# Homework 1/2  - Football Data from Transfermarkt

## Dataset

The dataset is composed of multiple CSV files with information on competitions, games, clubs, players and appearances that is automatically updated once a week. It includes:

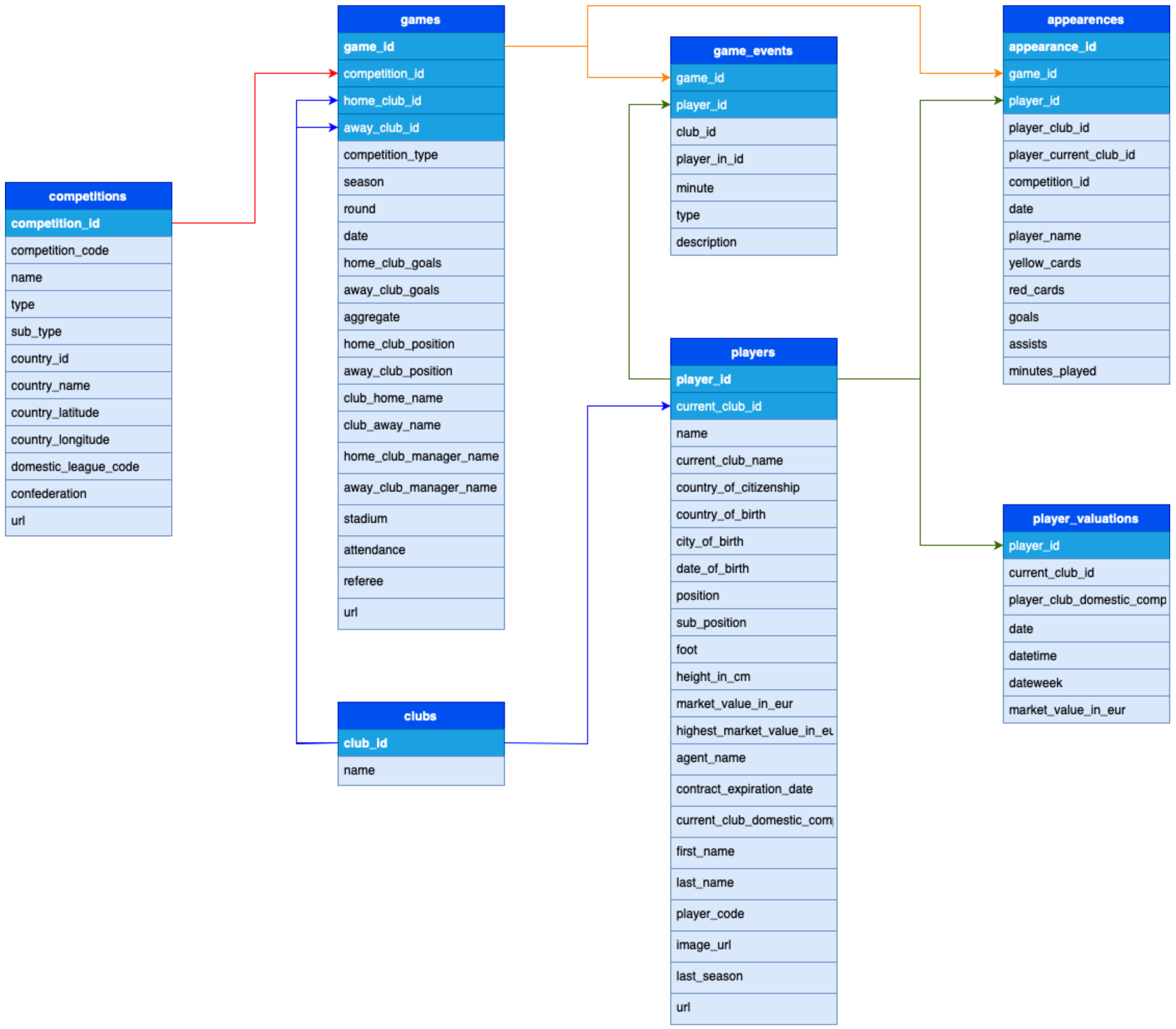
- 60.000+ games from many seasons on all major competitions
- 400+ clubs from those competitions
- 28.000+ players from those clubs
- 300.000+ player market valuations historical records
- 1.000.000+ player appearance records from all games

Football Data from Transfermarkt

Football (Soccer) data scraped from Transfermarkt website

https://www.kaggle.com/datasets/davidcariboo/player-scores

## Tables and Relationships


competitions
competition_id
competition_code
name
type
sub_type
country_id
country_name
country_latitude
country_longitude
domestic_league_code
confederation
url
games
game_id
competition_id
home_club_id
away_club_id
competition_type
season
round
date
home_club_goals
away_club_goals
aggregate
home_club_position
away_club_position
club_home_name
club_away_name
home_club_manager_name
away_club_manager_name
stadium
attendance
referee
url
clubs
club_id
name
game_events
game_id
player_id
club_id
player_in_id
minute
type
description
players
player_id
current_club_id
name
current_club_name
country_of_citizenship
country_of_birth
city_of_birth
date_of_birth
position
sub_position
foot
height_in_cm
market_value_in_eur
highest_market_value_in_eu
agent_name
contract_expiration_date
current_club_domestic_com
first_name
last_name
player_code
image_url
last_season
url
appearences
appearance_id
game_id
player_id
player_club_id
player_current_club_id
competition_id
date
player_name
yellow_cards
red_cards
goals
assists
minutes_played
player_valuations
player_id
current_club_id
player_club_domestic_comp
date
datetime
dateweek
market_value_in_eur

## Preprocessing

### Changed all texts that were not in utf-8 format and gave problems when importing data

- Table **players** (columns: “name”, “city_of_birth”, “first_name”, “last_name”, “agent_name”, “player_code”):

```
df_players = pd.read_csv('players.csv')

for index in df_players.index:
    if type(df_players.loc[index, 'name']) == str:
        df_players.loc[index, 'name'] = unicodedata.normalize('NFKD', df_players.loc[i
            .encode('ASCII', 'ignore').decode('utf-8')
    if type(df_players.loc[index, 'city_of_birth']) == str:
        df_players.loc[index, 'city_of_birth'] = unicodedata.normalize('NFKD', df_play
            .encode('ASCII', 'ignore').decode('utf-8')
    if type(df_players.loc[index, 'first_name']) == str:
        df_players.loc[index, 'first_name'] = unicodedata.normalize('NFKD', df_players
            .encode('ASCII', 'ignore').decode('utf-8')
    if type(df_players.loc[index, 'last_name']) == str:
        df_players.loc[index, 'last_name'] = unicodedata.normalize('NFKD', df_players
            .encode('ASCII', 'ignore').decode('utf-8')
    if type(df_players.loc[index, 'agent_name']) == str:
        df_players.loc[index, 'agent_name'] = unicodedata.normalize('NFKD', df_players
            .encode('ASCII', 'ignore').decode('utf-8')
    if type(df_players.loc[index, 'player_code']) == str:
        df_players.loc[index, 'player_code'] = unicodedata.normalize('NFKD', df_player
            .encode('ASCII', 'ignore').decode('utf-8')

df_players.to_csv('players_cleaned.csv', encoding='utf-8', index=False)
```

- Table **appearences** (column: “player_name”)

```
df_appearances = pd.read_csv('appearances.csv')

for index in df_appearances.index:
    if type(df_appearances.loc[index, 'player_name']) == str:
        df_appearances.loc[index, 'player_name'] = unicodedata.normalize('NFKD', df_ap
            .encode('ASCII', 'ignore').decode('utf-8')

df_appearances.to_csv('appearances_cleaned.csv', encoding='utf-8', index=False)
```

- Table **games** (columns: “home_club_manager_name”, “away_club_manager_name”, “stadium”, “referee”):

```
df_games = pd.read_csv('games.csv')
```

```
for index in df_games.index:
    if type(df_games.loc[index, 'home_club_manager_name']) == str:
        df_games.loc[index, 'home_club_manager_name'] = unicodedata.normalize('NFKD',
                        .loc[index, 'home_club_manager_name']).encode('ASCII', 'ignor
    if type(df_games.loc[index, 'away_club_manager_name']) == str:
            df_games.loc[index, 'away_club_manager_name'] = unicodedata.normalize('NF
                        .loc[index, 'away_club_manager_name']).encode('ASCII', 'ignor
    if type(df_games.loc[index, 'stadium']) == str:
        df_games.loc[index, 'stadium'] = unicodedata.normalize('NFKD', df_games.loc[i
            .encode('ASCII', 'ignore').decode('utf-8')
    if type(df_games.loc[index, 'referee']) == str:
        df_games.loc[index, 'referee'] = unicodedata.normalize('NFKD', df_games.loc[i
            .encode('ASCII', 'ignore').decode('utf-8')

df_games.to_csv('games_cleaned.csv', encoding='utf-8', index=False)
```

## Modified the tables to make the constraints consistent

- Table **clubs**:

```
games = pd.read_csv('games_cleaned.csv')
home = games[['home_club_id', 'club_home_name']]
away = games[['away_club_id', 'club_away_name']]
```

```
home = home.rename(columns={'home_club_id': 'club_id', 'club_home_name': 'name'})
away = away.rename(columns={'away_club_id': 'club_id', 'club_away_name': 'name'})
clubs_v2 = pd.concat([home, away])
clubs_v2 = clubs_v2.drop_duplicates()
clubs_v2.to_csv('clubs_v2.csv', encoding='utf-8', index=False)
```

- Table **players**:

```
players = pd.read_csv('players_cleaned.csv')
game_events = pd.read_csv('game_events.csv')
players_player_id = set(players.player_id)
ge_player_id = set(game_events.player_id)
diff = ge_player_id.difference(players_player_id)
new_df = pd.DataFrame(list(diff))
new_df = new_df.rename(columns={0: 'player_id'})
players_v2 = pd.concat([players, new_df], ignore_index=True)
players_v2['current_club_id'] = pd.to_numeric(players_v2['current_club_id'], errors='c
players_v2['height_in_cm'] = pd.to_numeric(players_v2['height_in_cm'], errors='coerce
players_v2['last_season'] = pd.to_numeric(players_v2['last_season'], errors='coerce')
players_v2.to_csv('players_v2.csv', encoding='utf-8', index=False)
```

## Schema and tables creation (Homework 1)

1. **Schema**

```
DROP SCHEMA IF EXISTS "HW_1" CASCADE;

CREATE SCHEMA IF NOT EXISTS "HW_1"
    AUTHORIZATION pg_database_owner;

COMMENT ON SCHEMA "HW_1"
    IS 'standard public schema';

GRANT USAGE ON SCHEMA "HW_1" TO PUBLIC;

GRANT ALL ON SCHEMA "HW_1" TO pg_database_owner;DROP TABLE IF EXISTS "HW_1".competiti

CREATE TABLE IF NOT EXISTS "HW_1".competitions
(
    competition_id character varying(300) COLLATE pg_catalog."default" NOT NULL,
    competition_code character varying(300) COLLATE pg_catalog."default",
    name character varying(300) COLLATE pg_catalog."default",
    type character varying(300) COLLATE pg_catalog."default",
    sub_type character varying(300) COLLATE pg_catalog."default",
    country_id character varying(300) COLLATE pg_catalog."default",
    country_name character varying(300) COLLATE pg_catalog."default",
    country_latitude numeric,
    country_longitude numeric,
    domestic_league_code character varying(300) COLLATE pg_catalog."default",
    confederation character varying(300) COLLATE pg_catalog."default",
    url character varying(300) COLLATE pg_catalog."default",
    CONSTRAINT competitions_pkey PRIMARY KEY (competition_id)
)

TABLESPACE pg_default;
```

2. **Competitions table** (competitions)

```
DROP TABLE IF EXISTS "HW_1".competitions CASCADE;

CREATE TABLE IF NOT EXISTS "HW_1".competitions
(
    competition_id character varying(300) COLLATE pg_catalog."default" NOT NULL,
    competition_code character varying(300) COLLATE pg_catalog."default",
    name character varying(300) COLLATE pg_catalog."default",
    type character varying(300) COLLATE pg_catalog."default",
    sub_type character varying(300) COLLATE pg_catalog."default",
    country_id character varying(300) COLLATE pg_catalog."default",
    country_name character varying(300) COLLATE pg_catalog."default",
    country_latitude numeric,
    country_longitude numeric,
    domestic_league_code character varying(300) COLLATE pg_catalog."default",
    confederation character varying(300) COLLATE pg_catalog."default",
    url character varying(300) COLLATE pg_catalog."default",
    CONSTRAINT competitions_pkey PRIMARY KEY (competition_id)
)

TABLESPACE pg_default;
```

3. **Clubs table** (clubs)

```
DROP TABLE IF EXISTS "HW_1".clubs CASCADE;

CREATE TABLE IF NOT EXISTS "HW_1".clubs
(
    club_id character varying(300) COLLATE pg_catalog."default" NOT NULL,
    name character varying(300) COLLATE pg_catalog."default",
    CONSTRAINT clubs_pkey PRIMARY KEY (club_id)
)

TABLESPACE pg_default;
```

4. G**ames table** (games)

```
DROP TABLE IF EXISTS "HW_1".games CASCADE;
```

```
CREATE TABLE IF NOT EXISTS "HW_1".games
(
    game_id character varying(300) COLLATE pg_catalog."default" NOT NULL,
    competition_id character varying(300) COLLATE pg_catalog."default",
    competition_type character varying(300) COLLATE pg_catalog."default",
    season integer,
    round character varying(300) COLLATE pg_catalog."default",
    date date,
    home_club_id character varying(300) COLLATE pg_catalog."default",
    away_club_id character varying(300) COLLATE pg_catalog."default",
    home_club_goals integer,
    away_club_goals integer,
    aggregate character varying(300) COLLATE pg_catalog."default",
    home_club_position integer,
    away_club_position integer,
    club_home_name character varying(300) COLLATE pg_catalog."default",
    club_away_name character varying(300) COLLATE pg_catalog."default",
    home_club_manager_name character varying(300) COLLATE pg_catalog."default",
    away_club_manager_name character varying(300) COLLATE pg_catalog."default",
    stadium character varying(300) COLLATE pg_catalog."default",
    attendance integer,
    referee character varying(300) COLLATE pg_catalog."default",
    url character varying(300) COLLATE pg_catalog."default",
    CONSTRAINT games_pkey PRIMARY KEY (game_id),
    CONSTRAINT competition_id_fkey FOREIGN KEY (competition_id)
        REFERENCES "HW_1".competitions (competition_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT home_club_id_fkey FOREIGN KEY (home_club_id)
        REFERENCES "HW_1".clubs (club_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT away_club_id_fkey FOREIGN KEY (away_club_id)
        REFERENCES "HW_1".clubs (club_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT uniqueness UNIQUE (competition_id, home_club_id, away_club_id, date)
)

TABLESPACE pg_default;
```

5. **Players table** (players)

```
DROP TABLE IF EXISTS "HW_1".players CASCADE;

CREATE TABLE IF NOT EXISTS "HW_1".players
(
    player_id character varying(300) COLLATE pg_catalog."default" NOT NULL,
    name character varying(300) COLLATE pg_catalog."default",
    current_club_id character varying(300) COLLATE pg_catalog."default",
    current_club_name character varying(300) COLLATE pg_catalog."default",
    country_of_citizenship character varying(300) COLLATE pg_catalog."default",
    country_of_birth character varying(300) COLLATE pg_catalog."default",
    city_of_birth character varying(300) COLLATE pg_catalog."default",
    date_of_birth date,
    "position" character varying(300) COLLATE pg_catalog."default",
    sub_position character varying(300) COLLATE pg_catalog."default",
    foot character varying(300) COLLATE pg_catalog."default",
    height_in_cm integer,
    market_value_in_eur numeric,
    highest_market_value_in_eur numeric,
    agent_name character varying(300) COLLATE pg_catalog."default",
    contract_expiration_date date,
    current_club_domestic_competition_id character varying(300) COLLATE pg_catalog."de
    first_name character varying(300) COLLATE pg_catalog."default",
    last_name character varying(300) COLLATE pg_catalog."default",
    player_code character varying(300) COLLATE pg_catalog."default",
    image_url character varying(300) COLLATE pg_catalog."default",
    last_season integer,
    url character varying(300) COLLATE pg_catalog."default",
    CONSTRAINT players_pkey PRIMARY KEY (player_id),
    CONSTRAINT current_club_id_fkey FOREIGN KEY (current_club_id)
        REFERENCES "HW_1".clubs (club_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID
)

TABLESPACE pg_default;
```

6. **Game Events table** (game_events)

```
DROP TABLE IF EXISTS "HW_1".game_events CASCADE;

CREATE TABLE IF NOT EXISTS "HW_1".game_events
(
    game_id character varying(300) COLLATE pg_catalog."default",
    minute integer,
    type character varying(300) COLLATE pg_catalog."default",
    club_id character varying(300) COLLATE pg_catalog."default",
    player_id character varying(300) COLLATE pg_catalog."default",
    description character varying(300) COLLATE pg_catalog."default",
    player_in_id character varying(300) COLLATE pg_catalog."default",
    CONSTRAINT game_id_fkey FOREIGN KEY (game_id)
        REFERENCES "HW_1".games (game_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT player_id_fkey FOREIGN KEY (player_id)
```

```
        REFERENCES "HW_1".players (player_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT club_id_fkey FOREIGN KEY (club_id)
        REFERENCES "HW_1".clubs (club_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID
)

TABLESPACE pg_default;
```

7. **Appearences table** (appearences)

```
DROP TABLE IF EXISTS "HW_1".appearences CASCADE;

CREATE TABLE IF NOT EXISTS "HW_1".appearences
(
    appearance_id character varying(300) COLLATE pg_catalog."default" NOT NULL,
    game_id character varying(300) COLLATE pg_catalog."default",
    player_id character varying(300) COLLATE pg_catalog."default",
    player_club_id character varying(300) COLLATE pg_catalog."default",
    player_current_club_id character varying(300) COLLATE pg_catalog."default",
    date date,
    player_name character varying(300) COLLATE pg_catalog."default",
    competition_id character varying(300) COLLATE pg_catalog."default",
    yellow_cards integer,
    red_cards integer,
    goals integer,
    assists integer,
    minutes_played integer,
    CONSTRAINT appearences_pkey PRIMARY KEY (appearance_id)
)

TABLESPACE pg_default;
```

8. **Player Valuations table** (player_valuations)

```
DROP TABLE IF EXISTS "HW_1".player_valuations CASCADE;

CREATE TABLE IF NOT EXISTS "HW_1".player_valuations
(
    date date,
    datetime date,
    dateweek date,
    player_id character varying(300) COLLATE pg_catalog."default",
    current_club_id character varying(300) COLLATE pg_catalog."default",
    market_value_in_eur integer,
    player_club_domestic_competition_id character varying(300) COLLATE pg_catalog."de
    CONSTRAINT player_id_fkey FOREIGN KEY (player_id)
        REFERENCES "HW_1".players (player_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID
)

TABLESPACE pg_default;
```

## Query (Homework 1)

1. Name, market value and number of goals scored with the head in ‘Champions League’ or ‘Europa League’ or ‘Club World Cup’ from defenders

```
SELECT
    p.name,
    count(*) AS num_goals,
    COALESCE(CAST(MAX(market_value_in_eur) AS INT), 0) AS max_market_value
FROM
    "HW_1".game_events ge,
    "HW_1".players p,
    "HW_1".games g,
    "HW_1".competitions c
WHERE
    p.player_id = ge.player_id
    AND g.game_id = ge.game_id
    AND c.competition_id = g.competition_id
    AND description LIKE '%Header%'
    AND position = 'Defender'
    AND attendance > 10000
    AND c.name IN ('Uefa Champions League', 'Europa League', 'Fifa Klub Wm')
GROUP BY
    p.name
ORDER BY
    num_goals DESC;
```

2. Players (with the number of goals) who have scored the most goals after the 80th minute in Serie A in the 2021 season

```
SELECT
    p.name,
    COUNT(*) AS num_goals
FROM
    "HW_1".players p
JOIN "HW_1".game_events ge ON p.player_id = ge.player_id
JOIN "HW_1".games g ON ge.game_id = g.game_id
JOIN "HW_1".competitions c ON g.competition_id = c.competition_id
WHERE
    ge.type = 'Goals'
    AND ge.minute > 80
    AND c.name = 'Serie A'
    AND g.season = 2021
GROUP BY
    p.name
ORDER BY
    num_goals DESC
LIMIT 10;
```

3. Top 10 defenders by number of yellow + red cards and how often they get them

```
SELECT
    a.player_name,
    SUM(a.yellow_cards + a.red_cards) AS total_cards,
    SUM(a.yellow_cards) AS yellows,
    SUM(a.red_cards) AS reds,
    SUM(a.minutes_played) AS minutes_played,
    SUM(a.minutes_played) / (SUM(a.yellow_cards + a.red_cards)+1) AS time_interval
FROM
    "HW_1".appearences a,
    "HW_1".players p
WHERE
```

```
    p.player_id = a.player_id
    AND position = 'Defender'
GROUP BY
  a.player_id,
  a.player_name
ORDER BY
    total_cards DESC
LIMIT 10;
```

3. Top 10 referees by appearances in European competitions (CL, EL, CL and EL qualifiers and Club World Cup)

```
SELECT
    referee,
    COUNT(*) AS Total_Appearances,
    SUM(CASE WHEN competition_id = 'CL' THEN 1 ELSE 0 END) AS Champions_League,
    SUM(CASE WHEN competition_id = 'EL' THEN 1 ELSE 0 END) AS Europa_League,
    SUM(CASE WHEN competition_id = 'USC' THEN 1 ELSE 0 END) AS Super_Cup,
    SUM(CASE WHEN competition_id = 'KLUB' THEN 1 ELSE 0 END) AS Club_World_Cup,
    SUM(CASE WHEN competition_id = 'ECLQ' THEN 1 ELSE 0 END) AS Conference_League_Qual
    SUM(CASE WHEN competition_id = 'CLQ' THEN 1 ELSE 0 END) AS Champions_League_Qual,
    SUM(CASE WHEN competition_id = 'ELQ' THEN 1 ELSE 0 END) AS Europa_League_Qual
FROM (SELECT
            referee,
            competition_id
      FROM
```

```
                 "HW_1".games
          WHERE
                 competition_id IN ('USC', 'CL', 'EL', 'KLUB', 'ECLQ', 'CLQ', 'ELQ')) A
GROUP BY
    referee
ORDER BY
    Total_Appearances DESC
LIMIT 10;
```

5. The 10 games with the most spectators in the history of the Allianz stadium with the match info (formatted with CONCAT and string_agg)

```
SELECT
    g.attendance,
    CONCAT(g.date, ' | ', g.club_home_name, ' - ', g.club_away_name,
           COALESCE (' (' || g.aggregate || ')')) AS match_result,
     COALESCE(string_agg(p.name || ' (' || ge.minute || ')', ' / ' ORDER BY ge.minute)
     CONCAT(c.name, ', ', g.round) AS match_info,
     g.referee
FROM "HW_1".games g
JOIN "HW_1".competitions c ON g.competition_id = c.competition_id
LEFT JOIN "HW_1".game_events ge ON g.game_id = ge.game_id AND ge.type = 'Goals'
LEFT JOIN "HW_1".players p ON ge.player_id = p.player_id
WHERE
    g.stadium = 'Allianz Stadium'
```

```
GROUP BY
    g.date, g.club_home_name, g.club_away_name, g.aggregate, c.name, g.round, g.refere
ORDER BY
    g.attendance DESC
LIMIT 10;
```

6. Teams that have won the most matches in the Champions League with a difference of at least 3 goals

```
SELECT
    CASE WHEN home_club_goals > away_club_goals + 2 THEN club_home_name ELSE club_away
    COUNT(*) AS num_wins
FROM "HW_1".games
JOIN "HW_1".competitions ON "HW_1".games.competition_id = "HW_1".competitions.competit
WHERE
    "HW_1".competitions.sub_type = 'uefa_champions_league'
    AND ((home_club_goals > away_club_goals + 2) OR (away_club_goals > home_club_goals
GROUP BY
    winning_team
ORDER BY
    num_wins DESC
LIMIT 10;
```

7. Players (excluding English players) who scored the most goals and assists in Premier League between 2015 and 2020 in January

```
SELECT
    p.name AS player_name,
    SUM(a.goals + a.assists) AS total_score,
    SUM(a.goals) AS num_goals,
    SUM(a.assists) AS num_assists
FROM "HW_1".appearences a
JOIN "HW_1".games g ON a.game_id = g.game_id
JOIN "HW_1".players p ON a.player_id = p.player_id
WHERE
    a.competition_id = 'GB1'
    AND p.country_of_citizenship <> 'England'
    AND EXTRACT(MONTH FROM g.date) = 1
    AND EXTRACT(YEAR FROM g.date) BETWEEN 2015 AND 2020
GROUP BY
    a.player_id, p.name
ORDER BY
    total_score DESC
LIMIT 10;
```

8. Top 5 coaches (nemesis) who have won the most games against Mourinho

```
SELECT
    manager_name,
    SUM(num_wins) AS total_wins
FROM (
    SELECT
            home_club_manager_name AS manager_name,
            COUNT(*) AS num_wins
    FROM "HW_1"."games"
    WHERE
            away_club_manager_name = 'Jose Mourinho'
            AND home_club_goals > away_club_goals
    GROUP BY
            home_club_manager_name
    UNION ALL
    SELECT
            away_club_manager_name AS manager_name,
            COUNT(*) AS num_wins
    FROM "HW_1"."games"
    WHERE
            home_club_manager_name = 'Jose Mourinho'
            AND away_club_goals > home_club_goals
    GROUP BY
            away_club_manager_name
) AS subquery
GROUP BY
    manager_name
ORDER BY
```

```
    total_wins DESC
LIMIT 5;
```

9. Agents/agencies sorted by value of assisting players

```
SELECT
    agent_name,
    SUM(market_value_in_eur) AS total_market_value,
    COUNT(*) AS num_players,
    SUM(market_value_in_eur)/COUNT(*) AS mean_player_value
FROM "HW_1".players
WHERE
    agent_name <> ''
    AND market_value_in_eur > 0
GROUP BY
    agent_name
ORDER BY
    total_market_value DESC;
```

10. Left-footed players with height < 175cm that scored with the right foot in top European countries (sorted by the number of goals)

```
SELECT
    p.name,
    count(*) AS num_goals
```

```
FROM
    "HW_1".game_events ge,
    "HW_1".players p,
    "HW_1".games g,
    "HW_1".competitions c
WHERE
    p.player_id = ge.player_id
    AND g.game_id = ge.game_id
    AND c.competition_id = g.competition_id
    AND description LIKE '%Right-footed%'
    AND p.foot = 'Left'
    AND height_in_cm < 175
    AND c.country_name  IN ('Italy', 'Germany', 'France', 'Spain', 'England')
GROUP BY
    p.name
ORDER BY
    num_goals DESC;
```

## Optimizing variables types and removing useless table/columns (Homework 2)

Drop the table player_valuations

```
DROP TABLE IF EXISTS "HW_2".player_valuations CASCADE;
```

Checked the max length of text cells and max value of number cells for each column with the following code so as to choose the optimal value.

```
-- Text
SELECT MAX(LENGTH(col_name))
FROM tab_name;

-- Numeric
SELECT MAX(col_name)
FROM tab_name;
```

1. **Appearences table** (appearences)

```
ALTER TABLE "HW_2".appearences
    ALTER COLUMN appearance_id TYPE character varying(15),
    ALTER COLUMN game_id TYPE character varying(7),
    ALTER COLUMN player_id TYPE character varying(7),
    ALTER COLUMN player_club_id TYPE character varying(6),
    ALTER COLUMN player_current_club_id TYPE character varying(6),
    ALTER COLUMN player_name TYPE character varying(35),
    ALTER COLUMN competition_id TYPE character varying(4),
    ALTER COLUMN yellow_cards TYPE smallint,
    ALTER COLUMN red_cards TYPE smallint,
    ALTER COLUMN goals TYPE smallint,
    ALTER COLUMN assists TYPE smallint,
    ALTER COLUMN minutes_played TYPE smallint;
```

2. **Clubs table** (clubs)

```
ALTER TABLE "HW_2".clubs
    ALTER COLUMN club_id TYPE character varying(6),
    ALTER COLUMN name TYPE character varying(35);
```

3. **Competitions table** (competitions)

```
ALTER TABLE "HW_2".competitions
    ALTER COLUMN competition_id TYPE character varying(4),
    ALTER COLUMN competition_code TYPE character varying(43),
    ALTER COLUMN name TYPE character varying(43),
    ALTER COLUMN type TYPE character varying(17),
    ALTER COLUMN sub_type TYPE character varying(40),
    ALTER COLUMN country_id TYPE character varying(3),
    ALTER COLUMN country_name TYPE character varying(11),
    ALTER COLUMN domestic_league_code TYPE character varying(4),
    ALTER COLUMN confederation TYPE character varying(6);

ALTER TABLE "HW_2".competitions DROP COLUMN country_latitude;
ALTER TABLE "HW_2".competitions DROP COLUMN country_longitude;
ALTER TABLE "HW_2".competitions DROP COLUMN url;
```

4. **Game Events table** (game_events)

```
ALTER TABLE "HW_2".game_events
    ALTER COLUMN game_id TYPE varchar(7),
    ALTER COLUMN minute TYPE smallint,
    ALTER COLUMN type TYPE varchar(13),
    ALTER COLUMN club_id TYPE varchar(6),
    ALTER COLUMN player_id TYPE varchar(7),
    ALTER COLUMN description TYPE varchar(48),
    ALTER COLUMN player_in_id TYPE varchar(7);
```

5. **Games table** (games)

```
ALTER TABLE "HW_2".games
    ALTER COLUMN season TYPE smallint,
    ALTER COLUMN home_club_goals TYPE smallint,
    ALTER COLUMN away_club_goals TYPE smallint,
    ALTER COLUMN home_club_position TYPE smallint,
    ALTER COLUMN away_club_position TYPE smallint,
    ALTER COLUMN game_id TYPE character varying(7),
    ALTER COLUMN competition_id TYPE character varying(4),
    ALTER COLUMN competition_type TYPE character varying(17),
    ALTER COLUMN round TYPE character varying(28),
    ALTER COLUMN home_club_id TYPE character varying(6),
    ALTER COLUMN away_club_id TYPE character varying(6),
    ALTER COLUMN aggregate TYPE character varying(5),
    ALTER COLUMN home_club_manager_name TYPE character varying(35),
    ALTER COLUMN away_club_manager_name TYPE character varying(35),
    ALTER COLUMN stadium TYPE character varying(50),
    ALTER COLUMN referee TYPE character varying(45);

ALTER TABLE "HW_2".games DROP COLUMN club_home_name;
ALTER TABLE "HW_2".games DROP COLUMN club_away_name;
ALTER TABLE "HW_2".games DROP COLUMN url;
```

6. **Players table** (players)

```
ALTER TABLE "HW_2".players
    ALTER COLUMN height_in_cm TYPE smallint,
    ALTER COLUMN last_season TYPE smallint,
    ALTER COLUMN market_value_in_eur TYPE numeric(10,1),
    ALTER COLUMN highest_market_value_in_eur TYPE numeric(10,1),
    ALTER COLUMN player_id TYPE character varying(7),
    ALTER COLUMN name TYPE character varying(35),
    ALTER COLUMN current_club_id TYPE character varying(6),
    ALTER COLUMN country_of_citizenship TYPE character varying(25),
    ALTER COLUMN country_of_birth TYPE character varying(30),
    ALTER COLUMN city_of_birth TYPE character varying(50),
    ALTER COLUMN position TYPE character varying(10),
    ALTER COLUMN sub_position TYPE character varying(20),
    ALTER COLUMN foot TYPE character varying(5),
    ALTER COLUMN agent_name TYPE character varying(50),
    ALTER COLUMN current_club_domestic_competition_id TYPE character varying(5),
    ALTER COLUMN first_name TYPE character varying(20),
    ALTER COLUMN last_name TYPE character varying(20),
    ALTER COLUMN player_code TYPE character varying(35);

ALTER TABLE "HW_2".players DROP COLUMN current_club_name;
ALTER TABLE "HW_2".players DROP COLUMN image_url;
ALTER TABLE "HW_2".players DROP COLUMN url;
```

## Updated Query (Homework 2)

1. Name, position and market value of the 10 most expensive players **(HW_1 —> 1)**

```
--- Creation of the View
CREATE VIEW "HW_2".vw_player_stats AS
SELECT p.name,
    count(*) num_goals,
    COALESCE(CAST(MAX(p.market_value_in_eur) AS INT), 0) AS max_market_value
FROM "HW_2".game_events ge
JOIN "HW_2".players p ON p.player_id = ge.player_id
JOIN "HW_2".games g ON g.game_id = ge.game_id
JOIN "HW_2".competitions c ON c.competition_id = g.competition_id
WHERE
    ge.description LIKE '%Header%'
    AND p.position = 'Defender'
    AND g.attendance > 10000
    AND c.name IN ('Uefa Champions League', 'Europa League', 'Fifa Klub Wm')
GROUP BY
    p.name;

--- New Query
SELECT
    name,
    num_goals,
    max_market_value
FROM
    "HW_2".vw_player_stats
```

```
ORDER BY
    num_goals DESC;
```

2. Top 10 defenders by number of yellow + red cards and how often they get them **(HW_1 —> 3)**

```
--- Creation of the Index
CREATE INDEX idx_appearences_player_id ON "HW_2".appearences(player_id);

--- Creation of the View
CREATE VIEW "HW_2".defenders_cards AS
    SELECT
        a.player_id,
        a.player_name,
        SUM(a.yellow_cards + a.red_cards) AS total_cards,
        SUM(a.yellow_cards) AS yellows,
        SUM(a.red_cards) AS reds,
        SUM(a.minutes_played) AS minutes_played
    FROM
        "HW_2".appearences a
    INNER JOIN "HW_2".players p ON p.player_id = a.player_id
    WHERE
        position = 'Defender'
    GROUP BY
        a.player_id,
        a.player_name;
```

```
--- New Query
SELECT
    player_name,
    total_cards,
    yellows,
    reds,
    minutes_played,
    minutes_played / (yellows + reds + 1) AS time_interval
FROM "HW_2".defenders_cards
ORDER BY
    total_cards DESC
LIMIT 10;
```

3. Top 10 referees by appearances in European competitions (CL, EL, CL and EL qualifiers and Club World Cup) **(HW_1 —> 4)**

```
--- Creation of the View
CREATE VIEW "HW_2".competitions_of_interest AS
SELECT referee, competition_id
FROM "HW_2".games
WHERE competition_id IN ('USC', 'CL', 'EL', 'KLUB', 'ECLQ', 'CLQ', 'ELQ');

--- Creation of the Indices
CREATE INDEX games_referee_idx ON "HW_2".games (referee);
CREATE INDEX games_competition_id_idx ON "HW_2".games (competition_id);

--- New Query
SELECT referee,
    COUNT(*) AS Total_Appearances,
     SUM(CASE WHEN competition_id = 'CL' THEN 1 ELSE 0 END) AS Champions_League,
     SUM(CASE WHEN competition_id = 'EL' THEN 1 ELSE 0 END) AS Europa_League,
     SUM(CASE WHEN competition_id = 'USC' THEN 1 ELSE 0 END) AS Super_Cup,
     SUM(CASE WHEN competition_id = 'KLUB' THEN 1 ELSE 0 END) AS Club_World_Cup,
     SUM(CASE WHEN competition_id = 'ECLQ' THEN 1 ELSE 0 END) AS Conference_League_
     SUM(CASE WHEN competition_id = 'CLQ' THEN 1 ELSE 0 END) AS Champions_League_Qu
     SUM(CASE WHEN competition_id = 'ELQ' THEN 1 ELSE 0 END) AS Europa_League_Qual
FROM "HW_2".competitions_of_interest
GROUP BY referee
ORDER BY Total_Appearances DESC
LIMIT 10;
```

4. Players (excluding English players) who scored the most goals and assists in Premier League between 2015 and 2020 **(HW_1 —> 7)**

```
--- Creation of the Indices
CREATE INDEX idx_appearences_player_id ON "HW_2".appearences(player_id);
CREATE INDEX idx_appearences_game_id ON "HW_2".appearences(game_id);
CREATE INDEX idx_games_date ON "HW_2".games(date);
```

```sql
--- New Query
SELECT
    p.name AS player_name,
    SUM(a.goals + a.assists) AS total_score,
    SUM(a.goals) AS num_goals,
    SUM(a.assists) AS num_assists
FROM "HW_2".appearences a
JOIN "HW_2".players p ON a.player_id = p.player_id
JOIN (
    SELECT game_id
    FROM "HW_2".games
    WHERE
        EXTRACT(MONTH FROM date) = 1
        AND EXTRACT(YEAR FROM date) BETWEEN 2015 AND 2020
) g ON a.game_id = g.game_id
WHERE
    a.competition_id = 'GB1'
    AND p.country_of_citizenship <> 'England'
GROUP BY
    a.player_id, p.name
ORDER BY
    total_score DESC
LIMIT 10;
```