LIVE SOCCER STATS DATABASE

CS. 601.415 Final Project Phase II

Application Domain

Our application allows users to browse raw and aggregate data from major domestic soccer leagues in Europe, including Germany, Italy, Spain, and France. We also provide mechanisms for users to query data, build custom watchlists, and bet on upcoming matches.

http://www.ugrad.cs.jhu.edu/~jxing8/

John Xing, Trenton Yuntian Wang
jxing8@jhu.edu, ywang382@jhu.edu
Dec. 18, 2018

Notes

The URL for our home page may take longer time than usual to load because it needs perform many background processes first, such as calling the API and updating the database before loading the page.

Our phase I is also attached at the end, but most of the information should be complete in phase II.

Background Information

Our database stores data related to major soccer leagues in Europe, so we provide some background information on them. Feel free to skip this section if you are already a cognoscenti.

As of now, our database has the most abundant live data about the following leagues (ongoing right now): Premier League (England), La Liga Primera (Spain), Ligue 1 (France), Bundesliga 1 (Germany), Serie A (Italy). These are the most competitive and popular leagues in their respective countries as well as the world. We also have other leagues in our database, but not enough data on them due to limits in API calls.

In each of these domestic leagues, there are about 20 teams and throughout a season lasting a full year, each team will play every other team in the league exactly twice: once at home and once away. This means that one team will play 38 matches in total in a league with 20 teams.

Matches or fixtures (these terms are synonymous and can be used interchangeably) can end in a draw, win, or lose. A team gets 3 points for a win, 1 point for a draw, and 0 point for a lose in one match. The points are added up for each team and the team with the most points after all matches are complete is crowned the league champion.

Many different statistics are particularly interesting for soccer fans. These include number of wins, loses, draws, average goals scored, average goals conceded, and so on. We briefly explain the following statistical terms:

- Standings: A league standing or table is a list of all teams in the league ranked by their current points earned, wins, goals difference, and many other stats.
- Clean sheet: A team is said to have kept a clean sheet if it conceded 0 goal during a match.
- Form: A team's most recent performance, given by letters. W for win, D for draw, L for lose.

Zip File Content

Here we provide a brief overview of the functions of the files that you should find in our submitted zip. If any of the following is missing, please contact us:

- apiCall.php: Functions to call the API, decode the data and load the database with appropriate values. These are called by index.php.
- bet.php: Bet page, including PHP codes to gather appropriate queries, call stored procedures, and display results in html tables
- contact.php: Help page, plain html for displaying text
- database.php: Database page, including PHP codes to call appropriate stored procedures
- index.php: Home page, including PHP codes to call appropriate stored procedures
- log.txt: Log we used to keep track of the transactions.
- login.php: Login page, calls server.php
- procedures.sql: All stored procedures used to create user profile and SimpleBet profile
- profile.php: My profile page, including PHP codes to call appropriate stored procedures
- public.php: Public file which all other files can access, include database connection information and error messages for the users
- queries.sql: All stored procedures for the filtering queries on the Database page
- register.php: Register page, calls server.php
- server.php: Deals with login, register and user accounts.
- table.sql: All relational tables specification in SQL Database Definition Language as well as defined triggers, checks, and assertions
- transaction.php: Takes care of PHP and SQL integrated transactions for a finished bet. Prints out transaction steps to log recovery file, commit, or rollback when there are errors found. Called by index.php

15 files in total

For many html of the pages, we included inline style sheet because css files do not seem to work properly on the JHU server for some reasons.

Our entire source code including PHP, html, and SQL codes, is submitted as required as a zip file on gradescope. You can also download the zip file at the bottom of our help page at: http://www.ugrad.cs.jhu.edu/~jxing8/contact.php Password: database415

(3) Project Description

The project description has not changed from Phase I> Our project is a dynamic soccer database storing data and statistics like leagues, teams, matches, scores, players… of some of the most popular domestic competitions in Europe, including but not limited to England, Germany, Italy, Spain, and France.

(4) Loading the Database

Our data come from api-football.com, which provides API on relevant live soccer data. We load the database with values by calling these API. For the static data like league name, we simply call the API and load the database. For dynamic data like matches, we make API calls up to 5 times per day (due to number of API calls limit) from the site.

First, we use API-call functions to retrieve data in the form of JSON. A timer is set on this call to grab new data every 6 hours. We can increase this frequency or grab data when the user logs in if we have access to more API calls (because this is not a free API).

```
// request leagues data
$uri = 'https://api-football-v1.p.rapidapi.com/leagues';
$reqPrefs['http']['method'] = 'GET';
$reqPrefs['http']['header'] = 'X-RapidAPI-Key:mGtlL0dL8QmshkBRwZ3bt92eIEssp13FVdSjsnbYn3JXpJAAXI';
$stream_context = stream_context_create($reqPrefs);
$response = file_get_contents($uri, false, $stream_context);
$leagues = json_decode($response, true);
$array_leagues = $leagues["api"]["leagues"];
```

We decode the JSON and store it in a PHP array. To update the database, we first clear the existing tables. Then, we loop through our PHP data array to repopulate the tables with updated data.

```
// populate leagues
foreach ($array_leagues as $row) {
$sql = "INSERT INTO Leagues
          VALUES ('" . $row["league_id"] ."', '" . $row["name"] ."', '" .   $row["country"] ."',
                  '" . $row["season"] ."','" .$row["season_start"] ."', '" . $row["season_end"] ."')";
mysqli_query($db, $sql);
}
```

(5) Our Database

We used mysql on dbase provided to us. We used the HW2 database, and created new tables there.

(6) User Guide

If you would want to look at the guide and use our interface concurrently, a simplified help page is available on our platform at http://www.ugrad.cs.jhu.edu/~jxing8/contact.php.

There are 3 main pages on our platform:

Database: http://www.ugrad.cs.jhu.edu/~jxing8/database.php
My Profile: http://www.ugrad.cs.jhu.edu/~jxing8/profile.php (login required)
SimpleBet: http://www.ugrad.cs.jhu.edu/~jxing8/bet.php (login required)

- In the home page, you can see basic live soccer statistics such as the matches happening today and current tables of the most popular European leagues, go to the home page. Here, the current standings for all 5 leagues are presented with each team's full team name.
- On our database page, you can query and filter different types of statistics including matches, teams, or players in our database. You can arbitrarily set the parameters which you want to filter on and simple leave the other parameters empty.
- To create an account and become a user, simply register by picking a username and your password. For your convenience, we have created some test user accounts which you can use to check out our features: However, feel free to register a new account if you like.

  Username: twang
  Password: 1

  Username: jxing
  Password: 123

  These passwords are set for convenience, you can't actually register an account with password this short for security reasons.

- You can login after creating an account. For security, you will be logged out after 30 minutes of inactivity.
- Once you are a registered user, you can create your custom team watchlist (your favorite teams), track their results, future matches, aggregate & detailed stats, squad list, and more.
- When registered, you can also bet against other registered users on upcoming matches using virtual credits. We support the simple betting scheme: "I bet user X that team Y will win in match Z". We list the betting procedures and betting rules below:
  - Every user is given 500 in credits when first registered.
  - To request a bet with someone, you need to know their username, the fixture you want to bet on, the team that you think will win, and the amount you wish to put in to the bet.
  - We identify each fixture with a unique fixture ID, which you can either find through your profile or search in the database.
  - Your opponent can choose to accept your request if they think your team will lose, or they can choose to simply reject your request.
  - If your opponent accepts your request, the bet is placed and changes cannot be made.
  - After the result of the game comes out, your balance will be automatically added or deducted by the betting amount depending on whether your team won or lost.
  - You cannot request a bet to yourself.
  - If your balance in credits drop below 5, your account will be deleted and all your progress and profiles will be lost. You will be forced to register for a new account and start over.
  - You can only place bets on matches that start in the future.
  - All bet requests on a match that have not been accepted/rejected will expire 10 minutes before the start of the match
  - If a match ends in a draw, no one will lose or gain credits from the placed bets of that match.

(7) Areas of Specialization

We touched on a variety of advanced database topics. They are categorized here and ranked from the major ones to the minor ones:

Dynamic Database & Time Sensitivity

- We use a timer to ensure that the corresponding API calls, stored procedures, and php functions are executed every 6 hours to keep all the time-sensitive data (matches, scores, etc) updated.
- We have to deal with constantly updating data, meaning that we have to keep track of the current time relative to the time of the matches, team standings etc.
- We ensure consistency in the database when updating data, old data that doesn't need updating are kept unchanged.

User Interactions & Interface:

- Our system of user accounts allows different users to customize their experience. Each user can have his unique favorite matches, query history, and receive the most relevant data based on these parameters.
- Each user has a credit balance and can use this credit balance to bet against other users on the upcoming matches through betting requests.

Transactions Control & Recovery:

- We design user-to-user transactions of credits (transfer X credits from user A to user B) to implement our betting system when a user wins a bet, see transaction.php.
- We use PHP integrated with SQL to ensure atomicity, security and consistency of transactions. All our transaction schedules are serial.
- We follow log-based recovery protocols by keeping a log of all transactions in log.txt as well as a history of transaction in the database. When errors are found, we execute transactions rollback.

Security & Design:

- We perform MD5 encryption when storing user passwords in our database to ensure the passwords are difficult to decipher for everyone.
- We use sessions to ensure that multiple users can access the platform simultaneously.
- We implement 30-minute session time-out to make sure system will not be secure and accessible if user forgets to log out.
- We have strict restrictions on user inputs to avoid malicious users from causing potential damage to our internal database.

Advanced SQL Topics:

- We implement various constraints on our tables, such as integrity, value, and foreign key constraints, checks, as well as assertions.

- We design triggers to help with the betting system, they are defined in table.sql. However, we found out later that we don't have permission to use triggers, so we have to also implement it with PHP code.

Natural Language Interfaces:

- We design multiple powerful filters which users can used to get the precise results that they want.
- These filters allow any number of parameters put in, some parameters can be left blank. This is a useful interface for users looking for specific data sets.

Optimization:

- For several tables, we optimize the database by adding indices to allow for faster index-based joins when running queries.

(8) Our Strengths

- The filters users can apply on the database page are all powered by adaptable, dynamic SQL procedures. These procedures make heavy use of the IF function to produce different queries with similar structure on the spot depending on user input. Such a design minimizes php logic, SQL code, and processing speed.
- Our front-end interface organizes relevant and specific data in a clear and east-to-navigate manner. We try to make the user experience as smooth as possible.
- We ensure that all transactions are recoverable by keeping track of transactions and performing transactions rollback immediately when errors are found.
- We always manipulate the data with time sensitivity in mind. For example, you can't bet on a past match and bet requests of past matches must be deleted.
- Our design allows users to get a personalized set of data specific to their favorite teams. This transparency makes it seems like each user has their own separate database.
- Our website does not have unnecessary redirection of pages. As soon as the user adds a team to favorite or requests a bet, the appropriate data are collected and fed to the user.
- Our platform is widely and practically applicable whether the user is a die-hard fan of a team, a soccer enthusiast, or just someone interesting in analyzing sport data in general.

(9) Limitations and Future Prospects

- Due to limit on number of API calls (the API is not free), we cannot populate all the tables that we have created in the initial design of the database. For example, some tables like odds and match_events are left empty.
- With other tables filled in with actual values, we can perform another interesting set of queries concerning players and fixtures such as type of goals scored, number of yellow cards received, etc. Users can probably also track their favorite players as well as teams.

- Due to limit of API calls, we can't really make our database as live as possible because we are updating every 6 hours. With more calls available, we can perhaps achieve real-times update, with delay from calling the API of course.
- If time permits, our SimpleBet can be expanded to a more complex betting system. Instead of users only only betting on match wins, they can bet on other attributes like players who scored, time of goals, types of goals, number of goals, and more.
- With more tables and data, we can also explore optimization of queries more by coming up with methods to reduce query processing time.

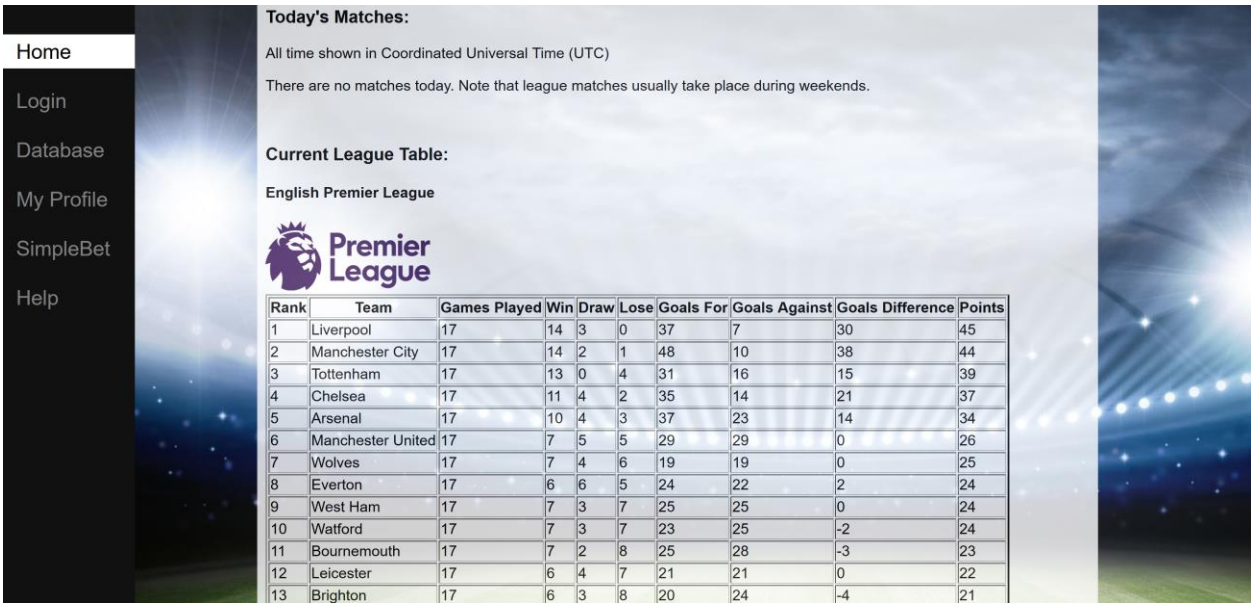
(10) Our Code

Although we sometimes refer to online guides and tutorials for help, our entire code base represents original work of the two of us.

(11) Sample Outputs

Below, we provide some screenshots of the expected outputs when using our user interface.

Home page view as of Dec.20, 2018

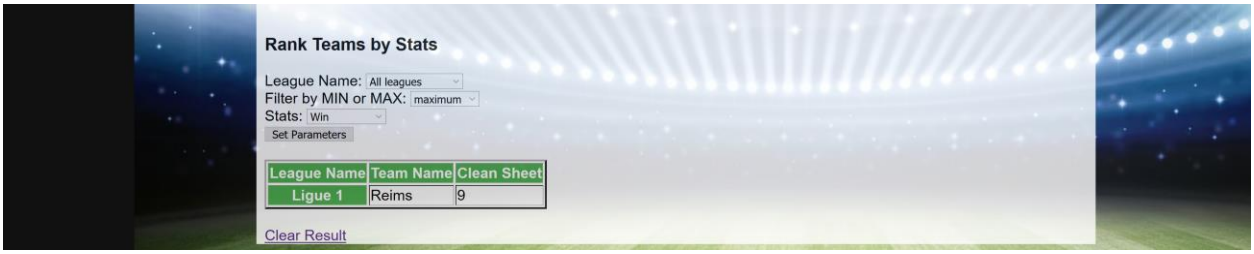


Rank	Team	Games Played	Win	Draw	Lose	Goals For	Goals Against	Goals Difference	Points
1	Liverpool	17	14	3	0	37	7	30	45
2	Manchester City	17	14	2	1	48	10	38	44
3	Tottenham	17	13	0	4	31	16	15	39
4	Chelsea	17	11	4	2	35	14	21	37
5	Arsenal	17	10	4	3	37	23	14	34
6	Manchester United	17	7	5	5	29	29	0	26
7	Wolves	17	7	4	6	19	19	0	25
8	Everton	17	6	6	5	24	22	2	24
9	West Ham	17	7	3	7	25	25	0	24
10	Watford	17	7	3	7	23	25	-2	24
11	Bournemouth	17	7	2	8	25	28	-3	23
12	Leicester	17	6	4	7	21	21	0	22
13	Brighton	17	6	3	8	20	24	-4	21

Filtering all leagues in China:

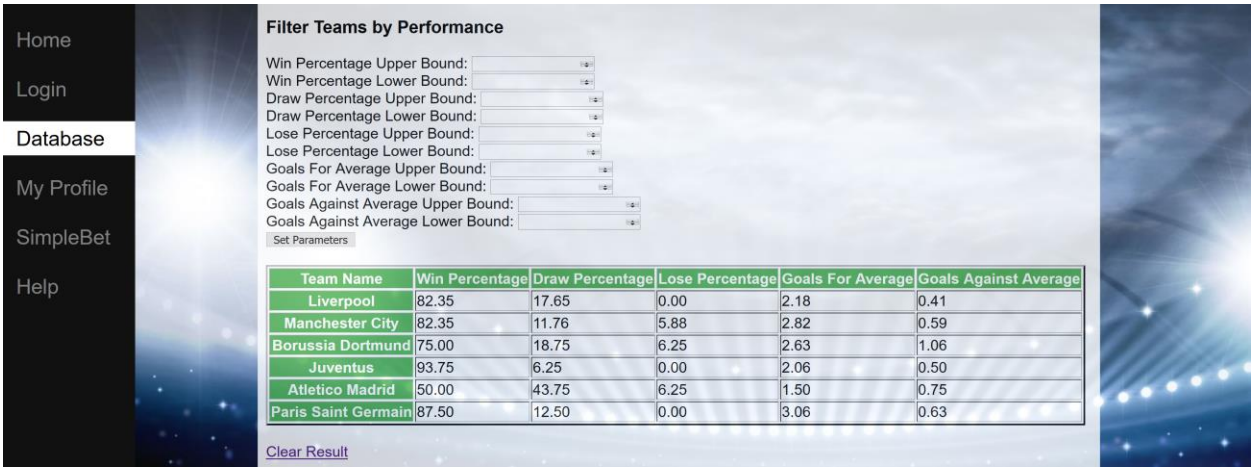
League ID	Name	Country	Season	Season Start	Season End	Standing
180	League One	China	2016	2016-03-12	2016-10-22	1
181	League One	China	2017	2017-03-11	2017-10-28	1
182	League One	China	2018	2018-03-10	2018-11-03	1
82	Super League	China	2018	2018-03-02	2018-11-11	1
83	Super League	China	2017	2017-03-03	2017-11-04	1
84	Super League	China	2016	2016-03-04	2016-10-30	1

Getting the team with the maximum clean sheets in Ligue 1:



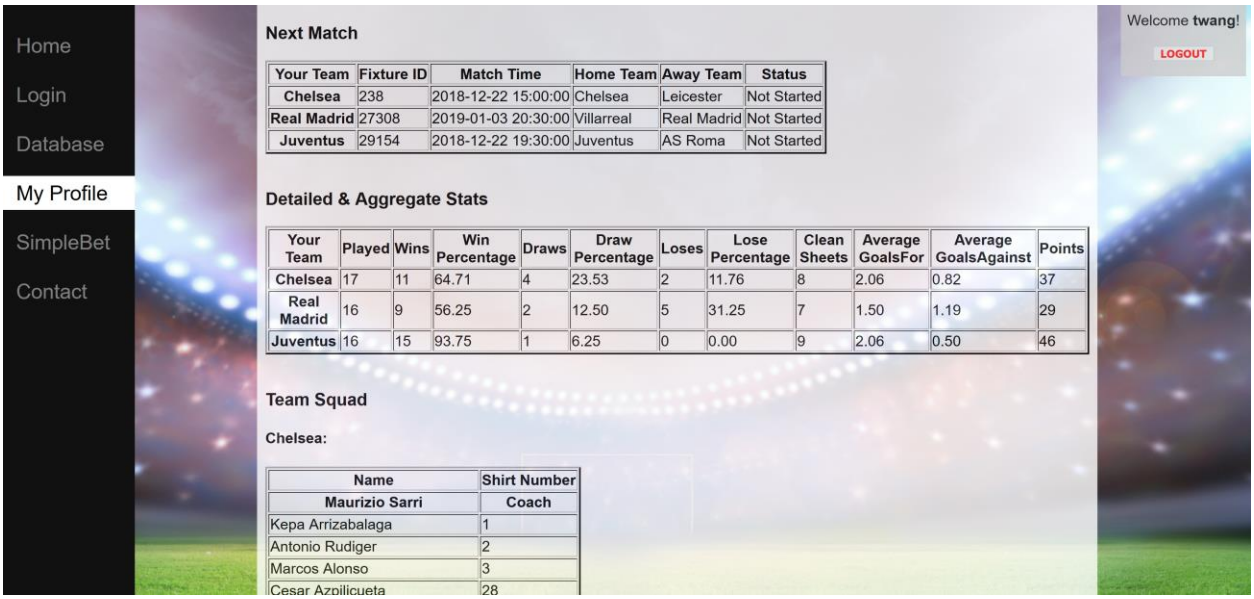
Filtering matches in Premier League that finished with the score 2 – 2:

Getting the teams with win percentage greater than 50% and lose percentage less than 10%:

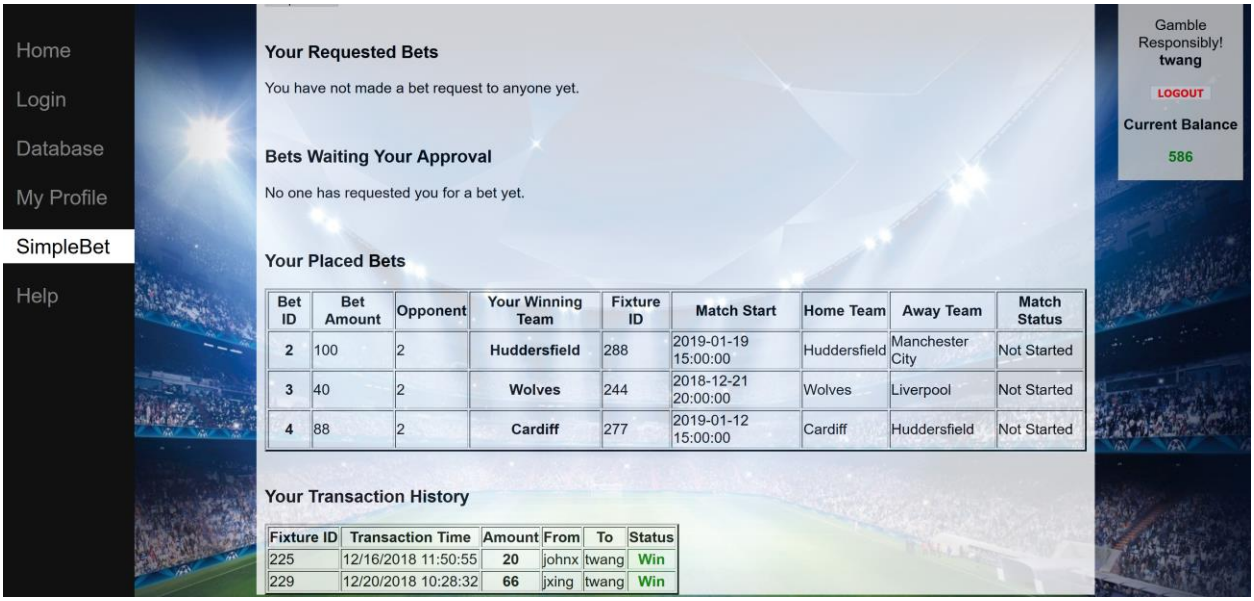


Part of jxing's profile. West Ham is his favorite team:

Part of twang's profile. He added Chelsea, Juventus, and Real Madrid to his favorite:



Part of twang's SimpleBet page, showing his placed bets and recent win/lose history:



(12) Our Tables

See tables.sql for our full implementation of tables in SQL.

(13, 14) Our Code Base

Our entire source code including PHP, html, and SQL codes, is submitted as required as a zip file on gradescope. You can also download the zip file on our help page at:
http://www.ugrad.cs.jhu.edu/~jxing8/contact.php Password: database415

Scroll down to the very bottom and enter the password, then a download link will be available on the bottom of the page.

FINAL PROJECT PHASE I

(1)

Name	*JHID*	*section*	*email*
John Xing	Jxing8	CS 601.415	jxing8@jhu.edu
Trenton Yuntian Wang	Ywang382	CS 601.415	ywang382@jhu.edu

(2)

Our project is a dynamic soccer database storing data and statistics like leagues, teams, matches, scores, players… of some of the most popular domestic competitions in Europe, including England, Germany, Italy, Spain, and France.

(3)

1. Find the player in Chelsea who received the greatest number of yellow cards between 2018-2019.
2. Get the first-round matches from the 2018 season where the final score between the two team differ more than one.
3. Get all the matches today in which the goal difference is greater than 5.
4. Find the finished matches in which the home team conceded more than 2 goals.
5. Get the leagues with the greatest number of teams whose team name starts with the letter B.
6. List the leagues in which two different players share the exact same name.
7. Get the all players whose player number is between 10 and 16 and who play for a team in Germany.
8. Find all teams that have won more often than they have lost in a particular season.
9. Get all upcoming matches where the odds for win is higher than the sum of odds for draw and lose for the home team.
10. Find all players who have scored more header goals than goals by left or right foot in the 2018 season.
11. List all players who have scored a hattrick for their club.
12. Find the matches in the upcoming week with the greatest odds that the home team will win.
13. List all pairs of Premier League matches in 2018 in which the two both ended with the same final score.
14. List all the teams with more clean sheets than wins for the entire season.
15. Find the Ligue 1 teams that have both lost and won against a same team in the 2018 season.
16. Find the matches in Bundesliga 1 in which the two teams used the same formation and the match ended in a draw.
17. Find the teams in which the coach and one of the players share the same name.
18. Find all the teams that have scored more goals at home than away but have had more draws away than at home.
19. Find the formation of the starting lineup which leads to the greatest number of wins in the database.
20. List all the players who have came on as a substitute for a match and scored during the 2017 season.

21. List all the players in Premier League ranked by number of goals scored, yellow cards received, and red cards received.
22. Find all matches in which one player scored a goal but was sent off by a red card later on in the match.
23. Find the coach of the team that scores the greatest number of goals in the 2017 season.

(4)

For our relational data model, we provide some examples of the tables which we created in our database with some sample tuples inserted as well. For the complete list of the tables in our database, including the user related tables, see tables.sql in submitted zip files.

LEAGUES:

League id	name	country	season	season_start	season_end	standings
112	Super Lig	Turkey	2017	2017-08-11	2018-05-19	true

FIXTURES:

Fixture id	Event_time	Event_date	League_id	round	homeTeam	awayTeam
225	1534609800	2018-08-11 15:00:00	2	Ligue 1 - 3	Paris Saint Germain	AS Monaco

status	goalHomeTeam	goalAwayTeam	Final_score
Match Finished	3	1	3 - 1

STANDINGS:

League id	Team name	Team id	played	win	draw	lose	goalsFor	goalsAgainst	goalsDiff
2	Chelsea	49	17	11	4	2	14	21	37

PLAYERS (Coach):

Team id	Pname	Pnumber
49	Antonio Rudiger	2
49	Mauricio Sarri	Coach

LineUps:

Fixture_id	coach	formation	captain	Starting11	substitues
225	Mauricio Sarri	4-3-3	Gary Cahill	*List of players*	*List of players*

MatchEvents:

Event id	Fixture_id	Event_time	Team_name	player	Type	Detail
13334	225	47	Chelsea	Eden Hazard	Goal	Header

Odds:

Fixture id	homeTeamWin	awayTeamWin	draw	bothTeamsScore	Over_3_goals
225	1.83	12.24	8.25	2.05	3.70

(5)

Below are a set of SQL statements that can implement some of the queries listed above.

```
--1
SELECT p.pname, p.pnumber
FROM Players AS p, Standings AS s, MatchEvents AS m,
    (SELECT MAX(COUNT(DISTINCT m1.event_id)) AS max_count
     FROM Players AS p1, MatchEvents AS m1
     INNER JOIN Standings AS s1 ON m1.team_name = s1.team_name
     WHERE m1.team_name = 'Chelsea' AND s1.team_id = p1.team_id AND
           m1.player = p1.pname AND m1.type = 'Yellow Card' AND
           m1.event_time > 2018 AND m1.event_time < 2019
           GROUP BY p1.pname) AS counts
WHERE m.team_name = 'Chelsea'
    AND m.team_name = s.team_name AND s.team_id = p.team_id
    AND m.player = p.pname AND m.type = 'Yellow Card'
    AND m.event_time > 2018 AND m.event_time < 2019
GROUP BY p.pname
HAVING COUNT(DISTINCT m.event_id) = counts.max_count;

--2
SELECT p.pname, p.pnumber
FROM Fixtures as f, Standings as s, Players as p, Leagues as l
WHERE f.league_id = l.league_id AND l.season = 2018 AND
      f.round = 1 AND ABS(f.goalHomeTeam - f.goalAwayTeam) > 1 AND
      f.league_id = s.league_id AND s.team_id = p.team_id AND f.round = 1
GROUP BY p.team_id;

--5
SELECT name
FROM Leagues as l, Standings as s
WHERE l.league_id = s.league_id AND s.team_name LIKE 'B%'
GROUP BY l.league_id
HAVING COUNT(DISTINCT team_id) >= ALL(
  SELECT COUNT(DISTINCT team_id)
  FROM Leagues as l, Standings as s
  WHERE l.league_id = s.league_id AND s.team_name LIKE 'B%'
  GROUP BY l.league_id
);



--10
SELECT p.pname
FROM Players as p, MatchEvents as me, Fixtures as f, Leagues as l
WHERE p.pname = me.player AND me.fixture_id = f.fixture_id AND f.league_id = l.league_id
  AND me.type = "Goal" AND me.detail = "Header" AND l.season = 2018
  AND COUNT(me.event_id) >= ALL(
    SELECT COUNT(me1.event_id)
    FROM Players as p1, MatchEvents as me1
    WHERE p1.pname = p.pname AND p1.pname = me1.player
      AND me1.type = "Goal"
      AND (me.detail = "Left foot" or me.detail = "Right foot")
    )
GROUP BY p.pname;
```

```
--11
SELECT p.pname
FROM Players as p, MatchEvents as me, Fixtures as f
WHERE p.pname = me.player AND me.fixture_id = f.fixture_id AND me.type = "Goal"
GROUP BY f.fixture_id
HAVING COUNT(event_id) >= 3;


--16
SELECT f.fixture_id, f.event_date, f.homeTeam, f.awayTeam, f.final_score
FROM Fixtures as f, LineUps as lu1, LineUps as lu2, Leagues as l
WHERE lu1.fixture_id = lu2.fixture_id
  AND f.fixture_id = lu1.fixture_id AND lu1.league_id = l.league_id
  AND l.name = "Bundesliga 1" AND lu1.team_name = f.homeTeam
  AND lu2.team_name = f.awayTeam AND lu1.formation = lu2.formation
  AND f.goalHomeTeam = f.goalAwayTeam;

--17
SELECT team_name
FROM Standings as s, Players as p
WHERE s.team_id = p.team_id AND p.pname IN(
  SELECT pname FROM Players as p1
  WHERE p1.team_id = p.team_id AND p1.pnumber = "Coach"
);

--19
SELECT formation
FROM Fixtures as f, LineUps as lu, Standins as s
WHERE f.fixture_id = lu.fixture_id AND lu.team = s.team_name
  AND IF(f.homeTeam = lu.team, f.goalHomeTeam > f.goalAwayTeam, f.goalHomeTeam < f.goalAwayTeam)
GROUP BY f.formation
HAVING COUNT(f.fixture_id) >= ALL(
  SELECT COUNT(f.fixture_id)
  FROM Fixtures as f, LineUps as lu, Standins as s
  WHERE f.fixture_id = lu.fixture_id AND lu.team = s.team_name
  AND IF(f.homeTeam = lu.team, f.goalHomeTeam > f.goalAwayTeam, f.goalHomeTeam < f.goalAwayTeam)
  GROUP BY f.formation
);

--22
SELECT f.fixture_id, f.event_date, f.homeTeam, f.awayTeam, f.final_score
FROM Fixtures as f, MatchEvets as m1, MatchEvents as m2
WHERE m1.fixture_id = m2.fixture_id AND f.fixture_id = m1.fixture_id
  AND m1.player = m2.player AND m1.time < m2.time
  AND m1.type = "Goal" AND m2.type = "Red Card";
```

(6)

We obtain the data we need up to 5 times per day (due to number of API calls limit) from api-football.com API. We then decode the JSON file from the API calls and then interpret the data from JSON and loading them to the database.

(7)

As a sports database, our outputs are data that are specified by our users. These include results primarily relevant time-related data (most recent score/ scores in past match), aggregate statics (total scores in specific season), and derived statistics (win-rate, avg goals per game, etc)

Also, we will provide each user with their unique set of data concerning their favorite teams, namely, their teams' recent results, future matches, squad, and forms.

In addition, we host a betting platform in which users can bet on future matches with other users on future matches using some sort of currencies.

In general, we hope to achieve a user-friendly and easy-to-navigate web interface for our frontend views.

(8)

We focused on a variety of advanced topics as listed below.

Security & Design, Dynamic Database, User Interactions, Advanced SQL Topics, Transactions & Recovery, Natural Language Interfaces, Optimization.