQL server, our choice for database software, and Microsoft Azure Database, our choice for database hosting.

Database Hosting

We choose Microsoft Azure Database for the project database hosting. The service is owned and maintained by Microsoft, and it offers compatibility with our chosen database software SQL Server. Maintenance jobs such as upgrades and backups are done automatically by the service provider, and it can host up to 4TB of data. It also works with the popular Visual Studio IDE which is used by all of the group members.

Access database with the following:

Server name:	css475-10.database.windows.net
Authentication:	SQL Server Authentication
Login:	css475
Pass:	475cssqaz#

You will need Microsoft SQL Server Management Studio to access.

Queries

Example queries of the database are as follows:

1. List players by age and FG percentage.
2. List divisions by wins
3. List coaches and championship wins
4. List stadiums by home team
5. List players by order of lowest FG percentage

Along with these examples, our UI also supports user input for some fields. This means that our UI also has the potential to generate a large number of queries as directed by the user. The code for this can be found in Form1.cs and SponsorQuery.cs of the submitted Visual Studio file “CSS475_Database_UI”.

Implementation of Basic Queries

Along with more complex queries, our UI also supports more basic queries that give the user the option to query any table for one, two, or all attributes of that table. By doing this, we provide the user with a number of different possible queries equivalent to all possible combinations of tables and their attributes.

Implementation of Complex Queries

- List players by FG percentage.
  Select PNAME,FG FROM PLAYER_PROFILE,PLAYER_STATS WHERE PLAYER_STATS.PLAYER_ID = PLAYER_PROFILE.PLAYER_ID ORDER BY FG DESC

- List divisions by wins
  Select divName,SUM(champWins) FROM DIVISION,TEAM WHERE d_Id = divId GROUP BY divName

- List coaches and championship wins
  Select coach_Name,c_champWins FROM COACH ORDER BY c_champWins

- List stadiums by home team
  Select team_Name,Stadium_Name FROM TEAM,STADIUM WHERE S_CityId = t_City

- Retrieve all players who have sponsor contract(s) in a team
  - Note: each row is a financial sponsorship contract term between a player and a sponsor. One player could have multiple sponsorship contracts with different sponsors.

SELECT Pname, sponsorName, amount, startDate, endDate, DATEDIFF(year, startDate, endDate) AS LENGTH
FROM PLAYER_PROFILE AS PP, PLAYER_CONTRACT AS PC, SPONSOR, SPONSOR_CONTRACT, TEAM
WHERE TEAM = team_Name AND team_Name = 'seattle sonics' AND PP.PLAYER_ID = PC.PLAYER_ID AND sponsorId = spId AND PP.PLAYER_ID = plId
ORDER BY Pname, sponsorName;

- Retrieve the total amount of sponsor values of player(s) in a team who achieved above average Field Goal percentage. This could be used as a performance metric between the player actual performance vs. financial incentives offered to such player.

SELECT Pname, FG, SUM(amount) as SP, COUNT(*) as CT

```
FROM PLAYER_PROFILE AS PP, PLAYER_CONTRACT AS PC, SPONSOR_CONTRACT,
TEAM, PLAYER_STATS AS PS
WHERE TEAM = team_Name AND team_Name = 'seattle sonics' AND PP.PLAYER_ID =
PC.PLAYER_ID AND PP.PLAYER_ID = plId AND PP.PLAYER_ID = PS.PLAYER_ID AND
endDate > GETDATE() AND FG >
                (SELECT AVG(FG)
                FROM PLAYER_PROFILE AS PP, PLAYER_STATS AS PS,
                        PLAYER_CONTRACT AS PC, TEAM
                WHERE TEAM = team_Name AND team_Name = 'seattle sonics' AND
                        PP.PLAYER_ID = PS.PLAYER_ID AND PC.PLAYER_ID =
                        PS.PLAYER_ID)
GROUP BY Pname, FG
ORDER BY SP DESC;
```

- Retrieve the top ten players with the highest amount of sponsorship in the whole NBA league. This could be used as one way to check for the current hottest players since sponsorship often favors the best performance players.

```
SELECT TOP (10) Pname, team_Name, SUM(amount) as SP,COUNT(*) as CT
FROM PLAYER_PROFILE AS PP, SPONSOR, SPONSOR_CONTRACT,
PLAYER_CONTRACT AS PC, TEAM
WHERE TEAM = team_Name AND PP.PLAYER_ID = PC.PLAYER_ID AND
                sponsorId = spId AND PP.PLAYER_ID = plId AND endDate > GETDATE()
GROUP BY Pname, team_Name
ORDER BY SP DESC;
```

- Retrieve the total amount of financial sponsor aggregated for each industry. This could be used as a market research metric to see which industries have a higher interest in advertising with NBA (maybe they have similar customer base or product).

```
SELECT sponsorBusiness, SUM(amount) AS Total
FROM SPONSOR, SPONSOR_CONTRACT
WHERE sponsorId = spId AND endDate > GETDATE()
GROUP BY sponsorBusiness
ORDER BY Total DESC;
```

Data Generation

We created a total of 202 inserts for each table to be used as testing data. Two entries for each table were created by us for easy verification of correct output during testing. The other 200 were automatically generated using Mockaroo (https://mockaroo.com/) with the purpose of testing how well our database works with a variety of different data entries and how well it scales. We processed all of this data through SQL queries in order to populate our database via inserts. Included Insert Statements subsection of the SQL Statements section are the SQL queries that we wrote ourselves.

Schedule

Week 2		Update design proposal	Meet for total of 1 hour this week
Week 3	Iteration 1	Update design proposal	Meet for total of 1 hour this week
Week 4		Tooling Assessment	Meet for total of 2 hours this week
Week 5	Iteration 2	Tooling Assessment	Meet for total of 2 hours this week
Week 6		Update version of iteration	Meet for total of 2 hours this week
Week 7		Create database	Meet for total of 2 hours this week
Week 8	Iteration 3	Create database and populate database	Meet for total of 2 hours this week
Week 9		Final report, poster, finish database if necessary	Meet for total of 2 hours this week
Week 10	Iteration 4	Final report and poster	Meet for total of 2 hours this week

Division of Work

Everyone will work on everything together, work will be done at home as necessary.

We decided to do a program that allow people to view and analyze information and statistic of both players teams of NBA. The planning phase of our project are done together. We first came up with an ER Diagram, then we added more table and attribute as the graders suggested. ER converted to RM, which then ported to SQL Server Management Studio. Lastly we populated the database, and make the front end of the program.

Peter Pitojo

- Planning phase
- Helped construct the first ER diagram
- Helped converting ER diagram into RM
- Improve existing table
- Create some of the table in SSMS

Peter Van:

I created everything for Application and Motivation on our poster. Did about a fourth to a third of the work when it came to inserting data into the database and creating the database. Specifically working on tables that had to do with the player.

Tien Huynh:

- Set up the Azure SQL Server, created and normalized the SPONSOR, SPONSOR_CONTRACT, DIVISION table
- Inserted data for DIVISION, PLAYER_CONTRACT, SPONSOR, SPONSOR_CONTRACT, TEAM
- Created four SQL queries for sponsorship and implemented the sub-UI for such queries
- Created the Milestones part of the poster

Alex Carswell:

- Put board together using everyone's content and created poster content for Methodology
- Integrated everyone's UI code and created the homepage of the UI (the form that appears on launch) which supports basic user queries for all tables and attributes
- Created and normalized Team, Coach, Owner, and Team_Worth tables
- Generated and inserted data for Coach, Owner, and Team_Worth tables
- Helped create Design Document

Project Evaluation

By the end of this project we had learned a great deal, and ended up producing a final result that we are all very proud of. As a result, there are a few areas that we had to come back to as we learned more in order to fix for the final submission.

Success

We had an easy time with creating the ER and RM diagrams in the iterations. Everything that we had learned in class ended up being pretty easy to do e.g creating the database. Our communication as a team was pretty good as well.

Issues

There were some areas of the project that surprised us with problems we didn't plan for. One example occured when we were inserting the data. Some exceptions got triggered e.g foreign key constraints and the fact that our random insert generator didn't have the exact categories for all our attributes.

Critiques

Content Critique

After working on this iteration, we found that the initial content we came up with in Iteration 0 was not enough to build a good sized database with. Therefore, we have created several more tables in order to bolster the scale of our database. These tables are Division, Sponsor, Sponsor Contract, and Stadium.

Entity Relationship Diagram Critique

After reevaluating the previous model, we decided to add a few relationships to create an appropriate project's scope.

- Each team belongs to a Division and a Conference.
- Each player has a general profile (age, height, weight, etc.) and separate statistics (field goal percentage, 3-point percentage, etc.).
- Each player has a contract with a team specifying start-end date, salary.
- There are financial sponsors for certain players specifying through contracts which contains the sponsor and player identity, sponsor amount, start-end date.
- Stadium is now a separate entity that contains its name, location, seat count, and built year.

Relational Data Model Critique

As a result of working on this iteration, we found that some of our current tables had far too many attributes and were not in normal form. We therefore set about normalizing all of our tables and ended up splitting some into multiple tables. The new tables we created through the normalization process were Division, Sponsor, Sponsor_Contract, Team_Worth, Player_Contract, Player_Stats, and Player_Profile. In addition, we also removed some redundant attributes from several tables that were not functionally dependant on their respective keys. We now have a total of 12 tables instead of the initial 5 we started with.