CS-3200 Homework

SQL procedures, functions, triggers and prepared statements in MySQL.

This assignment gives you an opportunity to create stored procedures functions, triggers and prepared statements from queries you created for the starswars schema. There is no starter file for this assignment. You should complete this assignment given the starwarsfinal schema provided to you for homework 4.

Please submit one .sql file to blackboard containing the SQL code for each question named LastnameFirstNameHwk5.sql where Lastname = your last name, and Firstname = you first name. The .sql file should be broken into a section per question. Each section starts with a comment that lists the question number as well as the question description and any other comment you believe help describe the solution. This is followed by the solution. The solution is followed by the code that runs the solution, make sure you provide different executions of the solution. Each question is worth 10 points.

1. Write a procedure track_character(character) that accepts a character name and returns a result set that contains a list of the movie scenes the character is in. For each movie, track the total number of scenes and the planet where the character appears. The result set should contain the character's name, the planet name, the movie name, and the sum of the movie scene length for that specific planet in that movie for that character.

2. Write a procedure track_planet(planet) that accepts a planet name and returns a result set that contain the planet name, the movie name, and the number of characters that appear on that planet during that movie.

3. Write a function named planet_hopping(character). It accepts a character name and returns the number of planets the character has appeared on.

4. Write a function named planet_most_visited(character) that accepts a character name and returns the name of the planet where the character appeared the most (as measured in scene counts).

5. Write a function named home_affiliation_same(character) that accepts a character name and returns TRUE if the character has the same affiliation as his home planet, FALSE if the character has a different affiliation than his home planet or NULL if the home planet or the affiliation is not known.

6. Write a function named planet_in_num_movies() that accepts a planet's name as an argument and returns the number of movies that the planet appeared in. Write code that uses the function under the function.

7. Write a procedure named character_with_affiliation(affiliation) that accepts an affiliation and returns the character records (all fields associated with the character) with that affiliation. Write code that calls the procedure.

8. Write a trigger that updates the field scenesinDB for the movie records in the Movies table. The field should contain the maximum scene number found in the timetable table for that movie. Call the trigger timetable_after_insert. Insert the following records into the database. Insert records into the timetable table that places 'Chewbacca', and ‘Princess Leia’ on 'Endor' in scenes 11 through 12 for movie 3. Ensure that the scenesinDB is properly updated for this data.

9. Create and execute a prepared statement from the SQL workbench that calls track_character with the argument ‘Princess Leia’. Use a user session variable to pass the argument to the function.

10. Create and execute a prepared statement that calls planet_in_num_movies() with the argument ‘Bespin’. Once again use a user session variable to pass the argument to the function.