INTRODUCTION TO BIG DATA

PROJECT REPORT - EUROPEAN SOCCER ANALYSIS - GROUP 10

SAHAJ GANDHI | ANIKET GIRIYALKAR | ADITYA KALYAN JAYANTI

CSCI 620 | 04/18/2018

Table of Contents

INTRODUCTION

Soccer is a global sport. It is played in almost every country in the world, and soccer competitions have global viewing numbers that blow all other sports (and even the Olympics) out of the water. The 2014 FIFA World Cup Final has been by far the most viewed sports event in recent times, with over 700 million viewers. It makes sense that its top professional leagues are different and superior in structure than those of any other sport in the world.

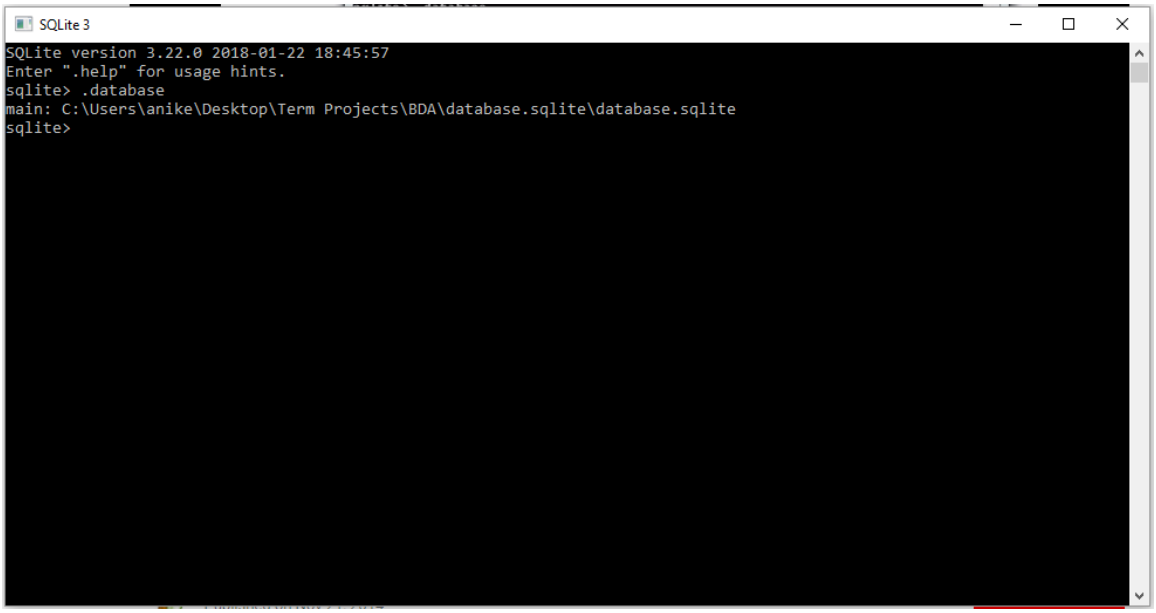
European Soccer is more competitive than any other sports leagues because of the format of the leagues, the multiple competitions, and the intense rivalry between not only each team, but in each league.

To understand what makes European Soccer the most competitive league in the world, we have chosen to examine this database, which contains information about over 2500 matches, about 10000 players representing over 11 European Countries playing in the top tiers of their championships.

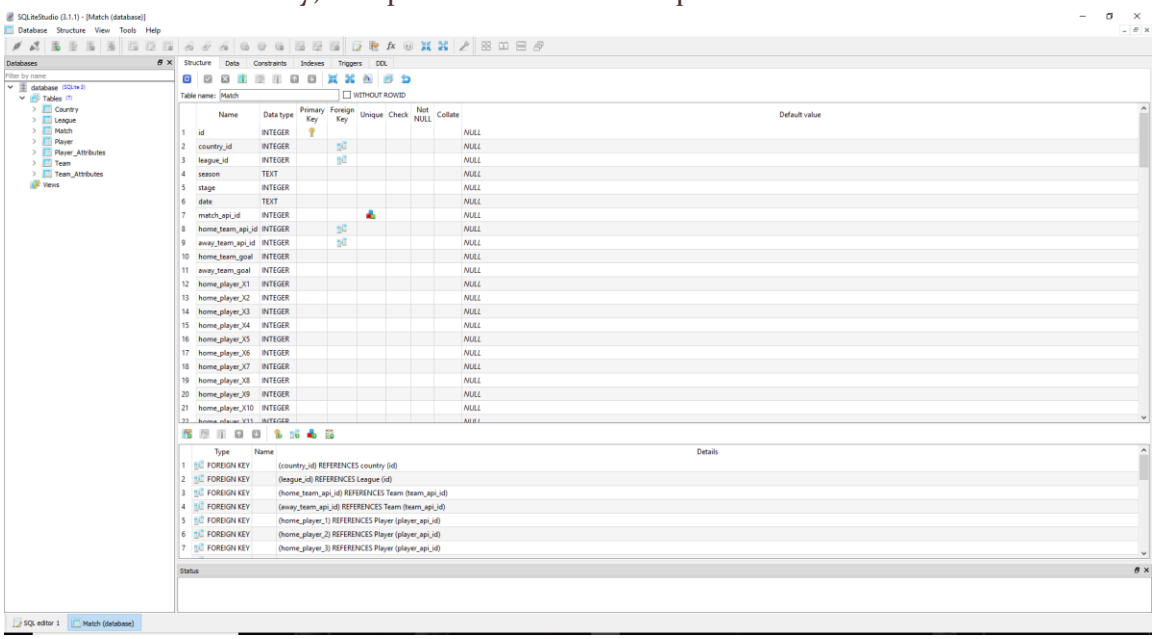
In this project we attempt to understand the factors of the teams and the players who play in Europe and analyze the player_attributes and team_attributes and generate insights from the data.

Data Migration

TRANSFERRING SQLITE FILE TO SERVER SQL

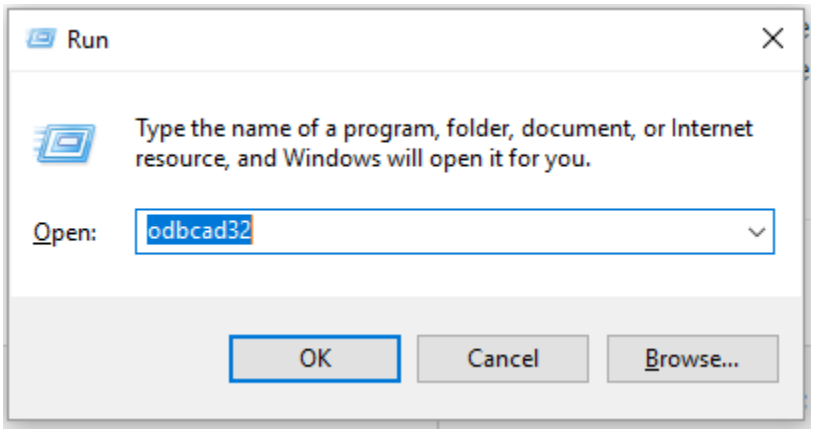


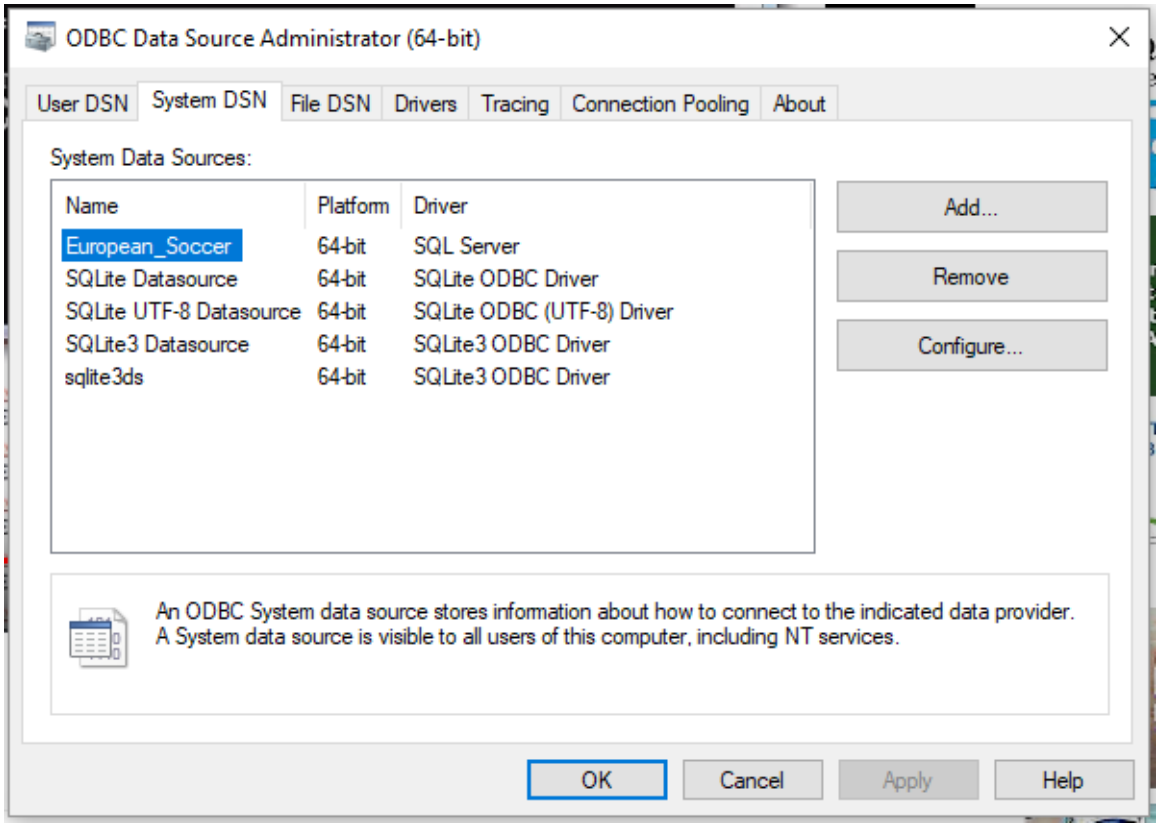
Firstly, we opened the dataset in Sqlite command line.



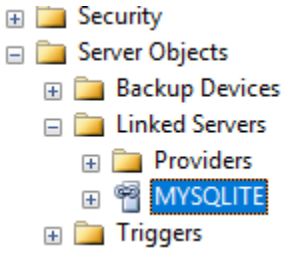
Then, we observed our dataset entities and all the table structures in the SQLite GUI.

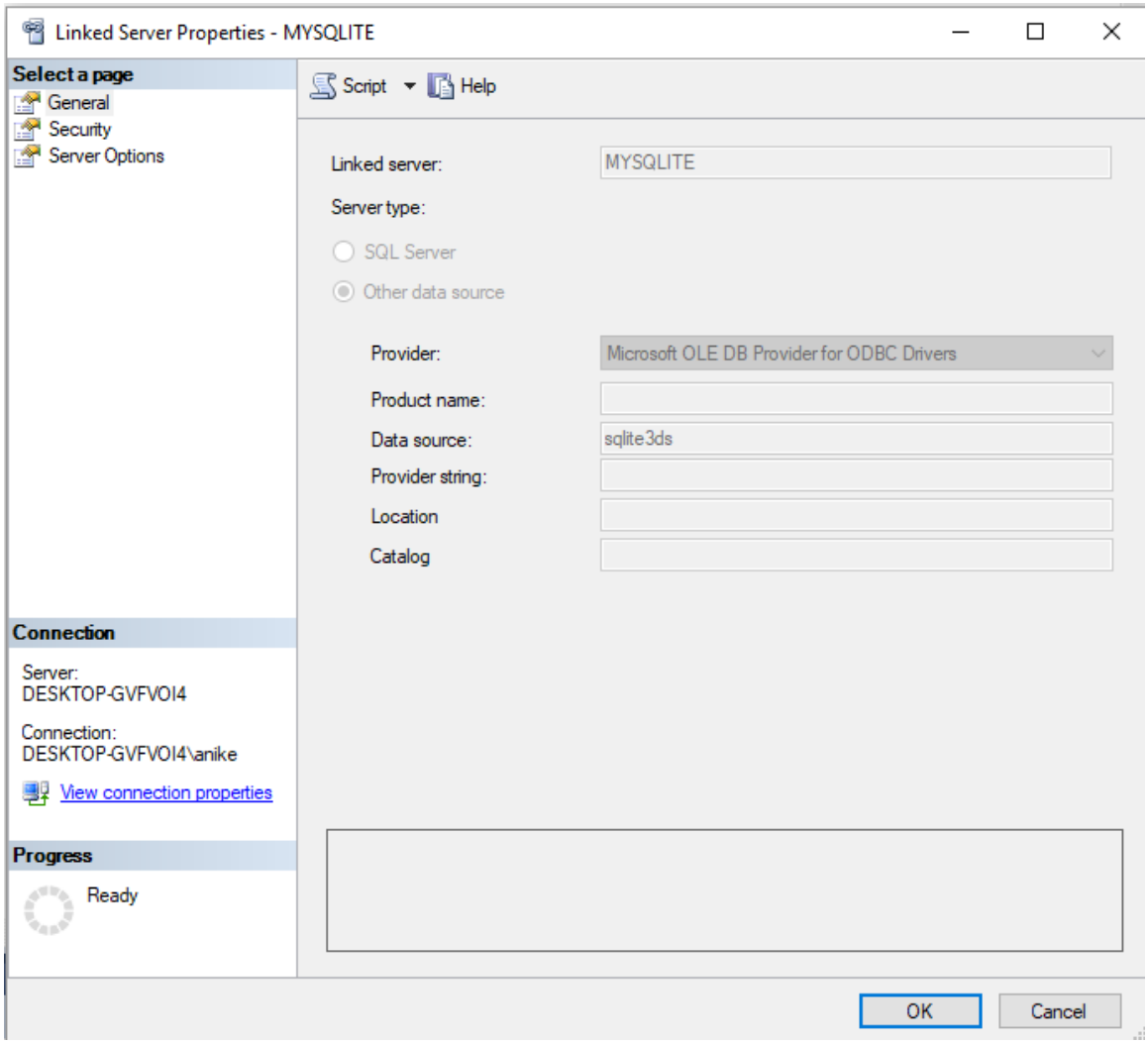
After this step we created the DSN for connecting the SQLite server to SSMS server. For this we type odbacd32 in the run command and create a DSN.





Now, in SSMS we create a new linked server and hence connect it with the DSN we have created in the previous step.





The data source has the same name as the name of the DSN we created for SQLite.

Now to get our data from the dataset, we create tables in SSMS having the same structure as the source Dataset and insert data from the tables of SQLite into the tables that we have created in SSMS.

```
Use European_Soccer
CREATE TABLE Country (
    id   INTEGER PRIMARY KEY,
    name Varchar(512)    UNIQUE
);
CREATE TABLE League (
    id         INTEGER PRIMARY KEY,
    country_id INTEGER,
    name       Varchar(512)   UNIQUE,
    FOREIGN KEY (country_id)
    REFERENCES country (id)
);
Use European_Soccer
Insert into Country(id, name)
select id,name
from openquery(MYSQLITE, 'select * from country ')
Use European_Soccer
Insert into League(id, country_id , name)
select id,country_id,name
from openquery(MYSQLITE, 'select * from league ')

CREATE TABLE Player (
    id                 INTEGER PRIMARY KEY,
    player_api_id      INTEGER UNIQUE,
    player_name        Varchar(512),
    player_fifa_api_id INTEGER UNIQUE,
    birthday           Varchar(512),
    height             INTEGER,
    weight             INTEGER
);

Use European_Soccer
Insert into Player([id]
      ,[player_api_id]
      ,[player_name]
      ,[player_fifa_api_id]
      ,[birthday]
      ,[height]
      ,[weight])
Select [id]
from openquery(MYSQLITE, '
Select * from [Match] where ( home_team_api_id in ( Select team_api_id from Team) and away_team_api_id in ( Select team_api_id from Team)) ')

from openquery(MYSQLITE, 'Select * from Player_Attributes where player_fifa_api_id in ( Select player_fifa_api_id from Player) ')
Select * from Player_Attributes
```

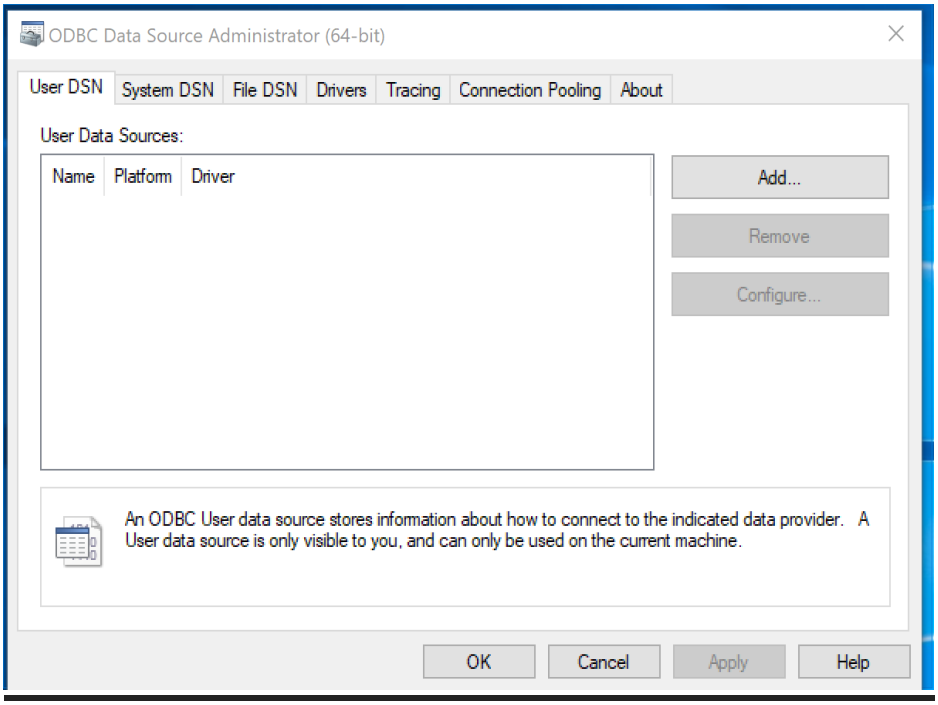
The above queries explain how the data is selected from tables of SQLite and inserted into SSMS

OPENING OUR DATASET IN R AND RATTLE

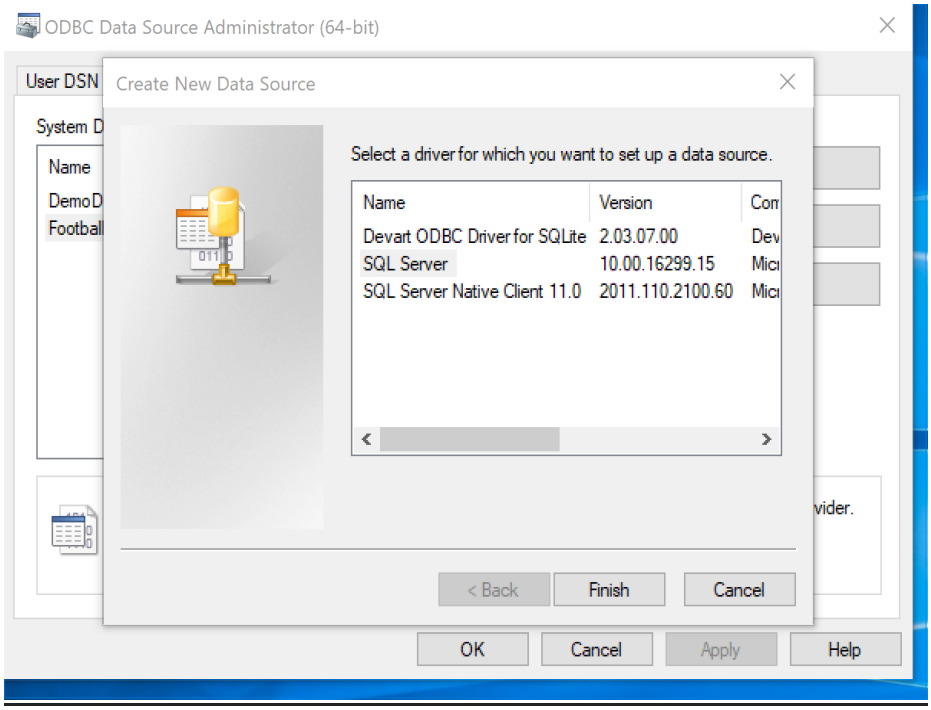
ODBC aims to provide common API access to SQL based data management systems.
A connection to a specific database is called a Data Source Name or DSN.
To make an ODBC connection:

- *library(RODBC)*
- *Channel –> odbcConnect("FootballDSN")*

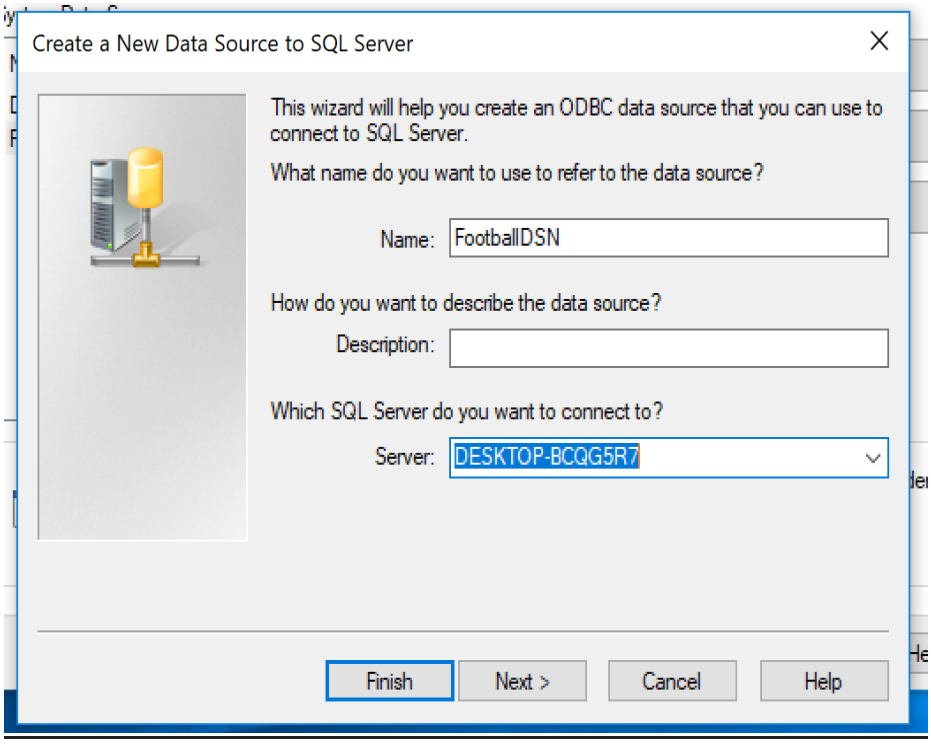
Select, ODBC data sources(64-bit) under administrative tools.



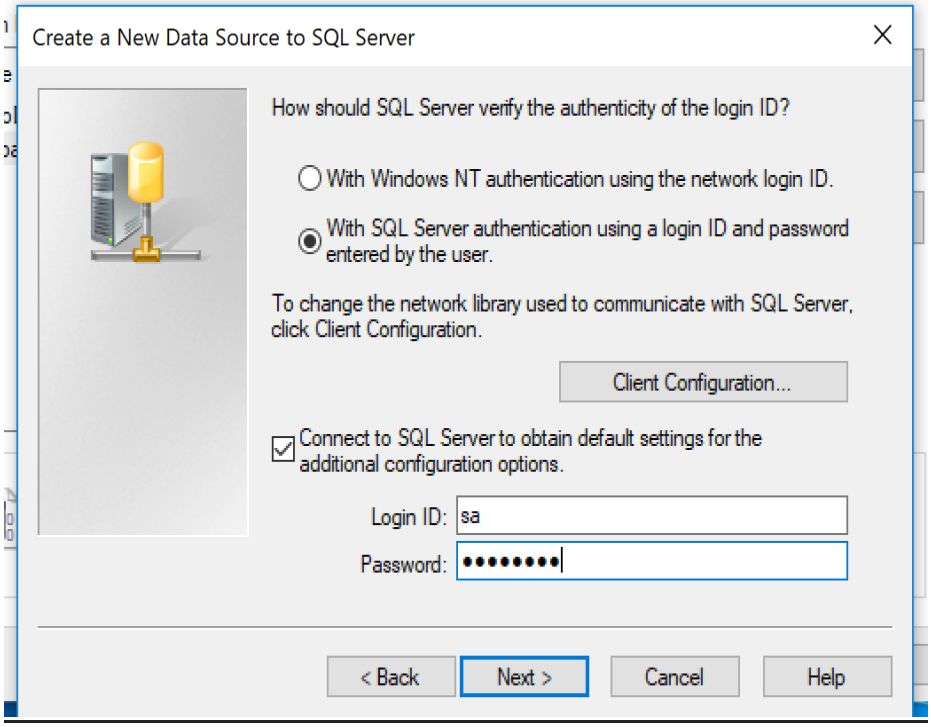
Adding a new User DSN in ODBC data sources.



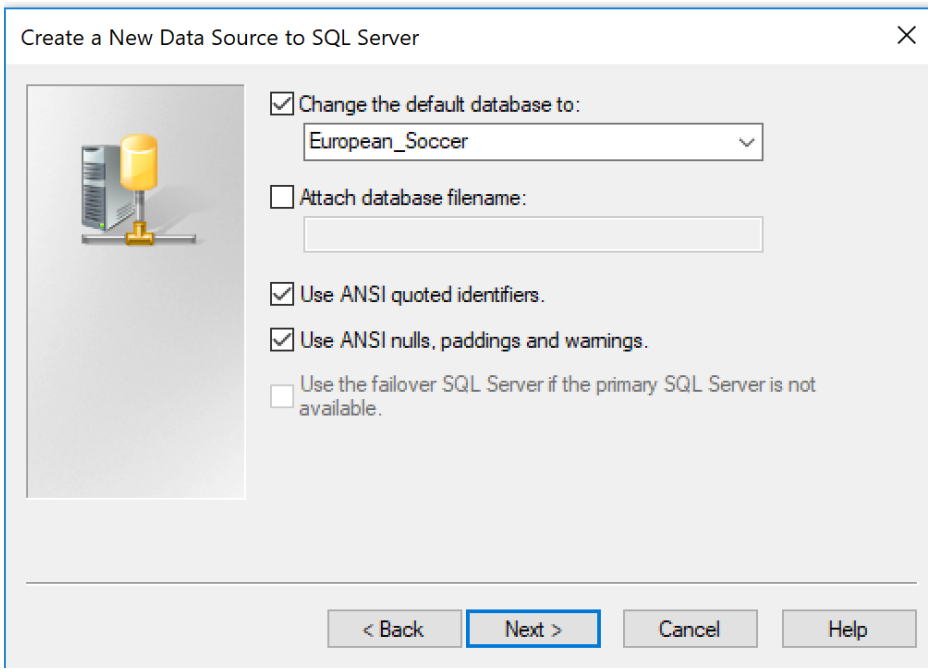
SQL Server driver is selected to set up the data source.



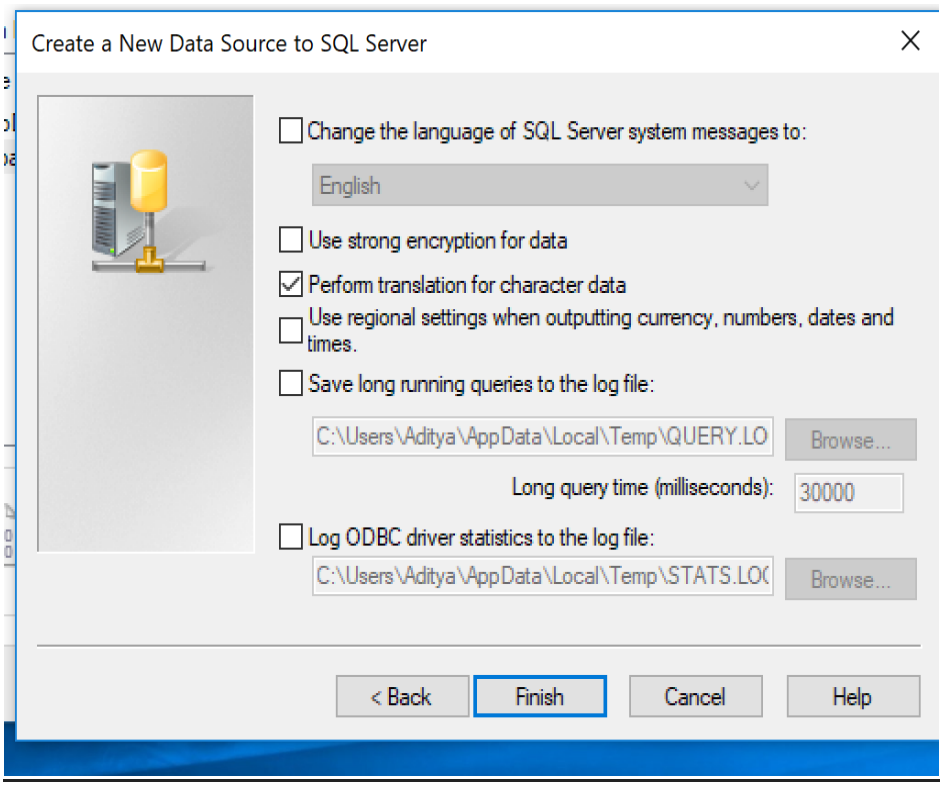
“FootballDSN” is a name given to the data source. The default SQL Server is selected.



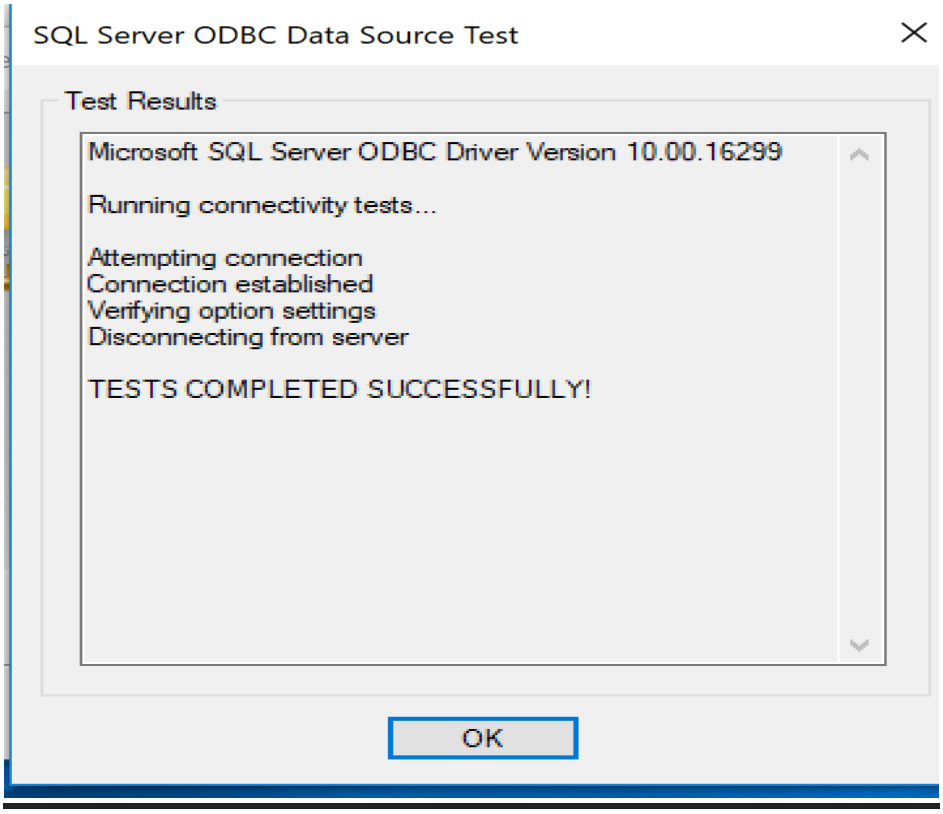
SQL Server is authenticated with log in ID and password. The default settings are used to obtain additional configuration options. With log in ID as “sa”. This authentication allows the user to access the current databases present in SQL Server Management System (SSMS).

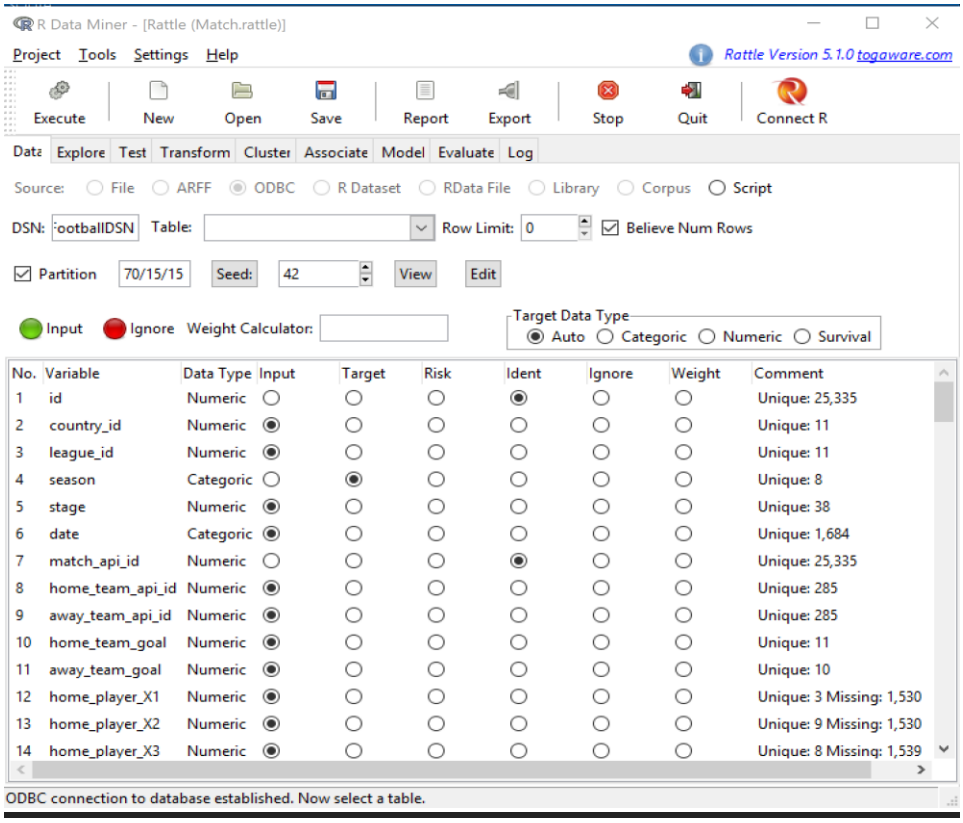


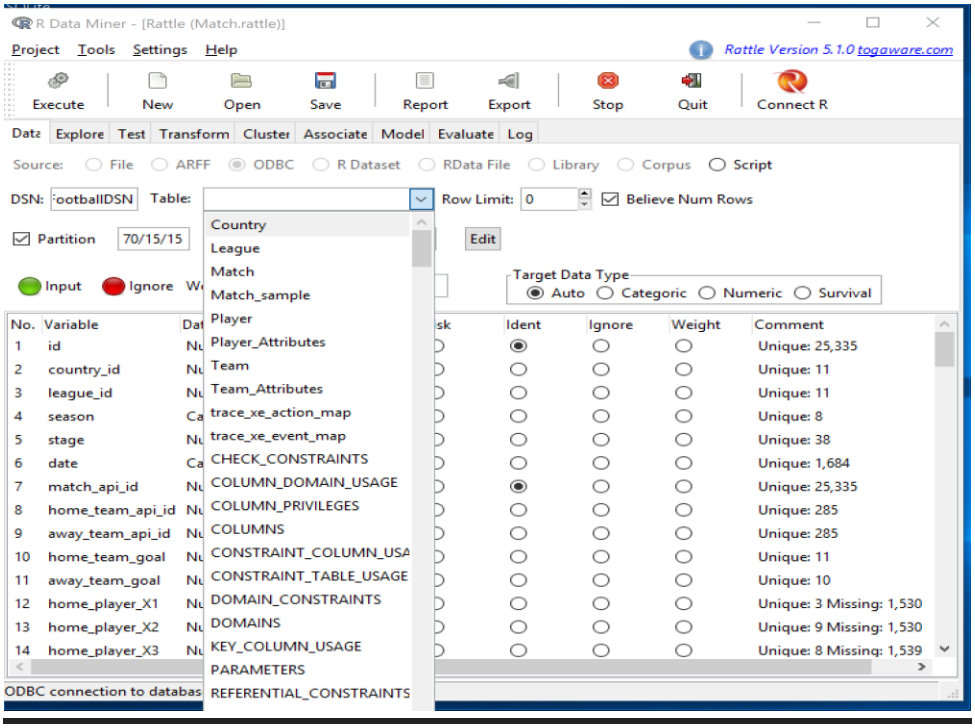
The default database is selected from the list of databases available in SSMS.



A new data source is created for the default database. Select “Finish” to test the data source.

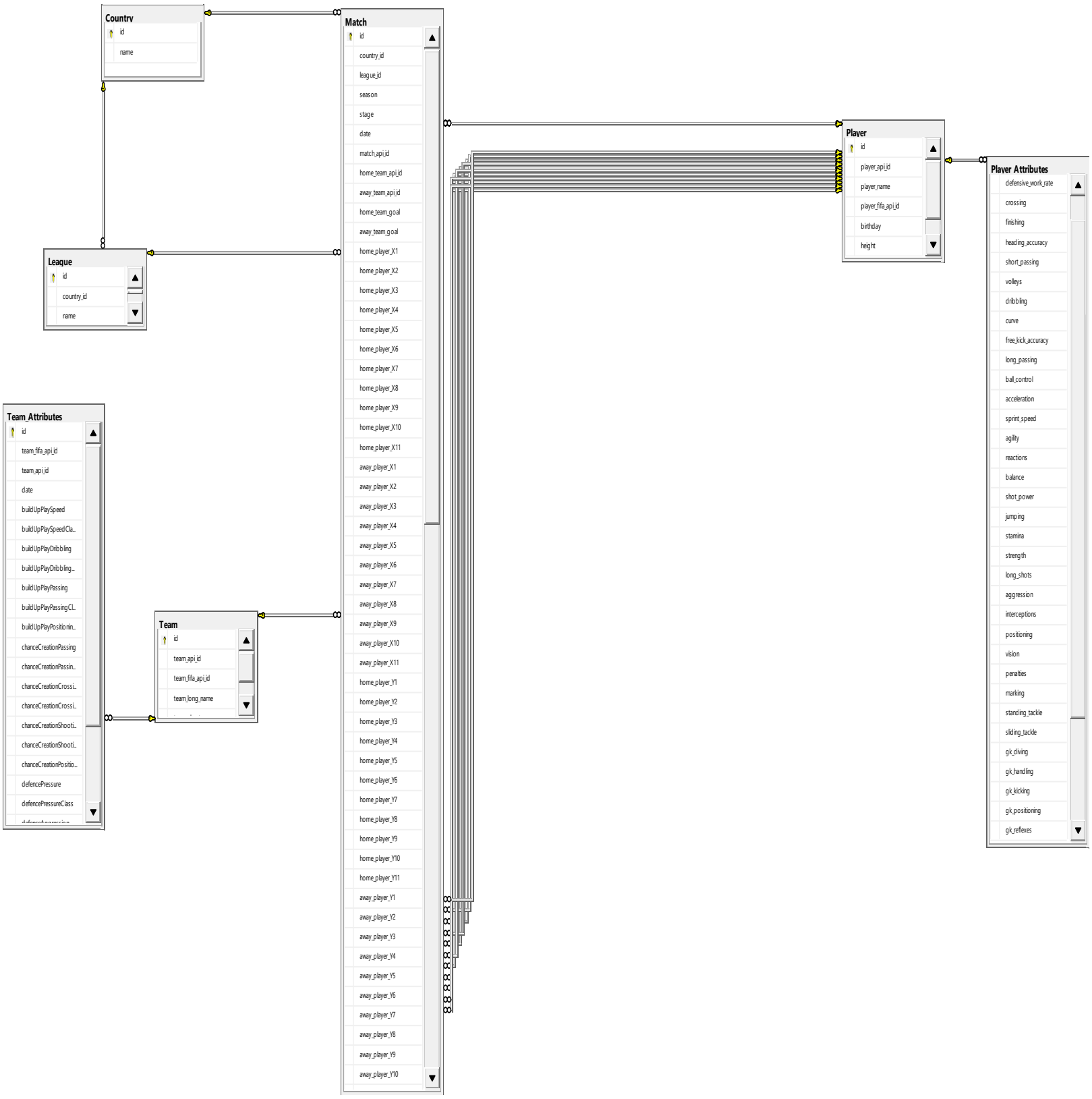






We select the table we want from the dropdown.

SCHEMA DIAGRAM



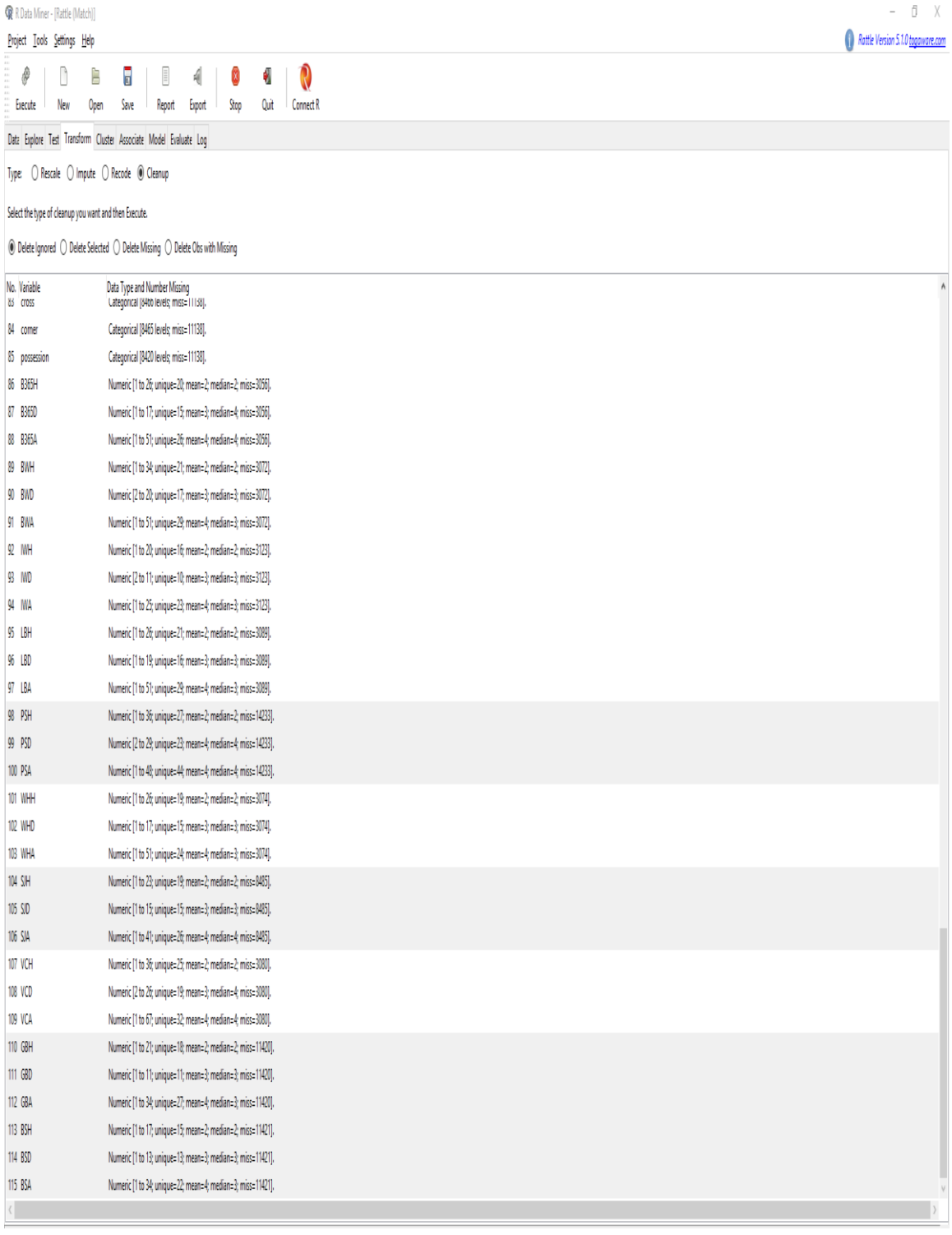
DATA CLEANING

After we have opened the dataset on R and Rattle we browse through each table to check if it needs to be cleaned. On careful observations we find that the tables Player, Team, League, Country, Player_Attributes, Team_Attributes are well organized and have very less missing values. But in the table match we observe that almost more than 30% of the data is missing in the following attributes:

- PSH and PH - Pinnacle home win odds
- PSD and PD - Pinnacle draw odds
- PSA and PA - Pinnacle away win odds
- SJH - Stan James home win odds
- SJD - Stan James draw odds
- SJA - Stan James away win odds
- GBH - Gamebookers home win odds
- GBD - Gamebookers draw odds
- GBA - Gamebookers away win odds
- BSH - Blue Square home win odds
- BSD - Blue Square draw odds
- BSA - Blue Square away win odds
- GOAL
- SHOTON
- SHOTOFF
- FOULCOMMIT
- CARD
- CROSS
- CORNER
- POSSESSION

Since 30% is huge amount of data compared to the total number of observations, we can't even replace the missing values with either mean, median, or any custom value and hence

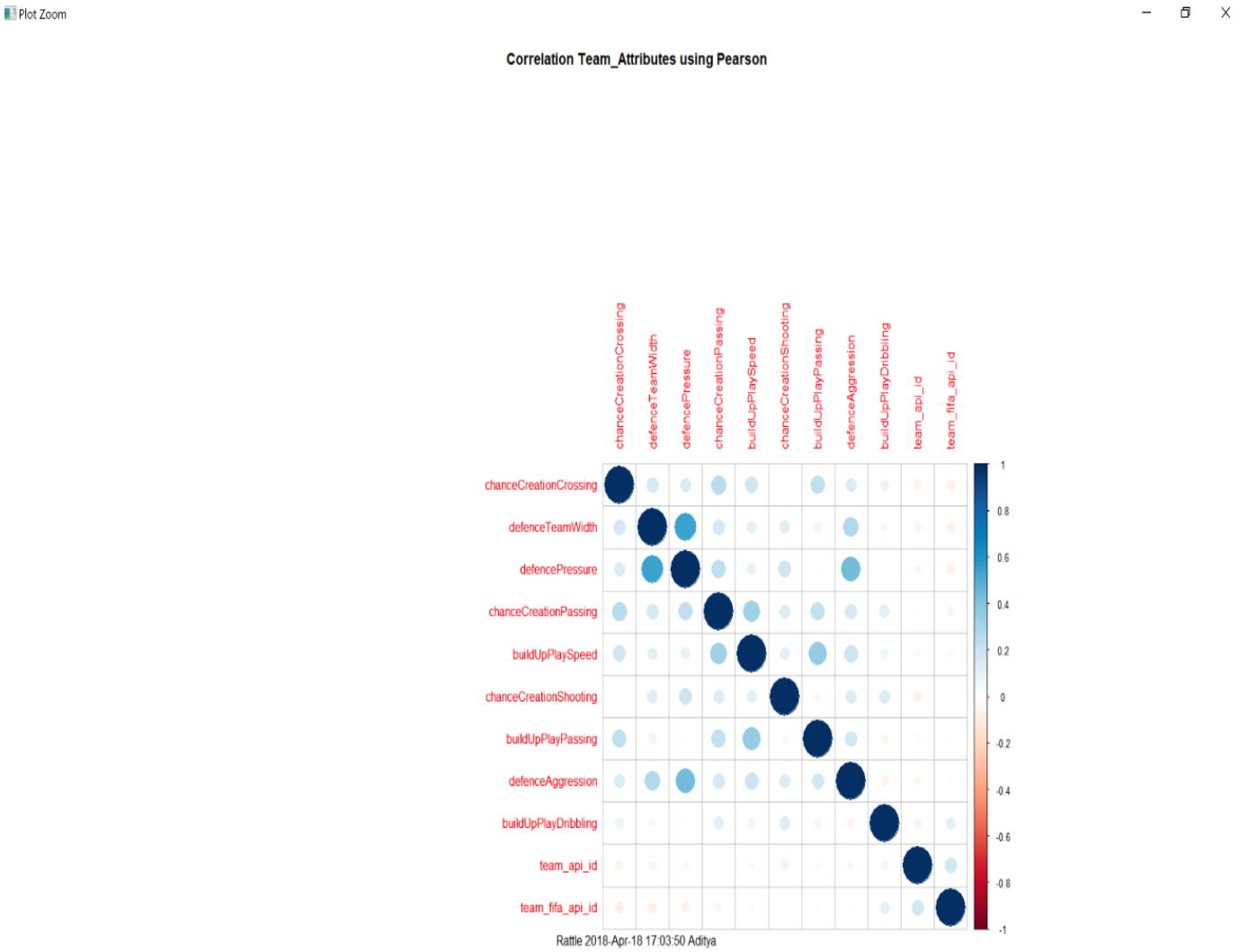
delete these columns.



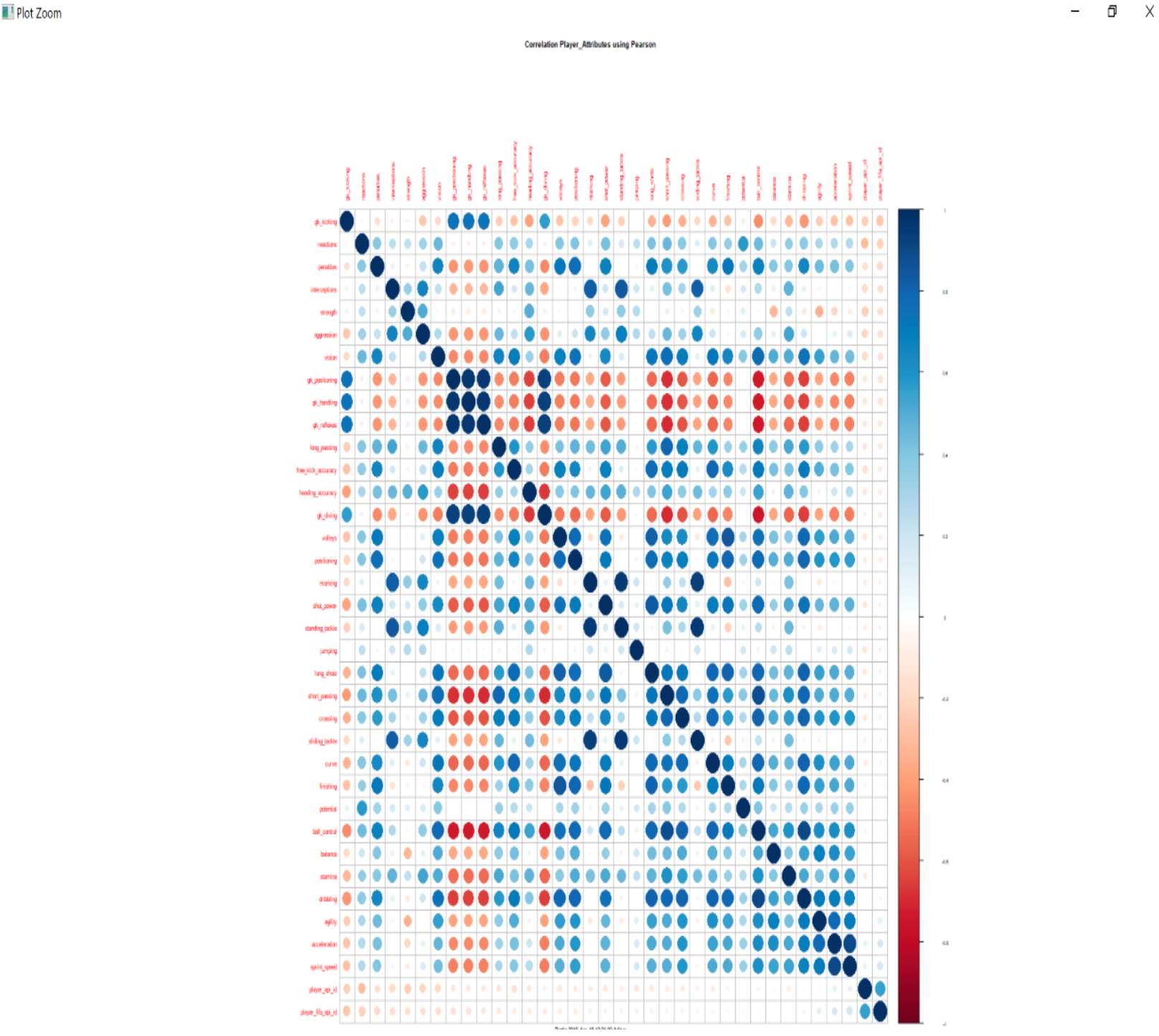
As we see in the figure above that the missing values are far too much and hence it justifies the deleting of the above-mentioned attributes.

DATA EXPLORATION

In data exploration, we explore all our tables to see if we can find any correlation between the attributes. Using the below two plots we observe the correlations between the numeric attributes of the tables Team_Attributes and Player_Attributes.



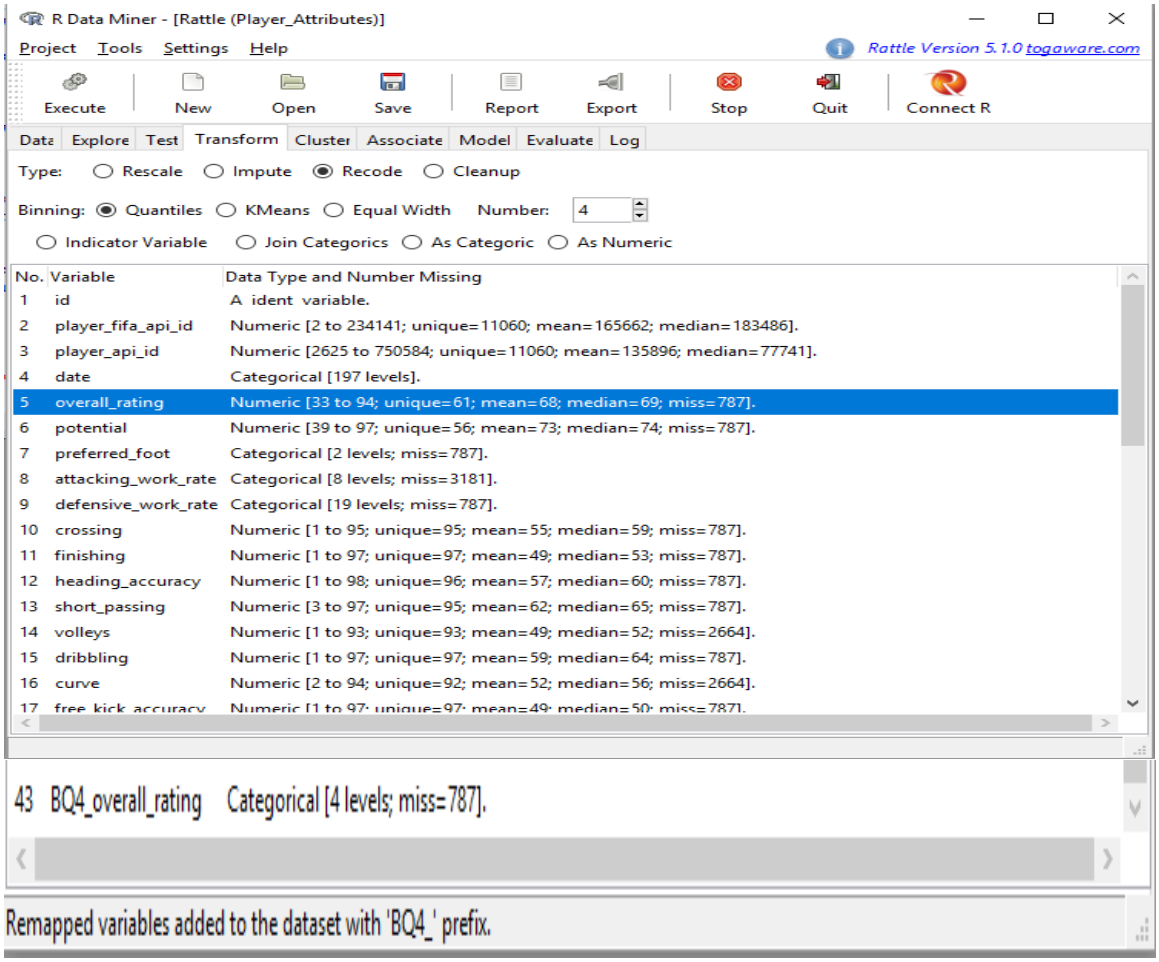
The blue circles denote the positive correlations between the attributes, and lighter the color less the correlation. Red circles indicate the negative correlation. We observe the attributes showing positive correlations more closely and visualize these to obtain further insights on these attributes and what they represent.



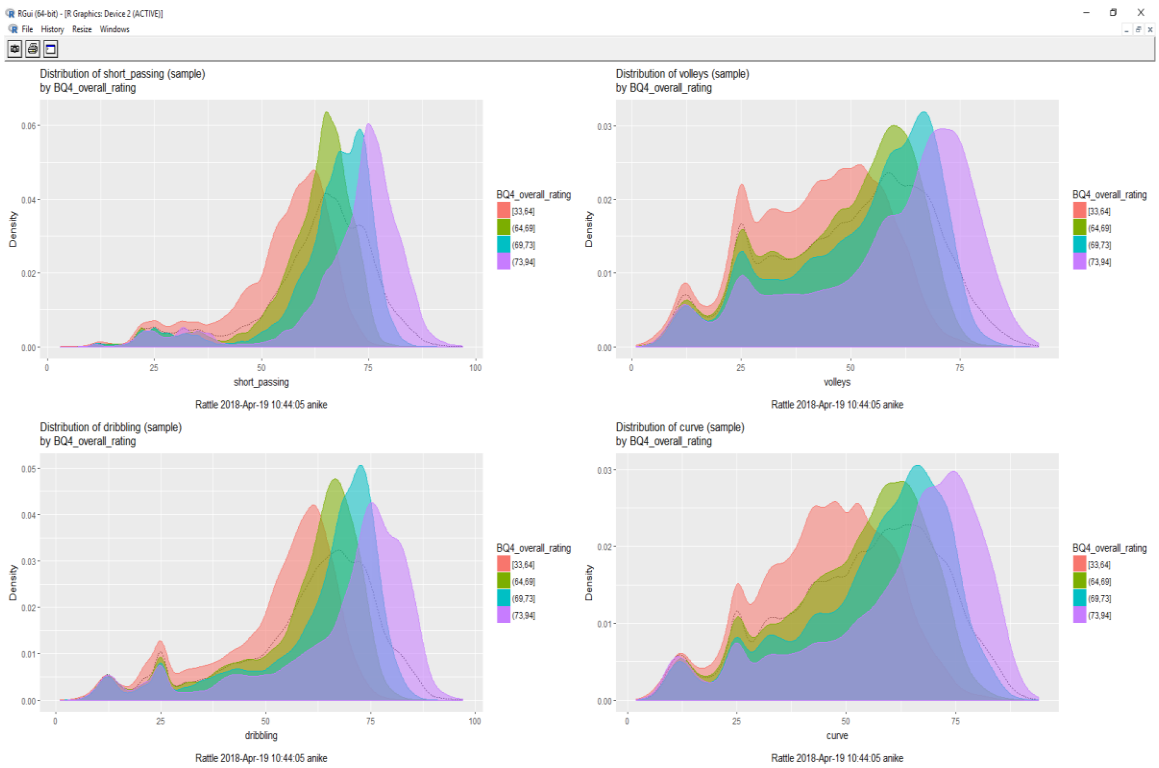
We see that in tables Team, League, Country, and Player we don't have any interesting correlations and hence we do not study the attributes of these tables as it doesn't make any sense to read for the relations that do not exist.

DATA TRANSFORMATION

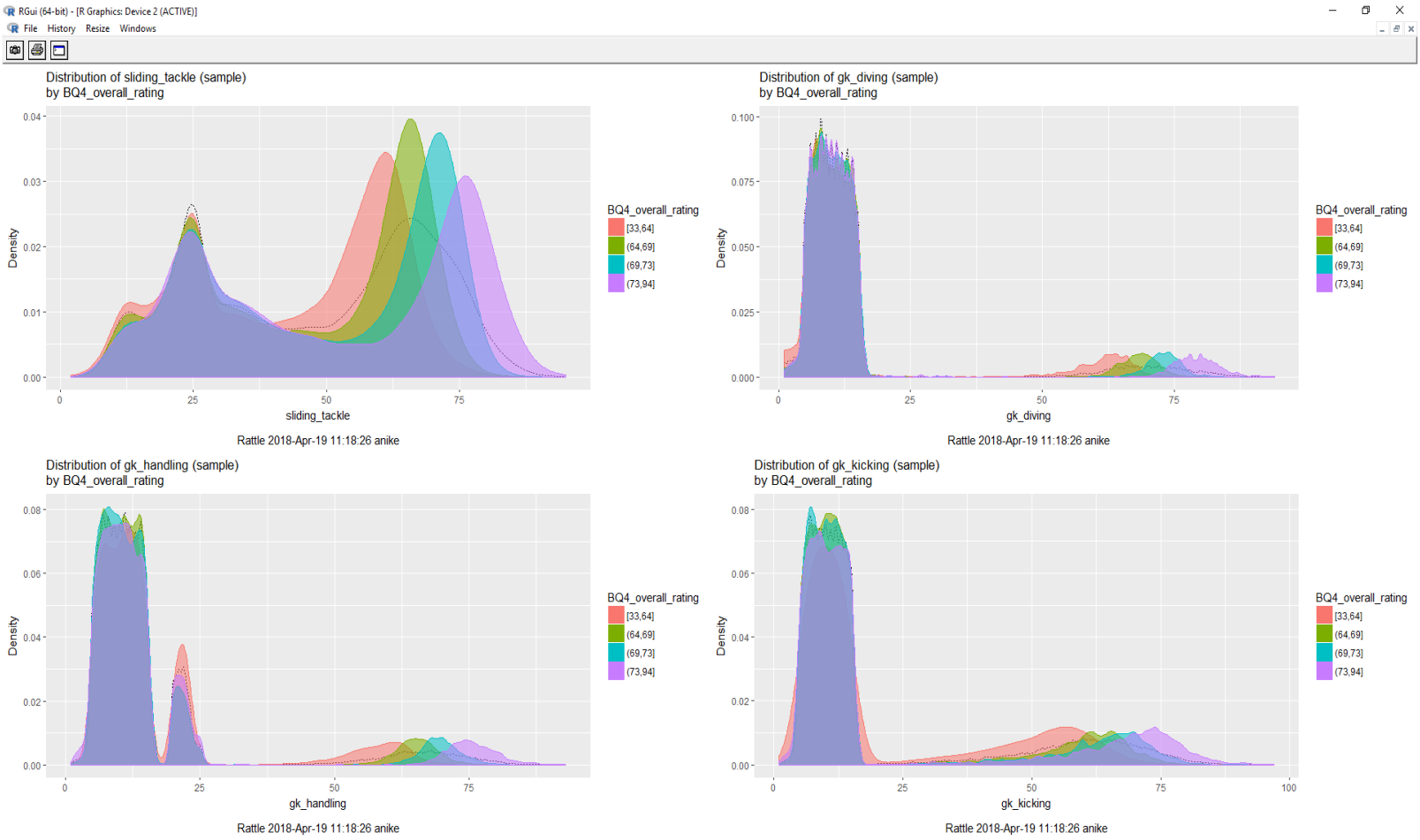
We find in the table player_attributes most of the values have very good correlation with overall_rating, but the original type of this target variable is numeric and has more than 60 unique values which makes it very hard for us to draw any meaningful analysis from this attribute. Hence, we use binning(recode) on the attribute overall_rating and make it our target variable. This target variable is of categoric type and has 4 unique values, each of this value is the range in which the original distribution is adjusted.



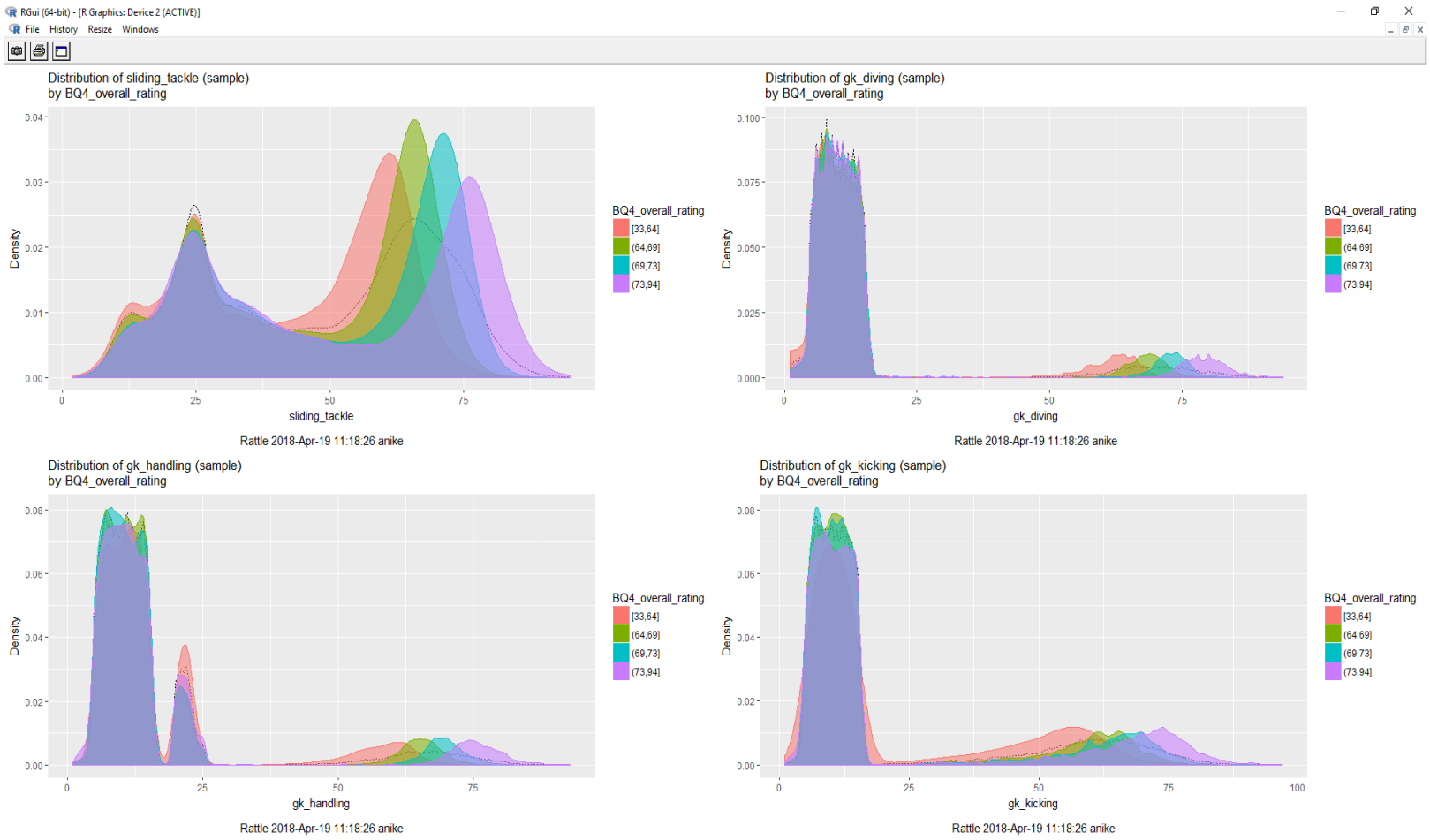
This attribute will help us to visualize our data in the player_attributes in a much better way. Also, applying transformations on attributes of other tables will not yield any interesting observations different from already existing attributes. Hence, this is the only attribute we will perform transformation on.



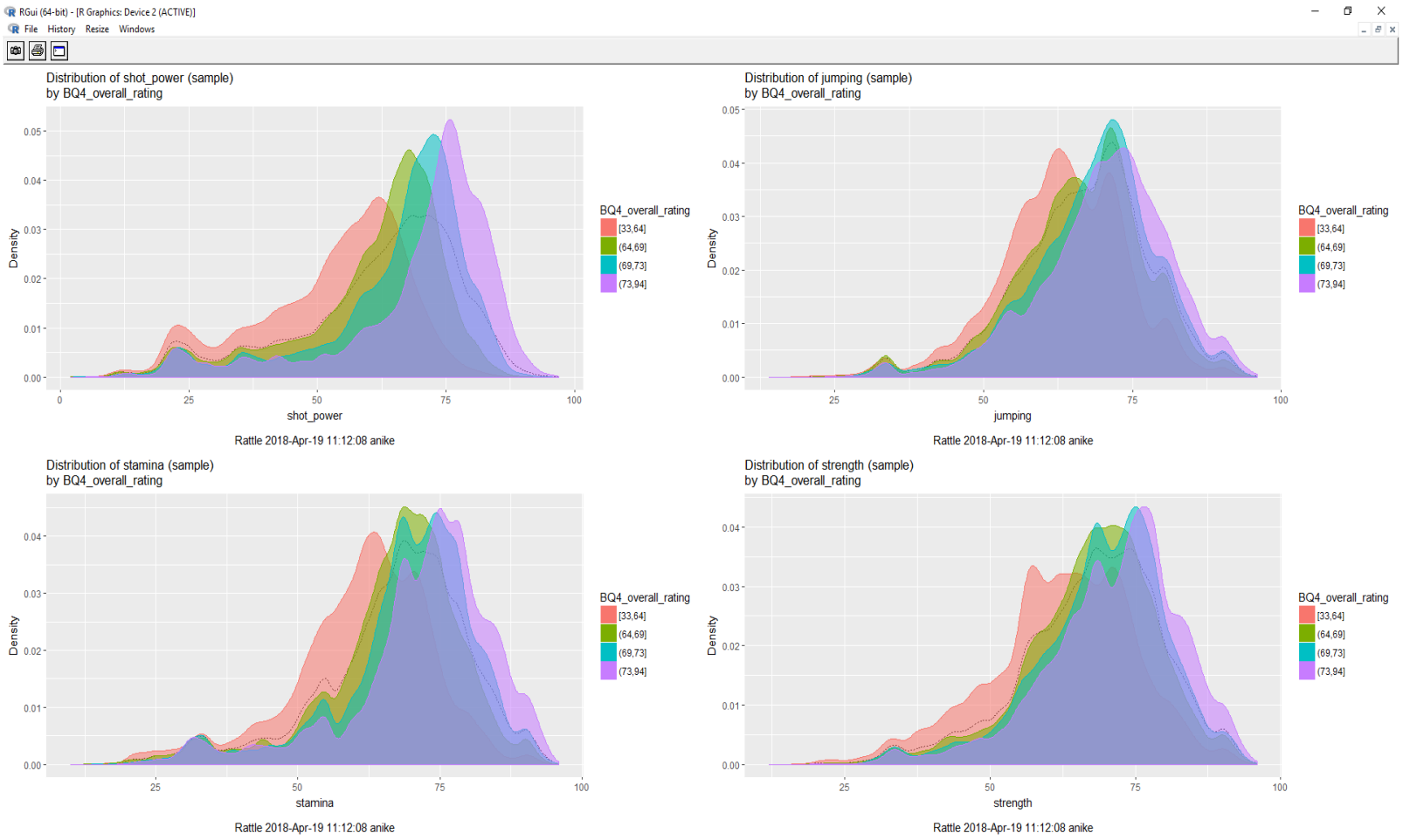
On observing relations between all the attributes and various plots from all the tables we separate out only the one's who provide information in some or the other. We find out that apart from tables Player_Attributes and Team_Attributes there is no other table whose visualization gives any useful result. The selected plots that we found interesting are given below with the description of what we observed for each.



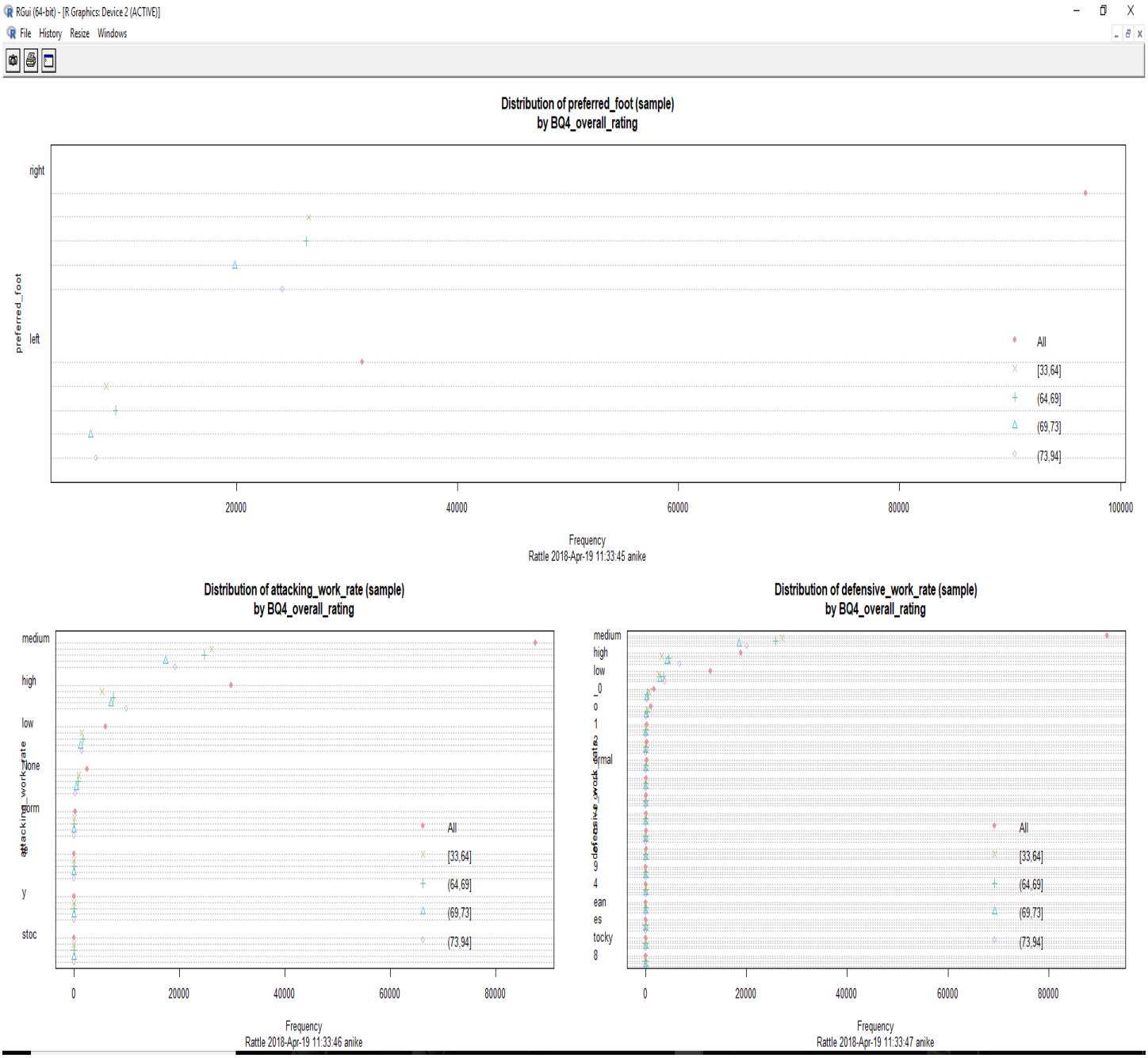
The above histogram plot is distribution of attributes short_passing, volleys, dribbling, and curve plotted against the transformed target variable overall_rating. All these groups are rightly skewed, indicating that most of the ratings of each are greater than the central value and each of these attributes have larger number of users having values more than 50 ratings. Also, for each attribute we observe that overall rating range is more than half the value and have more users with ratings in the 3rd and 4th ranges.



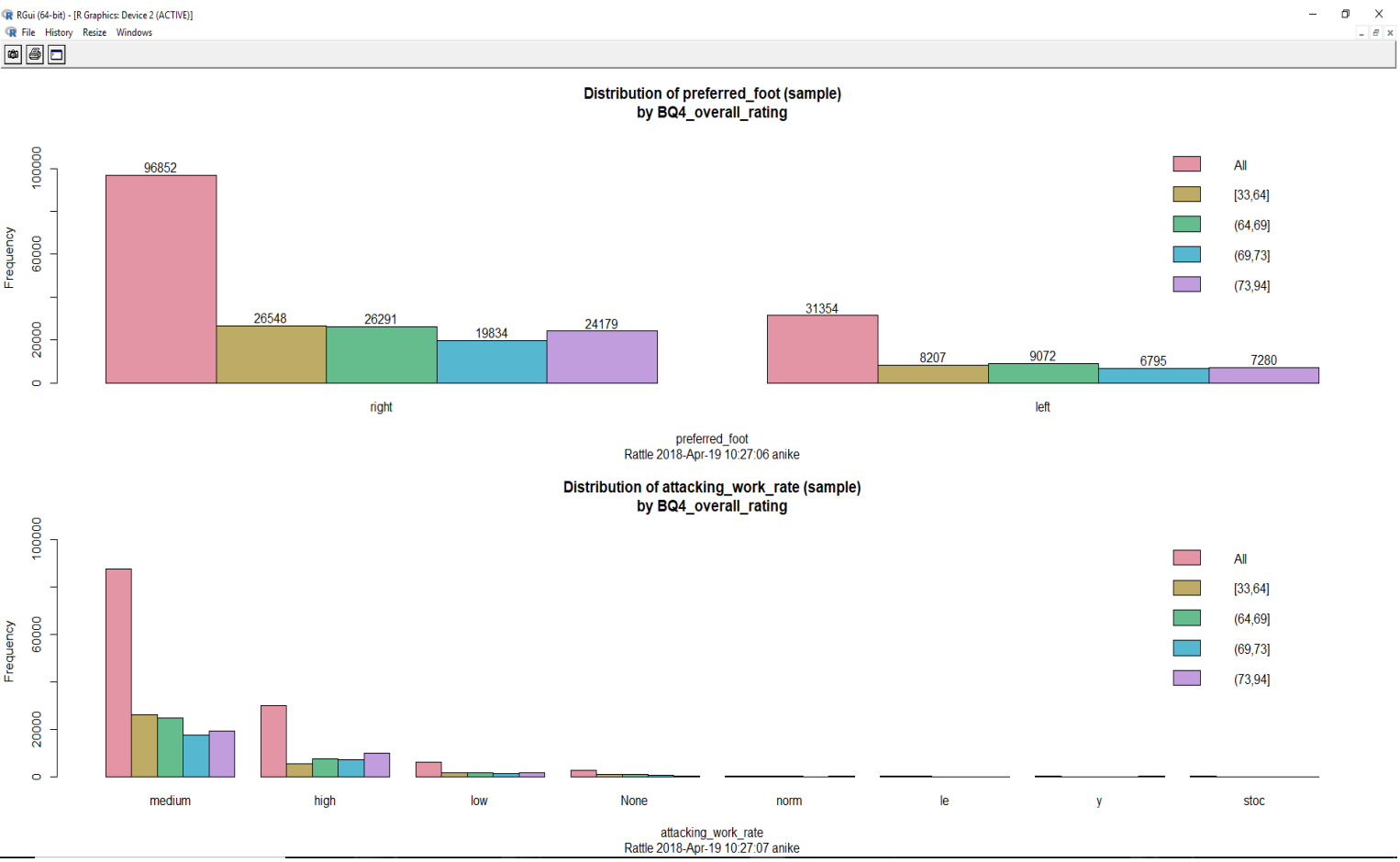
In the above plots we observe the distributions of goalkeepers and plot them against overall ratings. The data is skewed towards the left indicating most of the players are not goalkeepers. However, in the distribution of sliding_tackle vs overall_rating we observe that it has values on both left and right sides, indicating that these attributes might be present in defensive type of players like goalkeepers and defenders, from the left distributed values we can infer that certain position players like strikers might not posses this property.



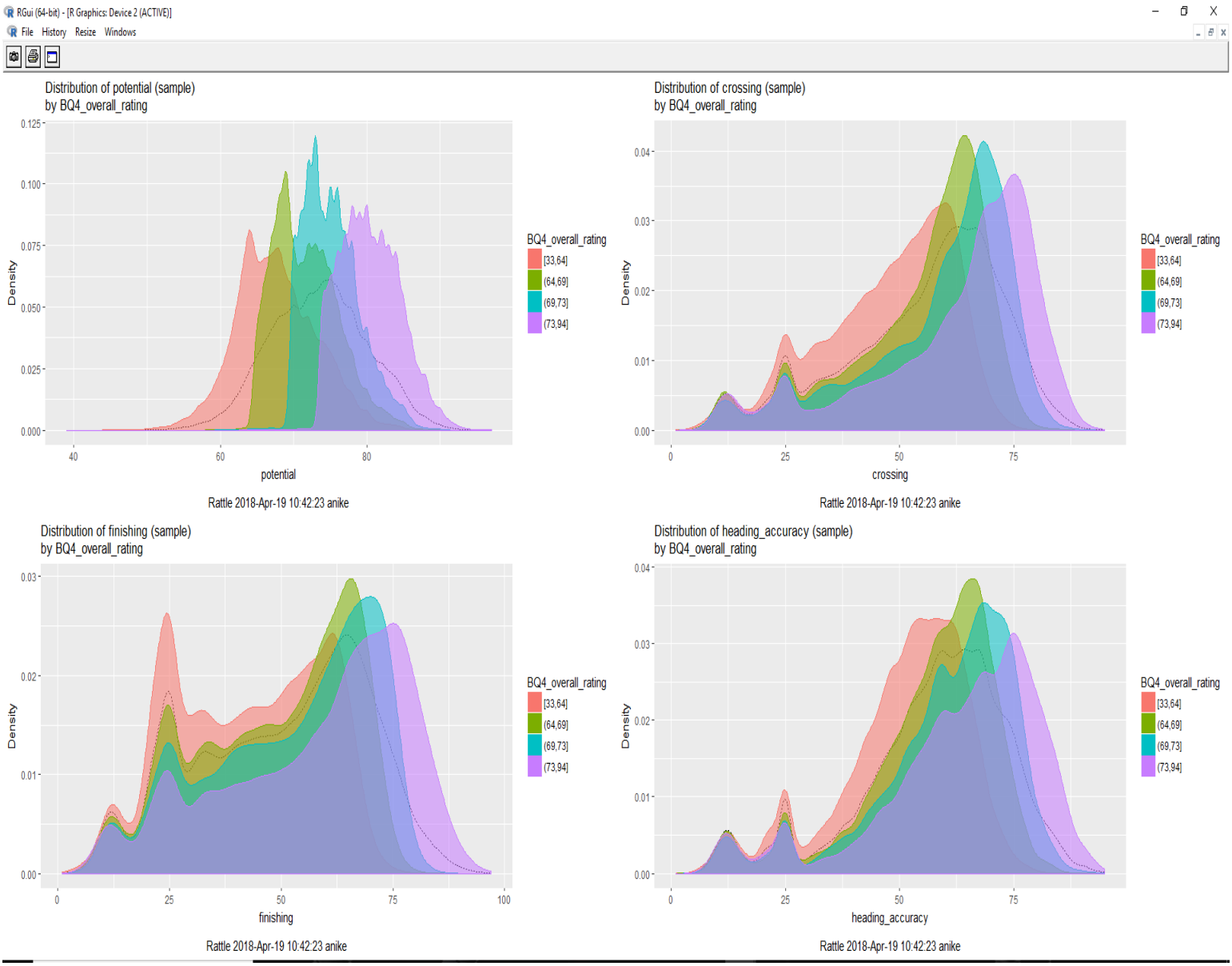
In this distribution we observe that all the plots are rightly skewed, indicating that these attributes are present on almost all the players and it can be inferred that this has close relation with fitness and it is expected by each professional player to have good fitness or at least have a threshold amount of fitness required by the game to survive the game.



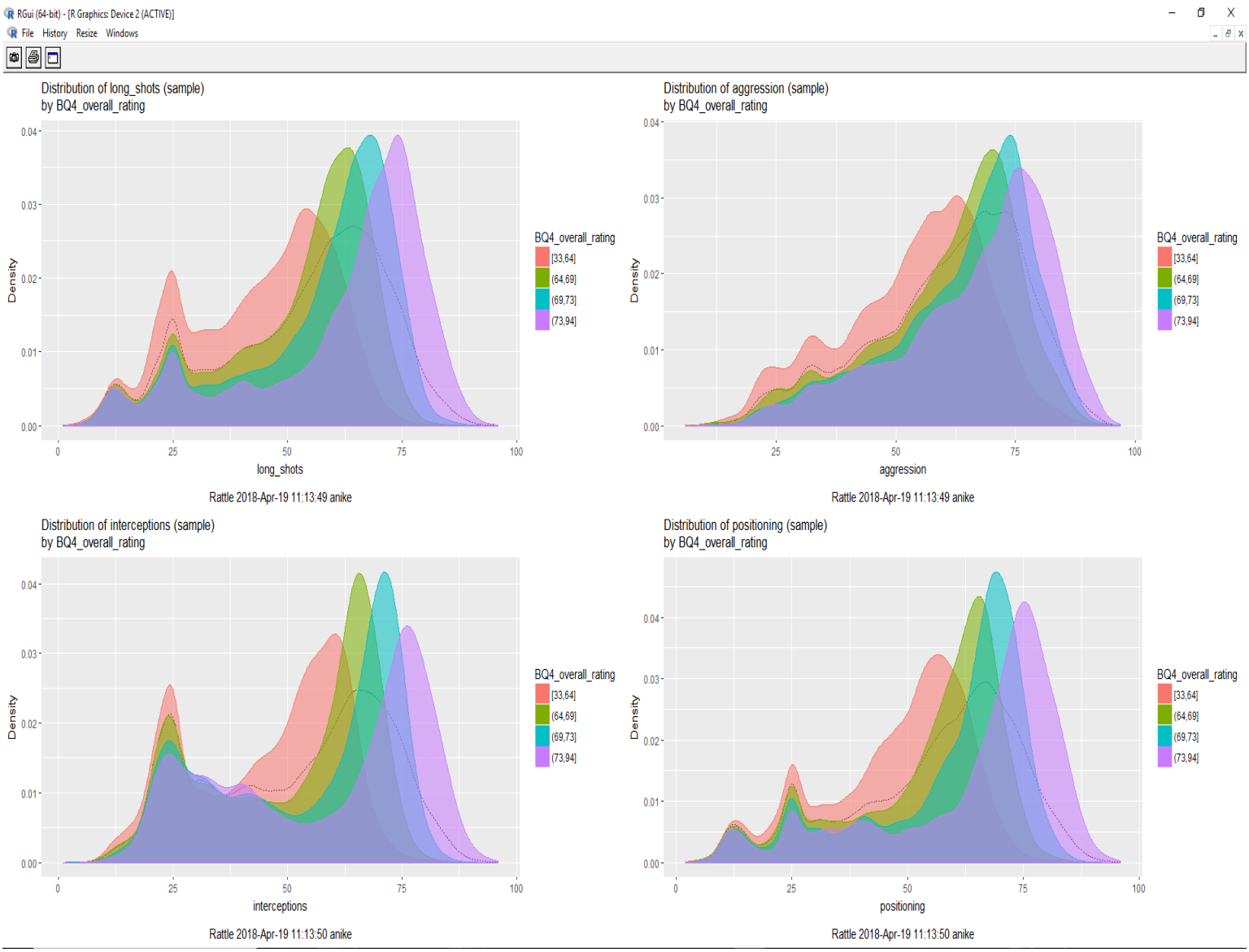
In the above plot, we observe the dot plots of preferred_foot, attacking_work_rate, defensive_work_rate with the overall rating. In the plots of attacking_work_rate and defensive_work_rate we observe that the values at zero do not yield in any useful results and other values can be used to get any meaningful results. However, in the plot of preferred_foot we see that the dot plot can yield meaningful results throughout the distribution.



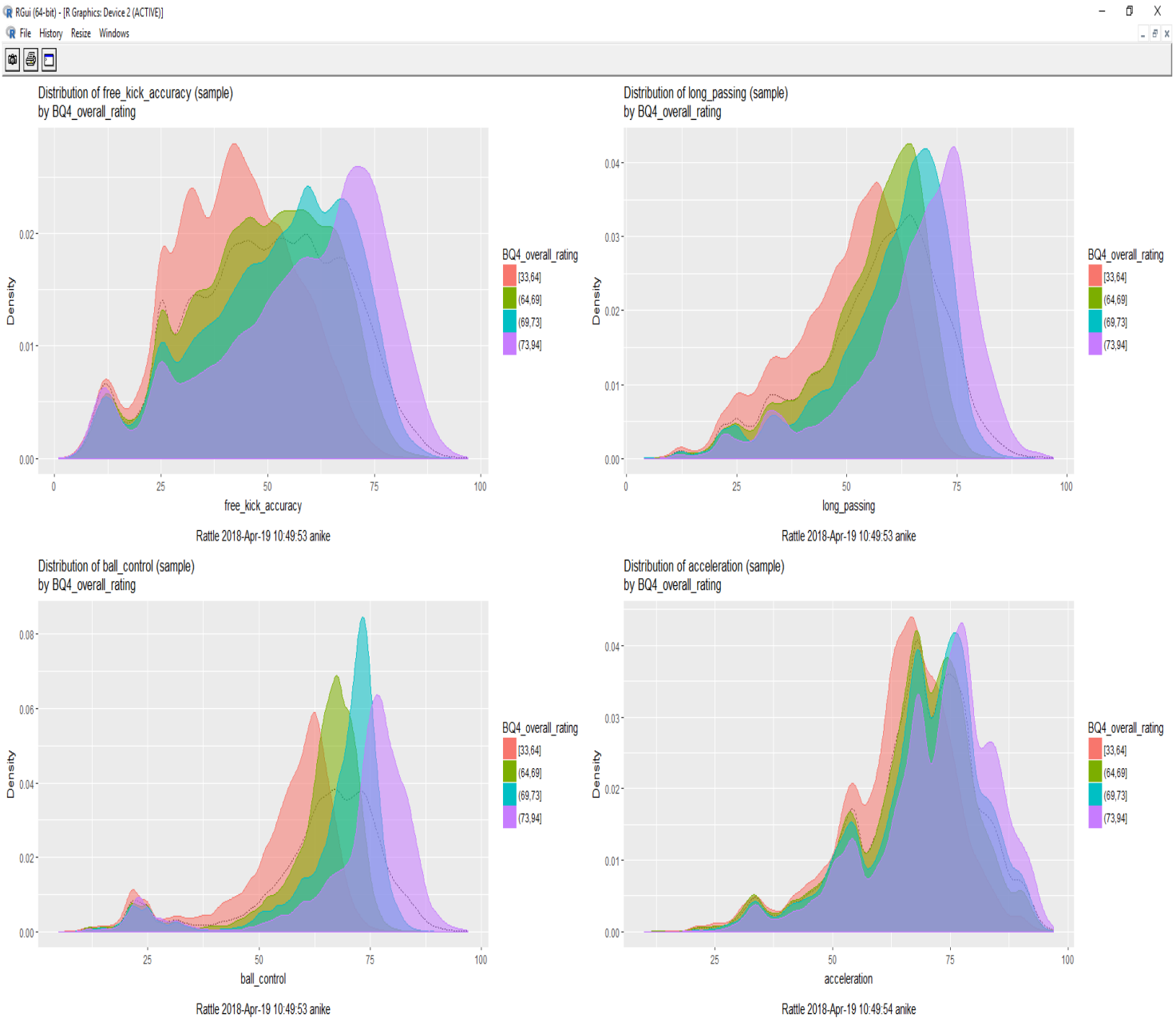
Above bar plot is a distribution of preferred_foot and attacking_work_rate against overall_rating. From this plot we can say that the overall_ratings for left and right footers is almost evenly distributed (with difference being in the range of 100-1000 observations). However, when the attacking_work_rate is high overall rating is in the range (73,94). From rest of the plot we cannot extract anything useful.



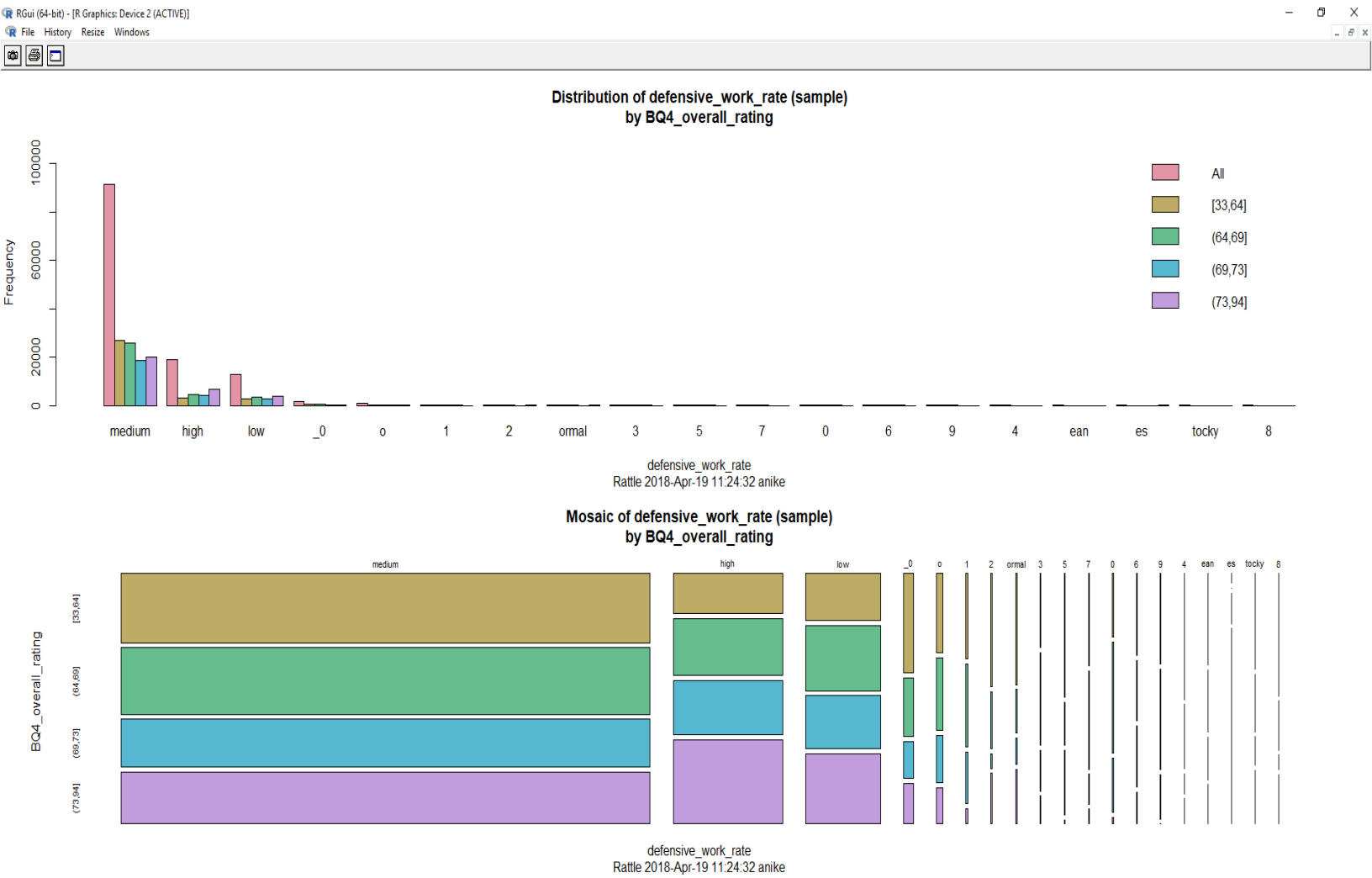
The above histogram plots are of potential, crossing, finishing, and heading_accuracy vs overall_rating. In the potential plot, we see that the attribute potential is almost directly proportional to the overall_rating, i.e. their ranges match. The plot of crossing and heading accuracy are rightly skewed indicating that these are qualities present in most of the players. Finishing has a distribution that is not particularly skewed to any one side, indicating that it is a quality present in many players but more in certain kind of players, like the attacking midfielders or strikers.



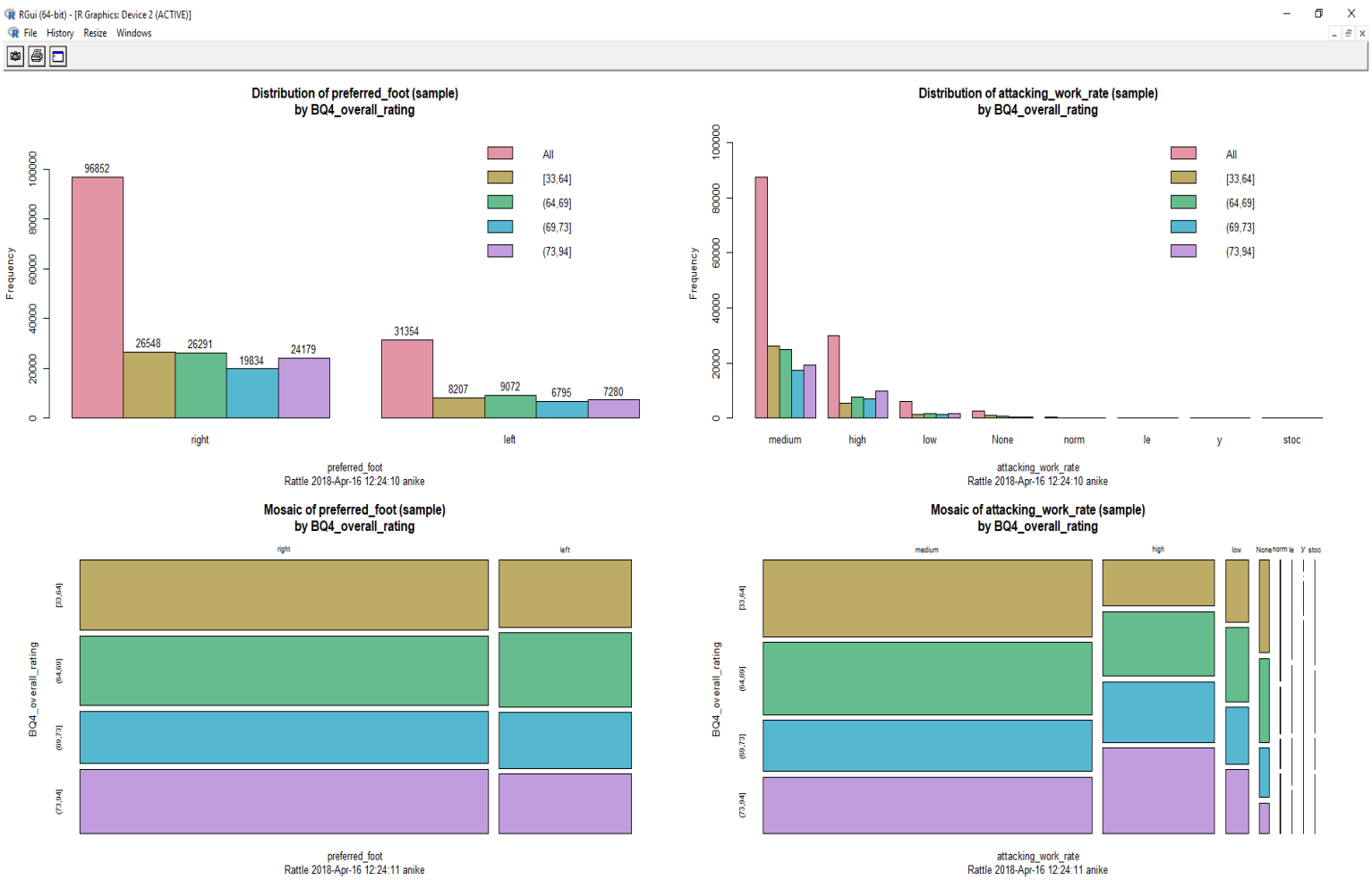
In the above plots we see that the long_shots, aggression, positioning, and interceptions vs overall_rating are all mostly skewed towards the right indicating that a certain type of players in each plot need not have higher value for that attribute. Positioning and Aggression is expected from all the players and hence we see that it has very less values on the left side. However, interceptions are a quality which may be less preferred for strikers than other players which is a reason why it has some values on the left side.



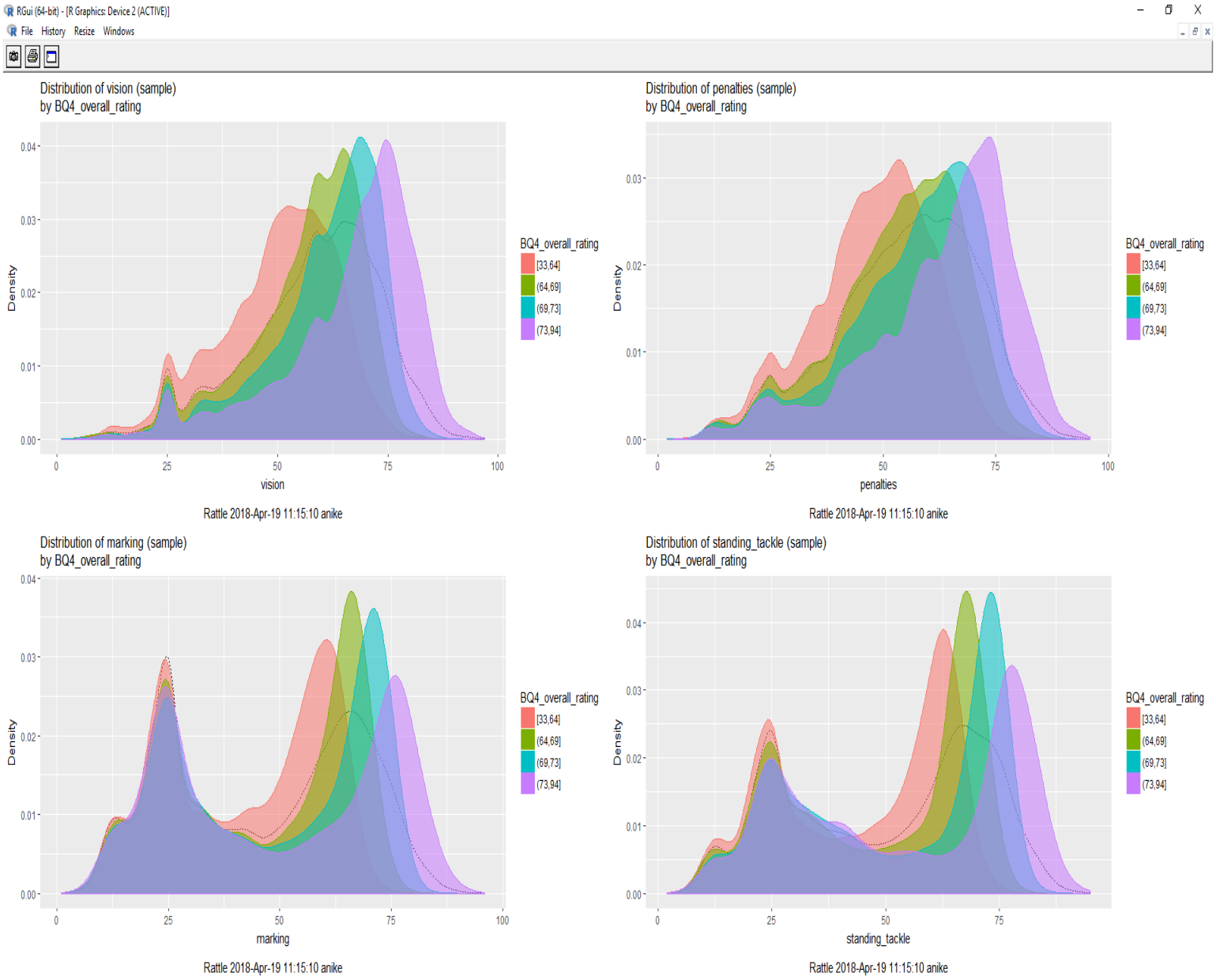
The above plots of free_kick_accuracy, long_passing, ball_control, and acceleration vs overall_ratings are all skewed towards the right (except for the attribute free_kick_accuracy which is almost evenly distributed) indicating that these are the qualities expected by each player. The free_kick_accuracy is expected more out of attacking players (attacking midfielders and strikers).



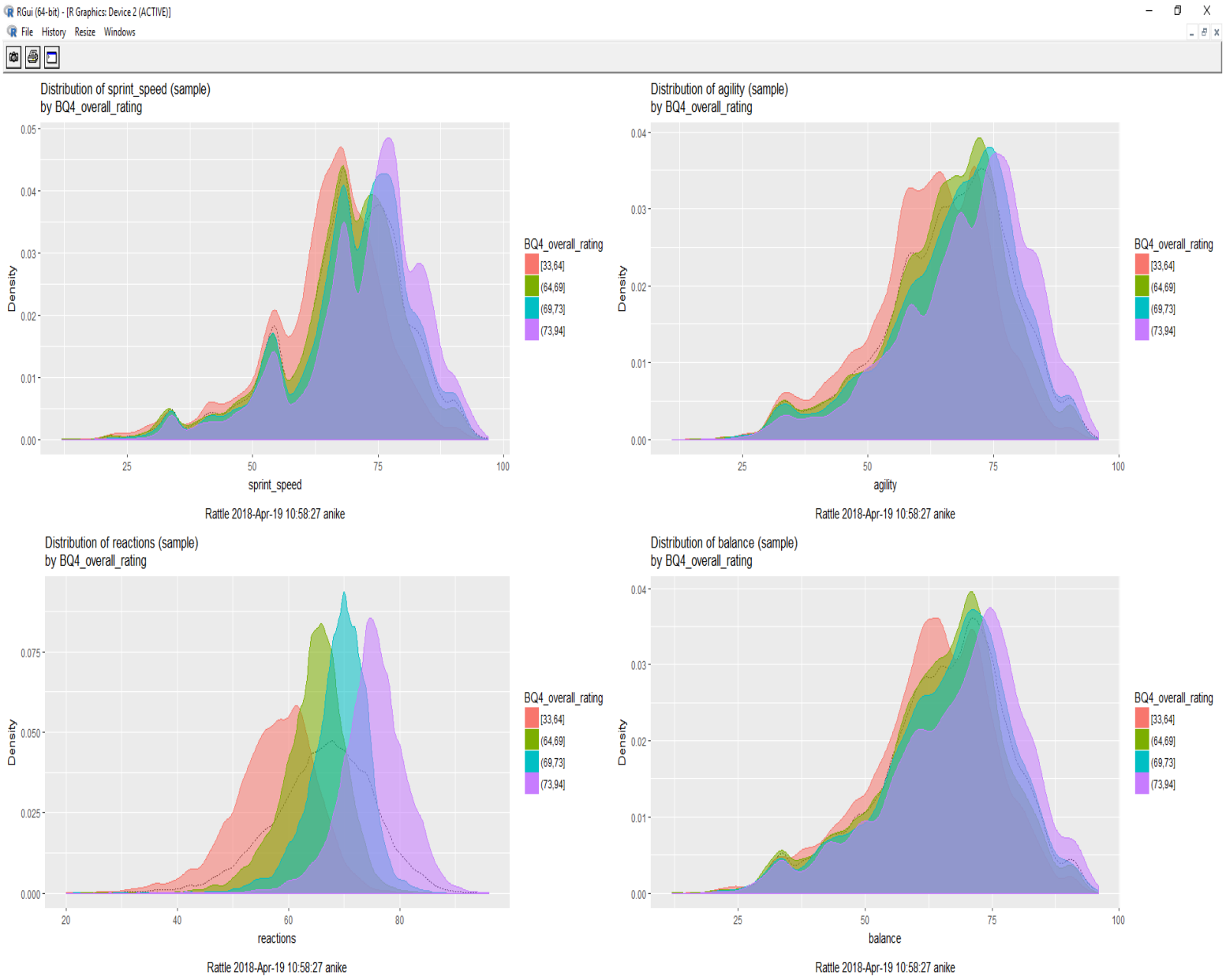
We observe the bar and mosaic plots of defensive_work_rate vs overall_rating. We see in the bar plot that when the defensive_work_rate is medium the overall_ratings are in the range 33,64 and 64,69. From the mosaic plot we observe that the frequency of medium defensive_work_rate is the maximum followed by high defensive_work_rate and then by the low defensive_work_rate. Thus, we can infer from this that most of the players have a medium defensive work rate.



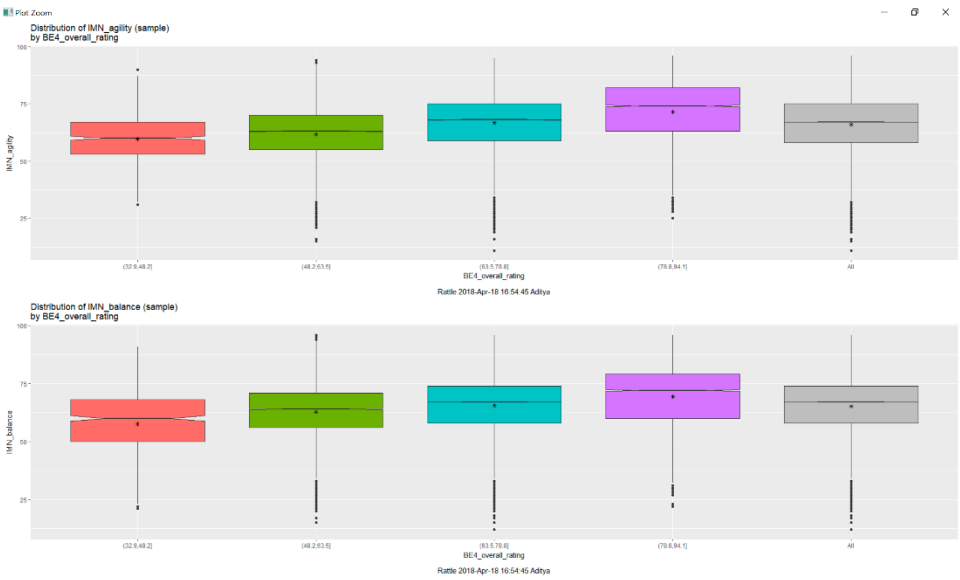
In the above mosaic and bar plots we plot preferred_foot and attacking_work_rate against overall rating. From the graphs we observe that there are more right footers than the left ones. Left footers have the maximum players in the range 64,69. When attacking_work_rate is the medium, we observe that most of the players have ratings in the range 33,64 and 64,69. We also observe that most of the players have medium attacking_work_rate, then followed by high attacking_work_rate and then by low attacking_work_rate.



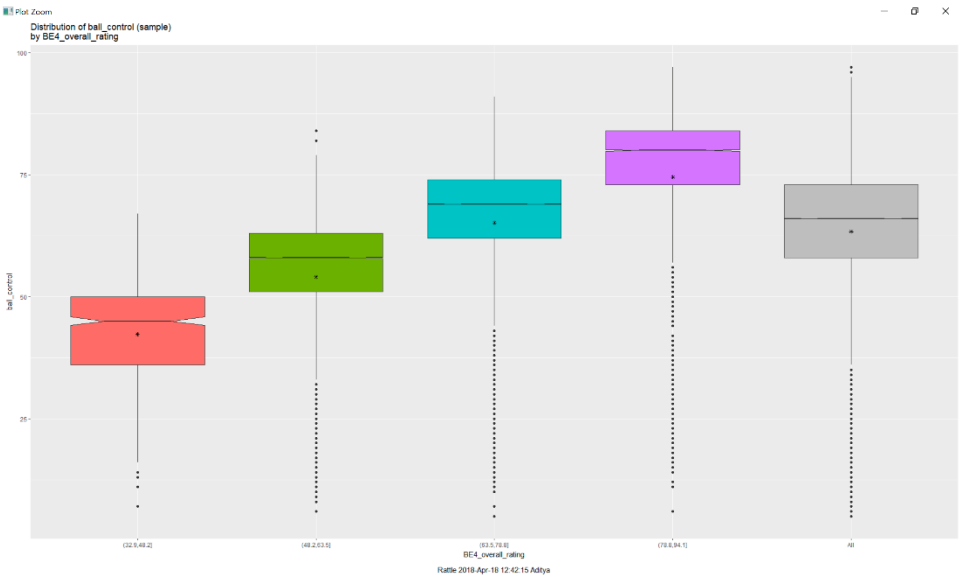
In the above observations we observe that the penalties and vision are skewed towards the right indicating that all the players possess this quality, the values on the left side can be the players who are goalkeeper as it is not essential for them to have higher values for these attributes. Marking and standing_tackle are qualities not expected from attacking midfielders or strikers who might represent the values on the left side.



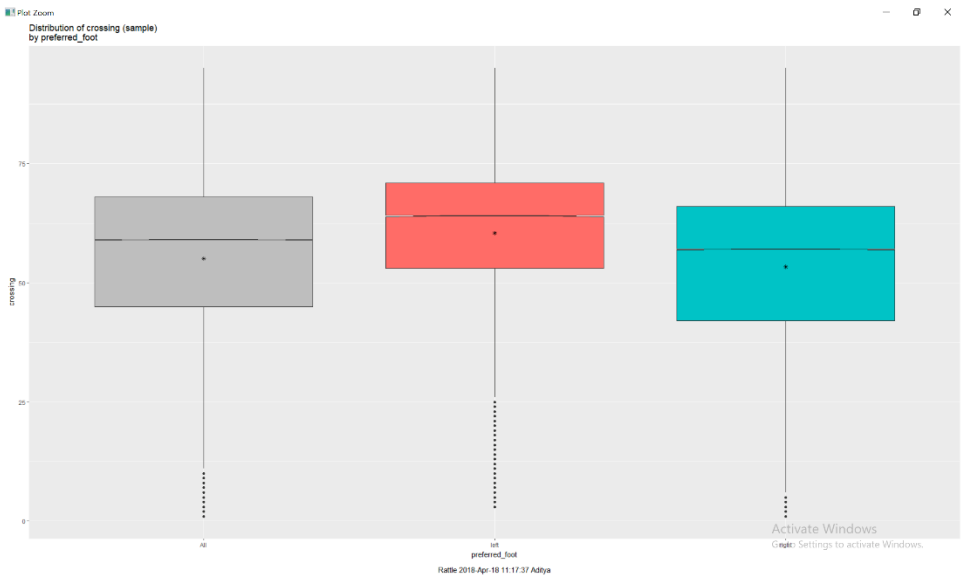
In these plots we see that balance, sprint and agility are skewed towards the right indicating these are the attributes expected from all the players and higher these values better are the overall_ratings. However, in the reactions plot, even though its skewed on the right, it can be observed that there are ranges of reactions in which each reaction occupies a slot in the overall_rating range.



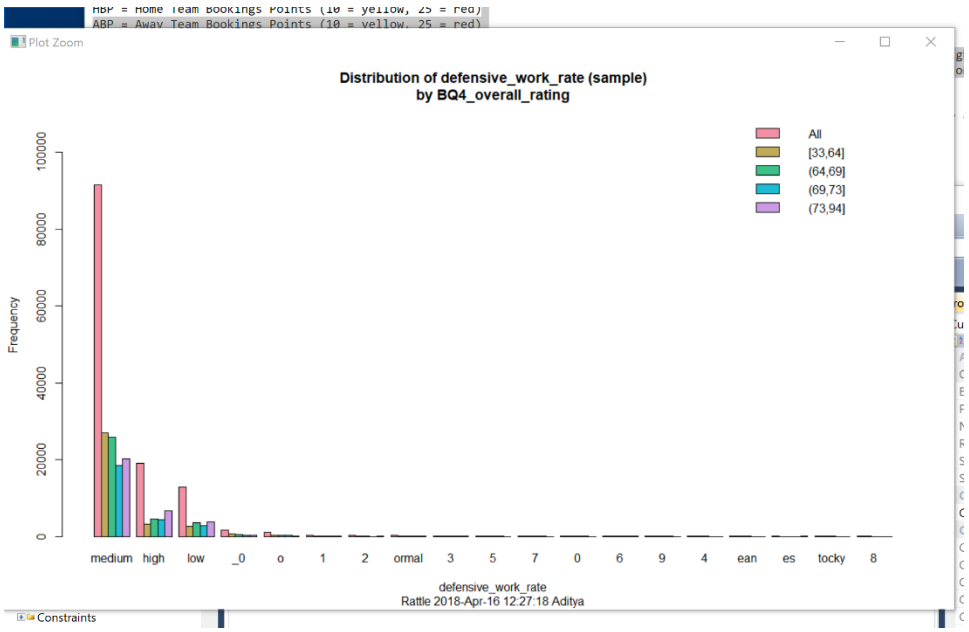
The above box plot shows distribution of player agility and balance with respect to overall rating. It can be observed that, when the overall rating is between (78.8 & 94.1) agility and balance has the greatest value which lies between 62.5 and 75.



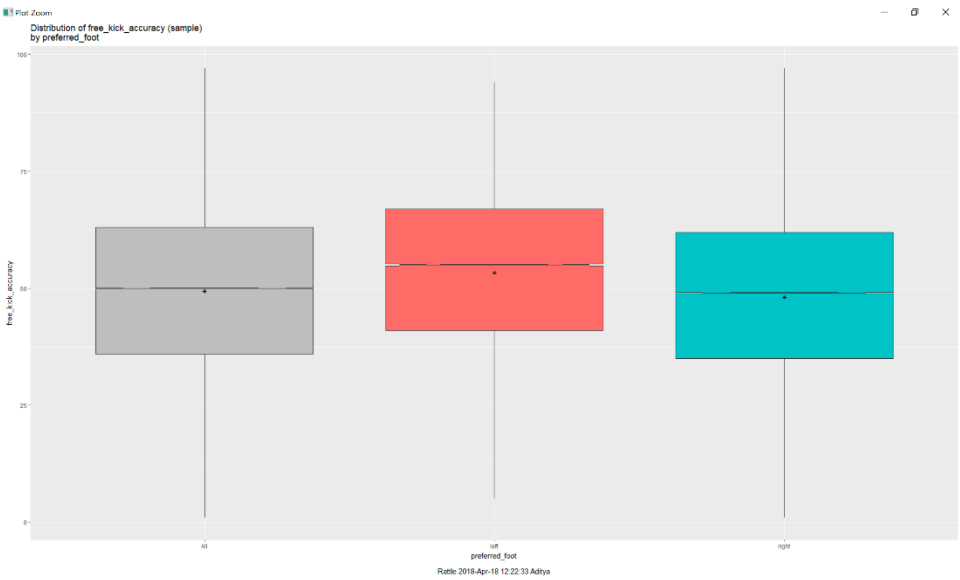
The box plot shows distribution of ball control with respect to overall rating. It can be observed that the value of ball control is the highest when the overall rating is between (78.8 & 94.1). The value of the mean is closer to 75. However, the median is greater 75.



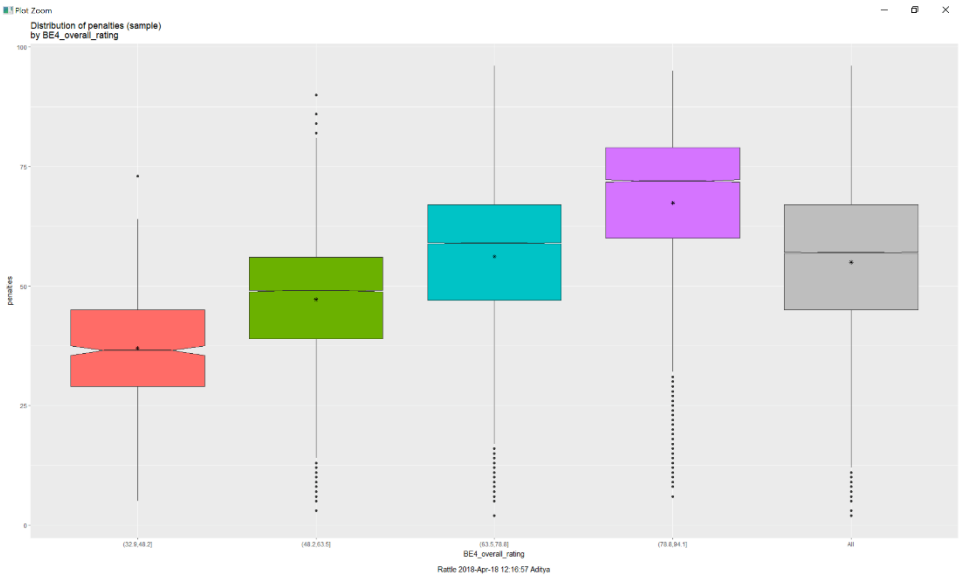
Visualizing distribution of crossing with respect to the preferred foot, With the mean/median of left greater than that of right. It can be inferred that most players prefer left foot over right while crossing.



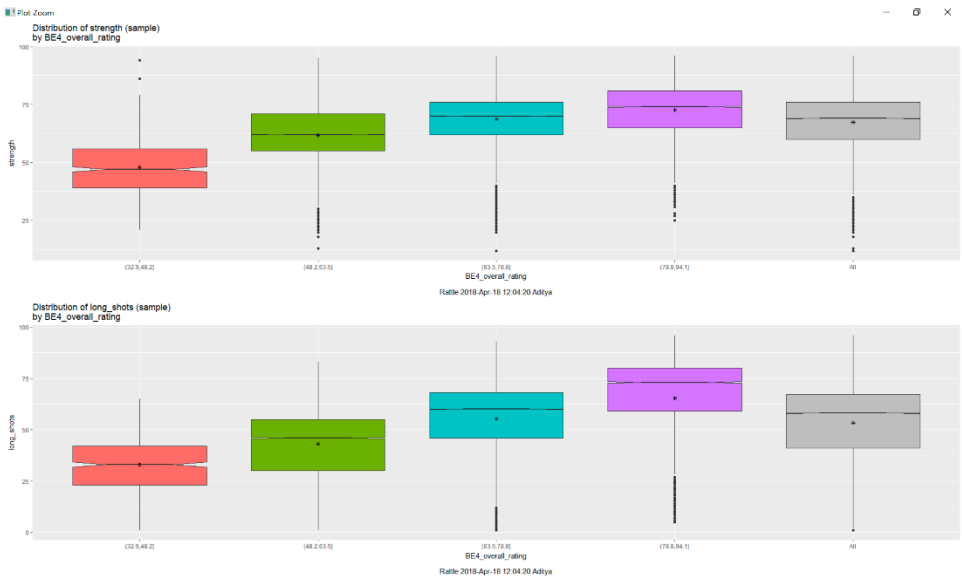
From the above plot it can be concluded that, more number of players with rating (33,64) prefer defensive work rate over those with rating (69,73) and (73,94).



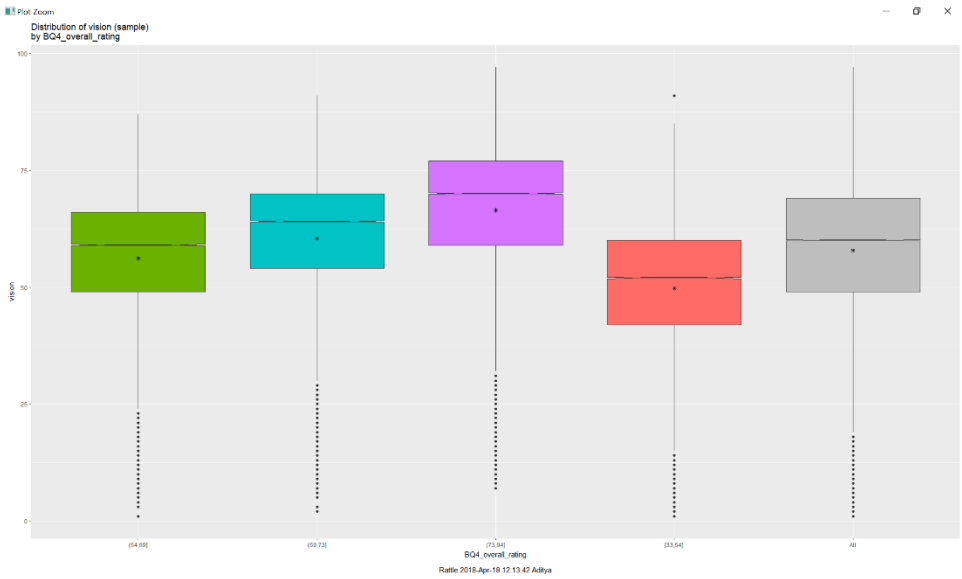
The inference of the above box plot is; players who prefer left foot over right have a greater accuracy when it comes to a free kick.



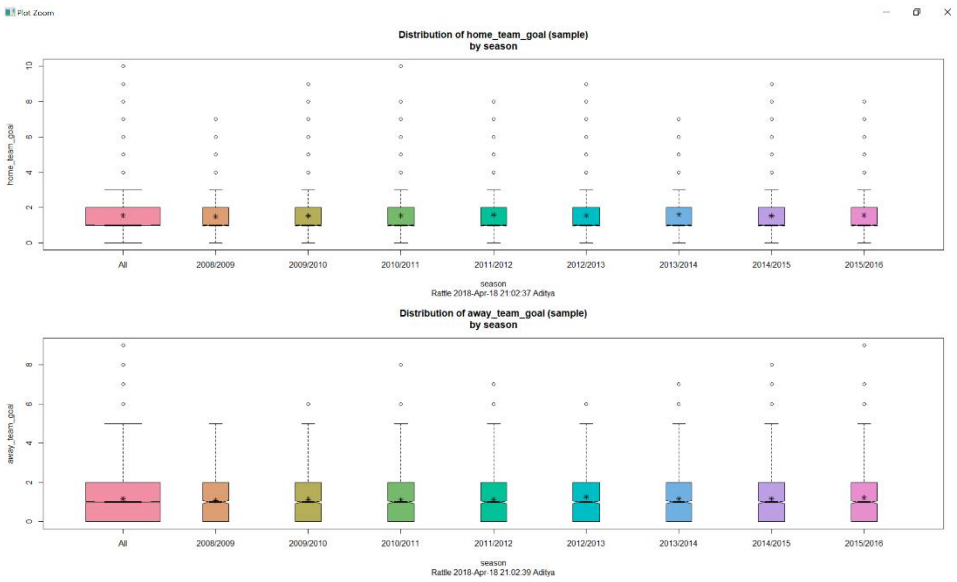
From the previous box plots, it was inferred that players with a high rating have a higher agility, balance and ball control. But, On the contrary there is a higher probability of a penalty among highly rated players.



The strength and ability to make a long shot is common among players with high rating and less common among players with low rating (<48.2).



All players have a vision rating between (50 & 75). But, the only exception is nearly half the players having a rating (<64) have a vision rating less than 50. From all the previous plots, it can be observed that the player with rating greater than 73 but less than 94 have greater value for all the attributes and hence are more likely to be expensive to buy. However, players with high rating are more susceptible to penalties.



In the above box plot, It can be inferred that there is no major difference among home team goals and away team goals from season 2009 to 2016.

PREDICTIVE ANALYSIS

TOP 20 GOALKEEPERS

```
select top 20 Player.player_name as Name, avg(Player_Attributes.gk_diving) as Diving,
avg(Player_Attributes.gk_handling) as Handling,

avg(Player_Attributes.gk_kicking) as Kicking, avg(Player_Attributes.gk_positioning) as Positioning,
avg(Player_Attributes.gk_reflexes) as Reflexes,

(2*avg(Player_Attributes.gk_diving) + avg(Player_Attributes.gk_handling) + avg(Player_Attributes.gk_kicking)
+ avg(Player_Attributes.gk_positioning) + 3*avg(Player_Attributes.gk_reflexes) )/8 as Rating,

Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) as Age,

(case
        when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) > 40
        then 'Retired'
 else
  case

         when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) <= 40

        then 'Playing'
 end

end) as Playing_Status

from Player_Attributes full outer join Player on Player_Attributes.player_api_id = Player.player_api_id

group by Player.player_name, Player.birthday

order by Rating desc
```

	Name	Diving	Handling	Kicking	Positioning	Reflexes	Rating	Age	Playing_Status
1	Iker Casillas	89	82	69	84	90	85	36	Playing
2	Robert Enke,30	83	84	78	86	87	84	40	Playing
3	Petr Cech	86	82	76	85	86	84	35	Playing
4	Manuel Neuer	86	81	86	82	86	84	32	Playing
5	Edwin van der Sar	83	86	85	91	79	83	47	Retired
6	Victor Valdes	83	78	80	86	86	83	36	Playing
7	Hugo Lloris	86	81	73	81	87	83	31	Playing
8	Gianluigi Buffon	87	82	68	90	83	82	40	Playing
9	Julio Cesar	84	80	71	83	85	82	38	Playing
10	Pepe Reina	82	78	87	82	84	82	35	Playing
11	Mickael Landreau	79	79	80	79	84	81	38	Playing
12	Rene Adler	85	79	70	80	85	81	33	Playing
13	Steve Mandanda	83	80	78	80	83	81	33	Playing
14	Joe Hart	83	76	76	78	85	81	31	Playing
15	David De Gea	83	76	81	76	84	81	27	Playing
16	Gregory Coupet	82	83	76	83	79	80	45	Retired
17	Frank Rost	80	78	78	83	81	80	44	Retired
18	Helton	81	78	75	79	84	80	39	Playing
19	Tim Howard	82	78	75	80	83	80	39	Playing
20	Diego Lopez	80	81	76	83	80	80	36	Playing

TOP 10 PLAYER STATISTICS

```
select top 10 Player.player_name as Name, avg(overall_rating) as Lifetime_Average,

avg(Player_Attributes.acceleration) as Acceleration, avg(Player_Attributes.aggression) as Aggresion,

avg(Player_Attributes.agility) as Agility, avg(Player_Attributes.balance) as Balance,

avg(Player_Attributes.ball_control) as Ball_control, avg(Player_Attributes.crossing) as Crossing,

avg(Player_Attributes.curve) as Curve, avg(Player_Attributes.dribbling) as Dribbling,

avg(Player_Attributes.finishing) as Finishing, avg(Player_Attributes.free_kick_accuracy) as
Free_Kick_Accuracy,

avg(Player_Attributes.heading_accuracy) as Header_Accuracy, avg(Player_Attributes.interceptions) as
Interceptions,

avg(Player_Attributes.jumping) as Jumping, avg(Player_Attributes.long_passing) as Long_Passing,

avg(Player_Attributes.long_shots) as Long_Shots, avg(Player_Attributes.marking) as Marking,

avg(Player_Attributes.penalties) as Penalties,avg(Player_Attributes.positioning) as Positioning,

avg(Player_Attributes.potential) as Potential, avg(Player_Attributes.reactions) as Reactions,

avg(Player_Attributes.short_passing) as Short_Passing, avg(Player_Attributes.shot_power) as Shot_Power,

avg(Player_Attributes.sliding_tackle) as Sliding_Tackle, avg(Player_Attributes.sprint_speed) as Sprint_Speed,

avg(Player_Attributes.stamina) as Stamina, avg(Player_Attributes.standing_tackle) as Standing_Tackle,

avg(Player_Attributes.strength) as Strength, avg(Player_Attributes.vision) as Vision,

avg(Player_Attributes.volleys) as Volleys

from Player_Attributes FULL OUTER JOIN Player on Player_Attributes.player_api_id = Player.player_api_id

where overall_rating > 0

group by Player_Attributes.player_api_id, Player.player_name

order by lifetime_average DESC;
```

	Name	Lifetime_Average	Acceleration	Aggresion	Agility	Balance	Ball_control	Crossing	Curve	Dribbling	Finishing	Free_Kick_Accuracy	Header_Accuracy	Interceptions	Jumping	Long_Passing	Long_Shots
1	Lionel Messi	92	95	49	94	92	95	83	87	96	92	84	69	34	69	76	86
2	Cristiano Ronaldo	91	91	61	92	74	93	83	88	92	91	81	85	35	93	71	89
3	Andres Iniesta	88	79	59	86	86	92	83	80	90	72	69	55	72	54	89	76
4	Franck Ribery	88	91	59	91	89	91	85	86	92	77	81	48	46	54	73	74
5	Zlatan Ibrahimovic	88	76	79	82	60	91	72	80	89	90	81	79	38	76	66	86
6	Arjen Robben	87	92	41	91	87	89	82	84	92	82	83	51	45	59	74	86
7	Wayne Rooney	87	77	90	78	78	86	78	83	82	87	79	78	48	79	83	85
8	Xavi Hernandez	87	70	55	83	87	93	86	84	82	73	86	54	76	53	91	76
9	Cesc Fabregas	86	69	51	68	75	89	83	77	82	77	80	70	67	67	91	79
10	David Silva	86	82	53	91	85	90	82	82	86	77	76	58	49	64	80	80

	Name	Lifetime_Average	Marking	Penalties	Positioning	Potential	Reactions	Short_Passing	Shot_Power	Sliding_Tackle	Sprint_Speed	Stamina	Standing_Tackle	Strength	Vision	Volleys
1	Lionel Messi	92	18	80	90	95	92	88	80	20	90	77	21	59	90	87
2	Cristiano Ronaldo	91	22	83	86	93	88	82	92	23	93	87	30	78	80	85
3	Andres Iniesta	88	58	71	88	90	89	93	66	56	76	77	59	60	92	74
4	Franck Ribery	88	18	81	84	89	86	87	75	26	89	74	25	63	86	82
5	Zlatan Ibrahimovic	88	20	88	83	90	82	84	91	26	80	75	43	90	82	90
6	Arjen Robben	87	26	81	85	88	86	84	82	25	91	71	24	62	82	84
7	Wayne Rooney	87	30	82	87	87	86	83	90	37	79	88	51	86	84	88
8	Xavi Hernandez	87	55	78	86	88	91	95	69	62	68	72	64	62	95	67
9	Cesc Fabregas	86	61	85	86	88	84	92	77	63	69	81	68	65	92	81
10	David Silva	86	26	76	83	89	84	89	76	29	75	70	31	56	90	80

TOP 20 STRIKERS

```
select top 20 Player.player_name as Name, avg(Player_Attributes.crossing) as Crossing,
avg(Player_Attributes.short_passing) as Short_Passing,

avg(Player_Attributes.vision) as Vision, avg(Player_Attributes.ball_control) as Ball_Control,

avg(Player_Attributes.penalties) as Penalties, avg(Player_Attributes.potential) as Potential,

avg(Player_Attributes.volleys) as Volleys, avg(Player_Attributes.finishing) as Finishing,
avg(Player_Attributes.dribbling) as Dribbling,

(avg(Player_Attributes.crossing) + avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) +
3*avg(Player_Attributes.ball_control) +

2*avg(Player_Attributes.penalties) + 2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) +
3*avg(Player_Attributes.finishing) +

3*avg(Player_Attributes.dribbling) )/18 as Rating,

Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) as Age,

(case

 when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) > 33

 then'Retired'

 else

  case

   when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) <= 33

   then 'Playing'

  end

end) as Playing_Status

from Player_Attributes full outer join Player on Player_Attributes.player_api_id = Player.player_api_id

group by Player.player_name, Player.birthday

order by Rating desc
```

	Name	Crossing	Short_Passing	Vision	Ball_Control	Penalties	Potential	Volleys	Finishing	Dribbling	Rating	Age	Playing_Status
1	Lionel Messi	83	88	90	95	80	95	87	92	96	90	30	Playing
2	Cristiano Ronaldo	83	82	80	93	83	93	85	91	92	88	33	Playing
3	Zlatan Ibrahimovic	72	84	82	91	88	90	90	90	89	87	36	Retired
4	Francesco Totti	83	90	89	91	89	82	89	82	83	86	41	Retired
5	Ronaldinho	83	87	87	93	85	84	83	75	89	85	38	Retired
6	Antonio Cassano	84	89	88	91	84	83	82	79	88	85	35	Retired
7	Franck Ribery	85	87	86	91	81	89	82	77	92	85	35	Retired
8	Robin van Persie	82	82	81	86	86	87	90	88	83	85	34	Retired
9	Arjen Robben	82	84	82	89	81	88	84	82	92	85	34	Retired
10	Cesc Fabregas	83	92	92	89	85	88	81	77	82	85	30	Playing
11	Sergio Aguero	70	82	83	89	82	88	85	88	90	85	29	Playing
12	Xavi Hernandez	86	95	95	93	78	88	67	73	82	84	38	Retired
13	David Villa	74	80	78	86	89	85	83	88	84	84	36	Retired
14	Carlos Tevez	74	82	82	87	83	86	87	87	85	84	34	Retired
15	Andres Iniesta	83	93	92	92	71	90	74	72	90	84	33	Playing
16	Wayne Rooney	78	83	84	86	82	87	88	87	82	84	32	Playing
17	David Silva	82	89	90	90	76	89	80	77	86	84	32	Playing
18	Luis Suarez	75	79	82	86	83	87	83	85	87	84	31	Playing
19	Antonio Di Natale	79	81	82	89	83	82	85	86	82	83	40	Retired
20	Dimitar Berbatov	77	84	82	87	82	83	87	87	82	83	37	Retired

```
select top 20 Player.player_name as Name, avg(Player_Attributes.crossing) as Crossing,
avg(Player_Attributes.short_passing) as Short_Passing,

avg(Player_Attributes.vision) as Vision, avg(Player_Attributes.ball_control) as Ball_Control,

avg(Player_Attributes.penalties) as Penalties, avg(Player_Attributes.potential) as Potential,

avg(Player_Attributes.volleys) as Volleys, avg(Player_Attributes.finishing) as Finishing,
avg(Player_Attributes.dribbling) as Dribbling,

(avg(Player_Attributes.crossing) + avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) +
3*avg(Player_Attributes.ball_control) +

2*avg(Player_Attributes.penalties) + 2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) +
3*avg(Player_Attributes.finishing) +

3*avg(Player_Attributes.dribbling) )/18 as Rating,

Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) as Age,

(case

 when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) > 33

 then'Retired'

 else

  case

   when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) <= 33

   then 'Playing'

  end
```

end) as Playing_Status

from Player_Attributes full outer join Player on Player_Attributes.player_api_id = Player.player_api_id

where Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) > 33

group by Player.player_name, Player.birthday

order by Rating desc

	Name	Crossing	Short_Passing	Vision	Ball_Control	Penalties	Potential	Volleys	Finishing	Dribbling	Rating	Age	Playing_Status
1	Zlatan Ibrahimovic	72	84	82	91	88	90	90	90	89	87	36	Retired
2	Francesco Totti	83	90	89	91	89	82	89	82	83	86	41	Retired
3	Ronaldinho	83	87	87	93	85	84	83	75	89	85	38	Retired
4	Antonio Cassano	84	89	88	91	84	83	82	79	88	85	35	Retired
5	Franck Ribery	85	87	86	91	81	89	82	77	92	85	35	Retired
6	Robin van Persie	82	82	81	86	86	87	90	88	83	85	34	Retired
7	Arjen Robben	82	84	82	89	81	88	84	82	92	85	34	Retired
8	Xavi Hernandez	86	95	95	93	78	88	67	73	82	84	38	Retired
9	David Villa	74	80	78	86	89	85	83	88	84	84	36	Retired
10	Carlos Tevez	74	82	82	87	83	86	87	87	85	84	34	Retired
11	Antonio Di Natale	79	81	82	89	83	82	85	86	82	83	40	Retired
12	Dimitar Berbatov	77	84	82	87	82	83	87	87	82	83	37	Retired
13	Kaka	79	85	84	87	85	83	84	79	85	83	35	Retired
14	Ryan Giggs	88	84	86	85	86	83	79	74	82	82	44	Retired
15	Ruud van Nistelr...	65	79	76	83	86	87	87	90	78	82	41	Retired
16	Jose Maria Guti	81	89	92	86	81	84	80	75	80	82	41	Retired
17	Raul	68	83	85	83	80	87	83	86	81	82	40	Retired
18	Frank Lampard	80	84	82	83	87	82	86	83	77	82	39	Retired
19	Mirko Vucinic	71	83	83	88	80	82	80	80	84	82	34	Retired
20	Alessandro Del ...	74	78	84	86	84	79	78	82	81	81	43	Retired

select top 20 Player.player_name as Name, avg(Player_Attributes.crossing) as Crossing,
avg(Player_Attributes.short_passing) as Short_Passing,

avg(Player_Attributes.vision) as Vision, avg(Player_Attributes.ball_control) as Ball_Control,

avg(Player_Attributes.penalties) as Penalties, avg(Player_Attributes.potential) as Potential,

avg(Player_Attributes.volleys) as Volleys, avg(Player_Attributes.finishing) as Finishing,
avg(Player_Attributes.dribbling) as Dribbling,

(avg(Player_Attributes.crossing) + avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) +
3*avg(Player_Attributes.ball_control) +

2*avg(Player_Attributes.penalties) + 2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) +
3*avg(Player_Attributes.finishing) +

3*avg(Player_Attributes.dribbling))/18 as Rating,

Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) as Age,

(case

```
 when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) > 33
 then'Retired'
 else
  case
   when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) <= 33
   then 'Playing'
  end
end) as Playing_Status
from Player_Attributes full outer join Player on Player_Attributes.player_api_id = Player.player_api_id
where Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) <= 33
group by Player.player_name, Player.birthday
order by Rating desc
```

	Name	Crossing	Short_Passing	Vision	Ball_Control	Penalties	Potential	Volleys	Finishing	Dribbling	Rating	Age	Playing_Status
1	Lionel Messi	83	88	90	95	80	95	87	92	96	90	30	Playing
2	Cristiano Ronaldo	83	82	80	93	83	93	85	91	92	88	33	Playing
3	Cesc Fabregas	83	92	92	89	85	88	81	77	82	85	30	Playing
4	Sergio Aguero	70	82	83	89	82	88	85	88	90	85	29	Playing
5	Andres Iniesta	83	93	92	92	71	90	74	72	90	84	33	Playing
6	Wayne Rooney	78	83	84	86	82	87	88	87	82	84	32	Playing
7	David Silva	82	89	90	90	76	89	80	77	86	84	32	Playing
8	Luis Suarez	75	79	82	86	83	87	83	85	87	84	31	Playing
9	Neymar	70	75	76	90	79	90	81	83	91	83	26	Playing
10	Wesley Sneijder	83	88	89	87	78	86	80	71	83	82	33	Playing
11	Bastian Schwei...	81	88	86	84	79	87	84	76	82	82	33	Playing
12	Luka Modric	75	87	89	87	79	88	76	72	83	82	32	Playing
13	Samir Nasri	80	85	86	87	80	86	73	76	86	82	30	Playing
14	Karim Benzema	74	82	79	85	80	87	77	87	82	82	30	Playing
15	Mesut Oezil	80	86	90	88	72	87	77	72	85	82	29	Playing
16	Eden Hazard	75	81	84	86	78	89	76	76	87	82	27	Playing
17	James Rodriguez	81	81	85	84	80	89	77	78	85	82	26	Playing
18	Diego	79	86	83	86	78	84	81	72	86	81	33	Playing
19	Hernanes	78	84	84	86	80	83	79	72	83	81	32	Playing
20	Hulk	77	78	78	82	81	85	72	83	83	81	31	Playing

TOP 20 DEFENDERS

```
select top 20 Player.player_name as Name, avg(Player_Attributes.heading_accuracy) as Heading_Accuracy,
avg(Player_Attributes.interceptions) as Interceptions,

avg(Player_Attributes.sliding_tackle) as Sliding_Tackles, avg(Player_Attributes.standing_tackle) as
Standing_Tackles,

avg(Player_Attributes.aggression) as Aggression, avg(Player_Attributes.strength) as Strength,

(avg(Player_Attributes.heading_accuracy) + 3*avg(Player_Attributes.interceptions) +
2*avg(Player_Attributes.sliding_tackle) +

2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) +
2*avg(Player_Attributes.strength) )/11 as Rating,

Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) as Age,

(case

 when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) > 33

 then'Retired'

 else

  case

   when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) <= 33

   then 'Playing'

  end

end) as Playing_Status

from Player_Attributes full outer join Player on Player_Attributes.player_api_id = Player.player_api_id

group by Player.player_name, Player.birthday

order by Rating desc
```

	Name	Heading_Accuracy	Interceptions	Sliding_Tackles	Standing_Tackles	Aggression	Strength	Rating	Age	Playing_Status
1	Carles Puyol	84	89	87	88	88	88	87	40	Retired
2	John Terry	91	84	84	87	89	87	86	37	Retired
3	Nemanja Vidic	89	83	83	88	89	90	86	36	Retired
4	Giorgio Chiellini	79	86	88	88	85	87	86	33	Playing
5	Pepe	81	84	92	85	88	83	85	35	Retired
6	Sergio Ramos	87	83	89	87	84	82	85	32	Playing
7	Patrick Vieira	83	86	80	84	87	88	84	41	Retired
8	Andrea Barzagli	77	88	85	87	77	83	84	36	Retired
9	Daniele De Rossi	85	86	83	85	84	82	84	34	Retired
10	Javier Mascherano	62	91	88	90	90	75	84	33	Playing
11	Per Mertesacker	90	85	82	87	70	88	84	33	Playing
12	Gary Neville	77	89	83	83	86	76	83	43	Retired
13	Mark van Bommel	79	83	82	80	93	86	83	40	Retired
14	Jamie Carragher	83	85	84	83	85	83	83	40	Retired
15	Walter Samuel	73	87	82	85	80	84	83	40	Retired
16	Luisao	85	82	80	84	81	87	83	37	Retired
17	Branislav Ivanovic	85	81	84	84	84	83	83	34	Retired
18	Thiago Silva	80	85	88	87	76	79	83	33	Playing
19	Vincent Kompany	79	83	84	87	75	88	83	32	Playing
20	Gerard Pique	82	83	87	87	70	83	83	31	Playing

```
select top 20 Player.player_name as Name, avg(Player_Attributes.heading_accuracy) as Heading_Accuracy,
avg(Player_Attributes.interceptions) as Interceptions,

avg(Player_Attributes.sliding_tackle) as Sliding_Tackles, avg(Player_Attributes.standing_tackle) as
Standing_Tackles,

avg(Player_Attributes.aggression) as Aggression, avg(Player_Attributes.strength) as Strength,

(avg(Player_Attributes.heading_accuracy) + 3*avg(Player_Attributes.interceptions) +
2*avg(Player_Attributes.sliding_tackle) +

2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) +
2*avg(Player_Attributes.strength) )/11 as Rating,

Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) as Age,

(case

 when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) > 33

 then'Retired'

 else

  case

   when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) <= 33

   then 'Playing'

  end

end) as Playing_Status
```

```
from Player_Attributes full outer join Player on Player_Attributes.player_api_id = Player.player_api_id

where Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) <= 33

group by Player.player_name, Player.birthday

order by Rating desc
```

	Name	Heading_Accuracy	Interceptions	Sliding_Tackles	Standing_Tackles	Aggression	Strength	Rating	Age	Playing_Status
1	Giorgio Chiellini	79	86	88	88	85	87	86	33	Playing
2	Sergio Ramos	87	83	89	87	84	82	85	32	Playing
3	Javier Mascherano	62	91	88	90	90	75	84	33	Playing
4	Per Mertesacker	90	85	82	87	70	88	84	33	Playing
5	Thiago Silva	80	85	88	87	76	79	83	33	Playing
6	Vincent Kompany	79	83	84	87	75	88	83	32	Playing
7	Gerard Pique	82	83	87	87	70	83	83	31	Playing
8	Raul Albiol	79	83	83	83	79	82	82	32	Playing
9	Javier Martinez	83	83	80	81	84	86	82	29	Playing
10	Mats Hummels	85	84	84	84	69	82	82	29	Playing
11	Daniel Agger	82	81	83	84	79	79	81	33	Playing
12	Martin Skrtel	85	78	82	82	84	83	81	33	Playing
13	Ezequiel Garay	82	80	81	85	75	83	81	31	Playing
14	Mehdi Benatia	80	82	81	84	81	82	81	31	Playing
15	Micah Richards	84	74	81	80	85	89	81	29	Playing
16	Robert Huth	83	75	75	79	85	92	80	33	Playing
17	Miranda	81	79	83	82	82	78	80	33	Playing
18	Nigel de Jong	67	83	82	81	87	78	80	33	Playing
19	Adil Rami	78	79	79	82	79	87	80	32	Playing
20	Diego Godin	82	80	87	79	80	77	80	32	Playing

```
select top 20 Player.player_name as Name, avg(Player_Attributes.heading_accuracy) as Heading_Accuracy,
avg(Player_Attributes.interceptions) as Interceptions,

avg(Player_Attributes.sliding_tackle) as Sliding_Tackles, avg(Player_Attributes.standing_tackle) as
Standing_Tackles,

avg(Player_Attributes.aggression) as Aggression, avg(Player_Attributes.strength) as Strength,

(avg(Player_Attributes.heading_accuracy) + 3*avg(Player_Attributes.interceptions) +
2*avg(Player_Attributes.sliding_tackle) +

2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) +
2*avg(Player_Attributes.strength) )/11 as Rating,

Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) as Age,

(case

 when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) > 33

 then'Retired'

 else
```

```
 case
  when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) <= 33
  then 'Playing'
 end
end) as Playing_Status
from Player_Attributes full outer join Player on Player_Attributes.player_api_id = Player.player_api_id
where Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) > 33
group by Player.player_name, Player.birthday
order by Rating desc
```

	Name	Heading_Accuracy	Interceptions	Sliding_Tackles	Standing_Tackles	Aggression	Strength	Rating	Age	Playing_Status
1	Carles Puyol	84	89	87	88	88	88	87	40	Retired
2	John Terry	91	84	84	87	89	87	86	37	Retired
3	Nemanja Vidic	89	83	83	88	89	90	86	36	Retired
4	Pepe	81	84	92	85	88	83	85	35	Retired
5	Patrick Vieira	83	86	80	84	87	88	84	41	Retired
6	Andrea Barzagli	77	88	85	87	77	83	84	36	Retired
7	Daniele De Rossi	85	86	83	85	84	82	84	34	Retired
8	Gary Neville	77	89	83	83	86	76	83	43	Retired
9	Mark van Bommel	79	83	82	80	93	86	83	40	Retired
10	Jamie Carragher	83	85	84	83	85	83	83	40	Retired
11	Walter Samuel	73	87	82	85	80	84	83	40	Retired
12	Luisao	85	82	80	84	81	87	83	37	Retired
13	Branislav Ivanovic	85	81	84	84	84	83	83	34	Retired
14	Alessandro Nesta	80	86	82	86	75	81	82	42	Retired
15	Gennaro Gattuso	56	84	85	86	92	83	82	40	Retired
16	Rio Ferdinand	83	82	83	86	80	82	82	39	Retired
17	Gabriel Milito	83	82	78	85	83	82	82	37	Retired
18	Ledley King	85	82	76	82	84	88	82	37	Retired
19	Kolo Toure	81	79	84	83	84	86	82	37	Retired
20	Alex	84	79	80	83	81	90	82	35	Retired

TOP 10 AND BOTTOM 10 LIFETIME AVERAGE RATINGS

```
select top 10 Player_Attributes.player_api_id, Player.player_name, avg(overall_rating) as lifetime_average
from Player_Attributes FULL OUTER JOIN Player on Player_Attributes.player_api_id = Player.player_api_id
where overall_rating > 0
group by Player_Attributes.player_api_id, Player.player_name
order by lifetime_average DESC;
```

	player_api_id	player_name	lifetime_average
1	30981	Lionel Messi	92
2	30893	Cristiano Ronaldo	91
3	30955	Andres Iniesta	88
4	30924	Franck Ribery	88
5	35724	Zlatan Ibrahimovic	88
6	30834	Arjen Robben	87
7	30829	Wayne Rooney	87
8	39854	Xavi Hernandez	87
9	30613	Cesc Fabregas	86
10	37459	David Silva	86

```
select top 10 Player_Attributes.player_api_id, Player.player_name, avg(overall_rating) as lifetime_average
from Player_Attributes FULL OUTER JOIN Player on Player_Attributes.player_api_id = Player.player_api_id
where overall_rating > 0
group by Player_Attributes.player_api_id, Player.player_name
order by lifetime_average ASC;
```

	player_api_id	player_name	lifetime_average
1	67396	Frederic Schaub	43
2	177471	Gianluca D'Angelo	43
3	107416	Badis Lebbihi	45
4	112561	Boubacar Mansaly	45
5	127141	Jordan Kirkpatrick	45
6	22343	Benjamin Fischer	46
7	107283	Liam Hughes	46
8	113186	Emir Sinanovic	47
9	177472	Enzo Ruiz	47
10	42149	Florian Berisha	47

TEAM MATCHES WITH LEAGUE DETAILS AND MATCH RESULTS

```
select top 25 M.id, Country.name as Country, League.name as 'League Name', m.season as Season,

substring(m.date,1,10) AS Date, HomeTeam.team_long_name as 'Home Team', AwayTeam.team_long_name as
'Away Team',

M.home_team_goal as 'Home Team Goals', M.away_team_goal as 'Away Team Goals',

(case

 when M.home_team_goal > M.away_team_goal

 then 'Home team won'

 else

  case

   when M.home_team_goal < M.away_team_goal

   then 'Away team won'

   else

    case

     when M.home_team_goal = M.away_team_goal

     then 'Match Drawn'

    end

  end

end) as 'Win/Loss/Draw Class'

from match as M

inner join team as HomeTeam on M.home_team_api_id = HomeTeam.team_api_id

inner join team as AwayTeam on M.away_team_api_id = AwayTeam.team_api_id

inner join Country on M.country_id = Country.id

inner join League on M.country_id = League.id
```

	id	Country	League Name	Season	Date	Home Team	Away Team	Home Team Goals	Away Team Goals	Win/Loss/Draw Class
1	25979	Switzerland	Switzerland Super League	2015/2016	2015-09-23	BSC Young Boys	FC Basel	4	3	Home team won
2	25977	Switzerland	Switzerland Super League	2015/2016	2015-09-23	Grasshopper Club Zürich	FC Sion	2	0	Home team won
3	25976	Switzerland	Switzerland Super League	2015/2016	2015-09-23	FC Vaduz	FC Luzern	1	2	Away team won
4	25975	Switzerland	Switzerland Super League	2015/2016	2015-09-22	FC St. Gallen	FC Thun	1	0	Home team won
5	25973	Switzerland	Switzerland Super League	2015/2016	2015-09-13	FC Zürich	FC Thun	3	3	Match Drawn
6	25972	Switzerland	Switzerland Super League	2015/2016	2015-09-13	FC Luzern	Grasshopper Club Zürich	3	3	Match Drawn
7	25971	Switzerland	Switzerland Super League	2015/2016	2015-09-12	BSC Young Boys	FC Vaduz	4	0	Home team won
8	25970	Switzerland	Switzerland Super League	2015/2016	2015-09-12	FC Basel	FC St. Gallen	2	1	Home team won
9	25969	Switzerland	Switzerland Super League	2015/2016	2015-08-30	Grasshopper Club Zürich	BSC Young Boys	3	2	Home team won
10	25968	Switzerland	Switzerland Super League	2015/2016	2015-08-30	FC Basel	FC Zürich	3	1	Home team won
11	25967	Switzerland	Switzerland Super League	2015/2016	2015-08-30	FC Thun	FC Sion	0	2	Away team won
12	25966	Switzerland	Switzerland Super League	2015/2016	2015-08-29	FC Vaduz	FC St. Gallen	1	0	Home team won
13	25964	Switzerland	Switzerland Super League	2015/2016	2015-08-22	FC Zürich	FC Luzern	2	5	Away team won
14	25963	Switzerland	Switzerland Super League	2015/2016	2015-08-23	FC Thun	FC Vaduz	1	0	Home team won
15	25962	Switzerland	Switzerland Super League	2015/2016	2015-08-23	FC St. Gallen	Grasshopper Club Zürich	0	2	Away team won
16	25961	Switzerland	Switzerland Super League	2015/2016	2015-08-23	FC Sion	BSC Young Boys	1	3	Away team won
17	25959	Switzerland	Switzerland Super League	2015/2016	2015-08-13	FC Vaduz	Grasshopper Club Zürich	3	3	Match Drawn
18	25958	Switzerland	Switzerland Super League	2015/2016	2015-08-13	FC Sion	FC Zürich	3	1	Home team won
19	25957	Switzerland	Switzerland Super League	2015/2016	2015-08-12	FC Luzern	FC St. Gallen	0	1	Away team won
20	25955	Switzerland	Switzerland Super League	2015/2016	2015-08-12	FC Basel	FC Thun	3	1	Home team won
21	25954	Switzerland	Switzerland Super League	2015/2016	2015-08-09	BSC Young Boys	FC Thun	3	1	Home team won
22	25953	Switzerland	Switzerland Super League	2015/2016	2015-08-09	FC Vaduz	FC Sion	1	1	Match Drawn
23	25952	Switzerland	Switzerland Super League	2015/2016	2015-08-09	FC St. Gallen	FC Zürich	0	2	Away team won
24	25951	Switzerland	Switzerland Super League	2015/2016	2015-08-08	FC Luzern	FC Basel	1	3	Away team won
25	25949	Switzerland	Switzerland Super League	2015/2016	2016-05-25	FC Zürich	FC Vaduz	3	1	Home team won

```
select top 25 M.id, Country.name as Country, League.name as 'League Name', m.season as Season,

substring(m.date,1,10) AS Date, HomeTeam.team_long_name as 'Home Team', AwayTeam.team_long_name as
'Away Team',

M.home_team_goal as 'Home Team Goals', M.away_team_goal as 'Away Team Goals',

(case

 when M.home_team_goal > M.away_team_goal

 then 'Home team won'

 else

  case

   when M.home_team_goal < M.away_team_goal

   then 'Away team won'

   else
```

```
   case
    when M.home_team_goal = M.away_team_goal
    then 'Match Drawn'
   end
 end
end) as 'Win/Loss/Draw Class'
from match as M
inner join team as HomeTeam on M.home_team_api_id = HomeTeam.team_api_id
inner join team as AwayTeam on M.away_team_api_id = AwayTeam.team_api_id
inner join Country on M.country_id = Country.id
inner join League on M.country_id = League.id
order by id DESC
```

	id	Country	League Name	Season	Date	Home Team	Away Team	Home Team Goals	Away Team Goals	Win/Loss/Draw Class
1	1	Belgium	Belgium Jupiler League	2008/2009	2008-08-17	KRC Genk	Beerschot AC	1	1	Match Drawn
2	2	Belgium	Belgium Jupiler League	2008/2009	2008-08-16	SV Zulte-Waregem	Sporting Lokeren	0	0	Match Drawn
3	3	Belgium	Belgium Jupiler League	2008/2009	2008-08-16	KSV Cercle Brugge	RSC Anderlecht	0	3	Away team won
4	4	Belgium	Belgium Jupiler League	2008/2009	2008-08-17	KAA Gent	RAEC Mons	5	0	Home team won
5	6	Belgium	Belgium Jupiler League	2008/2009	2008-09-24	KV Mechelen	Club Brugge KV	1	1	Match Drawn
6	7	Belgium	Belgium Jupiler League	2008/2009	2008-08-16	KSV Roeselare	KV Kortrijk	2	2	Match Drawn
7	9	Belgium	Belgium Jupiler League	2008/2009	2008-08-16	KVC Westerlo	Sporting Charleroi	1	0	Home team won
8	10	Belgium	Belgium Jupiler League	2008/2009	2008-11-01	Club Brugge KV	KV Kortrijk	4	1	Home team won
9	11	Belgium	Belgium Jupiler League	2008/2009	2008-10-31	Standard de Liège	Sporting Charleroi	1	2	Away team won
10	12	Belgium	Belgium Jupiler League	2008/2009	2008-11-02	SV Zulte-Waregem	KAA Gent	0	2	Away team won
11	13	Belgium	Belgium Jupiler League	2008/2009	2008-11-01	Sporting Lokeren	RAEC Mons	0	0	Match Drawn
12	15	Belgium	Belgium Jupiler League	2008/2009	2008-11-01	KV Mechelen	KSV Roeselare	1	2	Away team won
13	18	Belgium	Belgium Jupiler League	2008/2009	2008-11-02	Beerschot AC	RSC Anderlecht	1	3	Away team won
14	19	Belgium	Belgium Jupiler League	2008/2009	2008-11-08	RSC Anderlecht	Sporting Lokeren	2	3	Away team won
15	21	Belgium	Belgium Jupiler League	2008/2009	2008-11-09	Sporting Charleroi	Club Brugge KV	2	2	Match Drawn
16	22	Belgium	Belgium Jupiler League	2008/2009	2008-11-07	KSV Cercle Brugge	SV Zulte-Ware...	2	0	Home team won
17	25	Belgium	Belgium Jupiler League	2008/2009	2008-11-08	KV Kortrijk	KV Mechelen	0	0	Match Drawn
18	26	Belgium	Belgium Jupiler League	2008/2009	2008-11-08	KVC Westerlo	KRC Genk	1	0	Home team won
19	27	Belgium	Belgium Jupiler League	2008/2009	2008-11-09	Beerschot AC	Standard de Liè...	1	3	Away team won
20	28	Belgium	Belgium Jupiler League	2008/2009	2008-11-16	Club Brugge KV	RSC Anderlecht	1	1	Match Drawn
21	29	Belgium	Belgium Jupiler League	2008/2009	2008-11-15	KRC Genk	KSV Roeselare	1	1	Match Drawn
22	30	Belgium	Belgium Jupiler League	2008/2009	2008-11-15	SV Zulte-Waregem	Beerschot AC	2	2	Match Drawn
23	31	Belgium	Belgium Jupiler League	2008/2009	2008-11-16	Sporting Lokeren	Standard de Liè...	1	1	Match Drawn
24	33	Belgium	Belgium Jupiler League	2008/2009	2008-11-15	KV Mechelen	RAEC Mons	0	0	Match Drawn
25	36	Belgium	Belgium Jupiler League	2008/2009	2008-11-14	KVC Westerlo	KAA Gent	3	2	Home team won

RETIRED FOOTBALLERS WITH AGES - TOP AND BOTTOM

```
use European_Soccer
select top 25 player_name AS Name,Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) as Age
from Player
where Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) >= 34
order by Age asc
```

	Name	Age
1	Abdoulay Konko	34
2	Aboubacar Tandia	34
3	Abderrazak Jadid	34
4	Aco Stojkov	34
5	Achmed Ahahaoui	34
6	Adnan Alisic	34
7	Adrian Cristea	34
8	Adrian Mrowiec	34
9	Ahmed Saied Okka	34
10	Alain Cantareil	34
11	Albert Bunjaku	34
12	Alberto Giuliatto	34
13	Alberto Regazzoni	34
14	Alberto Rodriguez	34
15	Albin Ebondo	34
16	Aleksandar Todorovski	34
17	Alessandro Diamanti	34
18	Alessandro Potenza	34
19	Alessandro Rosina	34
20	Alessio Sestu	34
21	Alexandre Licata	34
22	Alexis Norambuena Ruz	34
23	Alfonso De Lucia	34
24	Alhassane Keita	34
25	Aloys Nong	34

```
use European_Soccer
select top 25 player_name AS Name,Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) as Age
from Player
```

where Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) >= 34

order by Age desc

	Name	Age
1	Alberto Fontana	51
2	Dean Windass	49
3	Luca Bucci	49
4	Paolo Maldini	49
5	Rob van Dijk	49
6	Francesco Antonioli	48
7	Hans Vonk	48
8	Jens Lehmann	48
9	Michael Tarnat	48
10	Antonio Chimenti	47
11	Chris Swailes	47
12	David James	47
13	David Weir	47
14	Dean Kiely	47
15	Edwin van der Sar	47
16	Eugenio Corini	47
17	Lee Bullen,29	47
18	Nico van Kerckhoven	47
19	Pedro Roma	47
20	Sander Boschker	47
21	Tugay Kerimoglou	47
22	Barry Wilson	46
23	Brad Friedel	46
24	Cesar Sanchez	46
25	Christophe Revault	46

POTENTIAL VS HEIGHT

```
use European_Soccer

SELECT

CASE

 WHEN ROUND(height,0)<165

 then 165

 WHEN ROUND(height,0)>195

 then 195

 ELSE ROUND(height,0)

END AS calc_height,

COUNT(height) AS distribution, (avg(PA_Grouped.avg_overall_rating)) AS avg_overall_rating,
(avg(PA_Grouped.avg_potential)) AS avg_potential,

AVG(weight) AS avg_weight

FROM PLAYER

LEFT JOIN (SELECT Player_Attributes.player_api_id,

avg(Player_Attributes.overall_rating) AS avg_overall_rating,

avg(Player_Attributes.potential) AS avg_potential

FROM Player_Attributes

GROUP BY Player_Attributes.player_api_id)

AS PA_Grouped ON PLAYER.player_api_id = PA_Grouped.player_api_id

GROUP BY player.height

ORDER BY calc_height
```

	calc_height	distribution	avg_overall_rating	avg_potential	avg_weight
1	165	62	66	72	140
2	165	9	68	74	137
3	165	2	74	78	124
4	165	1	66	71	117
5	167	118	67	72	144
6	170	403	67	72	147
7	172	530	66	72	152
8	175	1188	66	71	156
9	177	1489	65	71	160
10	180	1388	65	71	165
11	182	1954	66	71	170
12	185	1278	66	71	174
13	187	1305	66	71	179
14	190	652	66	71	184
15	193	470	67	72	188
16	195	129	67	72	194
17	195	60	66	71	199
18	195	1	61	66	243
19	195	12	65	71	204
20	195	9	65	69	190

PLAYER LIFETIME RATING AVERAGE

```
use European_Soccer

select top 25 Player_Attributes.player_api_id, Player.player_name, avg(overall_rating) as lifetime_average

from Player_Attributes FULL OUTER JOIN Player on Player_Attributes.player_api_id = Player.player_api_id

where overall_rating > 0

group by Player_Attributes.player_api_id, Player.player_name

order by lifetime_average DESC;
```

	player_api_id	player_name	lifetime_average
1	30981	Lionel Messi	92
2	30893	Cristiano Ronaldo	91
3	30955	Andres Iniesta	88
4	30924	Franck Ribery	88
5	35724	Zlatan Ibrahimovic	88
6	30834	Arjen Robben	87
7	30829	Wayne Rooney	87
8	39854	Xavi Hernandez	87
9	30613	Cesc Fabregas	86
10	37459	David Silva	86
11	30657	Iker Casillas	86
12	30894	Philipp Lahm	86
13	30843	Robin van Persie	86
14	37412	Sergio Aguero	86
15	30872	Bastian Schweinsteiger	85
16	30661	Carles Puyol	85
17	38817	Carlos Tevez	85
18	34520	Fabio Cannavaro	85
19	30717	Gianluigi Buffon	85
20	40636	Luis Suarez	85
21	27299	Manuel Neuer	85
22	27301	Marcelo Jose Bordon	85
23	30865	Nemanja Vidic	85
24	30859	Petr Cech	85
25	30962	Sergio Ramos	85

PLAYER IN TERMS OF PLAYING POSITION PREDICTED

```
use European_Soccer

select Player.player_name as Name, avg(overall_rating) as 'Lifetime Rating',

(avg(Player_Attributes.heading_accuracy) + 3*avg(Player_Attributes.interceptions) +
2*avg(Player_Attributes.sliding_tackle) +

2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) +
2*avg(Player_Attributes.strength) )/11 as 'Defender Rating',

(avg(Player_Attributes.crossing) + avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) +
3*avg(Player_Attributes.ball_control) +

2*avg(Player_Attributes.penalties) + 2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) +
3*avg(Player_Attributes.finishing) +

3*avg(Player_Attributes.dribbling) )/18 as 'Striker Rating',

(2*avg(Player_Attributes.gk_diving) + avg(Player_Attributes.gk_handling) + avg(Player_Attributes.gk_kicking)
+ avg(Player_Attributes.gk_positioning) + 3*avg(Player_Attributes.gk_reflexes) )/8 as 'Goalkeeper Rating',

(case
```

```
 when ((2*avg(Player_Attributes.gk_diving) + avg(Player_Attributes.gk_handling) +
avg(Player_Attributes.gk_kicking) + avg(Player_Attributes.gk_positioning) +
3*avg(Player_Attributes.gk_reflexes) )/8) > ((avg(Player_Attributes.crossing) +
avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) + 3*avg(Player_Attributes.ball_control)
+ 2*avg(Player_Attributes.penalties) + 2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) +
3*avg(Player_Attributes.finishing) + 3*avg(Player_Attributes.dribbling) )/18) and
((2*avg(Player_Attributes.gk_diving) + avg(Player_Attributes.gk_handling) +
avg(Player_Attributes.gk_kicking) + avg(Player_Attributes.gk_positioning) +
3*avg(Player_Attributes.gk_reflexes) )/8) > ((avg(Player_Attributes.heading_accuracy) +
3*avg(Player_Attributes.interceptions) + 2*avg(Player_Attributes.sliding_tackle) +
2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) +
2*avg(Player_Attributes.strength) )/11)

 then 'Goalkeeper'

 else

  case

   when ((avg(Player_Attributes.heading_accuracy) + 3*avg(Player_Attributes.interceptions) +
2*avg(Player_Attributes.sliding_tackle) + 2*avg(Player_Attributes.standing_tackle) +
avg(Player_Attributes.aggression) + 2*avg(Player_Attributes.strength) )/11) >=
((avg(Player_Attributes.crossing) + avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) +
3*avg(Player_Attributes.ball_control) + 2*avg(Player_Attributes.penalties) +
2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) + 3*avg(Player_Attributes.finishing) +
3*avg(Player_Attributes.dribbling) )/18) and ((avg(Player_Attributes.heading_accuracy) +
3*avg(Player_Attributes.interceptions) + 2*avg(Player_Attributes.sliding_tackle) +
2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) +
2*avg(Player_Attributes.strength) )/11) > ((2*avg(Player_Attributes.gk_diving) +
avg(Player_Attributes.gk_handling) + avg(Player_Attributes.gk_kicking) +
avg(Player_Attributes.gk_positioning) + 3*avg(Player_Attributes.gk_reflexes) )/8)

    then case

     when ((avg(Player_Attributes.heading_accuracy) + 3*avg(Player_Attributes.interceptions) +
2*avg(Player_Attributes.sliding_tackle) + 2*avg(Player_Attributes.standing_tackle) +
avg(Player_Attributes.aggression) + 2*avg(Player_Attributes.strength) )/11) - ((avg(Player_Attributes.crossing)
+ avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) +
3*avg(Player_Attributes.ball_control) + 2*avg(Player_Attributes.penalties) +
2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) + 3*avg(Player_Attributes.finishing) +
3*avg(Player_Attributes.dribbling) )/18) <= 20 and ((avg(Player_Attributes.heading_accuracy) +
3*avg(Player_Attributes.interceptions) + 2*avg(Player_Attributes.sliding_tackle) +
2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) +
2*avg(Player_Attributes.strength) )/11) - ((avg(Player_Attributes.crossing) +
avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) + 3*avg(Player_Attributes.ball_control)
+ 2*avg(Player_Attributes.penalties) + 2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) +
3*avg(Player_Attributes.finishing) + 3*avg(Player_Attributes.dribbling) )/18) >= 0

      then 'Defensive Midfielder'

     else 'Defender'

    end

  else
```

```
  case

   when ((avg(Player_Attributes.crossing) + avg(Player_Attributes.short_passing) +
2*avg(Player_Attributes.vision) + 3*avg(Player_Attributes.ball_control) + 2*avg(Player_Attributes.penalties) +
2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) + 3*avg(Player_Attributes.finishing) +
3*avg(Player_Attributes.dribbling) )/18) > ((avg(Player_Attributes.heading_accuracy) +
3*avg(Player_Attributes.interceptions) + 2*avg(Player_Attributes.sliding_tackle) +
2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) +
2*avg(Player_Attributes.strength) )/11) and ((avg(Player_Attributes.crossing) +
avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) + 3*avg(Player_Attributes.ball_control)
+ 2*avg(Player_Attributes.penalties) + 2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) +
3*avg(Player_Attributes.finishing) + 3*avg(Player_Attributes.dribbling) )/18) >
((2*avg(Player_Attributes.gk_diving) + avg(Player_Attributes.gk_handling) +
avg(Player_Attributes.gk_kicking) + avg(Player_Attributes.gk_positioning) +
3*avg(Player_Attributes.gk_reflexes) )/8)

   then case

     when ((avg(Player_Attributes.crossing) + avg(Player_Attributes.short_passing) +
2*avg(Player_Attributes.vision) + 3*avg(Player_Attributes.ball_control) + 2*avg(Player_Attributes.penalties) +
2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) + 3*avg(Player_Attributes.finishing) +
3*avg(Player_Attributes.dribbling) )/18) - ((avg(Player_Attributes.heading_accuracy) +
3*avg(Player_Attributes.interceptions) + 2*avg(Player_Attributes.sliding_tackle) +
2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) +
2*avg(Player_Attributes.strength) )/11) <= 25 and ((avg(Player_Attributes.crossing) +
avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) + 3*avg(Player_Attributes.ball_control)
+ 2*avg(Player_Attributes.penalties) + 2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) +
3*avg(Player_Attributes.finishing) + 3*avg(Player_Attributes.dribbling) )/18) -
((avg(Player_Attributes.heading_accuracy) + 3*avg(Player_Attributes.interceptions) +
2*avg(Player_Attributes.sliding_tackle) + 2*avg(Player_Attributes.standing_tackle) +
avg(Player_Attributes.aggression) + 2*avg(Player_Attributes.strength) )/11) >= 0

      then 'Attacking Midfielder'

     else 'Striker'

     end

  end

 end

end) as 'Ideal Player Position'

from Player_Attributes full outer join Player on Player_Attributes.player_api_id = Player.player_api_id

where Player_Attributes.heading_accuracy >= 0 and Player_Attributes.interceptions >= 0 and
Player_Attributes.sliding_tackle >= 0 and

Player_Attributes.standing_tackle >= 0 and Player_Attributes.aggression >= 0 and Player_Attributes.strength
>= 0 and

Player_Attributes.crossing >= 0 and Player_Attributes.short_passing >= 0 and Player_Attributes.vision >= 0
and
```

Player_Attributes.ball_control >= 0 and Player_Attributes.penalties >= 0 and Player_Attributes.potential >= 0 and

Player_Attributes.volleys >= 0 and Player_Attributes.finishing >= 0 and Player_Attributes.dribbling >= 0 and

Player_Attributes.gk_diving >= 0 and Player_Attributes.gk_handling >= 0 and Player_Attributes.gk_kicking >= 0 and

Player_Attributes.gk_positioning >= 0 and Player_Attributes.gk_reflexes >= 0

group by Player.player_name, Player.birthday

order by 'Lifetime Rating' DESC

	Name	Lifetime Rating	Defender Rating	Striker Rating	Goalkeeper Rating	Ideal Player Position
1	Lionel Messi	92	38	90	12	Striker
2	Cristiano Ronaldo	91	46	88	13	Striker
3	Franck Ribery	88	43	85	12	Striker
4	Andres Iniesta	88	61	84	12	Attacking Midfielder
5	Zlatan Ibrahimovic	88	53	87	15	Striker
6	Xavi Hernandez	87	64	84	12	Attacking Midfielder
7	Arjen Robben	87	40	85	13	Striker
8	Wayne Rooney	87	60	84	10	Attacking Midfielder
9	David Silva	86	44	84	11	Striker
10	Robin van Persie	86	44	85	9	Striker
11	Philipp Lahm	86	79	74	10	Defensive Midfielder
12	Iker Casillas	86	31	32	85	Goalkeeper
13	Cesc Fabregas	86	64	85	13	Attacking Midfielder
14	Sergio Aguero	86	38	85	14	Striker
15	Luis Suarez	85	53	84	29	Striker
16	Nemanja Vidic	85	86	53	11	Defender
17	Petr Cech	85	32	31	84	Goalkeeper
18	Gianluigi Buffon	85	35	33	82	Goalkeeper
19	Carles Puyol	85	87	58	13	Defender
20	Yaya Toure	85	81	78	11	Defensive Midfielder
21	Carlos Tevez	85	57	84	8	Striker
22	Bastian Schwein...	85	75	82	15	Attacking Midfielder
23	Manuel Neuer	85	32	33	84	Goalkeeper
24	Sergio Ramos	85	85	69	13	Defensive Midfielder
25	Vincent Kompany	84	83	67	10	Defensive Midfielder
26	Giorgio Chiellini	84	86	55	8	Defender
27	Wesley Sneijder	84	57	82	12	Attacking Midfielder
28	Daniel Alves	84	78	75	11	Defensive Midfielder
29	Edwin van der Sar	84	44	32	83	Goalkeeper
30	Andrea Pirlo	84	62	81	7	Attacking Midfielder
31	Xabi Alonso	84	77	77	12	Defensive Midfielder

HEIGHT AND HEADER ACCURACY CORRELATION

```
use European_Soccer

select Player.player_name as Name, avg(Player.height) as Height,avg(Player_Attributes.heading_accuracy) AS Header_Accuracy

from Player_Attributes FULL OUTER JOIN Player on Player_Attributes.player_api_id = Player.player_api_id

where height > 180

group by Player.player_name

order by Height ASC, Header_Accuracy ASC
```

	Name	Height	Header_Accuracy
1	Paul Andre Guerin	182	10
2	Benjamin Kirsten	182	12
3	Richard Kingson	182	12
4	Alex Calderoni	182	12
5	William Dutoit	182	13
6	Alan Martin	182	14
7	Bram Verbist	182	14
8	Landry Bonnefoi	182	14
9	Macedo Novaes	182	14
10	Giedrius Arlauskis	182	14
11	Victor Valdes	182	14
12	Jed Steer	182	15
13	Quim	182	15
14	Collin van Eijk	182	15
15	Gennaro Bracigliano	182	15
16	Mickael Landreau	182	15
17	Teddy Richert	182	15
18	David Mitchell	182	15
19	Benedikt Fernandez	182	15
20	Ivan Benito	182	15
21	Johann Durand	182	16
22	Jonathan Ligali	182	16
23	Daniel Gimenez	182	16
24	Laurent Pionnier	182	16
25	Mark Bunn	182	16
26	Jeremy Gavanon	182	16
27	Tom Mickel	182	17
28	Tomas Cerny	182	17
29	Darren Keet	182	17

GOAL SUMMARY BY SEASON

```
use European_Soccer

SELECT Country.name AS country_name, League.name AS league_name, season,count(distinct stage) AS
number_of_stages,

count(distinct HT.team_long_name) AS number_of_teams, avg(home_team_goal) AS avg_home_team_goals,

avg(away_team_goal) AS avg_away_team_goals, avg(home_team_goal-away_team_goal) AS avg_goal_dif,

avg(home_team_goal+away_team_goal) AS avg_goals, sum(home_team_goal+away_team_goal) AS
total_goals

from Match

JOIN Country on Country.id = Match.country_id

JOIN League on League.id = Match.league_id

LEFT JOIN Team AS HT on HT.team_api_id = Match.home_team_api_id

LEFT JOIN Team AS AT on AT.team_api_id = Match.away_team_api_id

GROUP BY Country.name, League.name, season

HAVING count(distinct stage) > 10

ORDER BY total_goals DESC, Country.name, League.name, season DESC
```

	country_name	league_name	season	number_of_stages	number_of_teams	avg_home_team_goals	avg_away_team_goals	avg_goal_dif	avg_goals	total_goals
1	Spain	Spain LIGA BBVA	2008/2009	38	20	1	1	0	2	1101
2	Spain	Spain LIGA BBVA	2012/2013	38	20	1	1	0	2	1091
3	England	England Premier League	2011/2012	38	20	1	1	0	2	1066
4	England	England Premier League	2012/2013	38	20	1	1	0	2	1063
5	England	England Premier League	2010/2011	38	20	1	1	0	2	1063
6	England	England Premier League	2009/2010	38	20	1	1	0	2	1053
7	England	England Premier League	2013/2014	38	20	1	1	0	2	1052
8	Spain	Spain LIGA BBVA	2011/2012	38	20	1	1	0	2	1050
9	Spain	Spain LIGA BBVA	2013/2014	38	20	1	1	0	2	1045
10	Spain	Spain LIGA BBVA	2015/2016	38	20	1	1	0	2	1043
11	Spain	Spain LIGA BBVA	2010/2011	38	20	1	1	0	2	1042
12	Italy	Italy Serie A	2013/2014	38	20	1	1	0	2	1035
13	Spain	Spain LIGA BBVA	2009/2010	38	20	1	1	0	2	1031
14	England	England Premier League	2015/2016	38	20	1	1	0	2	1026
15	Italy	Italy Serie A	2014/2015	38	20	1	1	0	2	1018
16	Spain	Spain LIGA BBVA	2014/2015	38	20	1	1	0	2	1009
17	Italy	Italy Serie A	2012/2013	38	20	1	1	0	2	1003
18	Netherlands	Netherlands Eredivisie	2011/2012	34	18	1	1	0	3	997
19	Italy	Italy Serie A	2009/2010	38	20	1	1	0	2	992
20	Italy	Italy Serie A	2008/2009	38	20	1	1	0	2	988
21	Netherlands	Netherlands Eredivisie	2010/2011	34	18	1	1	0	3	987
22	Italy	Italy Serie A	2015/2016	38	20	1	1	0	2	979
23	Netherlands	Netherlands Eredivisie	2013/2014	34	18	1	1	0	3	978
24	England	England Premier League	2014/2015	38	20	1	1	0	2	975
25	France	France Ligue 1	2012/2013	38	20	1	1	0	2	967
26	Germany	Germany 1. Bundesliga	2013/2014	34	18	1	1	0	3	967
27	Netherlands	Netherlands Eredivisie	2012/2013	34	18	1	1	0	3	964
28	France	France Ligue 1	2015/2016	38	20	1	1	0	2	960
29	France	France Ligue 1	2011/2012	38	20	1	1	0	2	956

FITNESS

```
use European_Soccer

select top 25 player.player_name as Name, cast(round(((weight*703*2.54*2.54)/(height*height)),0) as int) as
bmi,

avg(Player_Attributes.acceleration) as acceleration, avg(Player_Attributes.sprint_speed) as sprint_speed,

avg(Player_Attributes.agility) as agility, avg(Player_Attributes.jumping) as jumping ,
avg(Player_Attributes.stamina) as stamina,

((avg(Player_Attributes.acceleration) + avg(Player_Attributes.sprint_speed) + avg(Player_Attributes.agility) +

avg(Player_Attributes.jumping) +
avg(Player_Attributes.stamina))*cast(round(((weight*703*2.54*2.54)/(height*height)),0) as int))/(5*25) as
fitness

from player full outer join Player_Attributes on Player.player_api_id = Player_Attributes.player_api_id

group by player.player_name, height, weight

order by fitness DESC
```

	Name	bmi	acceleration	sprint_speed	agility	jumping	stamina	fitness
1	Ikechi Anya	26	92	90	89	89	78	91
2	Frank Acheampong	26	90	89	85	84	76	88
3	Royston Drenthe	28	87	84	85	79	59	88
4	Andy Kawaya	28	84	82	77	82	62	86
5	Wayne Rooney	27	77	79	78	79	88	86
6	Ahmed Musa	25	92	89	85	85	75	85
7	Fraizer Campbell	28	87	84	77	67	68	85
8	Lionel Messi	25	95	90	94	69	77	85
9	Wakaso Mubarak	25	84	83	84	90	86	85
10	Darius Vassell	27	84	86	72	67	81	84
11	Henrik Ojamaa	26	86	84	71	75	88	84
12	Jefferson Farfan	27	87	86	79	62	77	84
13	Jerry Uche Mbakogu	25	86	91	79	90	75	84
14	Obafemi Martins	25	93	92	84	86	68	84
15	Seydou Doumbia	25	88	90	79	78	85	84
16	Stephane Sessegnon	27	81	80	83	69	78	84
17	Cheik Tiote	27	74	71	77	77	86	83
18	Cristiano Ronaldo	23	91	93	92	93	87	83
19	Emmanuel Emenike	27	85	85	70	67	78	83
20	Kwadwo Asamoah	26	82	81	79	77	84	83
21	Victor Obinna	25	88	87	79	84	77	83
22	Caiuby	26	79	86	75	84	74	82
23	Cristian Benitez	25	87	86	82	80	79	82
24	Damian Suarez	25	79	83	82	81	87	82
25	Daniel Larsson	26	83	88	74	73	80	82

CURRENTLY PLAYING FOOTBALLERS

```
use European_Soccer

select top 25 player_name AS Name,Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) as Age

from Player

where Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) < 34

order by Age desc
```

	Name	Age
1	Abdessalam Benjelloun	33
2	Abdelkader Ghezzal	33
3	Abel Aguilar	33
4	Abraham Kudemor	33
5	Adam Collin	33
6	Adam Federici	33
7	Adrian Colunga	33
8	Adriano	33
9	Ahmed Apimah Barusso	33
10	Alan Hutton	33
11	Alan O'Brien	33
12	Albert Baning	33
13	Alberto Aquilani	33
14	Alberto Garcia	33
15	Alessandro Iandoli	33
16	Alessandro Matri	33
17	Alex Bruce	33
18	Alex Silva	33
19	Alexander Farnerud	33
20	Alexander Laas	33
21	Alexandros Tziolis	33
22	Alexis Thebaux	33
23	Almeida Borges Leandro	33
24	Alvaro Gonzalez	33
25	Amad Al Hosni	33

```
use European_Soccer
select top 25 player_name AS Name,Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) as Age
from Player
where Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) < 34
order by Age asc
```

	Name	Age
1	Jonathan Leko	18
2	Alban Lafont	19
3	Christian Pulisic	19
4	Felix Passlack	19
5	Gianluigi Donnarumma	19
6	Harry Souttar	19
7	Jonatan Montiel	19
8	Justin Hoogma	19
9	Kamil Jozwiak	19
10	Kylian Mbappe Lottin	19
11	Przemyslaw Mystkowski	19
12	Reece Oxford	19
13	Ronael Pierre-Gabriel	19
14	Tom Davies	19
15	Adalberto Penaranda	20
16	Adam Ryczkowski	20
17	Amadou Diawara	20
18	Assane Diousse	20
19	Bartlomiej Dragowski	20
20	Benjamin Tetteh	20
21	Bilal Ould-Chikh	20
22	Charles Pickel	20
23	Christopher McLaughlin	20
24	Christopher Nkunku	20
25	Colin Trachsel	20

GOALS AND PERCENTAGE BY FOOT

```
use European_Soccer

select count(preferred_foot) as Goals_by_foot, preferred_foot, (count(preferred_foot)*100.0/(select count(*)
from Player_Attributes)) as Percentage

from Player_Attributes

where preferred_foot != 'NULL'

group by preferred_foot

order by preferred_foot
```

	Goals_by_foot	preferred_foot	Percentage
1	44733	left	24.320797699112
2	138409	right	75.251319802749

DATA MODELS

LOOK ALIKE MODEL

```
use European_Soccer

select cols.Name, cols.[Lifetime Rating], cols.[Ideal Player Position],

(case

 when cols.[Ideal Player Position] = 'Goalkeeper'

 then 100 - cols.[Goalkeeper Rating]

 else

  case

   when cols.[Ideal Player Position] = 'Defender'

   then 100 - cols.[Defender Rating]

   else

    case

     when cols.[Ideal Player Position] = 'Defensive Midfielder'

     then 100 - ((cols.[Defender Rating] + cols.[Striker Rating])/2)

     else

      case

       when cols.[Ideal Player Position] = 'Striker'

       then 100 - cols.[Striker Rating]

       else

        case

         when cols.[Ideal Player Position] = 'Attacking Midfielder'

         then 100 - ((cols.[Defender Rating] + cols.[Striker Rating])/2)
```

```
      end

     end

   end

 end

end) as 'Position Rating'

from (

 select Player.player_name as Name, avg(overall_rating) as 'Lifetime Rating',

 (avg(Player_Attributes.heading_accuracy) + 3*avg(Player_Attributes.interceptions) +
2*avg(Player_Attributes.sliding_tackle) +

 2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) +
2*avg(Player_Attributes.strength) )/11 as 'Defender Rating',

 (avg(Player_Attributes.crossing) + avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) +
3*avg(Player_Attributes.ball_control) +

 2*avg(Player_Attributes.penalties) + 2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) +
3*avg(Player_Attributes.finishing) +

 3*avg(Player_Attributes.dribbling) )/18 as 'Striker Rating',

 (2*avg(Player_Attributes.gk_diving) + avg(Player_Attributes.gk_handling) +
avg(Player_Attributes.gk_kicking) + avg(Player_Attributes.gk_positioning) +
3*avg(Player_Attributes.gk_reflexes) )/8 as 'Goalkeeper Rating',

 (case

  when ((2*avg(Player_Attributes.gk_diving) + avg(Player_Attributes.gk_handling) +
avg(Player_Attributes.gk_kicking) + avg(Player_Attributes.gk_positioning) +
3*avg(Player_Attributes.gk_reflexes) )/8) > ((avg(Player_Attributes.crossing) +
avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) + 3*avg(Player_Attributes.ball_control)
+ 2*avg(Player_Attributes.penalties) + 2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) +
3*avg(Player_Attributes.finishing) + 3*avg(Player_Attributes.dribbling) )/18) and
((2*avg(Player_Attributes.gk_diving) + avg(Player_Attributes.gk_handling) +
avg(Player_Attributes.gk_kicking) + avg(Player_Attributes.gk_positioning) +
3*avg(Player_Attributes.gk_reflexes) )/8) > ((avg(Player_Attributes.heading_accuracy) +
3*avg(Player_Attributes.interceptions) + 2*avg(Player_Attributes.sliding_tackle) +
2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) +
2*avg(Player_Attributes.strength) )/11)

  then 'Goalkeeper'

  else

   case

    when ((avg(Player_Attributes.heading_accuracy) + 3*avg(Player_Attributes.interceptions) +
2*avg(Player_Attributes.sliding_tackle) + 2*avg(Player_Attributes.standing_tackle) +
avg(Player_Attributes.aggression) + 2*avg(Player_Attributes.strength) )/11) >=
```

```
((avg(Player_Attributes.crossing) + avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) +
3*avg(Player_Attributes.ball_control) + 2*avg(Player_Attributes.penalties) +
2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) + 3*avg(Player_Attributes.finishing) +
3*avg(Player_Attributes.dribbling) )/18) and ((avg(Player_Attributes.heading_accuracy) +
3*avg(Player_Attributes.interceptions) + 2*avg(Player_Attributes.sliding_tackle) +
2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) +
2*avg(Player_Attributes.strength) )/11) > ((2*avg(Player_Attributes.gk_diving) +
avg(Player_Attributes.gk_handling) + avg(Player_Attributes.gk_kicking) +
avg(Player_Attributes.gk_positioning) + 3*avg(Player_Attributes.gk_reflexes) )/8)

    then case

     when ((avg(Player_Attributes.heading_accuracy) + 3*avg(Player_Attributes.interceptions) +
2*avg(Player_Attributes.sliding_tackle) + 2*avg(Player_Attributes.standing_tackle) +
avg(Player_Attributes.aggression) + 2*avg(Player_Attributes.strength) )/11) - ((avg(Player_Attributes.crossing)
+ avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) +
3*avg(Player_Attributes.ball_control) + 2*avg(Player_Attributes.penalties) +
2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) + 3*avg(Player_Attributes.finishing) +
3*avg(Player_Attributes.dribbling) )/18) <= 20 and ((avg(Player_Attributes.heading_accuracy) +
3*avg(Player_Attributes.interceptions) + 2*avg(Player_Attributes.sliding_tackle) +
2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) +
2*avg(Player_Attributes.strength) )/11) - ((avg(Player_Attributes.crossing) +
avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) + 3*avg(Player_Attributes.ball_control)
+ 2*avg(Player_Attributes.penalties) + 2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) +
3*avg(Player_Attributes.finishing) + 3*avg(Player_Attributes.dribbling) )/18) >= 0

      then 'Defensive Midfielder'

     else 'Defender'

    end

  else

   case

    when ((avg(Player_Attributes.crossing) + avg(Player_Attributes.short_passing) +
2*avg(Player_Attributes.vision) + 3*avg(Player_Attributes.ball_control) + 2*avg(Player_Attributes.penalties) +
2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) + 3*avg(Player_Attributes.finishing) +
3*avg(Player_Attributes.dribbling) )/18) > ((avg(Player_Attributes.heading_accuracy) +
3*avg(Player_Attributes.interceptions) + 2*avg(Player_Attributes.sliding_tackle) +
2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) +
2*avg(Player_Attributes.strength) )/11) and ((avg(Player_Attributes.crossing) +
avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) + 3*avg(Player_Attributes.ball_control)
+ 2*avg(Player_Attributes.penalties) + 2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) +
3*avg(Player_Attributes.finishing) + 3*avg(Player_Attributes.dribbling) )/18) >
((2*avg(Player_Attributes.gk_diving) + avg(Player_Attributes.gk_handling) +
avg(Player_Attributes.gk_kicking) + avg(Player_Attributes.gk_positioning) +
3*avg(Player_Attributes.gk_reflexes) )/8)

    then case

     when ((avg(Player_Attributes.crossing) + avg(Player_Attributes.short_passing) +
2*avg(Player_Attributes.vision) + 3*avg(Player_Attributes.ball_control) + 2*avg(Player_Attributes.penalties) +
2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) + 3*avg(Player_Attributes.finishing) +
```

```
3*avg(Player_Attributes.dribbling) )/18) - ((avg(Player_Attributes.heading_accuracy) +
3*avg(Player_Attributes.interceptions) + 2*avg(Player_Attributes.sliding_tackle) +
2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) +
2*avg(Player_Attributes.strength) )/11) <= 25 and ((avg(Player_Attributes.crossing) +
avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) + 3*avg(Player_Attributes.ball_control)
+ 2*avg(Player_Attributes.penalties) + 2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) +
3*avg(Player_Attributes.finishing) + 3*avg(Player_Attributes.dribbling) )/18) -
((avg(Player_Attributes.heading_accuracy) + 3*avg(Player_Attributes.interceptions) +
2*avg(Player_Attributes.sliding_tackle) + 2*avg(Player_Attributes.standing_tackle) +
avg(Player_Attributes.aggression) + 2*avg(Player_Attributes.strength) )/11) >= 0

        then 'Attacking Midfielder'

       else 'Striker'

       end

    end

   end

 end) as 'Ideal Player Position'

 from Player_Attributes full outer join Player on Player_Attributes.player_api_id = Player.player_api_id

 where Player_Attributes.heading_accuracy >= 0 and Player_Attributes.interceptions >= 0 and
Player_Attributes.sliding_tackle >= 0 and

 Player_Attributes.standing_tackle >= 0 and Player_Attributes.aggression >= 0 and Player_Attributes.strength
>= 0 and

 Player_Attributes.crossing >= 0 and Player_Attributes.short_passing >= 0 and Player_Attributes.vision >= 0
and

 Player_Attributes.ball_control >= 0 and Player_Attributes.penalties >= 0 and Player_Attributes.potential >= 0
and

 Player_Attributes.volleys >= 0 and Player_Attributes.finishing >= 0 and Player_Attributes.dribbling >= 0 and

 Player_Attributes.gk_diving >= 0 and Player_Attributes.gk_handling >= 0 and Player_Attributes.gk_kicking
>= 0 and

 Player_Attributes.gk_positioning >= 0 and Player_Attributes.gk_reflexes >= 0

 group by Player.player_name, Player.birthday

) cols

order by 'Position Rating' ASC
```

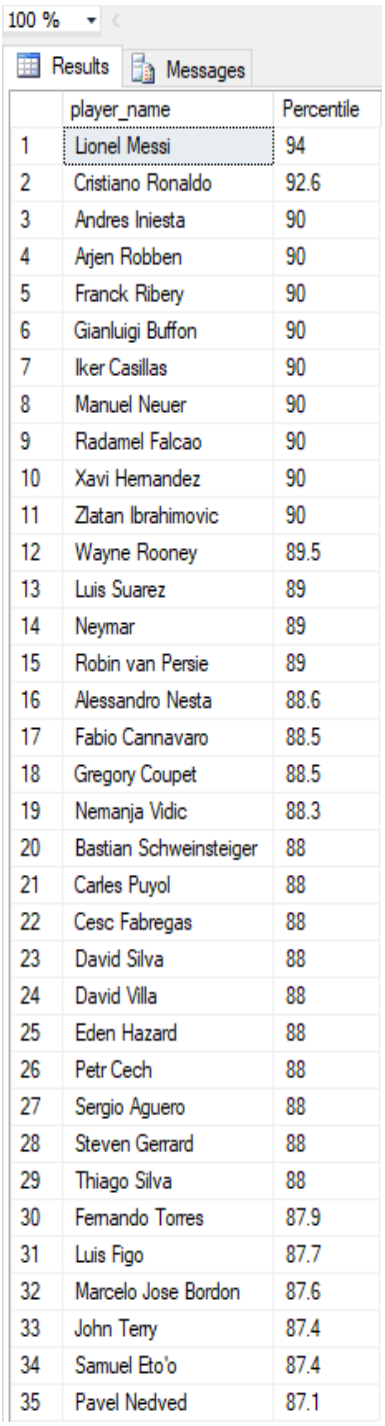
	Name	Lifetime Rating	Ideal Player Position	Position Rating
1	Lionel Messi	92	Striker	10
2	Cristiano Ronaldo	91	Striker	12
3	Carles Puyol	85	Defender	13
4	Zlatan Ibrahimovic	88	Striker	13
5	Nemanja Vidic	85	Defender	14
6	John Terry	84	Defender	14
7	Francesco Totti	81	Striker	14
8	Giorgio Chiellini	84	Defender	14
9	Arjen Robben	87	Striker	15
10	Pepe	83	Defender	15
11	Franck Ribery	88	Striker	15
12	Robin van Persie	86	Striker	15
13	Ronaldinho	82	Striker	15
14	Iker Casillas	86	Goalkeeper	15
15	Antonio Cassano	82	Striker	15
16	Sergio Aguero	86	Striker	15
17	Luis Suarez	85	Striker	16
18	Petr Cech	85	Goalkeeper	16
19	David Villa	84	Striker	16
20	Andrea Barzagli	83	Defender	16
21	Carlos Tevez	85	Striker	16
22	Per Mertesacker	82	Defender	16
23	David Silva	86	Striker	16
24	Manuel Neuer	85	Goalkeeper	16
25	Branislav Ivanovic	80	Defender	17
26	Dimitar Berbatov	82	Striker	17
27	Luisao	80	Defender	17
28	Victor Valdes	83	Goalkeeper	17
29	Kaka	83	Striker	17
30	Edwin van der Sar	84	Goalkeeper	17
31	Gary Neville	78	Defender	17
32	Walter Samuel	80	Defender	17

PERCENTILE OVERALL RATINGS

```
Use European_Soccer

Select *

from (SELECT DISTINCT

        player_name,

        /*STDEV(overall_rating) OVER (PARTITION BY [player_name]) as Standard_Deviation,*/

        PERCENTILE_CONT(0.9) WITHIN GROUP (ORDER BY overall_rating) OVER (PARTITION BY
player_name) as 'Percentile'

FROM    Player_Attributes,Player

where Player.player_api_id = Player_Attributes.player_api_id

) as col
```

where col.Percentile > 87

order by col.Percentile Desc



100 %

Results | Messages

	player_name	Percentile
1	Lionel Messi	94
2	Cristiano Ronaldo	92.6
3	Andres Iniesta	90
4	Arjen Robben	90
5	Franck Ribery	90
6	Gianluigi Buffon	90
7	Iker Casillas	90
8	Manuel Neuer	90
9	Radamel Falcao	90
10	Xavi Hernandez	90
11	Zlatan Ibrahimovic	90
12	Wayne Rooney	89.5
13	Luis Suarez	89
14	Neymar	89
15	Robin van Persie	89
16	Alessandro Nesta	88.6
17	Fabio Cannavaro	88.5
18	Gregory Coupet	88.5
19	Nemanja Vidic	88.3
20	Bastian Schweinsteiger	88
21	Carles Puyol	88
22	Cesc Fabregas	88
23	David Silva	88
24	David Villa	88
25	Eden Hazard	88
26	Petr Cech	88
27	Sergio Aguero	88
28	Steven Gerrard	88
29	Thiago Silva	88
30	Fernando Torres	87.9
31	Luis Figo	87.7
32	Marcelo Jose Bordon	87.6
33	John Terry	87.4
34	Samuel Eto'o	87.4
35	Pavel Nedved	87.1

PERCENTILE STRIKERS

```
select top 20 Player.player_name as Name, avg(Player_Attributes.crossing) as Crossing,
avg(Player_Attributes.short_passing) as Short_Passing,

avg(Player_Attributes.vision) as Vision, avg(Player_Attributes.ball_control) as Ball_Control,

avg(Player_Attributes.penalties) as Penalties, avg(Player_Attributes.potential) as Potential,

avg(Player_Attributes.volleys) as Volleys, avg(Player_Attributes.finishing) as Finishing,
avg(Player_Attributes.dribbling) as Dribbling,

(avg(Player_Attributes.crossing) + avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) +
3*avg(Player_Attributes.ball_control) +

2*avg(Player_Attributes.penalties) + 2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) +
3*avg(Player_Attributes.finishing) +

3*avg(Player_Attributes.dribbling) )/18 as Rating,

Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) as Age,

(case

 when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) > 33

 then'Retired'

 else

  case

   when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) <= 33

   then 'Playing'

  end

end) as Playing_Status,

PERCENTILE_CONT(0.9) within group(order by((avg(Player_Attributes.crossing) +
avg(Player_Attributes.short_passing) + 2*avg(Player_Attributes.vision) + 3*avg(Player_Attributes.ball_control)
+

2*avg(Player_Attributes.penalties) + 2*avg(Player_Attributes.potential) + avg(Player_Attributes.volleys) +
3*avg(Player_Attributes.finishing) +

3*avg(Player_Attributes.dribbling) )/18)) OVER(partition by Player.player_name) as 'Percentile'

from Player_Attributes full outer join Player on Player_Attributes.player_api_id = Player.player_api_id

group by Player.player_name, Player.birthday

order by Percentile desc
```

	Name	Crossing	Short_Passing	Vision	Ball_Control	Penalties	Potential	Volleys	Finishing	Dribbling	Rating	Age	Playing_Status	Percentile
1	Lionel Messi	83	88	90	95	80	95	87	92	96	90	30	Playing	90
2	Cristiano Ronaldo	83	82	80	93	83	93	85	91	92	88	33	Playing	88
3	Zlatan Ibrahimovic	72	84	82	91	88	90	90	90	89	87	36	Retired	87
4	Francesco Totti	83	90	89	91	89	82	89	82	83	86	41	Retired	86
5	Antonio Cassano	84	89	88	91	84	83	82	79	88	85	35	Retired	85
6	Arjen Robben	82	84	82	89	81	88	84	82	92	85	34	Retired	85
7	Cesc Fabregas	83	92	92	89	85	88	81	77	82	85	30	Playing	85
8	Franck Ribery	85	87	86	91	81	89	82	77	92	85	35	Retired	85
9	Robin van Persie	82	82	81	86	86	87	90	88	83	85	34	Retired	85
10	Ronaldinho	83	87	87	93	85	84	83	75	89	85	38	Retired	85
11	Sergio Aguero	70	82	83	89	82	88	85	88	90	85	29	Playing	85
12	Andres Iniesta	83	93	92	92	71	90	74	72	90	84	33	Playing	84
13	Carlos Tevez	74	82	82	87	83	86	87	87	85	84	34	Retired	84
14	David Silva	82	89	90	90	76	89	80	77	86	84	32	Playing	84
15	David Villa	74	80	78	86	89	85	83	88	84	84	36	Retired	84
16	Luis Suarez	75	79	82	86	83	87	83	85	87	84	31	Playing	84
17	Wayne Rooney	78	83	84	86	82	87	88	87	82	84	32	Playing	84
18	Xavi Hernandez	86	95	95	93	78	88	67	73	82	84	38	Retired	84
19	Antonio Di Natale	79	81	82	89	83	82	85	86	82	83	40	Retired	83
20	Dimitar Berbatov	77	84	82	87	82	83	87	87	82	83	37	Retired	83

PERCENTILE GOALKEEPER

```
select top 20 Player.player_name as Name, avg(Player_Attributes.gk_diving) as Diving,
avg(Player_Attributes.gk_handling) as Handling,

avg(Player_Attributes.gk_kicking) as Kicking, avg(Player_Attributes.gk_positioning) as Positioning,
avg(Player_Attributes.gk_reflexes) as Reflexes,

(2*avg(Player_Attributes.gk_diving) + avg(Player_Attributes.gk_handling) + avg(Player_Attributes.gk_kicking)
+ avg(Player_Attributes.gk_positioning) + 3*avg(Player_Attributes.gk_reflexes) )/8 as Rating,

Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) as Age,

(case

 when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) > 40

 then'Retired'

 else

  case

   when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) <= 40

   then 'Playing'

  end

end) as Playing_Status,

PERCENTILE_CONT(0.9) within group(order by((2*avg(Player_Attributes.gk_diving) +
avg(Player_Attributes.gk_handling) + avg(Player_Attributes.gk_kicking) +
```

avg(Player_Attributes.gk_positioning) + 3*avg(Player_Attributes.gk_reflexes))/8)) OVER(partition by Player.player_name) as 'Percentile'

from Player_Attributes full outer join Player on Player_Attributes.player_api_id = Player.player_api_id

group by Player.player_name, Player.birthday

order by Percentile DESC

	Name	Diving	Handling	Kicking	Positioning	Reflexes	Rating	Age	Playing_Status	Percentile
1	Iker Casillas	89	82	69	84	90	85	36	Playing	85
2	Manuel Neuer	86	81	86	82	86	84	32	Playing	84
3	Petr Cech	86	82	76	85	86	84	35	Playing	84
4	Robert Enke,30	83	84	78	86	87	84	40	Playing	84
5	Edwin van der Sar	83	86	85	91	79	83	47	Retired	83
6	Hugo Lloris	86	81	73	81	87	83	31	Playing	83
7	Victor Valdes	83	78	80	86	86	83	36	Playing	83
8	Gianluigi Buffon	87	82	68	90	83	82	40	Playing	82
9	Pepe Reina	82	78	87	82	84	82	35	Playing	82
10	David De Gea	83	76	81	76	84	81	27	Playing	81
11	Joe Hart	83	76	76	78	85	81	31	Playing	81
12	Mickael Landreau	79	79	80	79	84	81	38	Playing	81
13	Rene Adler	85	79	70	80	85	81	33	Playing	81
14	Steve Mandanda	83	80	78	80	83	81	33	Playing	81
15	Julio Cesar	71	68	69	70	74	71	31	Playing	80.9
16	Julio Cesar	84	80	71	83	85	82	38	Playing	80.9
17	Bernd Leno	81	78	81	77	83	80	26	Playing	80
18	Diego Alves	85	71	73	74	86	80	32	Playing	80
19	Diego Lopez	80	81	76	83	80	80	36	Playing	80
20	Frank Rost	80	78	78	83	81	80	44	Retired	80

PERCENTILE DEFENDERS

select top 20 Player.player_name as Name, avg(Player_Attributes.heading_accuracy) as Heading_Accuracy, avg(Player_Attributes.interceptions) as Interceptions,

avg(Player_Attributes.sliding_tackle) as Sliding_Tackles, avg(Player_Attributes.standing_tackle) as Standing_Tackles,

avg(Player_Attributes.aggression) as Aggression, avg(Player_Attributes.strength) as Strength,

(avg(Player_Attributes.heading_accuracy) + 3*avg(Player_Attributes.interceptions) + 2*avg(Player_Attributes.sliding_tackle) +

2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) + 2*avg(Player_Attributes.strength))/11 as Rating,

Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) as Age,

```
(case
 when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) > 33
 then'Retired'
 else
  case
   when Floor(datediff(day,substring(birthday,1,10), GETDATE())/365.25) <= 33
   then 'Playing'
  end
end) as Playing_Status,
PERCENTILE_CONT(0.9) within group(order by((avg(Player_Attributes.heading_accuracy) +
3*avg(Player_Attributes.interceptions) + 2*avg(Player_Attributes.sliding_tackle) +
2*avg(Player_Attributes.standing_tackle) + avg(Player_Attributes.aggression) +
2*avg(Player_Attributes.strength) )/11)) OVER(partition by Player.player_name) as 'Percentile'
from Player_Attributes full outer join Player on Player_Attributes.player_api_id = Player.player_api_id
group by Player.player_name, Player.birthday
order by Percentile desc
```

	Name	Heading_Accuracy	Interceptions	Sliding_Tackles	Standing_Tackles	Aggression	Strength	Rating	Age	Playing_Status	Percentile
1	Carles Puyol	84	89	87	88	88	88	87	40	Retired	87
2	Giorgio Chiellini	79	86	88	88	85	87	86	33	Playing	86
3	John Terry	91	84	84	87	89	87	86	37	Retired	86
4	Nemanja Vidic	89	83	83	88	89	90	86	36	Retired	86
5	Sergio Ramos	87	83	89	87	84	82	85	32	Playing	85
6	Andrea Barzagli	77	88	85	87	77	83	84	36	Retired	84
7	Daniele De Rossi	85	86	83	85	84	82	84	34	Retired	84
8	Javier Mascherano	62	91	88	90	90	75	84	33	Playing	84
9	Patrick Vieira	83	86	80	84	87	88	84	41	Retired	84
10	Per Mertesacker	90	85	82	87	70	88	84	33	Playing	84
11	Branislav Ivanovic	85	81	84	84	84	83	83	34	Retired	83
12	Gary Neville	77	89	83	83	86	76	83	43	Retired	83
13	Gerard Pique	82	83	87	87	70	83	83	31	Playing	83
14	Jamie Carragher	83	85	84	83	85	83	83	40	Retired	83
15	Luisao	85	82	80	84	81	87	83	37	Retired	83
16	Mark van Bommel	79	83	82	80	93	86	83	40	Retired	83
17	Thiago Silva	80	85	88	87	76	79	83	33	Playing	83
18	Vincent Kompany	79	83	84	87	75	88	83	32	Playing	83
19	Walter Samuel	73	87	82	85	80	84	83	40	Retired	83
20	Alessandro Nesta	80	86	82	86	75	81	82	42	Retired	82

NAIVE BAYES

```
Use European_Soccer

Select col.Country_of_Match,col.Win_Loss_Draw_Class as 'Home_win/Away_win/Draw',
count(col.Country_of_Match) as 'Counter'

from(select HomeTeam.team_long_name as 'Home Team', AwayTeam.team_long_name as 'Away Team',
        M.home_team_goal as 'Home Team Goals', M.away_team_goal as 'Away Team Goals',
        (case
                when M.home_team_goal > M.away_team_goal
                then '1'
                else
                        case
                                when M.home_team_goal < M.away_team_goal
                                then '2'
                                else
                                        case
                                                when M.home_team_goal = M.away_team_goal
                                                then '3'
                                        end
                        end
        end) as Win_Loss_Draw_Class,
        Country.name as Country_of_Match
        from (match as M
        inner join team as HomeTeam on M.home_team_api_id = HomeTeam.team_api_id
        inner join team as AwayTeam on M.away_team_api_id = AwayTeam.team_api_id
        inner join Country on M.country_id = Country.id
        inner join League on M.country_id = League.id)
        )col
        group by col.Country_of_Match, col.Win_Loss_Draw_Class
        order by col.Country_of_Match asc, col.Win_Loss_Draw_Class asc
```

Results | Messages

	Country_of_Match	Home_Win/Away_Win/Draw	Counter
1	Belgium	1	762
2	Belgium	2	466
3	Belgium	3	404
4	England	1	1390
5	England	2	867
6	England	3	783
7	France	1	1359
8	France	2	822
9	France	3	859
10	Germany	1	1107
11	Germany	2	744
12	Germany	3	597
13	Italy	1	1407
14	Italy	2	814
15	Italy	3	796
16	Netherlands	1	1154
17	Netherlands	2	687
18	Netherlands	3	573
19	Poland	1	741
20	Poland	2	443
21	Poland	3	442
22	Portugal	1	823
23	Portugal	2	559
24	Portugal	3	486
25	Scotland	1	760
26	Scotland	2	617
27	Scotland	3	447
28	Spain	1	1485
29	Spain	2	851
30	Spain	3	704
31	Switzerland	1	633
32	Switzerland	2	415
33	Switzerland	3	338

$$\mathrm{P(Country/HW-AW-D)} = \frac{100\ x\ P(Country)\ x\ P(HW-AW-D/Country)}{P(Country)}$$

P(country/Home_win/Away_win/Draw)	Naive Bayes Values
p(Belgium/home_win)	11.028221827511
p(Belgium/away_win)	6.744293138609
p(Belgium/draw)	5.846983751069
p(England/home_win)	20.117097559371
p(England/away_win)	12.547858693507
p(England/draw)	11.332149200710
p(France/home_win)	19.668442865600
p(France/away_win)	11.896585750937
p(France/draw)	12.432076837050
p(Germany/home_win)	16.021314387211
p(Germany/away_win)	10.767712650483
p(Germany/draw)	8.640221038089
p(Italy/home_win)	20.363134004341
p(Italy/away_win)	11.780803894480

p(Italy/draw)	11.520294717452
p(Netherlands/home_win)	16.701532793895
p(Netherlands /away_win)	9.942766923228
p(Netherlands/draw)	8.292875468719
p(Poland/home_win)	10.724294454312
p(Poland/away_win)	6.411420301295
p(Poland/draw)	6.396947569238
p(Portugal/home_win)	11.911058482994
p(Portugal/away_win)	8.090257219919
p(Portugal/draw)	7.033747779751
p(Scotland/home_win)	10.999276363397
p(Scotland/away_win)	8.929675679231
p(Scotland/draw)	6.469311229524
p(Spain/home_win)	21.492007104795
p(Spain/away_win)	12.316294980593
p(Spain/draw)	10.188803368199

p(Switzerland/home_win)	9.161239392145
p(Switzerland/away_win)	6.006183803697
p(Switzerland/draw)	4.891783435300

CONCLUSION

The soccer database in SQLite format was migrated to SSMS with an ODBC (64 bit) interface. A 'System DSN' with a linked server created was created in SSMS and connected to the database in SQLite. This data obtained from the dataset was recorded in tables created in SSMS. Further, a 'User DSN' was created to connect the soccer database in SSMS to R/Rattle to perform data cleaning, transformation, and visualization. Moreover, Fascinating outcomes were gathered through predictive analysis by SQL queries in SQL Server.

REFERENCES

- https://www.kaggle.com/hugomathien/soccer - Dataset Source
- https://www.mssqltips.com/sqlservertip/3087/creating-a-sql-server-linked-server-to-sqlite-to-import-data/
- http://www.football-data.co.uk/notes.txt - MetaData about
- https://stackoverflow.com/questions/40453995/split-one-large-denormalized-table-into-a-normalized-database
- https://www.youtube.com/watch?v-y-lbCiAtHsk (DSN for database on SQL Server).
- https://en.wikipedia.org/wiki/Data_source_name
- https://cran.r-project.org/web/packages/RODBC/vignettes/RODBC.pdf