Project 3: VolleyDB Application

CMPE 321, Introduction to Database Systems, Spring 2024

Due: 06 May 2024, 23:59 o'clock

1 Introduction

Last year, our national volleyball team, which became the European champion on the 100th anniversary of the Republic of Türkiye, made us all proud. Of course, the best team in the world cannot stay away from technological developments and must systematically manage its players' data. A well-designed database system is essential to ensure the reliability, consistency, and security of such systems. Turkish Volleyball Federation needs your help with this database design!

This project aims to develop a database application for the VolleyDB design.

2 Project Description

In project 1, you performed conceptual database design, drew ER diagrams, and converted these ER diagrams into relational tables for VolleyDB. In this project, you will go one step further and implement the VolleyDB database with a user interface (UI) using the structure you have designed in Project 1 or its improved version. Refining your schema using what you learned in the lectures is **strongly** advised. VolleyDB should contain the following information:

1. **Database Managers** These are the administrators of the database application that you are gonna create, they only have a username and password (which is provided to you). Their abilities will be given in the requirements section. No new managers can be added out of the already given ones.
2. **User** includes the following attributes; *username, password, name, surname.* Each user has a unique username. Also each user is either a player or a coach or a jury. Usernames must be different than preexisting database managers' usernames.
   (a) **Players** additionally have attributes of *date_of_birth*, *height* and *weight.* A player must be registered with at least one team. This information can be found at the PlayerTeams table. A player can be registered to different types of teams, but of course cannot play in matches where there are time conflicts. In addition, a player can play in different positions. There must be at least one position in which the player can play. However, in a match, he/she can only play in a position within the positions he/she can play (is

registered). The registered positions can be found at PlayerPositions table. The squad information (players that have played in a specific session) can be found at SessionSquads table. There are no age-team restrictions.

(b) **Coaches** additionally have *nationality* information. Each coach **must** have **only one** *nationality*. That is, dual citizenship is not allowed, and being in an agreement with more than one team at a time is not possible for coaches. Also, each team **must** be directed by a unique coach.

(c) **Juries** additionally have *nationality* information. Each jury **must** have **only one** *nationality*. That is, dual citizenship is not allowed. They rate match sessions. Every match has an assigned jury, and that assigned jury rates a match session only once. Juries can not rate matches that they are not assigned to. They can not edit/change their ratings.

3. **Position** includes the following attributes: *position_ID, position_name*. position_id must be unique. Normally, a player can play in different positions, but in a match, he/she has to play in just one position.

4. **Team** includes the following attributes: *team_ID, team_name, coach_username, contract_start, contract_finish, channel_ID*. team_ID must be unique, and each channel_ID is associated with a unique channel_name. Each team has an agreement with just one TV channel. Each team is led by just one coach. contract_start and contract_finish are used for coach-team agreements. A coach cannot direct more than one team at the same time. A TV channel can have agreements with more than one team.

5. **PlayerPositions** include the following attributes: *player_positions_ID, username, position*. Each player can be registered to multiple positions, at least 1.

6. **PlayerTeams** include the following attributes: *player_teams_ID, username, team*. Each player can be registered to multiple teams, at least 1.

7. **SessionSquads** include the following attributes: *squad_ID, session_ID, played_player_username, position_ID*. Each player can be registered to multiple teams, at least 1. For every session_ID, there must be 6 entries(rows) in this table, meaning a squad is created with 6 different players. Each of this 6 players must be registered to the same team that plays in that specific session. And each player can only play in a single position in his/her squad, and that position **must** be one of his/her registered positions. The entire team does NOT have to play in a session, 6 players is the only acceptable player amount for a session. There are no age or position restrictions when creating a squad.

8. **Match Sessions** include the following attributes: *session_ID, team_ID, stadium_id, stadium_name, stadium_country,*
time_slot, date, assigned_jury_username, rating. Each stadium_ID is associated with a unique stadium_name. The session_ID must be unique. We are only interested in the teams of our federation. So no opponent information or details are necessary.

- No two match sessions can overlap, both in terms of stadium and playing time.
- There are four time slots for each day.
- The duration of the match is closely related to the time slots. The time_slot attribute determines the starting time of the match, and the end time is determined by the duration. Each match has a duration of 2 time slots. (For example, if a match starts at time slot 2 and has a duration of 2, then the stadium is reserved for that match during the following time slots: [2, 3]).
- Each stadium_id corresponds to a physical location. Hence, stadium_name and stadium_country depend solely on the stadium_id.
- A jury can rate the same match only once and cannot edit.
- Each match will be rated by a jury assigned to that match.

Important Notice:

There will be **pre-existing** records that may be false (out of contract match sessions, out of team squads etc), they **must** be initially available at your databases **but no new records** can be inserted/added to the database. (tip:triggers)

3 Schema refinement and normalization

You should analyze your design from Project 1 in terms of functional dependencies (FDs) and normal forms, then, refine your design if needed. You should explicitly list all of the non-trivial FDs. Then, for each relation you should determine if it is in Boyce-Codd Normal Form (BCNF) and you should explain how the requirements of BCNF are met (or not met) in terms of FDs. If a relation is not in BCNF, you should check whether it is 3NF and explain how the requirements for 3NF are met (or not met). If a relation is not in BCNF, you should either decompose it into BCNF relations or provide a justification if you decide not to decompose it. If you decompose a relation, you should explain whether the decomposition is lossless-join and dependency preserving. It is possible that your initial schema is already in BCNF. If this is the case, you still need to explain how the requirements of BCNF are met in terms of FDs for each one of your relations.

In addition, you are expected to refine your design by capturing the constrains that you were not able to handle in Project 1 by using the CHECK construct and any other means possible (triggers, procedures etc.). Please describe which additional constraints you were able to handle by using CHECK.

4 Requirements

Your UI must support the following operations:

1. Database managers and users shall be able to log in to the system with their credentials (username and password).
2. Database managers shall be able to add new Users (Players/Juries/Coaches) to the system. (So no signup page is required.)
3. Database managers shall be able to update *stadium_name* according to the wills of the politicians. When a stadium is updated, all the preexisting records with that name should be changed.
4. Coaches shall be able to delete match sessions by providing session_ID. When a match session is deleted, all data regarding that match session must be deleted including the rating, date, stadium etc. Also, the squad info of that match session should be deleted.
5. Coaches shall be able to add a new match session, he/she can only put his/her current team_ID. Stadium info and date, time, timeslot info are up to the coach's choice but they should not be conflicting. You should check for any type of conflict with triggers. Also coach can choose(assign) his/her own session's assigned jury (by jury's name and surname). The rating of the newly added session should be left blank or null at first, till a jury logs in and rates the match.
6. Coaches shall be able to create a squad for his/her newly created session (however a new session can exist without a declared squad.). All the players that the coach chooses for his/her squad should be from the coach's current team. Coach shall be able to create squad using player names.
7. Coaches shall be able to see a list of all existing stadiums names and their countries.
8. Juries shall be able to view the average rating of all sessions that he/she rated also the count of total rated sessions by him/her.
9. Juries shall be able to rate a session that are assigned to them only if they haven't rated that session yet and if the current date (like the date of the Demo :)) is after the date of the specific match session.
10. Players shall be able to view all of the players' names and surnames that he/she has played within a session at least once. Also a player should see the height of the player that he/she played with the most. If there are more than one players that he/she played the most, he/she should see the average height of those players.

5 Notes

- In the scope of this project, we are more interested in the functionality of the web interface, rather than the styling of UI. In other words, it is totally fine to create a system that satisfies all requirements and has zero line of CSS and Javascript.
- The allowed languages are PHP, Java, JavaScript, and Python. You can use a framework, however, you **must** write the SQL queries and boot the database server yourself. Note that you should set up the database and create the tables on your own. You are **not** allowed to use any tool that helps with these parts such as ORM(Object Relational Mapping).
- Also, do **not** use database connector as your query supplier. Use it only as a query executor by running it with SQL strings written by you.
- You are not expected to deploy your system. So, it is **fine** if it runs in your local.
- Only MySQL is allowed.
- There will be a demo session in which we will want you to show some specific requirements of your design, like some tasks as a user of your application. The demo date and details will be announced afterwards.
- Demo session will take %70 percent of your grade and %30 will be based on Part 3 of this documentation.
- You will also need to submit all your code files, which will not be graded, but simillarity-checked.

6 Submission

This project can be implemented either individually or as a team of two people. You are free to change teams in the upcoming projects. The submission must include your **code**, your **description of the schema refinement step, your updated database design and ER diagram**. Place all the required files into a folder named with the student IDs of the team members separated by an underscore (e.g., 2017400200_2018700120). Zip the folder for submission and name the .zip file with the same name. Submit the .zip file through Moodle until the deadline. **Any other submission method is not allowed**. Each group should submit **one** .zip file. The system will be evaluated during a demo session that will be arranged. Demo day(s) will be announced. You **must** add the entries provided in the excel sheet (the sheet is exactly same as the data for Project 2, just the database managers are added) **before coming** to the demo.

7 Late Submission Policy

We will accept late submissions, however;

- One day late (even one minute will be considered a day late) would mean -30 points penalty.
- Two days late (even one minute and a day will be considered two days late) would mean -60 points penalty.
- Moodle will close after two days. No other submission method will be accepted.

8 Academic Honesty

Please read carefully the academic honesty part of the syllabus as we give utmost importance to academic honesty.