Database project CS-322

Release 0.1

Amos Wenger, Sebastien Zurfluh & Yoan Blanc

March 25, 2012

CONTENTS

1 Deliverable 1 **1**

1.1 Create the ER model for the data 1

1.2 Design the database and the constraints needed to maintain the database consistent 3

1.3 Create the SQL commands to create the tables in Oracle 3

1.4 Conclusion 9

CHAPTER

ONE

DELIVERABLE 1

The goal of this deliverable is to design an ER model, a corresponding relational schema and create the database tables in the given database. The organization of the data in files and the given description does not imply neither an ER model nor a relational schema. It is given to help the student understand the format of the data faster. Finally, a discussion about constraints and removing redundant information is expected.

1.1 Create the ER model for the data

1.1.1 The Schema

See the figure on next page.

1.1.2 Decision made building the Entity-Relationship schema

We tried to remove every denormalized fields, meaning fields that are representing information you can obtain using the rest of the data set.

People

- *Players* and *Coaches* are pointing to the same entities the coach's *coach_id* and the player's *ilkid* are defining the same thing (e.g. *MOEDO01*). So we grouped them under the *Person* entity.
- *Player*'s height will be converted into inches, *1ft = 12in*.
- *Player*'s fields like *first_season*, *last_season* or *college* can be obtained from the *Draft* and *Player Season* entities. They are just denormalized fields.

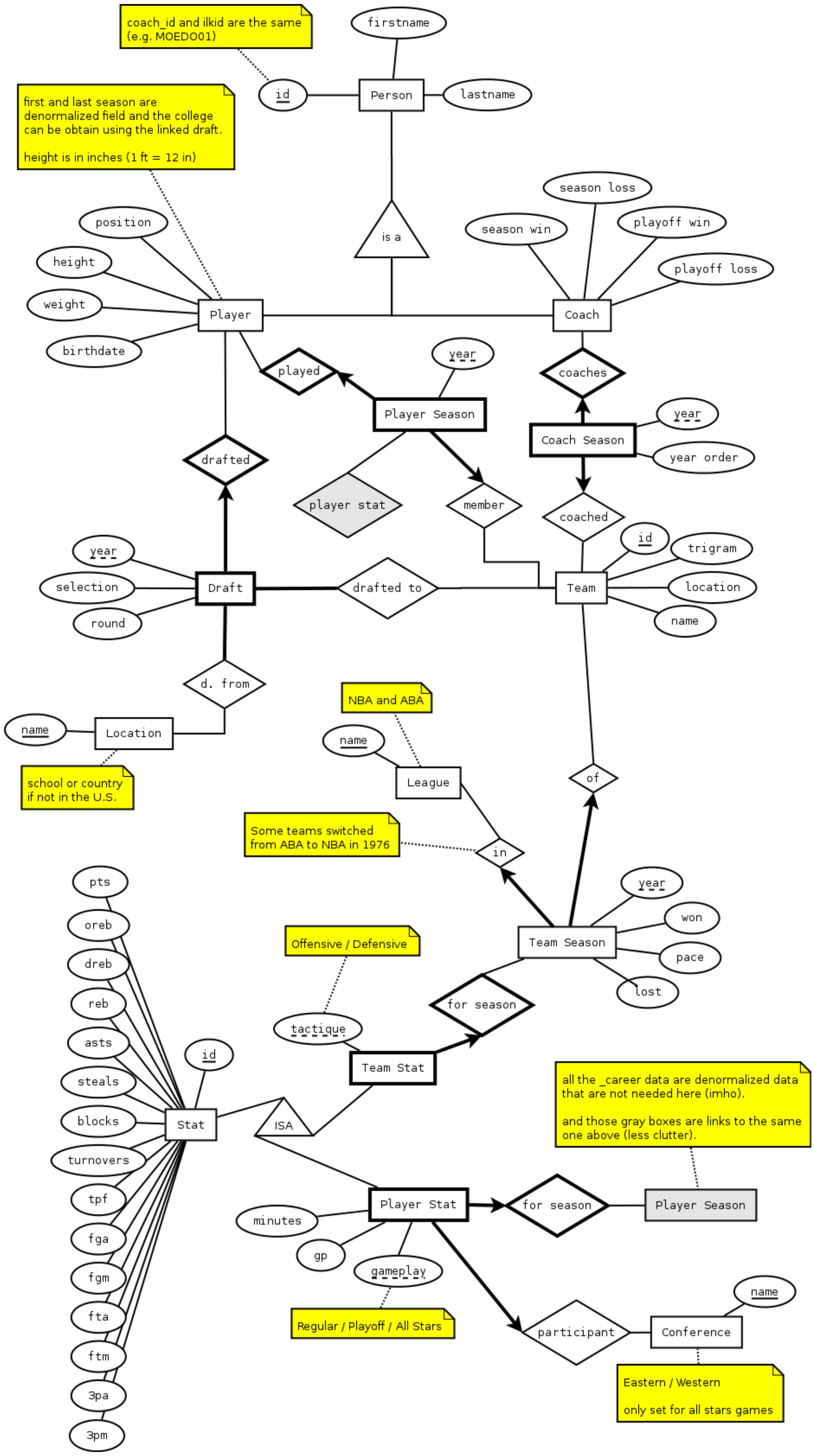
Teams

The teams pretty look like the CSV file, except that we removed the league from it since some teams switched from *ABA* to *NBA* when *ABA* merged into the later.

Conference and leagues

According to Wikipedia, a team belongs to a *Division* which belongs to a *Conference* (being *Eastern* and *Western*). We first linked the team to a *Conference* but the dataset gives the information about the *Conference* only on *All Star* games (where the best players of each *Conferences* create an *All Star* team and play against each other). So, the *Conference* information will only live there because of the dataset.

About *Leagues*, there are two of them *ABA* and *NBA* and a team may have changed during the year 1975 when *ABA* got merged into the *NBA*.



coach_id and ilkid are the same (e.g. MOEDO01)

first and last season are denormalized field and the college can be obtain using the linked draft.

height is in inches (1 ft = 12 in)

school or country if not in the U.S.

NBA and ABA

Some teams switched from ABA to NBA in 1976

Offensive / Defensive

all the _career data are denormalized data that are not needed here (imho).

and those gray boxes are links to the same one above (less clutter).

Regular / Playoff / All Stars

Eastern / Western

only set for all stars games

Figure 1.1: ER Schema made using Dia.

Drafts and Location

One major change from the dataset here is that the *People* who go drafted but never played will exist has a *Player*. Those kind of *Player* never played.

Location isn't really important, it's been moved out for further queries.

NB it doesn't have any link with the *Team*'s location which is a city when here, it's the College where the *Player* got drafted from.

Q: *Shouldn't we link Drafts to People and not Players in that case?*

A: No because even players who have never played for an ABA or NBA team have played before for a European team or a school team. Therefore they already have the characteristics of player (position, height, weight, birthdate) when they are drafted.

Stats and Seasons

The *Stats* (statistics) being very standard, it'll live as itself and being kind of *casted* into a *Player* or *Team* stat depending on the case. A *Stat* is all the time linked to a *Season* which is identified by the starting year in the dataset and our model (e.g. *1984* means *the season 84-85*).

Each *Player* and *Team* has a specific *Season* entity linked to it for each *year* played.

Then *Teams* have *offensive* and *defensive* statistics while the *Players* have statistics per kind of *Seasons* played:

- Regular
- Playoff
- and All Stars

NB: All the career stats were seen as denormalized and thus removed. We can get those data back from the yearly *Stats*.

1.2 Design the database and the constraints needed to maintain the database consistent

See the figure on next page.

1.3 Create the SQL commands to create the tables in Oracle

The following SQL schema is really a first shot, with very few constraints on numbers and strings (*varchar*).

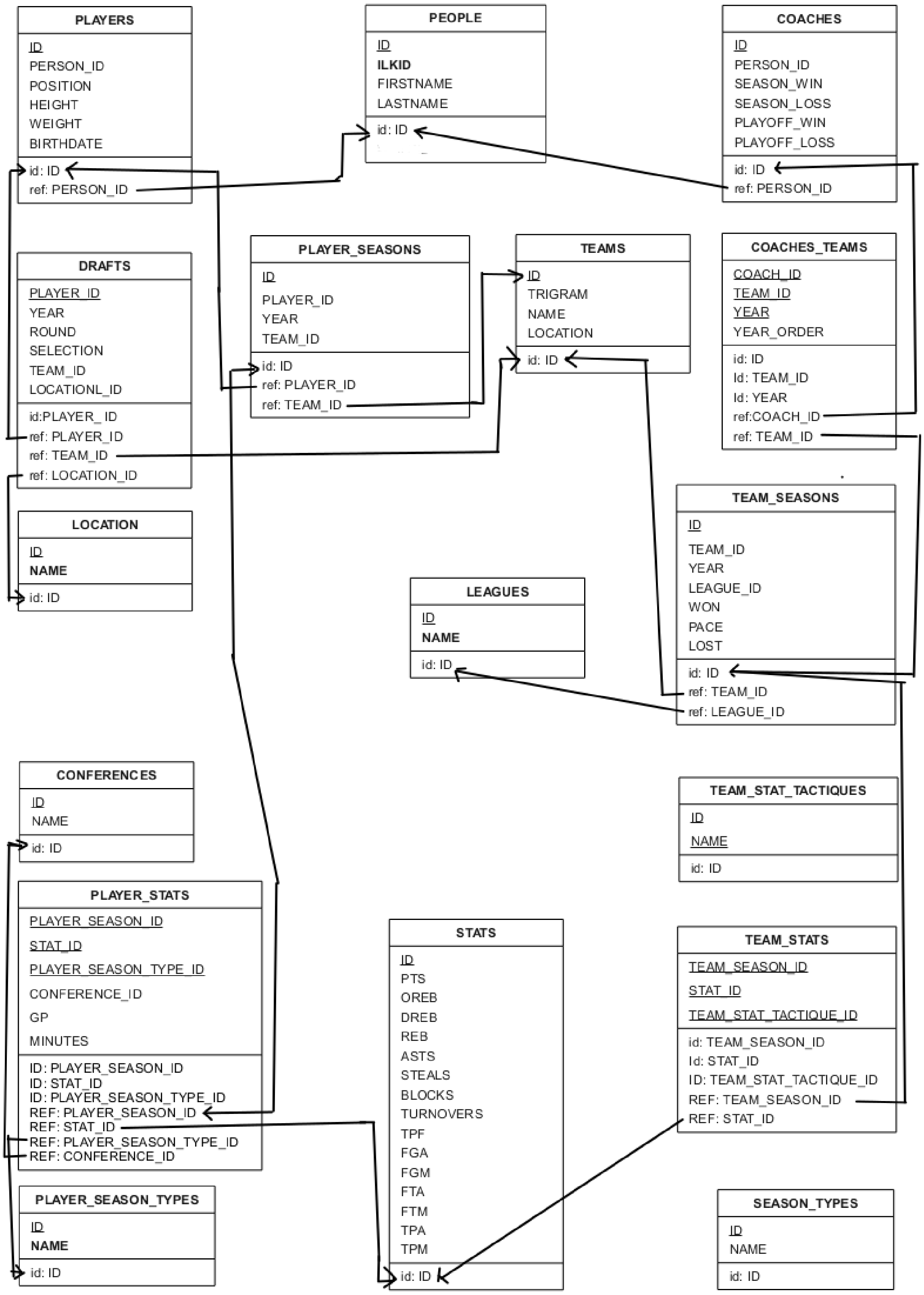
Not being familiar with the way Oracle works, we'll just explain some basic stuff.

Oracle being a very complex RDBMS (*Relational Database Management System*) that we are still learning, this section will simply clarify what we've discovered and wich might explain the following.

It cannot *auto increment* like MySQL or SQLite, so one must use a *sequence* and use it to get the current value or the next value when insert new rows.

```
INSERT INTO leagues (id, name) VALUES (leagues_seq.NEXTVAL, 'NBA');
```

It knows how to delete relations in cascade. It can remove an entire structure from one simple *DELETE*. If *PostgreSQL* can do that has well, the more common *MySQL* or *SQLite* cannot. In the following schema, we activated that cascaded deletion without thinking deeply about it. This schema will be refined in the future when used with the data.


PLAYERS
ID
PERSON_ID
POSITION
HEIGHT
WEIGHT
BIRTHDATE
id: ID
ref: PERSON_ID
PEOPLE
ID
ILKID
FIRSTNAME
LASTNAME
id: ID
COACHES
ID
PERSON_ID
SEASON_WIN
SEASON_LOSS
PLAYOFF_WIN
PLAYOFF_LOSS
id: ID
ref: PERSON_ID
DRAFTS
PLAYER_ID
YEAR
ROUND
SELECTION
TEAM_ID
LOCATIONL_ID
id:PLAYER_ ID
ref: PLAYER_ID
ref: TEAM_ID
ref: LOCATION_ID
PLAYER_SEASONS
ID
PLAYER_ID
YEAR
TEAM_ID
id: ID
ref: PLAYER_ID
ref: TEAM_ID
TEAMS
ID
TRIGRAM
NAME
LOCATION
id: ID
COACHES_TEAMS
COACH_ID
TEAM_ID
YEAR
YEAR_ORDER
id: ID
Id: TEAM_ID
Id: YEAR
ref:COACH_ID
ref: TEAM_ID
LOCATION
ID
NAME
id: ID
TEAM_SEASONS
ID
TEAM_ID
YEAR
LEAGUE_ID
WON
PACE
LOST
id: ID
ref: TEAM_ID
ref: LEAGUE_ID
LEAGUES
ID
NAME
id: ID
CONFERENCES
ID
NAME
id: ID
TEAM_STAT_TACTIQUES
ID
NAME
id: ID
PLAYER_STATS
PLAYER_SEASON_ID
STAT_ID
PLAYER_SEASON_TYPE_ID
CONFERENCE_ID
GP
MINUTES
ID: PLAYER_SEASON_ID
ID: STAT_ID
ID: PLAYER_SEASON_TYPE_ID
REF: PLAYER_SEASON_ID
REF: STAT_ID
REF: PLAYER_SEASON_TYPE_ID
REF: CONFERENCE_ID
STATS
ID
PTS
OREB
DREB
REB
ASTS
STEALS
BLOCKS
TURNOVERS
TPF
FGA
FGM
FTA
FTM
TPA
TPM
id: ID
TEAM_STATS
TEAM_SEASON_ID
STAT_ID
TEAM_STAT_TACTIQUE_ID
id: TEAM_SEASON_ID
Id: STAT_ID
ID: TEAM_STAT_TACTIQUE_ID
REF: TEAM_SEASON_ID
REF: STAT_ID
PLAYER_SEASON_TYPES
ID
NAME
id: ID
SEASON_TYPES
ID
NAME
id: ID

```
--
-- People
-- ======
--
-- A person can be a player and/or a coach at different time of
-- her life.
--
-- ilkit can be NULL for drafted only players
--

CREATE TABLE people (
    id NUMBER,
    ilkid VARCHAR(9),
    firstname VARCHAR(255) NOT NULL,
    lastname VARCHAR(255) NOT NULL,
    PRIMARY KEY (id),
    UNIQUE (ilkid)
);

CREATE SEQUENCE people_seq
    START WITH 1
    INCREMENT BY 1;

CREATE TABLE players (
    id NUMBER,
    person_id NUMBER NOT NULL,
    position CHAR(1) NOT NULL,
    height NUMBER, -- in inches
    weight NUMBER,
    birthdate DATE,
    PRIMARY KEY (id),
    FOREIGN KEY (person_id)
        REFERENCES people (id) ON DELETE CASCADE
);

CREATE SEQUENCE players_seq
    START WITH 1
    INCREMENT BY 1;

CREATE TABLE coaches (
    id NUMBER,
    person_id NUMBER NOT NULL,
    season_win NUMBER,
    season_loss NUMBER,
    playoff_win NUMBER,
    playoff_loss NUMBER,
    PRIMARY KEY (id),
    FOREIGN KEY (person_id)
        REFERENCES people (id) ON DELETE CASCADE
);

CREATE SEQUENCE coaches_seq
    START WITH 1
    INCREMENT BY 1;

--
-- Group of people
-- ===============
--
-- Teams, leagues and stuff
--

-- NBA/ABA
```

```
CREATE TABLE leagues (
    id NUMBER,
    name CHAR(3) NOT NULL,
    PRIMARY KEY (id),
    UNIQUE (name)
);

CREATE SEQUENCE leagues_seq
    START WITH 1
    INCREMENT BY 1;

CREATE TABLE conferences (
    id NUMBER,
    name VARCHAR(31) NOT NULL,
    PRIMARY KEY (id),
    UNIQUE (name)
);

CREATE SEQUENCE conferences_seq
    START WITH 1
    INCREMENT BY 1;

CREATE TABLE teams (
    id NUMBER,
    trigram CHAR(3) NOT NULL,
    name VARCHAR(255),
    location VARCHAR(255),
    PRIMARY KEY (id)
);

CREATE SEQUENCE teams_seq
    START WITH
    INCREMENT BY 1;

CREATE TABLE coaches_teams (
    coach_id NUMBER NOT NULL,
    team_id NUMBER NOT NULL,
    year NUMBER,
    year_order NUMBER,
    PRIMARY KEY (coach_id, team_id, year),
    FOREIGN KEY (coach_id)
        REFERENCES coaches (id) ON DELETE CASCADE,
    FOREIGN KEY (team_id)
        REFERENCES teams (id) ON DELETE CASCADE
);

--
-- Physical
-- ========
--
-- A school or a country if it’s outside the U.S.
--

CREATE TABLE location (
    id NUMBER,
    name VARCHAR(255),
    PRIMARY KEY (id),
    UNIQUE (name)
);

CREATE SEQUENCE location_seq
    START WITH 1
    INCREMENT BY 1;
```

```
--
-- Drafts
--

CREATE TABLE drafts (
    player_id NUMBER NOT NULL,
    year NUMBER NOT NULL,
    round NUMBER NOT NULL,
    selection NUMBER NOT NULL,
    team_id NUMBER NOT NULL,
    location_id NUMBER NULL,
    PRIMARY KEY (player_id),
    FOREIGN KEY (player_id)
        REFERENCES players (id) ON DELETE CASCADE,
    FOREIGN KEY (team_id)
        REFERENCES teams (id) ON DELETE CASCADE,
    FOREIGN KEY (l_id)
        REFERENCES location (id) ON DELETE CASCADE
);

--
-- Stats
-- =====
--
-- All the kind of statistical data
--

CREATE TABLE stats (
    id NUMBER,
    pts NUMBER,
    oreb NUMBER,
    dreb NUMBER,
    reb NUMBER,
    asts NUMBER,
    steals NUMBER,
    blocks NUMBER,
    turnovers NUMBER,
    tpf NUMBER,
    fga NUMBER,
    fgm NUMBER,
    fta NUMBER,
    ftm NUMBER,
    tpa NUMBER, -- 3pa
    tpm NUMBER, -- 3pm
    PRIMARY KEY (id)
);

CREATE SEQUENCE stats_seq
    START WITH 1
    INCREMENT BY 1;

--
-- Teams stats
-- ===========

CREATE TABLE team_seasons (
    id NUMBER,
    team_id NUMBER NOT NULL,
    year NUMBER NOT NULL,
    league_id NUMBER NOT NULL,
    won NUMBER,
    pace NUMBER,
    lost NUMBER,
    PRIMARY KEY (id),
    CONSTRAINT team_season_unique UNIQUE (team_id, year),
    FOREIGN KEY (team_id)
        REFERENCES teams (id) ON DELETE CASCADE,
    FOREIGN KEY (league_id)
        REFERENCES leagues (id) ON DELETE CASCADE
);

CREATE SEQUENCE team_seasons_seq
    START WITH 1
    INCREMENT BY 1;

CREATE TABLE team_stat_tactiques (
    id NUMBER NOT NULL,
    name VARCHAR(31),
    PRIMARY KEY (id),
    UNIQUE (name)
);

CREATE SEQUENCE team_stat_tactiques_seq
    START WITH 1
    INCREMENT BY 1;

CREATE TABLE team_stats (
    team_season_id NUMBER NOT NULL,
    stat_id NUMBER NOT NULL,
    team_stat_tactique_id NUMBER NOT NULL,
    PRIMARY KEY(team_season_id, stat_id, team_stat_tactique_id),
    FOREIGN KEY (team_season_id)
        REFERENCES team_seasons (id) ON DELETE CASCADE,
    FOREIGN KEY (stat_id)
        REFERENCES stats (id) ON DELETE CASCADE
);

--
-- Players stats
-- =============

CREATE TABLE player_seasons (
    id NUMBER,
    player_id NUMBER NOT NULL,
    year NUMBER NOT NULL,
    team_id NUMBER NOT NULL,
    PRIMARY KEY (id),
    CONSTRAINT player_season_unique UNIQUE (player_id, year),
    FOREIGN KEY (player_id)
        REFERENCES players (id) ON DELETE CASCADE,
    FOREIGN KEY (team_id)
        REFERENCES teams (id) ON DELETE CASCADE
);

CREATE SEQUENCE player_seasons_seq
    START WITH 1
    INCREMENT BY 1;

CREATE TABLE player_season_types (
    id NUMBER,
    name VARCHAR (31),
    PRIMARY KEY (id),
    UNIQUE (name)
);
```

```
CREATE SEQUENCE player_season_types_seq
    START WITH 1
    INCREMENT BY 1;

CREATE TABLE player_stats(
    player_season_id NUMBER NOT NULL,
    stat_id NUMBER NOT NULL,
    player_season_type_id NUMBER NOT NULL,
    conference_id NUMBER NULL, -- for all star games only
    gp NUMBER,
    minutes NUMBER,
    PRIMARY KEY (player_season_id, stat_id, player_season_type_id),
    FOREIGN KEY (player_season_id)
        REFERENCES player_seasons (id) ON DELETE CASCADE,
    FOREIGN KEY (stat_id)
        REFERENCES stats (id) ON DELETE CASCADE,
    FOREIGN KEY (player_season_type_id)
        REFERENCES player_season_types (id) ON DELETE CASCADE,
    FOREIGN KEY (conference_id)
        REFERENCES conferences (id) ON DELETE CASCADE
);
```

1.4 Conclusion

As we decided to go with Ruby on Rails, we expect future changes to be mainly imposed by any limitations that ActiveRecord has.