CIS 761- Project Database Design

E/R Diagram

Please see the attached E/R diagram png file. Weak entity sets and their relationships are denoted as dashed lines. Additionally, exactly one relationships are denoted as a line with two dashes through them.

Relations

Users(uid, username, password, first_name, last_name, created_on, favorite_team_name, favorite_athlete_id)

- favorite_team_name is a foreign key referencing Teams.team_name
- favorite_athlete_id is a foreign key referencing Athletes.athlete_id
- username is a unique key

Teams(team_name, location, abbreviation, venue_name, primary_color, secondary_color)

- venue_name is a foreign key referencing Venues.venue_name

Venues(venue_name, capacity, city, state, grass, indoor)

Games(game_id, date, attendance, home_team_name, away_team_name, venue_name, utc_time)

- date is a foreign key referencing Season_dates.date
- home_team_name is a foreign key referencing Teams.team_name
- away_team_name is a foreign key referencing Teams.team_name
- venue_name is a foreign key referencing Venues.venue_name
- date, home_team_name, away_team_name is a unique key

Season_dates(date, season_year, season_type, week)

Athletes(athlete_id, first_name, last_name, dob, height, weight, birth_city, birth_state)

Positions(position_name, abbreviation, platoon)

Rosters(team_name, athlete_id, position_name, start_date, end_date)

- team_name is a foreign key referencing teams.team_name
- athlete_id is a foreign key referencing players.player_id
- position_name is a foreign key referencing positions.position_name

Linescores(team_name, game_id, quarter, score)

- team_name is a foreign key referencing teams.team_name
- game_id is a foreign key referencing games.game_id

Plays(play_id, quarter, yards, score_value, play_type, text, seconds_remaining, start_down, end_down)

Player_Plays(play_id, player_id, game_id, type)

- play_id is a foreign key referencing plays.play_id
- player_id is a foreign key referencing Athletes.athlete_id
- game_id is a foreign key referencing Games.game_id

Functional Dependencies

Users:

$uid \rightarrow username, password, first\_name, last\_name, created\_on, favorite\_team\_name, favorite\_athlete\_name$

$username \rightarrow uid, password, first\_name, last\_name, created\_on, favorite\_team\_name, favorite\_athlete\_name$

Teams:

$team\_name \rightarrow location, abbreviation, venue\_name, primary\_color, secondary\_color$

$abbreviation \rightarrow location, team\_name, venue\_name, primary\_color, secondary\_color$

Venues

$venue\_name \rightarrow capacity, city, state, grass, indoor$

Games

$game\_id \rightarrow attendance, date, utc\_time, home\_team\_id, away\_team\_id, venue\_name$

$home\_team\_id, away\_team\_id, date \rightarrow game\_id, attendance, utc\_time, venue\_name$

Season_Dates

$date \rightarrow season\_year, season\_type, week$

Athletes

$athlete\_id \rightarrow first\_name, last\_name, dob, height, weight, birth\_city, birth\_state$

$first\_name, last\_name, dob, birth\_city, birth\_state \rightarrow athlete\_id, height, weight$

Positions

$position\_name \rightarrow abbreviation, platoon$

$abbreviation \rightarrow position\_name, platoon$

Rosters

$team\_name, athlete\_id, start\_date \rightarrow position\_name, end\_date$

$team\_name, athlete\_id, end\_date \rightarrow position\_name, start\_date$

Linescores

$team\_name, game\_id, quarter \rightarrow score$

Plays

$play\_id \rightarrow quarter, yards, score\_value, play\_type, text, seconds\_remaining, start\_down, end\_down$

Player_plays

$play\_id, player\_id \rightarrow game\_id, type$

BCNF

Yes, all of the relations in the schema are in BCNF. Each of the functional dependencies listed is a superkey.

Part E- Is there anything we don't like about the schema?

As it stands, we are happy with the schema as it stands. Originally the data contained numerical IDs for teams, venues, and position, which had the potential to result in duplicate entries. However, since position, team and venue names are unique, we decided to remove the numeric ID and use team/venue name as the key. The original data also had birth place as a string of the form CITY, STATE, which is not an optimal way to store this data. Therefore, we split the string and created an attribute for city and state separately. This will avoid string manipulation when working with this data.

Additional Notes

1. We assume the functional dependency $city \rightarrow state$ does not hold (ex. Kansas City in MO or KS)
2. While a single attribute (city, state) may be used instead of two attributes, the modelling we used is convienent as it allows us to more easily use the data and does not violate BCNF.
3. We may make small changes to the database design as we collect additional data.