Database Systems: Final Project Proposal

Mark Hudnall Sam Konowitch

April 2012

Domain

For our final project, we hope to tackle the domain of athletics software. In particular, we wanted to provide a fitness tracking web application for sports teams. We want to expand our initial fitness tracking vision to fully model a sports team hierarchy (including positions, captains, and coaches). We plan to provide specialized views for coaches, athletes on a team, and independent athletes (not in a sport, but wanting to track fitness regardless) that provide the relevant information for each role in a digestible way.

Key Features

At the base of our application, we hope to provide an easy, web-based system for tracking fitness and workouts. For coaches and athletes, we hope to model a team as close to reality as possible, including relevant team information like positions and captains. Coaches and captains will be able to set workouts for the week, while players can log their progress and view summary statistics. If we have extra time, we hope to expand our fitness tracking and team modeling to include in game statistics, which would be sport specific. We also hope to provide support for professional athletes, who have contracts and receive salaries.

Expected Results

By the end of the project, we hope to present a polished web application. The web application will provide a login system. Coaches register and add players, or players can register as independents. Coaches and captains will have an interface to specify required workouts. Players will be presented with a view of upcoming and previous workouts. Players can log their own progress, as well as view the progress of other members of the team (a team cannot view another teams progress). Coaches will be able to designate roles for players on his/her team, like positions or captain status. This will determine what information players can view in the webapp.

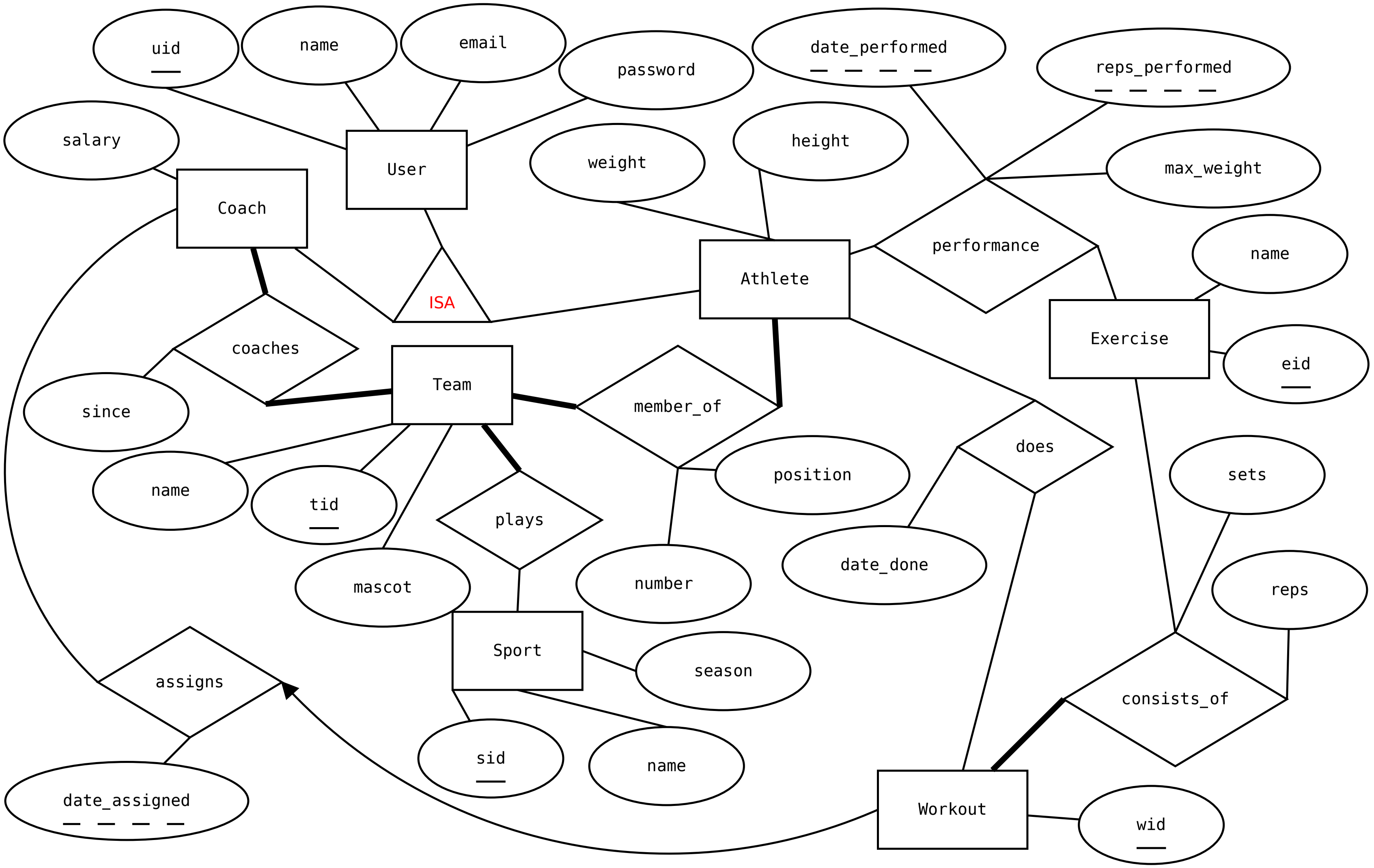
Tentative Schedule

Date	Goal	Status
April 2	ER Diagrams finalized	completed
April 6	Relations finalized	completed
April 9	Normalization complete	in progress
April 11	`schema.sql`	to do
April 12	Application scaffolded	to do
April 16	Coach workflow implemented	to do
April 23	Player workflow implemented	to do
April 26	App tested and feature complete	to do

Technologies

We plan to use MySQL as our DBMS. We will be using Python as our server side language, for both web development and database interfacing (via the mysql-python module). We are currently planning to use the Flask web framework. We are both comfortable with Python, and the Flask framework is very easy to use, which will let us focus more on the database portion of the assignment. `mysql-python` is essentially identical to `JDBC`.

ER Diagram


uid
name
email
password
date_performed
reps_performed
salary
User
weight
height
max_weight
Coach
performance
name
ISA
Athlete
Exercise
eid
coaches
Team
member_of
since
does
position
sets
name
tid
plays
date_done
mascot
number
reps
Sport
season
assigns
consists_of
sid
name
date_assigned
Workout
wid

Relational Model

```
User(*uid, name, email, password)
Coach(*uid, salary)
Athlete(*uid, weight, height)
coaches(*uid, *tid, since)
Team(*tid, name, mascot)
Sport(*sid, name, season)
plays(*tid, *sid)
member_of(*uid, *tid, position, number)
Workout(*wid)
Exercise(*eid, name)
consists_of(*wid, *eid, sets, reps)
assigns(*uid, *wid, *date_assigned)
does(*uid, *wid, date_done)
performance(*uid, *eid, *date_performed, *reps_performed, max_weight)
```

* denotes a primary key.

1 Functional Dependencies

We have identified three violations of BCNF in our relational model.

First, let us consider the Team relation. We observed that a team with a given name will have a given mascot (for instance, a team named “The Ducks” will have a mascot “Duck”, while a team named “The Lions” will have a mascot “Lion”). Thus there is a functional dependency from name to mascot. Thus we decompose Team into two relations: Team(tid, name) and TeamMascot(name, mascot). This eliminates the functional dependency in Team. For TeamMascot, we have a FD from name → mascot, but since name is a key, the relation is in BCNF. Thus both relations are in BCNF.

Next we consider the Sport relation. We observed that for the sports we deal with, the same sports are always in the same season. Thus there is a functional dependency from a sport’s name to its season. Thus we decompose Sport into two relations: Sport(sid, name) and SportSeason(name, season). This eliminates the FD in Sport. SportSeason has a FD from name to season, but since name is a key, SportSeason is in BCNF. Thus both relations are in BCNF.

Thus our final set of relations is:

```
User(*uid, name, email, password)
Coach(*uid, salary)
Athlete(*uid, weight, height)
coaches(*uid, *tid, since)
Team(*tid, name)
TeamMascot(*name, mascot)
```

```
Sport(*sid, name)
SportSeason(*name, season)
plays(*tid, *sid)
member_of(*uid, *tid, position, number)
Workout(*wid)
Exercise(*eid, name)
consists_of(*wid, *eid, sets, reps)
assigns(*uid, *wid, *date_assigned)
does(*uid, *wid, date_done)
performance(*uid, *eid, *date_performed, *reps_performed, max_weight)
```