

Documentation du projet Data Viz France

Introduction au projet

L'objectif de ce projet est de poursuivre le travail effectué lors de mon mémoire de M1 : Analyse comparative de 3 facteurs de performance dans le football : l'impact du 1er but, la distribution temporelle des buts et l'influence de l'avantage du terrain sur le match (domicile/extérieur) entre les équipes de jeunes (U17N et U19N) et le monde professionnel (Ligue 1).

Afin d'étendre cette analyse, on comparera cette fois ci les compétitions suivantes, et ceux des saisons 2021/2022 à 2024/2025 (lorsque cela est possible) :

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

Pour rappel, on s'occupera des facteurs suivants :

- l'influence du 1er but sur le match
- la distribution temporelle des buts
- l'influence du paramètre domicile/extérieur
- l'avantage du terrain x 1er but
- le nombre de buts par match

Ce travail s'articulera en plusieurs étapes avec :

- la récupération des données via l'utilisation du web scrapping auprès du site Sofa Score
- le stockage de ces données dans des tables via l'utilisation de Supabase
- l'analyse des données collectées
- la mise en page de l'application web
- le déploiement de cette application.

Web scraping / Stockage des données sur Supabase

Import des librairies

On va ensuite importer des librairies qui seront utiles pour plus tard tels que pandas, numpy ou encore BeautifulSoup. À noter qu'il faudra appliquer la ligne suivante au sein de votre terminal afin d'installer toutes les dépendances nécessaires au bon fonctionnement du projet :

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

Liaison avec la base de données Supabase

On se servira de variables d'environnement afin de stocker nos données personnelles (url du projet et la clé api associée) pour accéder au projet sur Supabase.

Création des requêtes permettant la création des tables sur Supabase (à faire sur Supabase)

Veillez créer également ces tables au sein de votre projet directement sur Supabase. On ajoutera les données plus tard dans le projet.

- Création de la table Compétition

```
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);
```

- Création de la table Saison

```
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);
```

- Création de la table Équipe

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

- Création de la table des informations des matchs

```
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));
```

- Création de la table des informations des buts sur les matchs

```
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));
```

Stocker les informations des compétitions françaises

Ensuite, on va créer la classe Compétition, et sa fonction permettant l'insertion de ces données dans la table sur Supabase associé (et créé précédemment).

Pour récupérer les données de compétitions, on va effectuer du web scrapping auprès du site Sofa Score et le lien ci dessous :

- <https://www.sofascore.com/fr/football/france>.

On va s'infiltrer dans la page source afin de récupérer les informations de chaque compétition via la librairie BeautifulSoup. Après avoir ciblé la classe contenant ces informations, on stockera les données suivantes :

- l'identifiant de la compétition

- le nom de la compétition

- le nom du pays de la compétition
- le lien url de la compétition.

Par la suite, ces dernières seront stockées dans un dataframe, puis insérer dans la table associée, via la fonction `insert_competition` créé précédemment.

Stocker les informations des saisons françaises

Par la suite, avec l'aide de la librairie `psycpg2`, on va accéder à la table précédemment créée, dans le but de stocker les informations des saisons disponibles via la table des compétitions. On utilisera la librairie `conn` pour effectuer la requête suivante : `SELECT id_competition, link_url FROM competition`. Cette requête nous ressort ainsi les identifiants et les liens urls de chacune des compétitions françaises disponibles sur le site Sofascore.

Dans la même logique que la section précédente, on va créer la classe `Season` et sa fonction associée. Cette classe contiendra les informations suivantes :

- l'identifiant de la saison
- le nom de la saison
- l'identifiant de la compétition
- le lien url de la saison.

On va ensuite initialiser un driver pour effectuer notre web scraping à partir de la requête, précédemment créé. À noter que l'on écartera les compétitions suivantes : Coupe de France, Trophée des Champions, Coupe de France Féminine, afin de se focaliser sur les compétitions n'étant pas à élimination directe, et s'établissant plus sur le long-terme. Dans une volonté d'analyser la tendance récente, on ne s'occupera uniquement des 4 dernières saisons française (de 2021/2022 à 2024/2025).

Pour résumer les principales étapes du stockage de ces informations, on va logiquement dans un 1er temps se connecter au lien de la compétition présent dans la table associée. On va ensuite fermer la page de cookie, pouvant bloquer la collecte de données. Pour information, sur ce site existe un menu déroulant afin de sélectionner la saison de notre choix. On s'en servira (dans la mesure du possible) pour cliquer sur les saisons suivantes :

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

Les informations essentielles figurant en 1ere page de ces saisons, on les collectera dans un objet, avant de passer à la saison suivante en effectuant la même opération. Les informations seront ensuite mises sous le format dataframe, et stocker dans la table `season` via la fonction `insert_seasons`. On fermera le driver une fois la tâche effectuée.

Recherche des informations de chaque match et des équipes associés

Pour cette étape, on va récolter les informations de chaque match, et des équipes impliquées à partir des saisons stockées précédemment. On va ré-utiliser la même logique qu'auparavant :

- Accession à la table `season` stockée sur Supabase

- Récupération des données à partir de la requête suivante : `SELECT id_season, link_url FROM season`, donnant l'identifiant de la saison et son lien url
- Création d'une classe `Team`, contenant l'identifiant de l'équipe, ainsi que son nom
- Création d'une fonction `insert_team`
- Création d'une classe `Match`, contenant l'identifiant du match, saison, équipe à domicile et extérieur, la date du match, et enfin son lien url
- Création d'une fonction `insert_matches`.

Étant donné la longueur plus conséquente de cette étape dans le but final de collecter les données sur les équipes et le match associé, on divisera chaque tâche par fonction. À noter que la logique reste semblable aux sections précédentes :

- Ouvrir un driver pour notre web scraping
- Fermeture de la page de cookies lorsque cela est nécessaire
- Récupération de la page d'une saison à partir de la requête créée précédemment
- Stockage des informations suivantes pour chaque match de la journée en cours dans un object (en ne prenant pas en compte les matchs reportés, abandonné, ou donnant lieu à un tapis vert
- Appuyer sur la touche permettant d'accéder à la journée précédente lorsque tous les matchs de la journée courante ont été collecté
- Non prise en compte des saisons passées déjà collecté dans la recherche des données à insérer (permet d'accélérer la collecte des données)
- Stockage de nouveau des informations de ces matchs, et équipes, jusqu'à la 1ère journée, pour enfin passer à la saison suivante
- Mise des données au format dataframe
- Utilisation de la fonction `insert_teams` et `insert_matches` pour stocker tout cela sur notre projet Supabase
- Fermeture du driver une fois toutes les tâches effectuées.

Récupération des informations sur les buts de chaque match

Enfin, pour cette étape, on va récolter les informations de but de chaque match stocké précédemment. On va ré-utiliser la même logique qu'auparavant :

- Accession à la table `info_match` stockée sur Supabase
- Récupération des données à partir de la requête suivante : `SELECT id_match, link_url, id_season FROM info_match`, donnant l'identifiant du match, de sa saison et son lien url
- Récupération de l'information sur tous les identifiants des saisons et leur nom à partir de la requête suivante : `SELECT DISTINCT s.id_season, s.season_name FROM Season s JOIN info_match im ON s.id_season = im.id_season`; Cela permettra de demander à l'utilisateur les saisons qu'il souhaite stocker
- Récupération des identifiants de matchs déjà présent dans la base de données de buts, cela permettant de ne pas récupérer la même information une nouvelle fois lorsque qu'elle a déjà été stocké précédemment
- Création d'une classe `Goal`, contenant l'identifiant du match, le score de l'équipe à domicile et à l'extérieur, le résultat du match, le nombre de buts inscrit par chaque équipe par tranche de 15 minutes et l'influence du 1er but sur le match
- Création d'une fonction `insert_goals`.

Encore une fois, étant donné la longueur plus conséquente de cette étape dans le but final de collecter les données sur les buts pour chacun des matchs, on divisera chaque tâche par fonction.

Il est important de souligner que l'on demande à l'utilisateur les saisons qu'il souhaite collecter dans une volonté réduire les chances que cela plante (dans le cas où trop de données sont à stocker). À noter que la logique reste semblable aux sections précédentes :

- Initialisation d'un driver pour notre web scraping. À noter que les visualisations superflues seront enlevées afin d'accélérer la collecte des informations
- Fermeture de la page de cookies lorsque cela est nécessaire
- Récupération de la page d'un match à partir de la requête créée précédemment (sans prendre en compte les matchs déjà collectés)
- Accession à la section incidents contenant les faits marquants du match, dont sur chaque but du match et le score final
- Extraction du score du match via les colonnes homeScore et awayScore
- Déduction du résultat du match en fonction de la fonction précédente (Victoire à domicile, Nul ou Victoire à l'extérieur)
- Extraction des informations de buts par intervalles grâce aux données incidents. Toutes les colonnes seront égales à 0 si le score est nul et vierge
- Déduction de l'influence du 1er but sur le match à partir de la fonction de result, et de l'information sur l'influence du 1er but
- Mise de toutes ces données au format dataframe
- Non prise en compte des saisons passées déjà collecté, et des matchs déjà collectés dans la recherche des données à insérer (permet d'accélérer la collecte des données)
- Réinitialisation du driver tous les 10 matchs, afin de réduire les chances que le code plante
- Utilisation de la fonction insert_goals pour stocker tout cela sur notre projet Supabase
- Fermeture du driver une fois toutes les tâches effectuées.

Liste des pages et sections

On aura 4 pages distinctes avec ses requêtes associées :

- l'analyse d'une équipe
- l'analyse des confrontations entre deux équipes
- l'analyse d'une saison
- l'analyse d'une compétition.

De plus, chaque page contiendra ces 5 sections :

- les statistiques générales de buts (nombre de buts inscrits et concédés, fréquence de score ect..)
- l'influence du 1er but sur le match
- la distribution temporelles des buts inscrits et concédés (par intervalle de 15 min ou 45min)
- l'influence du paramètre domicile/extérieur
- comparaison avec les autres saisons/compétitions/confrontations

Par ailleurs, des requêtes seront réalisés afin de constater quels sont les meilleures équipes à travers ces indicateurs sur les années récentes.

Arborescence du projet pour la mise en place de l'application

Application

- pages (Analyse_saison.py, Analyse_confrontations_équipe.py, Analyse_saison.py, Analyse_competition.py)
- Image
- CV
- Mémoire
- Documentation
- Accueil.py : Main
- README.md
- requirements.txt

Mise en place de l'application

Accueil

Cette page sert à afficher les objectifs de ce projet, accompagné de diverses ressources afin d'explorer en profondeur ce dernier (Mémoire, Code, Documentation), et en savoir plus sur moi-même (CV en Anglais et en Français).

Section analyse

En tête du projet

Logiquement, on commence chacun des fichiers réservés à l'analyse des données par l'import des librairies (streamlit, pandas, matplotlib, plotly, seaborn, supabase et python-dotenv). Ensuite, le titre de la page se doit d'être nécessairement placé au début du code, elle aura cette structure pour notre projet : `st.set_page_config(page_title="Data Viz 🌐 🇫🇷", page_icon="🇮🇹", layout="wide")`.

Par la suite, on accédera aux variables d'environnement du nom du projet et de l'anon key. Il s'agit des informations indispensables pour se connecter à la base de données sur Supabase en tant qu'utilisateur anonyme.

Initialisation des fonctions et création des pages

Via la fonction `supabase.rpc`, on va créer des fonctions pour chaque requête dans le but de récupérer les informations provenant de chaque procédure (préalablement créé sur Supabase). Pour cela, il faudra renseigner le nom de la procédure (exemple : `get_first_goal_season`) et la liste des paramètres à renseigner (exemple : `season_name`).

On créera également une fonction pour arrondir à deux chiffres après la virgule les valeurs si besoin, ou en entier selon cette dernière. Enfin des fonctions serviront à surligner les équipes/saisons/compétition choisi par l'utilisateur dans une couleur distinctive au sein d'un dataframe.

Affichage de l'application

Une barre latérale sera créée avec les pages disponibles dans le dossier pages, et une demande de sélection de l'équipe/saison/compétition du choix de l'utilisateur selon le type de page. À

partir des choix effectués par l'utilisateur, ce dernier disposera des 5 sections détaillées précédemment, dans lequel figure tous les graphiques ou tableaux associés. L'utilisateur cliquera sur la section de son choix pour accéder aux informations associées.

Chaque partie de code liée à l'affichage d'un graphique/tableaux/jauges dispose de la même logique.

- Dans un 1er temps, on récupérera les données via la fonction de la requête correspondant,
- Mise des données dans un dataframe en associant les données avec un nom de colonne associée,
- Affichage des graphiques/tableaux/jauges via les fonctions issues de plotly, dataframe. pie, ou encore pivot,
- Centrage et choix du titre, et traitement des valeurs numériques pour les afficher correctement (coloration de la colonne choisie si besoin).

Une boucle for sera créé afin de récupérer toutes les informations sur plusieurs saisons dans la section Comparaison entre les équipes/saisons/compétition. Par ailleurs, les données de maximum sur une saison ou une compétition seront stockées dans le cas d'un affichage des jauges. Enfin si l'on ne dispose de pas assez de données sur un aspect concernant une équipe, les graphiques associées ne seront pas affichés.

Annexe : Liste des requêtes/procédures SQL

- Information sur l'analyse d'une équipe

- 1 / Recherche des équipes

```
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/ Recherche des saisons disponibles pour une équipe donnée

```
create or replace function get_seasons(team_name_input text)
returns setof text
language sql
as $$
    select season_name
    from (
        -- Sélection des saisons où l'équipe était à domicile
        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
```

```

-- Sélection des saisons où l'équipe était à l'extérieur
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/ Recherche des statistiques de buts pour une équipe donnée

```

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
    -- Nom de la saison
    s.season_name,
    -- Nom de l'équipe
    t.team_name,
    -- Nombre de buts moyen par match
    ROUND(avg(ig.score_home + ig.score_away), 2) as avg_goals_per_match,
    -- Nombre de buts inscrits en moyenne par une équipe donnée
    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,
    -- Nombre de buts concédés en moyenne par une équipe donnée
    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,
    -- Nombre de buts inscrits en moyenne à domicile par une équipe donnée

```



```

        ROUND(avg(case when im.id_home_team = t.id_team then ig.score_home end), 2) as
avg_team_home_goals,
        -- Nombre de buts inscrits en moyenne à l'extérieur par une équipe donnée
        ROUND(avg(case when im.id_away_team = t.id_team then ig.score_away end), 2) as
avg_team_away_goals,
        -- Nombre de buts concédés à domicile en moyenne par une équipe donnée
        ROUND(avg(case when im.id_home_team = t.id_team then ig.score_away end), 2) as
avg_conceded_home_goals,
        -- Nombre de buts concédés à l'extérieur en moyenne par une équipe donnée
        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/ Recherche les buts inscrits d'une équipe donnée

```

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
        -- Nom de l'équipe
        team_name,
        -- Total de buts marqués
        coalesce(sum(case when is_home = 1 then score_home else score_away end), 0) as
total_goals_scored,
        -- Moyenne de buts marqués
        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 de buts marqués à domicile
        coalesce(sum(case when is_home = 1 then score_home else 0 end), 0) as
goals_scored_home,
        -- Moyenne de buts marqués à domicile
        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 de buts marqués à l'extérieur
coalesce(sum(case when is_home = 0 then score_away else 0 end), 0) as
goals_scored_away,
-- Moyenne de buts marqués à l'extérieur
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 (
-- Domicile
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
-- Extérieur
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/ Recherche les buts concédés d'une équipe donnée

```

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
-- Nom de l'équipe
team_name,
-- Total des buts encaissés
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,
-- Moyenne des buts encaissés
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 des buts encaissés à domicile
SUM(CASE WHEN is_home = 1 THEN score_away ELSE 0 END) AS
goals_conceded_home,
-- Moyenne des buts encaissés à domicile
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 des buts encaissés à l'extérieur
SUM(CASE WHEN is_home = 0 THEN score_home ELSE 0 END) AS
goals_conceded_away,
-- Moyenne des buts encaissés à l'extérieur
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 (
-- Données des matchs à domicile
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
-- Données des matchs à l'extérieur
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
-- Filtrer par saison
WHERE season_name = season_name_input
-- Regrouper par équipe
GROUP BY team_name
-- Filtrer pour les équipes avec au moins 5 matchs
HAVING COUNT(*) >= 5
-- Ordre décroissant des buts encaissés
ORDER BY total_goals_conceded DESC;
$$;

```

grant execute on function get_goals_conceded(text) to anon;

- 6/ Recherche de la fréquence des scores d'une équipe donnée

```
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
            -- Nom de l'équipe
            t.team_name,
            -- Nom de la saison
            s.season_name,
            -- Nombre de but à domicile
            ig.score_home,
            -- Nombre de but à l'extérieur
            ig.score_away,
            -- Fréquence du 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/ Recherche des informations de 1er but inscrit ou encaissé d'une équipe donnée

```

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
    -- Nom de la saison
    season_name,
    -- Nom de l'équipe

```

```

team_name,
-- Proportion que l'équipe marque le 1er but
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 qu'il n'y ait aucun but
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 que l'équipe encaisse le 1er but
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 que lorsque l'équipe évolue à domicile, elle inscrive le 1er but
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 que lorsque l'équipe évolue à domicile, il n'y ait aucun but
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 que lorsque l'équipe évolue à domicile, elle encaisse le 1er but
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 que lorsque l'équipe évolue à l'extérieur, elle inscrive le 1er but
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 que lorsque l'équipe évolue à l'extérieur, il n'y ait aucun but
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 que lorsque l'équipe évolue à l'extérieur, elle encaisse le 1er but
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 des victoires lorsque l'équipe marque le 1er but (en général, à domicile ou à
l'extérieur)
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 des nuls lorsque l'équipe marque le 1er but (en général, à domicile ou à
l'extérieur)

```

```

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 des défaites lorsque l'équipe marque le 1er but (en général, à domicile ou à
l'extérieur)
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 des victoires lorsque l'équipe à domicile marque le 1er but
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 des nuls lorsque l'équipe à domicile marque le 1er but
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 des défaites lorsque l'équipe à domicile marque le 1er but
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 des victoires lorsque l'équipe à l'extérieur marque le 1er but
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 des nuls lorsque l'équipe à l'extérieur marque le 1er but
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 des défaites lorsque l'équipe à l'extérieur marque le 1er but
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 des victoires lorsque l'équipe encaisse le 1er but (en général, à domicile ou
à l'extérieur)
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 des nuls lorsque l'équipe encaisse le 1er but (en général, à domicile ou à
l'extérieur)
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 des défaites lorsque l'équipe encaisse le 1er but (en général, à domicile ou à
l'extérieur)
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 des victoires lorsque l'équipe à domicile encaisse le 1er but
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 des nuls lorsque l'équipe à domicile encaisse le 1er but
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 des défaites lorsque l'équipe à domicile encaisse le 1er but
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 des victoires lorsque l'équipe à l'extérieur encaisse le 1er but
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 des nuls lorsque l'équipe à l'extérieur encaisse le 1er but
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 des défaites lorsque l'équipe à l'extérieur encaisse le 1er but
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 (
-- Données pour les équipes jouant à domicile
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
-- Données pour les équipes jouant à l'extérieur
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/ Recherche des informations de la distribution des buts d'une équipe donnée

```

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
    -- Nom de la saison
    season_name,
    -- Nom de l'équipe
    team_name,
    -- Proportion de buts inscrits en 1ère période
    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 de buts inscrits en 2ème période
    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 de buts inscrits dans les 15 premières minutes
    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 de buts inscrits entre la 16ème et la 30ème 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 de buts inscrits entre la 31ème et la 45ème 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 de buts inscrits entre la 46ème et la 60ème 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 de buts inscrits entre la 61ème et la 75ème 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 de buts inscrits entre la 76ème et la 90ème 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 de buts concédés en 1ère période

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_période,

-- Proportion de buts concédés en 2nd période

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_période,

-- Proportion de buts encaissés dans les 15 premières 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 de buts encaissé entre la 16ème et la 30ème 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 de buts encaissé entre la 31ème et la 45ème 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 de buts encaissé entre la 46ème et la 60ème 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 de buts encaissé entre la 61ème et la 75ème 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 de buts encaissé entre la 76ème et la 90ème 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,
-- Nombre de buts inscrits en 1ère période
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,
-- Nombre de buts inscrits en 2nd période
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,
-- Nombre de buts inscrits dans les 15 premières minutes
NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE away_0_15 END),
0) AS nbr_buts_0_15,
-- Nombre de buts inscrits entre la 16ème et la 30ème minute
NULLIF(SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE away_16_30
END), 0) AS nbr_buts_16_30,
-- Nombre de buts inscrits entre la 31ème et la 45ème minute
NULLIF(SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE away_31_45
END), 0) AS nbr_buts_31_45,
-- Nombre de buts inscrits entre la 46ème et la 60ème minute
NULLIF(SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60
END), 0) AS nbr_buts_46_60,
-- Nombre de buts inscrits entre la 61ème et la 75ème minute
NULLIF(SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75
END), 0) AS nbr_buts_61_75,
-- Nombre de buts inscrits entre la 76ème et la 90ème minute
NULLIF(SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90
END), 0) AS nbr_buts_76_90,
-- Nombre de buts concédés en 1ère période
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,
-- Nombre de buts concédés en 2nd période
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,
-- Nombre de buts encaissés dans les 15 premières minutes
NULLIF(SUM(CASE WHEN is_home = 1 THEN away_0_15 ELSE home_0_15 END),
0) AS buts_encaissés_0_15,
-- Nombre de buts encaissés entre la 16ème et la 30ème minute
NULLIF(SUM(CASE WHEN is_home = 1 THEN away_16_30 ELSE home_16_30
END), 0) AS buts_encaissés_16_30,
-- Nombre de buts encaissés entre la 31ème et la 45ème minute

```

```

        NULLIF(SUM(CASE WHEN is_home = 1 THEN away_31_45 ELSE home_31_45
END), 0) AS buts_encaissés_31_45,
        -- Nombre de buts encaissés entre la 46ème et la 60ème minute
        NULLIF(SUM(CASE WHEN is_home = 1 THEN away_46_60 ELSE home_46_60
END), 0) AS buts_encaissés_46_60,
        -- Nombre de buts encaissés entre la 61ème et la 75ème minute
        NULLIF(SUM(CASE WHEN is_home = 1 THEN away_61_75 ELSE home_61_75
END), 0) AS buts_encaissés_61_75,
        -- Nombre de buts encaissés entre la 76ème et la 90ème minute
        NULLIF(SUM(CASE WHEN is_home = 1 THEN away_76_90 ELSE home_76_90
END), 0) AS buts_encaissés_76_90
FROM (
    -- Données pour les équipes jouant à domicile
    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
    -- Données pour les équipes jouant à l'extérieur
    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/ Recherche des informations sur le facteur Domicile/Extérieur d'une équipe donnée

```

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
            -- Nom de la saison
            s.season_name,
            -- Nom de l'équipe
            t.team_name,
            -- Nombre de matchs joués à domicile
            COUNT(im.id_match) AS home_matches,
            -- Nombre de victoires à domicile
            COUNT(CASE WHEN ig.result = 1 THEN 1 END) AS home_wins,
            -- Nombre de matchs nuls à domicile
            COUNT(CASE WHEN ig.result = 0 THEN 1 END) AS home_draws,
            -- Nombre de défaite à domicile
            COUNT(CASE WHEN ig.result = 2 THEN 1 END) AS home_losses,
            -- Nombre de points obtenus à domicile
            (COUNT(CASE WHEN ig.result = 1 THEN 1 END) * 3 + COUNT(CASE WHEN
ig.result = 0 THEN 1 END)) AS home_points,
            -- Nombre de points moyens à domicile
            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
            -- Nom de la saison
            s.season_name,
            -- Nom de l'équipe
            t.team_name,
            -- Nombre de matchs joués à l'extérieur
            COUNT(im.id_match) AS away_matches,
            -- Nombre de victoire à l'extérieur
            COUNT(CASE WHEN ig.result = 2 THEN 1 END) AS away_wins,
            -- Nombre de matchs nuls à l'extérieur
            COUNT(CASE WHEN ig.result = 0 THEN 1 END) AS away_draws,
            -- Nombre de défaites à l'extérieur
            COUNT(CASE WHEN ig.result = 1 THEN 1 END) AS away_losses,
            -- Nombre de points obtenus à l'extérieur
            (COUNT(CASE WHEN ig.result = 2 THEN 1 END) * 3 + COUNT(CASE WHEN
ig.result = 0 THEN 1 END)) AS away_points,
            -- Nombre de poins moyens à l'extérieur

```

```

        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 de 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,
    -- Avantage du terrain (à domicile)
    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 de 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,
        -- Avantage du terrain (à domicile)
        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 sur l'analyse entre deux équipes

- 10/ Recherche des matchs entre deux équipes

```
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/ Recherche des matchs entre deux équipes

```
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
            -- Nom de la saison
            s.season_name AS season_name,
            -- Nom de l'équipe évoluant à domicile
            th.team_name AS home_team_name,
            -- Nom de l'équipe évoluant à l'extérieur
            ta.team_name AS away_team_name,
            -- Score de l'équipe à domicile
            ig.score_home AS score_home,
            -- Score de l'équipe à l'extérieur
            ig.score_away AS score_away,
            -- Date du match
            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/ Recherche des informations de buts entre deux équipes

```

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
        -- Moyenne des buts marqués par selected_team_home contre selected_team_away
        (domicile et extérieur)
        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,
        -- Moyenne des buts marqués par selected_team_away contre selected_team_home
        (domicile et extérieur)
        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,
        -- Moyenne des buts marqués par selected_team_home contre selected_team_away à
domicile
        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,
        -- Moyenne des buts marqués par selected_team_away contre selected_team_home à
domicile

```

```

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/ Recherche des informations de 1er but entre deux équipes

```

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
        -- Nom de l'équipe
        th.team_name AS team,
        -- Proportion de 1er but inscrit par l'équipe sélectionnée (domicile + extérieur)
        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 qu'il n'y ait aucun but inscrit
        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 de 1er but encaissé par l'équipe sélectionnée

```

```

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 de 1er but inscrit menant à la victoire
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 de 1er but inscrit menant au match nul
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 de 1er but inscrit menant à la défaite
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 de 1er but encaissé menant à la victoire
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 de 1er but encaissé menant au match nul
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 de 1er but encaissé menant à la défaite
    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/ Recherche des informations de distribution de buts entre deux équipes

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
            -- Nom de l'équipe
            th.team_name AS team,
            -- Type de match (Domicile/Extérieur)
            'home' AS match_type,
            -- Nombre de buts inscrits à domicile durant les 15 premières minutes
            SUM(ig.home_0_15) AS goals_0_15,

```

```

-- Nombre de buts inscrits à domicile entre la 15ème et 30ème minutes
SUM(ig.home_16_30) AS goals_16_30,
-- Nombre de buts inscrits à domicile entre la 31ème et 45ème minutes
SUM(ig.home_31_45) AS goals_31_45,
-- Nombre de buts inscrits à domicile entre la 46ème et 60ème minutes
SUM(ig.home_46_60) AS goals_46_60,
-- Nombre de buts inscrits à domicile entre la 61ème et 75ème minutes
SUM(ig.home_61_75) AS goals_61_75,
-- Nombre de buts inscrits à domicile entre la 76ème et 90ème minutes
SUM(ig.home_76_90) AS goals_76_90,
-- Nombre de buts inscrits à domicile durant la 1ère période
SUM(ig.home_0_15 + ig.home_16_30 + ig.home_31_45) AS first_period_goals,
-- Nombre de buts inscrits à domicile durant la 2nd période
SUM(ig.home_46_60 + ig.home_61_75 + ig.home_76_90) AS second_period_goals,
-- Nombre de buts inscrits à domicile au total
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
-- Nom de l'équipe
ta.team_name AS team,
-- Type de match (Domicile/Extérieur)
'away' AS match_type,
-- Nombre de buts inscrits à l'extérieur durant les 15 premières minutes
SUM(ig.away_0_15) AS goals_0_15,
-- Nombre de buts inscrits à l'extérieur entre la 15ème et 30ème minutes
SUM(ig.away_16_30) AS goals_16_30,
-- Nombre de buts inscrits à l'extérieur entre la 31ème et 45ème minutes
SUM(ig.away_31_45) AS goals_31_45,
-- Nombre de buts inscrits à l'extérieur entre la 46ème et 60ème minutes
SUM(ig.away_46_60) AS goals_46_60,
-- Nombre de buts inscrits à l'extérieur entre la 61ème et 75ème minutes
SUM(ig.away_61_75) AS goals_61_75,
-- Nombre de buts inscrits à l'extérieur entre la 76ème et 90ème minutes
SUM(ig.away_76_90) AS goals_76_90,
-- Nombre de buts inscrits à l'extérieur durant la 1ère période
SUM(ig.away_0_15 + ig.away_16_30 + ig.away_31_45) AS first_period_goals,
-- Nombre de buts inscrits à l'extérieur durant la 2nd période
SUM(ig.away_46_60 + ig.away_61_75 + ig.away_76_90) AS second_period_goals,
-- Nombre de buts inscrits à l'extérieur au total
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
-- Nom de l'équipe
team,
-- Nombre de buts inscrits en 1ère période
ROUND(SUM(first_period_goals) * 100.0 / NULLIF(SUM(total_goals), 0), 2) AS
proportion_0_45,
-- Nombre de buts inscrits en 2nd période
ROUND(SUM(second_period_goals) * 100.0 / NULLIF(SUM(total_goals), 0), 2) AS
proportion_46_90,
-- Nombre de buts inscrits durant les 15 premières minutes
ROUND(SUM(goals_0_15) * 100.0 / NULLIF(SUM(total_goals), 0), 2) AS
proportion_0_15,
-- Nombre de buts inscrits entre la 16ème et la 30ème minute
ROUND(SUM(goals_16_30) * 100.0 / NULLIF(SUM(total_goals), 0), 2) AS
proportion_16_30,
-- Nombre de buts inscrits entre la 31ème et la 45ème minute
ROUND(SUM(goals_31_45) * 100.0 / NULLIF(SUM(total_goals), 0), 2) AS
proportion_31_45,
-- Nombre de buts inscrits entre la 46ème minute et la 60ème minute
ROUND(SUM(goals_46_60) * 100.0 / NULLIF(SUM(total_goals), 0), 2) AS
proportion_46_60,
-- Nombre de buts inscrits entre la 61ème et la 75ème minute
ROUND(SUM(goals_61_75) * 100.0 / NULLIF(SUM(total_goals), 0), 2) AS
proportion_61_75,
-- Nombre de buts inscrits entre la 76ème et la 90ème 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/ Recherche de l'influence du facteur Domicile/Extérieur entre les deux équipes

```

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
        -- Nom de l'équipe
        th.team_name,
        -- Nombre de matchs à domicile
        COUNT(im.id_match) AS home_matches,
        -- Nombre de victoires à domicile
        COUNT(CASE WHEN ig.result = 1 THEN 1 END) AS home_wins,
        -- Nombre de matchs nuls à domicile
        COUNT(CASE WHEN ig.result = 0 THEN 1 END) AS home_draws,
        -- Nombre de défaites à domicile
        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
        -- Nom de l'équipe
        ta.team_name,
        -- Nombre de matchs à l'extérieur
        COUNT(im.id_match) AS away_matches,
        -- Nombre de victoires à l'extérieur
        COUNT(CASE WHEN ig.result = 2 THEN 1 END) AS away_wins,
        -- Nombre de matchs nuls à l'extérieur
        COUNT(CASE WHEN ig.result = 0 THEN 1 END) AS away_draws,
        -- Nombre de défaites à l'extérieur
        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
)
-- Sélection des statistiques des matchs à domicile et des totaux (domicile + extérieur)
SELECT
  -- Nom de l'équipe
  h.team_name AS team_name,
  -- Nombre de victoires à domicile
  h.home_wins AS home_wins,
  -- Nombre de nuls à domicile
  h.home_draws AS home_draws,
  -- Nombre de défaites à domicile
  h.home_losses AS home_losses,
  -- Calcul de l'avantage du terrain à domicile
  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,
  -- Somme des victoires, matchs nuls, et défaites (domicile + extérieur)
  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;

- Information sur l'analyse d'une saison

- 16 / Recherche des compétitions disponibles

```

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/ Recherche des saisons disponibles pour une équipe donnée

```

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/ Recherche des informations donnée sur les buts marquées sur une saison en prenant en compte la compétition

```
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
```

```
        -- Nom de la compétition
```

```
        c.competition_name,
```

```
        -- Nom de la saison
```

```
        s.season_name,
```

```
        -- Nombre de buts par match
```

```
        (SUM(ig.score_home) + SUM(ig.score_away)) * 1.0 / COUNT(im.id_match) AS
```

```
avg_goals_per_match,
```

```
        -- Nombre de buts par match pour l'équipe évoluant à domicile
```

```
        AVG(ig.score_home) AS avg_home_goals,
```

```
        -- Nombre de buts par match pour l'équipe évoluant à l'extérieur
```

```
        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/ Recherche de la fréquence des scores d'une équipe donnée

```
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 $$
WITH score_counts AS (
  SELECT
    -- Nom de la compétition
    c.competition_name,
    -- Nom de la saison
    s.season_name,
    -- Score de l'équipe à domicile
    ig.score_home,
    -- Score de l'équipe à l'extérieur
    ig.score_away,
    -- Fréquence d'apparition du score final
    COUNT(*) AS frequency
  FROM info_goal ig
  JOIN info_match im USING(id_match)
  JOIN season s USING(id_season)
  JOIN competition c USING(id_competition)
  GROUP BY c.competition_name, s.season_name, ig.score_home, ig.score_away
), total_matches AS (
  SELECT
    -- Nom de la compétition
    c.competition_name,
    -- Nom de la saison
    s.season_name,
    -- Nombre de match au total
    COUNT(id_match) AS total_matches
  FROM info_match
  JOIN season s USING(id_season)
  JOIN competition c USING(id_competition)
  GROUP BY c.competition_name, s.season_name
)
SELECT
  -- Score de l'équipe à domicile
  sc.score_home,
  -- Score de l'équipe à l'exterieur
  sc.score_away,
  -- Pourcentage d'apparition du score
  ROUND((sc.frequency * 100.0) / NULLIF(tm.total_matches, 0), 2) AS percentage
FROM score_counts sc
JOIN total_matches tm ON sc.competition_name = tm.competition_name AND
sc.season_name = tm.season_name
WHERE sc.season_name = season_name_input
ORDER BY percentage DESC;
$$;

```

grant execute on function get_frequent_score_by_season(text) to anon;

- 20/ Recherche des meilleures équipes en termes de buts inscrits

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
            -- Nom de l'équipe
            team_name,
            -- Nom de la saison
            season_name,
            -- Nombre de buts inscrits sur la saison
            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,
            -- Moyenne de buts inscrits
            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,
            -- Nombre de buts inscrits à domicile
            SUM(CASE WHEN is_home = 1 THEN score_home ELSE 0 END) AS
goals_scored_home,
            -- Moyenne de buts inscrits à domicile
            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,
            -- Nombre de buts inscrits à l'extérieur
            SUM(CASE WHEN is_home = 0 THEN score_away ELSE 0 END) AS
goals_scored_away,
            -- Moyenne de buts inscrits à l'extérieur
            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/ Recherche des équipes ayant encaissé le moins de but

```

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
        -- Nom de l'équipe
        team_name,
        -- Nom de la saison
        season_name,
        -- Nombre de buts concédés
        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,
        -- Moyenne de buts concédés
        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,
        -- Nombre de buts concédés à domicile
        SUM(CASE WHEN is_home = 1 THEN score_away ELSE 0 END) AS
goals_conceded_home,

```

```

-- Moyenne de buts concédés à domicile
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,
-- Nombre de buts concédés à l'extérieur
SUM(CASE WHEN is_home = 0 THEN score_home ELSE 0 END) AS
goals_conceded_away,
-- Moyenne de buts concédés à l'extérieur
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/ Recherche des informations de 1er but inscrit à l'échelle d'une saison

```

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
    -- Nom de la saison
    season_name,
    -- Proportion des matchs où aucun but n'est inscrit
    ROUND(COUNT(CASE WHEN squad_1st_goal = 0 THEN 1 END) * 100 /
COUNT(squad_1st_goal),2) AS proportion_no_goal,
    -- Proportion des matchs où l'équipe à domicile inscrit le 1er but
    ROUND(COUNT(CASE WHEN squad_1st_goal = 1 THEN 1 END) * 100 /
COUNT(squad_1st_goal),2) AS proportion_1st_goal_home,
    -- Proportion des matchs où l'équipe à l'extérieur inscrit le 1er but
    ROUND(COUNT(CASE WHEN squad_1st_goal = 2 THEN 1 END) * 100 /
COUNT(squad_1st_goal),2) AS proportion_1st_goal_away,
    -- Proportion des matchs où l'équipe ayant inscrit le 1er but gagne le match
    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 des matchs où l'équipe ayant inscrit le 1er but fait match nul
    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 des matchs où l'équipe ayant inscrit le 1er but perd le match
    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 des matchs où l'équipe à domicile inscrit le 1er but et gagne le 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 des matchs où l'équipe à domicile inscrit le 1er but et fait match nul
    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 des matchs où l'équipe à domicile inscrit le 1er but mais perd le 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 des matchs où l'équipe à l'extérieur inscrit le 1er but et gagne le 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 des matchs où l'équipe à l'extérieur inscrit le 1er but et fait match nul
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 des matchs où l'équipe à l'extérieur inscrit le 1er but mais perd le 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/ Recherche des informations des meilleurs équipes au niveau du 1er but inscrit

```

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
    -- Nom de la saison
    season_name,
    -- Nom de l'équipe
    team_name,
    -- Proportion que l'équipe marque le 1er but
    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 (
    -- Données pour les équipes jouant à domicile
    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
    -- Données pour les équipes jouant à l'extérieur
    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/ Recherche des informations des meilleurs équipes au niveau du 1er but inscrit vainqueur

```

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
    -- Nom de la saison
    season_name,
    -- Nom de l'équipe
    team_name,
    -- Proportion des victoires lorsque l'équipe marque le 1er but
    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 (
    -- Données pour les équipes jouant à domicile
    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
    -- Données pour les équipes jouant à l'extérieur

```

```

        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/ Recherche des informations des meilleurs équipes au niveau du 1er but concédés mais tout de même vainqueur

```

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
        -- Nom de la saison
        season_name,
        -- Nom de l'équipe
        team_name,
        -- Proportion des victoires lorsque l'équipe encaisse le 1er but (en général, à domicile ou
à l'extérieur)
        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 (
        -- Données pour les équipes jouant à domicile
        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
-- Données pour les équipes jouant à l'extérieur
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/ Recherche des informations sur la distribution des buts à l'échelle d'une saison

```

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
    -- Nom de la saison
    season_name,
    -- Proportion de buts inscrit en 1ère période
    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 de buts inscrit en 2nd période
    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,

```

```

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 de buts inscrit dans les 15 premières 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 de buts inscrit entre la 16ème et la 30ème 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 de buts inscrit entre la 31ème et la 45ème 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 de buts inscrit entre la 46ème et la 60ème 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 de buts inscrit entre la 61ème et la 75ème 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 de buts inscrit durant les 15 dernières minutes du match
    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/ Recherche des meilleures équipes en 1ère période au regard d'une compétition donnée

```
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
        -- Nom de la saison
        season_name,
        -- Nom de l'équipe
        team_name,
        -- Proportion de buts inscrits en 1ère période
        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,
        -- Nombre de buts inscrits en 1ère période
        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 de buts inscrits dans les 15 premières 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,
        -- Nombre de buts inscrits dans les 15 premières minutes
        SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE away_0_15 END) AS
nbr_buts_0_15,
        -- Proportion de buts inscrits entre la 16ème et la 30ème 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,
-- Nombre de buts inscrits entre la 16ème et la 30ème minute
SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE away_16_30 END) AS
nbr_buts_16_30,
-- Proportion de buts inscrits entre la 31ème et la 45ème 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,
-- Nombre de buts inscrits entre la 31ème et la 45ème minute
SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE away_31_45 END) AS
nbr_buts_31_45
FROM (
-- Données pour les équipes jouant à domicile
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
-- Données pour les équipes jouant à l'extérieur
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/ Recherche des meilleures équipes en 2nd période au regard d'une compétition donnée

```

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
    -- Nom de la saison
    season_name,
    -- Nom de l'équipe
    team_name,
    -- Proportion de buts inscrits en 2ème période
    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,
    -- Nombre de buts inscrits en 2ème période
    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 de buts inscrits entre la 46ème et la 60ème 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,
    -- Proportion de buts inscrits entre la 46ème et la 60ème minute
    SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) AS
nbr_buts_46_60,
    -- Proportion de buts inscrits entre la 61ème et la 75ème 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,
    -- Proportion de buts inscrits entre la 61ème et la 75ème minute

```

```

SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) AS
nbr_buts_61_75,
-- Proportion de buts inscrits entre la 76ème et la 90ème 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,
-- Proportion de buts inscrits entre la 76ème et la 90ème minute
SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END) AS
nbr_buts_76_90
FROM (
-- Données pour les équipes jouant à domicile
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
-- Données pour les équipes jouant à l'extérieur
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/ Recherche des meilleures équipes dans les 15 dernières minutes au regard d'une compétition donnée

```

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 $$
    SELECT
        -- Nom de la saison
        season_name,
        -- Nom de l'équipe
        team_name,
        -- Proportion de buts inscrits entre la 76ème et la 90ème minute
        SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END) * 100.0
/ NULLIF(SUM(CASE WHEN is_home = 1 THEN home_0_15 ELSE away_0_15 END) +
SUM(CASE WHEN is_home = 1 THEN home_16_30 ELSE away_16_30 END) +
SUM(CASE WHEN is_home = 1 THEN home_31_45 ELSE away_31_45 END) +
SUM(CASE WHEN is_home = 1 THEN home_46_60 ELSE away_46_60 END) +
SUM(CASE WHEN is_home = 1 THEN home_61_75 ELSE away_61_75 END) +
SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END), 0) AS
proportion_buts_76_90,
        -- Nombre de buts inscrits entre la 76ème et la 90ème minute
        SUM(CASE WHEN is_home = 1 THEN home_76_90 ELSE away_76_90 END) AS
nbr_buts_76_90
    FROM (
        -- Données pour les équipes jouant à domicile
        SELECT t.team_name, ig.*, 1 AS is_home, s.season_name, result, c.competition_name
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN team t ON im.id_home_team = t.id_team
        JOIN info_goal ig USING(id_match)
        JOIN competition c USING(id_competition)
        UNION ALL
        -- Données pour les équipes jouant à l'extérieur
        SELECT t.team_name, ig.*, 0 AS is_home, s.season_name, result, c.competition_name
        FROM info_match im
        JOIN season s USING(id_season)
        JOIN team t ON im.id_away_team = t.id_team
        JOIN info_goal ig USING(id_match)
        JOIN competition c USING(id_competition)
    ) AS all_matches
    WHERE competition_name = competition_name_input
    GROUP BY season_name, team_name
    HAVING COUNT(*) >= 5
    ORDER BY proportion_buts_76_90 DESC
    LIMIT 5;
$$;

```

grant execute on function get_top_teams_last_minutes(text) to anon;

- 30/ Recherche des informations sur l'avantage du terrain à l'échelle d'une saison

```

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 $$
    SELECT
        -- Nom de la saison
        season_name,
        -- Proportion des victoires lorsque l'équipe évolue à domicile
        ROUND(COUNT(CASE WHEN is_home = 1 AND result = 1 THEN 1 END) * 100 /
COUNT(CASE WHEN is_home = 1 THEN 1 END),2) AS proportion_home_win,
        -- Proportion de nuls lorsque l'équipe évolue à domicile
        ROUND(COUNT(CASE WHEN is_home = 1 AND result = 0 THEN 1 END) * 100 /
COUNT(CASE WHEN is_home = 1 THEN 1 END),2) AS proportion_draw,
        -- Proportion de défaites lorsque l'équipe évolue à domicile
        ROUND(COUNT(CASE WHEN is_home = 1 AND result = 2 THEN 1 END) * 100 /
COUNT(CASE WHEN is_home = 1 THEN 1 END),2) AS proportion_away_win,
        -- Calcul de l'avantage du terrain
        ROUND((COUNT(CASE WHEN is_home = 1 AND result = 1 THEN 1 END) * 3.0 +
COUNT(CASE WHEN is_home = 1 AND result = 0 THEN 1 END)) * 100 / (COUNT(CASE
WHEN is_home = 1 AND result = 1 THEN 1 END) * 3.0 + COUNT(CASE WHEN is_home
= 1 AND result = 0 THEN 1 END) + COUNT(CASE WHEN is_home = 0 AND result = 2
THEN 1 END) * 3.0 + COUNT(CASE WHEN is_home = 0 AND result = 0 THEN 1
END))),2) AS home_advantage
    FROM (
        -- Données pour les équipes jouant à domicile
        SELECT ig.*, 1 AS is_home, competition_name, season_name
        FROM info_match
        JOIN season USING(id_season)
        JOIN info_goal ig USING(id_match)
        JOIN competition USING(id_competition)
        UNION ALL
        -- Données pour les équipes jouant à l'extérieur
        SELECT ig.*, 0 AS is_home, competition_name, season_name
        FROM info_match
        JOIN season USING(id_season)
        JOIN info_goal ig USING(id_match)
        JOIN competition USING(id_competition)
    ) AS all_matches
    GROUP BY season_name;
$$;

grant execute on function get_home_away_advantage() to anon;

```

- 31/ Recherche des informations sur le classement à domicile à l'échelle d'une saison

```

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 $$
    SELECT
        -- Nom de l'équipe
        t.team_name,
        -- Nombre de matchs
        COUNT(im.id_match) AS all_matches,
        -- Nombre de victoire à domicile
        COUNT(CASE WHEN ig.result = 1 THEN 1 END) AS number_home_win,
        -- Nombre de nul à domicile
        COUNT(CASE WHEN ig.result = 0 THEN 1 END) AS number_home_draw,
        -- Nombre de défaites à domicile
        COUNT(CASE WHEN ig.result = 2 THEN 1 END) AS number_home_lose,
        -- Nombre de points à domicile
        (COUNT(CASE WHEN ig.result = 1 THEN 1 END) * 3 + COUNT(CASE WHEN
ig.result = 0 THEN 1 END)) AS home_points,
        -- Moyenne de points par match à domicile
        ROUND((COUNT(CASE WHEN ig.result = 1 THEN 1 END) * 3.0 + COUNT(CASE
WHEN ig.result = 0 THEN 1 END)) / COUNT(im.id_match),2) AS avg_home_points
    FROM info_match im
    JOIN info_goal ig ON im.id_match = ig.id_match
    JOIN season s ON im.id_season = s.id_season
    JOIN team t ON im.id_home_team = t.id_team
    WHERE s.season_name = season_name_input
    GROUP BY t.team_name
    HAVING COUNT(*) >= 5
    ORDER BY home_points DESC;
$$;

```

```

grant execute on function get_rank_home_season(text) to anon;

```

- 32/ Recherche des informations sur le classement à l'extérieur à l'échelle d'une saison

```

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 $$
    SELECT
        -- Nom de l'équipe
        t.team_name,
        -- Nombre de matchs
        COUNT(im.id_match) AS all_matches,
        -- Nombre de victoires à l'extérieur
        COUNT(CASE WHEN ig.result = 2 THEN 1 END) AS number_away_win,
        -- Nombre de nuls à l'extérieur
        COUNT(CASE WHEN ig.result = 0 THEN 1 END) AS number_away_draw,
        -- Nombre de défaites à l'extérieur
        COUNT(CASE WHEN ig.result = 1 THEN 1 END) AS number_away_lose,
        -- Nombre de points à l'extérieur
        (COUNT(CASE WHEN ig.result = 2 THEN 1 END) * 3 + COUNT(CASE WHEN
ig.result = 0 THEN 1 END)) AS away_points,
        -- Moyenne de points par match à l'extérieur
        ROUND((COUNT(CASE WHEN ig.result = 2 THEN 1 END) * 3.0 + COUNT(CASE
WHEN ig.result = 0 THEN 1 END)) / COUNT(im.id_match),2) AS avg_away_points
    FROM info_match im
    JOIN info_goal ig ON im.id_match = ig.id_match
    JOIN season s ON im.id_season = s.id_season
    JOIN team t ON im.id_away_team = t.id_team
    WHERE s.season_name = season_name_input
    GROUP BY t.team_name
    HAVING COUNT(*) >= 5
    ORDER BY away_points DESC;
$$;

```

grant execute on function get_rank_away_season(text) to anon;

- 32/ Recherche des 5 meilleurs équipes à domicile d'une compétition donnée

```

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
    -- Nom de la saison
    s.season_name,
    -- Nom de l'équipe
    t.team_name,
    -- Nombre de matchs
    COUNT(im.id_match) AS all_matches,
    -- Nombre de victoires à domicile
    COUNT(CASE WHEN ig.result = 1 THEN 1 END) AS number_home_win,
    -- Nombre de nuls à domicile
    COUNT(CASE WHEN ig.result = 0 THEN 1 END) AS number_home_draw,
    -- Nombre de défaite à domicile
    COUNT(CASE WHEN ig.result = 2 THEN 1 END) AS number_home_lose,
    -- Nombre de points à domicile
    (COUNT(CASE WHEN ig.result = 1 THEN 1 END) * 3 + COUNT(CASE WHEN
ig.result = 0 THEN 1 END)) AS home_points,
    -- Nombre de points moyen à domicile
    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/ Recherche des meilleurs équipes à l'extérieur à l'échelle d'une saison

```

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

```

```

-- Nom de la saison
s.season_name,
-- Nom de l'équipe
t.team_name,
-- Nombre de matchs
COUNT(im.id_match) AS all_matches,
-- Nombre de victoires à l'extérieur
COUNT(CASE WHEN ig.result = 2 THEN 1 END) AS number_away_win,
-- Nombre de nuls à l'extérieur
COUNT(CASE WHEN ig.result = 0 THEN 1 END) AS number_away_draw,
-- Nombre de défaite à l'extérieur
COUNT(CASE WHEN ig.result = 1 THEN 1 END) AS number_away_lose,
-- Nombre de points à l'extérieur
(COUNT(CASE WHEN ig.result = 2 THEN 1 END) * 3 + COUNT(CASE WHEN
ig.result = 0 THEN 1 END)) AS away_points,
-- Nombre de points moyen à l'extérieur
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 sur l'analyse d'une compétition

- 34/ Recherche des informations donnée sur les buts marquées sur une saison en prenant en compte la compétition de manière générale

```

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
    -- Nom de la compétition
    c.competition_name,
    -- Moyenne de buts par match

```

```

        ROUND((SUM(ig.score_home) + SUM(ig.score_away)) * 1.0 /
COUNT(im.id_match),2) AS avg_goals_per_match,
        -- Moyenne de buts par match pour l'équipe évoluant à domicile
        ROUND(AVG(ig.score_home),2) AS avg_home_goals,
        -- Moyenne de buts par match pour l'équipe évoluant à l'extérieur
        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/ Recherche de la fréquence des scores d'une compétition donnée

```

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
            -- Nom de la compétition
            c.competition_name,
            -- Nombre de but inscrit par l'équipe évoluant à domicile sur le match
            ig.score_home,
            -- Nombre de but inscrit par l'équipe évoluant à l'extérieur sur le match
            ig.score_away,
            -- Fréquence d'apparition du score dans la base de données
            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
            -- Nom de la compétition
            c.competition_name,
            -- Nombre de match au total
            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 de l'équipe à domicile
        sc.score_home,
        -- Score de l'équipe à l'extérieur
        sc.score_away,
        -- Pourcentage d'apparition du score final d'un match sur l'échantillon total
        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/ Recherche des informations de 1er but inscrit à l'échelle d'une saison

```

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
        -- Nom de la compétition
        competition_name,
        -- Proportion des matchs où aucun but n'est inscrit
        ROUND(COUNT(CASE WHEN squad_1st_goal = 0 THEN 1 END) * 100 /
COUNT(squad_1st_goal),2) AS proportion_no_goal,
        -- Proportion des matchs où l'équipe à domicile inscrit le 1er but
        ROUND(COUNT(CASE WHEN squad_1st_goal = 1 THEN 1 END) * 100 /
COUNT(squad_1st_goal),2) AS proportion_1st_goal_home,
        -- Proportion des matchs où l'équipe à l'extérieur inscrit le 1er but

```



```

ROUND(COUNT(CASE WHEN squad_1st_goal = 2 THEN 1 END) * 100 /
COUNT(squad_1st_goal),2) AS proportion_1st_goal_away,
-- Proportion des matchs où l'équipe ayant inscrit le 1er but gagne le match
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 des matchs où l'équipe ayant inscrit le 1er but fait match nul
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 des matchs où l'équipe ayant inscrit le 1er but perd le match
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 des matchs où l'équipe à domicile inscrit le 1er but et gagne le 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 des matchs où l'équipe à domicile inscrit le 1er but et fait match nul
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 des matchs où l'équipe à domicile inscrit le 1er but mais perd le 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 des matchs où l'équipe à l'extérieur inscrit le 1er but et gagne le 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 des matchs où l'équipe à l'extérieur inscrit le 1er but et fait match nul
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 des matchs où l'équipe à l'extérieur inscrit le 1er but mais perd le 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/ Recherche des informations sur la distribution des buts à l'échelle d'une compétition

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

 -- Nom de la competition

 competition_name,

 -- Proportion de buts inscrit en 1ère période

 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 de buts inscrit en 2nd période

 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 de buts inscrit dans les 15 premières 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 de buts inscrit entre la 16ème et la 30ème 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 de buts inscrit entre la 31ème et la 45ème 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 de buts inscrit entre la 46ème et la 60ème 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 de buts inscrit entre la 61ème et la 75ème 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 de buts inscrit durant les 15 dernières minutes du match
    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/ Recherche des informations sur l'avantage du terrain à l'échelle d'une saison

```

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
    -- Nom de la compétition
    competition_name,
    -- Proportion des victoires lorsque l'équipe évolue à domicile
    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 de nuls lorsque l'équipe évolue à domicile

```

```

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 de défaites lorsque l'équipe évolue à domicile
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,
-- Calcul de l'avantage du terrain
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 (
-- Données pour les équipes jouant à domicile
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
-- Données pour les équipes jouant à l'extérieur
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;

```