Barclays Premier League

Statistics & Rankings Database

Designed By: Matt Pineau

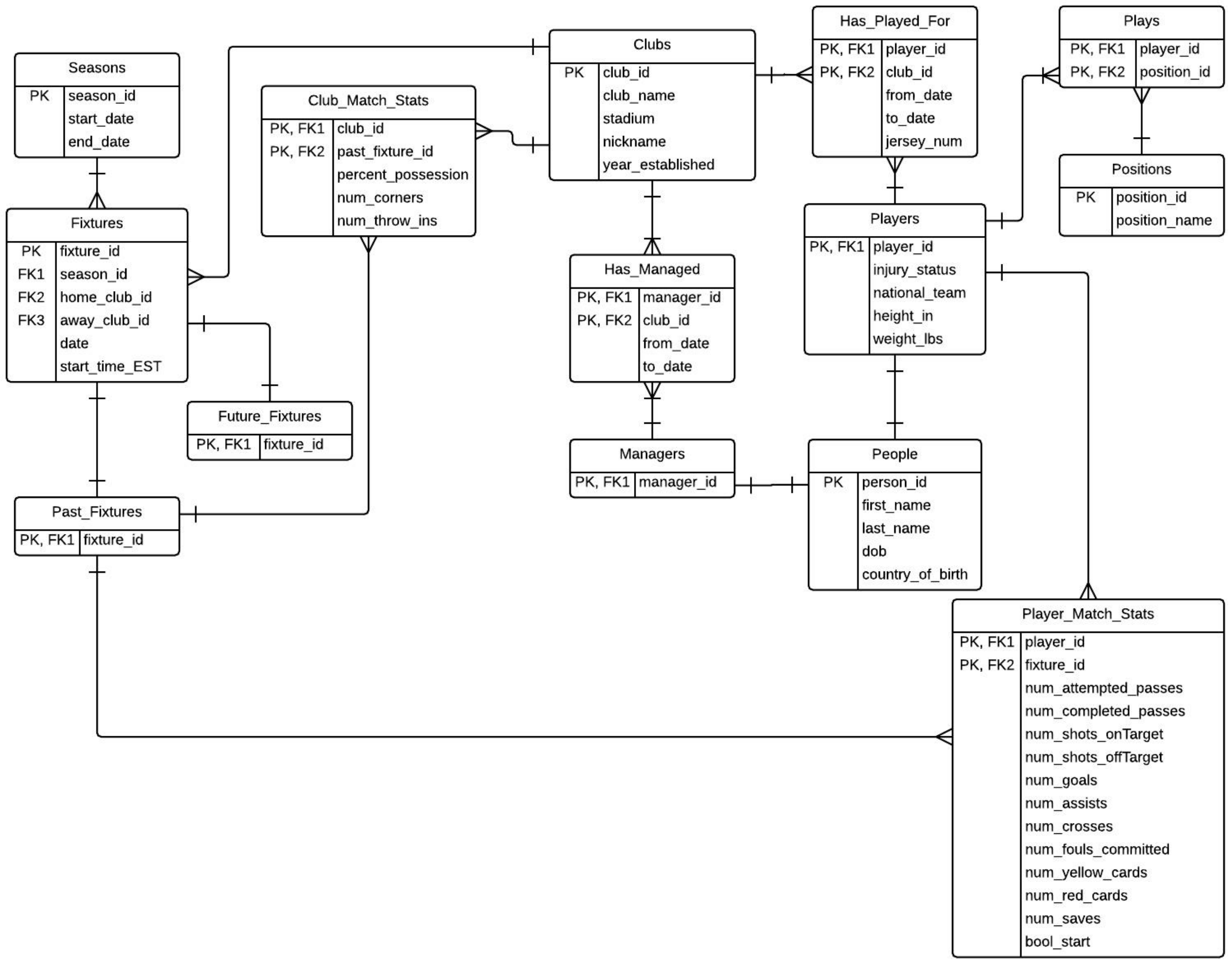
Table of Contents

Executive Summary:

This database is designed to keep a record of individual players' statistics, teams' statistics, and the current rankings of teams in the Barclays Premier League. This database design can also easily be altered for use in other soccer (football) leagues, as long as they use the same three-points per win scoring system as the Premier League.

The database is intended to be used by fans that wish to keep track of their favorite teams and players as they compete in the Barclays Premier League or look back on past seasons. This document includes an entity-relationship diagram of this database, information about the tables that comprise this database (create statements, functional dependencies, sample data), security permissions for different users, and useful views, reports, stored procedures, and triggers.

Entity-Relationship Diagram:



Tables:

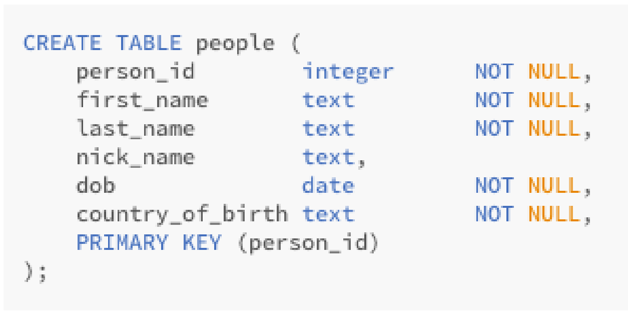
<u>**People Table**</u>

The people table holds basic information about the people that will be included in the database. The fields in this table will be inherited by its subtables (players, managers).

Functional Dependencies:

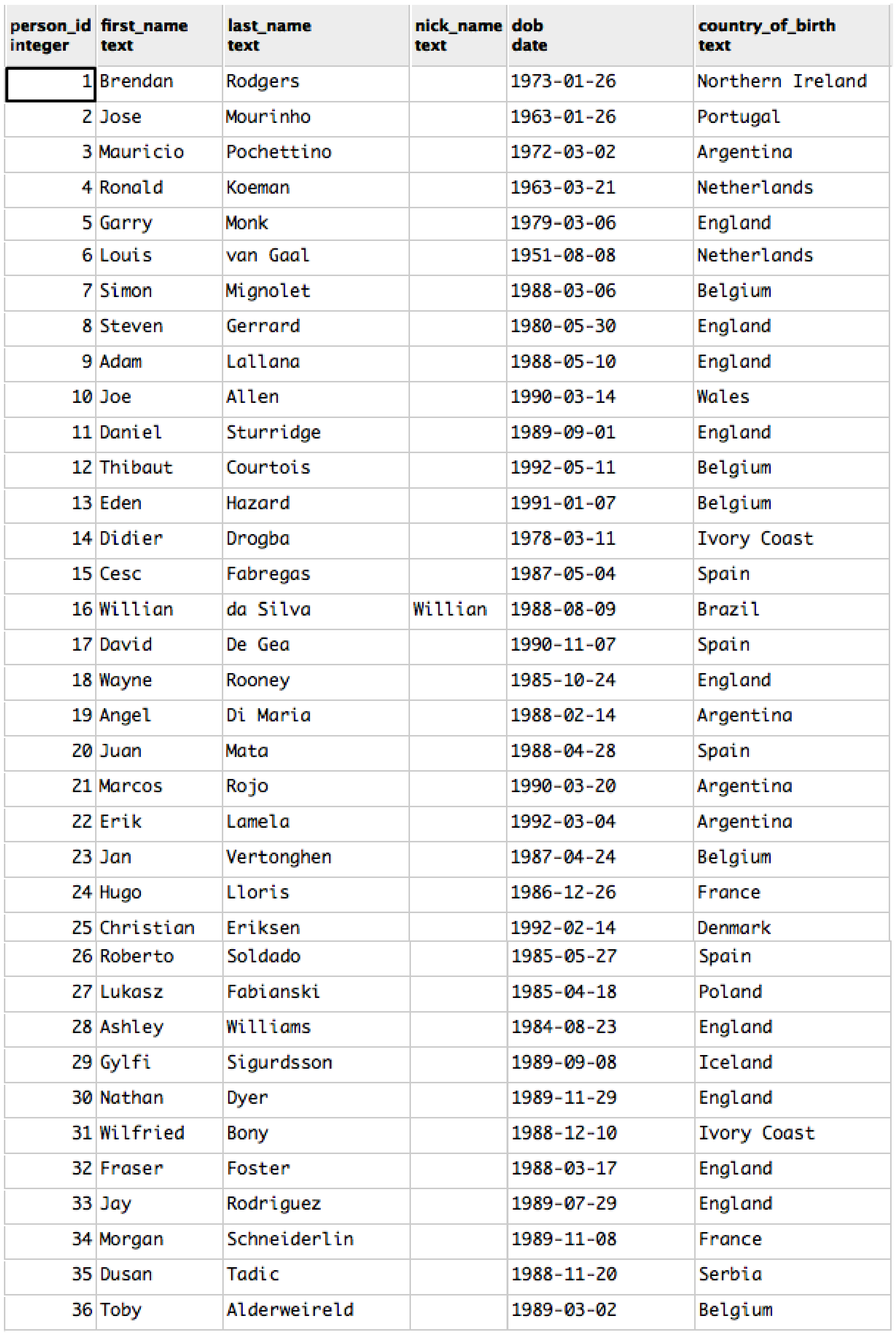
person_id → first_name, last_name, nick_name, dob, country_of_birth

Create Statement:



```
CREATE TABLE people (
    person_id        integer      NOT NULL,
    first_name       text         NOT NULL,
    last_name        text         NOT NULL,
    nick_name        text,
    dob              date         NOT NULL,
    country_of_birth text         NOT NULL,
    PRIMARY KEY (person_id)
);
```

Sample Data:



person_id integer	first_name text	last_name text	nick_name text	dob date	country_of_birth text
1	Brendan	Rodgers		1973-01-26	Northern Ireland
2	Jose	Mourinho		1963-01-26	Portugal
3	Mauricio	Pochettino		1972-03-02	Argentina
4	Ronald	Koeman		1963-03-21	Netherlands
5	Garry	Monk		1979-03-06	England
6	Louis	van Gaal		1951-08-08	Netherlands
7	Simon	Mignolet		1988-03-06	Belgium
8	Steven	Gerrard		1980-05-30	England
9	Adam	Lallana		1988-05-10	England
10	Joe	Allen		1990-03-14	Wales
11	Daniel	Sturridge		1989-09-01	England
12	Thibaut	Courtois		1992-05-11	Belgium
13	Eden	Hazard		1991-01-07	Belgium
14	Didier	Drogba		1978-03-11	Ivory Coast
15	Cesc	Fabregas		1987-05-04	Spain
16	Willian	da Silva	Willian	1988-08-09	Brazil
17	David	De Gea		1990-11-07	Spain
18	Wayne	Rooney		1985-10-24	England
19	Angel	Di Maria		1988-02-14	Argentina
20	Juan	Mata		1988-04-28	Spain
21	Marcos	Rojo		1990-03-20	Argentina
22	Erik	Lamela		1992-03-04	Argentina
23	Jan	Vertonghen		1987-04-24	Belgium
24	Hugo	Lloris		1986-12-26	France
25	Christian	Eriksen		1992-02-14	Denmark
26	Roberto	Soldado		1985-05-27	Spain
27	Lukasz	Fabianski		1985-04-18	Poland
28	Ashley	Williams		1984-08-23	England
29	Gylfi	Sigurdsson		1989-09-08	Iceland
30	Nathan	Dyer		1989-11-29	England
31	Wilfried	Bony		1988-12-10	Ivory Coast
32	Fraser	Foster		1988-03-17	England
33	Jay	Rodriguez		1989-07-29	England
34	Morgan	Schneiderlin		1989-11-08	France
35	Dusan	Tadic		1988-11-20	Serbia
36	Toby	Alderweireld		1989-03-02	Belgium

Players Table

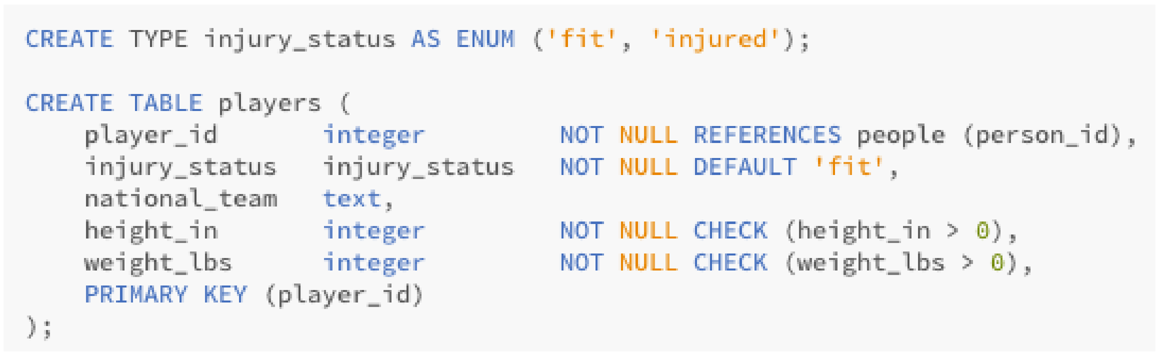
The players table is an extension of the people table. It specifies that players are people, but also have some extra characteristics.

Functional Dependencies:

player_id → injury_status, national_team, height_in, weight_lbs

Create Statement:

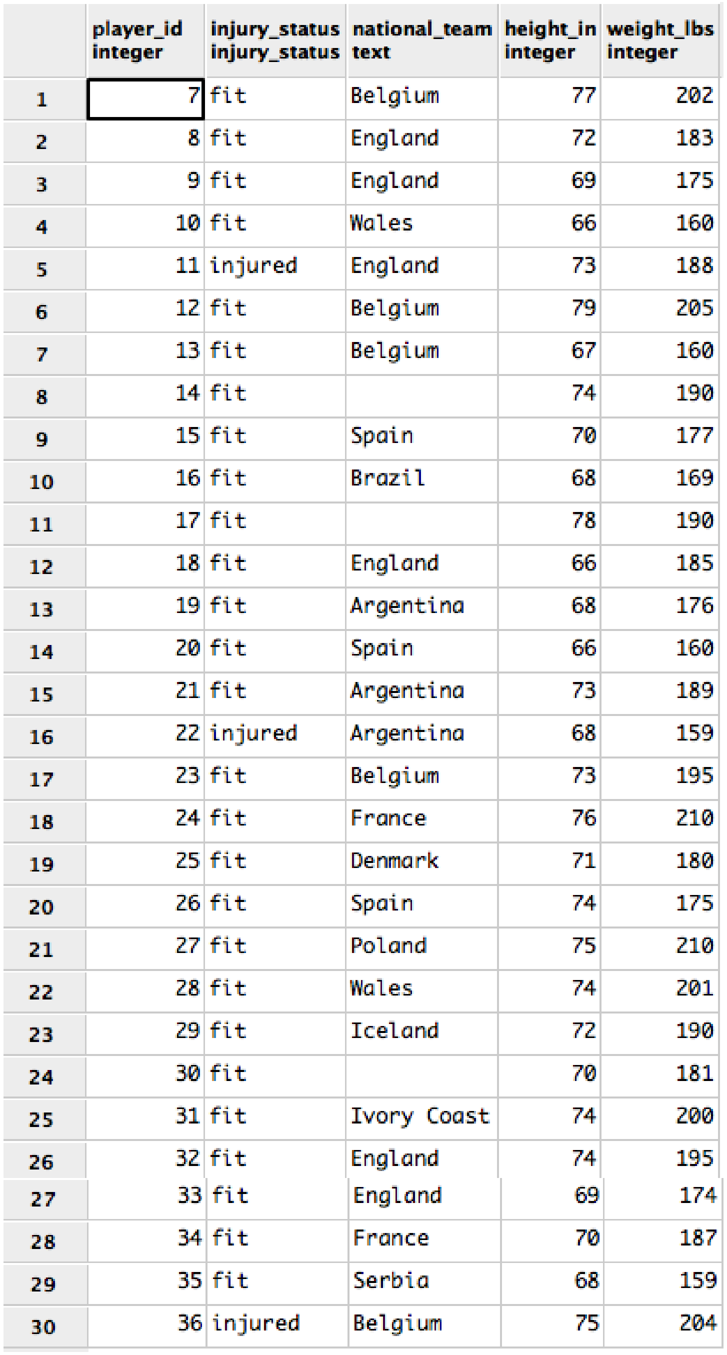
Note: Above the create statement for the players table, an enumerated type injury_status is created to specify that a player can only be 'fit' or 'injured'.



```
CREATE TYPE injury_status AS ENUM ('fit', 'injured');

CREATE TABLE players (
    player_id       integer         NOT NULL REFERENCES people (person_id),
    injury_status   injury_status   NOT NULL DEFAULT 'fit',
    national_team   text,
    height_in       integer         NOT NULL CHECK (height_in > 0),
    weight_lbs      integer         NOT NULL CHECK (weight_lbs > 0),
    PRIMARY KEY (player_id)
);
```

Sample Data:



	player_id integer	injury_status injury_status	national_team text	height_in integer	weight_lbs integer
1	7	fit	Belgium	77	202
2	8	fit	England	72	183
3	9	fit	England	69	175
4	10	fit	Wales	66	160
5	11	injured	England	73	188
6	12	fit	Belgium	79	205
7	13	fit	Belgium	67	160
8	14	fit		74	190
9	15	fit	Spain	70	177
10	16	fit	Brazil	68	169
11	17	fit		78	190
12	18	fit	England	66	185
13	19	fit	Argentina	68	176
14	20	fit	Spain	66	160
15	21	fit	Argentina	73	189
16	22	injured	Argentina	68	159
17	23	fit	Belgium	73	195
18	24	fit	France	76	210
19	25	fit	Denmark	71	180
20	26	fit	Spain	74	175
21	27	fit	Poland	75	210
22	28	fit	Wales	74	201
23	29	fit	Iceland	72	190
24	30	fit		70	181
25	31	fit	Ivory Coast	74	200
26	32	fit	England	74	195
27	33	fit	England	69	174
28	34	fit	France	70	187
29	35	fit	Serbia	68	159
30	36	injured	Belgium	75	204

<u>Managers Table</u>

The managers table is an extension of the people table. It specifies that a manager is a person, but must be specified as a manager so they can be connected to a club that they manage.

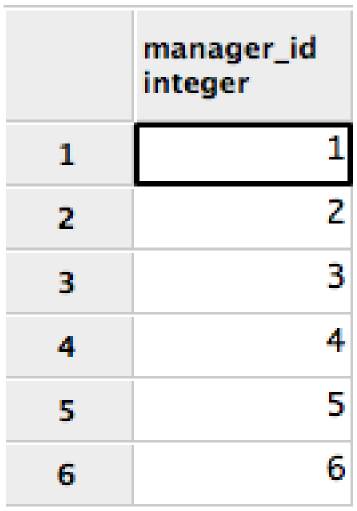
Functional Dependencies:

None

Create Statement:

```
CREATE TABLE managers (
    manager_id integer      NOT NULL REFERENCES people (person_id),
    PRIMARY KEY (manager_id)
);
```

Sample Data:



	manager_id integer
1	1
2	2
3	3
4	4
5	5
6	6

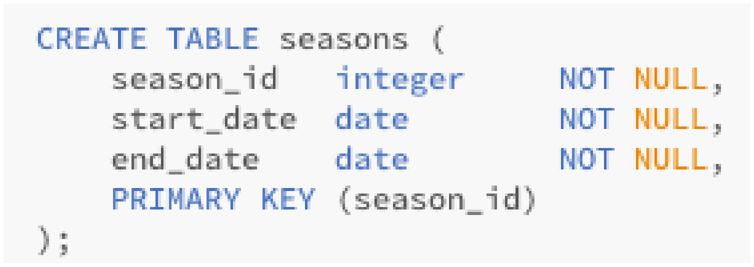
Seasons Table

The seasons table is used to keep track of not only the data of the current season, but all seasons prior.

Functional Dependencies:

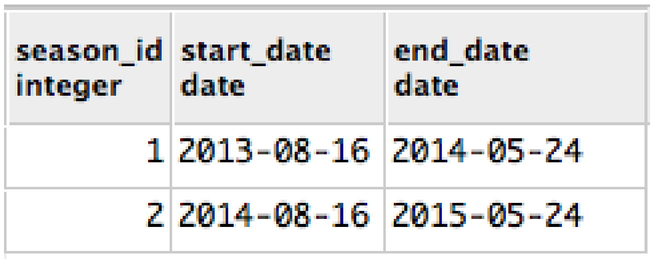
season_id → start_date, end_date

Create Statement:



```
CREATE TABLE seasons (
    season_id   integer     NOT NULL,
    start_date  date        NOT NULL,
    end_date    date        NOT NULL,
    PRIMARY KEY (season_id)
);
```

Sample Data:



season_id integer	start_date date	end_date date
1	2013-08-16	2014-05-24
2	2014-08-16	2015-05-24

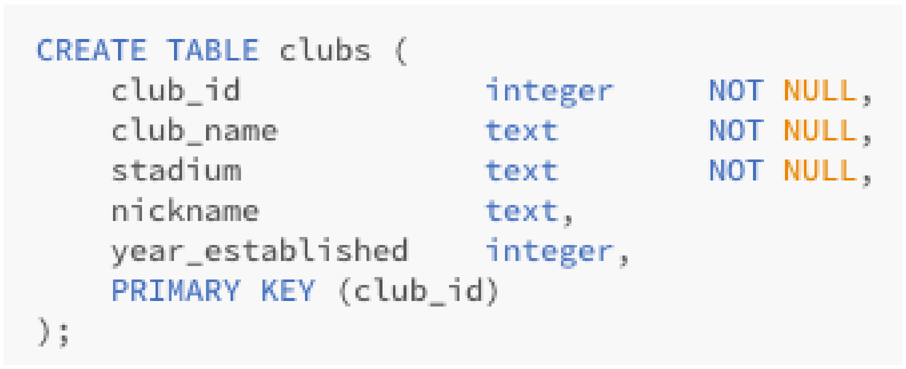
Clubs Table

The clubs table contains data about all of the clubs that participate in the league.

Functional Dependencies:

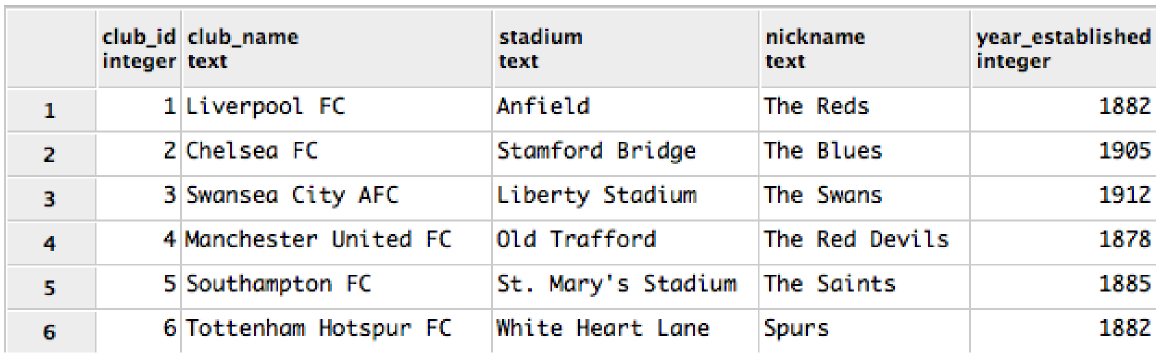
club_id $\rightarrow$ club_name, stadium, nickname, year_established

Create Statement:



```
CREATE TABLE clubs (
    club_id             integer     NOT NULL,
    club_name           text        NOT NULL,
    stadium             text        NOT NULL,
    nickname            text,
    year_established    integer,
    PRIMARY KEY (club_id)
);
```

Sample Data:



	club_id integer	club_name text	stadium text	nickname text	year_established integer
1	1	Liverpool FC	Anfield	The Reds	1882
2	2	Chelsea FC	Stamford Bridge	The Blues	1905
3	3	Swansea City AFC	Liberty Stadium	The Swans	1912
4	4	Manchester United FC	Old Trafford	The Red Devils	1878
5	5	Southampton FC	St. Mary's Stadium	The Saints	1885
6	6	Tottenham Hotspur FC	White Heart Lane	Spurs	1882

Fixtures Table

A fixture is a sort of European soccer term for a match. The fixtures table in this database keeps track of general information about a fixture. It will then be broken down into two subtables: past_fixtures and future_fixtures.

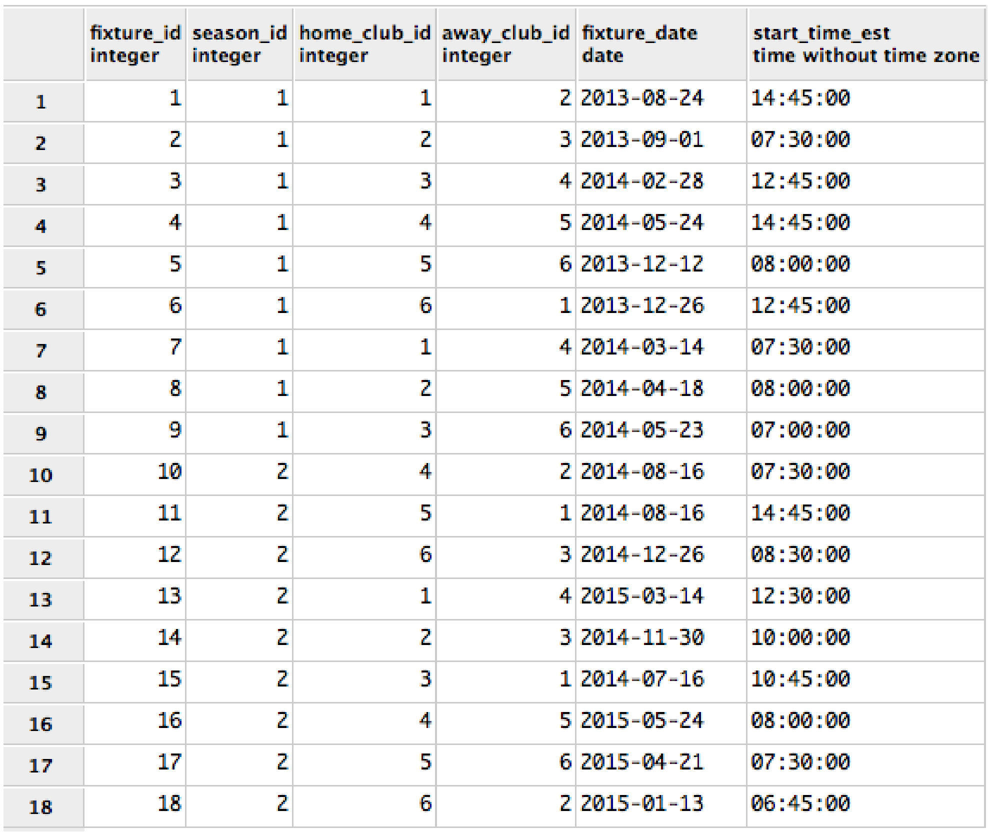
Functional Dependencies:

fixture_id → season_id, home_club_id, away_club_id, fixture_date, start_time_EST

Create Statement:

```
CREATE TABLE fixtures (
    fixture_id      integer     NOT NULL,
    season_id       integer     NOT NULL REFERENCES seasons (season_id),
    home_club_id    integer     NOT NULL REFERENCES clubs (club_id),
    away_club_id    integer     NOT NULL REFERENCES clubs (club_id),
    fixture_date    date        NOT NULL,
    start_time_EST  time        NOT NULL,
    PRIMARY KEY (fixture_id)
);
```

Sample Data:



	fixture_id integer	season_id integer	home_club_id integer	away_club_id integer	fixture_date date	start_time_est time without time zone
1	1	1	1	2	2013-08-24	14:45:00
2	2	1	2	3	2013-09-01	07:30:00
3	3	1	3	4	2014-02-28	12:45:00
4	4	1	4	5	2014-05-24	14:45:00
5	5	1	5	6	2013-12-12	08:00:00
6	6	1	6	1	2013-12-26	12:45:00
7	7	1	1	4	2014-03-14	07:30:00
8	8	1	2	5	2014-04-18	08:00:00
9	9	1	3	6	2014-05-23	07:00:00
10	10	2	4	2	2014-08-16	07:30:00
11	11	2	5	1	2014-08-16	14:45:00
12	12	2	6	3	2014-12-26	08:30:00
13	13	2	1	4	2015-03-14	12:30:00
14	14	2	2	3	2014-11-30	10:00:00
15	15	2	3	1	2014-07-16	10:45:00
16	16	2	4	5	2015-05-24	08:00:00
17	17	2	5	6	2015-04-21	07:30:00
18	18	2	6	2	2015-01-13	06:45:00

Past Fixtures Table

The past_fixtures table is an extension of the fixtures table. It specifies that these fixtures have already happened and that there are statistics about these fixtures and the players from these fixtures that can be gathered.

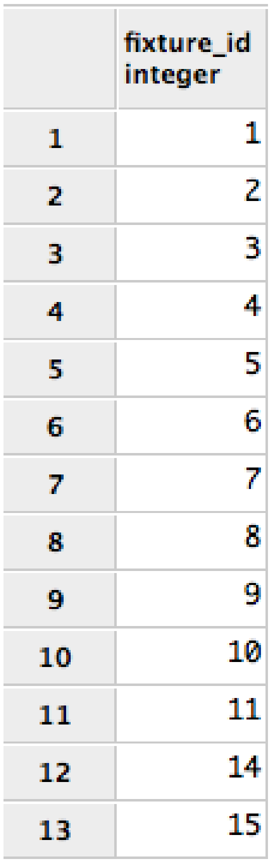
Functional Dependencies:

None

Create Statement:

```
CREATE TABLE past_fixtures (
    fixture_id  integer      NOT NULL REFERENCES fixtures (fixture_id),
    PRIMARY KEY (fixture_id)
);
```

Sample Data:



	fixture_id integer
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10
11	11
12	14
13	15

Future_Fixtures Table

The future_fixtures table is an extension of the fixtures table that specifies that the fixture has not occurred yet, therefore there are no match statistics to be gathered.

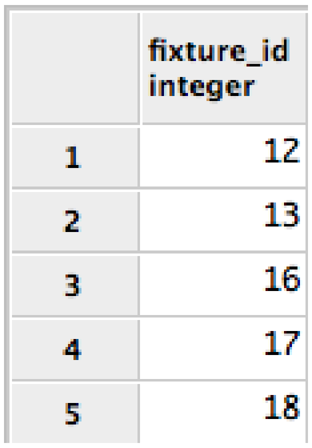
Functional Dependencies:

None

Create Statement:

```sql
CREATE TABLE future_fixtures (
    fixture_id  integer      NOT NULL REFERENCES fixtures (fixture_id),
    PRIMARY KEY (fixture_id)
);
```

Sample Data:



	fixture_id integer
1	12
2	13
3	16
4	17
5	18

Club_Match_Stats_Table Table

The club_match_stats_table holds all of the team statistics for a particular match (e.g. possession, corners, throw-ins)

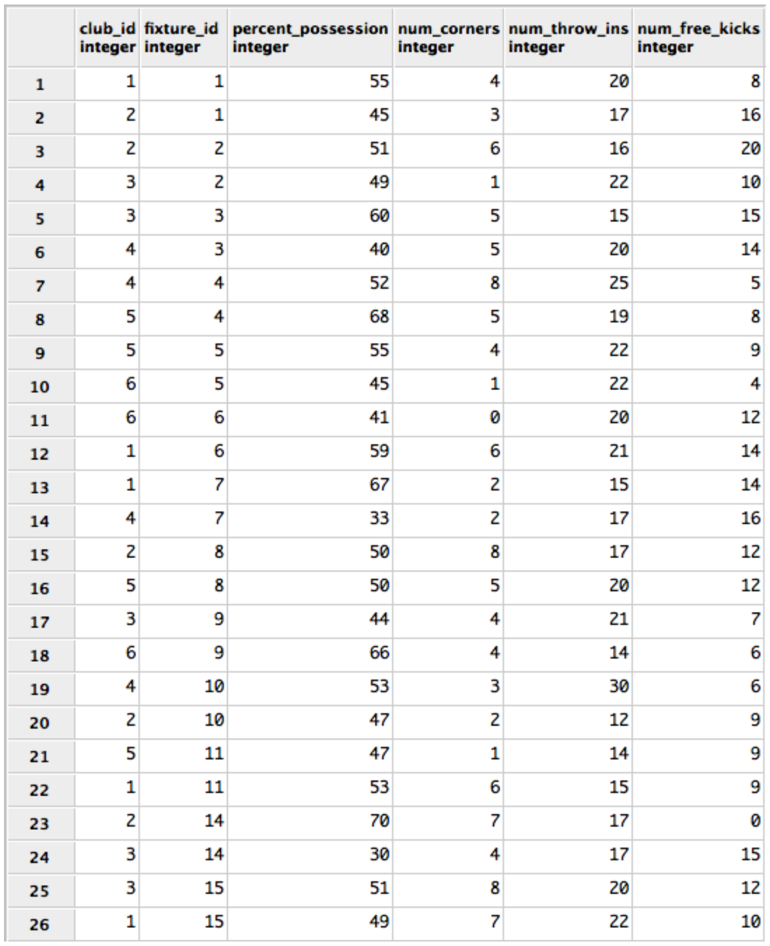
Functional Dependencies:

(club_id, fixture_id) → percent_possession, num_corners, num_throw_ins, num_free_kicks

Create Statement:

```
CREATE TABLE club_match_stats (
    club_id             integer     NOT NULL REFERENCES clubs (club_id),
    fixture_id          integer     NOT NULL REFERENCES past_fixtures (fixture_id),
    percent_possession  integer     NOT NULL CHECK (percent_possession >= 0 AND percent_possession <= 100),
    num_corners         integer     NOT NULL CHECK (num_corners >= 0),
    num_throw_ins       integer     NOT NULL CHECK (num_throw_ins >= 0),
    num_free_kicks      integer     NOT NULL CHECK (num_free_kicks >= 0),
    PRIMARY KEY (club_id, fixture_id)
);
```

Sample Data:



	club_id integer	fixture_id integer	percent_possession integer	num_corners integer	num_throw_ins integer	num_free_kicks integer
1	1	1	55	4	20	8
2	2	1	45	3	17	16
3	2	2	51	6	16	20
4	3	2	49	1	22	10
5	3	3	60	5	15	15
6	4	3	40	5	20	14
7	4	4	52	8	25	5
8	5	4	68	5	19	8
9	5	5	55	4	22	9
10	6	5	45	1	22	4
11	6	6	41	0	20	12
12	1	6	59	6	21	14
13	1	7	67	2	15	14
14	4	7	33	2	17	16
15	2	8	50	8	17	12
16	5	8	50	5	20	12
17	3	9	44	4	21	7
18	6	9	66	4	14	6
19	4	10	53	3	30	6
20	2	10	47	2	12	9
21	5	11	47	1	14	9
22	1	11	53	6	15	9
23	2	14	70	7	17	0
24	3	14	30	4	17	15
25	3	15	51	8	20	12
26	1	15	49	7	22	10

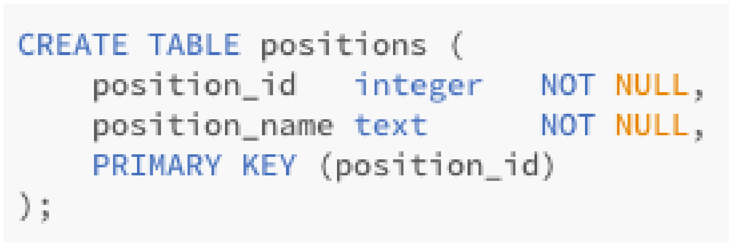
Positions Table

The positions table lists the different positions in soccer.

Functional Dependencies:

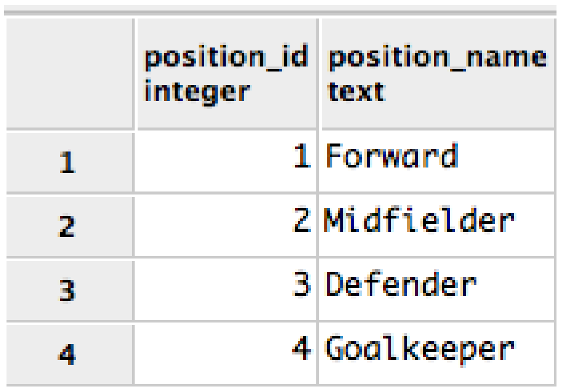
position_id → position_name

Create Statement:



```
CREATE TABLE positions (
    position_id   integer   NOT NULL,
    position_name text      NOT NULL,
    PRIMARY KEY (position_id)
);
```

Sample Data:



	position_id integer	position_name text
1	1	Forward
2	2	Midfielder
3	3	Defender
4	4	Goalkeeper

Plays Table

The plays table is a weak entity that lists a player and a position that he can play.

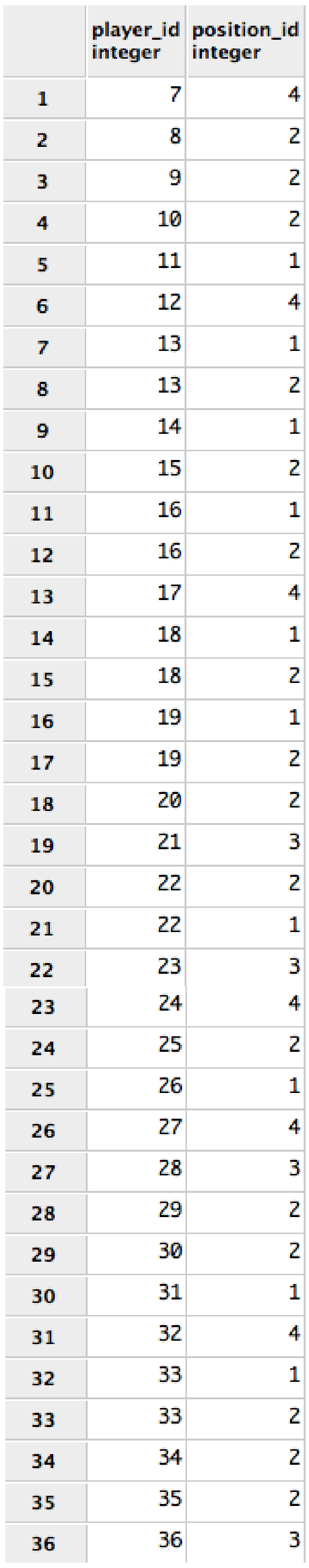
Functional Dependencies:

None

Create Statement:

```
CREATE TABLE plays (
    player_id   integer     NOT NULL REFERENCES players (player_id),
    position_id integer     NOT NULL REFERENCES positions (position_id),
    PRIMARY KEY (player_id, position_id)
);
```

Sample Data:



	player_id integer	position_id integer
1	7	4
2	8	2
3	9	2
4	10	2
5	11	1
6	12	4
7	13	1
8	13	2
9	14	1
10	15	2
11	16	1
12	16	2
13	17	4
14	18	1
15	18	2
16	19	1
17	19	2
18	20	2
19	21	3
20	22	2
21	22	1
22	23	3
23	24	4
24	25	2
25	26	1
26	27	4
27	28	3
28	29	2
29	30	2
30	31	1
31	32	4
32	33	1
33	33	2
34	34	2
35	35	2
36	36	3

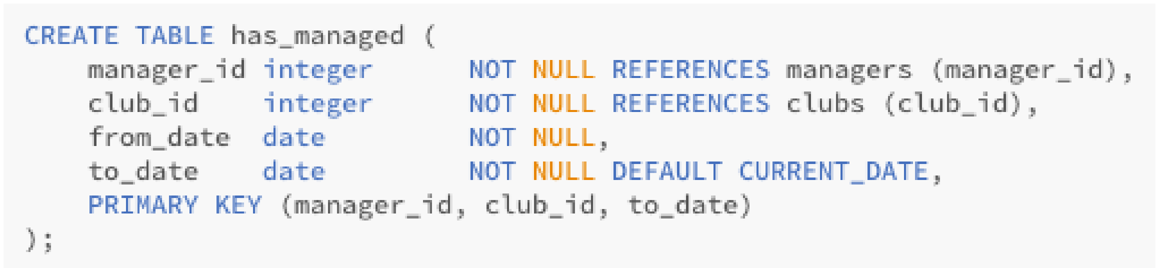
Has_Managed Table

The has_managed table combines a manager_id, club_id, and to_date to specify which teams a manager has managed, and during what time period.

Functional Dependencies:

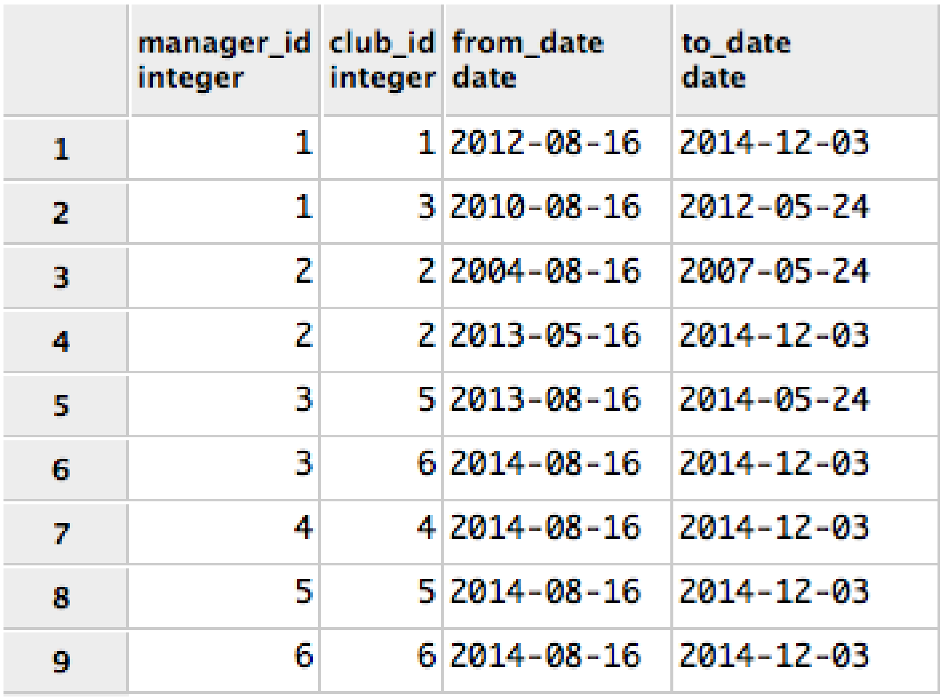
(manager_id, club_id, to_date) → from_date

Create Statement:



```
CREATE TABLE has_managed (
    manager_id integer      NOT NULL REFERENCES managers (manager_id),
    club_id    integer      NOT NULL REFERENCES clubs (club_id),
    from_date  date         NOT NULL,
    to_date    date         NOT NULL DEFAULT CURRENT_DATE,
    PRIMARY KEY (manager_id, club_id, to_date)
);
```

Sample Data:



	manager_id integer	club_id integer	from_date date	to_date date
1	1	1	2012-08-16	2014-12-03
2	1	3	2010-08-16	2012-05-24
3	2	2	2004-08-16	2007-05-24
4	2	2	2013-05-16	2014-12-03
5	3	5	2013-08-16	2014-05-24
6	3	6	2014-08-16	2014-12-03
7	4	4	2014-08-16	2014-12-03
8	5	5	2014-08-16	2014-12-03
9	6	6	2014-08-16	2014-12-03

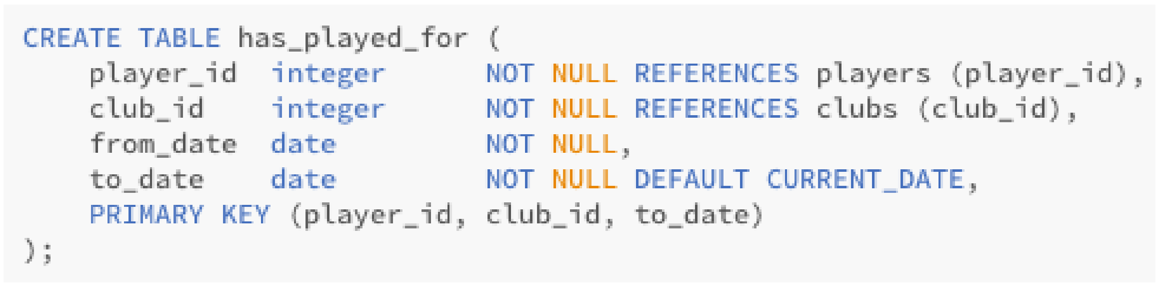
Has_Played_For Table

The has_played_for table gives a list of players and each team in the league that they have played for, as well as the dates that they were there.

Functional Dependencies:

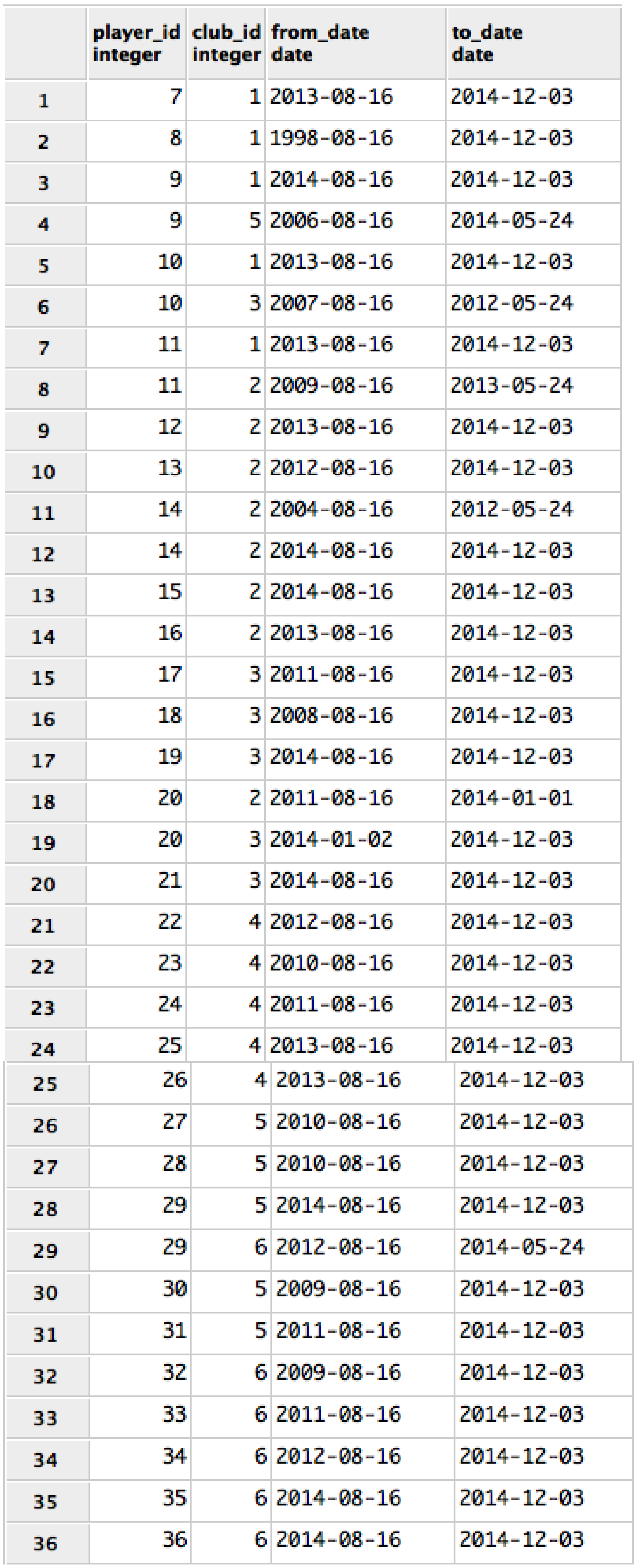
(player_id, club_id, to_date) → from_date

Create Statement:



```
CREATE TABLE has_played_for (
    player_id  integer      NOT NULL REFERENCES players (player_id),
    club_id    integer      NOT NULL REFERENCES clubs (club_id),
    from_date  date         NOT NULL,
    to_date    date         NOT NULL DEFAULT CURRENT_DATE,
    PRIMARY KEY (player_id, club_id, to_date)
);
```

Sample Data:



	player_id integer	club_id integer	from_date date	to_date date
1	7	1	2013-08-16	2014-12-03
2	8	1	1998-08-16	2014-12-03
3	9	1	2014-08-16	2014-12-03
4	9	5	2006-08-16	2014-05-24
5	10	1	2013-08-16	2014-12-03
6	10	3	2007-08-16	2012-05-24
7	11	1	2013-08-16	2014-12-03
8	11	2	2009-08-16	2013-05-24
9	12	2	2013-08-16	2014-12-03
10	13	2	2012-08-16	2014-12-03
11	14	2	2004-08-16	2012-05-24
12	14	2	2014-08-16	2014-12-03
13	15	2	2014-08-16	2014-12-03
14	16	2	2013-08-16	2014-12-03
15	17	3	2011-08-16	2014-12-03
16	18	3	2008-08-16	2014-12-03
17	19	3	2014-08-16	2014-12-03
18	20	2	2011-08-16	2014-01-01
19	20	3	2014-01-02	2014-12-03
20	21	3	2014-08-16	2014-12-03
21	22	4	2012-08-16	2014-12-03
22	23	4	2010-08-16	2014-12-03
23	24	4	2011-08-16	2014-12-03
24	25	4	2013-08-16	2014-12-03
25	26	4	2013-08-16	2014-12-03
26	27	5	2010-08-16	2014-12-03
27	28	5	2010-08-16	2014-12-03
28	29	5	2014-08-16	2014-12-03
29	29	6	2012-08-16	2014-05-24
30	30	5	2009-08-16	2014-12-03
31	31	5	2011-08-16	2014-12-03
32	32	6	2009-08-16	2014-12-03
33	33	6	2011-08-16	2014-12-03
34	34	6	2012-08-16	2014-12-03
35	35	6	2014-08-16	2014-12-03
36	36	6	2014-08-16	2014-12-03

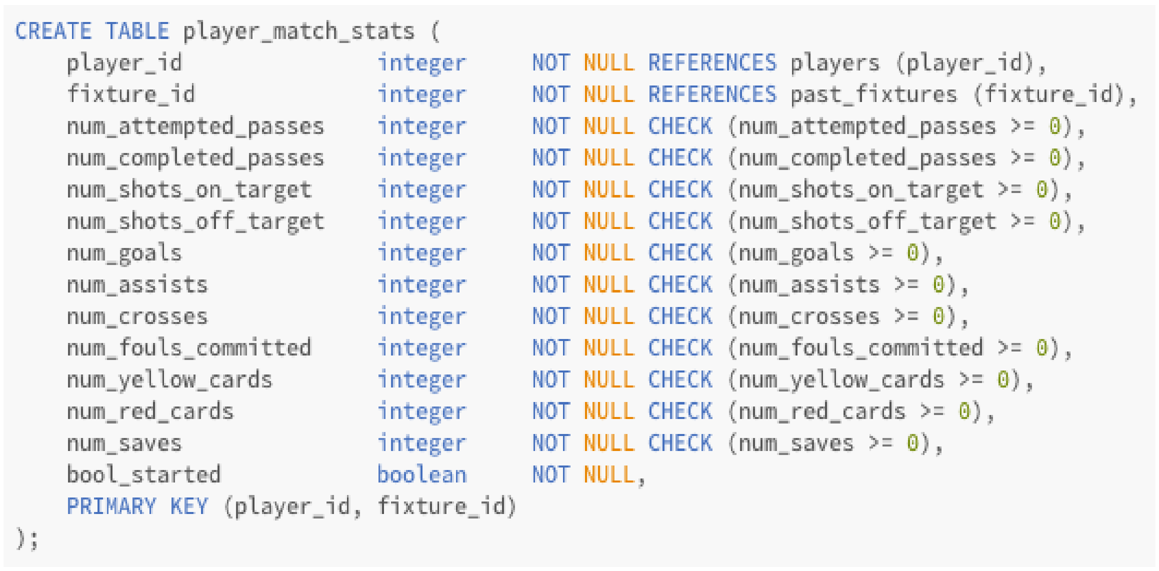
Player_Match_Stats Table

The player_match_stats table lists the statistics of all players in each match that they have played in (e.g. goals, assists, attempted passes, completed passes, etc.).

Functional Dependencies:

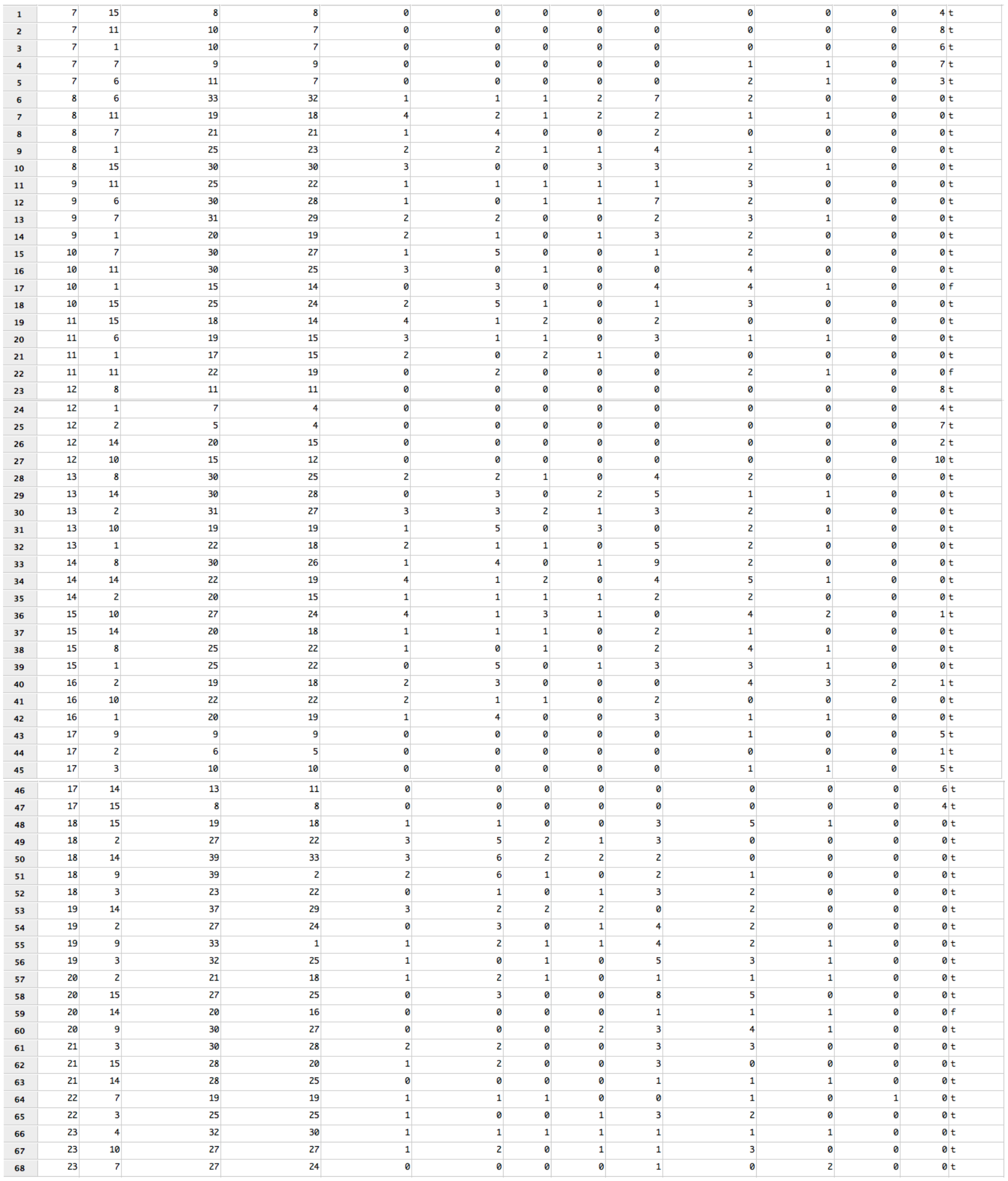
(player_id, fixture_id) → num_attempted_passes, num_completed_passes, num_shots_on_target, num_shots_off_target, num_goals, num_assists, num_crosses, num_fouls_committed, num_yellow_cards, num_red_cards, num_saves, bool_started

Create Statement:



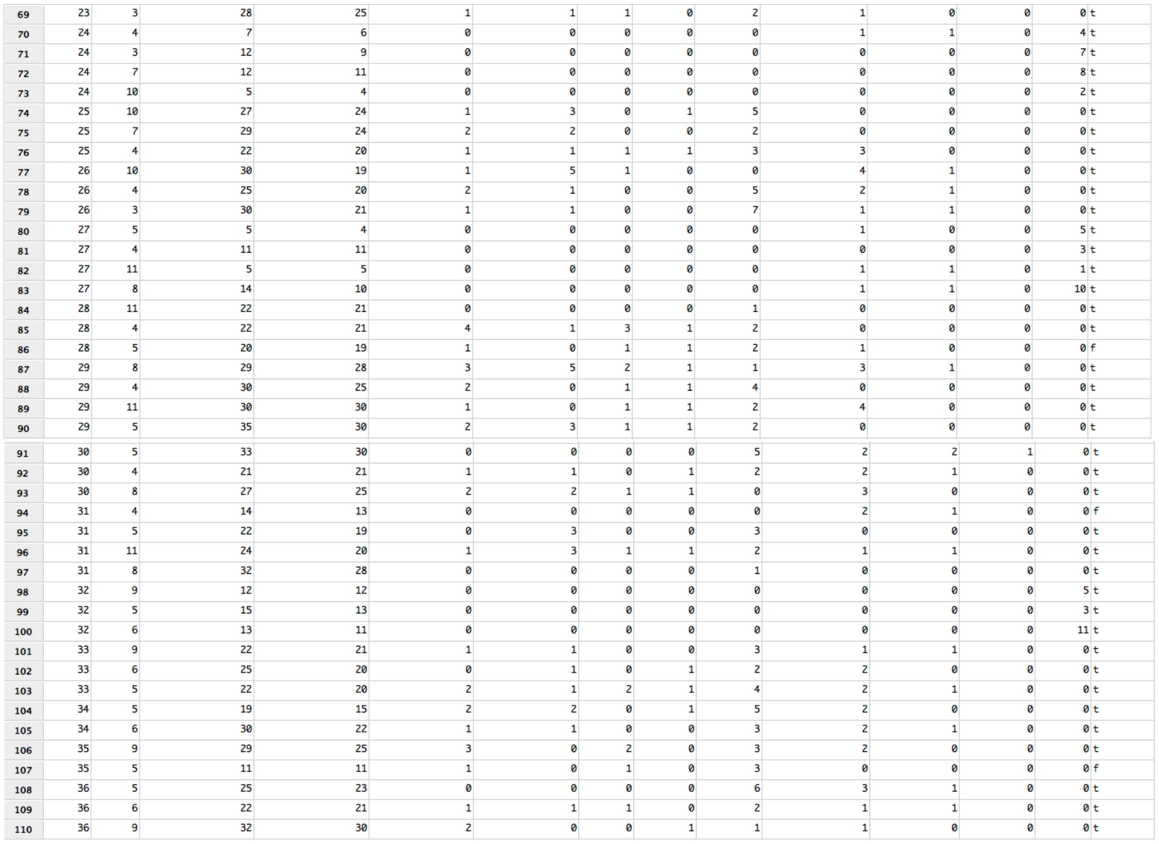
```
CREATE TABLE player_match_stats (
    player_id               integer     NOT NULL REFERENCES players (player_id),
    fixture_id              integer     NOT NULL REFERENCES past_fixtures (fixture_id),
    num_attempted_passes    integer     NOT NULL CHECK (num_attempted_passes >= 0),
    num_completed_passes    integer     NOT NULL CHECK (num_completed_passes >= 0),
    num_shots_on_target     integer     NOT NULL CHECK (num_shots_on_target >= 0),
    num_shots_off_target    integer     NOT NULL CHECK (num_shots_off_target >= 0),
    num_goals               integer     NOT NULL CHECK (num_goals >= 0),
    num_assists             integer     NOT NULL CHECK (num_assists >= 0),
    num_crosses             integer     NOT NULL CHECK (num_crosses >= 0),
    num_fouls_committed     integer     NOT NULL CHECK (num_fouls_committed >= 0),
    num_yellow_cards        integer     NOT NULL CHECK (num_yellow_cards >= 0),
    num_red_cards           integer     NOT NULL CHECK (num_red_cards >= 0),
    num_saves               integer     NOT NULL CHECK (num_saves >= 0),
    bool_started            boolean     NOT NULL,
    PRIMARY KEY (player_id, fixture_id)
);
```

Sample Data:



1	7	15	8	8	0	0	0	0	0	0	0	0	4	t
2	7	11	10	7	0	0	0	0	0	0	0	0	8	t
3	7	1	10	7	0	0	0	0	0	0	0	0	6	t
4	7	7	9	9	0	0	0	0	0	1	1	0	7	t
5	7	6	11	7	0	0	0	0	0	2	1	0	3	t
6	8	6	33	32	1	1	1	2	7	2	0	0	0	t
7	8	11	19	18	4	2	1	2	2	1	1	0	0	t
8	8	7	21	21	1	4	0	0	2	0	0	0	0	t
9	8	1	25	23	2	2	1	1	4	1	0	0	0	t
10	8	15	30	30	3	0	0	3	3	2	1	0	0	t
11	9	11	25	22	1	1	1	1	1	3	0	0	0	t
12	9	6	30	28	1	0	1	1	7	2	0	0	0	t
13	9	7	31	29	2	2	0	0	2	3	1	0	0	t
14	9	1	20	19	2	1	0	1	3	2	0	0	0	t
15	10	7	30	27	1	5	0	0	1	2	0	0	0	t
16	10	11	30	25	3	0	1	0	0	4	0	0	0	t
17	10	1	15	14	0	3	0	0	4	4	1	0	0	f
18	10	15	25	24	2	5	1	0	1	3	0	0	0	t
19	11	15	18	14	4	1	2	0	2	0	0	0	0	t
20	11	6	19	15	3	1	1	0	3	1	1	0	0	t
21	11	1	17	15	2	0	2	1	0	0	0	0	0	t
22	11	11	22	19	0	2	0	0	0	2	1	0	0	f
23	12	8	11	11	0	0	0	0	0	0	0	0	8	t
24	12	1	7	4	0	0	0	0	0	0	0	0	4	t
25	12	2	5	4	0	0	0	0	0	0	0	0	7	t
26	12	14	20	15	0	0	0	0	0	0	0	0	2	t
27	12	10	15	12	0	0	0	0	0	0	0	0	10	t
28	13	8	30	25	2	2	1	0	4	2	0	0	0	t
29	13	14	30	28	0	3	0	2	5	1	1	0	0	t
30	13	2	31	27	3	3	2	1	3	2	0	0	0	t
31	13	10	19	19	1	5	0	3	0	2	1	0	0	t
32	13	1	22	18	2	1	1	0	5	2	0	0	0	t
33	14	8	30	26	1	4	0	1	9	2	0	0	0	t
34	14	14	22	19	4	1	2	0	4	5	1	0	0	t
35	14	2	20	15	1	1	1	1	2	2	0	0	0	t
36	15	10	27	24	4	1	3	1	0	4	2	0	1	t
37	15	14	20	18	1	1	1	0	2	1	0	0	0	t
38	15	8	25	22	1	0	1	0	2	4	1	0	0	t
39	15	1	25	22	0	5	0	1	3	3	1	0	0	t
40	16	2	19	18	2	3	0	0	0	4	3	2	1	t
41	16	10	22	22	2	1	1	0	2	0	0	0	0	t
42	16	1	20	19	1	4	0	0	3	1	1	0	0	t
43	17	9	9	9	0	0	0	0	0	1	0	0	5	t
44	17	2	6	5	0	0	0	0	0	0	0	0	1	t
45	17	3	10	10	0	0	0	0	0	1	1	0	5	t
46	17	14	13	11	0	0	0	0	0	0	0	0	6	t
47	17	15	8	8	0	0	0	0	0	0	0	0	4	t
48	18	15	19	18	1	1	0	0	3	5	1	0	0	t
49	18	2	27	22	3	5	2	1	3	0	0	0	0	t
50	18	14	39	33	3	6	2	2	2	0	0	0	0	t
51	18	9	39	2	2	6	1	0	2	1	0	0	0	t
52	18	3	23	22	0	1	0	1	3	2	0	0	0	t
53	19	14	37	29	3	2	2	2	0	2	0	0	0	t
54	19	2	27	24	0	3	0	1	4	2	0	0	0	t
55	19	9	33	1	1	2	1	1	4	2	1	0	0	t
56	19	3	32	25	1	0	1	0	5	3	1	0	0	t
57	20	2	21	18	1	2	1	0	1	1	1	0	0	t
58	20	15	27	25	0	3	0	0	8	5	0	0	0	t
59	20	14	20	16	0	0	0	0	1	1	1	0	0	f
60	20	9	30	27	0	0	0	2	3	4	1	0	0	t
61	21	3	30	28	2	2	0	0	3	3	0	0	0	t
62	21	15	28	20	1	2	0	0	3	0	0	0	0	t
63	21	14	28	25	0	0	0	0	1	1	1	0	0	t
64	22	7	19	19	1	1	1	0	0	1	0	1	0	t
65	22	3	25	25	1	0	0	1	3	2	0	0	0	t
66	23	4	32	30	1	1	1	1	1	1	1	0	0	t
67	23	10	27	27	1	2	0	1	1	3	0	0	0	t
68	23	7	27	24	0	0	0	0	1	0	2	0	0	t

Sample Data Cont.:



69	23	3	28	25	1	1	1	0	2	1	0	0	0	t
70	24	4	7	6	0	0	0	0	0	1	1	0	4	t
71	24	3	12	9	0	0	0	0	0	0	0	0	7	t
72	24	7	12	11	0	0	0	0	0	0	0	0	8	t
73	24	10	5	4	0	0	0	0	0	0	0	0	2	t
74	25	10	27	24	1	3	0	1	5	0	0	0	0	t
75	25	7	29	24	2	2	0	0	2	0	0	0	0	t
76	25	4	22	20	1	1	1	1	3	3	0	0	0	t
77	26	10	30	19	1	5	1	0	0	4	1	0	0	t
78	26	4	25	20	2	1	0	0	5	2	1	0	0	t
79	26	3	30	21	1	1	0	0	7	1	1	0	0	t
80	27	5	5	4	0	0	0	0	0	1	0	0	5	t
81	27	4	11	11	0	0	0	0	0	0	0	0	3	t
82	27	11	5	5	0	0	0	0	0	1	1	0	1	t
83	27	8	14	10	0	0	0	0	0	1	1	0	10	t
84	28	11	22	21	0	0	0	0	1	0	0	0	0	t
85	28	4	22	21	4	1	3	1	2	0	0	0	0	t
86	28	5	20	19	1	0	1	1	2	1	0	0	0	f
87	29	8	29	28	3	5	2	1	1	3	1	0	0	t
88	29	4	30	25	2	0	1	1	4	0	0	0	0	t
89	29	11	30	30	1	0	1	1	2	4	0	0	0	t
90	29	5	35	30	2	3	1	1	2	0	0	0	0	t
91	30	5	33	30	0	0	0	0	5	2	2	1	0	t
92	30	4	21	21	1	1	0	1	2	2	1	0	0	t
93	30	8	27	25	2	2	1	1	0	3	0	0	0	t
94	31	4	14	13	0	0	0	0	0	2	1	0	0	f
95	31	5	22	19	0	3	0	0	3	0	0	0	0	t
96	31	11	24	20	1	3	1	1	2	1	1	0	0	t
97	31	8	32	28	0	0	0	0	1	0	0	0	0	t
98	32	9	12	12	0	0	0	0	0	0	0	0	5	t
99	32	5	15	13	0	0	0	0	0	0	0	0	3	t
100	32	6	13	11	0	0	0	0	0	0	0	0	11	t
101	33	9	22	21	1	1	0	0	3	1	1	0	0	t
102	33	6	25	20	0	1	0	1	2	2	0	0	0	t
103	33	5	22	20	2	1	2	1	4	2	1	0	0	t
104	34	5	19	15	2	2	0	1	5	2	0	0	0	t
105	34	6	30	22	1	1	0	0	3	2	1	0	0	t
106	35	9	29	25	3	0	2	0	3	2	0	0	0	t
107	35	5	11	11	1	0	1	0	3	0	0	0	0	f
108	36	5	25	23	0	0	0	0	6	3	1	0	0	t
109	36	6	22	21	1	1	1	0	2	1	1	0	0	t
110	36	9	32	30	2	0	0	1	1	1	0	0	0	t

Views:

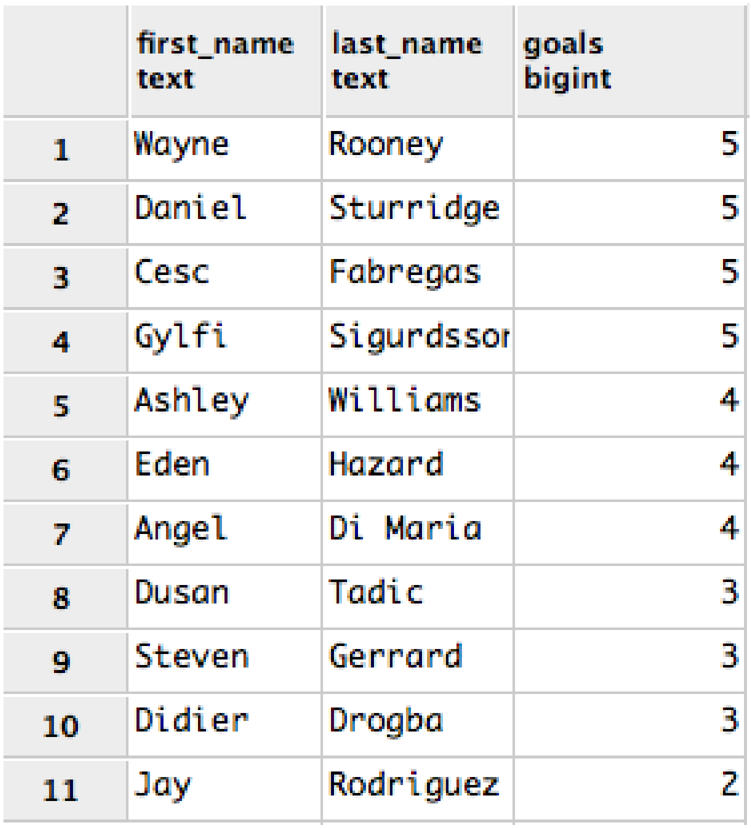
<u>career_leading_scorers</u>

This view gives you the first name, last name, and total number of goals scored during their career in the league, ordered from most goals to least goals.

SQL:

```
CREATE VIEW career_leading_scorers
AS
SELECT people.first_name, people.last_name, SUM(player_match_stats.num_goals) AS goals
FROM people
INNER JOIN players
ON people.person_id = players.player_id
INNER JOIN player_match_stats
ON players.player_id = player_match_stats.player_id
GROUP BY people.first_name, people.last_name
ORDER BY goals DESC;
```

Sample Data:



	first_name text	last_name text	goals bigint
1	Wayne	Rooney	5
2	Daniel	Sturridge	5
3	Cesc	Fabregas	5
4	Gylfi	Sigurdssor	5
5	Ashley	Williams	4
6	Eden	Hazard	4
7	Angel	Di Maria	4
8	Dusan	Tadic	3
9	Steven	Gerrard	3
10	Didier	Drogba	3
11	Jay	Rodriguez	2

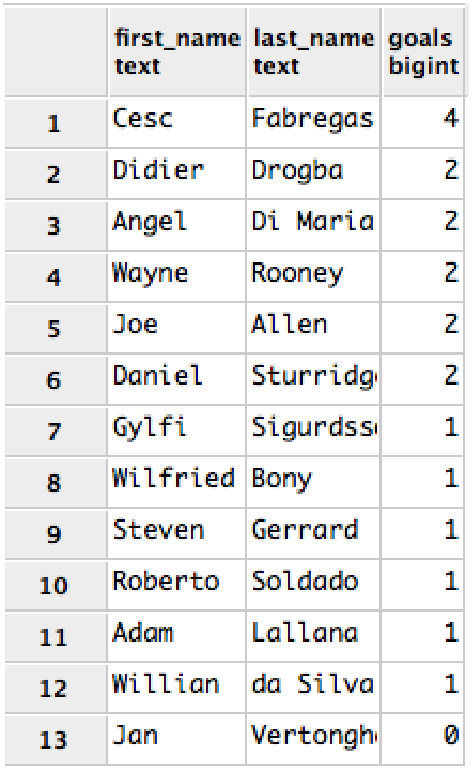
season_leading_scorers

This view gives you the first name, last name, and number of goals scored for the current season.

SQL:

```
CREATE VIEW career_leading_scorers
AS
SELECT people.first_name, people.last_name, SUM(player_match_stats.num_goals) AS goals
FROM people
INNER JOIN players
ON people.person_id = players.player_id
INNER JOIN player_match_stats
ON players.player_id = player_match_stats.player_id
INNER JOIN past_fixtures,
ON player_match_stats.fixture_id = past_fixtures.fixture_id
INNER JOIN fixtures
ON past_fixtures.fixture_id = fixtures.fixture_id
INNER JOIN seasons
ON fixtures.season_id = seasons.season_id
WHERE now() >= seasons.start_date AND now()<= seasons.end_date
GROUP BY people.first_name, people.last_name
ORDER BY goals DESC;
```

Sample Data:



	first_name text	last_name text	goals bigint
1	Cesc	Fabregas	4
2	Didier	Drogba	2
3	Angel	Di Maria	2
4	Wayne	Rooney	2
5	Joe	Allen	2
6	Daniel	Sturridg	2
7	Gylfi	Sigurdss	1
8	Wilfried	Bony	1
9	Steven	Gerrard	1
10	Roberto	Soldado	1
11	Adam	Lallana	1
12	Willian	da Silva	1
13	Jan	Vertongh	0

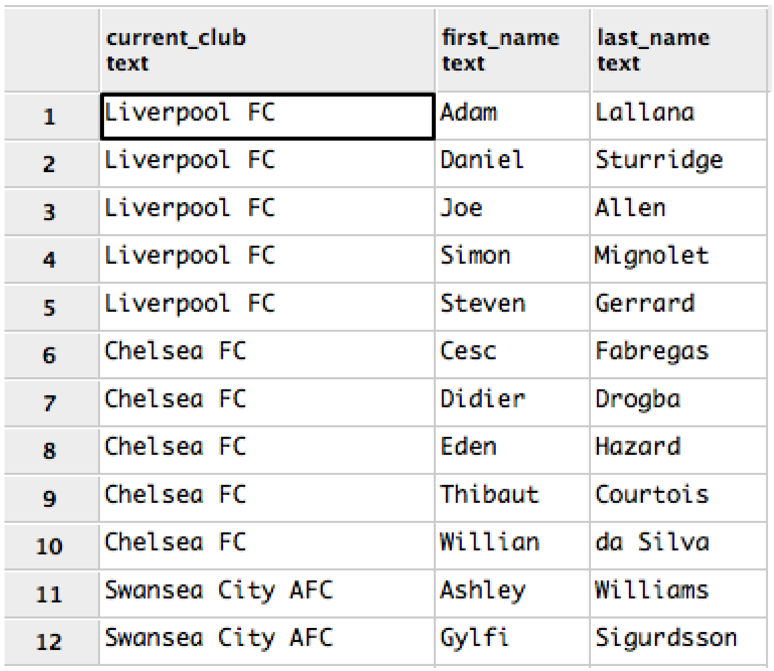
<u>current_players</u>

This view gives you a list of the first and last names of all players currently playing in the league and the team that they play for.

SQL:

```sql
CREATE VIEW current_players
AS
SELECT clubs.club_name as current_club, people.first_name, people.last_name
FROM clubs
INNER JOIN has_played_for
ON clubs.club_id = has_played_for.club_id
INNER JOIN players
ON has_played_for.player_id = players.player_id
INNER JOIN people
ON players.player_id = people.person_id
WHERE has_played_for.to_date in (SELECT max(has_played_for.to_date)
                                        FROM has_played_for)
GROUP BY clubs.club_name, people.first_name, people.last_name, clubs.club_id
ORDER BY clubs.club_id;
```

Sample Data:



	current_club text	first_name text	last_name text
1	Liverpool FC	Adam	Lallana
2	Liverpool FC	Daniel	Sturridge
3	Liverpool FC	Joe	Allen
4	Liverpool FC	Simon	Mignolet
5	Liverpool FC	Steven	Gerrard
6	Chelsea FC	Cesc	Fabregas
7	Chelsea FC	Didier	Drogba
8	Chelsea FC	Eden	Hazard
9	Chelsea FC	Thibaut	Courtois
10	Chelsea FC	Willian	da Silva
11	Swansea City AFC	Ashley	Williams
12	Swansea City AFC	Gylfi	Sigurdsson

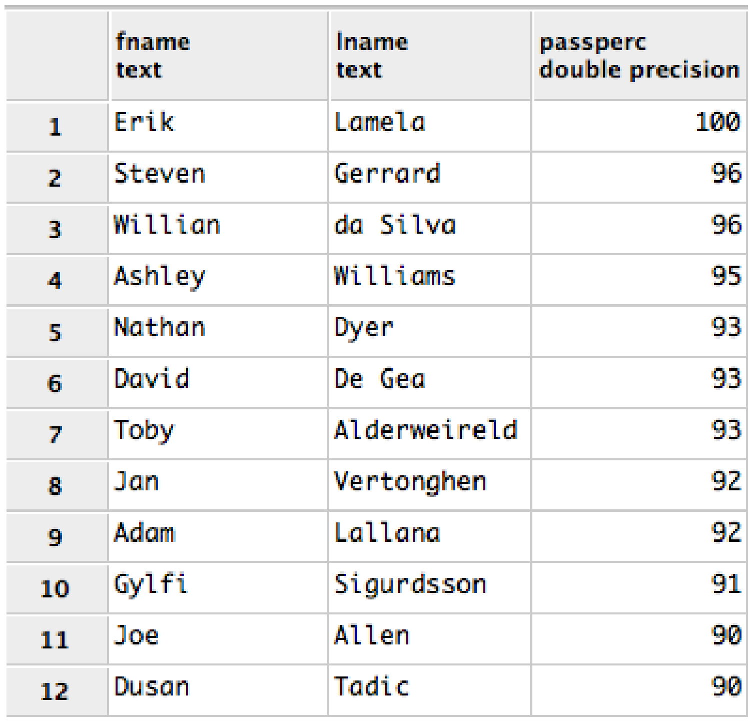
best_career_passers

This view provides a list of players and their career passing percentages.

SQL:

```sql
CREATE VIEW best_career_passers
AS
SELECT people.first_name fname, people.last_name lname,
floor(cast(SUM(player_match_stats.num_completed_passes) as
float)/cast(SUM(player_match_stats.num_attempted_passes) as float)*100) passPerc
FROM people
INNER JOIN players
ON people.person_id = players.player_id
INNER JOIN player_match_stats
ON players.player_id = player_match_stats.player_id
GROUP BY people.first_name, people.last_name
ORDER BY passPerc DESC;
```

Sample Data:



	fname text	lname text	passperc double precision
1	Erik	Lamela	100
2	Steven	Gerrard	96
3	Willian	da Silva	96
4	Ashley	Williams	95
5	Nathan	Dyer	93
6	David	De Gea	93
7	Toby	Alderweireld	93
8	Jan	Vertonghen	92
9	Adam	Lallana	92
10	Gylfi	Sigurdsson	91
11	Joe	Allen	90
12	Dusan	Tadic	90

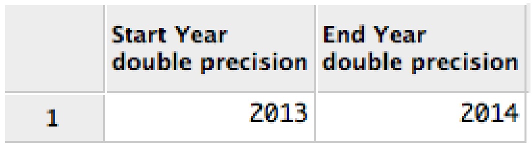
Reports & Their Queries:

Report 1:
Typically, soccer leagues begin in one year and end in the next. This query will return the start and end years of all past seasons (not the present league). This is useful for finding out which seasons have complete data.

SQL:

```
SELECT extract(year from seasons.start_date) AS "Start Year", extract(year from seasons.end_date) AS "End Year"
FROM seasons
WHERE now() > seasons.end_date;
```

Sample Data:



	Start Year double precision	End Year double precision
1	2013	2014

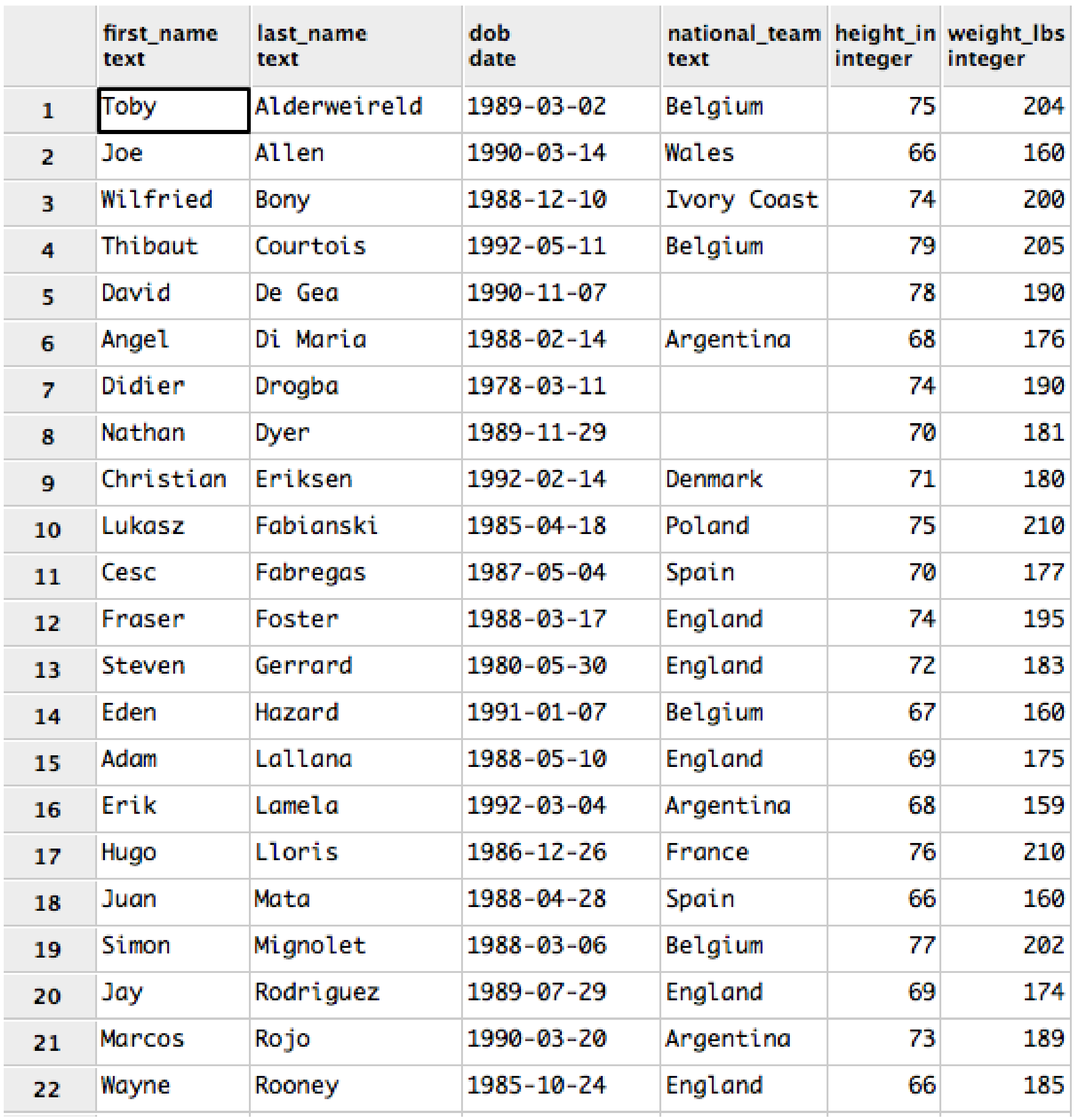
Report 2:

This query will give you a list of all players in the league and their basic information (e.g. name, date of birth, height, weight, etc.) This report is especially useful because you can change the "Order By" clause and list the players by age, height, weight, national teams, etc.

SQL:

```
SELECT people.first_name, people.last_name, people.dob, players.national_team, players.height_in, players.weight_lbs
FROM people
INNER JOIN players
ON people.person_id= players.player_id;
ORDER BY people.last_name ASC;
```

Sample Data:



	first_name	last_name	dob	national_team	height_in	weight_lbs
	text	text	date	text	integer	integer
1	Toby	Alderweireld	1989-03-02	Belgium	75	204
2	Joe	Allen	1990-03-14	Wales	66	160
3	Wilfried	Bony	1988-12-10	Ivory Coast	74	200
4	Thibaut	Courtois	1992-05-11	Belgium	79	205
5	David	De Gea	1990-11-07		78	190
6	Angel	Di Maria	1988-02-14	Argentina	68	176
7	Didier	Drogba	1978-03-11		74	190
8	Nathan	Dyer	1989-11-29		70	181
9	Christian	Eriksen	1992-02-14	Denmark	71	180
10	Lukasz	Fabianski	1985-04-18	Poland	75	210
11	Cesc	Fabregas	1987-05-04	Spain	70	177
12	Fraser	Foster	1988-03-17	England	74	195
13	Steven	Gerrard	1980-05-30	England	72	183
14	Eden	Hazard	1991-01-07	Belgium	67	160
15	Adam	Lallana	1988-05-10	England	69	175
16	Erik	Lamela	1992-03-04	Argentina	68	159
17	Hugo	Lloris	1986-12-26	France	76	210
18	Juan	Mata	1988-04-28	Spain	66	160
19	Simon	Mignolet	1988-03-06	Belgium	77	202
20	Jay	Rodriguez	1989-07-29	England	69	174
21	Marcos	Rojo	1990-03-20	Argentina	73	189
22	Wayne	Rooney	1985-10-24	England	66	185

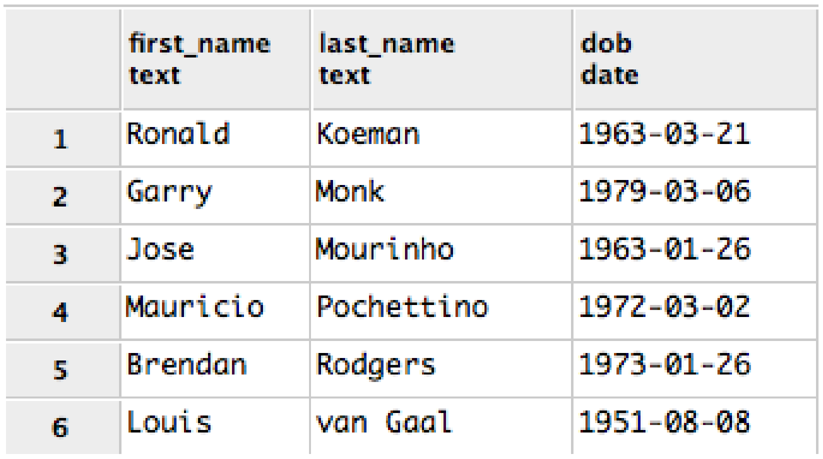
Report 3:

This query will return a list of the managers in the league and some basic information like their first name, last name, and date of birth.

SQL:

```sql
SELECT people.first_name, people.last_name, people.dob
FROM people
INNER JOIN managers
ON people.person_id = managers.manager_id
ORDER BY people.last_name ASC;
```

Sample Data:



	first_name	last_name	dob
	text	text	date
1	Ronald	Koeman	1963-03-21
2	Garry	Monk	1979-03-06
3	Jose	Mourinho	1963-01-26
4	Mauricio	Pochettino	1972-03-02
5	Brendan	Rodgers	1973-01-26
6	Louis	van Gaal	1951-08-08

Stored Procedures:

<u>getPlayersByPosition()</u>

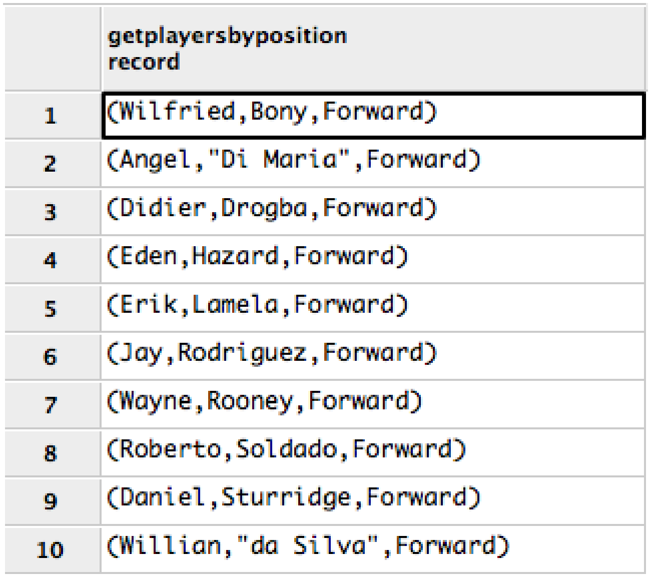
Returns all players that can play a specified position.

SQL:

```
CREATE OR REPLACE FUNCTION getPlayersByPosition(pos text)
RETURNS TABLE("First Name" text, "Last Name" text, "Position" text)
AS
$$
BEGIN
RETURN QUERY
SELECT people.first_name AS "First Name", people.last_name AS "Last Name", positions.position_name AS "Position"
FROM people
INNER JOIN players
ON people.person_id = players.player_id
INNER JOIN plays
ON players.player_id = plays.player_id
INNER JOIN positions
ON plays.position_id = positions.position_id
WHERE positions.position_name = pos
GROUP BY people.first_name, people.last_name, positions.position_name
ORDER BY people.last_name;
END;
$$
LANGUAGE plpgsql;
```

Sample Data:

SELECT getPlayersByPosition('Forward');



	getplayersbyposition record
1	(Wilfried,Bony,Forward)
2	(Angel,"Di Maria",Forward)
3	(Didier,Drogba,Forward)
4	(Eden,Hazard,Forward)
5	(Erik,Lamela,Forward)
6	(Jay,Rodriguez,Forward)
7	(Wayne,Rooney,Forward)
8	(Roberto,Soldado,Forward)
9	(Daniel,Sturridge,Forward)
10	(Willian,"da Silva",Forward)

getClubRoster()

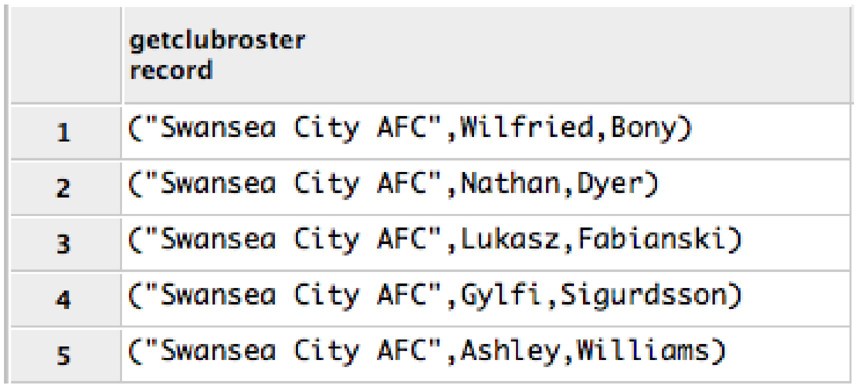
Returns the current player roster for a specified club.

SQL:

```sql
CREATE OR REPLACE FUNCTION getClubRoster(club text)
RETURNS TABLE("Club" text, "First Name" text, "Last Name" text)
AS
$$
BEGIN
RETURN QUERY
SELECT clubs.club_name AS "Club", people.first_name AS "First Name", people.last_name AS "Last Name"
FROM people
INNER JOIN players
ON people.person_id = players.player_id
INNER JOIN has_played_for
ON players.player_id = has_played_for.player_id
INNER JOIN clubs
ON has_played_for.club_id = clubs.club_id
WHERE club = clubs.club_name AND has_played_for.to_date in (SELECT max(has_played_for.to_date)
                                                            FROM has_played_for)
GROUP BY clubs.club_name, people.first_name, people.last_name
ORDER BY people.last_name ASC;
END;
$$
LANGUAGE plpgsql;
```

Sample Data:

SELECT getClubRoster('Swansea City AFC');



	getclubroster record
1	("Swansea City AFC",Wilfried,Bony)
2	("Swansea City AFC",Nathan,Dyer)
3	("Swansea City AFC",Lukasz,Fabianski)
4	("Swansea City AFC",Gylfi,Sigurdsson)
5	("Swansea City AFC",Ashley,Williams)

<u>matchOnDate()</u>

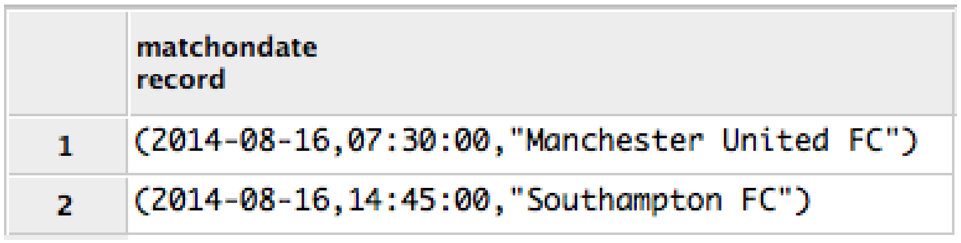
This function accepts a date and returns a list of fixtures on this date, the time of the fixtures, and the home team taking part in fixture.

SQL:

```sql
CREATE OR REPLACE FUNCTION matchOnDate(whatDate date)
RETURNS TABLE("Date" date, "Time" time, "Home Team" text)
AS
$$
BEGIN
RETURN QUERY
SELECT fixtures.fixture_date AS "Date", fixtures.start_time_EST AS "Time", clubs.club_name AS "Home Team"
FROM fixtures
INNER JOIN clubs
ON fixtures.home_club_id = clubs.club_id
WHERE whatDate = fixtures.fixture_date
GROUP BY fixtures.fixture_date, fixtures.start_time_EST, clubs.club_name
ORDER BY fixtures.start_time_EST ASC;
END;
$$
LANGUAGE plpgsql;
```

Sample Data:

SELECT matchOnDate('2014-08-16');



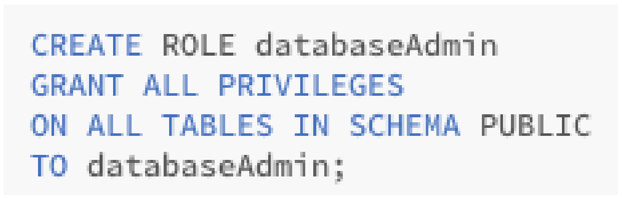
	matchondate record
1	(2014-08-16,07:30:00,"Manchester United FC")
2	(2014-08-16,14:45:00,"Southampton FC")

Security:

<u>Role: databseAdmin</u>

This is the role that allows the database administrator to do anything with the database. In the case that data or tables need to be deleted or another user's privileges need to be revoked, the database administrator can do this.

SQL:

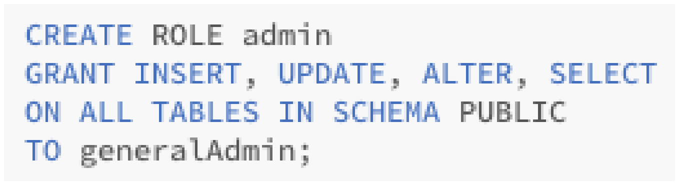


```
CREATE ROLE databaseAdmin
GRANT ALL PRIVILEGES
ON ALL TABLES IN SCHEMA PUBLIC
TO databaseAdmin;
```

<u>Role: generalAdmin</u>

This is the role of the person who will be inputting information about players, clubs, etc. as it happens. They will also be able to select from tables in the case that they are users and update tables for certain cases (dates of future fixtures being changed).

SQL:

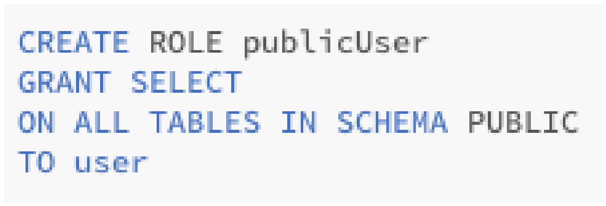


```
CREATE ROLE admin
GRANT INSERT, UPDATE, ALTER, SELECT
ON ALL TABLES IN SCHEMA PUBLIC
TO generalAdmin;
```

Role: publicUser

This is the role of the general public who access the database. They are allowed to select from any tables in the database, as they may need to in order to extract some useful information.

SQL:



```
CREATE ROLE publicUser
GRANT SELECT
ON ALL TABLES IN SCHEMA PUBLIC
TO user
```

Implementation Notes:

- The data in the example database pertains to the Barclays Premier League, but it could also be implemented to most soccer leagues to track stats and biographical information.
- The database is designed to be able to extract any information (within reason) about a single league that is desired, but not all of the views/queries provided will make it readily available. Additional queries would need to be written to access this information.

Known Problems:

- One of the major issues with this database is that it can only keep track of players' past teams are in the same league. It is very common in soccer for players to be traded between leagues.
- Another issue is the fact that the database does not keep track of final scores of past fixtures. This leads to very long and complicated queries when trying to calculate the points a team has accumulated over a season and current positions depending on their points. However, it can be done by calculating the sum of the goals by players on each team in a fixture and comparing them.

Future Enhancements:

In the future, I would like to expand this database to include multiple leagues, making it really become a worldwide soccer database. This would allow me to include information about tournaments such as the Champions League and would solve the known issue mentioned above by keeping track of teams from other leagues that players have played for. There are also some extra queries and views that I would have written given more time that would allow me to extract other interesting information such as the points that a team has accumulated and the current positions of teams.