Experiment- 07

Student Name: Anish Anand **UID: 22BCS10055**
Branch: BE-CSE **Section:22BCS_IOT_639-A**
Semester: 6th **Date :25/03/25**
Subject: PBLJ **Subject Code:22CSH-359**

1. **Aim:** Create Java applications with JDBC for database connectivity, CRUD operations, and MVC architecture.

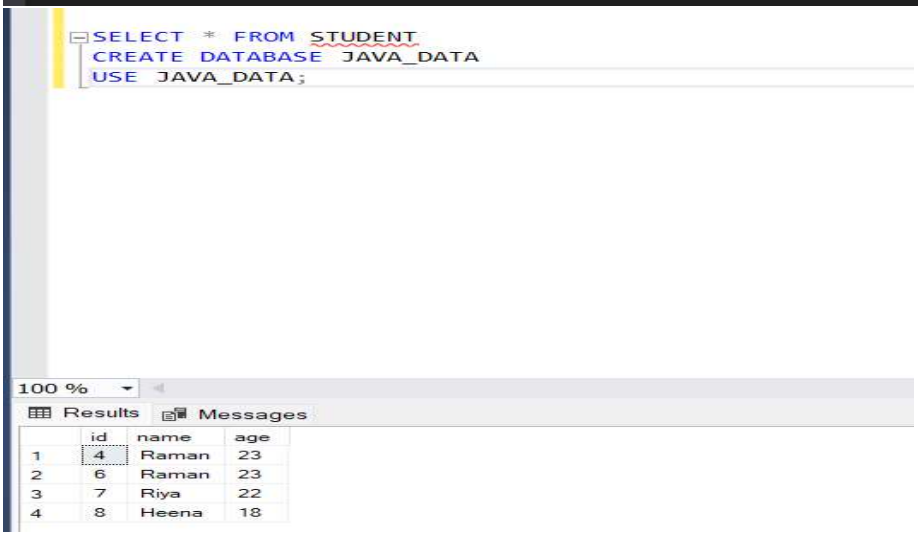
2. **Objective:** To create Java applications with JDBC for database connectivity, CRUD operations, and MVC architecture.

3. **Code:**

```
package com;
import java.sql.*;
public class App2 {
public static void main(String[] args) {
String url=
"jdbc:sqlserver://localhost:1433;databaseName=JAVA_DATA;encrypt=true;trustServerCert
ificate=true;integratedSecurity=true";
// Establish connection
 String username = "hp\\22BCS12789";
 String password = "1234";
 try{
 Connection conn=DriverManager.getConnection(url, username, password);
 System.out.println("Connected to database");
 //Create the statement
  Statement stmt=conn.createStatement();
  //create table
  String newtable="create table student("
  +"id int IDENTITY(1,1) PRIMARY KEY,"
  +"name varchar(50),"
  +"age int)";
  /stmt.executeUpdate(newtable);
   System.out.println("Table created successfully");
   //insert table
   String insertquery="insert into student(name,age) VALUES
('sukh',21),('Raman',23),('Riya',22),('Heena',18)";
```

```
stmt.executeUpdate(insertquery);
//update data
String updatequery="update student set age=20 where name='Sukh'";
stmt.executeUpdate(updatequery);
//delete data
String deletequery="delete from student where name='sukh'";
stmt.executeUpdate(deletequery);
//read data
String selectQuery="select * from student";
ResultSet rs=stmt.executeQuery(selectQuery);
while(rs.next()){
System.out.println("ID:"+rs.getInt("id")+"name:"+rs.getString("name")+"age:"+rs.get
Int("age"));
}
} catch(SQLException e){
System.out.println(e);
}
}
}
```

4. Output:



5. Learning Outcomes:

- Learn how to establish a connection between a Java application and a relational database using JDBC.
- Gain proficiency in executing SQL queries, retrieving results, and handling database transactions effectively.
- Implement Create, Read, Update, and Delete (CRUD) functionalities using JDBC.
- Apply best practices for handling exceptions, managing connections, and optimizing database interactions.