Soccer Database Project

Table of Contents

[Introduction 3](#_Toc499585346)

[Database Solution Outline 3](#_Toc499585347)

[Soccer rules 4](#_Toc499585348)

[Key Data Entity Definitions 5](#_Toc499585349)

[Player 5](#_Toc499585350)

[Team 5](#_Toc499585351)

[PlayerMatch 5](#_Toc499585352)

[Team\_league\_season 5](#_Toc499585353)

[League 5](#_Toc499585354)

[Season 5](#_Toc499585355)

[Stadium 5](#_Toc499585356)

[Key Data Issues 5](#_Toc499585357)

[Logical Data Model 6](#_Toc499585358)

[Table Design and Implementation 7](#_Toc499585359)

[Data Integration: Extract, Load, and Transformation 7](#_Toc499585360)

[Data extraction 7](#_Toc499585361)

[Data load and transformation 9](#_Toc499585362)

[DATA ANALYSIS 11](#_Toc499585363)

[Structured data 11](#_Toc499585364)

[Unstructured Data 17](#_Toc499585365)

[Mongo DB: Unstructured Data analysis 19](#_Toc499585366)

[DATA SUMMARY CONCLUSION 21](#_Toc499585367)

[APPENDIX A – Table creation script 29](#_Toc499585368)

[APPENDIX B – DDL and Load SQL 29](#_Toc499585369)

[APPENDIX C – Stage Table Model 29](#_Toc499585370)

[APPENDIX D – Run Log 29](#_Toc499585371)

[APPENDIX E – Results By County Output 29](#_Toc499585372)

[APPENDIX F – County Growth Model Output 29](#_Toc499585373)

# Introduction

**Soccer**, is a team sport played between two teams of eleven players with a spherical ball. It is played by 250 million players in over 200 countries and dependencies, making it the world's most popular sport.

Each team consists of a maximum of eleven players (excluding substitutes), one of whom must be the goalkeeper. The game is played on a rectangular field with a goal at each end. The object of the game is to score by getting the ball into the opposing goal.

Our project aims to create a database which can be utilized to predict outcomes, entertaining story reporting statistics that enhance the viewers experience of soccer matches and gathering valuable insights to make relevant comparisons by taking into account player data, match data and corresponding data from soccer leagues. Data utilized for this project will be from the past two years. The leagues for which data is mined are from English Premier League and La-Liga.

# Database Solution Outline

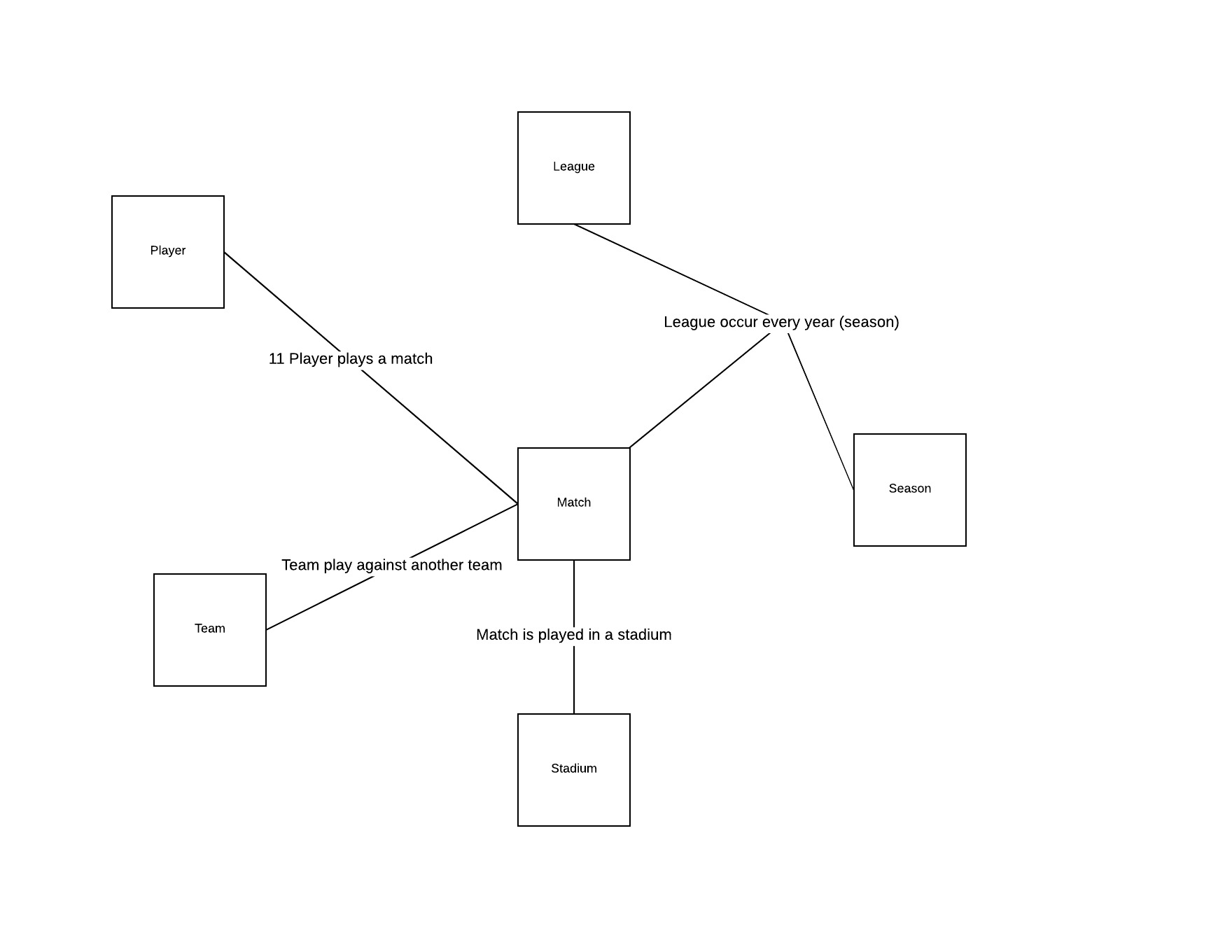
* Data Sources:
  + [ESPN FC Census](http://www.espn.com/soccer) - Match data was extracted from this website.
  + [worldfootball net api](http://football-api.com/plans/world-football/) - Player attributes were web scraped from this website.
  + [Twitter](http://www.twitter.com) - Reviews of users on Team to get sentiment analysis
* Data extraction:
  + Data extracted from source HTML content using python libraries 'BeautfulSoup4' and 'getlibrary'
  + User reviews on teams from twitter.
* Model creation steps:
  + Conceptualized and created Crow foot ER diagram considering entities and attributes of soccer
  + Creation of ID for each attribute in various tables
  + Normalize the data to 3NF
* Database integration and setup:
  + MySQL 6.3 used for creation of SQL tables and data upload
* Data analysis:
  + Data analyzed by creating SQL queries and visualized using Tableau for representation
  + Sentiment analysis of team user review using Semantria exel add-on.

# Soccer rules

A league consists of 20 teams. Each team faces every other team in the league twice, one on the home ground and other on away ground. In total, there are 380 matches in a league. League happens every year called seasons. We have considered two seasons 2015-16 and 2016-17 for our database design. Each match which is played by home team and away team consists of 11 player each with some substitutes.

Player are specialized as attacker, defender or midfielder. Each team tries to maintain the composition of all three specialties. Player help in scoring goals, assisting or saving a goal. They also get penalized for fouls committed and get yellow/red card based on severity of fouls.

When a match is played between teams the winner is decided based on number of goals on another team. The team which scores more number of gaols wins the game and in case of equal number of goals scored, the match is considered as a draw.

A stadium where a match is played is home ground for a club. Each stadium has a different seating capacity.

## Key Data Entity Definitions

### Player

Player table consist of attributes of player; Name, Date of birth, Height, Position, Nationality, Weight.

Data of 2189 players are captured.

### Team

Team table consist of all the 46-team that played in two leagues for past two seasons

### PlayerMatch

This table consist of the record of player performance in each match, which including substitutues. Attributes include: GoalsSaves, TotalGoals, TotalShots, ShotsonTarget, fouls committed, and number of cards due to fouls.

### Team\_league\_season

This table represent team performance in a league for a particular season. Attributes captured are Team

Rank in season.

### League

Table consist of list of leagues. We have considered English premier league and La-Liga for our project.

### Season

League occurs every season. Data of past two past seasons 2015-16 and 2016-17 are extracted.

### Stadium

Every team has a home ground. Each ground host many matches. We have data of 1331 stadiums with their attributes: city, club, country and name.

## Key Data Issues

* Data collected from various data sources and they had variation in format and nomenclature. For example, player names are not uniform throughout sources. So, creation of unique ID and mapping was a challenge. We tried to make them uniform using excel and python.
* During the bidding process, a club of a player might be changed. We do not have the transfer data so we assumed that a player's team is the same in the match in which the player last played.
* Rank of a team is not derived only from point and different leagues have different rules regarding rank of a team, this lead us to create a new entity team league season in our database.

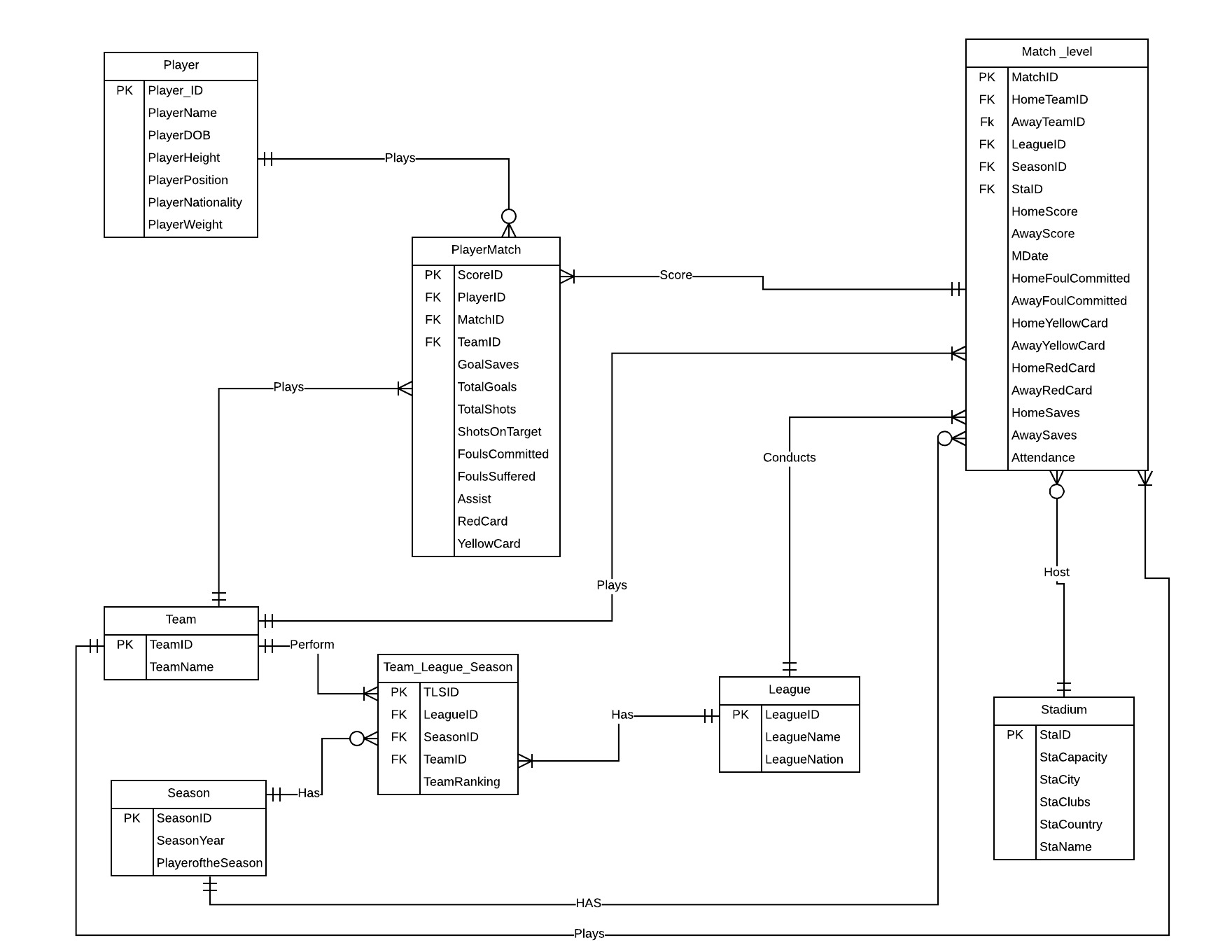
# Logical Data Model

The following is the ER diagram for our soccer database. Relations are described using text in connection lines.

Mathc\_level, PlayerMatch and team league season are ternary relationship. Primary keys are mentioned as PK and foreign key as FK. We have taken some assumption for this study:

Rules and assumptions:

* A player plays many matches and each match has many players.. Some player might not play the match(Assumption)
* Team plays a match in a league and for various seasons. Team plays against another team. One team is home team, and another called away team.
* Team can play in multiple league and multiple season.
* Stadium can host multiple matches, but a match is only played in one stadium. Some stadium will not have any match(Assumption)
* Each season has many matches but a match can be part of only one season.



# Table Design and Implementation

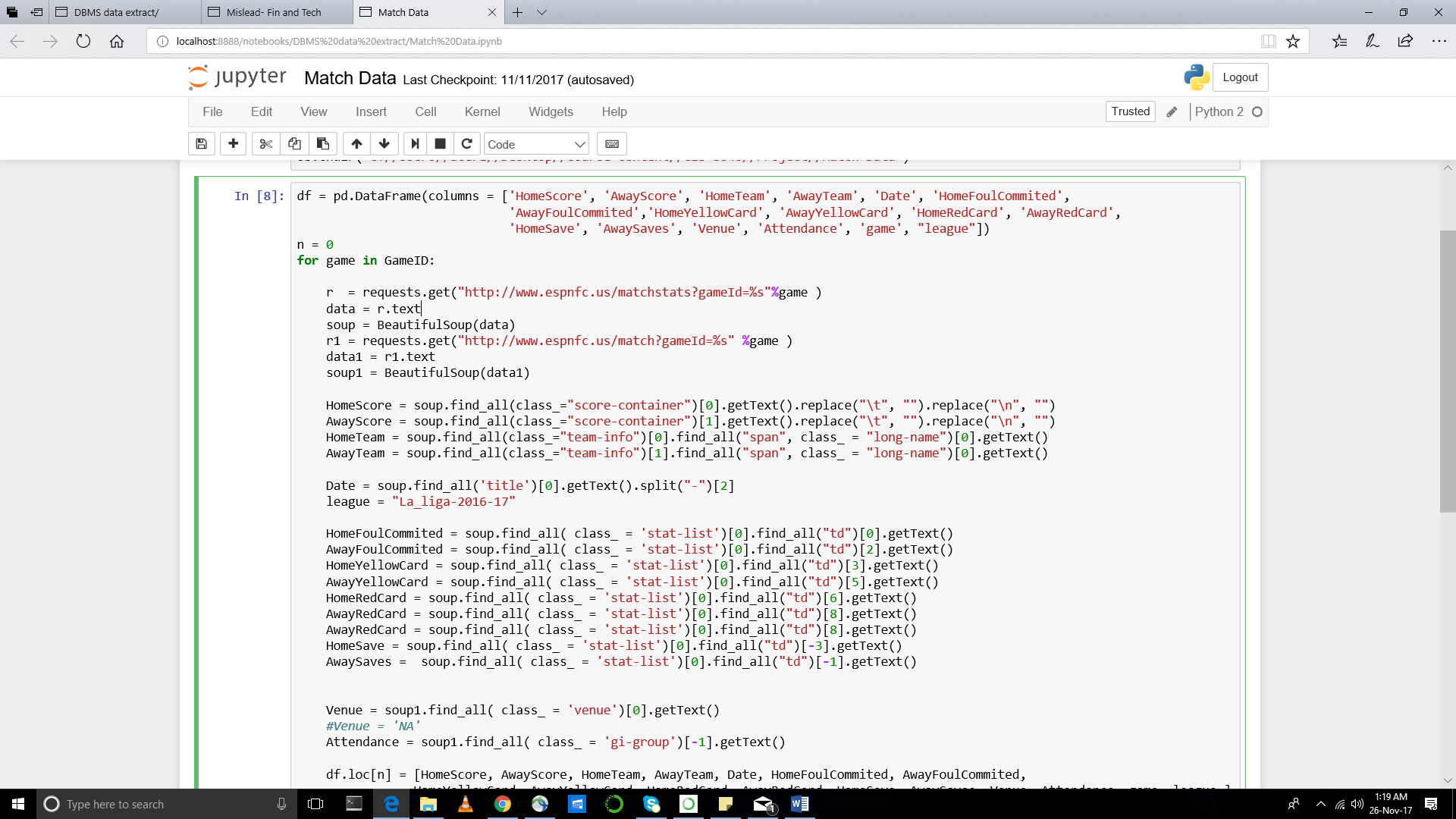
Initially the data was loaded into the table from sources without normalization. All the primary keys were identified. Then we removed all the partial and transitive dependencies in the data in order to convert our data into 3NF form. Also, checks were made to remove all the data redundancies and the corresponding columns were adjusted. For associative entities, we created a single new primary key attribute to replace a composite primary key, in order to efficiently retrieve the data from tables.

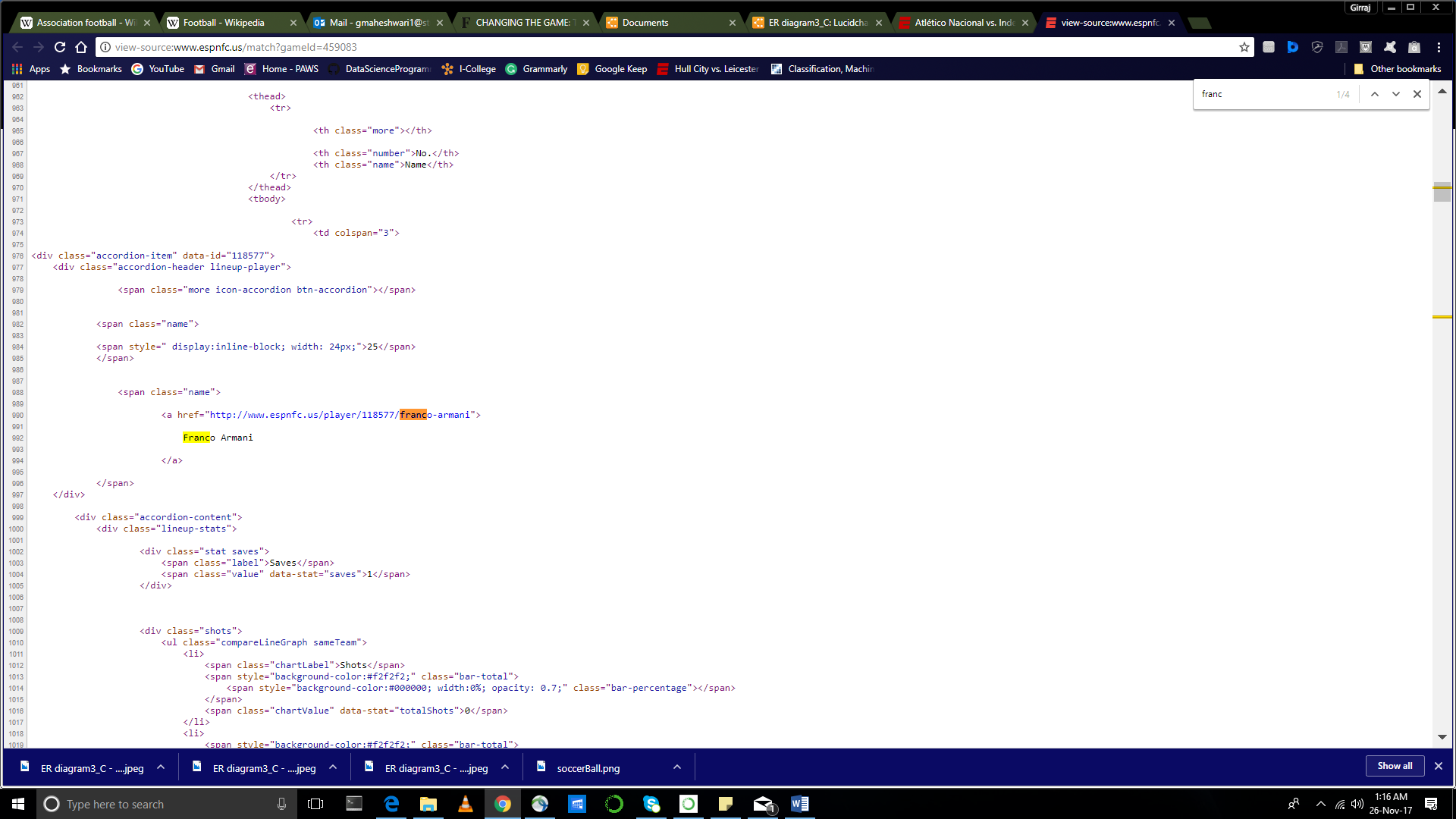
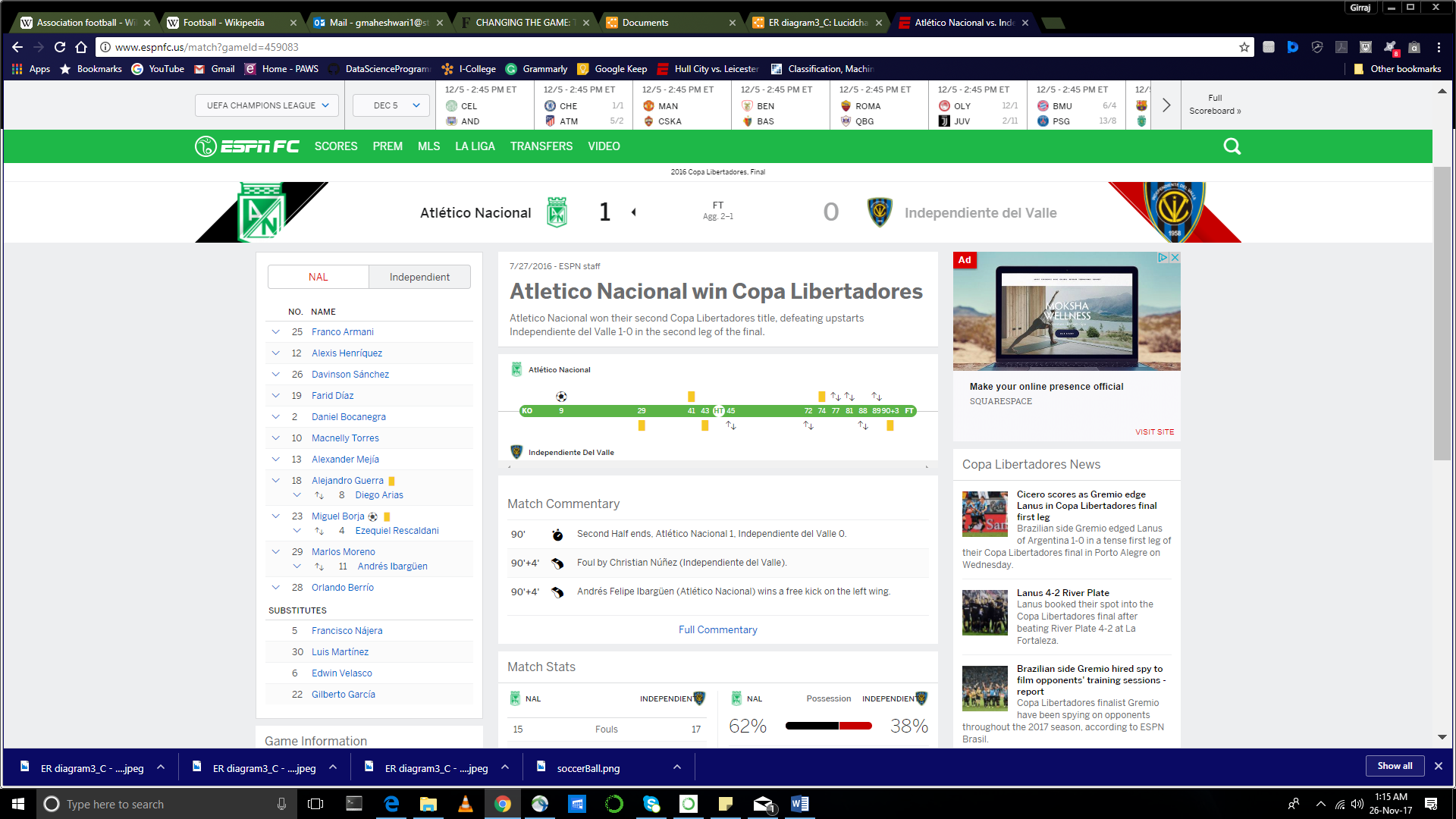
# Data Integration: Extract, Load, and Transformation

### Data extraction

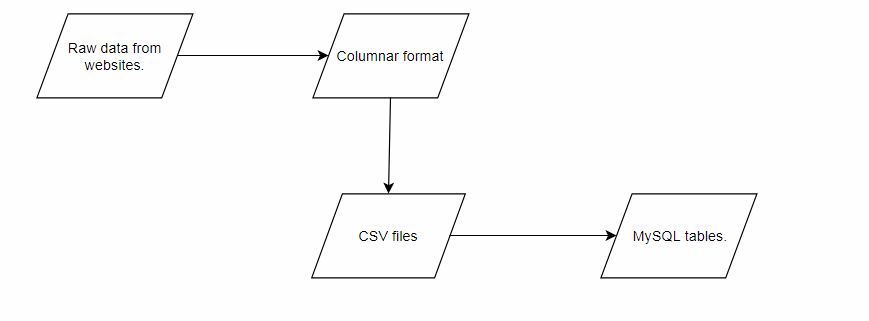
* Data was extracted using python re packages from the source websites (ref: Database source outline)
* HTML source code has been extracted and wrote python query to extract the same in structured format
* Once data is extracted, name of the tables and attributes are labeled. Created unique ID for attributes in every table. Unique ID assigned for composite primary keys.

Sample Code to extract the data from HTML source:





### Data load and transformation



* All of the data was scraped from website sources as mentioned above. Python scripts were utilized to pull data and pre-process data.
* The data was then transformed into columnar format by using Microsoft Excel.
* In this process while in excel format, all primary key values and foreign key values were inserted into the columns. Also date formats were adjusted to match SQL DATE ‘YYYY-MM-DD’ format.
* Since the number of rows were large, it would be a tedious task to write insert queries for each row. Hence the data has was converted into a comma delimited (.csv) file format.
* Database was created in MySQL using DDL scripts (Appendix A). Below are some sample statements for creating tables and defining relationships between tables.

CREATE TABLE `league` (

`LeagueID` varchar(15) NOT NULL,

`LeagueName` varchar(255) DEFAULT NULL,

`LeagueNation` varchar(255) DEFAULT NULL,

PRIMARY KEY (`LeagueID`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `team\_league\_season` (

`TLSID` varchar(15) NOT NULL,

`LeagueID` varchar(15) DEFAULT NULL,

`SeasonID` varchar(15) DEFAULT NULL,

`TEAMID` varchar(15) DEFAULT NULL,

`TeamRanking` varchar(15) DEFAULT NULL,

PRIMARY KEY (`TLSID`),

KEY `fk\_TLSteamid` (`TEAMID`),

KEY `fk\_TLSleagueid` (`LeagueID`),

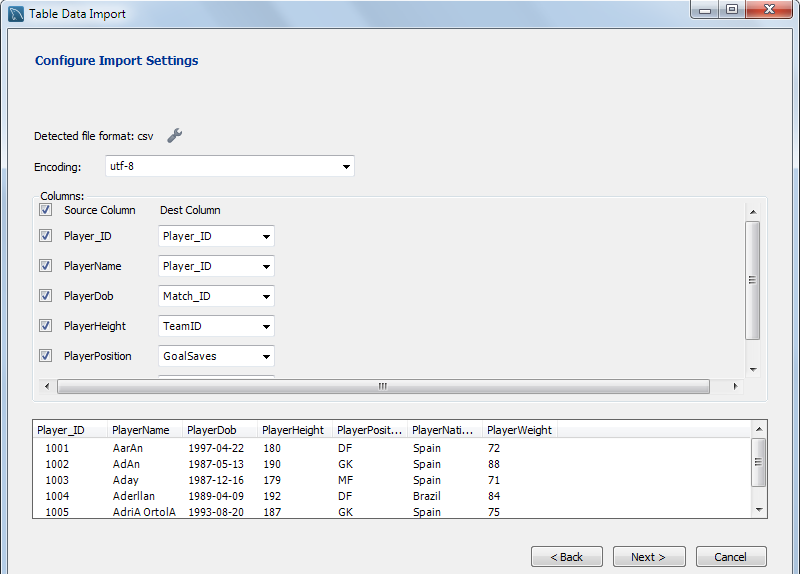
KEY `fk\_TLSseasonID` (`SeasonID`),

CONSTRAINT `fk\_TLSleagueid` FOREIGN KEY (`LeagueID`) REFERENCES `league` (`LeagueID`),

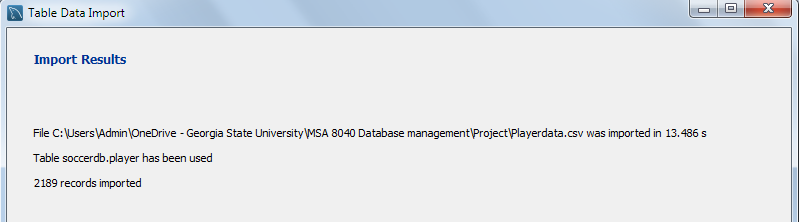
CONSTRAINT `fk\_TLSseasonID` FOREIGN KEY (`SeasonID`) REFERENCES `season` (`SeasonID`),

CONSTRAINT `fk\_TLSteamid` FOREIGN KEY (`TEAMID`) REFERENCES `team` (`TEAMID`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

* MySQL workbench’s data import wizard utility provides a simple solution to the data loading process. The source .csv file is selected as per the table which is to be loaded. The wizard reads the comma delimited file and automatically recognizes columns corresponding to the table.  
    
  

After making a few settings changes, the load process is quickly inserts data into each table.



# DATA ANALYSIS

This section contains a series of data analysis queries mean to explore the soccer database and point out some interesting findings. \*\*\*Please check\*\*\* In addition, at the end of the section, we first give an example of the difference in writing the SQL against the raw staging data and final data model.

In the next section we will show the data conclusion from these queries

### Structured data

1. **Points Table: Team Standings in different leagues in different seasons-**  This table will give a glimpse of overall season with each team's performance. Team's ranking with key performance parameters will be shown when you put the season and league's name in the global variable.
   1. **Code**

set @league := 'English Prem League';

set @season := '2015-16';

select \*,(3\*WIN+Draw) as Points

from

(

select M.teamname,M.seasonyear,M.leaguename,

home\_matches+away\_matches as Matches,

Home\_wins+away\_wins as WIN,

Home\_loss+away\_loss as Loss,

Home\_draw+away\_draw as Draw,

home\_goals\_scored+away\_goals\_scored as Goal\_Scored,

home\_goals\_conceded+away\_goals\_conceded as Goal\_conceded,

(home\_goals\_scored+away\_goals\_scored)-(home\_goals\_conceded+away\_goals\_conceded) as Goal\_Difference

from

(

select a.teamname,c.seasonyear,d.leaguename,

count(distinct b.match\_id) as home\_matches,

count(distinct case when homescore>awayscore then b.match\_id else null end) as Home\_wins,

count(distinct case when homescore<awayscore then b.match\_id else null end) as Home\_loss,

count(distinct case when homescore=awayscore then b.match\_id else null end) as Home\_draw,

sum(HomeScore) as home\_goals\_scored,

sum(awayScore) as home\_goals\_conceded,

sum(HomeScore)-sum(awayScore) as home\_goal\_difference

from team as a

left join match\_level as b

on a.TEAMID=b.HomeTeamID

left join season as c

on b.seasonid=c.SeasonID

left join league as d

on b.leagueid=d.LeagueID

where seasonyear= @season

and LeagueName= @league

group by 1,2

order by 5 desc) as M

left join

(

select a.teamname,c.seasonyear,d.leaguename,

count(distinct b.match\_id) as away\_matches,

count(distinct case when awayscore>homescore then b.match\_id else null end) as Away\_wins,

count(distinct case when awayscore<homescore then b.match\_id else null end) as Away\_loss,

count(distinct case when awayscore=homescore then b.match\_id else null end) as Away\_draw,

sum(AwayScore) as away\_goals\_scored,

sum(homeScore) as away\_goals\_conceded,

sum(AwayScore)-sum(homeScore) as away\_goal\_difference

from team as a

left join match\_level as b

on a.TEAMID=b.AwayTeamID

left join season as c

on b.seasonid=c.SeasonID

left join league as d

on b.leagueid=d.LeagueID

where seasonyear= @season

and LeagueName= @league

group by 1,2

order by 5 desc) as N

on M.teamname=N.teamname

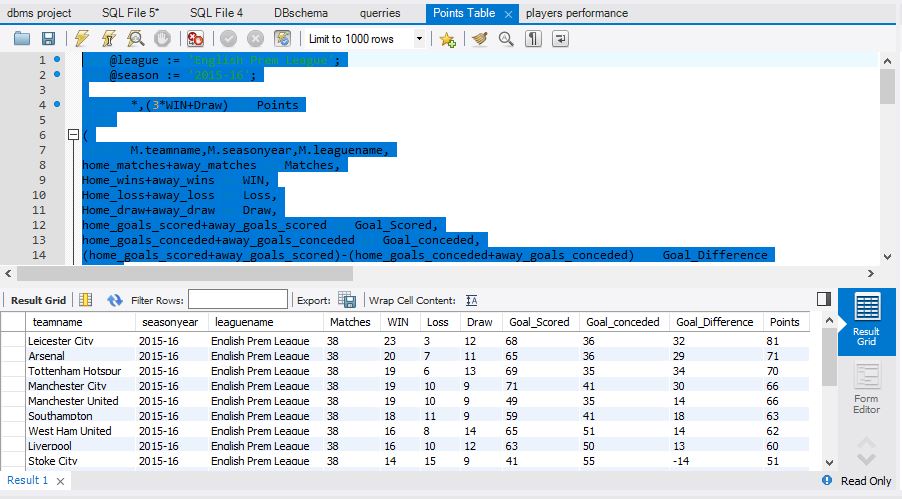
and M.seasonyear=N.seasonyear

and M.leaguename=N.leaguename

) as P

order by points desc

* 1. **Result**



* 1. **Interpretation**

**This query joins several table and gives us the final standings of the teams in a particular league in a particular season. We just have to enter the league name and season at the top in global variable declaration. It also presents a complete picture of a team in that season with variables like - No. of Wins, loss, points, goal scored & goals conceded etc.**

**Table Used:** team, match\_level, season, league

1. **Player Statistics: This code will give the key player's performance parameter which will help in judging a player's ability.**
   1. **Code**

**select \*,**

**round(Goals\_2015\_16\*100/shots\_2015\_16,2) as accuracy\_2015\_16,**

**round(Goals\_2016\_17\*100/shots\_2016\_17,2) as accuracy\_2016\_17**

**from(**

**select playername,**

**sum(totalgoals) as TotalGoals,**

**sum(case when seasonyear='2015-16' then totalgoals else null end) as Goals\_2015\_16,**

**sum(case when seasonyear='2016-17' then totalgoals else null end) as Goals\_2016\_17,**

**sum(case when seasonyear='2015-16' then assist else null end) as assist\_2015\_16,**

**sum(case when seasonyear='2016-17' then assist else null end) as assist\_2016\_17,**

**sum(case when seasonyear='2015-16' then totalshots else null end) as shots\_2015\_16,**

**sum(case when seasonyear='2016-17' then totalshots else null end) as shots\_2016\_17**

**from (**

**select a.scoreid,a.player\_id,d.playername,a.match\_id,a.totalshots,a.totalgoals,a.assist,**

**b.seasonid,c.seasonyear,c.playeroftheseason,b.hometeamid,b.awayteamid**

**from playermatch a**

**left join match\_level b**

**on a.Match\_ID=b.match\_ID**

**left join season as c**

**on b.seasonID=c.SeasonID**

**left join player as d**

**on a.Player\_ID=d.Player\_ID**

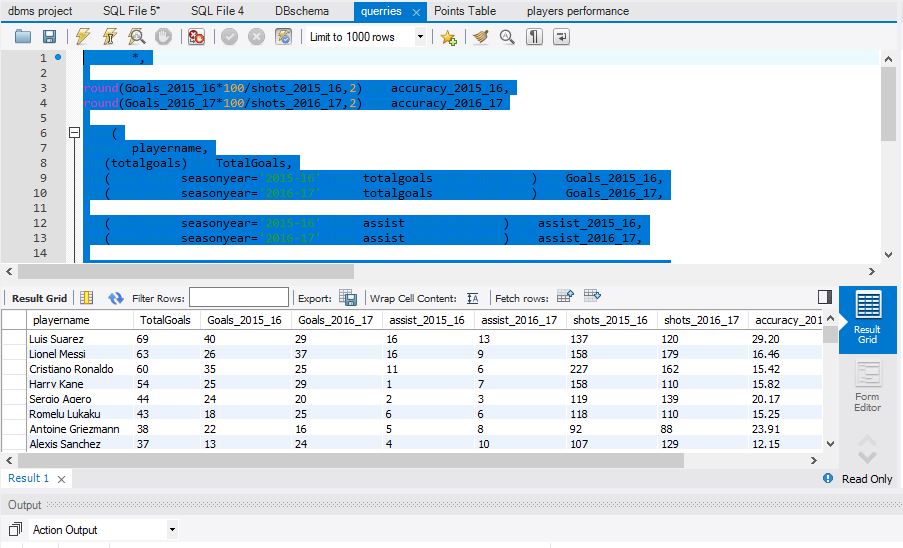
**group by a.scoreid,a.player\_id,d.playername,a.match\_id,a.totalshots,a.totalgoals,a.assist,**

**b.seasonid,c.seasonyear,c.playeroftheseason,b.hometeamid,b.awayteamid) as M**

**group by 1**

**order by TotalGoals desc) as N**

* 1. **Results**



* 1. **Interpretations**

**This query gives every player’s complete analysis over the years. It includes goal scored, assists made, shots taken, accuracy etc in different season. By this we can compare player’s performance with respect to others as well as his own performance over the years. Luis Surez, FC Barcelona striker is the most lethal player in past two years as far as scoring goals is concerned.**

1. **Stadiums: This will help us in finding the key attributes of a stadium and let us know interesting facts like which is the biggest stadium in Europe, which stadium is mostly fuly filled and in which European city it is situated etc.**
   1. **Code**

**select a.staid,a.staname,a.stacapacity,**

**a.staCountry ,a.StaClubs,**

**round(avg(b.attendance),0) as attendance**

**from stadium as a**

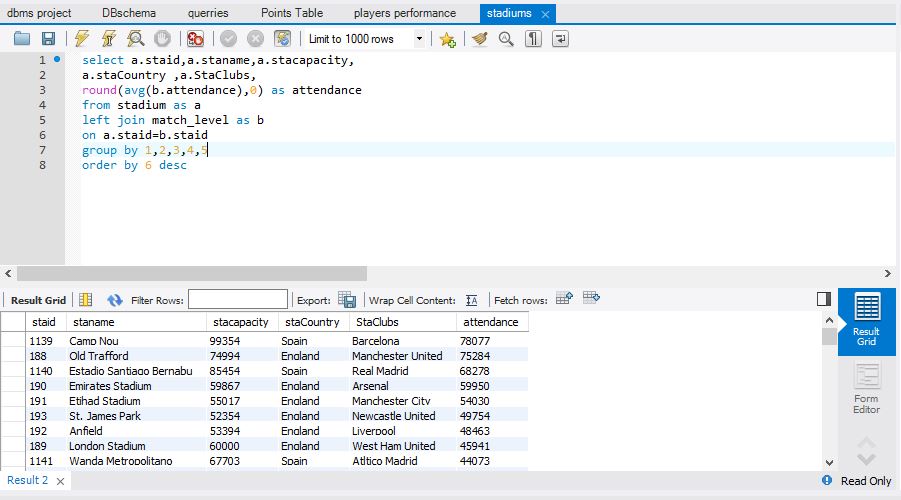
**left join match\_level as b**

**on a.staid=b.staid**

**group by 1,2,3,4,5**

**order by 6 desc**

* 1. **Results**



* 1. **Interpretation**

**This table gives us the avg no. of attendance of all the games in a stadium and thus compares the fan base of different teams. Camp Nou situated in Barcelona is the biggest stadium in Europe with a capacity of nearly 100,000 and has an average attendance of 78,000**

1. **Goalkeeper saves: This will help us in comparing goalkeeper's performance over the full season. Key goalkeeper's attributes are very different than that of an attacker which is why we have analyzed it in this separate table.**
   1. **Code**

**select playername,**

**sum(GoalSaves) as TotalGoalsaves**

**from (**

**select d.playername,a.match\_id,a.totalshots,a.totalgoals,a.assist,**

**b.seasonid,c.seasonyear,b.hometeamid,b.awayteamid,a.GoalSaves,b.HomeScore,b.AwayScore**

**from playermatch a**

**left join match\_level b**

**on a.Match\_ID=b.match\_ID**

**left join season as c**

**on b.seasonID=c.SeasonID**

**left join player as d**

**on a.Player\_ID=d.Player\_ID**

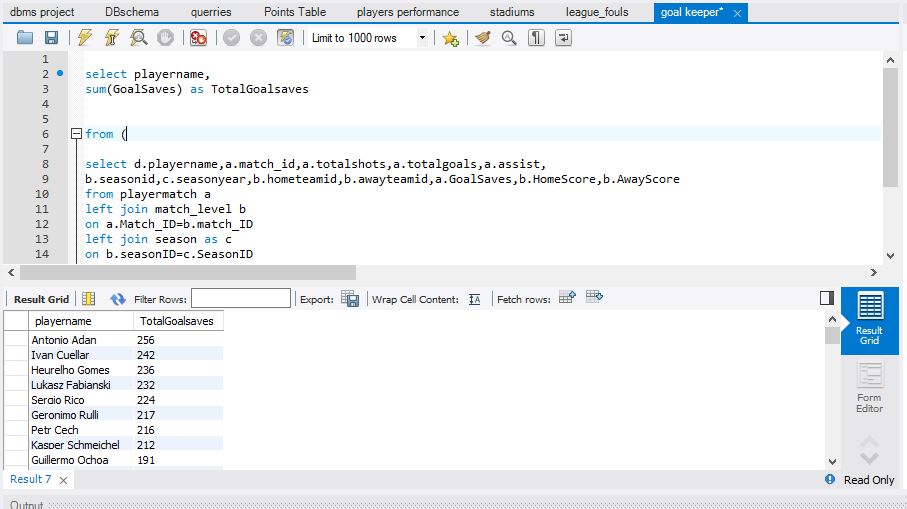
**group by d.playername,a.match\_id,a.totalshots,a.totalgoals,a.assist,**

**b.seasonid,c.seasonyear,b.hometeamid,b.awayteamid,a.GoalSaves,b.HomeScore,b.AwayScore) as M**

**group by 1**

**order by TotalGoalsaves desc**

* 1. **Results**



* 1. **Interpretation**

**This query compares the expertise of a goalkeeper across both the season**

1. **Ballon’dor Winner: This table will give us the name of the best player of the season in Europe and amount of goals he has scored and assists that he has made etc.**
   1. **Code**

select A.seasonyear,A.Ballondor\_winner,B.matches,B.goals,B.assists

from

(

select a.SeasonID, a.seasonyear,b.Player\_ID,b.PlayerName as Ballondor\_winner

from season as a

left join

player as b

on a.Playeroftheseason=b.Player\_ID

group by 1,2,3,4) as A

left join

(

select a.player\_id,b.seasonID,

count(distinct a.Match\_ID) as matches,

sum(TotalGoals) as goals,

sum(Assist) as assists

from playermatch as a

left join match\_level as b

on a.Match\_ID=b.match\_ID

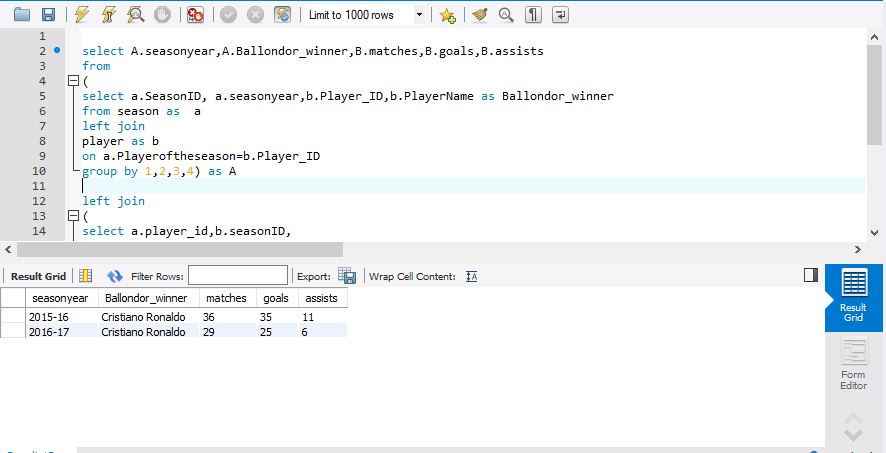
group by 1,2) As B

on A.SeasonID=B.SeasonID

and A.Player\_ID=B.Player\_ID

where seasonyear in ('2015-16','2016-17')

* 1. **Result**



* 1. **Interpretation**

**This query shows the performance of the Europe’s best player in respective season**

1. **League Comparison: Different leagues have different intensity of a game. This table will give the no. Fouls commited, red cards,yellow cards etc in different leagues in different seasons. We can judge which league is more tough than the other.**
   1. **Code**

**select b.leaguename ,c.seasonyear,sum(HomeFoulCommited)+sum(AwayFoulCommited) as foul\_commited,**

**sum(a.HomeRedCard)+sum(AwayRedCard) as Red\_Card,**

**sum(a.HomeYellowCard)+sum(AwayYellowCard) as Yellow\_Card**

**from match\_level as a**

**left join league as b**

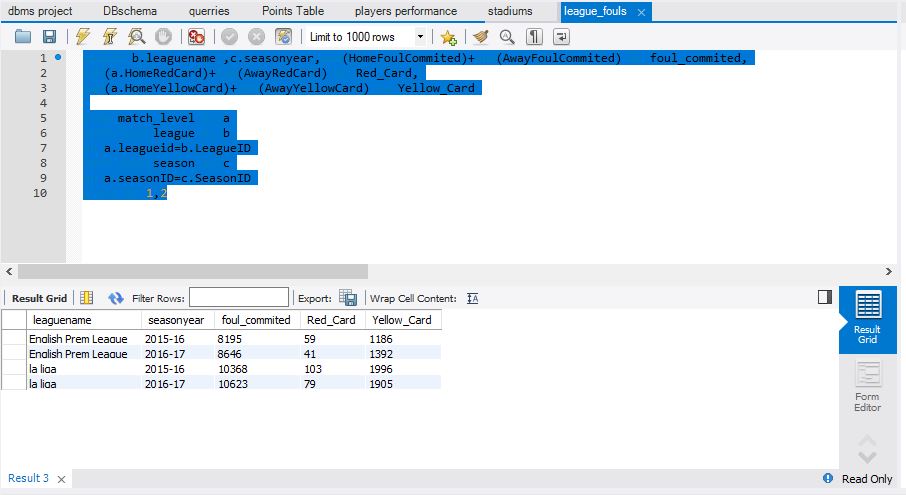
**on a.leagueid=b.LeagueID**

**left join season as c**

**on a.seasonID=c.SeasonID**

**group by 1,2**

* 1. **Results**



* 1. **Interpretation**

**This query clearly shows that La Liga is rasher as compared to EPL as no. of fouls committed, red cards yellow cards is way more in la liga in both the seasons as compared to EPL.**

### Unstructured Data

1. **Sentiment Analysis: Using Python**

Data: Twitter data which consist user review on teams. Sentiments are analysed using below python code

* 1. **Code**

**import nltk.classify.util  
from nltk.classify import NaiveBayesClassifier  
from nltk.corpus import names  
  
  
def word\_feats(words):  
    return dict([(word, True) for word in words])  
  
positive\_vocab = [ 'awesome', 'outstanding', 'fantastic', 'terrific', 'good', 'nice', 'great']  
negative\_vocab = [ 'bad', 'terrible','useless', 'hate']  
neutral\_vocab = [ 'the','sound','was','is','actors','did','know','words','not']  
   
positive\_features = [(word\_feats(pos), 'pos') for pos in positive\_vocab]  
negative\_features = [(word\_feats(neg), 'neg') for neg in negative\_vocab]  
neutral\_features = [(word\_feats(neu), 'neu') for neu in neutral\_vocab]  
   
train\_set = negative\_features + positive\_features + neutral\_features  
   
classifier = NaiveBayesClassifier.train(train\_set)   
  
 # Predict  
  
def sentiment(sentence,typ):  
    neg = 0  
    pos = 0  
    net = 0  
    #sentence = "Awesome movie, I liked it"  
    sentence = sentence.lower()  
    words = sentence.split(' ')  
    for word in words:  
        classResult = classifier.classify( word\_feats(word))  
        if classResult == 'neg':  
            neg = neg + 1  
        if classResult == 'pos':  
            pos = pos + 1  
        if classResult == 'neu':  
            net = net + 1  
    poy = float(pos)/len(words)  
    noy = float(neg)/len(words)  
    noty = float(net)/len(words)  
    if typ == 'pos':  
        return poy  
    if typ == 'noy':  
        return noy  
    if typ == 'noty':  
        return noty  
      
dff1.loc[:,"positive"] = dfff['Text'].apply(lambda x: sentiment(x,'poy'))  
dff1.loc[:,"negative"] = dfff['Text'].apply(lambda x: sentiment(x,'noy'))  
dff1.loc[:,"neutral"] = dfff['Text'].apply(lambda x: sentiment(x,'noty'))  
dff1.to\_excel('/Users/sidkapoor/Desktop/twit/sentiments.xlsx')**

* 1. **Results:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Teams | Negative | Neutral | Positive | Grand Total | Neg% | Net% | Pos% |
| Arsenal | 5 | 56 | 763 | 824 | 1% | 7% | 93% |
| Atletico | 1 | 3 | 66 | 70 | 1% | 4% | 94% |
| Chelsea | 79 | 70 | 704 | 853 | 9% | 8% | 83% |
| Espanyol |  | 3 | 17 | 20 | 0% | 15% | 85% |
| Fcbarcelona | 2 | 8 | 204 | 214 | 1% | 4% | 95% |
| LeicesterCity |  | 10 | 158 | 168 | 0% | 6% | 94% |
| MachesterCity | 1 | 1 | 179 | 181 | 1% | 1% | 99% |
| ManchesterUnited | 2 | 5 | 274 | 281 | 1% | 2% | 98% |
| RM | 17 | 29 | 269 | 315 | 5% | 9% | 85% |
| **Grand Total** | **107** | **185** | **2634** | **2926** | **4%** | **6%** | **90%** |

* 1. **Interpretations:**

**Most negative reviews: Chelsea followed Real Madrid**

**Most positive review: Manchester city followed MU followed**

#### **This interpretation similar to the current rankings in EPL.**

#### **Real Madrid not performing good in recent times which is reflected in above table**

1. **Sentiment Analysis: Using Semantria**

Following analysis done on semantria using 1000 tweets

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Count of Entity Mentions Sentiment +/-** | **Entity Mentions Sentiment +/-** |  |  |  |
| **Entity** | **negative** | **neutral** | **positive** | **Grand Total** |
| Wenger | 1 | 44 | 1 | 46 |
| Mugabe |  | 31 |  | 31 |
| Arsene Wenger | 4 | 24 | 1 | 29 |
| manager | 1 | 5 | 10 | 16 |
| Tottenham |  | 14 |  | 14 |
| RT Portman Road |  | 12 |  | 12 |
| Jack Wilshere | 1 | 11 |  | 12 |
| Chelsea | 1 | 10 |  | 11 |
| Leon Goretzka |  | 11 |  | 11 |
| Barcelona | 1 | 9 |  | 10 |
| Arsenal |  | 10 |  | 10 |
| **Grand Total** | **9** | **181** | **12** | **202** |

1. **News Report coverage**

Data: News report of every 760-match extracted from ESPN.

Analysis platform: **Python**

The following word cloud show how much media attention/ or influence of a player in the match.

Credit: Word it out (<https://worditout.com>)

Here is one sample news report of a match between Barcelona and Granada

“Luis Suarez fires Barcelona past Granada to secure La Liga title”



As indicated by the cloud that Messi, Neymar, Suarez comes in the news report more frequently.

### Mongo DB: Unstructured Data analysis

Below are 10 MongoDB queries with result and explanation

Data Set: Match report data of all 760 matches

Query1

Code - db.getCollection('match\_comments').find({'LeagueID': 'EPL'}).count()

Result – 760

Explanation - In this, we are finding the number of matches played in La Liga league for two seasons. (2015-16, 2016-17)

Query2

Code - db.getCollection('match\_comments').find({'LeagueID': 'LaLiga'}).count()

Result – 760

Explanation - In this, we are finding the number of matches played in EPL for the two seasons.

(2015-16, 2016-17)

Query3

Code - db.getCollection('match\_comments').find({ $where: "this.HomeScore == this.AwayScore", 'LeagueID':'EPL'}).count()

Result – 191

Explanation - We find the total number of matches whose result was a draw and were played in EPL league. We see that 25% of the total matches ended up in a draw under the EPL League for two seasons.

Query4

Code - db.getCollection('match\_comments').find({ $where: "this.HomeScore == this.AwayScore", 'LeagueID':'LaLiga'}).count()

Result – 181

Explanation - We find the total number of matches whose result was a draw and were played in La Liga league. We see that 23% of the total matches ended up in a draw under the La Liga League for two seasons.

Query5

Code - db.getCollection('match\_comments').find({ $where: "this.HomeScore > this.AwayScore", 'LeagueID':'LaLiga'}).count()

Result – 364

Explanation - Here we find the number of Home teams which have won a match in La Liga league. Almost 47% of the home teams won the match in their home ground.

Query6

Code - db.getCollection('match\_comments').find({ $where: "this.HomeScore > this.AwayScore", 'LeagueID':'EPL'}).count()

Result – 344

Explanation - Here we find the number of Home teams which have won a match in EPL league. Almost 45% of the home teams won the match in their home ground.

Also taking the draw data into consideration, we may conclude that home teams always have a better chance of winning than the away teams. So next time while betting, we must definitely try our luck on the home teams!!!

Query7

Code - db.match\_comments.find( { LeagueID: 'LaLiga',Comments: { $regex: ".\*Barcelona\*" } } ).count()

Result – 78

Explanation - Here, we see that in our report data, there are around 78 reports about team Barcelona.

Query8

Code - db.match\_comments.find( { Comments: { $regex: ".\*Rafinha.\*" } } )

Result – 1

Explanation - Out of those 78 reports, there is only one report about Rafinha.

Query9

Code - db.match\_comments.find( { Comments: { $regex: ".\*Neymar.\*" } } )

Result – 15

Explanation - And there are 15 reports about Neymar.

Query10

Code - db.match\_comments.find( { Comments: { $regex: ".\*Messi\*" } } )

Result – 39

Explanation - Whereas, we have around 39 reports for Messi, which is 50% of all the total reports of Barcelona team. This makes it obvious that Messi is a very efficient player and is on the headlines of most of the reports.

# DATA SUMMARY CONCLUSION

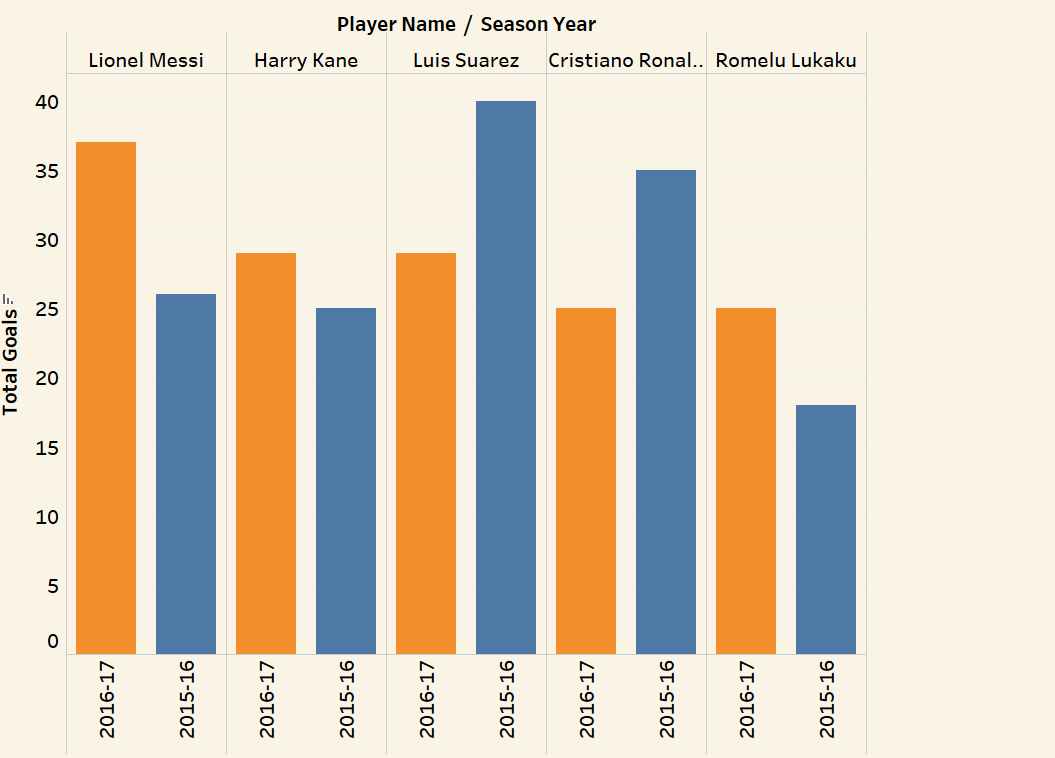
In this section we will represent the conclusion we got from the data and SQL queries.

Tool used of creation of below charts: **Tableau**

We pulled out the *data from above SQL querie*s and plot them on Tableau for Visualization and analysis

1. TOTAL NUMBER OF GOALS SCORED BY EACH PLAYER

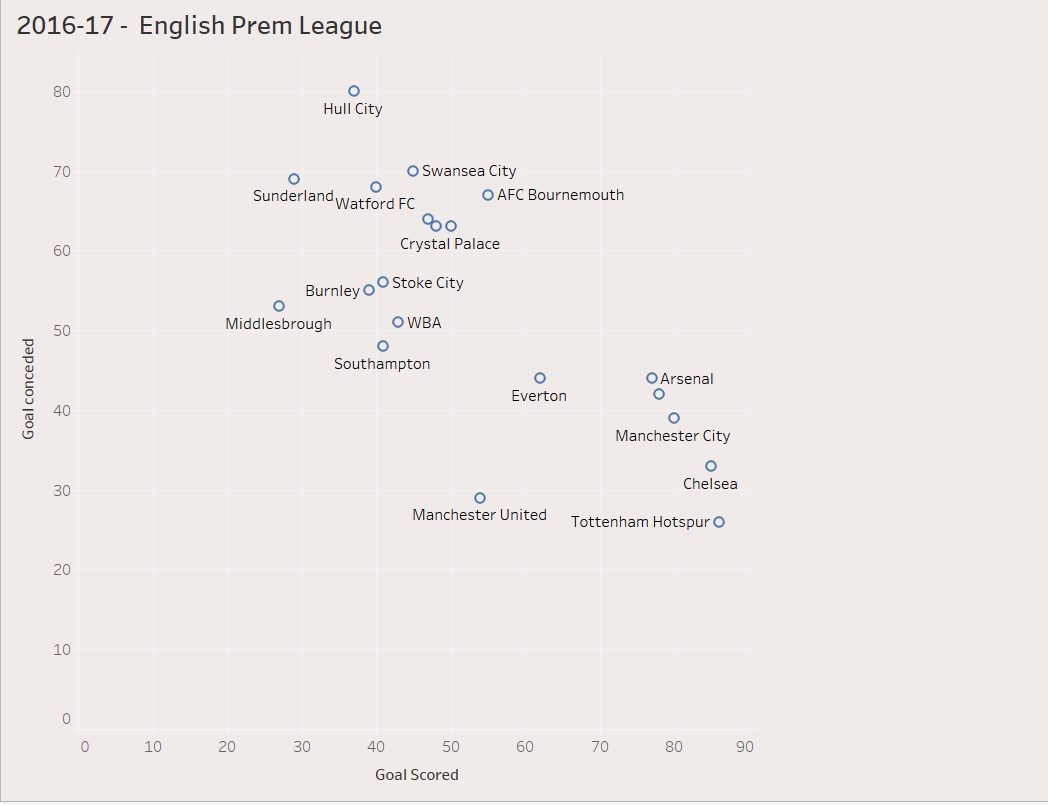
Query 1: Ref (2.1)



The above bar graph is showing total number of goals scored by the top 5 players in both the seasons, 2016-17 and 2017-18 for both the leagues. Blue bars indicate the goals scored in 2015-16 and orange bars indicate the goals scored in season 2016-17.

1. GOALS CONCEDED VS GOALS SCORED

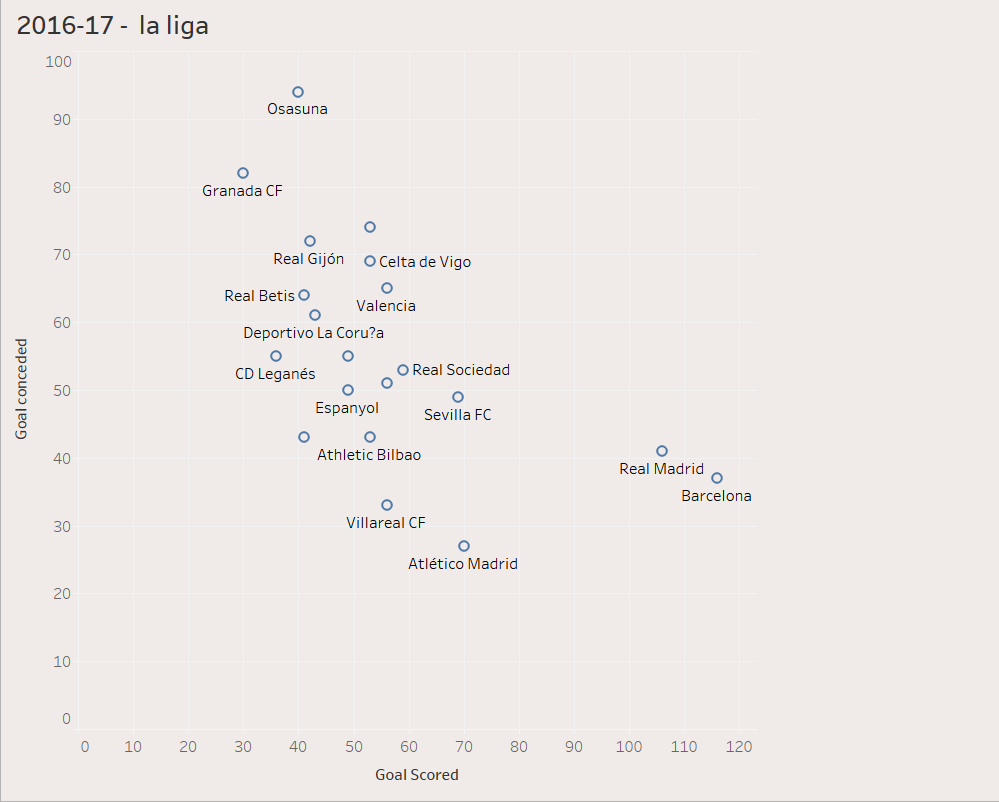
Query1 : Ref (1.1)



The above scatter plot shows the goals scored vs goals conceded for all the 20 teams played in 2016-17 in English Premiere League where some teams have scored more goals, and some have conceded more goals.

1. GOALS CONCEDED VS GOALS SCORED

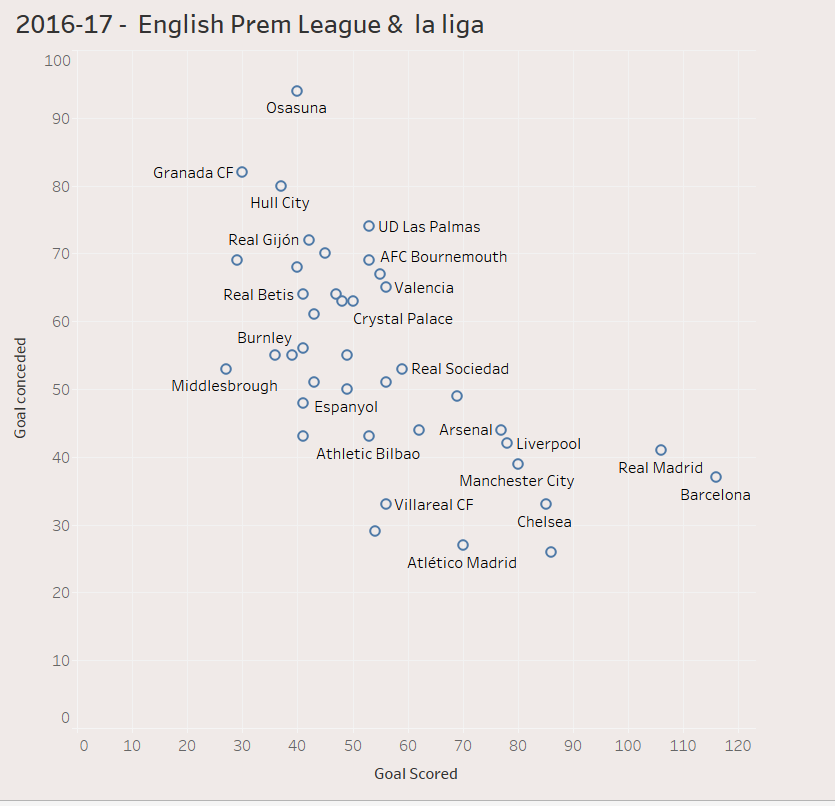
Query 1: Ref(1.1)



The above scatter plot shows the goals scored vs goals conceded for all the 20 teams played in 2016-17 in La Liga League where some teams like Osasuna scored more goals than goals conceded and some teams like Barcelona and Real Madrid have conceded more goals than goals scored.

1. GOALS CONCEDED VS GOALS SCORED

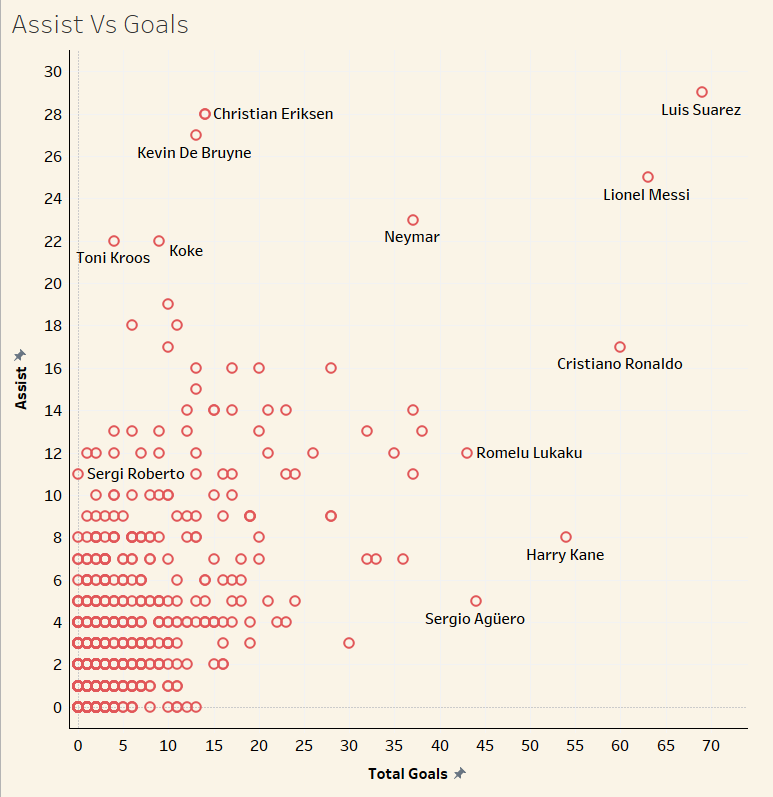
Query 1: Ref(1.1)



This scatter plot shows the relation between goals conceded vs goals scored for both English Premiere League and La Liga League in 2016-17.

1. TOTAL GOALS VS TOTAL ASSISTS

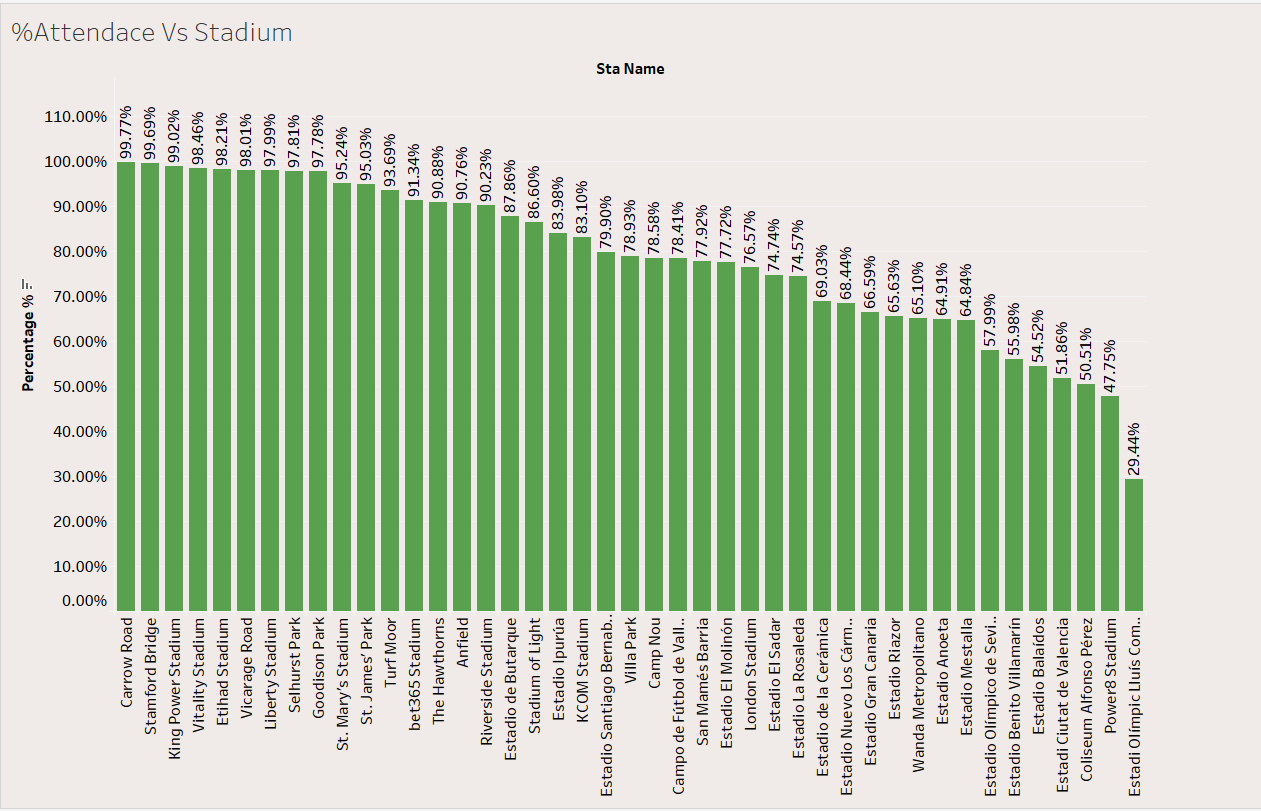
Query 1: Ref(2.1)



The above graph is a scatter plot showing the relation between total goals and assists for all the players. Most of the players are concentrated at the origin with moderate number of goals and assists but there are few good players like Suarez and Messi who have scored more goals and also assists.

1. % ATTENDENCE VS STADIUM CAPACITY

Query 1: Ref(3.1)

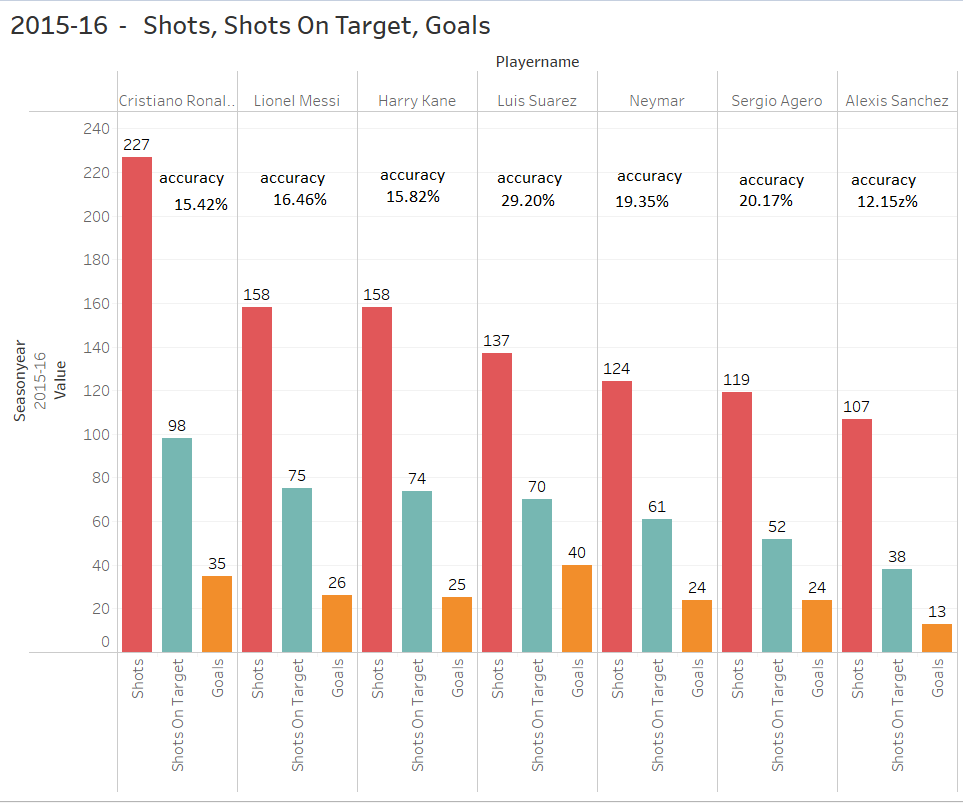


The bar graph shows the percentage of people attended and stadium’s capacity for each stadium. Here x-axis shows the stadium name and y-axis shows the percentage of stadium filled.

Stadiums like Carrow Road and Stamford Bridge have highest percentage of attendees and Estadi Olimplic has least number of attendees.

1. FORWARD’S SHOT ACCURACIES IN 2015-16

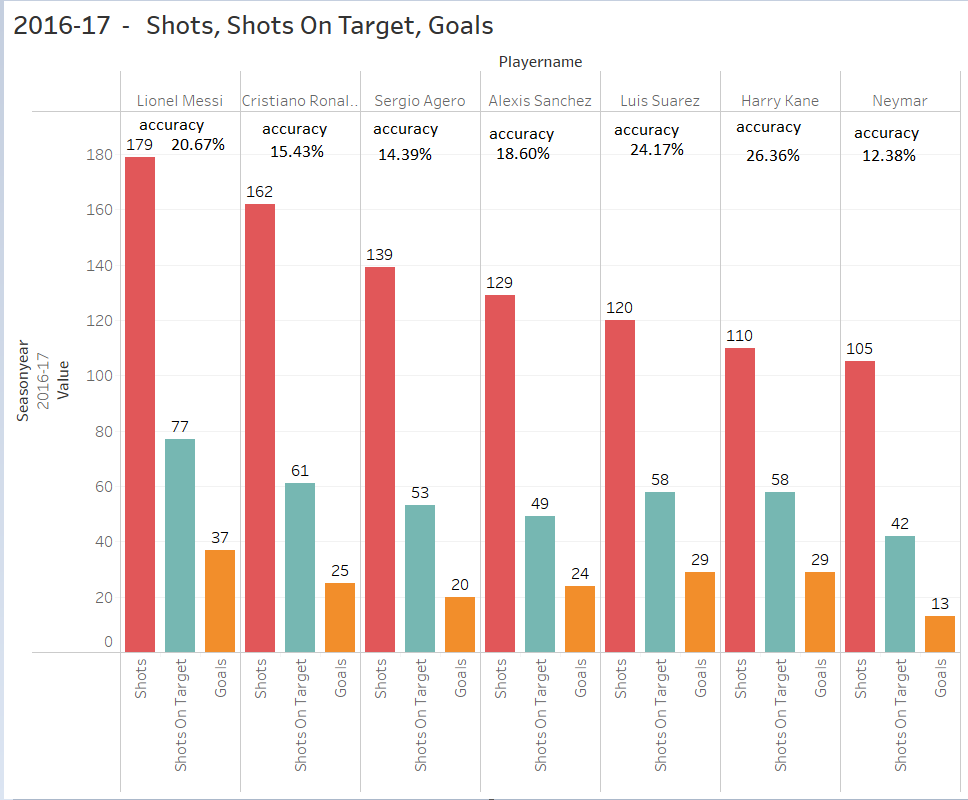
Query 1: Ref(2.1)



The above bar graph displays the accuracy in scoring goals based on number of shots, shots on target and goals scored for top 6 players in the year 2015-16. Here Ronaldo is hitting highest number of shots but Agero has maximum accuracy, so accuracy does not depend just on shots but on both goals and shots.

1. FORWARD’S SHOT ACCURACIES IN 2016-17

Query 1: Ref(2.1)



The above bar graph displays the accuracy in scoring goals based on number of shots, shots on target and goals scored for top 6 players in the year 2016-17. Here Messi is hitting highest number of shots and goals, but Harry Kane has maximum accuracy, so accuracy does not depend just on shots but on both goals and shots.

# APPENDIX A – Table creation script



# APPENDIX B – Semantria results

# APPENDIX C – SQL Results