DBF Implementation  
for Action Soccer Manager

# Procedures and processes involved

In the process of balancing teams, the goal is to make the resulting teams that play against each other have a more competitive game, rather than one team overpowering all the other teams. This helps the game be more suspenseful and enjoyable.

In this process, the match organizer will do one of the following to the players:

* Place in a new team
* Transfer to a different team
* Leave them, allowing the player to remain in their current team

In a team, there has to be a captain as a liaison between the match officials and team players. Since a player can be transferred to a new team – including team captains – a new captain would need to be appointed.

A team may be disbanded, but records of its performance should still be kept. This can result in a player not being a part of any team for a moment in time, like a ‘free agent’.

The following business rules further explain the procedure for helping the match officials make better decisions to improve in team balancing.

## Business rules

A USER must have LOGIN credentials. The LOGIN credentials include the user’s unique username (LOGIN\_UNAME) and is verified by their password (LOGIN\_PASS). All users are uniquely identified by their user ID (USER\_ID).

A USER is either a SPECTATOR or is further classified as either a PLAYER or MATCH\_OFFICIAL. This is indicated by their USER\_TYPE. A USER is further characterized by their first name (USER\_FNAME), last name (USER\_LNAME), their date of birth (USER\_DOB) and their age (USER\_AGE). Their age is continuously calculated using their date of birth to remain up to date. They may also upload a profile picture (USER\_PICTURE).  
A USER has only one set of LOGIN credentials, and a set of LOGIN credentials can be associated with only one user. The relationship between the USER and LOGIN entities is 1:1 (mandatory one-to-one).

A PLAYER is a specific type of USER who participates in the football matches. They are uniquely identified by their user ID (USER\_ID), and are further characterized by their dominant foot they use to play with (PLAYER\_FOOT), the total number of games they have played (PLAYER\_GAMES\_PCOUNT), the total number of games they have won (PLAYER\_TWINS), the total number of games they have lost (PLAYER\_TLOSSES), their overall average rating (PLAYER\_AVG\_RATING), and the team they are currently playing for (TEAM\_ID). All these attributes (excluding their USER\_ID, PLAYER\_FOOT and TEAM\_ID) need to be updated after each match to remain current.  
Since a PLAYER is a specific type of USER, and a USER is a generalized PLAYER, the relationship between the USER and PLAYER entities is 1:1 (mandatory one-to-one).  
A PLAYER can be the captain of no TEAM or at most one TEAM (the one they are currently playing for). A TEAM can have none or one PLAYER as its team captain. The relationship between the PLAYER and TEAM entities is 1:1 (optional one-to-one).

A MATCH\_OFFICIAL is the only type of USER who may officiate the TEAMs and GAMEs played between teams. They are also uniquely identified by their user ID (USER\_ID). They can either be a referee, match organizer, or both. Only referees (MOF\_REFEREE) may record the results from a GAME they umpired in. Only match organizers (MOF\_ORGANIZER) may create a TEAM and pair them with another TEAM to play each other in a GAME.  
Since a MATCH\_OFFICIAL is a specific type of USER, and a USER is a generalized MATCH\_OFFICIAL, the relationship between MATCH\_OFFICIAL and USER entities is 1:1 (mandatory one-to-one).  
A MATCH\_OFFICIAL – who is a referee – can referee none or multiple GAMEs. A GAME is umpired by only one MATCH\_OFFICIAL – that is a referee. The relationship between the MATCH\_OFFICIAL and GAME entities is 1:M (optional one-to-many).  
A MATCH\_OFFICIAL ­– who is an organizer – can create none, or multiple TEAMs. A TEAM is created by only one MATCH\_OFFICIAL – who is an organizer. The relationship between the MATCH\_OFFICIAL and TEAM entities is 1:N (optional one-to-many).

A TEAM is uniquely identified by its team ID (TEAM\_ID). It is described by its team name (TEAM\_NAME), the date it was founded (TEAM\_FOUNDED), the total number of games it has played (TEAM\_PCOUNT), the number of wins (TEAM\_WINS), losses (TEAM\_LOSSES) and draws (TEAM\_DRAWS) it has had. It is further described by its total goal difference (TEAM\_G\_DIFF), the total amount of goals for (TEAM\_G\_FOR) and against (TEAM\_G\_DIFF) the team. A TEAM can have a team captain, identified by their user ID (USER\_ID). A TEAM can also be characterized by its team logo (TEAM\_LOGO).  
Two TEAMs may play no GAME or multiple GAMES against each other. In turn, a GAME is played by only two teams. The relationship between the TEAM and GAME entities is 2:P (optional many-to-many).   
A TEAM consists of none or up to 5 PLAYERs and a PLAYER plays for either none, or only one TEAM. The relationship between the TEAM and PLAYER entities is 1:5 (optional one-to-many).

A GAME is uniquely identified by a combination of the game’s ID (GAME\_ID) and the MATCH\_OFFICIAL umpiring the GAME. It is initiated on a certain date (GAME\_DATE) at a certain time (GAME\_TIME). It consists of two teams competing against each other – identified by their TEAM\_IDs – and having a respective home/away score (GAME\_HOME\_SCORE and GAME\_AWAY\_SCORE).  
As described before, a GAME has to consist of only two teams, where two teams may play multiple games against each other. The relationship between GAME and TEAM entities is P:2 (optional many-to-many).  
Also described above, the relationship between the MATCH\_OFFICIAL and GAME entities is 1:M (optional one-to-many).

# Initial ERD

# Recognition of issues

## M:N Relationships

One problematic issue is that the TEAM and GAME entities that have a many-to-many relationship with each other. This will cause using and maintaining the database to be problematic.

## NULL values

The following events will result in an attribute being NULL:

* A TEAM has no team captain, when it is disbanded
* A PLAYER is not part of a team, i.e. A player is a ‘free agent’

Since we know the reason for these values being null, it would be more appropriate if they were associated with a flag indicating these cases.

## Multivalued attributes

We have ensured that there are no multivalued attributes, such as the date and time when a game is played, and whether a match official is a referee, organizer or both, by splitting these attributes into two fields.

## Historical data

Since we would like to balance the teams better, we would need to keep track of a PLAYER’s movements in between TEAMs and their detailed performance in a match – such as the position they played in and their rating for that game.  
It would also be wise to keep track of which MATCH\_OFFICIAL created a TEAM, as we can keep track of whose judgment was best.

## Normalization

### 1NF

The database is not yet in the first normal form as there are possible attributes which can be NULL, as stated above.

# Solution and updated ERD

## M:N Relationships NULL values and Historical data

To address the many-to-many relationships, NULL valued attributes and historical data, two bridging entities need to be implemented.

### MATCH entity

A MATCH entity will be identified by the GAME\_ID, TEAM\_ID and USER\_ID of the referee of a specific match. This will allow a single score to be captured (TEAM\_SCORE) for a team playing in a specific GAME.

A TEAM can play none or multiple MATCHes. A MATCH describes only one TEAM. The relationship between the TEAM and MATCH entities is 1:P (optional one-to-many).

A GAME consists of two TEAMs MATCHed against each other, and only two MATCHed TEAMs describe only one GAME. The relationship between the GAME and MATCH entities is 1:2 (mandatory one-to-many).

### MATCH entity

A PLAYER\_ARCHIVE will consist details of all the games a PLAYER has participated in. It is uniquely identified by a combination of the GAME\_ID, TEAM\_ID and USER\_ID of the PLAYER. It consists of the position the PLAYER played in a GAME (PARC\_POSITION), the number of goals scored in that GAME by the PLAYER (PARC\_GOALS\_SCORED) and the player’s rating for that GAME (PARC\_RATING). It will also indicate whether that player was a captain (PARC\_CAPTAIN).

A PLAYER may have no PLAYER\_ARCHIVE, i.e. not played a single game, or have multiple PLAYER\_ARCHIVEs. A PLAYER\_ARCHIVE describes only one PLAYER’s performance in a GAME. The relationship between the PLAYER and PLAYER\_ARCHIVE is 1:S (optional one-to-many).

A GAME is further detailed by only ten PLAYER\_ARCHIVES – since a game consists of ten players. A PLAYER\_ARCHIVE describes the player’s performance in only one GAME. The relationship between the GAME and PLAYER\_ARCHIVE entities is 1:10 (mandatory one-to-many).

A TEAM may have no PLAYER\_ARCHIVE – as it may be newly constructed or nor games have been organized with this team – or multiple PLAYER\_ARCHIVEs. A PLAYER\_ARCHIVE describes the player’s performance for only one TEAM. The relationship between the TEAM and PLAYER\_ARCHIVE is 1:R (optional one-to-many).

## Normalization

I believe the database is of the 2nd Normal Form for the following reasons.

### 1NF

All Primary Keys identified are unique. There are no repeating groups of attributes within an entity.

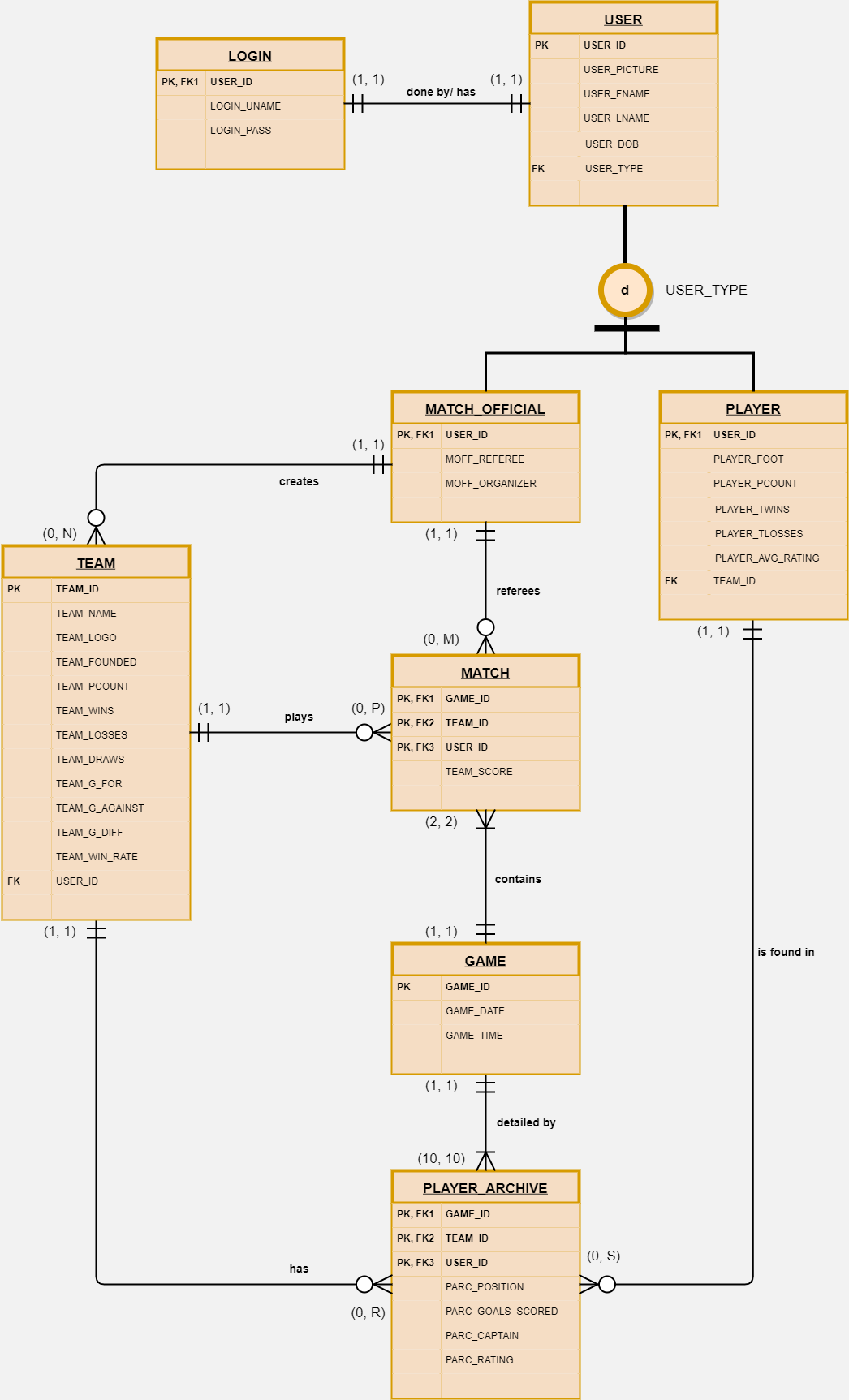
2NF

All non-key attributes are functionally dependent on their Primary Keys.

### 3NF

I do not feel that the database is Normalized to the 3rd Normal Form, as there are attributes which are calculated, that are dependent on the non key attributes of the entity – such as TEAM\_PCOUNT, PLAYER\_PCOUNT. Aside from this, if there were no such calculated fields, this database would be easily identifiable to be in the 3rd Normal Form

## Final ERD



# Implementation

## Tables

## Functions

## Procedures

## Views

## Triggers