ASSIGNMENT: Day_21

Task 1: Establishing Database Connections

Write a Java program that connects to a SQLite database and prints out the connection object to confirm successful connection.

ANS:

```
import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;


public class Databaseconnection {


   public static void main(String[] args) {

      Connection connection = null;


      try {


         Class.forName("com.mysql.cj.jdbc.Driver");




         String url = "jdbc:mysql://localhost:3306/training" ;

         String username = "root";

         String password = "Asdfghjkl1234@";

         connection = DriverManager.getConnection(url, username, password);


         if (connection != null) {

            System.out.println("Connected to the database.");
```

```
            System.out.println("Connection object: " + connection);
        } else {
            System.out.println("Failed to connect to the database.");
        }

    } catch (ClassNotFoundException e) {
        System.out.println("MySQL JDBC driver not found.");
        e.printStackTrace();
    } catch (SQLException e) {
        System.out.println("Database connection error.");
        e.printStackTrace();
    } finally {

        if (connection != null) {
            try {
                connection.close();
                System.out.println("Database connection closed.");
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
//code_by_RUBY
```

Task 2: SQL Queries using JDBC

Create a table 'User' with a following schema 'User ID' and 'Password' stored as hash format (note you have research on how to generate hash from a string), accept "User ID" and "Password" as input and check in the table if they match to confirm whether user access is allowed or not.

ANS:

TABLE NAME `User` with columns `UserID` and `PasswordHash`.

```
CREATE TABLE User (
    UserID VARCHAR(50) PRIMARY KEY,
    PasswordHash VARCHAR(100)
);
```

JAVA :

```
import java.sql.*;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException;


public class UserAuthentication {

   static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";

   static final String DB_URL = "jdbc:mysql://localhost/your_database_name";

   static final String USER = "your_database_username";

   static final String PASS = "your_database_password";


   public static void main(String[] args) {

      Connection conn = null;

      Statement stmt = null;

      try {
```

```
Class.forName("com.mysql.jdbc.Driver");

System.out.println("Connecting to database...");
conn = DriverManager.getConnection(DB_URL, USER, PASS);

// Accepting UserID and Password from user input
String userID = "user_input_here";
String password = "user_input_here";

// Generate hash for password
String passwordHash = generateHash(password);

// Query to check if UserID and PasswordHash match
String sql = "SELECT * FROM User WHERE UserID = ? AND PasswordHash = ?";
PreparedStatement preparedStatement = conn.prepareStatement(sql);
preparedStatement.setString(1, userID);
preparedStatement.setString(2, passwordHash);
ResultSet rs = preparedStatement.executeQuery();

if (rs.next()) {
   System.out.println("Access granted. User authenticated successfully.");
} else {
   System.out.println("Access denied. Invalid UserID or Password.");
}
```

```
            rs.close();
            preparedStatement.close();
            conn.close();
        } catch (SQLException se) {
            se.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (stmt != null) stmt.close();
            } catch (SQLException se2) {
            }
            try {
                if (conn != null) conn.close();
            } catch (SQLException se) {
                se.printStackTrace();
            }
        }
    }

    // Method to generate hash for password
    private static String generateHash(String password) throws NoSuchAlgorithmException {
        MessageDigest md = MessageDigest.getInstance("SHA-256");
        md.update(password.getBytes());
        byte[] byteData = md.digest();
```

```
        // Convert byte array to hexadecimal string
        StringBuilder sb = new StringBuilder();
        for (byte b : byteData) {
            sb.append(Integer.toString((b & 0xff) + 0x100, 16).substring(1));
        }
        return sb.toString();
    }
}
#code-by-RUBY
```

Task 3: PreparedStatement

Modify the SELECT query program to use PreparedStatement to parameterize the query and prevent SQL injection.

ANS:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Scanner;

public class SqlQueriesAndPreparedStatement {

    static final String URL = "jdbc:mysql://localhost:3306/training";
```

```
    static final String USERNAME = "root";
    static final String PASSWORD = "Asdfghjkl1234@";

    static final String CREATE_TABLE_QUERY = "CREATE TABLE IF NOT EXISTS User (user_id INT PRIMARY KEY, password INT)";
    static final String INSERT_USER_QUERY = "INSERT INTO User (user_id, password) VALUES (?, ?)";
    static final String SELECT_USER_QUERY = "SELECT * FROM User WHERE user_id = ? AND password = ?";

    public static void main(String[] args) {
        try {
            Connection connection = DriverManager.getConnection(URL, USERNAME, PASSWORD);

            createTable(connection);

            Scanner scanner = new Scanner(System.in);
            System.out.println("Enter User ID:");
            int userId = scanner.nextInt();
            System.out.println("Enter Password:");
            String password = scanner.next();

            int hashedPassword = password.hashCode();

            insertUser(connection, userId, hashedPassword);
```

```
            boolean isAuthenticated = authenticateUser(connection, userId, hashedPassword);
            if (isAuthenticated) {
                System.out.println("User authenticated. Access granted!");
            } else {
                System.out.println("Invalid User ID or Password. Access denied!");
            }

            connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    private static void createTable(Connection connection) throws SQLException {
        PreparedStatement preparedStatement = connection.prepareStatement(CREATE_TABLE_QUERY);
        preparedStatement.executeUpdate();
        System.out.println("User table created (if not exists).");
    }

    private static void insertUser(Connection connection, int userId, int hashedPassword) throws SQLException {
        PreparedStatement preparedStatement = connection.prepareStatement(INSERT_USER_QUERY);
        preparedStatement.setInt(1, userId);
        preparedStatement.setInt(2, hashedPassword);
        preparedStatement.executeUpdate();
        System.out.println("User inserted into database.");
```

```
    }



    private static boolean authenticateUser(Connection connection, int userId, int hashedPassword) throws SQLException {
        PreparedStatement preparedStatement = connection.prepareStatement(SELECT_USER_QUERY);
        preparedStatement.setInt(1, userId);
        preparedStatement.setInt(2, hashedPassword);
        ResultSet resultSet = preparedStatement.executeQuery();
        return resultSet.next();
    }
   /*   private static boolean authenticateUser(Connection connection, int userId, int hashedPassword) throws SQLException {
        Statement statement = connection.createStatement();
        String query = "SELECT * FROM User WHERE user_id = " + userId + " AND password = " + hashedPassword;
        ResultSet resultSet = statement.executeQuery(query);
        return resultSet.next();
    }
   */
}
//code_by_RUBY
```