COSC 416A – BerkeleyDB – Lab 11

Answer Key:

1.
   ```
   CREATE TABLE games (id INTEGER, name VARCHAR(256), publisher VARCHAR(256), releasedate
   DATETIME, rating DOUBLE);
   CREATE TABLE players (id INTEGER, fname VARCHAR(256), lname VARCHAR(256), bdate
   DATETIME, sex VARCHAR(10), pic VARCHAR(256));
   CREATE TABLE playerGames (pid INTEGER, gid INTEGER, score INTEGER);
   ```

2.
   ```
   .mode tabs
   .import players.txt players
   .import games.txt games
   .import player_games.txt playerGames
   ```

3.
   ```
   SELECT gid, name, count(playerGames.pid) playerCount
   FROM games LEFT OUTER JOIN playerGames ON games.id == playerGames.gid
   GROUP BY gid
   ORDER BY playerCount DESC;
   ```

4.
   ```
   select highscores.gid, highscores.name, highscores.maxscore, P.id, P.fname, P.lname FROM
   playerGames PG, players P, (SELECT games.id as gid, name, max(score) as maxscore FROM
   playerGames, games WHERE playerGames.gid == games.id  GROUP BY games.id) as highscores
   WHERE PG.score = highscores.maxscore AND highscores.gid = PG.gid AND P.id = PG.pid;
   ```

5.
   ```
   SELECT pid, fname, lname, count(best) highScores FROM (select highscores.gid,
   highscores.name, highscores.maxscore, P.id, P.fname, P.lname FROM
   playerGames PG, players P, (SELECT games.id as gid, name, max(score) as maxscore FROM
   playerGames, games WHERE playerGames.gid == games.id  GROUP BY games.id) as highscores
   WHERE PG.score = highscores.maxscore AND highscores.gid = PG.gid AND P.id = PG.pid)
   GROUP BY pid
   HAVING highScores == 2;
   ```

PART 2: The sections that need to be filled out by the students are in **BOLD**

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.sql.*;

public class BerkeleyDB  {

       public static void main(String args[]) {

         //Set connection info
              String url = "jdbc:sqlite:/31317092";
              Connection con;
              Statement ddl, stmt, stmt2;
              PreparedStatement pstmt;

              //Load the driver class
              try {
                     Class.forName("SQLite.JDBCDriver");

              } catch(java.lang.ClassNotFoundException e) {
                     System.err.print("ClassNotFoundException: ");
                     System.err.println(e.getMessage());
              }

              try {
            //Create a connection to the db
                     con = DriverManager.getConnection(url, "myLogin", "myPassword");

                     //Create tables
                     ddl = con.createStatement();
                     ddl.executeUpdate("CREATE TABLE playerGames (pid INTEGER,
                                        gid INTEGER, score INTEGER)");
                     ddl.executeUpdate("CREATE TABLE topScores (pid INTEGER,
                                        gid INTEGER, score INTEGER)");

            //Read in player_games.txt content and insert into the database.
                     try
                     {
                              BufferedReader br = new BufferedReader(
                                        new FileReader("player_games.txt"));
                              String strLine;
                              int linenum = 1;
                              int totalLines = 100;

                                   pstmt = con.prepareStatement("INSERT INTO playerGames
                                   (pid, gid, score) VALUES (?, ?, ?)");

                              //Read File Line By Line
                              System.out.println("Building playerGames table from
file:");
                              double start = System.currentTimeMillis();
                              System.out.print("|          | 0%\r");
                              while ((strLine = br.readLine()) != null &&
                                   linenum<=totalLines)
                              {
                                     // Print the content on the console
                                     String[] values = strLine.split("\t");

                                     pstmt.setInt(1, Integer.parseInt(values[0]));
                                     pstmt.setInt(2, Integer.parseInt(values[1]));
                                     pstmt.setInt(3, Integer.parseInt(values[2]));
                                     pstmt.executeUpdate();
```

```
                        //Update progress bar
                        int bars = linenum/(totalLines/10);
                      int percent = linenum*100/totalLines;
                        String bar = "|";
                        for(int i=0; i<10; i++) {
                          bar += (i<bars)?"=":" ";
                        }
                        bar = bar+"| " + percent + "%\r";
                        System.out.print(bar);

                                       linenum++;
                                }
                                double total = System.currentTimeMillis()-start;
                                System.out.println("Done!          in "+total+"ms");
                                br.close();
                                  }
                      catch (Exception e){//Catch exception if any
                                System.err.println("Error: " + e.getMessage());
                                }

                      stmt =
con.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_READ_ONLY);
                      stmt2 = con.createStatement();

            //Query for selecing the top score for each player and the game id where
they got the score.
                      String selectScores = "SELECT pid, gid, max(score) FROM
playerGames GROUP BY pid";
                      ResultSet rs = stmt.executeQuery(selectScores);

            System.out.println("Inserting top scores into topScores table:");
                                double start = System.currentTimeMillis();
                                System.out.print("|          | 0%\r");
                              int linenum =1;
                              rs.last();
                              int totalLines = rs.getRow();
                              rs.first();
                      while (rs.next()) {
                              String insertTopScore = "INSERT INTO topScores VALUES (" +
rs.getInt(1) + ", " + rs.getInt(2) +", " + rs.getInt(3) + ")";
                              stmt2.executeUpdate(insertTopScore);
                       //Update progress bar
                        int bars = linenum/(totalLines/10);
                      int percent = linenum*100/totalLines;
                        String bar = "|";
                        for(int i=0; i<10; i++) {
                          bar += (i<bars)?"=":" ";
                        }
                        bar = bar+"| " + percent + "%\r";
                        System.out.print(bar);

                                       linenum++;
                      }
                                double total = System.currentTimeMillis()-start;
                                System.out.println("Done!          in "+total+"ms");

                      //Confirm it works by outputting the contents of topScores table
                      String getTopScores = "SELECT * FROM topScores";
                      Statement getScores = con.createStatement();
                      ResultSet scores = getScores.executeQuery(getTopScores);

            System.out.println("Contents of topScores:");
```

```
                    while(scores.next()) {
                          String outputLine = "pid: " + scores.getString(1) + "\tgid: 
" + scores.getString(2) + "\tTop Score: " + scores.getString(3);
                          System.out.println(outputLine);
                    }

                    stmt.close();
                    stmt2.close();
                    con.close();

              } catch(SQLException ex) {
                    System.err.print("SQLException: ");
                    System.err.println(ex.getMessage());
              }
        }
}
```