COMP 533-Final Report

Team Name: Animal Crossing

Database Name: European soccer market system

Team members: Siting Chen(sc122), Jian Fang(jf58)

**Abstract**

Soccer is one of the most popular sports around the world. Our team focuses on designing a system which helps people acquire and analyze the data in European football market. Firstly, we use PostgreSQL-supported methods to build relationships among players, teams, matches, leagues as well as countries. And then our team filters valid and valuable data we actually need. Finally, our task is to process data for player performance comparisons, player potential analysis, team rating analysis, league comparisons as well as predict future match results and trends.

**Problem Description**

**Motivation**

Simply speaking, soccer can be regarded as one of the most popular sports around the world. One regular match could attract thousands of people to purchase tickets. Take European Soccer market as one example, different roles may want to get access to different information. For bosses of clubs, they really want to spend limited money to trade promising players. Besides, in order to find potential players, they need a comprehensive report about the personal values for each player for evaluating. As for players, they want to play in the most comfortable and popular courtiers/leagues . They are eager to become teammates with potential players so that it is easier for them to win the championship. For our fans, we want to take part in some lotteries to predict the winner teams. Therefore, our team builds a multifunctional query system for different demands

Here is the main tasks we are going to solve:

* What is the best relationship for player’s height so that they could be more likely to own higher personal values?
* What is the teams rating in one specific season in terms of top players’ potential values?
* Which league/teams that scored the most in a given season?
* How the distribution of countries where lead players prefer to play for is like?
* Which team has the highest ratio of total cost as well as players’ ability values(that is cost-efficiency)?
* When given a home team and away team, which team is more likely to win the match based on their past records?
* What is the competition trends like being offensive-oriented currently?

**Dataset**

Our original datasets include two data source. One is about information on teams, players and match records of major leagues in major European countries from 2008 to 2016 and the other one is about the latest (2019) player information in the FIFA database.

**Design**

Our group firstly concludes the main entities and draw the entity relational diagram:

A picture containing text

Description automatically generated

**Figure 1.** Entity Relational Diagram

* Each *Country* has multiple Leagues. Each *Country* is identified by country\_id and has its country\_name.
* *Leagues* are tracked through league\_id and have league\_name. There are multiple teams in one league and *Leagues* also take the responsibility to hold various competitions.
* *Competitions* were held by leagues and teams will compete in *competitions*. Each competition must have a unique competition\_id, its holding season, stage, holding date as well as the game score.
* There are multiple *Teams*, each with a unique team\_id, team name, establishment date, team\_size and the number of champions achieved. *Teams* will participate in different competitions and they are governed by leagues.
* *Players* work for teams. Each player has a unique player\_id, name, birthday, height, weight. For different players, their performances are evaluated to judge their values.
* *Performance* is used to track what a player performs in his/her matches. It has performance\_id, values for potential, attacking\_work\_rate, defensive\_work\_rate, penalties, jumping, strength, reactions as well as agility. *Performance* is entirely dependent on the player.

**Methods**

**Data loaded**

Our dataset has two data sources, one is in SQLite format and the other is in csv format. We take different methods for data loading.

* For data in SQLite format, it is a complete database with nine tables in it. Using SQLiteSpy, we can directly export the tables and data into SQL format files, after modifying some incompatible syntax, we can insert them into our PostgreSQL database directly.
* For CSV files, since the data type of their attributes is character, using the import method in PostgreSQL to import data and try to modify the data type of the attribute will report errors. Therefore, we switch to using python, read the records in the csv line by line, then convert the data type according to our needs, and finally execute the insert statement to insert data. This method can make the data type of each attribute convenient for our subsequent processing and analysis.

**Data Populated**

Our original data is very complete, but many attribute values are not needed for our subsequent analysis, such as the odds of each oddsmaker for each game. Therefore, we have established 7 views to exclude unnecessary redundant data and link the data in multiple tables in the view to facilitate the efficiency of subsequent analysis.

**Data Manipulated**

After establishing the views, we created 9 functions to obtain and process the data we need. The main functions of these nine functions are: Get the league with the most goals scored in the season we input, get the team with the most goals scored in the season we input, and predict the wins and losses of the two teams in future matches. The specific contents will be described in the next section.

**Data Visualizing**

After using the function to process data, we visualize the data for analyzing their relationship or comparisons.

**Deliverables**

**Tables**

We use declarative SQLs to insert/delete/set/update data into our database. There are eight tables in our database, shown as figure 2:

A screenshot of a cell phone

Description automatically generated

**Figure 2.** Tables Used in Project

**RoadMap**

* Jian Fang focuses on importing dataset, creating players views, country views, prediction part as well as analyzing the relationship between height and players ability values.
* Siting Chen is responsible for leagueGoalviews, teamGoalviews and the main ranking functions.

*Notes:*

source codes are uploaded on <https://github.com/AllieChen02/RICE_COMP533_PROJECT>

|  |  |
| --- | --- |
| Functions & Views | Purposes |
| LeagueMatchView.sql | Create the view of LeagueMatchView |
| countryPlayerView.sql | Create the view of countryPlayerView |
| playerWithAllAttribute.sql | Create the view of playerWithAllAttributes |
| playerteamview\_V2.sql | Create the view of PlayerTeamView\_V2 |
| populate\_league\_score.sql | Create the view of LeagueGoalView |
| populate\_player\_team.sql | Create the view of PlayerTeamView |
| populate\_team\_score.sql | Create the views of HomeTeamGoalView and AwayTeamGoalView |
| getAvgScoreOfLeague.sql | Create and execute function getAvgScoreOfLeague |
| getRatioOfValueToWage.sql | Create and execute function getRatioOfValueToWage |
| getRelationBetweenHeightAndPotential.sql | Create and execute function getRelationBetweenHeightAndPotential |
| getTopScoreLeague.sql | Create and execute function getTopScoreLeague |
| getTopScoreTeam.sql | Create and execute function getTopTeamOfScore |
| gettoppotentialteam\_v2.sql | Create and execute function getTopPotentialTeam\_V2 |
| numberOfTopplayerByCountry\_rating.sql | Create and execute function numberOfTopPlayerByCountry\_rating |
| numberOfTopPlayerByCountry\_topValue.sql | Create and execute function numberOfTopPlayerByCountry\_topValue |
| predictWinner\_V2.sql | Create and execute function predictWinner\_V2 |

**Functions**

* **Part1**

Firstly, we established a view called LeagueGoalView (figure 3). This view join the match table and league table on league\_id, and join the team table on team\_api\_id, and it is grouped by each season, calculating the average goals per game for the home team and the average goals per game for the away team , the average home team goal difference and the total number of goals in the season.

A screenshot of a cell phone

Description automatically generated

**Figure 3**. LeagueGoalView View

Secondly, we created a function called getTopScoreLeague (figure 4). This function is to obtain the top league which has the highest total goals in a given season. This function requires two parameters, the first parameter is the season, and the second parameter is the number of the top ranking. For example, enter ‘2015/2016’ as the first parameter and 5 as the second parameter to return to the top five leagues in total goals scored for the 2015/2016 season.

A screenshot of a social media post

Description automatically generated

**Figure 4**. getTopScoreLeagues Function

Finally, we used the following query to execute the function and gain the result:

A screenshot of a cell phone

Description automatically generated

**Figure 5.** Example of execution of the function

* **Part2**

Firstly, we established two views HomeTeamGoalView and AwayTeamGoalView (Figure 6). One of the views is to calculate the data of the home team, and the other is to calculate the data of the away team. They all join the team table with the match table on team\_api\_id and are grouped by the team\_name and season. And then calculate total goals and average goals scored by each team as home team or away team each season.

A screenshot of a social media post

Description automatically generated

**Figure 6.** Views of HomeTeamGoalView and AwayTeamGoalView

Secondly, a function called getTopTeamOfScore is created (Figure 7). This function is to obtain the top team which has the highest total goals in a given season. This function requires two parameters, the first parameter is the season, and the second parameter is the number of the top ranking. For example, enter ‘2008/2009’’ as the first parameter and 5 as the second parameter to return to the top five teams in total goals scored for the 2008/2009 season.

A screenshot of a social media post

Description automatically generated

**Figure 7.** Function of getTopTeamOfScore

Finally, we used the following query to execute the function and gain the result (figure 8). Through the results, we can find that the team with the most goals in the 08/09 season is Barcelona, and those familiar with football know that Barcelona won the Champions League that year. Real Madrid is in third place in the query result.

A screenshot of a cell phone

Description automatically generated

**Figure 8.** Example of execution of the function

* **Part3**

Firstly, we established a view called VIEW PlayerTeamView\_V2(Figure 9). This view will obtain the player's id and name, the name of the club where the player plays, the player's ability rating, potential rating, value and salary (in euros) from the 2019 player attributes table (player\_update), and filter out records where the player's club name is empty.

A screenshot of a cell phone

Description automatically generated

**Figure 9.** View of PlayerTeamView\_V2

Secondly, based on this view we created two functions, getTopPotentialTeam\_V2(figure 10)

and getRatioOfValueToWage. The getTopPotentialTeam\_V2 function requires two parameters. The first parameter is the player's ability rating, and the second is the number of top rankings. We will calculate the average ability ratings of the players whose personal rating is greater than or equal to the first parameter in a team to represent the overall strength of this team, and sort according to this average value in descending order. For example, if the first parameter is 75 and the second parameter is 5, then the top five teams with the average ability ratings of players whose personal rating is greater than 75 will be returned.

A screenshot of a social media post

Description automatically generated

**Figure 10.** Function of getTopPotentialTeam\_V2

A screenshot of a social media post

Description automatically generated

**Figure 11.** Function of getRatioOfValueToWage

The getRatioOfValueToWage function also requires two parameters, the first is the player's ability value, and the second is the number of top rankings. The function will return the ratio of the total value and total salary of the players whose personal rating is greater than the first parameter, in ascending order. Ranked first is the team with the lowest performance / price ratio (low value, high salary).

The above two functions have a parameter (player ability ratings) to filter out some players, because there are generally some reserve players in the team, the ratings of them are very low and cannot represent the overall strength of the team, so they should be filtered.

Finally, we used the following queries to execute functions and gain the results:

A screenshot of a cell phone

Description automatically generated

**Figure 12.** Example of function getTopPotentialTeam\_V2 execution

A screenshot of a cell phone

Description automatically generated

**Figure 13.** Example of execution of the function getRatioOfValueToWage

* **Part4**

Firstly, we established a view called countryPlayerView. This view joins the player\_update table (player information in 2019), country table, team table, and match table. The final view contains the name of the country where the player plays, the player's id, the player's overall ratings, potential ratings, personal value and wages.

手机屏幕截图

描述已自动生成

**Figure 14.** View of countryPlayerView

Secondly, based on the above view, we created two functions, numberOfTopPlayerByCountry\_rating and numberOfTopPlayerByCountry\_topValue.

A screenshot of a social media post

Description automatically generated

**Figure 15.** Function of numberOfTopPlayerByCountry\_rating

A screenshot of a social media post

Description automatically generated

**Figure 16.** Function of numberOfTopPlayerByCountry\_topValue

Finally, we used the following queries to execute functions and gain the results (figure 17, figure 18)

A screenshot of a cell phone

Description automatically generated

**Figure 17.** Example of numberOfTopPlayerByCountry\_rating execution

A screenshot of a cell phone

Description automatically generated

**Figure 18**. Example of the execution of numberOfTopPlayerByCountry\_topValue

* **Part5**

Firstly, we established a view called LeagueMatchView. This view joins the match table and the league table. The final view contains the name of the league, game id, date, season, home team id, away team id, home team goal and away team goal.

A picture containing wooden, table, holding, people

Description automatically generated

**Figure 19.** View of LeagueMatchView

Secondly,We created a function called getAvgScoreOfleague(figure 20). This function has only one input, that is, the season. If there is no data for this season in the database, then the corresponding notice is returned. If there is, then the average of the number of goals in the leagues included in the database during this season will be returned.

A screenshot of a social media post

Description automatically generated

**Figure 20.** Function of getAvgScoreOfleague

Finally, we used the following query to execute the function and gain the result (figure 21). The query results are shown in the figure above. By querying the data of multiple seasons and analyzing them using statistical methods, you can find some rules.

A screenshot of a cell phone

Description automatically generated

**Figure 21.** The example of the execution of the function

* **Part6**

Firstly, we established a view called playerWithAllAttributes. (Figure 22). This view joins the player table and the player\_attributes table (contains the data from 2008 to 2016), and converts the player's birthday and record date into a timestamp format, and filters out the data of the birthday, record date and height which are empty.

A screenshot of a cell phone

Description automatically generated

**Figure 22.** The view of playerWithAllAttributes

Secondly, we created a function called getRelationBetweenHeightAndPotential (figure 23). This function has three parameters, the first is the lower limit of the players’ birthday, the second is the upper limit of the players' birthday, and the third is the year of the record. We will select the data for the year entered. If a player still has multiple data in that year, then the latest one will be selected by default. Then select the players whose birthday is within the input range, and finally group them by height to calculate the average potential ratings of these players.

A screenshot of a social media post

Description automatically generated

**Figure 23.** The function of getRelationBetweenHeightAndPotential

Finally, we used the following query to execute the function and gain the result. This query returns the relationship between height and average potential ratings for players whose birthdays were between 1992 and 1997 based on the data in 2016.

A screenshot of a social media post

Description automatically generated

**Figure 24.** The example of the execution of the function

* **Part7**

In this part, we do not establish new views, we directly use the match table.

Firstly, we created a function called predictWinner\_V2 (figure 25)

A screenshot of a social media post

Description automatically generated

**Figure 25.** The function of predictWinner\_V2

This function needs three parameters, the first parameter is the number of past games referenced (marked as N), the second is the full name of the home team, and the third is the full length of the away team. The function will match the home team and the away team's match records from 08 to 16, and select the most recent N match records, calculate the home team's goals and the away team's goals, we will predict the team who scored more goals to be the winner. if the total goals are equal, then we will predict a draw. If there is no corresponding match record or N records are not reached, the corresponding notice will also be returned.

Secondly, we used the following query to execute the function and gain the result:

A screenshot of a cell phone

Description automatically generated

**Figure 26.** The example of execution of the function

**Results**

The final part is to visualize the results:

* Acquire K leagues ranked according to their scores in one specific season

E.g. Take K= 5 and season = 2012/2013, and the best league is LIGA BBVA.

A screenshot of a cell phone

Description automatically generated

**Figure 27.** Leagues ranking according to scores in 2012/2013

* Acquire K teams ranked according to their scores in one specific season

E.g. Take K= 20 and season = 2008/2009, and the best team is FC Barcelona.

A screenshot of a cell phone

Description automatically generated

**Figure 28.** Teams ranking according to scores in 2008/2009

* Rank the best K teams according to the sum of lead player's ability values in the team

E.g. Take K= 5 and the best team is FC Barcelona (Based on the player attributes data in 2019)

A screenshot of a cell phone

Description automatically generated

**Figure 29.** Teams ranking according to top players potential value

* Acquire the distribution of countries where players whose personal value is top 100 prefer to play.

As is shown, England is the most ideal country where top players prefer to play.

A picture containing screenshot

Description automatically generated

**Figure 30.** Distributions of countries where top players prefer to play

* Rank teams by their ratio of sum of lead player’s personal value and the sum of their corresponding wages

E.g. the higher the ratio is, the more cost-efficient the team is. As is shown, the team named Derby County spend much money in introducing players, but their performance cannot match their price.

A screenshot of a cell phone

Description automatically generated

**Figure 31.** Distributions of countries where top players prefer to play

* Analyze the mode of play of 11 leagues in the past few years

The average number of goals per game in the Dutch League is first in most seasons.

A screenshot of a map

Description automatically generated

**Figure 32.** Mode of play

* Analyze the relationship between player personal value and height

E.g. by analyzing the relationship between the height and potential ratings of players whose birthdays are between 1993 and 1998.(Based on the 2016 record), We can find that there are two peaks in the height range of 168 to 173 and 193 to 198, indicating that players in these two height ranges have more potential. This is also realistic, because most football players need to be agile, so their height cannot be too high. However, some positions, such as center and guard, require tall players, so there are also high-potential players.

A screenshot of text

Description automatically generated

**Figure 33.** Relationship between height and personal value

* Predict the result of one future match by analyzing their recent K match records.

E.g. by analyzing the recent 5 matches between Liverpool and Arsenal, our team believe that Liverpool will be the winner as a home team. On the contrary, referring to the past 5 games, if Arsenal are the home team, then we predict that Arsenal will be the winner.

A screenshot of a social media post

Description automatically generated

**Figure 34.** Prediction of winner team

**Discussion**

## **Challenges**

* When importing tables of player attributes, pgAdmin always prompts the error of losing connection with the database due to large amounts of data (180000+ rows). In this case, our team use the SQL shell(pgsql) to import data, first connect to our database and then use the \i syntax to execute inserting in the SQL file, which will be more stable
* When designing the model for predicting the future match, our team firstly just counts total goals of all the past records for the home team as well the away team. However, the method was quite imprecise because matches in five years ago almost have little effects on the present and the records with other teams are also meaningless. Therefore, we must come up with another idea to do prediction. We just make use of the recent match records of the two teams to do prediction.

## **Changes from Original Plan**

The original dataset cannot form a relational database management system. In the beginning, our group is going to analyze the relationship between US household income and Median price per Sqft in different cities. However, the number of entities is limited so it is hard to find multiple relations in the two datasets.

## **Next steps**

* Improve query efficiency. For example, when we want to do an analysis of the relationship between player’s height and weight, it takes much time for querying. The main reason is that for each year, there will be many records in different months for one player. Our query should start with querying from month first and then select the latest record for analysis, which will take a large amount of time.
* Improve predictive accuracy. In our project, we predict the result for one specific match by analyzing the recent K matches between the home team and away team. In fact, we have to take the player movement, player form,etc. Into consideration to enhance predictive accuracy.

**Conclusion**

Our team successfully established a system for users to query the information for European football market. For players, they could clearly see the distribution of leagues where top players prefer to play for. For bosses of clubs, they could use this system to check the cost-efficiency of their team to adjust the team structure, avoiding spending much money on less potential players. Besides, for fans, they could easily query the most potential players in that season or with help of the system, they can participate in lotteries because they could use the system analysis to predict which team could be the winner.

Therefore, our project has practical significance and users of different roles could benefit from it. In the future, our team will pay more attention to the query efficiency to improve the system.

**Reference**

1. <https://www.kaggle.com/hugomathien/soccer>
2. <https://data.world/raghav333/fifa-players/workspace/file?filename=fifa_cleaned.csv>
3. https://www.kaggle.com/dimarudov/data-analysis-using-sql