```
/*
 * @author Jake Gudenkauf
 */
package cs363;

import java.sql.CallableStatement;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.Types;
import java.util.Scanner;

public class hw4 {

	public static void main(String[] args) {
		String dbServer = "jdbc:mysql://localhost:3306/sakila_mod?
useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC";
//		String dbServer = "jdbc:mysql://localhost:3306/sakila_mod?
useSSL=false";
		String userName = "cs363";
		String password = "363F2020";
		Connection conn;
		Statement stmt;
		Scanner s;

		try {
			Class.forName("com.mysql.cj.jdbc.Driver");
			conn = DriverManager.getConnection(dbServer, userName, password);
			stmt = conn.createStatement();
			String option = "";
			String instruction = "Enter A: Insert new actor\n" + "Enter B:
Delete customer\n"
					+ "Enter C: Report total sales for a month\n" +
"Enter E: Exit The Program\n";
			s = new Scanner(System.in);

			while (true) {
				System.out.println(instruction);
				option = s.next();
				if (option.equals("A") || option.equals("a")) {
					System.out.println("Please enter new actor's first
name\n");
					String fname = s.next();
					System.out.println("Please enter new actor's last
name\n");
					String lname = s.next();
					String actorNames[] = new String[2];
					actorNames[0] = fname;
					actorNames[1] = lname;
					insertActor(conn, actorNames);
				} else if (option.equals("B") || option.equals("b")) {
					System.out.println("Please enter customer ID to
delete\n");
					String cID = s.next();
					deleteCustomer(conn, cID);
				} else if (option.equals("C") || option.equals("c")) {
```

```
                              System.out.println("Please enter month number to
report sales\n");
                              String monthNo = s.next();
                              reportMonthSales(conn, monthNo);
                        } else {
                              System.out.println("Exiting program\n");
                              break;
                        }
                  }
                  stmt.close();
                  conn.close();
            } catch (Exception e) {
                  System.out.println("Program terminates due to errors\n");
                  e.printStackTrace();
            }
      }

      /**
       * @param actor, 0 index is first name, 1 index is last name
       * @param sqlQuery
       * @throws SQLException
       */
      public static void insertActor(Connection connection, String[] actor) {
            //if either parameter is null, throw null pointer exception
            if (connection == null || actor[0] == null || actor[1] == null) {
                  throw new NullPointerException();
            } else {
                  try {
                        connection.setAutoCommit(false);
                        Statement stmt = connection.createStatement();
                        ResultSet rs;
                        int id=0;

                        // get the maximum id from the actor table
                        rs = stmt.executeQuery("select max(actor_id) from actor");
                        while (rs.next()) {
                              id = rs.getInt(1);
                        }
                        rs.close();
                        stmt.close();

                        //prepare statement to insert actor
                        PreparedStatement inststmt =
                                    connection.prepareStatement("insert into actor
(actor_id, first_name, last_name, last_update) values(?,?,?,?)");

                        //set values in statement
                        inststmt.clearParameters();
                        inststmt.setInt(1, id+1);
                        inststmt.setString(2, actor[0]);
                        inststmt.setString(3, actor[1]);
                        inststmt.setString(4, "current_date()");

                        //find and output number of rows updated
                        int rowcount = inststmt.executeUpdate();
                        System.out.println("Number of rows updated:" + rowcount);

                        //close statement and commit changes
                        inststmt.close();
```

```
                        connection.commit();
                  } catch(SQLException e) {
                        e.printStackTrace();
                  } catch (Exception e) {
                        e.printStackTrace();
                  }
            }
      }

      /**
       * @param cID, is the customer ID of which will be deleted
       * @param sqlQuery
       * @throws SQLException
       */
      public static void deleteCustomer(Connection connection, String cID) {
            //if either parameter is null, throw null pointer exception
            if (connection == null || cID == null) {
                  throw new NullPointerException();
            } else {
                  try {
                        connection.setAutoCommit(false);
                        Scanner s = new Scanner(System.in);
                        int toDelete = Integer.parseInt(cID);

                        //prepare statements
                        PreparedStatement custDelete =
                                    connection.prepareStatement("delete from
customer where customer_id = ?");
                        custDelete.setInt(1, toDelete);
                        PreparedStatement rentalDelete =
                                    connection.prepareStatement("delete from rental
where customer_id = ?");
                        rentalDelete.setInt(1, toDelete);
                        PreparedStatement paymentDelete =
                                    connection.prepareStatement("delete from
payment where customer_id = ?");
                        paymentDelete.setInt(1, toDelete);

                        System.out.println("Are you sure you want to delete
customer " + cID + "?");
                        System.out.println("Enter 'y' for yes or 'n' for no");
                        String option = s.next();

                        if (option.equals("y")) {
                              //find and output number of rows updated
                              int countcust = custDelete.executeUpdate();
                              int countrental = rentalDelete.executeUpdate();
                              int countpayment = paymentDelete.executeUpdate();
                              System.out.println("Number of rows updated from
customer table:" + countcust);
                              System.out.println("Number of rows updated from
rental table:" + countrental);
                              System.out.println("Number of rows updated from
payment table:" + countpayment);

                              //close statement and commit changes
                              custDelete.close();
                              rentalDelete.close();
                              paymentDelete.close();
```

```
                        connection.commit();
                    } else {
                        System.out.println("Customer " + cID + " not
deleted");
                    }
                    s.close();

                } catch(SQLException e) {
                    e.printStackTrace();
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
      }

      /**
       * @param monthNo, the month of which the sales report will be generated
       * @param sqlQuery
       * @throws SQLException
       */
      public static void reportMonthSales(Connection connection, String monthNo) {
            //if either parameter is null, throw null pointer exception
            if (connection == null || monthNo == null) {
                  throw new NullPointerException();
            } else {
                  try {
                        connection.setAutoCommit(false);
                        int monthNum = Integer.getInteger(monthNo);

                        String query = "{ call my_total_sales(?) }";

                        //callable statement
                        CallableStatement call = connection.prepareCall(query);
                        call.setInt(1, monthNum);
                        call.registerOutParameter(2, Types.DOUBLE);
                        call.executeQuery();

                        //find and output number of rows updated
                        System.out.println("Total Sales in month " + monthNo +
"is: " + call.getDouble(2));

                        //close statement and commit changes
                        call.close();
                        connection.commit();
                  } catch(SQLException e) {
                        e.printStackTrace();
                  } catch (Exception e) {
                        e.printStackTrace();
                  }
            }
      }

}
```