Data Viz France project documentation

Introduction to the project

The aim of this project is to continue the work carried out during my M1 dissertation: Comparative analysis of 3 performance factors in soccer: the impact of the 1st goal, the temporal distribution of goals and the influence of home advantage on the match (home/away) between youth teams (U17N and U19N) and the professional world (Ligue 1).

In order to extend this analysis, this time we will compare the following competitions, and those of the 2021/2022 to 2024/2025 seasons (where possible):

- Ligue 1
- Ligue 2
- National 1
- National 2
- Championnat U19N
- D1 Féminine
- D2 Féminine

As a reminder, we'll be looking at the following factors:

- the influence of the 1st goal on the match
- the temporal distribution of goals
- the influence of the home/away parameter
- the home advantage x 1st goal
- the number of goals per match

This work will be carried out in several stages:
- retrieval of data using web scrapping from the Sofa Score site
- storage of this data in tables using Supabase
- analysis of the data collected
- layout of the web application
- deployment of this application.

Web scraping / Supabase data storage

Importing libraries

We'll then import some libraries that will be useful later on, such as pandas, numpy and BeautifulSoup. Note that you'll need to apply the following line in your terminal in order to install all the dependencies required for the project to function properly:

```
pip install -r requirements.txt
```

Link to Supabase database

We'll use environment variables to store our personal data (project url and associated api key) to access the project on Supabase.

Creation of queries to create tables on Supabase (to be done on Supabase)

Please also create these tables within your project directly on Supabase. We'll add the data later in the project.

- Creating the Competition table

```
CREATE TABLE Competition (id_competition SERIAL PRIMARY KEY,competition_name
VARCHAR(255) NOT NULL,country_name VARCHAR(255) NOT NULL,link_url TEXT
NOT NULL);
```

- Creating the Season table

```
CREATE TABLE Season (id_season SERIAL PRIMARY KEY,season_name VARCHAR(255)
NOT NULL,id_competition INT NOT NULL,link_url TEXT NOT NULL,CONSTRAINT
fk_competition FOREIGN KEY (id_competition) REFERENCES
Competition(id_competition) ON DELETE CASCADE);
```

- Creating the Team table

```
CREATE TABLE team (id_team INT PRIMARY KEY,team_name TEXT NOT NULL);
```

- Creating the match information table

```
CREATE TABLE info_match (id_match INT PRIMARY KEY,id_season INT NOT
NULL,id_home_team INT NOT NULL,id_away_team INT NOT NULL,match_date DATE
NOT NULL,link_url TEXT,FOREIGN KEY (id_home_team) REFERENCES
team(id_team),FOREIGN KEY (id_away_team) REFERENCES team(id_team));
```

- Creation of a table of match goal information

```
CREATE TABLE Info_goal (id_match INT NOT NULL,score_home INT,score_away
INT,result INT,home_0_15 INT,away_0_15 INT,home_16_30 INT,away_16_30
INT,home_31_45 INT,away_31_45 INT,home_46_60 INT,away_46_60 INT,home_61_75
INT,away_61_75 INT,home_76_90 INT,away_76_90 INT,PRIMARY KEY
(id_match),FOREIGN KEY (id_match) REFERENCES info_match(id_match));
```

Store information on French competitions

Next, we'll create the Competition class, and its function for inserting this data into the associated Supabase table (created earlier).

To retrieve the competition data, we'll do some web scrapping from the Sofa Score site and the link below:
- https://www.sofascore.com/fr/football/france.

We'll infll into the source page to retrieve the information for each competition via the BeautifulSoup library. After targeting the class containing this information, we'll store the following data:

- the competition identifier
- the competition name
- the competition country name

- the competition url link.

These will then be stored in a dataframe, and inserted into the associated table via the insert_competition function created earlier.

Store information on French seasons

Next, with the help of the psycopg2 library, we'll access the previously created table, in order to store the season information available via the competitions table. We'll use the conn library to perform the following query: SELECT id_competition, link_url FROM competition. This query returns the identifiers and URL links for each of the French competitions available on the Sofascore site.

Following the same logic as the previous section, we'll create the Season class and its associated function. This class will contain the following information:

- the season identifier
- the season name
- the competition identifier
- the season url link.

We'll then initialize a driver to perform our web scraping from the previously created request. Note that the following competitions will not be included: Coupe de France, Trophée des Champions, Coupe de France Féminine, in order to focus on competitions that are not knockout events, and are more long-term in nature. In order to analyze recent trends, we will only be looking at the last 4 French seasons (from 2021/2022 to 2024/2025).

To summarize the main steps involved in storing this information, the first logical step is to connect to the competition link in the associated table. We then close the cookie page, thus blocking data collection. For your information, this site has a drop-down menu for selecting the season of your choice. We'll use it (as far as possible) to click on the following seasons:

- 2021/2022
- 2022/2023
- 2023/2024
- 2024/2025 (or 2024/25).

The essential information appearing on the 1st page of these seasons will be collected in an object, before moving on to the next season by performing the same operation. The information will then be put into dataframe format, and stored in the season table via the insert_seasons function. We close the driver once the task has been completed.

Search for match information and associated teams

For this stage, we will collect information about each match and the teams involved from the seasons stored previously. We'll use the same logic as before:

- Access the season table stored on Supabase
- Retrieve the data using the following query: SELECT id_season, link_url FROM season, giving the season identifier and its url link
- Create a Team class, containing the team identifier and its name
- Create an insert_team function

- Create a Match class, containing the match identifier, season, home and away team, the date of the match, and its url link
- Create an insert_matches function.

Given the greater length of this stage in the final aim of collecting data on the teams and the associated match, we'll divide each task by function. Note that the logic remains similar to the previous sections:

- Open a driver for our web scraping
- Close the cookies page when necessary
- Retrieve a season's page from the query created previously
- Store the following information for each match of the current day in an object (not taking into account matches postponed, abandoned, or giving rise to a green carpet
- Press the button to access the previous day when all the matches for the current day have been collected
- Past seasons already collected are not taken into account when searching for data to insert (speeds up data collection)
- Information for these matches is stored again, and teams, up to the 1st day, to finally move on to the next season
- Data is put into dataframe format
- Use of the insert_teams and insert_matchs functions to store all this on our Supabase project
- Closure of the driver once all the tasks have been completed.

Retrieving information on the goals in each match

Finally, for this stage, we're going to collect the goal information for each match stored previously. We'll use the same logic as before:
- Access the info_match table stored on Supabase
- Retrieve the data using the following query: SELECT id_match, link_url, id_season FROM info_match, giving the match identifier, its season and its url link
- Retrieve information on all season identifiers and their names using the following query: SELECT DISTINCT s.id_season, s.season_name FROM Season s JOIN info_match im ON s.id_season = im. id_season; This will ask the user which seasons they wish to store
- Retrieve match identifiers already present in the goals database, so that the same information is not retrieved again when it has already been stored previously
- Create a Goal class, containing the match identifier, the home and away team score, the match result, the number of goals scored by each team per 15-minute period and the influence of the 1st goal on the match
- Create an insert_goals function.

Once again, given the greater length of this stage in the final aim of collecting goal data for each match, we will divide each task by function. It is important to stress that the user is asked for the seasons they wish to collect in order to reduce the chances of it crashing (if too much data needs to be stored). Note that the logic remains similar to the previous sections:

- Initialising a driver for our web scraping. Note that superfluous visualisations will be removed in order to speed up the collection of information
- Closing the cookies page when necessary

- Retrieving a match page from the previously created request (without taking into account matches already collected)
- Accessing the incidents section containing the match highlights, including each goal scored in the match and the final score
- Extraction of the match score via the homeScore and awayScore columns
- Deduction of the match result based on the previous function (home win, draw or away win)
- Extraction of goal information by interval using the incident data. All columns will be equal to 0 if the score is zero
- Deduction of the influence of the 1st goal on the match from the result function, and information on the influence of the 1st goal
- Putting all this data in dataframe format
- Not taking into account past seasons already collected, and matches already collected in the search for data to insert (speeds up data collection)
- Resetting the driver every 10 matches, to reduce the chances of the code crashing
- Using the insert_goals function to store all this on our Supabase project
- Closing the driver once all the tasks have been completed.

List of pages and sections

There will be 4 separate pages with associated queries:

- analysis of a team
- analysis of matches between two teams
- analysis of a season
- analysis of a competition.

In addition, each page will contain these 5 sections:

- general goal statistics (number of goals scored and conceded, score frequency, etc.)
- the influence of the 1st goal on the match
- the time distribution of goals scored and conceded (by 15min or 45min interval)
- the influence of the home/away parameter
- comparison with other seasons/competitions/confrontations

In addition, queries will be made to find out which are the best teams through these indicators in recent years.

Project tree for setting up the application

```
Application
├── Image
├── CV
├── Mémoire
├── Documentation
├── Accueil.py : Main
├── README.md
└── requirements.txt
```

Setting up the application

Home

This page is used to display the objectives of this project, along with various resources for exploring it in depth (Memoir, Code, Documentation), and to find out more about myself (CV in English and French).

Analysis section

At the head of the project

Logically, we start each of the files reserved for data analysis by importing the libraries (streamlit, pandas, matplotlib, plotly, seaborn, supabase and python-dotenv). Next, the page title needs to be placed at the beginning of the code, so it will have this structure for our project: st.set_page_config(page_title="Data Viz ⚽ 🇫🇷", page_icon="📊", layout="wide").

Next, we'll access the environment variables for the project name and the anon key. This information is essential for connecting to the Supabase database as an anonymous user.

Initialising functions

Using the supabase.rpc function, we will create functions for each query in order to retrieve information from each procedure (previously created in Supabase). To do this, we need to enter the name of the procedure (e.g. get_first_goal_season) and the list of parameters to be entered (e.g. season_name).

We'll also create a function to round up values to two decimal places if necessary, or to an integer if required. Finally, functions will be used to highlight the teams/seasons/competitions chosen by the user in a distinctive colour within a dataframe.

Application display

A horizontal bar will be created with the pages available in the pages folder, and a request to select the team/season/competition of the user's choice according to the type of page. Based on the choices made by the user, he or she will be presented with the 5 sections detailed above, in which all the associated graphs or tables appear. Users click on the section of their choice to access the associated information.

Each part of the code linked to the display of a graph/table/gauge follows the same logic.
- Initially, the data will be retrieved using the corresponding query function,
- Data will be put into a dataframe by associating the data with an associated column name,
- Graphs/tables/gauges will be displayed using functions from plotly, dataframe. pie, or pivot,
- Centring and choice of title, and processing of numerical values to display them correctly (colouring of the chosen column if necessary).

A for loop will be created to retrieve all the information over several seasons in the Comparison between teams/seasons/competition section. In addition, the maximum data for a season or competition will be stored if the gauges are displayed. Finally, if there is not enough data on a particular aspect of a team, the associated graphs will not be displayed.

Appendix: List of SQL queries/procedures

- Information on team analysis

- 1 / Team search

```
create or replace function get_teams()
returns setof text
language sql
as $$
   select distinct team_name from team;
$$;

grant execute on function get_teams() to anon;
```

- 2/ Search for seasons available for a given team

```
create or replace function get_seasons(team_name_input text)
returns setof text
language sql
as $$
   select season_name
   from (
      -- Selection of seasons when the team was at home
      select s.season_name
      from info_match im
      join season s on im.id_season = s.id_season
      join team t on im.id_home_team = t.id_team
      where t.team_name = team_name_input
      union all
      -- Selection of seasons when the team was away from home
      select s.season_name
      from info_match im
      join season s on im.id_season = s.id_season
      join team t on im.id_away_team = t.id_team
      where t.team_name = team_name_input
   ) as all_matches
   group by season_name
   having count(*) >= 5
   order by season_name desc;

$$;

grant execute on function get_seasons(text) to anon;
```

- 3/ Search for goal statistics for a given team

```
create or replace function get_avg_goals_stats(season_name_input text)
returns table (
   season_name text,
   team_name text,
   avg_goals_per_match numeric(10,2),
   avg_team_goals_per_match numeric(10,2),
   avg_team_goals_conceded_per_match numeric(10,2),
   avg_team_home_goals numeric(10,2),
   avg_team_away_goals numeric(10,2),
   avg_conceded_home_goals numeric(10,2),
   avg_conceded_away_goals numeric(10,2)
)
language sql
as $$
   select
       -- Season name
       s.season_name,
       -- Team name
       t.team_name,
       -- Average number of goals per game
       ROUND(avg(ig.score_home + ig.score_away), 2) as avg_goals_per_match,
       -- Average number of goals scored per game for a team
       ROUND(avg(case
           when im.id_home_team = t.id_team then ig.score_home
           when im.id_away_team = t.id_team then ig.score_away
       end), 2) as avg_team_goals_per_match,
       -- Average number of goals conceded per game for a team
       ROUND(avg(case
           when im.id_home_team = t.id_team then ig.score_away
           when im.id_away_team = t.id_team then ig.score_home
       end), 2) as avg_team_goals_conceded_per_match,
       -- Average number of goals scored per game for a team at home
       ROUND(avg(case when im.id_home_team = t.id_team then ig.score_home end), 2) as
avg_team_home_goals,
       -- Average number of goals scored per game for a team at away
       ROUND(avg(case when im.id_away_team = t.id_team then ig.score_away end), 2) as
avg_team_away_goals,
       -- Average number of goals conceded per game for a team at home
       ROUND(avg(case when im.id_home_team = t.id_team then ig.score_away end), 2) as
avg_conceded_home_goals,
       -- Average number of goals conceded per game for a team at away
       ROUND(avg(case when im.id_away_team = t.id_team then ig.score_home end), 2) as
avg_conceded_away_goals
   from info_match im
   join season s on im.id_season = s.id_season
   join team t on im.id_home_team = t.id_team or im.id_away_team = t.id_team
   join info_goal ig on im.id_match = ig.id_match
   where s.season_name = season_name_input
   group by s.season_name, t.team_name;
$$;

grant execute on function get_avg_goals_stats(text) to anon;
```

- 4/ Find the goals scored by a given team

```
create or replace function get_goals_scored(season_name_input text)
```

```
returns table (
    team_name text,
    total_goals_scored numeric(10,2),
    avg_goals_scored numeric(10,2),
    goals_scored_home numeric(10,2),
    avg_goals_scored_home numeric(10,2),
    goals_scored_away numeric(10,2),
    avg_goals_scored_away numeric(10,2)
)
language sql
as $$
    select
        -- Team name
        team_name,
        -- Total goals scored
        coalesce(sum(case when is_home = 1 then score_home else score_away end), 0) as
total_goals_scored,
        -- Average goals scored per match
        round(coalesce(sum(case when is_home = 1 then score_home else score_away end), 0) *
1.0 / nullif(count(*), 0), 2) as avg_goals_scored,
        -- Total goals scored per match at home
        coalesce(sum(case when is_home = 1 then score_home else 0 end), 0) as
goals_scored_home,
        -- Average goals scored per match at home
        round(coalesce(sum(case when is_home = 1 then score_home else 0 end), 0) * 1.0 /
nullif(count(case when is_home = 1 then 1 else null end), 0), 2) as avg_goals_scored_home,
        -- Total goals scored per match at away
        coalesce(sum(case when is_home = 0 then score_away else 0 end), 0) as
goals_scored_away,
        -- Average goals scored per match at away
        round(coalesce(sum(case when is_home = 0 then score_away else 0 end), 0) * 1.0 /
nullif(count(case when is_home = 0 then 1 else null end), 0), 2) as avg_goals_scored_away
    from (
        -- Home
        select t.team_name, ig.score_home, ig.score_away, 1 as is_home
        from info_match im
        join season s on im.id_season = s.id_season
        join team t on im.id_home_team = t.id_team
        join info_goal ig on ig.id_match = im.id_match
        where s.season_name = season_name_input
        union all
        -- Away
        select t.team_name, ig.score_home, ig.score_away, 0 as is_home
        from info_match im
        join season s on im.id_season = s.id_season
        join team t on im.id_away_team = t.id_team
        join info_goal ig on ig.id_match = im.id_match
        where s.season_name = season_name_input
    ) as all_matches
    group by team_name
```

```
  having count(*) >= 5
  order by total_goals_scored desc;
$$;

grant execute on function get_goals_scored(text) to anon;
```

- 5/ Find the goals conceded by a given team

```
CREATE OR REPLACE FUNCTION get_goals_conceded(season_name_input TEXT)
RETURNS TABLE (
  team_name TEXT,
  total_goals_conceded NUMERIC(10,2),
  avg_goals_conceded NUMERIC(10,2),
  goals_conceded_home NUMERIC(10,2),
  avg_goals_conceded_home NUMERIC(10,2),
  goals_conceded_away NUMERIC(10,2),
  avg_goals_conceded_away NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
  SELECT
    -- Team Name
    team_name,
    -- Total goals conceded
    SUM(CASE WHEN is_home = 1 THEN score_away ELSE 0 END) +
    SUM(CASE WHEN is_home = 0 THEN score_home ELSE 0 END) AS
total_goals_conceded,
    -- Average goals conceded
    ROUND(
      (SUM(CASE WHEN is_home = 1 THEN score_away ELSE 0 END) +
       SUM(CASE WHEN is_home = 0 THEN score_home ELSE 0 END)) * 1.0 /
NULLIF(COUNT(*), 0), 2
    ) AS avg_goals_conceded,
    -- Total goals conceded at home
    SUM(CASE WHEN is_home = 1 THEN score_away ELSE 0 END) AS
goals_conceded_home,
    -- Average goals conceded at home
    ROUND(
      SUM(CASE WHEN is_home = 1 THEN score_away ELSE 0 END) * 1.0 /
      NULLIF(COUNT(CASE WHEN is_home = 1 THEN 1 ELSE NULL END), 0), 2
    ) AS avg_goals_conceded_home,
    -- Total goals conceded at away
    SUM(CASE WHEN is_home = 0 THEN score_home ELSE 0 END) AS
goals_conceded_away,
    -- Average goals conceded at away
    ROUND(
      SUM(CASE WHEN is_home = 0 THEN score_home ELSE 0 END) * 1.0 /
      NULLIF(COUNT(CASE WHEN is_home = 0 THEN 1 ELSE NULL END), 0), 2
    ) AS avg_goals_conceded_away
  FROM (
    -- Data from match at home
    SELECT t.team_name, ig.score_home, ig.score_away, 1 AS is_home, s.season_name
    FROM info_match im
    JOIN season s USING(id_season)
    JOIN team t ON im.id_home_team = t.id_team
    JOIN info_goal ig USING(id_match)
    UNION ALL
    -- Date from match at away
    SELECT t.team_name, ig.score_home, ig.score_away, 0 AS is_home, s.season_name
    FROM info_match im
    JOIN season s USING(id_season)
    JOIN team t ON im.id_away_team = t.id_team
    JOIN info_goal ig USING(id_match)
  ) AS all_matches
  -- Filter per season
  WHERE season_name = season_name_input
  -- Regrouper par équipe
  GROUP BY team_name
  -- Filter for the team who have more than 5 matches played
  HAVING COUNT(*) >= 5
  -- Goals conceded in descending order
  ORDER BY total_goals_conceded DESC;
$$;

grant execute on function get_goals_conceded(text) to anon;
```

- 6/ Search for the frequency of scores for a given team

```
CREATE OR REPLACE FUNCTION get_frequent_score(
  team_name_input TEXT,
  season_name_input TEXT
)
RETURNS TABLE (
  score_home INT,
  score_away INT,
  percentage NUMERIC(5,2)
)
LANGUAGE SQL
AS $$
  WITH score_counts AS (
    SELECT
      -- Team name
      t.team_name,
      -- Season name
      s.season_name,
      -- Number of goals at home
      ig.score_home,
      -- Number of goals at away
      ig.score_away,
      -- Frequency of score
```

```
            COUNT(*) AS frequency
        FROM info_match im
        JOIN season s ON im.id_season = s.id_season
        JOIN team t ON im.id_home_team = t.id_team OR im.id_away_team = t.id_team
        JOIN info_goal ig ON im.id_match = ig.id_match
        WHERE s.season_name = season_name_input
        GROUP BY t.team_name, s.season_name, ig.score_home, ig.score_away
    ), total_matches AS (
        SELECT
            t.team_name,
            s.season_name,
            -- Total de matchs
            COUNT(im.id_match) AS total_matches
        FROM info_match im
        JOIN season s ON im.id_season = s.id_season
        JOIN team t ON im.id_home_team = t.id_team OR im.id_away_team = t.id_team
        WHERE s.season_name = season_name_input
        GROUP BY t.team_name, s.season_name
    )
    SELECT
        sc.score_home,
        sc.score_away,
        -- Pourcentage d'apparition du résultat
        ROUND((sc.frequency * 100.0) / NULLIF(tm.total_matches, 0), 2) AS percentage
    FROM score_counts sc
    JOIN total_matches tm ON sc.team_name = tm.team_name AND sc.season_name =
tm.season_name
    WHERE sc.team_name = team_name_input
    ORDER BY percentage DESC;
$$;

grant execute on function get_frequent_score(text,text) to anon;
```

- 7/ Search for information on the 1st goal scored or conceded by a given team

```
CREATE OR REPLACE FUNCTION get_first_goal_season(
    season_name_input TEXT
)
RETURNS TABLE (
    season_name text,
    team_name text,
    proportion_1st_goal_for numeric(10,2),
    proportion_no_goal numeric(10,2),
    proportion_1st_goal_against numeric(10,2),
    proportion_1st_goal_home_for numeric(10,2),
    proportion_no_goal_home numeric(10,2),
    proportion_1st_goal_home_against numeric(10,2),
    proportion_1st_goal_away_for numeric(10,2),
    proportion_no_goal_away numeric(10,2),
    proportion_1st_goal_away_against numeric(10,2),
    first_goal_win numeric(10,2),
    first_goal_draw numeric(10,2),
    first_goal_lose numeric(10,2),
    proportion_1st_goal_home_win numeric(10,2),
    proportion_1st_goal_home_draw numeric(10,2),
    proportion_1st_goal_home_lose numeric(10,2),
    proportion_1st_goal_away_win numeric(10,2),
    proportion_1st_goal_away_draw numeric(10,2),
    proportion_1st_goal_away_lose numeric(10,2),
    first_goal_conceded_win numeric(10,2),
    first_goal_conceded_draw numeric(10,2),
    first_goal_conceded_lose numeric(10,2),
    proportion_1st_goal_conceded_home_win numeric(10,2),
    proportion_1st_goal_conceded_home_draw numeric(10,2),
    proportion_1st_goal_conceded_home_lose numeric(10,2),
    proportion_1st_goal_conceded_away_win numeric(10,2),
    proportion_1st_goal_conceded_away_draw numeric(10,2),
    proportion_1st_goal_conceded_away_lose numeric(10,2)
)
LANGUAGE SQL
AS $$
    SELECT
        -- Season name
        season_name,
        -- Team name
        team_name,
        -- Proportion of team scoring 1st goal
        ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND is_home = 1 THEN 1 END)
+ COUNT(CASE WHEN squad_1st_goal = 2 AND is_home = 0 THEN 1 END)) * 100 /
COALESCE(NULLIF(COUNT(squad_1st_goal),
        0),1),2) AS proportion_1st_goal_for,
        -- Proportion with no goal
        ROUND((COUNT(CASE WHEN squad_1st_goal = 0 AND is_home = 1 THEN 1 END)
+ COUNT(CASE WHEN squad_1st_goal = 0 AND is_home = 0 THEN 1 END))
        * 100 / COALESCE(NULLIF(COUNT(squad_1st_goal), 0),1),2) AS
proportion_no_goal,
        -- Proportion of team conceding 1st goal
        ROUND((COUNT(CASE WHEN squad_1st_goal = 2 AND is_home = 1 THEN 1 END)
+ COUNT(CASE WHEN squad_1st_goal = 1 AND is_home = 0 THEN 1 END))
        * 100 / COALESCE(NULLIF(COUNT(squad_1st_goal), 0),1),2) AS
proportion_1st_goal_against,
        -- Proportion of home games scoring the 1st goal
        ROUND(COUNT(CASE WHEN squad_1st_goal = 1 AND is_home = 1 THEN 1 END)
* 100 / COALESCE(NULLIF(COUNT(CASE WHEN is_home = 1 THEN 1 END), 0),1),2)
AS proportion_1st_goal_home_for,
        -- Proportion of home games without a goal
        ROUND(COUNT(CASE WHEN squad_1st_goal = 0 AND is_home = 1 THEN 1 END)
* 100 / COALESCE(NULLIF(COUNT(CASE WHEN is_home = 1 THEN 1 END), 0),1),2)
AS proportion_no_goal_home,
        -- Proportion of home games where the team concedes the 1st goal
    ROUND(COUNT(CASE WHEN squad_1st_goal = 2 AND is_home = 1 THEN 1 END)
* 100 / COALESCE(NULLIF(COUNT(CASE WHEN is_home = 1 THEN 1 END), 0),1),2)
AS proportion_1st_goal_home_against,
    -- Proportion of 1st goals scored by away teams
    ROUND(COUNT(CASE WHEN squad_1st_goal = 2 AND is_home = 0 THEN 1 END)
* 100 / COALESCE(NULLIF(COUNT(CASE WHEN is_home = 0 THEN 1 END), 0),1),2)
AS proportion_1st_goal_away_for,
    -- Proportion of goals scored when playing away from home
    ROUND(COUNT(CASE WHEN squad_1st_goal = 0 AND is_home = 0 THEN 1 END)
* 100 / COALESCE(NULLIF(COUNT(CASE WHEN is_home = 0 THEN 1 END), 0),1),2)
AS proportion_no_goal_away,
    -- Proportion of 1st goal conceded when playing away from home
    ROUND(COUNT(CASE WHEN squad_1st_goal = 1 AND is_home = 0 THEN 1 END)
* 100 / COALESCE(NULLIF(COUNT(CASE WHEN is_home = 0 THEN 1 END), 0),1),2)
AS proportion_1st_goal_away_against,
    -- Proportion of wins when team scores 1st goal (usually home or away)
    ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND result = 1 AND is_home = 1
THEN 1 END) + COUNT(CASE WHEN squad_1st_goal = 2 AND result = 2 AND is_home
= 0 THEN 1 END)) * 100 / COALESCE(NULLIF((COUNT(CASE WHEN squad_1st_goal =
1 AND is_home = 1 THEN 1 END) +
    COUNT(CASE WHEN squad_1st_goal = 2 AND is_home = 0 THEN 1 END)), 0), 1),2)
AS first_goal_win,
    -- Proportion of draws where the team scores the 1st goal (usually home or away)
    ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND result = 0 AND is_home = 1
THEN 1 END) + COUNT(CASE WHEN squad_1st_goal = 2 AND result = 0 AND is_home
= 0 THEN 1 END)) * 100 / COALESCE(NULLIF((COUNT(CASE WHEN squad_1st_goal =
1 AND is_home = 1 THEN 1 END) +
    COUNT(CASE WHEN squad_1st_goal = 2 AND is_home = 0 THEN 1 END)), 0), 1),2)
AS first_goal_draw,
    -- Proportion of defeats where the team scores the 1st goal (usually home or away)
    ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND result = 2 AND is_home = 1
THEN 1 END) + COUNT(CASE WHEN squad_1st_goal = 2 AND result = 1 AND is_home
= 0 THEN 1 END)) * 100 / COALESCE(NULLIF((COUNT(CASE WHEN squad_1st_goal =
1 AND is_home = 1 THEN 1 END) +
    COUNT(CASE WHEN squad_1st_goal = 2 AND is_home = 0 THEN 1 END)), 0), 1),2)
AS first_goal_lose,
    -- Proportion of wins when home team scores 1st goal
    ROUND(COUNT(CASE WHEN squad_1st_goal = 1 AND result = 1 AND is_home = 1
THEN 1 END) * 100 / COALESCE(NULLIF(COUNT(CASE WHEN squad_1st_goal = 1
AND is_home = 1 THEN 1 END), 0), 1,2)) AS proportion_1st_goal_home_win,
    -- Proportion of draws where home team scores 1st goal
    ROUND(COUNT(CASE WHEN squad_1st_goal = 1 AND result = 0 AND is_home = 1
THEN 1 END) * 100 / COALESCE(NULLIF(COUNT(CASE WHEN squad_1st_goal = 1
AND is_home = 1 THEN 1 END), 0),
    1),2) AS proportion_1st_goal_home_draw,
    -- Proportion of defeats where home team scores 1st goal
    ROUND(COUNT(CASE WHEN squad_1st_goal = 1 AND result = 2 AND is_home = 1
THEN 1 END) * 100 / COALESCE(NULLIF(COUNT(CASE WHEN squad_1st_goal = 1
AND is_home = 1 THEN 1 END), 0),
    1),2) AS proportion_1st_goal_home_lose,
    -- Proportion of wins when away team scores 1st goal
    ROUND(COUNT(CASE WHEN squad_1st_goal = 2 AND result = 2 AND is_home = 0
THEN 1 END) * 100 / COALESCE(NULLIF(COUNT(CASE WHEN squad_1st_goal = 2
AND is_home = 0 THEN 1 END), 0), 1),2) AS proportion_1st_goal_away_win,
    -- Proportion of draws when away team scores 1st goal
    ROUND(COUNT(CASE WHEN squad_1st_goal = 2 AND result = 0 AND is_home = 0
THEN 1 END) * 100 / COALESCE(NULLIF(COUNT(CASE WHEN squad_1st_goal = 2
AND is_home = 0 THEN 1 END), 0),
    1),2) AS proportion_1st_goal_away_draw,
    -- Proportion of defeats where the away team scores the 1st goal
    ROUND(COUNT(CASE WHEN squad_1st_goal = 2 AND result = 1 AND is_home = 0
THEN 1 END) * 100 / COALESCE(NULLIF(COUNT(CASE WHEN squad_1st_goal = 2
AND is_home = 0 THEN 1 END), 0), 1),2) AS proportion_1st_goal_away_lose,
    -- Proportion of wins when team concedes 1st goal (usually home or away)
    ROUND((COUNT(CASE WHEN squad_1st_goal = 2 AND result = 1 AND is_home = 1
THEN 1 END) + COUNT(CASE WHEN squad_1st_goal = 1 AND result = 2 AND is_home
= 0 THEN 1 END)) * 100 / COALESCE(NULLIF((COUNT(CASE WHEN squad_1st_goal =
2 AND is_home = 1 THEN 1 END) +
    COUNT(CASE WHEN squad_1st_goal = 1 AND is_home = 0 THEN 1 END)), 0), 1),2)
AS first_goal_conceded_win,
    -- Proportion of draws where team concedes 1st goal (usually home or away)
    ROUND((COUNT(CASE WHEN squad_1st_goal = 2 AND result = 0 AND is_home = 1
THEN 1 END) + COUNT(CASE WHEN squad_1st_goal = 1 AND result = 0 AND is_home
= 0 THEN 1 END)) * 100 / COALESCE(NULLIF((COUNT(CASE WHEN squad_1st_goal =
2 AND is_home = 1 THEN 1 END) +
    COUNT(CASE WHEN squad_1st_goal = 1 AND is_home = 0 THEN 1 END)), 0), 1),2)
AS first_goal_conceded_draw,
    -- Proportion of defeats where team concedes 1st goal (usually home or away)
    ROUND((COUNT(CASE WHEN squad_1st_goal = 2 AND result = 2 AND is_home = 1
THEN 1 END) + COUNT(CASE WHEN squad_1st_goal = 1 AND result = 1 AND is_home
= 0 THEN 1 END)) * 100 / COALESCE(NULLIF((COUNT(CASE WHEN squad_1st_goal =
2 AND is_home = 1 THEN 1 END) +
    COUNT(CASE WHEN squad_1st_goal = 1 AND is_home = 0 THEN 1 END)), 0), 1),2)
AS first_goal_conceded_lose,
    -- Proportion of wins when home team concedes 1st goal
    ROUND(COUNT(CASE WHEN squad_1st_goal = 2 AND result = 1 AND is_home = 1
THEN 1 END) * 100 / COALESCE(NULLIF(COUNT(CASE WHEN squad_1st_goal = 2
AND is_home = 1 THEN 1 END), 0), 1),2) AS proportion_1st_goal_conceded_home_win,
    -- Proportion of draws where home team concedes 1st goal
    ROUND(COUNT(CASE WHEN squad_1st_goal = 2 AND result = 0 AND is_home = 1
THEN 1 END) * 100 / COALESCE(NULLIF(COUNT(CASE WHEN squad_1st_goal = 2
AND is_home = 1 THEN 1 END), 0), 1),2) AS proportion_1st_goal_conceded_home_draw,
    -- Proportion of defeats where home team concedes 1st goal
    ROUND(COUNT(CASE WHEN squad_1st_goal = 2 AND result = 2 AND is_home = 1
THEN 1 END) * 100 / COALESCE(NULLIF(COUNT(CASE WHEN squad_1st_goal = 2
AND is_home = 1 THEN 1 END), 0), 1),2) AS proportion_1st_goal_conceded_home_lose,
    -- Proportion of wins when away team concedes 1st goal
    ROUND(COUNT(CASE WHEN squad_1st_goal = 1 AND result = 2 AND is_home = 0
THEN 1 END) * 100 / COALESCE(NULLIF(COUNT(CASE WHEN squad_1st_goal = 1
AND is_home = 0 THEN 1 END), 0), 1),2) AS proportion_1st_goal_conceded_away_win,
    -- Proportion of draws where away team concedes 1st goal
    ROUND(COUNT(CASE WHEN squad_1st_goal = 1 AND result = 0 AND is_home = 0
THEN 1 END) * 100 / COALESCE(NULLIF(COUNT(CASE WHEN squad_1st_goal = 1
AND is_home = 0 THEN 1 END), 0), 1),2) AS proportion_1st_goal_conceded_away_draw,
         -- Proportion of defeats where the away team concedes the 1st goal
    ROUND(COUNT(CASE WHEN squad_1st_goal = 1 AND result = 1 AND is_home = 0
THEN 1 END) * 100 / COALESCE(NULLIF(COUNT(CASE WHEN squad_1st_goal = 1
AND is_home = 0 THEN 1 END), 0), 1),2) AS proportion_1st_goal_conceded_away_lose
    FROM (
    -- Data for teams playing from home
    SELECT t.team_name, ig.squad_1st_goal, 1 AS is_home, s.season_name, ig.result
    FROM info_match im
    JOIN season s USING(id_season)
    JOIN team t ON im.id_home_team = t.id_team
    JOIN info_goal ig USING(id_match)
    UNION ALL
    -- Data for teams playing from away
    SELECT t.team_name, ig.squad_1st_goal, 0 AS is_home, s.season_name, ig.result
    FROM info_match im
    JOIN season s USING(id_season)
    JOIN team t ON im.id_away_team = t.id_team
    JOIN info_goal ig USING(id_match)
    ) AS all_matches
    WHERE season_name = season_name_input
    GROUP BY season_name, team_name
    HAVING COUNT(*) >= 5;
$$;

grant execute on function get_first_goal_season(text) to anon;
```

- 8/ Find information about the distribution of goals for a given team

```
CREATE OR REPLACE FUNCTION get_distribution_goals_season(
    season_name_input TEXT
)
RETURNS TABLE (
    season_name text,
    team_name text,
    proportion_buts_inscrit_1ere_periode numeric(10,2),
    proportion_buts_inscrit_2nde_periode numeric(10,2),
    proportion_buts_0_15 numeric(10,2),
    proportion_buts_16_30 numeric(10,2),
    proportion_buts_31_45 numeric(10,2),
    proportion_buts_46_60 numeric(10,2),
    proportion_buts_61_75 numeric(10,2),
    proportion_buts_76_90 numeric(10,2),
    proportion_buts_encaissés_1ere_periode numeric(10,2),
    proportion_buts_encaissés_2nde_periode numeric(10,2),
    proportion_buts_encaissés_0_15 numeric(10,2),
    proportion_buts_encaissés_16_30 numeric(10,2),
    proportion_buts_encaissés_31_45 numeric(10,2),
    proportion_buts_encaissés_46_60 numeric(10,2),
    proportion_buts_encaissés_61_75 numeric(10,2),
    proportion_buts_encaissés_76_90 numeric(10,2),
    buts_inscrit_1ere_periode numeric(10,2),
    buts_inscrit_2nde_periode numeric(10,2),
    nbr_buts_0_15 numeric(10,2),
    nbr_buts_16_30 numeric(10,2),
    nbr_buts_31_45 numeric(10,2),
    nbr_buts_46_60 numeric(10,2),
    nbr_buts_61_75 numeric(10,2),
    nbr_buts_76_90 numeric(10,2),
    buts_encaissés_1ere_periode numeric(10,2),
    buts_encaissés_2nde_periode numeric(10,2),
    buts_encaissés_0_15 numeric(10,2),
    buts_encaissés_16_30 numeric(10,2),
    buts_encaissés_31_45 numeric(10,2),
    buts_encaissés_46_60 numeric(10,2),
    buts_encaissés_61_75 numeric(10,2),
    buts_encaissés_76_90 numeric(10,2)
)
LANGUAGE SQL
AS $$
    SELECT
        -- Season name
        season_name,
        -- Team name
        team_name,
        -- Proportion of goals scored in the 1st half
        ROUND(SUM(CASE WHEN is_home = 1 THEN home_0_15 + home_16_30 +
home_31_45 ELSE away_0_15 + away_16_30 + away_31_45 END) * 100.0 /
        NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 + home_16_30 +
home_31_45 + home_46_60 + home_61_75 + home_76_90 ELSE away_0_15 + away_16_30
+ away_31_45 + away_46_60 + away_61_75 + away_76_90 END), 0),2) AS
proportion_buts_inscrit_1ere_periode,
        -- Proportion of goals scored in the 2nd half
        ROUND(SUM(CASE WHEN is_home = 1 THEN home_46_60 + home_61_75 +
home_76_90 ELSE away_46_60 + away_61_75 + away_76_90 END) * 100.0 /
        NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 + home_16_30 +
home_31_45 + home_46_60 + home_61_75 + home_76_90 ELSE away_0_15 + away_16_30
+ away_31_45 + away_46_60 + away_61_75 + away_76_90 END), 0),2) AS
proportion_buts_inscrit_2nde_periode,
        -- Proportion of goals scored in the first 15th min
        ROUND(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE away_0_15 END)
* 100.0 / NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE away_0_15
END) + SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE away_16_30 END) +
SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE away_31_45 END) +
        SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END), 0),2) AS
proportion_buts_0_15,
        -- Proportion of goals scored between the 16th and 30th minute
        ROUND(SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE away_16_30
END) * 100.0 / NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE
away_0_15 END) + SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE
away_16_30 END) + SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE
away_31_45 END) +
        SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END), 0),2) AS
proportion_buts_16_30,
        -- Proportion of goals scored between the 31st and 45th minute
        ROUND(SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE away_31_45
END) * 100.0 / NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE
away_0_15 END) + SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE
away_16_30 END) + SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE
away_31_45 END) +
        SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END), 0),2) AS
proportion_buts_31_45,
        -- Proportion of goals scored between the 46th and 60th minute
        ROUND(SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60
END) * 100.0 / NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE
away_0_15 END) + SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE
away_16_30 END) + SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE
away_31_45 END) +
        SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END), 0),2) AS
proportion_buts_46_60,
        -- Proportion of goals scored between the 61st and 75th minute
        ROUND(SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75
END) * 100.0 / NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE
away_0_15 END) + SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE
away_16_30 END) + SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE
away_31_45 END) +
        SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END), 0),2) AS
proportion_buts_61_75,
        -- Proportion of goals scored between the 76th and 90th minute
        ROUND(SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90
END) * 100.0 / NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE
away_0_15 END) + SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE
away_16_30 END) + SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE
away_31_45 END) +
        SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END), 0),2) AS
proportion_buts_76_90,
        -- Proportion of goals conceded in the 1st half
        ROUND(SUM(CASE WHEN is_home = 1 THEN away_0_15 + away_16_30 +
away_31_45 ELSE home_0_15 + home_16_30 + home_31_45 END) * 100.0 /
NULLIF(SUM(CASE WHEN is_home = 1 THEN away_0_15 + away_16_30 + away_31_45
+ away_46_60 + away_61_75 + away_76_90 ELSE home_0_15 + home_16_30 +
home_31_45 + home_46_60 + home_61_75 + home_76_90 END), 0),2) AS
proportion_buts_encaissés_1ere_periode,
        -- Proportion of goals conceded in the 2nd half
        ROUND(SUM(CASE WHEN is_home = 1 THEN away_46_60 + away_61_75 +
away_76_90 ELSE home_46_60 + home_61_75 + home_76_90 END) * 100.0 /
NULLIF(SUM(CASE WHEN is_home = 1 THEN away_0_15 + away_16_30 + away_31_45
+ away_46_60 + away_61_75 + away_76_90 ELSE home_0_15 + home_16_30 +
home_31_45 + home_46_60 + home_61_75 + home_76_90 END), 0),2) AS
proportion_buts_encaissés_2nde_periode,
        -- Proportion of goals conceded in the first 15 minutes
        ROUND(SUM(CASE WHEN is_home = 1 THEN away_0_15 ELSE home_0_15 END)
* 100.0 / NULLIF(SUM(CASE WHEN is_home = 1 THEN away_0_15 ELSE home_0_15
END) + SUM(CASE WHEN is_home = 1 THEN away_16_30 ELSE home_16_30 END) +
SUM(CASE WHEN is_home = 1 THEN away_31_45 ELSE home_31_45 END) +
        SUM(CASE WHEN is_home = 1 THEN away_46_60 ELSE home_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN away_61_75 ELSE home_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN away_76_90 ELSE home_76_90 END), 0),2) AS
proportion_buts_encaissés_0_15,
        -- Proportion of goals conceded between the 16th and 30th minute
        ROUND(SUM(CASE WHEN is_home = 1 THEN away_16_30 ELSE home_16_30
END) * 100.0 / NULLIF(SUM(CASE WHEN is_home = 1 THEN away_0_15 ELSE
home_0_15 END) + SUM(CASE WHEN is_home = 1 THEN away_16_30 ELSE
home_16_30 END) + SUM(CASE WHEN is_home = 1 THEN away_31_45 ELSE
home_31_45 END) + SUM(CASE WHEN is_home = 1 THEN away_46_60 ELSE
home_46_60 END) + SUM(CASE WHEN is_home = 1 THEN away_61_75 ELSE
home_61_75 END) + SUM(CASE WHEN is_home = 1 THEN away_76_90 ELSE
home_76_90 END), 0),2) AS proportion_buts_encaissés_16_30,
        -- Proportion of goals conceded between the 31st and 45th minute
        ROUND(SUM(CASE WHEN is_home = 1 THEN away_31_45 ELSE home_31_45
END) * 100.0 / NULLIF(SUM(CASE WHEN is_home = 1 THEN away_0_15 ELSE
home_0_15 END) + SUM(CASE WHEN is_home = 1 THEN away_16_30 ELSE
home_16_30 END) + SUM(CASE WHEN is_home = 1 THEN away_31_45 ELSE
home_31_45 END) +
        SUM(CASE WHEN is_home = 1 THEN away_46_60 ELSE home_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN away_61_75 ELSE home_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN away_76_90 ELSE home_76_90 END), 0),2) AS
proportion_buts_encaissés_31_45,
        -- Proportion of goals conceded between the 46th and 60th minute
        ROUND(SUM(CASE WHEN is_home = 1 THEN away_46_60 ELSE home_46_60
END) * 100.0 / NULLIF(SUM(CASE WHEN is_home = 1 THEN away_0_15 ELSE
home_0_15 END) + SUM(CASE WHEN is_home = 1 THEN away_16_30 ELSE
home_16_30 END) + SUM(CASE WHEN is_home = 1 THEN away_31_45 ELSE
home_31_45 END) +
    SUM(CASE WHEN is_home = 1 THEN away_46_60 ELSE home_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN away_61_75 ELSE home_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN away_76_90 ELSE home_76_90 END), 0),2) AS
proportion_buts_encaissés_46_60,
    -- Proportion of goals conceded between the 61st and 75th minute
    ROUND(SUM(CASE WHEN is_home = 1 THEN away_61_75 ELSE home_61_75
END) * 100.0 / NULLIF(SUM(CASE WHEN is_home = 1 THEN away_0_15 ELSE
home_0_15 END) + SUM(CASE WHEN is_home = 1 THEN away_16_30 ELSE
home_16_30 END) + SUM(CASE WHEN is_home = 1 THEN away_31_45 ELSE
home_31_45 END) +
    SUM(CASE WHEN is_home = 1 THEN away_46_60 ELSE home_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN away_61_75 ELSE home_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN away_76_90 ELSE home_76_90 END), 0),2) AS
proportion_buts_encaissés_61_75,
    -- Proportion of goals conceded between the 76th and 90th minute
    ROUND(SUM(CASE WHEN is_home = 1 THEN away_76_90 ELSE home_76_90
END) * 100.0 / NULLIF(SUM(CASE WHEN is_home = 1 THEN away_0_15 ELSE
home_0_15 END) + SUM(CASE WHEN is_home = 1 THEN away_16_30 ELSE
home_16_30 END) + SUM(CASE WHEN is_home = 1 THEN away_31_45 ELSE
home_31_45
END) +
    SUM(CASE WHEN is_home = 1 THEN away_46_60 ELSE home_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN away_61_75 ELSE home_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN away_76_90 ELSE home_76_90 END), 0),2) AS
proportion_buts_encaissés_76_90,
    -- Number of goals scored in the first half
    NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 + home_16_30 +
home_31_45 ELSE away_0_15 + away_16_30 + away_31_45 END), 0) AS
buts_inscrit_1ere_periode,
    -- Number of goals scored in the 2nd half
    NULLIF(SUM(CASE WHEN is_home = 1 THEN home_46_60 + home_61_75 +
home_76_90 ELSE away_46_60 + away_61_75 + away_76_90 END), 0) AS
buts_inscrit_2nde_periode,
    -- Number of goals scored in the first 15 minutes
    NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE away_0_15 END),
0) AS nbr_buts_0_15,
    -- Number of goals scored between the 16th and 30th minute
    NULLIF(SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE away_16_30
END), 0) AS nbr_buts_16_30,
    -- Number of goals scored between the 31st and 45th minute
    NULLIF(SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE away_31_45
END), 0) AS nbr_buts_31_45,
    -- Number of goals scored between the 46th and 60th minute
    NULLIF(SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60
END), 0) AS nbr_buts_46_60,
    -- Number of goals scored between the 61st and 75th minute
    NULLIF(SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75
END), 0) AS nbr_buts_61_75,
    -- Number of goals scored between the 76th and 90th minute
```

```
        NULLIF(SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90
END), 0) AS nbr_buts_76_90,
        -- Number of goals conceded in the first half
        NULLIF(SUM(CASE WHEN is_home = 1 THEN away_0_15 + away_16_30 +
away_31_45 ELSE home_0_15 + home_16_30 + home_31_45 END), 0) AS
buts_encaissés_1ere_periode,
        -- Number of goals conceded in the 2nd half
        NULLIF(SUM(CASE WHEN is_home = 1 THEN away_46_60 + away_61_75 +
away_76_90 ELSE home_46_60 + home_61_75 + home_76_90 END), 0) AS
buts_encaissés_2nde_periode,
        -- Number of goals conceded in the first 15 minutes
        NULLIF(SUM(CASE WHEN is_home = 1 THEN away_0_15 ELSE home_0_15 END),
0) AS buts_encaissés_0_15,
        -- Number of goals conceded between the 16th and 30th minute
        NULLIF(SUM(CASE WHEN is_home = 1 THEN away_16_30 ELSE home_16_30
END), 0) AS buts_encaissés_16_30,
        -- Number of goals conceded between the 31st and 45th minute
NULLIF(SUM(CASE WHEN is_home = 1 THEN away_31_45 ELSE home_31_45 END), 0)
AS buts_encaissés_31_45,
        -- Number of goals conceded between the 46th and 60th minute
        NULLIF(SUM(CASE WHEN is_home = 1 THEN away_46_60 ELSE home_46_60
END), 0) AS buts_encaissés_46_60,
        -- Number of goals conceded between the 61st and 75th minute
        NULLIF(SUM(CASE WHEN is_home = 1 THEN away_61_75 ELSE home_61_75
END), 0) AS buts_encaissés_61_75,
        -- Number of goals conceded between the 76th and 90th minute
        NULLIF(SUM(CASE WHEN is_home = 1 THEN away_76_90 ELSE home_76_90
END), 0) AS buts_encaissés_76_90
    FROM (
        -- Data from teams playing from home
        SELECT t.team_name, ig.*, 1 AS is_home, s.season_name, result
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN team t ON im.id_home_team = t.id_team
        JOIN info_goal ig USING(id_match)
        UNION ALL
        -- Data from teams playing from away
        SELECT t.team_name, ig.*, 0 AS is_home, s.season_name, result
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN team t ON im.id_away_team = t.id_team
        JOIN info_goal ig USING(id_match)
    ) AS all_matches
    WHERE season_name = season_name_input
    GROUP BY season_name, team_name
    HAVING COUNT(*) >= 5;
$$;

grant execute on function get_distribution_goals_season(text) to anon;
```

- 9/ Search for information on the Home/Away factor for a given team

```
CREATE OR REPLACE FUNCTION get_rank_season(
    season_name_input TEXT
)
RETURNS TABLE (
    type text,
    season_name text,
    team_name text,
    matches numeric(10,2),
    wins numeric(10,2),
    draws numeric(10,2),
    losses numeric(10,2),
    points numeric(10,2),
    avg_points numeric(10,2),
    home_advantage numeric(10,2)

)
LANGUAGE SQL
AS $$
    WITH home_stats AS (
        SELECT
            -- Season name
            s.season_name,
            -- Team name
            t.team_name,
            -- Number of matches played from home
            COUNT(im.id_match) AS home_matches,
            -- Number of wins at home
            COUNT(CASE WHEN ig.result = 1 THEN 1 END) AS home_wins,
            -- Number of draws at home
            COUNT(CASE WHEN ig.result = 0 THEN 1 END) AS home_draws,
            -- Number of defeats at home
            COUNT(CASE WHEN ig.result = 2 THEN 1 END) AS home_losses,
            -- Number of points obtained at home
            (COUNT(CASE WHEN ig.result = 1 THEN 1 END) * 3 + COUNT(CASE WHEN
ig.result = 0 THEN 1 END)) AS home_points,
            -- Number of points averaged at home
            ROUND((COUNT(CASE WHEN ig.result = 1 THEN 1 END) * 3.0 + COUNT(CASE
WHEN ig.result = 0 THEN 1 END)) / COUNT(im.id_match),2) AS avg_home_points
        FROM info_match im
        JOIN info_goal ig ON im.id_match = ig.id_match
        JOIN season s ON im.id_season = s.id_season
        JOIN team t ON im.id_home_team = t.id_team
        WHERE s.season_name = season_name_input
        GROUP BY s.season_name, t.team_name
        HAVING COUNT(im.id_match) >= 5
    ),
    away_stats AS (
        SELECT
            -- Season name
            s.season_name,
            -- Team name
            t.team_name,
            -- Number of matches played from away
            COUNT(im.id_match) AS away_matches,
            -- Number of wins from away
            COUNT(CASE WHEN ig.result = 2 THEN 1 END) AS away_wins,
            -- Number of draws from away
            COUNT(CASE WHEN ig.result = 0 THEN 1 END) AS away_draws,
            -- Number of defeats from away
            COUNT(CASE WHEN ig.result = 1 THEN 1 END) AS away_losses,
            -- Number of points obtained from away
            (COUNT(CASE WHEN ig.result = 2 THEN 1 END) * 3 + COUNT(CASE WHEN
ig.result = 0 THEN 1 END)) AS away_points,
            -    Number of points averaged from away
            ROUND((COUNT(CASE WHEN ig.result = 2 THEN 1 END) * 3.0 + COUNT(CASE
WHEN ig.result = 0 THEN 1 END)) / COUNT(im.id_match),2) AS avg_away_points
        FROM info_match im
        JOIN info_goal ig ON im.id_match = ig.id_match
        JOIN season s ON im.id_season = s.id_season
        JOIN team t ON im.id_away_team = t.id_team
        WHERE s.season_name = season_name_input
        GROUP BY s.season_name, t.team_name
        HAVING COUNT(im.id_match) >= 5
    )
    SELECT
        -- Type of match
        'Home' AS type,
        h.season_name AS season_name,
        h.team_name AS team_name,
        h.home_matches AS matches,
        h.home_wins AS wins,
        h.home_draws AS draws,
        h.home_losses AS losses,
        h.home_points AS points,
        h.avg_home_points AS avg_points,
        -- Home advantage (at home)
        ROUND(((h.home_wins * 3.0 + h.home_draws) / ((h.home_wins * 3.0 + h.home_draws
* 2.0 + h.home_losses * 3.0))) * 100, 2) AS home_advantage
    FROM home_stats h
    LEFT JOIN away_stats a ON h.team_name = a.team_name AND h.season_name =
a.season_name
    UNION ALL
    SELECT
        -- Type of match
        'Away' AS type,
        a.season_name AS season_name,
        a.team_name AS team_name,
        a.away_matches AS matches,
        a.away_wins AS wins,
        a.away_draws AS draws,
        a.away_losses AS losses,
        a.away_points AS points,
        a.avg_away_points AS avg_points,
        -- Home advantage (at home)
        ROUND(((a.away_losses * 3.0 + a.away_draws) /
        ((a.away_wins * 3.0 + a.away_draws * 2.0 + a.away_losses * 3.0))) * 100, 2) AS
home_advantage
    FROM away_stats a
    LEFT JOIN home_stats h ON a.team_name = h.team_name AND a.season_name =
h.season_name
    ORDER BY type, points DESC NULLS LAST;
$$;

grant execute on function get_rank_season(text) to anon;
```

- Information on the analysis between two teams

- 10/ Search for matches between two teams

```
create or replace function get_teams_in_season(season_name_input text)
RETURNS TABLE (
    team_name TEXT
)
LANGUAGE SQL
AS $$
    SELECT t.team_name as team_name
    FROM info_match im
    JOIN season s ON im.id_season = s.id_season
    JOIN team t ON im.id_home_team = t.id_team OR im.id_away_team = t.id_team
    WHERE s.season_name = season_name_input
    GROUP BY team_name
    HAVING COUNT(*) >= 5
$$;

grant execute on function get_teams_in_season(text) to anon;
```

- 11/ Search for matches between two teams

```
create or replace function get_matches_between_teams(selected_team_home_input text,
selected_team_away_input text)
RETURNS TABLE (
    season_name TEXT,
    home_team_name TEXT,
    away_team_name TEXT,
    score_home NUMERIC(10,2),
    score_away NUMERIC(10,2),
    match_date DATE
)
```

```
LANGUAGE SQL
AS $$
   WITH match_details AS (
      SELECT
         -- Season name
         s.season_name AS season_name,
         -- Team name from home
         th.team_name AS home_team_name,
         -- Team name from away
         ta.team_name AS away_team_name,
         -- Score home team
         ig.score_home AS score_home,
         -- Score away team
         ig.score_away AS score_away,
         -- Date
         im.match_date AS match_date
      FROM info_match im
      JOIN season s ON im.id_season = s.id_season
      JOIN team th ON im.id_home_team = th.id_team
      JOIN team ta ON im.id_away_team = ta.id_team
      JOIN info_goal ig ON im.id_match = ig.id_match
      WHERE (th.team_name = selected_team_home_input AND ta.team_name =
selected_team_away_input)
         OR (th.team_name = selected_team_away_input AND ta.team_name =
selected_team_home_input)
   )
   SELECT * FROM match_details
   ORDER BY match_date DESC;
$$;

grant execute on function get_matches_between_teams(text, text) to anon;
```

- 12/ Search for goal information between two teams

```
create or replace function get_avg_goals_stats_between_teams(selected_team_home_input
text, selected_team_away_input text)
RETURNS TABLE (
   avg_goals_selected_home NUMERIC(10,2),
   avg_goals_selected_away NUMERIC(10,2),
   avg_goals_home_at_home NUMERIC(10,2),
   avg_goals_away_at_away NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
   SELECT
      -- Average goals scored by selected_team_home against selected_team_away (home and
away)
      ROUND(AVG(CASE WHEN th.team_name = selected_team_home_input AND
ta.team_name = selected_team_away_input THEN ig.score_home
```

```
          WHEN ta.team_name = selected_team_home_input AND th.team_name =
selected_team_away_input THEN ig.score_away END),2) AS avg_goals_selected_home,
      -- Average goals scored by selected_team_away against selected_team_home (home and
away)
      ROUND(AVG(CASE WHEN ta.team_name = selected_team_away_input AND
th.team_name = selected_team_home_input THEN ig.score_away
             WHEN th.team_name = selected_team_away_input AND ta.team_name =
selected_team_home_input THEN ig.score_home END),2) AS avg_goals_selected_away,
      -- Average goals scored by selected_team_home against selected_team_away at home
      ROUND(AVG(CASE WHEN th.team_name = selected_team_home_input AND
ta.team_name = selected_team_away_input THEN ig.score_home END),2) AS
avg_goals_home_at_home,
      -- Average goals scored by selected_team_away against selected_team_home at home
      ROUND(AVG(CASE WHEN th.team_name = selected_team_home_input AND
ta.team_name = selected_team_away_input THEN ig.score_away END),2) AS
avg_goals_away_at_away
   FROM info_match im
   JOIN team th ON im.id_home_team = th.id_team
   JOIN team ta ON im.id_away_team = ta.id_team
   JOIN info_goal ig ON im.id_match = ig.id_match
   WHERE (th.team_name = selected_team_home_input AND ta.team_name =
selected_team_away_input) OR (ta.team_name = selected_team_home_input AND
th.team_name = selected_team_away_input)
$$;

grant execute on function get_avg_goals_stats_between_teams(text, text) to anon;
```

- 13/ Search for 1st goal information between two teams

```
create or replace function get_1st_goal_stats_between_teams(selected_team_home_input text,
selected_team_away_input text)
RETURNS TABLE (
   team TEXT,
   proportion_1st_goal_for NUMERIC(10,2),
   proportion_no_goal NUMERIC(10,2),
   proportion_1st_goal_against NUMERIC(10,2),
   proportion_1st_goal_win NUMERIC(10,2),
   proportion_1st_goal_draw NUMERIC(10,2),
   proportion_1st_goal_lose NUMERIC(10,2),
   proportion_1st_goal_conceded_win NUMERIC(10,2),
   proportion_1st_goal_conceded_draw NUMERIC(10,2),
   proportion_1st_goal_conceded_lose NUMERIC(10,2)

)
LANGUAGE SQL
AS $$
   SELECT
      -- Team name
      th.team_name AS team,
      -- Proportion of 1st goals scored by the selected team (home + away)
```

```
    ROUND(COUNT(*) FILTER (WHERE (ig.squad_1st_goal = 1 AND im.id_home_team
= th.id_team)
        OR (ig.squad_1st_goal = 2 AND im.id_away_team = th.id_team)) * 100 /
COALESCE(NULLIF(
    COUNT(*) FILTER (WHERE im.id_home_team = th.id_team) +
    COUNT(*) FILTER (WHERE im.id_away_team = th.id_team), 0), 1),2) AS
proportion_1st_goal_for,
    -- Proportion of no goals scored
    ROUND(COUNT(*) FILTER (WHERE ig.squad_1st_goal = 0) * 100 /
    COALESCE(NULLIF(COUNT(*) FILTER (WHERE im.id_home_team = th.id_team) +
    COUNT(*) FILTER (WHERE im.id_away_team = th.id_team), 0), 1),2) AS
proportion_no_goal,
    -- Proportion of 1st goals conceded by the selected team
    ROUND(COUNT(*) FILTER (WHERE (ig.squad_1st_goal = 1 AND im.id_away_team
= th.id_team)
    OR (ig.squad_1st_goal = 2 AND im.id_home_team = th.id_team)) * 100 /
    COALESCE(NULLIF(COUNT(*) FILTER (WHERE im.id_home_team = th.id_team) +
    COUNT(*) FILTER (WHERE im.id_away_team = th.id_team), 0), 1),2) AS
proportion_1st_goal_against,
    -- Proportion of 1st goals scored leading to victory
    ROUND(COUNT(*) FILTER (WHERE (ig.squad_1st_goal = 1 AND ig.result = 1 AND
im.id_home_team = th.id_team)
    OR (ig.squad_1st_goal = 2 AND ig.result = 2 AND im.id_away_team = th.id_team)) *
100 /
    COALESCE(NULLIF(COUNT(*) FILTER (WHERE (ig.squad_1st_goal = 1 AND
im.id_home_team = th.id_team)
    OR (ig.squad_1st_goal = 2 AND im.id_away_team = th.id_team)), 0), 1),2) AS
proportion_1st_goal_win,
    -- Proportion of 1st goals scored leading to a draw
    ROUND(COUNT(*) FILTER (WHERE (ig.squad_1st_goal = 1 AND ig.result = 0 AND
im.id_home_team = th.id_team)
    OR (ig.squad_1st_goal = 2 AND ig.result = 0 AND im.id_away_team = th.id_team)) *
100 /
    COALESCE(NULLIF(COUNT(*) FILTER (WHERE (ig.squad_1st_goal = 1 AND
im.id_home_team = th.id_team)
    OR (ig.squad_1st_goal = 2 AND im.id_away_team = th.id_team)), 0), 1),2) AS
proportion_1st_goal_draw,
    -- Proportion of 1st goals scored leading to a defeat
    ROUND(COUNT(*) FILTER (WHERE (ig.squad_1st_goal = 1 AND ig.result = 2 AND
im.id_home_team = th.id_team)
    OR (ig.squad_1st_goal = 2 AND ig.result = 1 AND im.id_away_team = th.id_team)) *
100 /
    COALESCE(NULLIF(COUNT(*) FILTER (WHERE (ig.squad_1st_goal = 1 AND
im.id_home_team = th.id_team)
    OR (ig.squad_1st_goal = 2 AND im.id_away_team = th.id_team)), 0), 1),2) AS
proportion_1st_goal_lose,
    -- Proportion of 1st goals conceded leading to a win
    ROUND(COUNT(*) FILTER (WHERE (ig.squad_1st_goal = 1 AND ig.result = 2 AND
im.id_away_team = th.id_team)
```

```
      OR (ig.squad_1st_goal = 2 AND ig.result = 1 AND im.id_home_team = th.id_team)) *
100 /
      COALESCE(NULLIF(COUNT(*) FILTER (WHERE (ig.squad_1st_goal = 1 AND
im.id_away_team = th.id_team)
      OR (ig.squad_1st_goal = 2 AND im.id_home_team = th.id_team)), 0), 1),2) AS
proportion_1st_goal_conceded_win,
      -- Proportion of 1st goals conceded leading to a draw
      ROUND(COUNT(*) FILTER (WHERE (ig.squad_1st_goal = 1 AND ig.result = 0 AND
im.id_away_team = th.id_team)
      OR (ig.squad_1st_goal = 2 AND ig.result = 0 AND im.id_home_team = th.id_team)) *
100 /
      COALESCE(NULLIF(COUNT(*) FILTER (WHERE (ig.squad_1st_goal = 1 AND
im.id_away_team = th.id_team)
      OR (ig.squad_1st_goal = 2 AND im.id_home_team = th.id_team)), 0), 1),2) AS
proportion_1st_goal_conceded_draw,
      -- Proportion of 1st goals conceded leading to a defeat
      ROUND(COUNT(*) FILTER (WHERE (ig.squad_1st_goal = 1 AND ig.result = 1 AND
im.id_away_team = th.id_team)
      OR (ig.squad_1st_goal = 2 AND ig.result = 2 AND im.id_home_team = th.id_team)) *
100 /
      COALESCE(NULLIF(COUNT(*) FILTER (WHERE (ig.squad_1st_goal = 1 AND
im.id_away_team = th.id_team)
      OR (ig.squad_1st_goal = 2 AND im.id_home_team = th.id_team)), 0), 1),2) AS
proportion_1st_goal_conceded_lose
   FROM info_match im
   JOIN team th ON im.id_home_team = th.id_team
   JOIN team ta ON im.id_away_team = ta.id_team
   JOIN info_goal ig ON im.id_match = ig.id_match
   WHERE (th.team_name = selected_team_home_input AND ta.team_name =
selected_team_away_input)
   OR (th.team_name = selected_team_home_input AND ta.team_name =
selected_team_away_input)
   GROUP BY th.team_name;
$$;

grant execute on function get_1st_goal_stats_between_teams(text, text) to anon;
```

- 14/ Search for information on the distribution of goals between two teams

```
create or replace function get_distrib_goal_between_teams(selected_team_home_input text,
selected_team_away_input text)
RETURNS TABLE (
   team TEXT,
   proportion_0_45 NUMERIC(10,2),
   proportion_46_90 NUMERIC(10,2),
   proportion_0_15 NUMERIC(10,2),
   proportion_16_30 NUMERIC(10,2),
   proportion_31_45 NUMERIC(10,2),
   proportion_46_60 NUMERIC(10,2),
   proportion_61_75 NUMERIC(10,2),
    proportion_76_90 NUMERIC(10,2)

)
LANGUAGE SQL
AS $$
    WITH data_team AS (
        SELECT
            -- Team name
            th.team_name AS team,
             -- Type of match (Home/Away)
            'home' AS match_type,
            -- Number of goals scored in the first 15 min
            SUM(ig.home_0_15) AS goals_0_15,
            -- Number of goals scored between 16th and 30th minute
            SUM(ig.home_16_30) AS goals_16_30,
            -- Number of goals scored between 31th and 45th minute
            SUM(ig.home_31_45) AS goals_31_45,
            -- Number of goals scored between 46th and 60th minute
            SUM(ig.home_46_60) AS goals_46_60,
            -- Number of goals scored between 61st and 75th minute
            SUM(ig.home_61_75) AS goals_61_75,
            -- Number of goals scored between 76th and 90th minute
            SUM(ig.home_76_90) AS goals_76_90,
            -- Number of goals scored from home team in the first half
            SUM(ig.home_0_15 + ig.home_16_30 + ig.home_31_45) AS first_period_goals,
            -- Number of goals scored from home team in the 2nd half
            SUM(ig.home_46_60 + ig.home_61_75 + ig.home_76_90) AS second_period_goals,
            -- Total of goals scored from home team
            SUM(ig.home_0_15 + ig.home_16_30 + ig.home_31_45 + ig.home_46_60 +
ig.home_61_75 + ig.home_76_90) AS total_goals
        FROM info_match im
        JOIN team th ON im.id_home_team = th.id_team
        JOIN team ta ON im.id_away_team = ta.id_team
        JOIN info_goal ig ON im.id_match = ig.id_match
        WHERE (th.team_name = selected_team_home_input AND ta.team_name =
selected_team_away_input) OR (th.team_name = selected_team_away_input AND
ta.team_name = selected_team_home_input)
        GROUP BY th.team_name
        UNION ALL
        SELECT
            -- Team name
            ta.team_name AS team,
            -- Type of match (Home/Away)
            'away' AS match_type,
            -- Number of goals scored from away team in the first 15 minutes
            SUM(ig.away_0_15) AS goals_0_15,
            -- Number of goals scored from away team between 16th and 31th minute
            SUM(ig.away_16_30) AS goals_16_30,
            -- Number of goals scored from away team between 31st  and 45th minute
            SUM(ig.away_31_45) AS goals_31_45,
            -- Number of goals scored from away team between 46th and 60th minute
            SUM(ig.away_46_60) AS goals_46_60,
            -- Number of goals scored from away team between 61th and 75th minute
SUM(ig.away_61_75) AS goals_61_75,
            -- Number of goals scored from away team between 76th and 90th minute
            SUM(ig.away_76_90) AS goals_76_90,
            -- Number of goals scored from away team in the first half
            SUM(ig.away_0_15 + ig.away_16_30 + ig.away_31_45) AS first_period_goals,
            -- Number of goals scored from away team in the second half
            SUM(ig.away_46_60 + ig.away_61_75 + ig.away_76_90) AS second_period_goals,
            -    Number of goals scored from away team
            SUM(ig.away_0_15 + ig.away_16_30 + ig.away_31_45 + ig.away_46_60 +
ig.away_61_75 + ig.away_76_90) AS total_goals
        FROM info_match im
        JOIN team th ON im.id_home_team = th.id_team
        JOIN team ta ON im.id_away_team = ta.id_team
        JOIN info_goal ig ON im.id_match = ig.id_match
        WHERE (th.team_name = selected_team_home_input AND ta.team_name =
selected_team_away_input) OR (th.team_name = selected_team_away_input AND
ta.team_name = selected_team_home_input)
        GROUP BY ta.team_name
      )
      SELECT
        -- Team name
        team,
        -- Number of goals scored in the first half
        ROUND(SUM(first_period_goals) * 100.0 / NULLIF(SUM(total_goals), 0), 2) AS
proportion_0_45,
        -- Number of goals scored in the 2nd half
        ROUND(SUM(second_period_goals) * 100.0 / NULLIF(SUM(total_goals), 0), 2) AS
proportion_46_90,
        -- Number of goals scored in the first 15 minutes
        ROUND(SUM(goals_0_15) * 100.0 / NULLIF(SUM(total_goals), 0), 2) AS
proportion_0_15,
        -- Number of goals scored between the 16th  and 30th minute
        ROUND(SUM(goals_16_30) * 100.0 / NULLIF(SUM(total_goals), 0), 2) AS
proportion_16_30,
        -- Number of goals scored between the 31st  and 46th minute
        ROUND(SUM(goals_31_45) * 100.0 / NULLIF(SUM(total_goals), 0), 2) AS
proportion_31_45,
        -- Number of goals scored between the 46th  and 60th minute
        ROUND(SUM(goals_46_60) * 100.0 / NULLIF(SUM(total_goals), 0), 2) AS
proportion_46_60,
        -- Number of goals scored between the 61st  and 75th minute
        ROUND(SUM(goals_61_75) * 100.0 / NULLIF(SUM(total_goals), 0), 2) AS
proportion_61_75,
        -- Number of goals scored between the 75th   and 90th minute
        ROUND(SUM(goals_76_90) * 100.0 / NULLIF(SUM(total_goals), 0), 2) AS
proportion_76_90
      FROM data_team
```

```
   GROUP BY team;
$$;

grant execute on function get_distrib_goal_between_teams(text, text) to anon;
```

- 15/ Research into the influence of the home/away factor between the two teams

```
create or replace function get_home_away_selected_teams(selected_team_home_input text,
selected_team_away_input text)
RETURNS TABLE (
   team_name TEXT,
   home_win NUMERIC(10,2),
   home_draws NUMERIC(10,2),
   home_losses NUMERIC(10,2),
   home_advantage NUMERIC(10,2),
   total_wins NUMERIC(10,2),
   total_draws NUMERIC(10,2),
   total_losses NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
   WITH home_stats AS (
      SELECT
         -- Team name
         th.team_name,
         -- Number of matches played at home
         COUNT(im.id_match) AS home_matches,
         -- Number of wins at home
         COUNT(CASE WHEN ig.result = 1 THEN 1 END) AS home_wins,
         -- Number of draws at home
         COUNT(CASE WHEN ig.result = 0 THEN 1 END) AS home_draws,
         -- Number of defeat at home
         COUNT(CASE WHEN ig.result = 2 THEN 1 END) AS home_losses
      FROM info_match im
      JOIN team th ON im.id_home_team = th.id_team
      JOIN team ta ON im.id_away_team = ta.id_team
      JOIN info_goal ig ON im.id_match = ig.id_match
      WHERE (th.team_name = selected_team_home_input AND ta.team_name =
selected_team_away_input)
      OR (th.team_name = selected_team_away_input AND ta.team_name =
selected_team_home_input)
      GROUP BY th.team_name
   ),
   away_stats AS (
      SELECT
         -- Team name
         ta.team_name,
         -- Number of matchs at away
         COUNT(im.id_match) AS away_matches,
         -- Number of wins at away
      COUNT(CASE WHEN ig.result = 2 THEN 1 END) AS away_wins,
      -- Number of draws at away
      COUNT(CASE WHEN ig.result = 0 THEN 1 END) AS away_draws,
      -- Number of defeats at way
      COUNT(CASE WHEN ig.result = 1 THEN 1 END) AS away_losses
    FROM info_match im
    JOIN team th ON im.id_home_team = th.id_team
    JOIN team ta ON im.id_away_team = ta.id_team
    JOIN info_goal ig ON im.id_match = ig.id_match
    WHERE (th.team_name = selected_team_home_input AND ta.team_name =
selected_team_away_input)
    OR (th.team_name = selected_team_away_input AND ta.team_name =
selected_team_home_input)
    GROUP BY ta.team_name
  )
  -- Selection of home match statistics and totals (home + away)
  SELECT
    -- Team name
    h.team_name AS team_name,
    -- Number of wins at home
    h.home_wins AS home_wins,
    -- Number of draws at home
    h.home_draws AS home_draws,
    -- Number of defeat at home
    h.home_losses AS home_losses,
    -- Calculation of home advantage
    ROUND(((h.home_wins * 3.0 + h.home_draws) / (h.home_wins * 3.0 + h.home_draws
* 2.0 + h.home_losses * 3.0)) * 100, 2) AS home_advantage,
    -- Sum of wins, draws, and defeats
    COALESCE(h.home_wins, 0) + COALESCE(a.away_wins, 0) AS total_wins,
    COALESCE(h.home_draws, 0) + COALESCE(a.away_draws, 0) AS total_draws,
    COALESCE(h.home_losses, 0) + COALESCE(a.away_losses, 0) AS total_losses
  FROM home_stats h
  LEFT JOIN away_stats a ON h.team_name = a.team_name
  ORDER BY h.team_name;
$$;

grant execute on function get_home_away_selected_teams(text, text) to anon;
```

- Season analysis information

- 16 / Search for available competitions

```
create or replace function get_competitions()
returns setof text
language sql
as $$
  SELECT DISTINCT competition.competition_name
  FROM competition
  JOIN season ON competition.id_competition = season.id_competition;
$$;
```

```
grant execute on function get_competitions() to anon;
```

- 17/ Search for seasons available for a given team

```
create or replace function get_seasons_by_competition(competition_name_input text)
returns setof text
language sql
as $$
    SELECT DISTINCT season.season_name
    FROM season
    JOIN competition ON season.id_competition = competition.id_competition
    WHERE competition.competition_name = competition_name_input
   ORDER BY season.season_name DESC;

$$;

grant execute on function get_seasons_by_competition(text) to anon;
```

- 18/ Search for information on goals scored over a season, taking into account the competition

```
create or replace function get_avg_goals_stats_by_competition()
returns table (
    competition_name text,
    season_name text,
    avg_goals_per_match numeric(10,2),
    avg_home_goals numeric(10,2),
    avg_away_goals numeric(10,2)
)
language sql
as $$
    SELECT
        -- Competition name
        c.competition_name,
        -- Season name
        s.season_name,
        -- Number of goals per match
        (SUM(ig.score_home) + SUM(ig.score_away)) * 1.0 / COUNT(im.id_match) AS avg_goals_per_match,
        -- Number of goals from home teams
        AVG(ig.score_home) AS avg_home_goals,
        -- Number of goals from away teams
        AVG(ig.score_away) AS avg_away_goals
    FROM info_match im
    JOIN season s USING(id_season)
    JOIN competition c USING(id_competition)
    JOIN info_goal ig USING(id_match)
    GROUP BY c.competition_name, s.season_name;
$$;
```

```
grant execute on function get_avg_goals_stats_by_competition() to anon;
```

- 19/ Search for the frequency of scores for a given team

```
CREATE OR REPLACE FUNCTION get_frequent_score_by_season(season_name_input TEXT)
RETURNS TABLE (
    score_home INT,
    score_away INT,
    percentage NUMERIC(5,2)
)
LANGUAGE SQL
AS $$
    WITH score_counts AS (
        SELECT
            -- Competition name
            c.competition_name,
            -- Season name
            s.season_name,
            -- Home score
            ig.score_home,
            -- Away score
            ig.score_away,
            -- Frequenct of score
            COUNT(*) AS frequency
        FROM info_goal ig
        JOIN info_match im USING(id_match)
        JOIN season s USING(id_season)
        JOIN competition c USING(id_competition)
        GROUP BY c.competition_name, s.season_name, ig.score_home, ig.score_away
    ), total_matches AS (
        SELECT
            -- Competition name
            c.competition_name,
            -- Season name
            s.season_name,
            -- Number of matches
            COUNT(id_match) AS total_matches
        FROM info_match
        JOIN season s USING(id_season)
        JOIN competition c USING(id_competition)
        GROUP BY c.competition_name, s.season_name
    )
    SELECT
        -- Home score
        sc.score_home,
        -- Away score
        sc.score_away,
        -- Percentage score
        ROUND((sc.frequency * 100.0) / NULLIF(tm.total_matches, 0), 2) AS percentage
    FROM score_counts sc
    JOIN total_matches tm ON sc.competition_name = tm.competition_name AND
sc.season_name = tm.season_name
    WHERE sc.season_name = season_name_input
    ORDER BY percentage DESC;
$$;

grant execute on function get_frequent_score_by_season(text) to anon;
```

- 20/ Search for the best teams in terms of goals scored

```
create or replace function get_top5_goals_scored(competition_name_input text)
returns table (
    team_name text,
    season_name text,
    total_goals_scored numeric(10,2),
    avg_goals_scored numeric(10,2),
    goals_scored_home numeric(10,2),
    avg_goals_scored_home numeric(10,2),
    goals_scored_away numeric(10,2),
    avg_goals_scored_away numeric(10,2)
)
language sql
as $$
    WITH team_avg_goals AS (
        SELECT
            -- Team name
            team_name,
            -- Season name
            season_name,
            -- Number of goals scored
            SUM(CASE WHEN is_home = 1 THEN score_home ELSE 0 END) + SUM(CASE
WHEN is_home = 0 THEN score_away ELSE 0 END) AS total_goals_scored,
            -- Average goals scored
            ROUND((SUM(CASE WHEN is_home = 1 THEN score_home ELSE 0 END) +
SUM(CASE WHEN is_home = 0 THEN score_away ELSE 0 END)) * 1.0 /
            NULLIF(COUNT(CASE WHEN is_home = 1 THEN 1 ELSE NULL END) +
COUNT(CASE WHEN is_home = 0 THEN 1 ELSE NULL END), 0),2) AS
avg_goals_scored,
            -- Number of goals scored at home
            SUM(CASE WHEN is_home = 1 THEN score_home ELSE 0 END) AS
goals_scored_home,
            -- Average of goals scored at home
            ROUND(SUM(CASE WHEN is_home = 1 THEN score_home ELSE 0 END) * 1.0 /
NULLIF(COUNT(CASE WHEN is_home = 1 THEN 1 ELSE NULL END), 0),2) AS
avg_goals_scored_home,
            -- Number of goals scored at away
            SUM(CASE WHEN is_home = 0 THEN score_away ELSE 0 END) AS
goals_scored_away,
            -- Average of goals scored at away
```

```
      ROUND(SUM(CASE WHEN is_home = 0 THEN score_away ELSE 0 END) * 1.0 /
NULLIF(COUNT(CASE WHEN is_home = 0 THEN 1 ELSE NULL END), 0),2) AS
avg_goals_scored_away
    FROM (
      SELECT t.team_name, s.season_name, ig.score_home, ig.score_away, 1 AS is_home
      FROM info_match im
      JOIN season s USING(id_season)
      JOIN competition c USING(id_competition)
      JOIN team t ON im.id_home_team = t.id_team
      JOIN info_goal ig USING(id_match)
      WHERE c.competition_name = competition_name_input
      UNION ALL
      SELECT t.team_name, s.season_name, ig.score_home, ig.score_away, 0 AS is_home
      FROM info_match im
      JOIN season s USING(id_season)
      JOIN competition c USING(id_competition)
      JOIN team t ON im.id_away_team = t.id_team
      JOIN info_goal ig USING(id_match)
      WHERE c.competition_name = competition_name_input
    ) AS all_matches
    GROUP BY team_name, season_name
    HAVING COUNT(*) >= 5
  )
  SELECT * FROM team_avg_goals
  ORDER BY avg_goals_scored DESC
  LIMIT 5;
$$;

grant execute on function get_top5_goals_scored(text) to anon;
```

- 21/ Search for teams that have conceded the fewest goals

```
CREATE OR REPLACE FUNCTION get_top5_goals_conceded(competition_name_input
TEXT)
RETURNS TABLE (
  team_name TEXT,
  season_name text,
  total_goals_conceded NUMERIC(10,2),
  avg_goals_conceded NUMERIC(10,2),
  goals_conceded_home NUMERIC(10,2),
  avg_goals_conceded_home NUMERIC(10,2),
  goals_conceded_away NUMERIC(10,2),
  avg_goals_conceded_away NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
  WITH team_avg_goals AS (
    SELECT
      -- Team name
      team_name,
      -- Season name
```

```
        season_name,
        -- Number of goals conceded
        SUM(CASE WHEN is_home = 1 THEN score_away ELSE 0 END) + SUM(CASE
WHEN is_home = 0 THEN score_home ELSE 0 END) AS total_goals_conceded,
        -- Average of goals conceded
        ROUND((SUM(CASE WHEN is_home = 1 THEN score_away ELSE 0 END) +
SUM(CASE WHEN is_home = 0 THEN score_home ELSE 0 END)) * 1.0 /
        NULLIF(COUNT(CASE WHEN is_home = 1 THEN 1 END) + COUNT(CASE
WHEN is_home = 0 THEN 1 END), 0),2) AS avg_goals_conceded,
        -- Number of goals conceded at home
        SUM(CASE WHEN is_home = 1 THEN score_away ELSE 0 END) AS
goals_conceded_home,
        -- Average of goals conceded at home
        ROUND(SUM(CASE WHEN is_home = 1 THEN score_away ELSE 0 END) * 1.0 /
NULLIF(COUNT(CASE WHEN is_home = 1 THEN 1 END), 0),2) AS
avg_goals_conceded_home,
        -- Average of goals conceded at away
        SUM(CASE WHEN is_home = 0 THEN score_home ELSE 0 END) AS
goals_conceded_away,
        -    Average of goals conceded at away
        ROUND(SUM(CASE WHEN is_home = 0 THEN score_home ELSE 0 END) * 1.0 /
NULLIF(COUNT(CASE WHEN is_home = 0 THEN 1 END), 0),2) AS
avg_goals_conceded_away
      FROM (
        SELECT t.team_name, s.season_name, ig.score_home, ig.score_away, 1 AS is_home
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN competition c USING(id_competition)
        JOIN team t ON im.id_home_team = t.id_team
        JOIN info_goal ig USING(id_match)
        WHERE c.competition_name = competition_name_input
        UNION ALL
        SELECT t.team_name, s.season_name, ig.score_home, ig.score_away, 0 AS is_home
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN competition c USING(id_competition)
        JOIN team t ON im.id_away_team = t.id_team
        JOIN info_goal ig USING(id_match)
        WHERE c.competition_name = competition_name_input
      ) AS all_matches
      GROUP BY team_name, season_name
      HAVING COUNT(*) >= 5
   )
   SELECT * FROM team_avg_goals
   ORDER BY avg_goals_conceded ASC
   LIMIT 5;
$$;

grant execute on function get_top5_goals_conceded(text) to anon;
```

- 22/ Search for information on 1st goal scored over a season

```
CREATE OR REPLACE FUNCTION get_first_goal_stats(season_name_input TEXT)
RETURNS TABLE (
    season_name text,
    proportion_no_goal NUMERIC(10,2),
    proportion_1st_goal_home NUMERIC(10,2),
    proportion_1st_goal_away NUMERIC(10,2),
    first_goal_win NUMERIC(10,2),
    first_goal_draw NUMERIC(10,2),
    first_goal_lose NUMERIC(10,2),
    first_goal_home_win NUMERIC(10,2),
    first_goal_home_draw NUMERIC(10,2),
    first_goal_home_lose NUMERIC(10,2),
    first_goal_away_win NUMERIC(10,2),
    first_goal_away_draw NUMERIC(10,2),
    first_goal_away_lose NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
    SELECT
        -- Season name
        season_name,
        -- Proportion of games without a goal
        ROUND(COUNT(CASE WHEN squad_1st_goal = 0 THEN 1 END) * 100 /
COUNT(squad_1st_goal),2) AS proportion_no_goal,
        -- Proportion of games in which home team scores 1st goal
        ROUND(COUNT(CASE WHEN squad_1st_goal = 1 THEN 1 END) * 100 /
COUNT(squad_1st_goal),2) AS proportion_1st_goal_home,
        -- Proportion of games in which away team scores 1st goal
        ROUND(COUNT(CASE WHEN squad_1st_goal = 2 THEN 1 END) * 100 /
COUNT(squad_1st_goal),2) AS proportion_1st_goal_away,
        -- Proportion of matches won by the team that scored the 1st goal
        ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND result = 1 THEN 1 END) +
COUNT(CASE WHEN squad_1st_goal = 2 AND result = 2 THEN 1 END)) * 100 /
NULLIF(COUNT(CASE WHEN squad_1st_goal = 1 THEN 1 END) + COUNT(CASE
WHEN squad_1st_goal = 2 THEN 1 END), 0),2) AS first_goal_win,
        -- Proportion of games in which the team that scored the 1st goal draws
        ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND result = 0 THEN 1 END) +
COUNT(CASE WHEN squad_1st_goal = 2 AND result = 0 THEN 1 END)) * 100 /
NULLIF(COUNT(CASE WHEN squad_1st_goal = 1 THEN 1 END) + COUNT(CASE
WHEN squad_1st_goal = 2 THEN 1 END), 0),2) AS first_goal_draw,
        -- Proportion of games in which the team that scored the 1st goal defeats
        ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND result = 2 THEN 1 END) +
COUNT(CASE WHEN squad_1st_goal = 2 AND result = 1 THEN 1 END)) * 100 /
NULLIF(COUNT(CASE WHEN squad_1st_goal = 1 THEN 1 END) + COUNT(CASE
WHEN squad_1st_goal = 2 THEN 1 END), 0),2) AS first_goal_lose,
        -- Proportion of matches where the home team scores the 1st goal and wins the match
        ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND result = 1 THEN 1 END)) *
100 / NULLIF(COUNT(CASE WHEN squad_1st_goal = 1 THEN 1 END), 0),2) AS
first_goal_home_win,
        -- Proportion of matches where the home team scores the 1st goal and draws
        ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND result = 0 THEN 1 END)) *
100 / NULLIF(COUNT(CASE WHEN squad_1st_goal = 1 THEN 1 END), 0),2) AS
first_goal_home_draw,
        -- Proportion of matches where the home team scores the 1st goal and defeats the match
        ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND result = 2 THEN 1 END)) *
100 / NULLIF(COUNT(CASE WHEN squad_1st_goal = 1 THEN 1 END), 0),2) AS
first_goal_home_lose,
        -- Proportion of matches where the away team scores the 1st goal and wins the match
        ROUND((COUNT(CASE WHEN squad_1st_goal = 2 AND result = 2 THEN 1 END)) *
100 / NULLIF(COUNT(CASE WHEN squad_1st_goal = 2 THEN 1 END), 0),2) AS
first_goal_away_win,
        -- Proportion of matches where the away team scores the 1st goal and draws
        ROUND((COUNT(CASE WHEN squad_1st_goal = 2 AND result = 0 THEN 1 END)) *
100 / NULLIF(COUNT(CASE WHEN squad_1st_goal = 2 THEN 1 END), 0),2) AS
first_goal_away_draw,
        -- Proportion of matches where the away team scores the 1st goal and defeats the match
        ROUND((COUNT(CASE WHEN squad_1st_goal = 2 AND result = 1 THEN 1 END)) *
100 / NULLIF(COUNT(CASE WHEN squad_1st_goal = 2 THEN 1 END), 0),2) AS
first_goal_away_lose
    FROM info_match
    JOIN season USING(id_season)
    JOIN info_goal USING(id_match)
    WHERE season_name = season_name_input
    GROUP BY season_name;
$$;

grant execute on function get_first_goal_stats(text) to anon;
```

- 23/ Search for information on the best teams in terms of 1st goal scored

```
CREATE OR REPLACE FUNCTION get_top_teams_first_goal(competition_name_input
TEXT)
RETURNS TABLE (
    season_name text,
    team_name text,
    proportion_1st_goal_for NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
    SELECT
        -- Season name
        season_name,
        -- Team name
        team_name,
        -- Proportion of matches where the team scores the 1st goal
        ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND is_home = 1 THEN 1 END)
+ COUNT(CASE WHEN squad_1st_goal = 2 AND is_home = 0 THEN 1 END)) * 100 /
COALESCE(NULLIF(COUNT(squad_1st_goal),
        0),1),2) AS proportion_1st_goal_for
    FROM (
        -- Data from home teams
        SELECT t.team_name, ig.squad_1st_goal, 1 AS is_home, s.season_name, ig.result,
c.competition_name
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN team t ON im.id_home_team = t.id_team
        JOIN info_goal ig USING(id_match)
        JOIN competition c USING(id_competition)
        UNION ALL
        -- Data from away teams
        SELECT t.team_name, ig.squad_1st_goal, 0 AS is_home, s.season_name, ig.result,
c.competition_name
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN team t ON im.id_away_team = t.id_team
        JOIN info_goal ig USING(id_match)
        JOIN competition c USING(id_competition)
    ) AS all_matches
    WHERE competition_name = competition_name_input
    GROUP BY season_name, team_name
    HAVING COUNT(*) >= 5
    ORDER BY proportion_1st_goal_for DESC
    LIMIT 5;
$$;

grant execute on function get_top_teams_first_goal(text) to anon;
```

-- 24/ Search for information on the best teams in terms of the 1st winning goal scored

```
CREATE OR REPLACE FUNCTION
get_top_teams_first_goal_win(competition_name_input TEXT)
RETURNS TABLE (
    season_name text,
    team_name text,
    first_goal_win NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
    SELECT
        -- Season name
        season_name,
        -- Team name
        team_name,
        -- Proportion of matches where the team scores the 1st goal and wins
        ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND result = 1 AND is_home = 1
THEN 1 END) + COUNT(CASE WHEN squad_1st_goal = 2 AND result = 2 AND is_home
= 0 THEN 1 END)) * 100 /
        COALESCE(NULLIF((COUNT(CASE WHEN squad_1st_goal = 1 AND is_home = 1
THEN 1 END) + COUNT(CASE WHEN squad_1st_goal = 2 AND is_home = 0 THEN 1
END)), 0), 1),2) AS first_goal_win
```

```
    FROM (
        -- Data from home teams
        SELECT t.team_name, ig.squad_1st_goal, 1 AS is_home, s.season_name, ig.result,
c.competition_name
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN team t ON im.id_home_team = t.id_team
        JOIN info_goal ig USING(id_match)
        JOIN competition c USING(id_competition)
        UNION ALL
        -- Data from away teams
        SELECT t.team_name, ig.squad_1st_goal, 0 AS is_home, s.season_name, ig.result,
c.competition_name
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN team t ON im.id_away_team = t.id_team
        JOIN info_goal ig USING(id_match)
        JOIN competition c USING(id_competition)
    ) AS all_matches
    WHERE competition_name = competition_name_input
    GROUP BY season_name, team_name
    HAVING COUNT(*) >= 5
    ORDER BY first_goal_win DESC
    LIMIT 5;
$$;

grant execute on function get_top_teams_first_goal_win(text) to anon;
```

- 25/ Search for information on the best teams in terms of 1st goals conceded but still won

```
CREATE OR REPLACE FUNCTION
get_top_teams_first_goal_conceded_win(competition_name_input TEXT)
RETURNS TABLE (
    season_name text,
    team_name text,
    first_goal_conceded_win NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
    SELECT
        -- Season name
        season_name,
        -- Team name
        team_name,
        -- Proportion of matches where the team concedes the 1st goal and wins
        ROUND((COUNT(CASE WHEN squad_1st_goal = 2 AND result = 1 AND is_home = 1
THEN 1 END) + COUNT(CASE WHEN squad_1st_goal = 1 AND result = 2 AND is_home
= 0 THEN 1 END)) * 100 / COALESCE(NULLIF((COUNT(CASE WHEN squad_1st_goal =
2 AND is_home = 1 THEN 1 END) + COUNT(CASE WHEN squad_1st_goal = 1 AND
is_home = 0 THEN 1 END)), 0), 1),2) AS first_goal_conceded_win
    FROM (
        -- Data from home teams
        SELECT t.team_name, ig.squad_1st_goal, 1 AS is_home, s.season_name, ig.result,
c.competition_name
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN team t ON im.id_home_team = t.id_team
        JOIN info_goal ig USING(id_match)
        JOIN competition c USING(id_competition)
        UNION ALL
        -- Data from away teams
        SELECT t.team_name, ig.squad_1st_goal, 0 AS is_home, s.season_name, ig.result,
c.competition_name
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN team t ON im.id_away_team = t.id_team
        JOIN info_goal ig USING(id_match)
        JOIN competition c USING(id_competition)
    ) AS all_matches
    WHERE competition_name = competition_name_input
    GROUP BY season_name, team_name
    HAVING COUNT(*) >= 5
    ORDER BY first_goal_conceded_win DESC
    LIMIT 5;
$$;

grant execute on function get_top_teams_first_goal_conceded_win(text) to anon;
```

- 26/ Search for information on the distribution of goals over a season

```
CREATE OR REPLACE FUNCTION get_distribution_goals(season_name_input TEXT)
RETURNS TABLE (
  season_name text,
  proportion_buts_1ere_periode NUMERIC(10,2),
  proportion_buts_2nde_periode NUMERIC(10,2),
  proportion_buts_0_15 NUMERIC(10,2),
  proportion_buts_16_30 NUMERIC(10,2),
  proportion_buts_31_45 NUMERIC(10,2),
  proportion_buts_46_60 NUMERIC(10,2),
  proportion_buts_61_75 NUMERIC(10,2),
  proportion_buts_76_90 NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
  SELECT
    -- Season name
    season_name,
    -- Proportion of goals scored in the first half
    ROUND((SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45)) * 100.0 /
NULLIF(SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45) + SUM(home_46_60) +
SUM(away_46_60) + SUM(home_61_75) + SUM(away_61_75) + SUM(home_76_90) +
SUM(away_76_90), 0),2) AS proportion_buts_1ere_periode,
    -- Proportion of goals scored in the 2nd half
    ROUND((SUM(home_46_60) + SUM(away_46_60) + SUM(home_61_75) +
SUM(away_61_75) + SUM(home_76_90) + SUM(away_76_90)) * 100.0 /
NULLIF(SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45) + SUM(home_46_60) +
SUM(away_46_60) + SUM(home_61_75) + SUM(away_61_75) + SUM(home_76_90) +
SUM(away_76_90), 0),2) AS proportion_buts_2nde_periode,
    -- Proportion of goals scored in the first 15 minutes
    ROUND((SUM(home_0_15) + SUM(away_0_15)) * 100.0 /
NULLIF(SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45) + SUM(home_46_60) +
SUM(away_46_60) + SUM(home_61_75) + SUM(away_61_75) + SUM(home_76_90) +
SUM(away_76_90), 0),2) AS proportion_buts_0_15,
    -- Proportion of goals scored between the 16th and 30th minute
    ROUND((SUM(home_16_30) + SUM(away_16_30)) * 100.0 /
NULLIF(SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45) + SUM(home_46_60) +
SUM(away_46_60) + SUM(home_61_75) + SUM(away_61_75) + SUM(home_76_90) +
SUM(away_76_90), 0),2) AS proportion_buts_16_30,
    -- Proportion of goals scored between the 31st and 45th minute
    ROUND((SUM(home_31_45) + SUM(away_31_45)) * 100.0 /
NULLIF(SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45) + SUM(home_46_60) +
SUM(away_46_60) + SUM(home_61_75) + SUM(away_61_75) + SUM(home_76_90) +
SUM(away_76_90), 0),2) AS proportion_buts_31_45,
    -- Proportion of goals scored between the 46th and 60th minute
    ROUND((SUM(home_46_60) + SUM(away_46_60)) * 100.0 /
NULLIF(SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45) + SUM(home_46_60) +
SUM(away_46_60) + SUM(home_61_75) + SUM(away_61_75) + SUM(home_76_90) +
SUM(away_76_90), 0),2) AS proportion_buts_46_60,
    -- Proportion of goals scored between the 61st and 75th minute
    ROUND((SUM(home_61_75) + SUM(away_61_75)) * 100.0 /
NULLIF(SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45) + SUM(home_46_60) +
SUM(away_46_60) + SUM(home_61_75) + SUM(away_61_75) + SUM(home_76_90) +
SUM(away_76_90), 0),2) AS proportion_buts_61_75,
    -- Proportion of goals scored between the 76th and 90th minute
    ROUND((SUM(home_76_90) + SUM(away_76_90)) * 100.0 /
NULLIF(SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45) + SUM(home_46_60) +
SUM(away_46_60) + SUM(home_61_75) + SUM(away_61_75) + SUM(home_76_90) +
SUM(away_76_90), 0),2) AS proportion_buts_76_90
  FROM
```

```
    info_match
    JOIN season USING(id_season)
    JOIN info_goal USING(id_match)
    WHERE season_name = season_name_input
    GROUP BY
        season_name
    HAVING COUNT(*) >= 5;
$$;

grant execute on function get_distribution_goals(text) to anon;
```

- 27/ Search for the best teams in the 1st period for a given competition

```
CREATE OR REPLACE FUNCTION get_top_teams_1st_period(competition_name_input
TEXT)
RETURNS TABLE (
    season_name text,
    team_name text,
    proportion_buts_1ere_periode NUMERIC(10,2),
    nbr_buts_inscrit_1ere_periode NUMERIC(10,2),
    proportion_buts_0_15 NUMERIC(10,2),
    nbr_buts_0_15 NUMERIC(10,2),
    proportion_buts_16_30 NUMERIC(10,2),
    nbr_buts_16_30 NUMERIC(10,2),
    proportion_buts_31_45 NUMERIC(10,2),
    nbr_buts_31_45 NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
    SELECT
        -- Season name
        season_name,
        -- Team name
        team_name,
        -- Proportion of goals scored in the first half
        SUM(CASE WHEN is_home = 1 THEN home_0_15 + home_16_30 + home_31_45
ELSE away_0_15 + away_16_30 + away_31_45 END) * 100.0 / NULLIF(SUM(CASE
WHEN is_home = 1 THEN home_0_15 + home_16_30 + home_31_45 + home_46_60 +
home_61_75 + home_76_90 ELSE away_0_15 + away_16_30 + away_31_45 + away_46_60
+ away_61_75 + away_76_90 END), 0) AS proportion_buts_inscrit_1ere_periode,
        -- Number of goals scored in the first half
        SUM(CASE WHEN is_home = 1 THEN home_0_15 + home_16_30 + home_31_45
ELSE away_0_15 + away_16_30 + away_31_45 END) AS nbr_buts_inscrit_1ere_periode,
        -- Proportion of goals scored in the first 15 minutes
        SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE away_0_15 END) * 100.0 /
NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE away_0_15 END) +
SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE away_16_30 END) +
SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE away_31_45 END) +
SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END), 0) AS
proportion_buts_0_15,
        -- Number of goals scored in the first 15 minutes
        SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE away_0_15 END) AS
nbr_buts_0_15,
        -- Proportion of goals scored between the 16th and 30th minute
        SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE away_16_30 END) * 100.0
/ NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE away_0_15 END) +
SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE away_16_30 END) +
SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE away_31_45 END) +
        SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END), 0) AS
proportion_buts_16_30,
        -- Number of goals scored between the 16th and 30th minute
        SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE away_16_30 END) AS
nbr_buts_16_30,
        -- Proportion of goals scored between the 31st and 45th minute
        SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE away_31_45 END) * 100.0
/ NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE away_0_15 END) +
SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE away_16_30 END) +
SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE away_31_45 END) +
        SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END), 0) AS
proportion_buts_31_45,
        -- Number of goals scored between the 31th and 45th minutes
        SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE away_31_45 END) AS
nbr_buts_31_45
    FROM (
        -- Data from home teams
        SELECT t.team_name, ig.*, 1 AS is_home, s.season_name, result, c.competition_name
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN team t ON im.id_home_team = t.id_team
        JOIN info_goal ig USING(id_match)
        JOIN competition c USING(id_competition)
        UNION ALL
        -- Data from away teams
        SELECT t.team_name, ig.*, 0 AS is_home, s.season_name, result, c.competition_name
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN team t ON im.id_away_team = t.id_team
        JOIN info_goal ig USING(id_match)
        JOIN competition c USING(id_competition)
    ) AS all_matches
    WHERE competition_name = competition_name_input
    GROUP BY season_name, team_name
    HAVING COUNT(*) >= 5
    ORDER BY proportion_buts_inscrit_1ere_periode DESC
    LIMIT 5;
$$;

grant execute on function get_top_teams_1st_period(text) to anon;
```

- 28/ Search for the best teams in the 2nd period for a given competition

```
CREATE OR REPLACE FUNCTION get_top_teams_2nd_period(competition_name_input
TEXT)
RETURNS TABLE (
    season_name text,
    team_name text,
    proportion_buts_inscrit_2nde_periode NUMERIC(10,2),
    nbr_buts_inscrit_2nde_periode NUMERIC(10,2),
    proportion_buts_46_60 NUMERIC(10,2),
    nbr_buts_46_60 NUMERIC(10,2),
    proportion_buts_61_75 NUMERIC(10,2),
    nbr_buts_61_75 NUMERIC(10,2),
    proportion_buts_76_90 NUMERIC(10,2),
    nbr_buts_76_90 NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
    SELECT
        -- Season name
        season_name,
         -- Team name
        team_name,
        -- Proportion of goals scored in the 2nd half
        SUM(CASE WHEN is_home = 1 THEN home_46_60 + home_61_75 + home_76_90
ELSE away_46_60 + away_61_75 + away_76_90 END) * 100.0 / NULLIF(SUM(CASE
WHEN is_home = 1 THEN home_0_15 + home_16_30 + home_31_45 + home_46_60 +
home_61_75 + home_76_90 ELSE away_0_15 + away_16_30 + away_31_45 + away_46_60
+ away_61_75 + away_76_90 END), 0) AS proportion_buts_inscrit_2nde_periode,
        -- Number of goals scored in the 2nd half
        SUM(CASE WHEN is_home = 1 THEN home_46_60 + home_61_75 + home_76_90
ELSE away_46_60 + away_61_75 + away_76_90 END) AS nbr_buts_inscrit_2nde_periode,
        -- Proportion of goals scored between the 46th and 60th  minute
        SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) * 100.0
/ NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE away_0_15 END) +
SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE away_16_30 END) +
SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE away_31_45 END) +
        SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END), 0) AS
proportion_buts_46_60,
        -- Number of goals scored between the 46th and 60th  minute
        SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) AS
nbr_buts_46_60,
        -- Proportion of goals scored between the 61st and  75th minute
        SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) * 100.0
/ NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE away_0_15 END) +
SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE away_16_30 END) +
SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE away_31_45 END) +
        SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END), 0) AS
proportion_buts_61_75,
        -- Number of goals scored between the 61st and  75th minute
        SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) AS
nbr_buts_61_75,
        -- Proportion of goals scored between the 76st and  90th minute
        SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END) * 100.0
/ NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE away_0_15 END) +
SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE away_16_30 END) +
SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE away_31_45 END) +
        SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END), 0) AS
proportion_buts_76_90,
        -- Number of goals scored between the 76st and  90th minute
        SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END) AS
nbr_buts_76_90
    FROM (
        -- Data from home teams
        SELECT t.team_name, ig.*, 1 AS is_home, s.season_name, result, c.competition_name
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN team t ON im.id_home_team = t.id_team
        JOIN info_goal ig USING(id_match)
        JOIN competition c USING(id_competition)
        UNION ALL
        -- Data from away teams
        SELECT t.team_name, ig.*, 0 AS is_home, s.season_name, result, c.competition_name
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN team t ON im.id_away_team = t.id_team
        JOIN info_goal ig USING(id_match)
        JOIN competition c USING(id_competition)
    ) AS all_matches
    WHERE competition_name = competition_name_input
    GROUP BY season_name, team_name
    HAVING COUNT(*) >= 5
    ORDER BY proportion_buts_inscrit_2nde_periode DESC
    LIMIT 5;
$$;

grant execute on function get_top_teams_2nd_period(text) to anon;
```

- 29/ Search for the best teams in the last 15 minutes for a given competition

```
CREATE OR REPLACE FUNCTION get_top_teams_last_minutes(competition_name_input
TEXT)
RETURNS TABLE (
    season_name text,
    team_name text,
    proportion_buts_76_90 NUMERIC(10,2),
    nbr_buts_76_90 NUMERIC(10,2)

)
LANGUAGE SQL
AS $$
    SELECT
        -- Season name
        season_name,
        -- Team name
        team_name,
        -- Proportion of goals scored between the 76th and 90th minute
        SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END) * 100.0
/ NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE away_0_15 END) +
SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE away_16_30 END) +
SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE away_31_45 END) +
        SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END), 0) AS
proportion_buts_76_90,
        -- Number of goals scored between the 76th and 90th minute
        SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END) AS
nbr_buts_76_90
    FROM (
        -- Data from home teams
        SELECT t.team_name, ig.*, 1 AS is_home, s.season_name, result, c.competition_name
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN team t ON im.id_home_team = t.id_team
        JOIN info_goal ig USING(id_match)
        JOIN competition c USING(id_competition)
        UNION ALL
        -- Data from away teams
        SELECT t.team_name, ig.*, 0 AS is_home, s.season_name, result, c.competition_name
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN team t ON im.id_away_team = t.id_team
        JOIN info_goal ig USING(id_match)
        JOIN competition c USING(id_competition)
    ) AS all_matches
    WHERE competition_name = competition_name_input
    GROUP BY season_name, team_name
    HAVING COUNT(*) >= 5
    ORDER BY proportion_buts_76_90 DESC
    LIMIT 5;
$$;

grant execute on function get_top_teams_last_minutes(text) to anon;
```

- 30/ Search for information on the terrain advantage over a season

```
CREATE OR REPLACE FUNCTION get_home_away_advantage()
RETURNS TABLE (
    season_name text,
    proportion_home_win NUMERIC(10,2),
    proportion_draw NUMERIC(10,2),
    proportion_away_win NUMERIC(10,2),
    home_advantage NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
    SELECT
        -- Season name
        season_name,
        -- Proportion of wins for home teams
        ROUND(COUNT(CASE WHEN is_home = 1 AND result = 1 THEN 1 END) * 100 /
COUNT(CASE WHEN is_home = 1 THEN 1 END),2) AS proportion_home_win,
        -- Proportion of draws for home teams
        ROUND(COUNT(CASE WHEN is_home = 1 AND result = 0 THEN 1 END) * 100 /
COUNT(CASE WHEN is_home = 1 THEN 1 END),2) AS proportion_draw,
        -- Proportion of defeats for home teams
        ROUND(COUNT(CASE WHEN is_home = 1 AND result = 2 THEN 1 END) * 100 /
COUNT(CASE WHEN is_home = 1 THEN 1 END),2) AS proportion_away_win,
        -- Calculation of home advantage
        ROUND((COUNT(CASE WHEN is_home = 1 AND result = 1 THEN 1 END) * 3.0 +
COUNT(CASE WHEN is_home = 1 AND result = 0 THEN 1 END)) * 100 / (COUNT(CASE
WHEN is_home = 1 AND result = 1 THEN 1 END) * 3.0 + COUNT(CASE WHEN is_home
= 1 AND result = 0 THEN 1 END) + COUNT(CASE WHEN is_home = 0 AND result = 2
THEN 1 END) * 3.0 + COUNT(CASE WHEN is_home = 0 AND result = 0 THEN 1
END)),2) AS home_advantage
    FROM (
        -- Data from home teams
        SELECT ig.*, 1 AS is_home, competition_name, season_name
        FROM info_match
        JOIN season USING(id_season)
        JOIN info_goal ig USING(id_match)
        JOIN competition USING(id_competition)
        UNION ALL
        -- Data from away teams
        SELECT ig.*, 0 AS is_home, competition_name, season_name
        FROM info_match
        JOIN season USING(id_season)
        JOIN info_goal ig USING(id_match)
        JOIN competition USING(id_competition)
    ) AS all_matches
    GROUP BY season_name;
$$;

grant execute on function get_home_away_advantage() to anon;
```

- 31/ Search for information on home rankings over the course of a season

```
CREATE OR REPLACE FUNCTION get_rank_home_season(season_name_input TEXT)
RETURNS TABLE (
    team_name text,
    all_matches NUMERIC(10,2),
    number_home_win NUMERIC(10,2),
    number_home_draw NUMERIC(10,2),
    number_home_lose NUMERIC(10,2),
    home_points NUMERIC(10,2),
    avg_home_points NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
    SELECT
        -- Team name
        t.team_name,
        -- Number of matches
        COUNT(im.id_match) AS all_matches,
        -- Number of wins at home
        COUNT(CASE WHEN ig.result = 1 THEN 1 END) AS number_home_win,
        -- Number of draws at home
        COUNT(CASE WHEN ig.result = 0 THEN 1 END) AS number_home_draw,
        -- Number of defeats at home
        COUNT(CASE WHEN ig.result = 2 THEN 1 END) AS number_home_lose,
        -- Number of points obtained at home
        (COUNT(CASE WHEN ig.result = 1 THEN 1 END) * 3 + COUNT(CASE WHEN
ig.result = 0 THEN 1 END)) AS home_points,
        -- Average of points obtained at home
        ROUND((COUNT(CASE WHEN ig.result = 1 THEN 1 END) * 3.0 + COUNT(CASE
WHEN ig.result = 0 THEN 1 END)) / COUNT(im.id_match),2) AS avg_home_points
    FROM info_match im
    JOIN info_goal ig ON im.id_match = ig.id_match
    JOIN season s ON im.id_season = s.id_season
    JOIN team t ON im.id_home_team = t.id_team
    WHERE s.season_name = season_name_input
    GROUP BY t.team_name
    HAVING COUNT(*) >= 5
    ORDER BY home_points DESC;
$$;

grant execute on function get_rank_home_season(text) to anon;
```

- 32/ Search for information on season-long away rankings

```
CREATE OR REPLACE FUNCTION get_rank_away_season(season_name_input TEXT)
RETURNS TABLE (
   team_name text,
   all_matches NUMERIC(10,2),
   number_away_win NUMERIC(10,2),
   number_away_draw NUMERIC(10,2),
   number_away_lose NUMERIC(10,2),
   away_points NUMERIC(10,2),
   avg_away_points NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
   SELECT
      -- Team name
      t.team_name,
      -- Number of matches
      COUNT(im.id_match) AS all_matches,
      -- Number of wins at away
      COUNT(CASE WHEN ig.result = 2 THEN 1 END) AS number_away_win,
      -- Number of draws at away
      COUNT(CASE WHEN ig.result = 0 THEN 1 END) AS number_away_draw,
      -- Number of defeats at away
      COUNT(CASE WHEN ig.result = 1 THEN 1 END) AS number_away_lose,
      -- Number of points obtained at away
      (COUNT(CASE WHEN ig.result = 2 THEN 1 END) * 3 + COUNT(CASE WHEN
ig.result = 0 THEN 1 END)) AS away_points,
      -- Average of points obtained per match at away
      ROUND((COUNT(CASE WHEN ig.result = 2 THEN 1 END) * 3.0 + COUNT(CASE
WHEN ig.result = 0 THEN 1 END)) / COUNT(im.id_match),2) AS avg_away_points
   FROM info_match im
   JOIN info_goal ig ON im.id_match = ig.id_match
   JOIN season s ON im.id_season = s.id_season
   JOIN team t ON im.id_away_team = t.id_team
   WHERE s.season_name = season_name_input
   GROUP BY t.team_name
   HAVING COUNT(*) >= 5
   ORDER BY away_points DESC;
$$;

grant execute on function get_rank_away_season(text) to anon;
```

- 32/ Search for the top 5 home teams in a given competition

```
CREATE OR REPLACE FUNCTION
get_top5_home_rank_competition(competition_name_input TEXT)
RETURNS TABLE (
   season_name text,
   team_name text,
   all_matches NUMERIC(10,2),
   number_home_win NUMERIC(10,2),
   number_home_draw NUMERIC(10,2),
   number_home_lose NUMERIC(10,2),
   home_points NUMERIC(10,2),
   avg_home_points NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
   SELECT
       -- Season name
       s.season_name,
       -- Team name
       t.team_name,
       -- Number of matches
       COUNT(im.id_match) AS all_matches,
       -- Number of wins at home
       COUNT(CASE WHEN ig.result = 1 THEN 1 END) AS number_home_win,
       -- Number of draws at home
       COUNT(CASE WHEN ig.result = 0 THEN 1 END) AS number_home_draw,
       -- Number of defeats at home
       COUNT(CASE WHEN ig.result = 2 THEN 1 END) AS number_home_lose,
       -- Number of points obtained at home
       (COUNT(CASE WHEN ig.result = 1 THEN 1 END) * 3 + COUNT(CASE WHEN
ig.result = 0 THEN 1 END)) AS home_points,
       -- Average of points obtained at home per match
       ROUND((COUNT(CASE WHEN ig.result = 1 THEN 1 END) * 3.0 + COUNT(CASE
WHEN ig.result = 0 THEN 1 END)) / COUNT(im.id_match),2) AS avg_home_points
   FROM info_match im
   JOIN info_goal ig ON im.id_match = ig.id_match
   JOIN season s ON im.id_season = s.id_season
   JOIN competition c ON s.id_competition = c.id_competition
   JOIN team t ON im.id_home_team = t.id_team
   WHERE c.competition_name = competition_name_input
   GROUP BY s.season_name, t.team_name
   HAVING COUNT(*) >= 5
   ORDER BY avg_home_points DESC
   LIMIT 5;
$$;

grant execute on function get_top5_home_rank_competition(text) to anon;
```

- 33/ Season-long search for the best away teams

```
CREATE OR REPLACE FUNCTION
get_top5_away_rank_competition(competition_name_input TEXT)
RETURNS TABLE (
    season_name text,
    team_name text,
    all_matches NUMERIC(10,2),
    number_away_win NUMERIC(10,2),
    number_away_draw NUMERIC(10,2),
   number_away_lose NUMERIC(10,2),
   away_points NUMERIC(10,2),
   avg_away_points NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
   SELECT
      -- Season name
      s.season_name,
      -- Team name
      t.team_name,
      -- Number of matches
      COUNT(im.id_match) AS all_matches,
      -- Number of wins at away
      COUNT(CASE WHEN ig.result = 2 THEN 1 END) AS number_away_win,
      -- Number of draws at away
      COUNT(CASE WHEN ig.result = 0 THEN 1 END) AS number_away_draw,
      -- Number of defeats at away
      COUNT(CASE WHEN ig.result = 1 THEN 1 END) AS number_away_lose,
      -- Number of points obtained at away
      (COUNT(CASE WHEN ig.result = 2 THEN 1 END) * 3 + COUNT(CASE WHEN
ig.result = 0 THEN 1 END)) AS away_points,
      -- Average of points obtained at away per match
      ROUND((COUNT(CASE WHEN ig.result = 2 THEN 1 END) * 3.0 + COUNT(CASE
WHEN ig.result = 0 THEN 1 END)) / COUNT(im.id_match),2) AS avg_away_points
   FROM info_match im
   JOIN info_goal ig ON im.id_match = ig.id_match
   JOIN season s ON im.id_season = s.id_season
   JOIN competition c ON s.id_competition = c.id_competition
   JOIN team t ON im.id_away_team = t.id_team
   WHERE c.competition_name = competition_name_input
   GROUP BY s.season_name, t.team_name
   HAVING COUNT(*) >= 5
   ORDER BY avg_away_points DESC
   LIMIT 5;
$$;

grant execute on function get_top5_away_rank_competition(text) to anon;
```

- Information on analysing a competition

- 34/ Search for information on goals scored over a season, taking into account the competition in general.

```
create or replace function get_avg_goals_stats_by_competition_2()
returns table (
   competition_name text,
   avg_goals_per_match numeric(10,2),
   avg_home_goals numeric(10,2),
   avg_away_goals numeric(10,2)
)
```

```
language sql
as $$
   SELECT
      -- Competition name
      c.competition_name,
      -- Average goals per match
      ROUND((SUM(ig.score_home) + SUM(ig.score_away)) * 1.0 /
COUNT(im.id_match),2) AS avg_goals_per_match,
      -- Average goals per match for home teams
      ROUND(AVG(ig.score_home),2) AS avg_home_goals,
      -- Average goals per match for away teams
      ROUND(AVG(ig.score_away),2) AS avg_away_goals
   FROM info_match im
   JOIN season s USING(id_season)
   JOIN competition c USING(id_competition)
   JOIN info_goal ig USING(id_match)
   GROUP BY c.competition_name;
$$;

grant execute on function get_avg_goals_stats_by_competition_2() to anon;
```

- 35/ Search for the frequency of scores in a given competition

```
CREATE OR REPLACE FUNCTION
get_frequent_score_by_competition(competition_name_input TEXT)
RETURNS TABLE (
   score_home INT,
   score_away INT,
   percentage NUMERIC(5,2)
)
LANGUAGE SQL
AS $$
   WITH score_counts AS (
      SELECT
         -- Competition name
         c.competition_name,
         -- Number of goals scored for home teams
         ig.score_home,
         -- Number of goals scored for away teams
         ig.score_away,
         -- Frequency of score
         COUNT(*) AS frequency
      FROM info_goal ig
      JOIN info_match im USING(id_match)
      JOIN season s USING(id_season)
      JOIN competition c USING(id_competition)
      GROUP BY c.competition_name, ig.score_home, ig.score_away
   ), total_matches AS (
      SELECT
         -- Competition name
```

```
            c.competition_name,
            -- Number of matches
            COUNT(id_match) AS total_matches
        FROM info_match
        JOIN season USING(id_season)
        JOIN competition c USING(id_competition)
        GROUP BY c.competition_name
    )
    SELECT
        -- Score home
        sc.score_home,
        -- Score away
        sc.score_away,
        -- Percentage of score
        ROUND((sc.frequency * 100.0) / NULLIF(tm.total_matches, 0), 2) AS percentage
    FROM score_counts sc
    JOIN total_matches tm ON sc.competition_name = tm.competition_name
    WHERE sc.competition_name = competition_name_input
    ORDER BY percentage DESC;
$$;

grant execute on function get_frequent_score_by_competition(text) to anon;
```

- 36/ Search for information on 1st goal scored over a season

```
CREATE OR REPLACE FUNCTION
get_first_goal_stats_by_competition(competition_name_input TEXT)
RETURNS TABLE (
    competition_name text,
    proportion_no_goal NUMERIC(10,2),
    proportion_1st_goal_home NUMERIC(10,2),
    proportion_1st_goal_away NUMERIC(10,2),
    first_goal_win NUMERIC(10,2),
    first_goal_draw NUMERIC(10,2),
    first_goal_lose NUMERIC(10,2),
    first_goal_home_win NUMERIC(10,2),
    first_goal_home_draw NUMERIC(10,2),
    first_goal_home_lose NUMERIC(10,2),
    first_goal_away_win NUMERIC(10,2),
    first_goal_away_draw NUMERIC(10,2),
    first_goal_away_lose NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
    SELECT
        -- Competition name
        competition_name,
        -- Proportion of matches with no goal
        ROUND(COUNT(CASE WHEN squad_1st_goal = 0 THEN 1 END) * 100 /
COUNT(squad_1st_goal),2) AS proportion_no_goal,
        -- Proportion of first goal scored by home teams
        ROUND(COUNT(CASE WHEN squad_1st_goal = 1 THEN 1 END) * 100 /
COUNT(squad_1st_goal),2) AS proportion_1st_goal_home,
        -- Proportion of first goal scored by away teams
        ROUND(COUNT(CASE WHEN squad_1st_goal = 2 THEN 1 END) * 100 /
COUNT(squad_1st_goal),2) AS proportion_1st_goal_away,
        -- Proportion of goal scored by a team and wins
        ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND result = 1 THEN 1 END) +
COUNT(CASE WHEN squad_1st_goal = 2 AND result = 2 THEN 1 END)) * 100 /
NULLIF(COUNT(CASE WHEN squad_1st_goal = 1 THEN 1 END) + COUNT(CASE
WHEN squad_1st_goal = 2 THEN 1 END), 0),2) AS first_goal_win,
        -- Proportion of goal scored by a team and draws
        ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND result = 0 THEN 1 END) +
COUNT(CASE WHEN squad_1st_goal = 2 AND result = 0 THEN 1 END)) * 100 /
NULLIF(COUNT(CASE WHEN squad_1st_goal = 1 THEN 1 END) + COUNT(CASE
WHEN squad_1st_goal = 2 THEN 1 END), 0),2) AS first_goal_draw,
        -- Proportion of goal scored by a team but loose
        ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND result = 2 THEN 1 END) +
COUNT(CASE WHEN squad_1st_goal = 2 AND result = 1 THEN 1 END)) * 100 /
NULLIF(COUNT(CASE WHEN squad_1st_goal = 1 THEN 1 END) + COUNT(CASE
WHEN squad_1st_goal = 2 THEN 1 END), 0),2) AS first_goal_lose,
        -- Proportion of matches in which the home team scores the 1st goal and wins the match
ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND result = 1 THEN 1 END)) * 100
/ NULLIF(COUNT(CASE WHEN squad_1st_goal = 1 THEN 1 END), 0),2) AS
first_goal_home_win,
        -- Proportion of matches in which the home team scores the 1st goal and draws
        ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND result = 0 THEN 1 END)) *
100 / NULLIF(COUNT(CASE WHEN squad_1st_goal = 1 THEN 1 END), 0),2) AS
first_goal_home_draw,
        -- Proportion of matches in which the home team scores the 1st goal and looses the
match
        ROUND((COUNT(CASE WHEN squad_1st_goal = 1 AND result = 2 THEN 1 END)) *
100 / NULLIF(COUNT(CASE WHEN squad_1st_goal = 1 THEN 1 END), 0),2) AS
first_goal_home_lose,
        -- Proportion of matches in which the away team scores the 1st goal and wins the match
        ROUND((COUNT(CASE WHEN squad_1st_goal = 2 AND result = 2 THEN 1 END)) *
100 / NULLIF(COUNT(CASE WHEN squad_1st_goal = 2 THEN 1 END), 0),2) AS
first_goal_away_win,
        -- Proportion of matches in which the away team scores the 1st goal and draws
        ROUND((COUNT(CASE WHEN squad_1st_goal = 2 AND result = 0 THEN 1 END)) *
1.0 / NULLIF(COUNT(CASE WHEN squad_1st_goal = 2 THEN 1 END), 0),2) AS
first_goal_away_draw,
        -- Proportion of matches in which the away team scores the 1st goal and looses the match
        ROUND((COUNT(CASE WHEN squad_1st_goal = 2 AND result = 1 THEN 1 END)) *
100 / NULLIF(COUNT(CASE WHEN squad_1st_goal = 2 THEN 1 END), 0),2) AS
first_goal_away_lose
    FROM info_match
    JOIN season USING(id_season)
    JOIN competition USING(id_competition)
    JOIN info_goal USING(id_match)
    WHERE competition_name = competition_name_input
    GROUP BY competition_name;
$$;

grant execute on function get_first_goal_stats_by_competition(text) to anon;
```

- 37/ Search for information on the distribution of goals across a competition

```
CREATE OR REPLACE FUNCTION
get_distribution_goals_by_competition(competition_name_input TEXT)
RETURNS TABLE (
    competition_name text,
    proportion_buts_1ere_periode NUMERIC(10,2),
    proportion_buts_2nde_periode NUMERIC(10,2),
    proportion_buts_0_15 NUMERIC(10,2),
    proportion_buts_16_30 NUMERIC(10,2),
    proportion_buts_31_45 NUMERIC(10,2),
    proportion_buts_46_60 NUMERIC(10,2),
    proportion_buts_61_75 NUMERIC(10,2),
    proportion_buts_76_90 NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
    SELECT
        -- Competition name
        competition_name,
        -- Proportion of goals scored in the first half
        ROUND((SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45)) * 100.0 /
NULLIF(SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45) + SUM(home_46_60) +
SUM(away_46_60) + SUM(home_61_75) + SUM(away_61_75) + SUM(home_76_90) +
SUM(away_76_90), 0),2) AS proportion_buts_1ere_periode,
        -- Proportion of goals scored in the second half
        ROUND((SUM(home_46_60) + SUM(away_46_60) + SUM(home_61_75) +
SUM(away_61_75) + SUM(home_76_90) + SUM(away_76_90)) * 100.0 /
NULLIF(SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45) + SUM(home_46_60) +
SUM(away_46_60) + SUM(home_61_75) + SUM(away_61_75) + SUM(home_76_90) +
SUM(away_76_90), 0),2) AS proportion_buts_2nde_periode,
        -- Proportion of goals scored in the first 15 minutes
        ROUND((SUM(home_0_15) + SUM(away_0_15)) * 100.0 /
NULLIF(SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45) + SUM(home_46_60) +
SUM(away_46_60) + SUM(home_61_75) + SUM(away_61_75) + SUM(home_76_90) +
SUM(away_76_90), 0),2) AS proportion_buts_0_15,
        -- Proportion of goals scored between the 16th and 30th minute
        ROUND((SUM(home_16_30) + SUM(away_16_30)) * 100.0 /
NULLIF(SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45) + SUM(home_46_60) +
SUM(away_46_60) + SUM(home_61_75) + SUM(away_61_75) + SUM(home_76_90) +
SUM(away_76_90), 0),2) AS proportion_buts_16_30,
    -- Proportion of goals scored between the 31st and 45th minute
    ROUND((SUM(home_31_45) + SUM(away_31_45)) * 100.0 /
NULLIF(SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45) + SUM(home_46_60) +
SUM(away_46_60) + SUM(home_61_75) + SUM(away_61_75) + SUM(home_76_90) +
SUM(away_76_90), 0),2) AS proportion_buts_31_45,
    -- Proportion of goals scored between the 46th and 60th minute
    ROUND((SUM(home_46_60) + SUM(away_46_60)) * 100.0 /
NULLIF(SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45) + SUM(home_46_60) +
SUM(away_46_60) + SUM(home_61_75) + SUM(away_61_75) + SUM(home_76_90) +
SUM(away_76_90), 0),2) AS proportion_buts_46_60,
    -- Proportion of goals scored between the 61st and 75th minute
    ROUND((SUM(home_61_75) + SUM(away_61_75)) * 100.0 /
NULLIF(SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45) + SUM(home_46_60) +
SUM(away_46_60) + SUM(home_61_75) + SUM(away_61_75) + SUM(home_76_90) +
SUM(away_76_90), 0),2) AS proportion_buts_61_75,
    -- Proportion of goals scored between the 76th and 90th minute
    ROUND((SUM(home_76_90) + SUM(away_76_90)) * 100.0 /
NULLIF(SUM(home_0_15) + SUM(away_0_15) + SUM(home_16_30) +
SUM(away_16_30) + SUM(home_31_45) + SUM(away_31_45) + SUM(home_46_60) +
SUM(away_46_60) + SUM(home_61_75) + SUM(away_61_75) + SUM(home_76_90) +
SUM(away_76_90), 0),2) AS proportion_buts_76_90
  FROM
    info_match
    JOIN season USING(id_season)
    JOIN competition USING(id_competition)
    JOIN info_goal USING(id_match)
  WHERE competition_name = competition_name_input
  GROUP BY competition_name;
$$;
grant execute on function get_distribution_goals_by_competition(text) to anon;
```

- 38/ Search for information on the terrain advantage over a season

```
CREATE OR REPLACE FUNCTION get_home_away_advantage_by_competition()
RETURNS TABLE (
  competition_name text,
  proportion_home_win NUMERIC(10,2),
  proportion_draw NUMERIC(10,2),
  proportion_away_win NUMERIC(10,2),
  home_advantage NUMERIC(10,2)
)
LANGUAGE SQL
AS $$
  SELECT
```

```
    -- Competition name
    competition_name,
    -- Proportion of wins from home team
    ROUND(COUNT(CASE WHEN is_home = 1 AND result = 1 THEN 1 END) * 100 /
COUNT(CASE WHEN is_home = 1 THEN 1 END),2) AS proportion_home_win,
    -- Proportion of draws from home team
    ROUND(COUNT(CASE WHEN is_home = 1 AND result = 0 THEN 1 END) * 100 /
COUNT(CASE WHEN is_home = 1 THEN 1 END),2) AS proportion_draw,
    -- Proportion of defeat from home team
    ROUND(COUNT(CASE WHEN is_home = 1 AND result = 2 THEN 1 END) * 100 /
COUNT(CASE WHEN is_home = 1 THEN 1 END),2) AS proportion_away_win,
    -- Calculation of home advantage
    ROUND((COUNT(CASE WHEN is_home = 1 AND result = 1 THEN 1 END) * 3.0 +
COUNT(CASE WHEN is_home = 1 AND result = 0 THEN 1 END)) * 100 / (COUNT(CASE
WHEN is_home = 1 AND result = 1 THEN 1 END) * 3.0 + COUNT(CASE WHEN is_home
= 1 AND result = 0 THEN 1 END) + COUNT(CASE WHEN is_home = 0 AND result = 2
THEN 1 END) * 3.0 + COUNT(CASE WHEN is_home = 0 AND result = 0 THEN 1
END)),2) AS home_advantage
  FROM (
    -- Data from home teams
    SELECT ig.*, 1 AS is_home, competition_name
    FROM info_match
    JOIN season USING(id_season)
    JOIN info_goal ig USING(id_match)
    JOIN competition USING(id_competition)
    UNION ALL
    -- Data from away teams
    SELECT ig.*, 0 AS is_home, competition_name
    FROM info_match
    JOIN season USING(id_season)
    JOIN info_goal ig USING(id_match)
    JOIN competition USING(id_competition)
  ) AS all_matches
  GROUP BY competition_name;
$$;

grant execute on function get_home_away_advantage_by_competition() to anon;
```