CS 5/7330 Fall 2022

Project – NoSQL Databases

The goal of this project is for you to get some exposure of NoSQL database by developing a small application using either MongoDB or Neo4j. You can use any programming language to develop your program.

Application – Sports League Database

I want to build a database to keep track of sports league. There are a few key pieces of information that need to be stored:

League

A sport league has a unique name. It also has a commissioner (we need to store his/her name and SSN).

Teams

Each league has a set of teams associated with it. For each team, we need to store the following information:

- Name of the team (assume it is unique)
- City where the team is located (for this project, assume all teams is in the United States). Notice that a city may have more than one team.
- The name of the field/court that it plays in. (Notice a team may have NO home field/court). Notice that the name of the court is unique only within a city. (i.e. There may be a "cowboys stadium" in Dallas, TX, but also a "cowboys stadium" in Dallas, GA).

Seasons and Games

Each league has seasons. Each season has a start date and an end date. Within a season, teams within the league will play games against one another. Each season, each team is required to play a fix number of games (this can change from season to season).

Each game is between two teams. Each game has a location (which is the name of a field/court) and a date. After the game is played, the we need to enter the score of each team (which can be ANY floating point number, including negative numbers). For each game, the team that scores more is the winner.

Ratings

Each team has a numeric (floating point, can be negative) rating associated with it. The rating may change at any given time. For now, we will let the user to update ratings of a team.

Standings

We need to show the standings of league for a season (even the current season). For each season, we assign a certain number of points for a game won, drawn, lost respectively, and calculate the total points for each team, and rank them accordingly. Ties are broken arbitrarily.

Tasks

You need to implement a system (with a GUI) that support the following tasks

- League data entry. The system should allow users to
  - Enter a new league (and initialize the first season)
  - Enter a new team (and assign team to a league)
- Season set up (this can also be used to initialize the first season)
  - Enter all information needed
  - Either
    - Enter game information (date, teams, location)
    - Allow the system to randomly generate the schedule
- Entering game results, and update ratings (if desired).
- Update the current date, and apply the all actions that is needed (e.g. entering results up to that date, initializing a new season etc.)
- Move teams:
  - Teams can be moved from one league to another between seasons. However, both the league that is it moving from and to cannot be in the middle of a season.
- Queries
  - League queries
    - Return basic information of a league (e.g. name, commissioner, number of seasons)
    - Champions: given a league, return the team with the highest point each season (if there is a tie, then print all champions). Print the year, champions, and its record.
  - Team queries
    - Given a team, return its basic information (name, city, home court, current rating).
    - Given a team, return its record for each season (games played, # of wins, # of draws, # of loses, sum of scores for its games, sum of its opponent scores' in games, and the total number of points)
  - Game queries
    - Given two teams, list all games that they played, and score (if available – some games' score may not have been entered). List the date and the league where they are competing also.
  - Season queries
    - Given a league and a season, print the standing of teams, ordered.
  - Rating query (Extra credit)
    - Given a league, and a team, find a list of team t1, t2, t3,… tk as follows:
      - ti should have beaten t(i+1) in the current season.
      - ti's rating should be less than (or equal to) t(i+1)'s rating.

Due Date

Each group should make a 10 minute presentation that demo your system. It can be either done in class on either 11/29 or 12/1, or you can make a video and submit to me on or before 11/29 (Mon) at 11:59pm and I will play it during one of the two lectures I will make comments and ask you to do certain things. You will than have a few days to make final changes. The project is due 12/5 (Mon) 11:59pm.

Notice that the majority of the grade will be based on what have been achieved by the presentation. You will need to provide resources needed to run your program (except materials/code/libraries that can be downloaded from other sites). You should also provide a 2-4 page user manual, and a 2-4 page developer manual (contain enough information for other people to continue your task).