**CS 5200 Homework 7**

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 music schema. There is no starter file for this assignment. You should complete this assignment given the *Music* schema provided to you for homework 5.

Please submit one .sql file to canvas named Hwk7LastnameFirstInitial.sql where Lastname = your last name, and Firstname = you first letter of your first name. The file should contain the SQL code for each question named. 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 helps to describe the solution. This is followed by the solution. The solution is followed by test code that runs the solution, make sure you provide different executions of the solution.

1. Write a function num\_songs\_with\_genre(genre\_p) that accepts a genre name and returns the number of songs with the genre. (5 points)

2. Write a procedure get\_artists\_with\_label(label\_p) that accepts a record label name and returns a result set of all artist names and the corresponding label name. (5 points)

3. Write a procedure named song\_has\_genre(genre\_p) that accepts a genre name and returns a result set of the songs with that genre. The result should contain the song id , the song name, and the album name. If a genre is provided that is not found in the genre table, generate an error from the procedure stating that the passed genre is not valid and use SIGNAL to throw error ‘45000’. (10 points)

4. Write a function named album\_length(length\_p) that accepts one parameter, a count of songs and returns the number of albums with that length (5 points)

5. Write a procedure named get\_song\_details() that accepts a song name as an argument and returns the song name, the song id, the recording label, the album name, the genre name and the mood name. (10 points)

6. Write a function named more\_followers(artist1,artist2). It accepts 2 artist names and returns 1 if artist1 has more followers than artist2, 0 if they have the same number of followers , and -1 if artist2 has more followers that artist1. (5 points)

7. Create a procedure named create\_song( title\_p, artist\_p, record\_label\_p, mood\_p, genre\_p, album\_title) that inserts a song into the database . Make sure you create the appropriate tuples in the album and other required tables before attempting to insert the song. Also, ensure that the specified record label, genre name and mood name already exist in the database. If they do not exist, use SIGNAL with error number 45000. When adding a song, it can be associated with a known artist’s current existing album or the song could belong to a new album for the artist. Also, assume the producer of the song performs on the song. (HINT: The combination of album name and musician is unique for each album. ) Insert the following song into the song table.Title = “Me about You” , Artist = “The Turtles”, recording\_label = “Def Jam Recordings“ , genre = “Pop”, mood = “Calm”, album = “Happy Together”. Please also provide SELECT statements that verify the tuples have been inserted into the appropriate tables. (10 points)

8. Write a procedure named get\_songs\_with\_mood() that accepts a mood name and returns the song name, the mood name, mood description and the artist who released the song. (5 points)

9. Modify the artists table to contain a field called num\_released of type INTEGER and write a procedure called set\_num\_released\_count(artist) that accepts an artist name and initializes the num\_released field to the number of albums the artist has released. The artist table modification can occur outside or inside of the procedure but must be executed only once. (10 points)

10. Create a procedure named update\_all\_artists\_num\_releases( ) that assigns the artist.num\_releases to the correct value. The correct value is determined by the number of albums the artist has released. Use the procedure from problem 9 to complete this procedure. You will need a cursor and a handler to complete this procedure (5 points)

11. Write a trigger that updates the artist table when an album tuple is inserted into the database. The trigger will need to assign the correct value of albums released for the artist. Name the trigger artist\_update\_after\_insert\_album. Insert an album into the album table to verify your trigger is working; The album name = “Justice”, Artist = “Justin Beiber”. (10 points)

12. Write a trigger that updates the artist table when an album is deleted from the album table. The trigger will need to assign the correct value to the artist.num\_released field for the corresponding artist. Name the trigger artist\_update\_after\_delete\_artist. Delete an album from the album table to verify your trigger is working; The album name = “Justice”, Artist = “Justin Beiber”. (5 points)

13.Create and execute a prepared statement from the SQL workbench that calls the function more\_followers(artist1,artist2). Use 2 user session variables to pass the two arguments to the function. Pass the values “Vanilla” and “The Turtles” as the author values. (5 points)

14. Create and execute a prepared statement from the SQL workbench that calls the function num\_songs\_with\_genre(genre\_p) . Use a user session variable to pass the genre name to the function. Pass the value “Rock” as the length (5 points)