Databases Exercise 6: PL/pgSQL

Due date: Thursday 15.12.16, 23:55

Submission instructions:

- This exercise consists of programming in PL/pgSQL, and should be submitted electronically, via the submission link on the course homepage. Please run your code one more time just before submission to check that it works! Submit your exercise as a **zip** file containing all the necessary files.
- Each question should be independent of the others. That means if you've written code you need in multiple questions, you should write it in each of your files.
- You should also hand in a file called **drop.sql** that deletes all functions and triggers created from all the questions (Use if exists: drop trigger if exists q1 on match;).

In this exercise you will add some triggers to the following gaming database. (You are provided with a schema for the database in an sql file to use. This is a short description of the attached schema.)

- Player(pid, firstname, lastname, nickname, rating, inactive)
  - This table contains information about players.
  - **Pid** is an auto incremented id for the players.
  - **Firstname, lastname** and **nickname** are the player's first name, last name and nickname.
  - **rating** is an integer representing the player's current rating.
  - **inactive** indicates whether or not the player is inactive. Default is 0 and indicates an active player. The value is 1 when the player is inactive.
- Location(locid, name)
  - This represents a location at which a match can take place.
  - **Locid** is an auto incremented id for the locations.
  - **Name** is the location's name.
- Match(matchid, winnerId, loserId, winnerscore, loserscore, locationid, time)
  - This represents a match. Either a ranked match (for which the scores are kept) or an unranked one in which case both winnerscore and loserscore will be NULL.
  - **matchid** - autoincrement id for match

  - **winnerId** - Player id of the winner.
  - **loserId** - Player id of the loser.
  - **locationid** - Location id of the location at which the match took place.
  - **winnerscore** - The winner's score. Null if it is not ranked.
  - **losescore** - The loser's score. Null if it is not ranked.

- Top(<u>pid</u>)
  - This contains the players with the highest rating.
  - **pid** - id of player.

Question 1

In a file named **q1.sql** write a trigger that does the following:

Whenever a ranked match is added, add 1 point to the winner's rating and subtract one from the loser. If, at this point, the loser has -5 points, set the loser inactive value to 1.

Question 2

In a file named **q2.sql** write a trigger that does the following:

Before adding a match, check that the both players are active (inactive value is 0 means the player is active). If one of the players is inactive, raise an exception with the message "Inactive players may not play".

Question 3

In a file named **q3.sql** write a trigger that does the following:

Write a trigger that ensures that the table Top always contains the top-1 player(s). In other words, the table Top should always contain the player with the highest rating (and in the case of a tie, will contain all players with the highest rating).

Question 4

In a file named **q4.sql** write a trigger that does the following:

In this question, you'll enforce the availability of playing locations. When all available locations are deleted (i.e. none are left as a result of a delete command), add a new location called "Always Available". Do not give it an id, as the autoincrement should take care of that.

Question 5

In a file named **q5.sql** write a function that does the following:

Once in a while, we might wish to reactivate inactive players.

Write a function called **reactivate** that increases the rating of all inactive players by 2. For every player that is inactive and has at least 0 as ranking, change his status to active (by setting inactive = 0).

Good luck!