European Soccer Team Data Integration Project

Problem Definition:

The European Soccer Database contains information on teams, players, managers, matches, goals, transfers and more across European professional soccer leagues. By loading the European Soccer Database into a data warehouse platform, performing transformations as needed, and building analytics models, we can gain insights to help soccer clubs and managers make better business and team strategy decisions.

In summary, the European Soccer Database can be leveraged in a data analytics environment to uncover insights around club finance, player valuation, ticket sales optimization and more to drive improved business performance for soccer organizations.

Objective:

Key business questions that could be answered by analyzing this data in a data warehouse include:

- Which soccer clubs have the highest revenue and profitability?
- How can we predict the future transfer value of players based on age, position, goals scored and other attributes?
- What soccer clubs over or underperform their wage spending and transfers?
- How can ticket sales be optimized by analyzing attendance data and team performance?

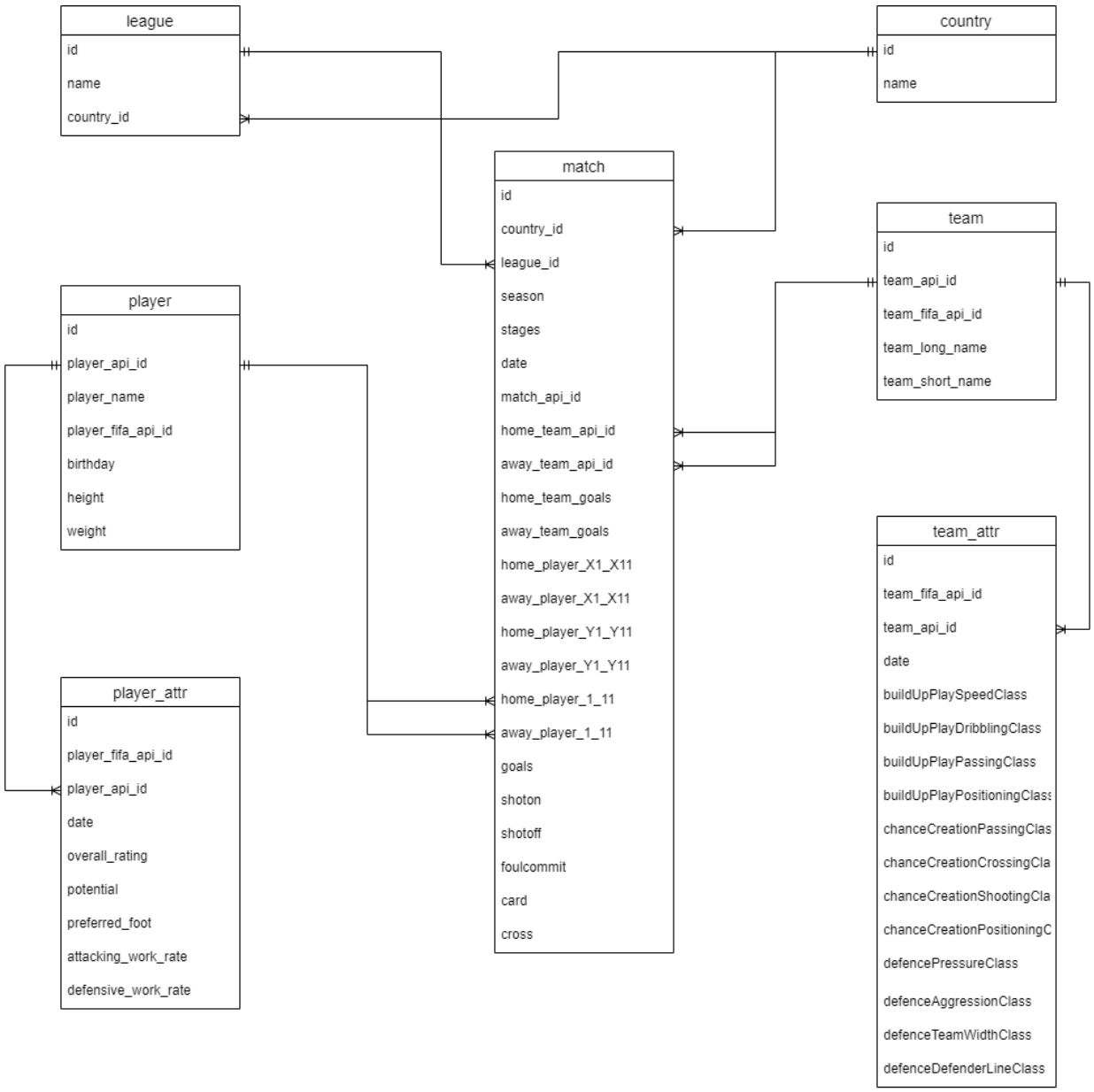
Dataset:

The dataset consists of:

- +25,000 matches
- +10,000 players
- 11 European Countries with their lead championship
- Seasons 2008 to 2016
- Players and Teams' attributes* sourced from EA Sports' FIFA video game series, including the weekly updates.
- Team lines up with squad formation (X, Y coordinates)
- Betting odds from up to 10 providers
- Detailed match events (goal types, possession, corner, cross, fouls, cards etc…) for +10,000 matches

European Soccer Database

ERD:



The Entity-Relationship Diagram (ERD) for the European Soccer Database provides a visual representation of key entities and their relationships within the dataset. It encompasses entities such as 'Matches,' 'Teams,' 'Players,' and 'Leagues,' all interconnected by various relationships. These relationships capture essential aspects of soccer data, including match outcomes, player-team associations, and league affiliations. The ERD serves as a valuable tool for understanding the structural organization of the dataset, facilitating data analysis, and supporting queries related to soccer matches, teams, players, and leagues. It offers a clear overview of how these entities interact, helping analysts and researchers gain insights into the intricate world of European soccer.

Relational Model:

The relational dataset from the European Soccer Database consists of interconnected tables representing matches, teams, players, and leagues. These tables are linked by relationships, allowing for comprehensive soccer data analysis. For example, match records connect to home and away teams, enabling historical performance analysis. This structured format facilitates insights into match outcomes, player statistics, team performance, and league trends.

League

Id, name, country_id
Id, country_id (Not Null)

Player

Id, player_api_id, player_name, player_fifa_api_id, birthday, height, weight

Player_Attributes

Id, player_fifa_api_id, player_api_id, date, overall_rating, potential, preferred_foot, attacking_work_rate, defensice_work_rate

Match

Id, country_id, league_id, season, stages, date, match_api_id, home_team_api-id, away_team_api_id, home_team_goals, away_team_goals, home_player_1_11, away_player_1_11

Player_Mini

Match_api_id, player_api_id

Country

Id, name

Team

Id, team_api_id, team_fifa_api_id, team_long_name, team_short_name

Team_attributes

Id, team_fifa_api_id, team_api_id, date, buildUpPlaySpeedClass, buildUpPlayDribblingClass, buildUpPlayPassingClass, buildUpPlayPositioningClass, chanceCreationPassingClass, chanceCreationCrossingClass, chanceCreationShootingClass, chanceCreationPositioningClass, defencePressureClass, defenceAggressionClass, defenceTeamWidthClass, defenceDefenderLineClass

Keys, Referential Integrity Constraints, and Non-Null Constraints:

- Primary Keys:

- League: id
- Player: player_api_id
- Player_Attributes: id
- Match: id
- Country: id
- Team: team_api_id

- Team_Attributes: id

- Foreign Keys and Referential Integrity Constraints:
- League (country_id) references Country (id)
- match(league id) references League(id)
- match(country id) country(id)
- player_attr(player_api_id) references player(player_api_id)
- match(home_player_1_11) (all the home players) references player(player_api_id)
- match(away_player_1_11) references player(player_api_id)
- match(home_team_api_id) references team(team_api_id)
- match(away_team_api_id) references team(team_api_id)
- team_attr(team_api_id) references team(team_api_id)
- match_api_id(player_mini) references match(match_api_id)
- player_api_id(player_mini) references player(player_api_id)

Methodology Implementation:

The data has been loaded by creating tables and then importing the data using the import/export wizard

League:

```
CREATE TABLE IF NOT EXISTS public."League"
(
    id integer NOT NULL,
    country_id integer,
    name character(256) COLLATE pg_catalog."default" DEFAULT 256,
    CONSTRAINT "League_pkey" PRIMARY KEY (id),
    CONSTRAINT abc FOREIGN KEY (country_id)
        REFERENCES public."Country" (id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."League"
    OWNER to postgres;
```

Query | Query History

```
1  select * from public."League"
2  limit 5;
3
```

Data Output | Messages | Notifications

	id [PK] integer	country_id integer	name character (256)
1	1	1	Belgium Jupiler League
2	1729	1729	England Premier League
3	4769	4769	France Ligue 1
4	7809	7809	Germany 1. Bundesliga
5	10257	10257	Italy Serie A

Country:

```
CREATE TABLE IF NOT EXISTS public."Country"
(
    id integer NOT NULL,
    name character(50) COLLATE pg_catalog."default" DEFAULT 50,
    CONSTRAINT "Country_pkey1" PRIMARY KEY (id)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."Country"
    OWNER to postgres;
```

```
1  select * from public."Country"
2  limit 5;
3
```

Data Output | Messages | Notifications

	id [PK] integer	name character (50)
1	1	Belgium
2	1729	England
3	4769	France
4	7809	Germany
5	10257	Italy

Match:

```
CREATE TABLE IF NOT EXISTS public."Match"
(
    id integer NOT NULL,
    country_id integer,
    league_id integer,
    season character(256) COLLATE pg_catalog."default" DEFAULT 256,
    stage integer,
    date date,
    match_api_id integer NOT NULL,
    home_team_api_id integer,
    away_team_id integer,
    "home-team_goals" integer,
    away_team_goals integer,
    home_player_1 integer,
    home_player_2 integer,
    home_player_3 integer,
    home_player_4 integer,
```

home_player_5 integer,
home_player_6 integer,
home_player_7 integer,
home_player_8 integer,
home_player_9 integer,
home_player_10 integer,
home_player_11 integer,
away_player_1 integer,
away_player_2 integer,
away_player_3 integer,
away_player_4 integer,
away_player_5 integer,
away_player_6 integer,
away_player_7 integer,
away_player_8 integer,
away_player_9 integer,
away_player_10 integer,
away_player_11 integer,
CONSTRAINT "Match_pkey" PRIMARY KEY (match_api_id),
CONSTRAINT a FOREIGN KEY (home_player_2)
REFERENCES public."Player" (player_api_id) MATCH SIMPLE
ON UPDATE NO ACTION
ON DELETE NO ACTION
NOT VALID,
CONSTRAINT abc FOREIGN KEY (league_id)
REFERENCES public."League" (id) MATCH SIMPLE
ON UPDATE NO ACTION
ON DELETE NO ACTION
NOT VALID,
CONSTRAINT b FOREIGN KEY (home_player_3)
REFERENCES public."Player" (player_api_id) MATCH SIMPLE
ON UPDATE NO ACTION
ON DELETE NO ACTION
NOT VALID,
CONSTRAINT c FOREIGN KEY (home_player_3)
REFERENCES public."Player" (player_api_id) MATCH SIMPLE
ON UPDATE NO ACTION
ON DELETE NO ACTION
NOT VALID,
CONSTRAINT d FOREIGN KEY (home_player_4)
REFERENCES public."Player" (player_api_id) MATCH SIMPLE
ON UPDATE NO ACTION
ON DELETE NO ACTION
NOT VALID,

CONSTRAINT def FOREIGN KEY (country_id)
REFERENCES public."Country" (id) MATCH SIMPLE
ON UPDATE NO ACTION
ON DELETE NO ACTION
NOT VALID,
CONSTRAINT e FOREIGN KEY (home_player_5)
REFERENCES public."Player" (player_api_id) MATCH SIMPLE
ON UPDATE NO ACTION
ON DELETE NO ACTION
NOT VALID,
CONSTRAINT f FOREIGN KEY (home_player_6)
REFERENCES public."Player" (player_api_id) MATCH SIMPLE
ON UPDATE NO ACTION
ON DELETE NO ACTION
NOT VALID,
CONSTRAINT g FOREIGN KEY (home_player_7)
REFERENCES public."Player" (player_api_id) MATCH SIMPLE
ON UPDATE NO ACTION
ON DELETE NO ACTION
NOT VALID,
CONSTRAINT ghi FOREIGN KEY (home_team_api_id)
REFERENCES public."Team" (team_api_id) MATCH SIMPLE
ON UPDATE NO ACTION
ON DELETE NO ACTION
NOT VALID,
CONSTRAINT h FOREIGN KEY (home_player_8)
REFERENCES public."Player" (player_api_id) MATCH SIMPLE
ON UPDATE NO ACTION
ON DELETE NO ACTION
NOT VALID,
CONSTRAINT i FOREIGN KEY (home_player_9)
REFERENCES public."Player" (player_api_id) MATCH SIMPLE
ON UPDATE NO ACTION
ON DELETE NO ACTION
NOT VALID,
CONSTRAINT j FOREIGN KEY (home_player_10)
REFERENCES public."Player" (player_api_id) MATCH SIMPLE
ON UPDATE NO ACTION
ON DELETE NO ACTION
NOT VALID,
CONSTRAINT jkl FOREIGN KEY (away_team_id)
REFERENCES public."Team" (team_api_id) MATCH SIMPLE
ON UPDATE NO ACTION
ON DELETE NO ACTION

```
    NOT VALID,
CONSTRAINT k FOREIGN KEY (home_player_11)
    REFERENCES public."Player" (player_api_id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
    NOT VALID,
CONSTRAINT l FOREIGN KEY (away_player_1)
    REFERENCES public."Player" (player_api_id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
    NOT VALID,
CONSTRAINT m FOREIGN KEY (away_player_2)
    REFERENCES public."Player" (player_api_id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
    NOT VALID,
CONSTRAINT mmm FOREIGN KEY (home_player_1)
    REFERENCES public."Player" (player_api_id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
    NOT VALID,
CONSTRAINT n FOREIGN KEY (away_player_3)
    REFERENCES public."Player" (player_api_id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
    NOT VALID,
CONSTRAINT o FOREIGN KEY (away_player_4)
    REFERENCES public."Player" (player_api_id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
    NOT VALID,
CONSTRAINT p FOREIGN KEY (away_player_5)
    REFERENCES public."Player" (player_api_id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
    NOT VALID,
CONSTRAINT q FOREIGN KEY (away_player_6)
    REFERENCES public."Player" (player_api_id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
    NOT VALID,
CONSTRAINT r FOREIGN KEY (away_player_7)
    REFERENCES public."Player" (player_api_id) MATCH SIMPLE
    ON UPDATE NO ACTION
```

```
    ON DELETE NO ACTION
    NOT VALID,
  CONSTRAINT s FOREIGN KEY (away_player_8)
    REFERENCES public."Player" (player_api_id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
    NOT VALID,
  CONSTRAINT t FOREIGN KEY (away_player_9)
    REFERENCES public."Player" (player_api_id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
    NOT VALID,
  CONSTRAINT u FOREIGN KEY (away_player_10)
    REFERENCES public."Player" (player_api_id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
    NOT VALID,
  CONSTRAINT v FOREIGN KEY (away_player_11)
    REFERENCES public."Player" (player_api_id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
    NOT VALID
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."Match"
  OWNER to postgres;
```

Query Query History

```
1  select * from public."Match"
2  limit 5;
3
```

Data Output Messages Notifications

	id integer	country_id integer	league_id integer	season character (256)	sta int
1	1031	1	1	2012/2013	
2	1032	1	1	2012/2013	
3	1033	1	1	2012/2013	
4	1034	1	1	2012/2013	
5	1035	1	1	2012/2013	

Player:

```
CREATE TABLE IF NOT EXISTS public."Player"
(
    id integer,
    player_api_id integer NOT NULL,
    player_name character(150) COLLATE pg_catalog."default" DEFAULT 150,
    player_fifa_id integer,
    birthday date,
    height double precision,
    weight double precision,
    CONSTRAINT "Player_pkey" PRIMARY KEY (player_api_id)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."Player"
    OWNER to postgres;
```

Query | Query History

```
1  select * from public."Player"
2  limit 5;
3
```

Data Output | Messages | Notifications

	id integer	player_api_id [PK] integer	player_name character (150)	player_fifa_id integer	birthday date	height double precision	weight double precisi
1	1	505942	Aaron Appindangoye	218353	1992-02-29	182.88	
2	2	155782	Aaron Cresswell	189615	1989-12-15	170.18	
3	3	162549	Aaron Doran	186170	1991-05-13	170.18	
4	4	30572	Aaron Galindo	140161	1982-05-08	182.88	
5	5	23780	Aaron Hughes	17725	1979-11-08	182.88	

Player_Attributes:

```
CREATE TABLE IF NOT EXISTS public."PlayerAttributes"
(
    id integer NOT NULL,
    player_fifa_api_id integer,
    player_api_id integer,
    date date,
    overall_rating integer,
    potential integer,
    preferred_foot character(10) COLLATE pg_catalog."default" DEFAULT 10,
    attacking_work_rate character(20) COLLATE pg_catalog."default" DEFAULT 20,
    defensive_work_rate character(20) COLLATE pg_catalog."default" DEFAULT 20,
```

```
    CONSTRAINT "PlayerAttributes_pkey" PRIMARY KEY (id),
    CONSTRAINT abc FOREIGN KEY (player_api_id)
        REFERENCES public."Player" (player_api_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."PlayerAttributes"
    OWNER to postgres;
```

Query | Query History

```
1  select * from public."PlayerAttributes"
2  limit 5;
3
```

Data Output | Messages | Notifications

	id [PK] integer	player_fifa_api_id integer	player_api_id integer	date date	overall_rating integer	potential integer	preferred_foot character (10)	attacking_work_rate character (20)	defensive_work_rate character (20)
1	1	218353	505942	2016-02-18	67	71	right	medium	medium
2	2	218353	505942	2015-11-19	67	71	right	medium	medium
3	3	218353	505942	2015-09-21	62	66	right	medium	medium
4	4	218353	505942	2015-03-20	61	65	right	medium	medium
5	5	218353	505942	2007-02-22	61	65	right	medium	medium

Team:

```
CREATE TABLE IF NOT EXISTS public."Team"
(
    id integer,
    team_api_id integer NOT NULL,
    team_fifa_api_id integer,
    team_long_name character(256) COLLATE pg_catalog."default" DEFAULT 256,
    team_short_name character(256) COLLATE pg_catalog."default" DEFAULT 256,
    CONSTRAINT "Team_pkey" PRIMARY KEY (team_api_id)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."Team"
    OWNER to postgres;
```

Query Query History

```
1  select * from public."Team"
2  limit 5;
3
```

Data Output Messages Notifications

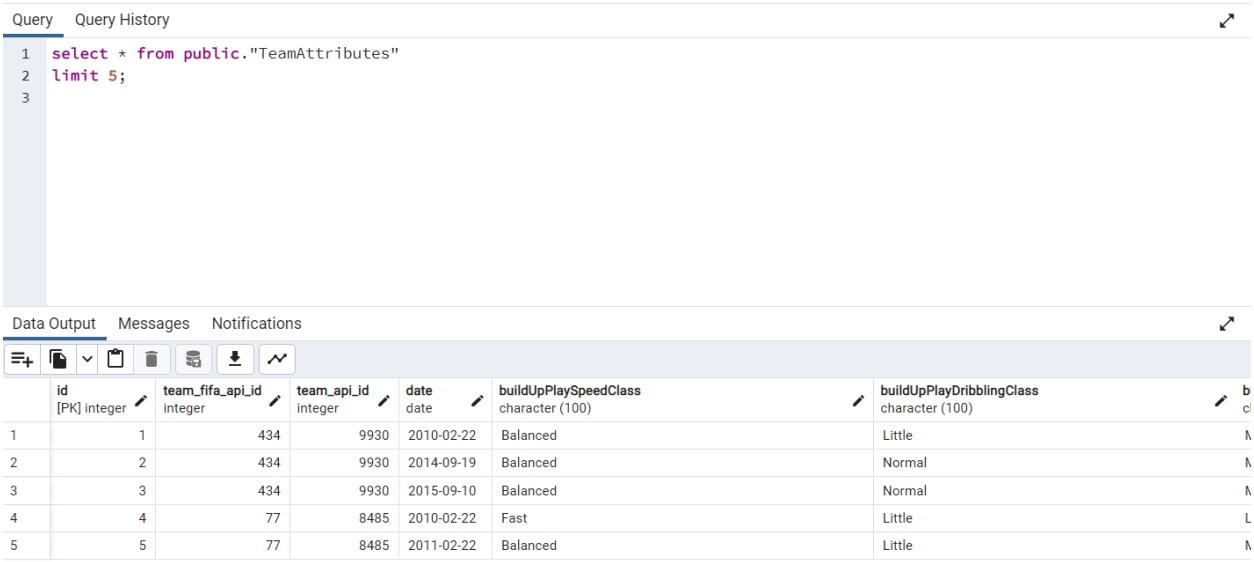
	id integer	team_api_id [PK] integer	team_fifa_api_id integer	team_long_name character (256)
1	1	9987	673	KRC Genk
2	2	9993	675	Beerschot AC
3	3	10000	15005	SV Zulte-Waregem
4	4	9994	2007	Sporting Lokeren
5	5	9984	1750	KSV Cercle Brugge

Team_Attributes:

```
CREATE TABLE IF NOT EXISTS public."TeamAttributes"
(
    id integer NOT NULL,
    team_fifa_api_id integer,
    team_api_id integer,
    date date,
    "buildUpPlaySpeedClass" character(100) COLLATE pg_catalog."default" DEFAULT 100,
    "buildUpPlayDribblingClass" character(100) COLLATE pg_catalog."default" DEFAULT 100,
    "buildUpPlayPassingClass" character(100) COLLATE pg_catalog."default" DEFAULT 100,
    "buildUpPlayPositioningClass" character(100) COLLATE pg_catalog."default" DEFAULT 100,
    "chanceCreationPassingClass" character(100) COLLATE pg_catalog."default" DEFAULT 100,
    "chanceCreationCrossingClass" character(100) COLLATE pg_catalog."default" DEFAULT 100,
    "chanceCreationShootingClass" character(100) COLLATE pg_catalog."default" DEFAULT 100,
    "chanceCreationPositioningClass" character(100) COLLATE pg_catalog."default" DEFAULT 100,
    "defencePressureClass" character(100) COLLATE pg_catalog."default" DEFAULT 100,
    "defenceAggressionClass" character(100) COLLATE pg_catalog."default" DEFAULT 100,
    "defenceTeamWidthClass" character(100) COLLATE pg_catalog."default" DEFAULT 100,
    "defenceDefenderLineClass" character(100) COLLATE pg_catalog."default" DEFAULT 100,
    CONSTRAINT "TeamAttributes_pkey" PRIMARY KEY (id),
    CONSTRAINT abc FOREIGN KEY (team_api_id)
        REFERENCES public."Team" (team_api_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID
)
```

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public."TeamAttributes"
OWNER to postgres;



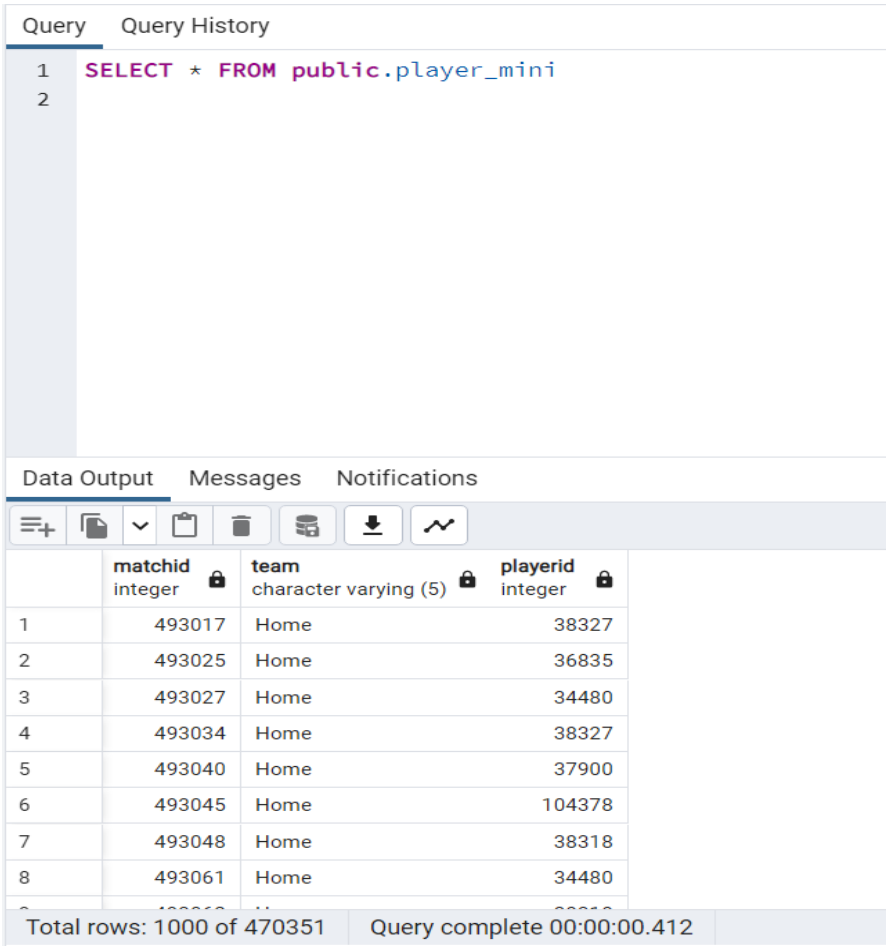
	id [PK] integer	team_fifa_api_id integer	team_api_id integer	date date	buildUpPlaySpeedClass character (100)	buildUpPlayDribblingClass character (100)
1	1	434	9930	2010-02-22	Balanced	Little
2	2	434	9930	2014-09-19	Balanced	Normal
3	3	434	9930	2015-09-10	Balanced	Normal
4	4	77	8485	2010-02-22	Fast	Little
5	5	77	8485	2011-02-22	Balanced	Little

Player_mini:
A bridge table was constructed to account for the many-to-many relationship between the match and the player table.

-- Create the player_mini table
CREATE TABLE public.player_mini (
MatchID INTEGER,
Team VARCHAR(5),
PlayerID INTEGER
);

-- Unpivot and insert data for all players from the "Match" table into "player_mini"
INSERT INTO public.player_mini (MatchID, Team, PlayerID)
SELECT MatchID, 'Home', home_player_1 FROM public."Match" WHERE home_player_1 IS NOT NULL
UNION ALL
SELECT MatchID, 'Home', home_player_2 FROM public."Match" WHERE home_player_2 IS NOT NULL
UNION ALL
SELECT MatchID, 'Home', home_player_3 FROM public."Match" WHERE home_player_3 IS NOT NULL
UNION ALL
SELECT MatchID, 'Home', home_player_4 FROM public."Match" WHERE home_player_4 IS NOT NULL
UNION ALL
SELECT MatchID, 'Home', home_player_5 FROM public."Match" WHERE home_player_5 IS NOT NULL
UNION ALL

```
SELECT MatchID, 'Home', home_player_6 FROM public."Match" WHERE home_player_6 IS NOT NULL
UNION ALL
SELECT MatchID, 'Home', home_player_7 FROM public."Match" WHERE home_player_7 IS NOT NULL
UNION ALL
SELECT MatchID, 'Home', home_player_8 FROM public."Match" WHERE home_player_8 IS NOT NULL
UNION ALL
SELECT MatchID, 'Home', home_player_9 FROM public."Match" WHERE home_player_9 IS NOT NULL
UNION ALL
SELECT MatchID, 'Home', home_player_10 FROM public."Match" WHERE home_player_10 IS NOT NULL
UNION ALL
SELECT MatchID, 'Home', home_player_11 FROM public."Match" WHERE home_player_11 IS NOT NULL
UNION ALL
SELECT MatchID, 'Away', away_player_1 FROM public."Match" WHERE away_player_1 IS NOT NULL
UNION ALL
SELECT MatchID, 'Away', away_player_2 FROM public."Match" WHERE away_player_2 IS NOT NULL
UNION ALL
SELECT MatchID, 'Away', away_player_3 FROM public."Match" WHERE away_player_3 IS NOT NULL
UNION ALL
SELECT MatchID, 'Away', away_player_4 FROM public."Match" WHERE away_player_4 IS NOT NULL
UNION ALL
SELECT MatchID, 'Away', away_player_5 FROM public."Match" WHERE away_player_5 IS NOT NULL
UNION ALL
SELECT MatchID, 'Away', away_player_6 FROM public."Match" WHERE away_player_6 IS NOT NULL
UNION ALL
SELECT MatchID, 'Away', away_player_7 FROM public."Match" WHERE away_player_7 IS NOT NULL
UNION ALL
SELECT MatchID, 'Away', away_player_8 FROM public."Match" WHERE away_player_8 IS NOT NULL
UNION ALL
SELECT MatchID, 'Away', away_player_9 FROM public."Match" WHERE away_player_9 IS NOT NULL
UNION ALL
SELECT MatchID, 'Away', away_player_10 FROM public."Match" WHERE away_player_10 IS NOT NULL
UNION ALL
SELECT MatchID, 'Away', away_player_11 FROM public."Match" WHERE away_player_11 IS NOT NULL;
```



Data Warehousing:

An operational database is designed to store, manage and retrieve real-time transactional data. However, it is not powerful enough to handle the complexity and volume of data that can arise from handling a real-time utility management solution. Hence, it requires a database solution that is optimized for complex data analysis, provides scalability for large volumes of data, ensures data quality and consistency, and offers a subject-oriented approach to data storage facilitating sound decision-making. Therefore, we propose a data warehouse design which will act as a central repository of our data from various sources and offer optimized data retrieval and analysis.

Following is the proposed dimensions and measures that will be used in the data warehouse design:

Facts:

- Match

Measure

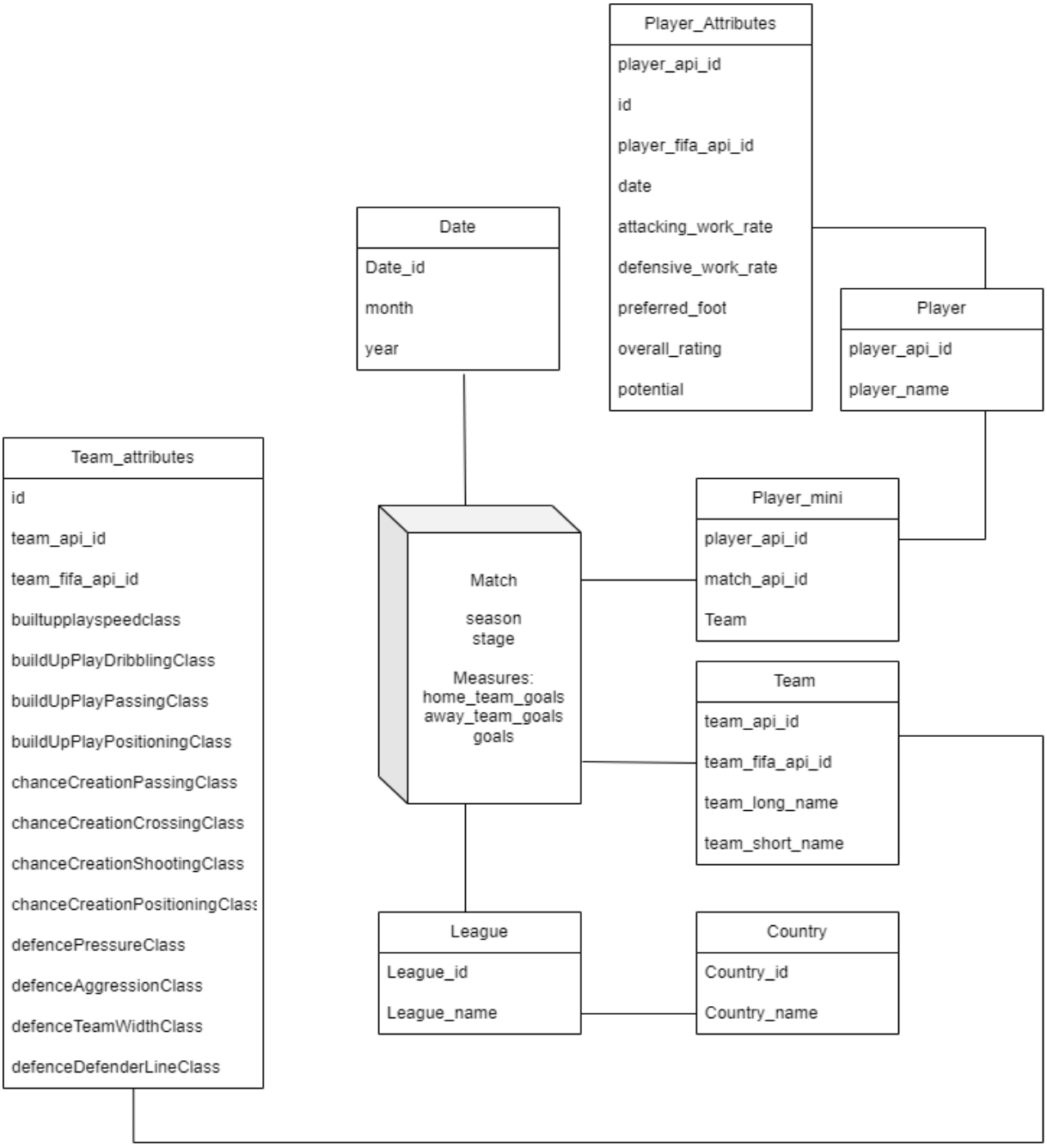
- home_team_goal
- away_team_goal

Dimension:

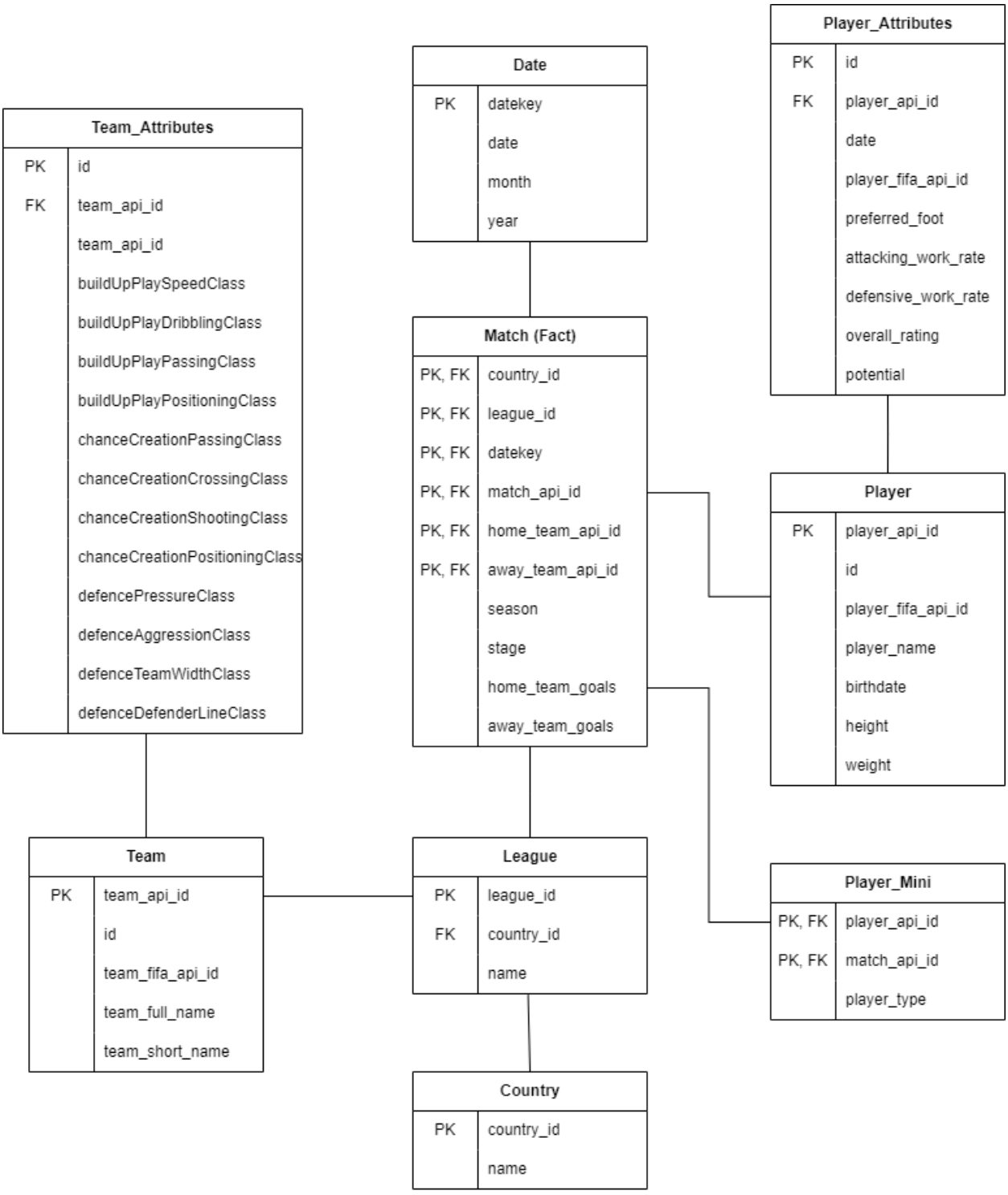
- Player

- Player_mini
- Date
- Player_attributes
- League
- Country
- Team
- Team_Attributes

Conceptual Diagram:



Logical Diagram:



Create Schema:

Country_Dim:

CREATE TABLE IF NOT EXISTS public.country_dim

```
(
    country_id integer NOT NULL,
    name character varying COLLATE pg_catalog."default",
    CONSTRAINT country_dim_pkey PRIMARY KEY (country_id)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.country_dim
    OWNER to postgres;
```

Date_Dim:

```
CREATE TABLE IF NOT EXISTS public.date_dim
(
    datekey integer NOT NULL,
    date integer,
    month integer,
    year integer,
    CONSTRAINT date_dim_pkey PRIMARY KEY (datekey)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.date_dim
    OWNER to postgres;
```

League_Dim:

```
CREATE TABLE IF NOT EXISTS public.league_dim
(
    league_id integer NOT NULL,
    country_id integer,
    name character varying COLLATE pg_catalog."default",
    CONSTRAINT league_dim_pkey PRIMARY KEY (league_id),
    CONSTRAINT yiulk FOREIGN KEY (country_id)
        REFERENCES public.country_dim (country_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.league_dim
    OWNER to postgres;
```

Match_Fact:

```
CREATE TABLE IF NOT EXISTS public.match_fact
(
    id integer,
    country_id integer,
    league_id integer,
    season character varying COLLATE pg_catalog."default",
    stage integer,
    date date,
    match_api_id integer,
    home_api_id integer,
    away_api_id integer,
    home_team_goals integer,
    away_team_goals integer
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.match_fact
    OWNER to postgres;
```

Player_Attributes_Dim:

```
CREATE TABLE IF NOT EXISTS public.player_attributes_dim
(
    id integer NOT NULL,
    player_fifa_api_id integer,
    player_api_id integer,
    date date,
    overall_rating integer,
    potential integer,
    preferred_foot bpchar COLLATE pg_catalog."default" DEFAULT '10'::bpchar,
    attacking_work_rate bpchar COLLATE pg_catalog."default" DEFAULT '20'::bpchar,
    defensive_work_rate bpchar COLLATE pg_catalog."default" DEFAULT '20'::bpchar,
    CONSTRAINT player_attributes_dim_pkey PRIMARY KEY (id)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.player_attributes_dim
    OWNER to postgres;
```

Player_Mini:

```
CREATE TABLE IF NOT EXISTS public.player_mini
```

(
match integer,
player_type character varying COLLATE pg_catalog."default",
player_api_id integer
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.player_mini
OWNER to postgres;

Player_Dim:
CREATE TABLE IF NOT EXISTS public.player_dim
(
id integer,
player_api_id integer,
player_name character varying COLLATE pg_catalog."default",
player_fifa_api_id integer,
birthday date,
height double precision,
weight double precision
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.player_dim
OWNER to postgres;

Team_Attributes_Dim:
CREATE TABLE IF NOT EXISTS public.team_attributes_dim
(
id integer NOT NULL,
team_api_id integer,
datekey integer,
team_fifa_api_id integer,
"buildUpPlaySpeedClass" character varying COLLATE pg_catalog."default",
"buildUpPlayDribblingClass" character varying COLLATE pg_catalog."default",
"buildUpPlayPassingClass" character varying COLLATE pg_catalog."default",
"buildUpPlayPositioningClass" character varying COLLATE pg_catalog."default",
"chanceCreationPassingClass" character varying COLLATE pg_catalog."default",
"chanceCreationCrossingClass" character varying COLLATE pg_catalog."default",
"chanceCreationShootingClass" character varying COLLATE pg_catalog."default",
"chanceCreationPositioningClass" character varying COLLATE pg_catalog."default",
"defencePressureClass" character varying COLLATE pg_catalog."default",

```
    "defenceAggressionClass" character varying COLLATE pg_catalog."default",
    "defenceTeamWidthClass" character varying COLLATE pg_catalog."default",
    "defenceDefenderLineClass" character varying COLLATE pg_catalog."default",
    CONSTRAINT team_attributes_dim_pkey PRIMARY KEY (id),
    CONSTRAINT njkhbnjk FOREIGN KEY (datekey)
        REFERENCES public.date_dim (datekey) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT uik FOREIGN KEY (team_api_id)
        REFERENCES public.team_dim (team_api_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.team_attributes_dim
    OWNER to postgres;
```

Team_Dim

```
CREATE TABLE IF NOT EXISTS public.team_dim
(
    team_api_id integer NOT NULL,
    team_id integer,
    team_fifa_api_id integer,
    team_full_name character varying COLLATE pg_catalog."default",
    team_short_name character varying COLLATE pg_catalog."default",
    CONSTRAINT team_dim_pkey PRIMARY KEY (team_api_id)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.team_dim
    OWNER to postgres;
```

ROLAP Operations:

1. Rolls up the total goals scored from matches to the team level to the country level, and returns the 3 countries with the max total goals.

   Goals < ROLLUP(Matches, Team --> Country, SUM(Goals))
   Result & MAX(Goals, Goals, 3)

2. Rolls up average player ratings by league and returns the top 10 leagues by average player rating.

   Player_Ratings < ROLLUP(Players, Team > League, AVG(Overall_Rating))
   Result & TOP(Player_Ratings, Rating, 10, DESC)

3. Rolls up home and away goals scored by country and returns the 5 countries with the most total goals.
   Goals < ROLLUP(Matches, Team > Country, SUM(Home_Goals), SUM(Away_Goals))
   Result & MAX(Goals, Total_Goals, 5)

4. Rolls up average player height and weight by league and ranks the leagues with the tallest players.

   Player_Size < ROLLUP(Players, Team > League, AVG(Height), AVG(Weight))
   Result & RANK(Player_Size, Height, 10)

5. Rolls up average potential by league and nationality and returns the bottom 5 leagues by average player potential.
   Potential < ROLLUP(Players, Nationality > League, AVG(Potential))
   Result & BOTTOM(Potential, Avg_Potential, 5)

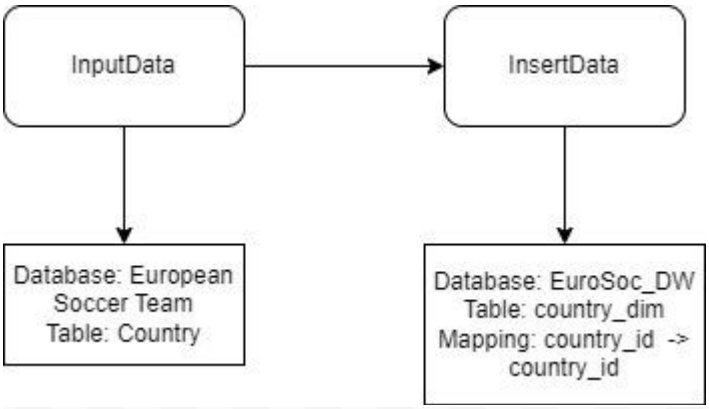
6. Rolls up average player market value by position and league and calculates 3-year growth for average value.

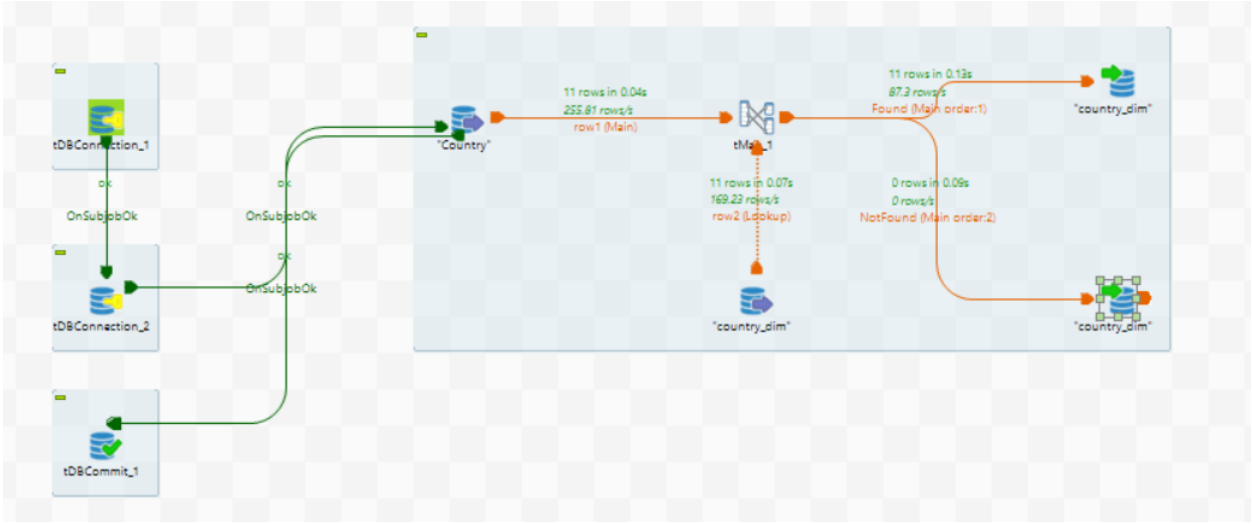
   Player_Values < ROLLUP(Players, Position > League, AVG(Market_Value))
   Result & Growth(Player_Values, Avg_Value, 3, DESC)

7. Rolls up home and away goals scored by player and returns the top 10 goal scorers
   Goals < ROLLUP(Matches, Team > Player, SUM(Home_Goals), SUM(Away_Goals))
   Result & TOP(Goals, Total_Goals, 10, DESC)

Flowchart and ETL:

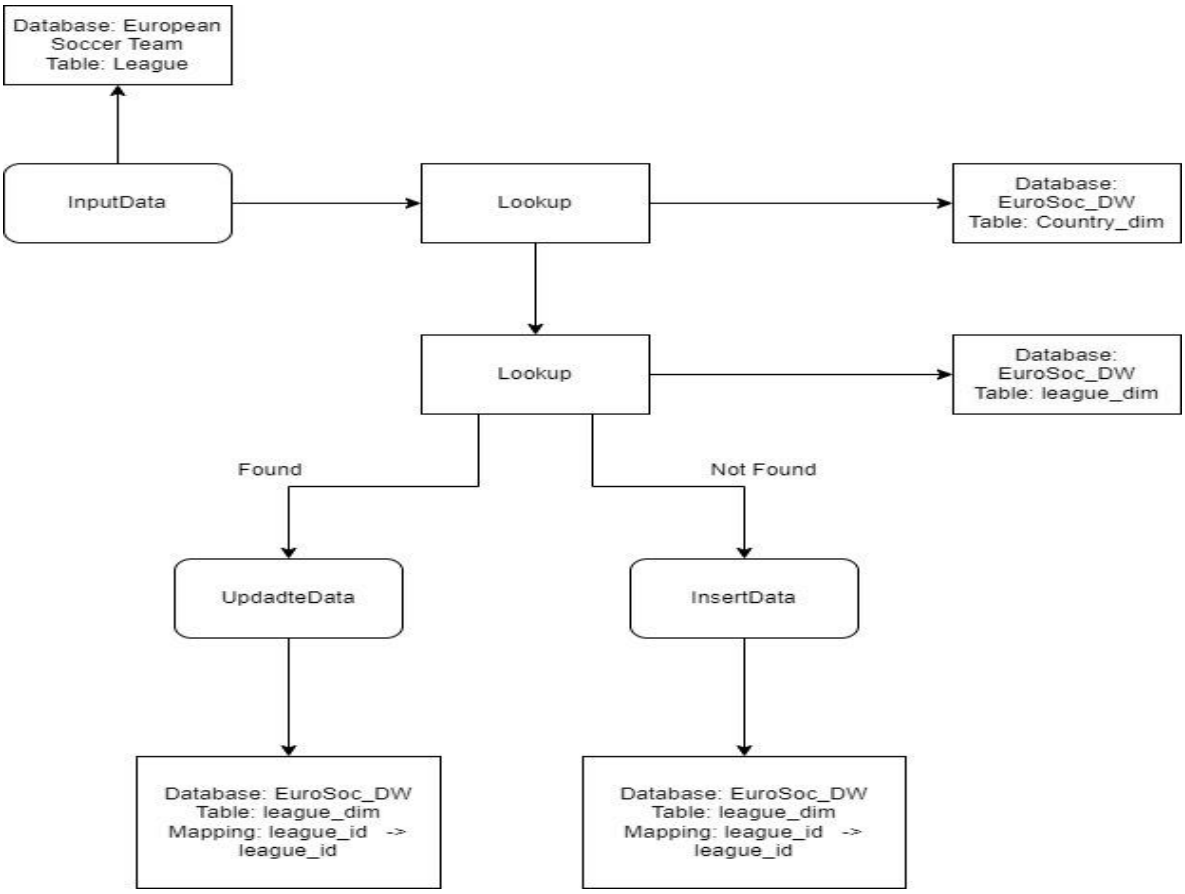
Flowchart and Loading the Country Dimension:

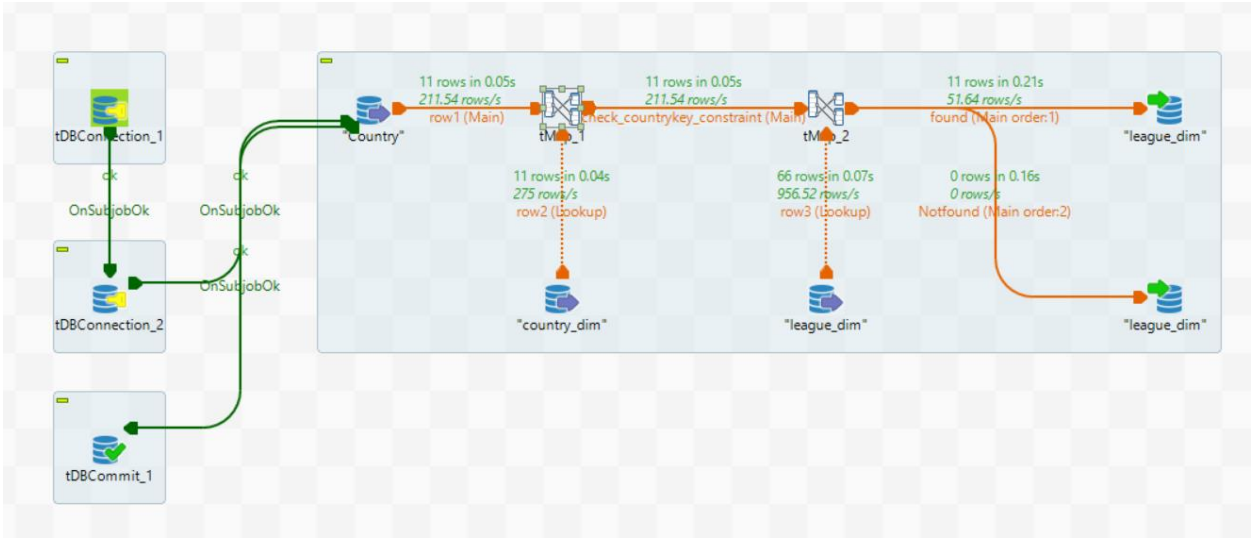




In the country dimension we used Type 1 Slowly Changing dimensions to enable simple overwrite data in dimension. When the country record is initiated, we may not get all attributes. Therefore, when the country record is initiated at the operational database, there will be null values in the record. Once ETL is executed, those null values will be created in the data warehouse. Once these attributes are filled in the operating database, it must be updated in the data warehouse.

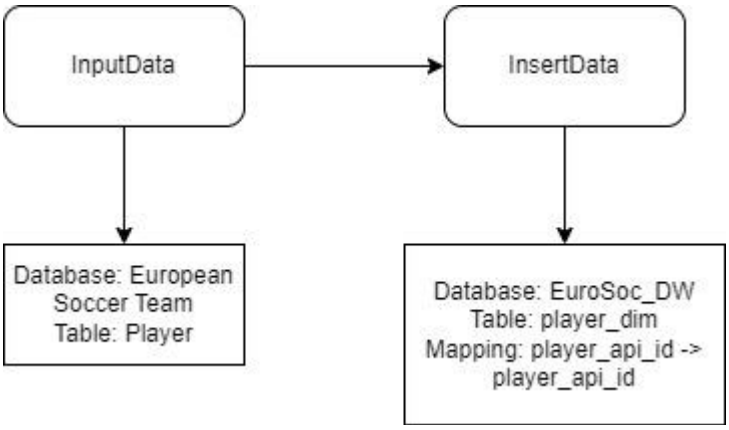
Flowchart and Loading the League Dimension:

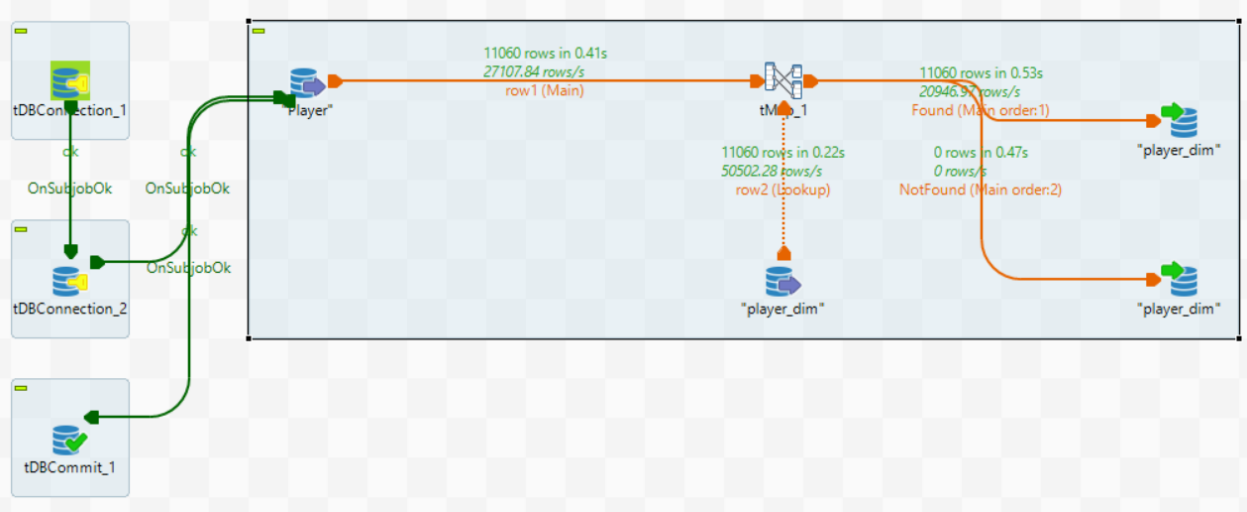




In the league dimension, we look for the country_id and then proceed to insert data in thew league dimension table.

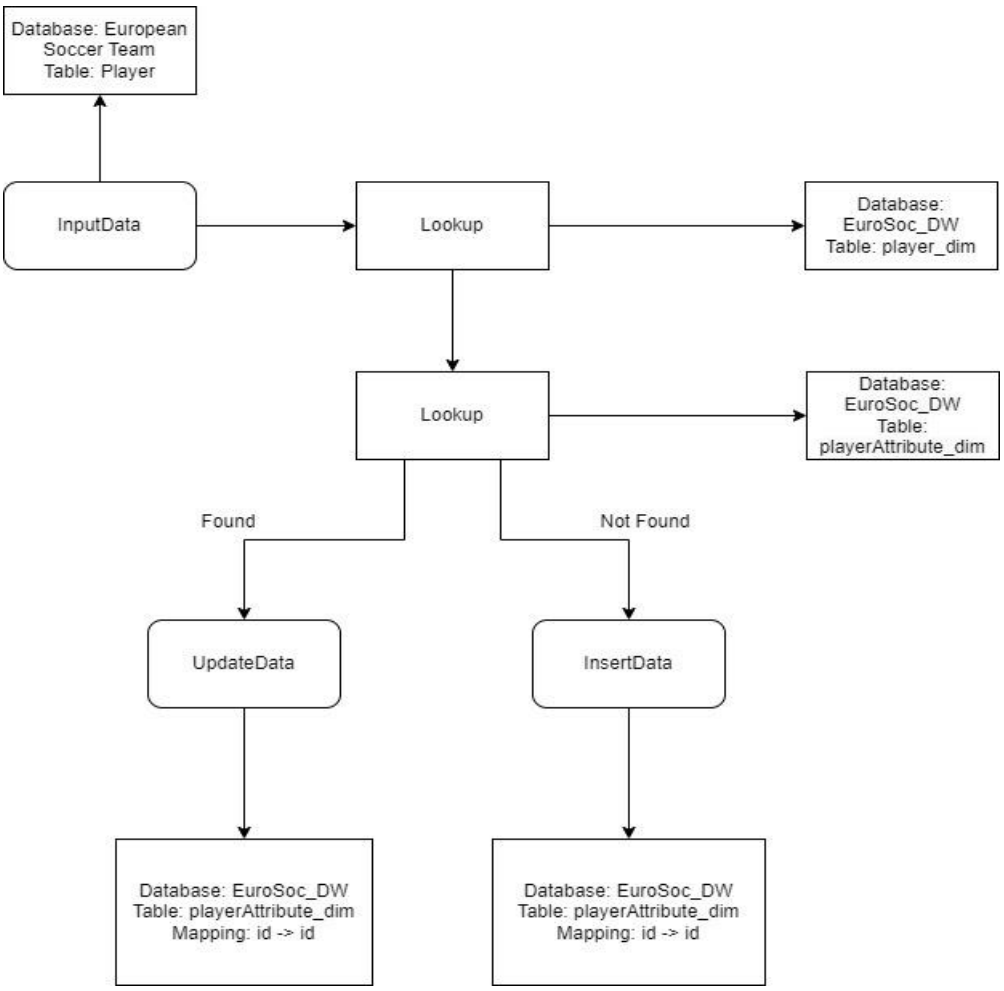
Flowchart and Loading the Player Dimension:

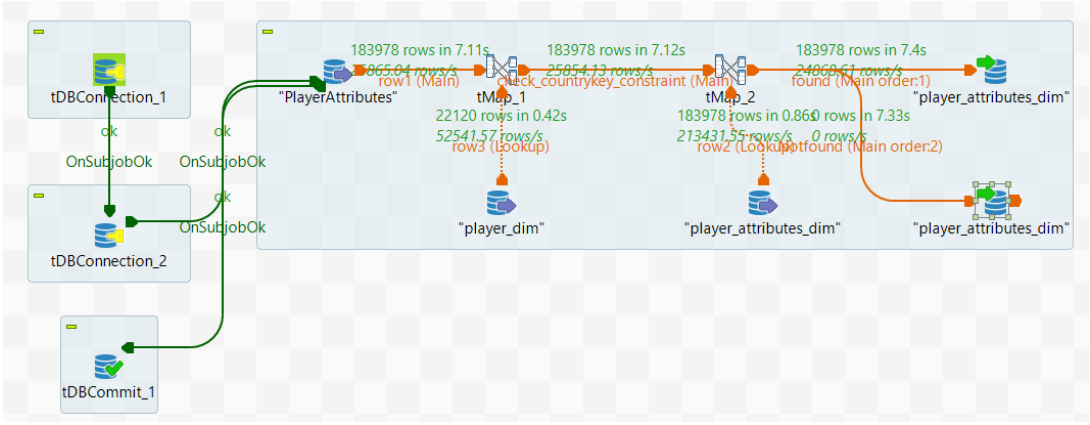




In the player dimension, we look for the player_api_id and then proceed to insert data in thew player dimension table.

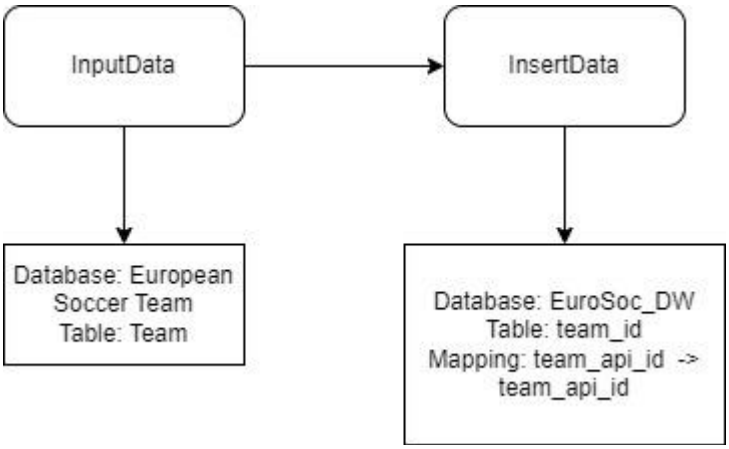
Flowchart and Loading the Player_Attributes Dimension:

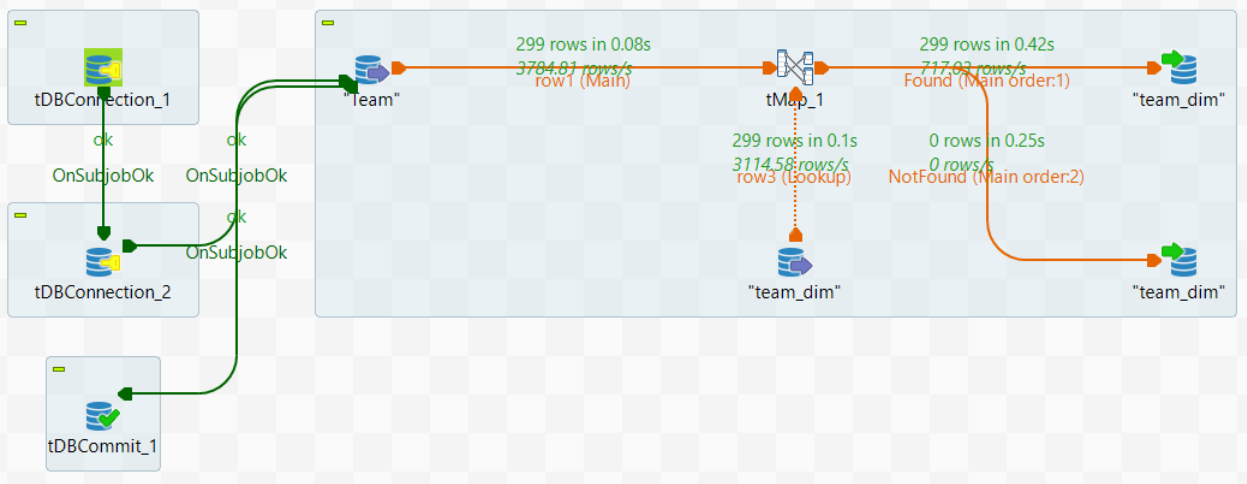




In the player attribute dimension, we look for the player_api_id and then check for id and then proceed to insert data in thew player attribute dimension table.

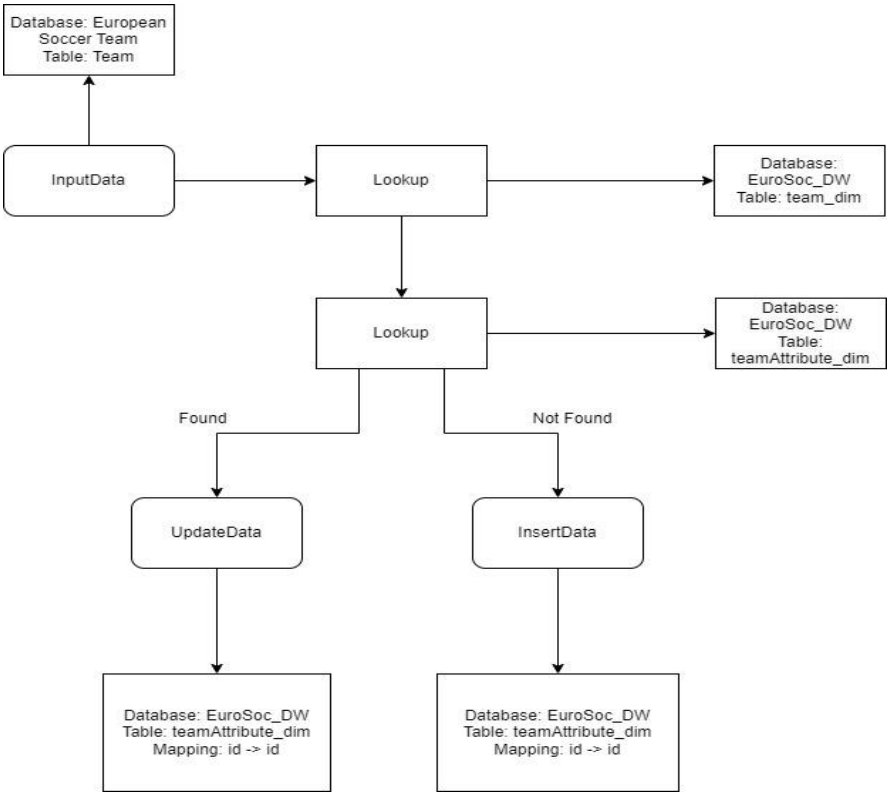
Flowchart and Loading the Team Dimension:

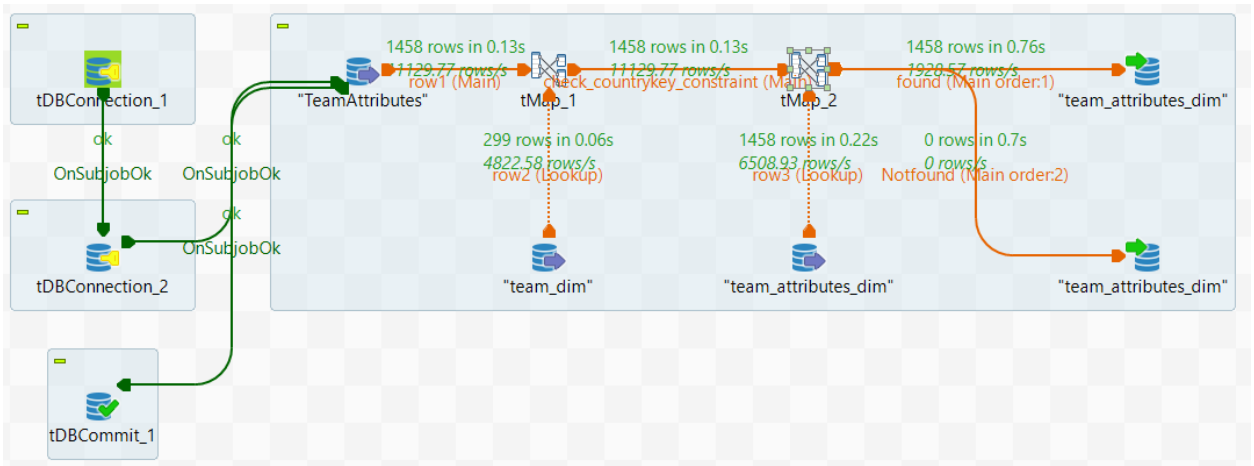




In the Team dimension, we look for the team_api_id and then proceed to insert data in thew league dimension table.

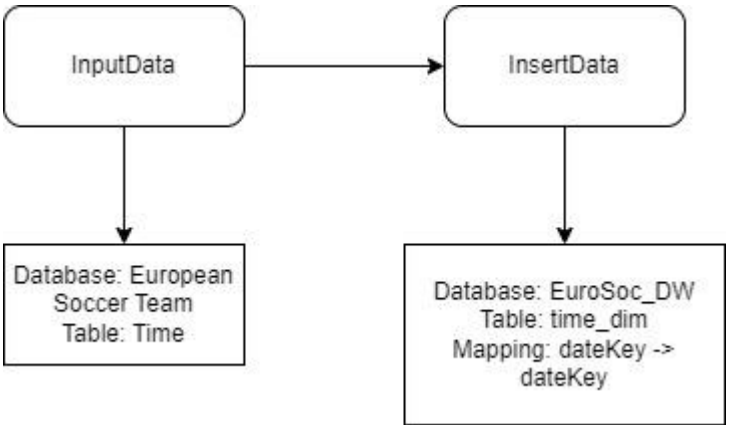
Flowchart and Loading the Team_Attributes Dimension:

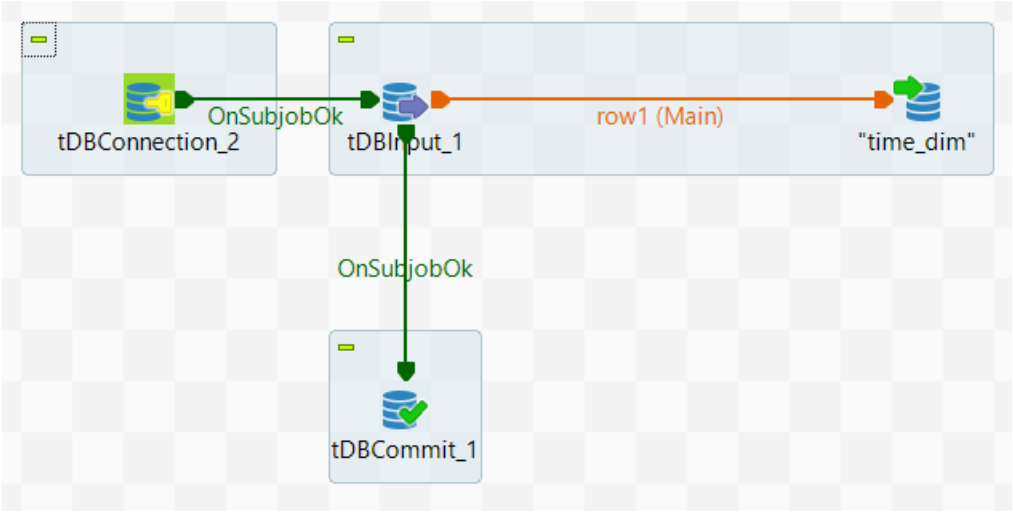




In the team attribute dimension, we look for the team_api_id and then check for id and then proceed to insert data in thew team attribute dimension table.

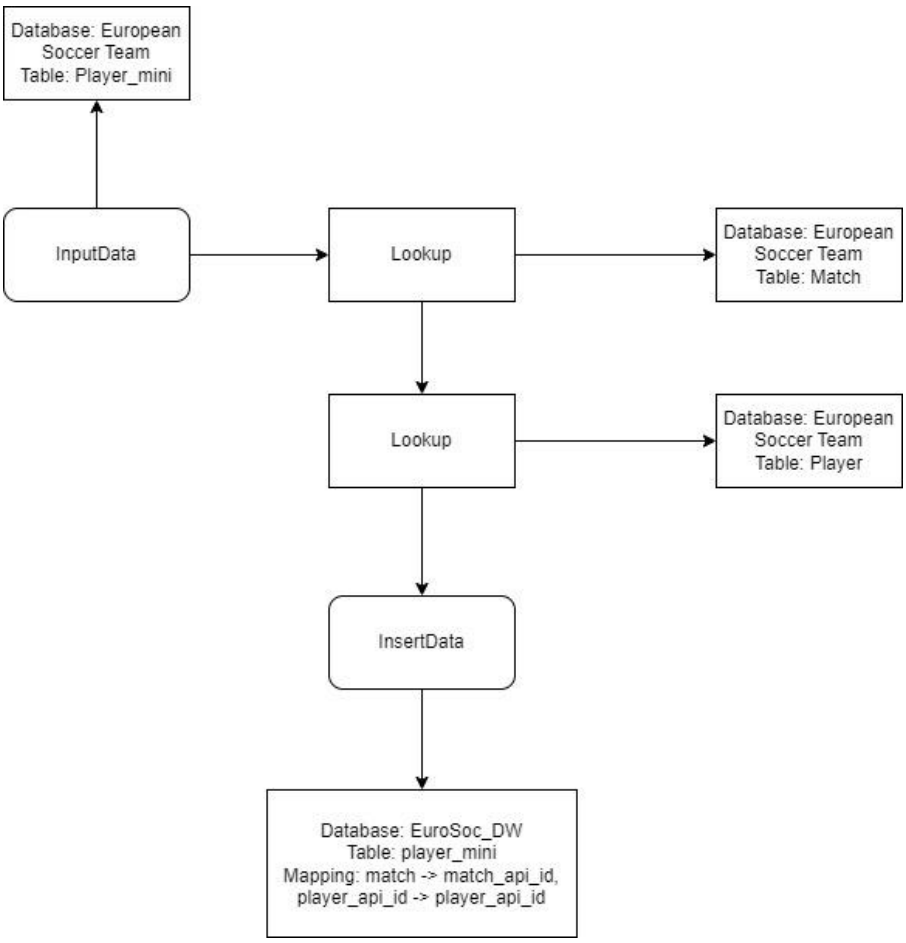
Flowchart and Loading the Time Dimension:

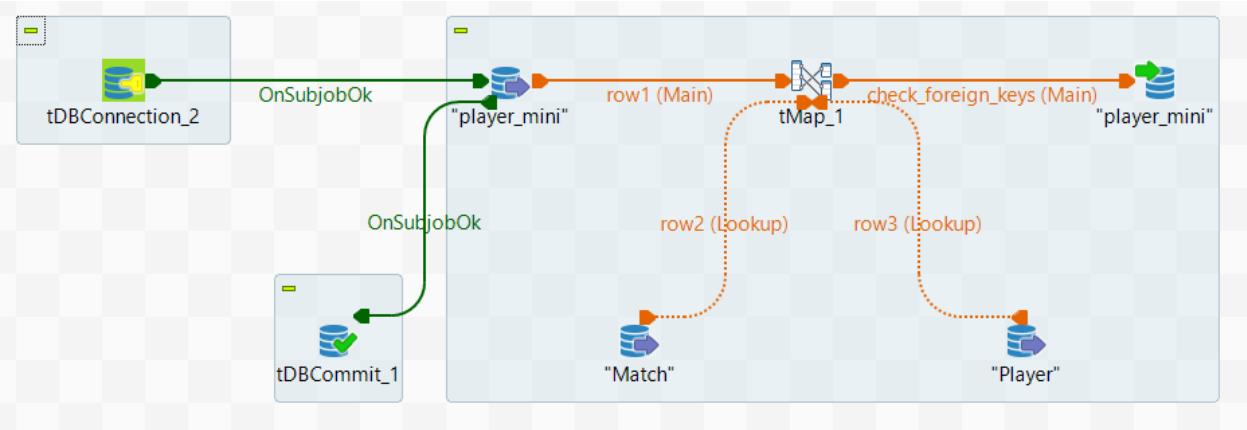




In the above ETL we extracted and loaded the data from the match table and transferred it to the time dimension in data warehouse after matching Date Key.

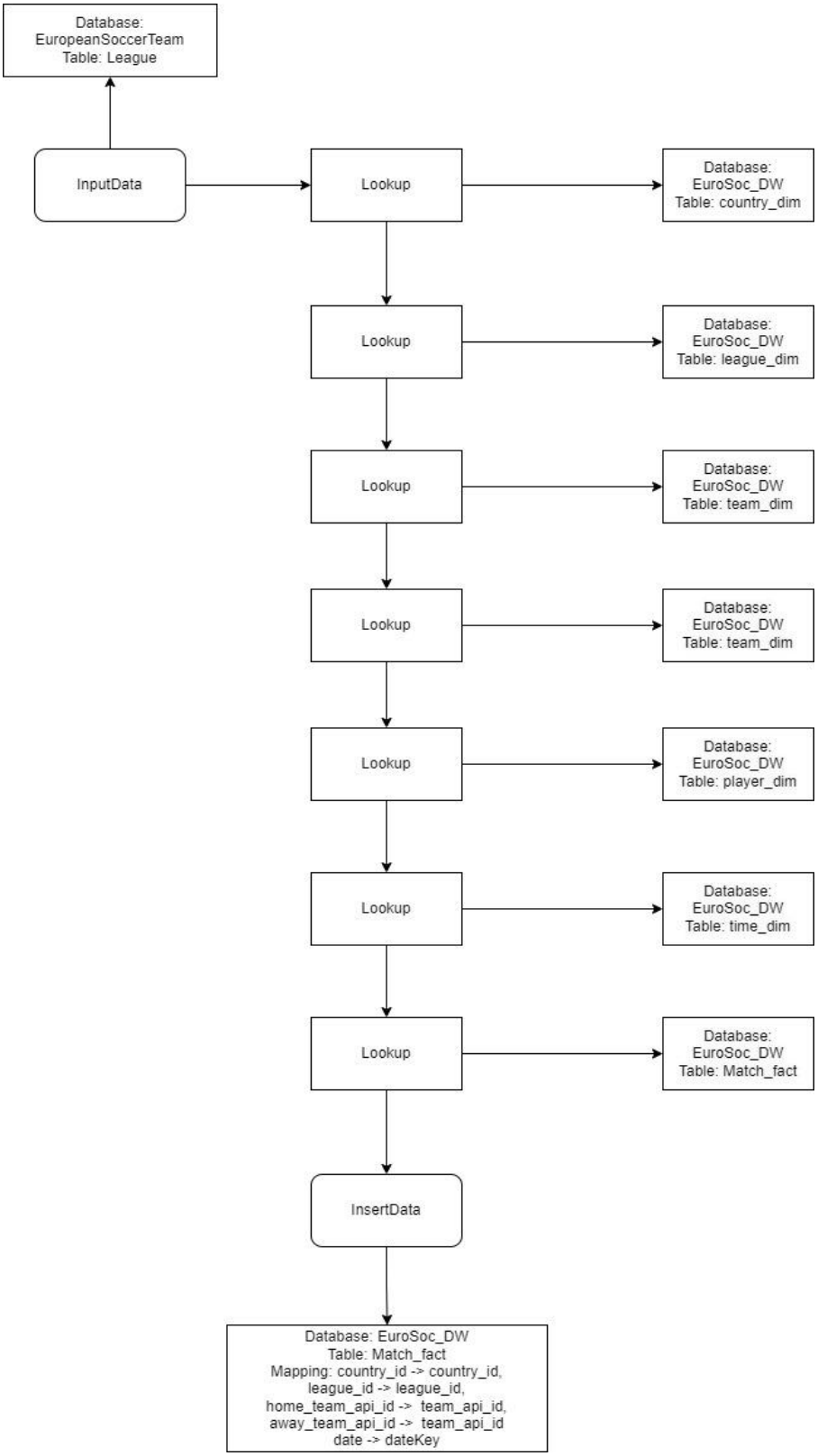
Flowchart and Loading the Player_Mini Dimension:

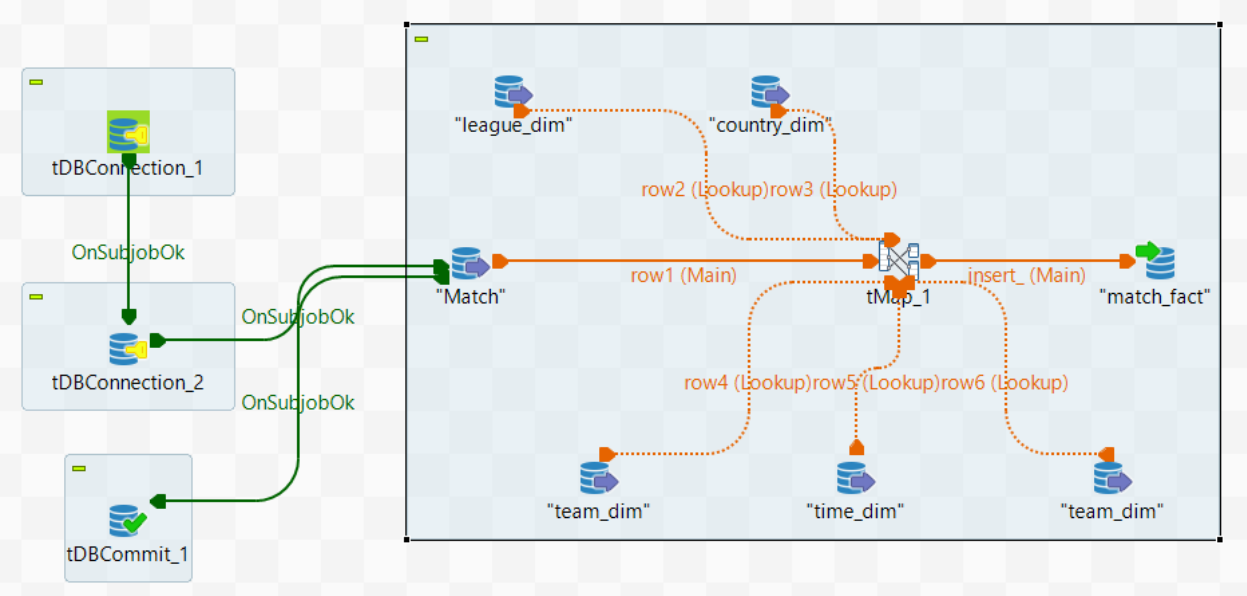




We have 22 players, 11 players from the home team and 11 from away team. We created a bridge table between the match and the players. This was preprocessed externally and loaded in player_mini. In the player mini dimension, we look for the match_api_id and player_api_id and then proceed to insert data in thew player mini dimension table.

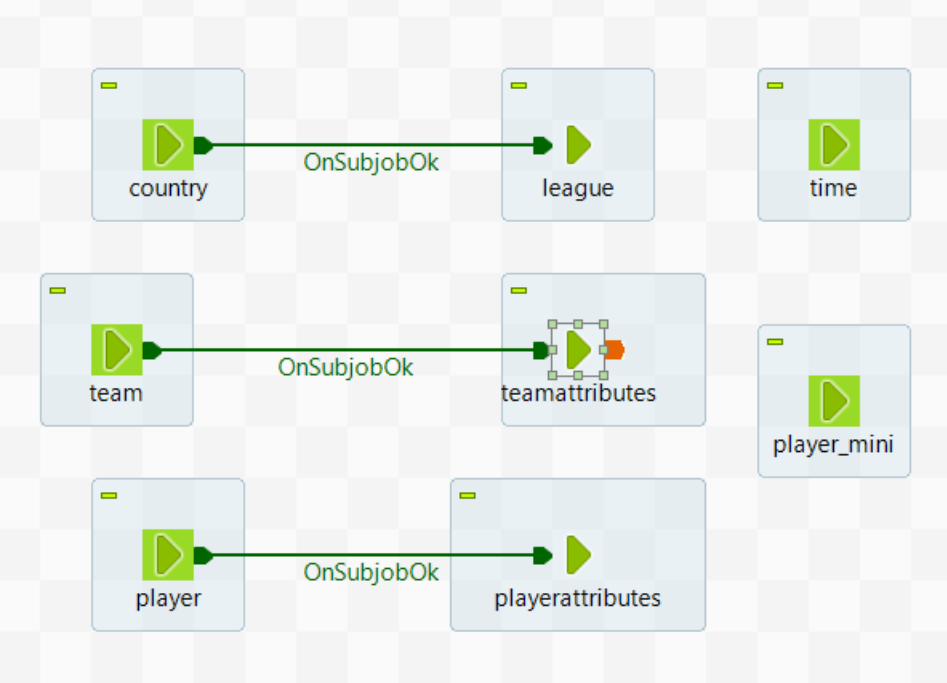
Flowchart and Loading the Match Fact:

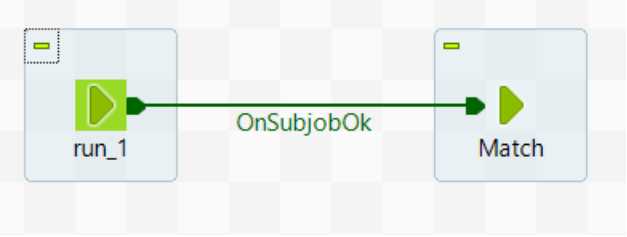

Database:
EuropeanSoccerTeam
Table: League
InputData
Lookup
Database:
EuroSoc_DW
Table: country_dim
Lookup
Database:
EuroSoc_DW
Table: league_dim
Lookup
Database:
EuroSoc_DW
Table: team_dim
Lookup
Database:
EuroSoc_DW
Table: team_dim
Lookup
Database:
EuroSoc_DW
Table: player_dim
Lookup
Database:
EuroSoc_DW
Table: time_dim
Lookup
Database:
EuroSoc_DW
Table: Match_fact
InsertData
Database: EuroSoc_DW
Table: Match_fact
Mapping: country_id -> country_id,
league_id -> league_id,
home_team_api_id -> team_api_id,
away_team_api_id -> team_api_id
date -> dateKey



This is our fact table. It counts for goals scored for the home and away team goals.

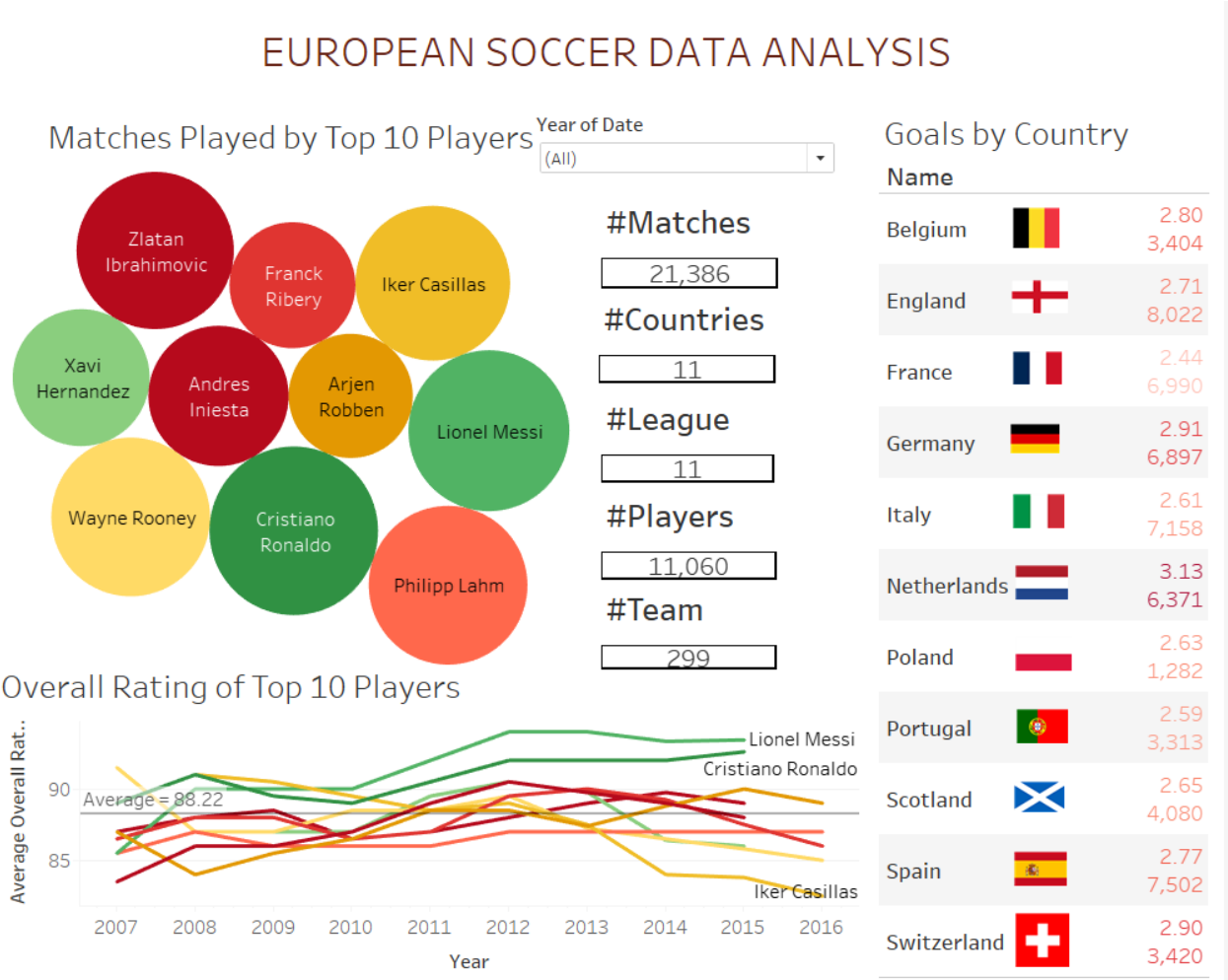
Complete Run:





This is the complete run where run_1 runs all dimensions. After loading all the dimensions, it then proceeds to populate the match fact table.

Dashboard:



After completing Talend jobs, Once the data is pushed successfully to the Data warehouse in Postgres, It is connected to the Tableau to perform further analysis.

1. First filtering the top 10 players who have played the most matches from 2008-2016

2. Choosing those top 10 players gives their average overall rating across years for all the matches they players

3. The figures displayed gives the count of the respective attribute

4. The number of goals scored in each country with their respective averages