**Ex 8**

**Write a Java Program to implement the SQL commands using JDBC.**

iimport java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.Statement;

public class MySQLJDBCExample {

// MySQL Database credentials

static final String JDBC\_DRIVER = "com.mysql.cj.jdbc.Driver";

static final String DB\_URL = "jdbc:mysql://localhost:3306/your\_database\_name"; // Replace 'your\_database\_name'

// MySQL credentials

static final String USER = "your\_username"; // Replace 'your\_username'

static final String PASS = "your\_password"; // Replace 'your\_password'

public static void main(String[] args) {

Connection conn = null;

Statement stmt = null;

try {

// Register MySQL JDBC Driver

Class.forName(JDBC\_DRIVER);

// Open a connection

System.out.println("Connecting to database...");

conn = DriverManager.getConnection(DB\_URL, USER, PASS);

// Create a statement object to send SQL commands

stmt = conn.createStatement();

// Create a table

String createTableSQL = "CREATE TABLE Employees "

+ "(id INTEGER not NULL, "

+ " name VARCHAR(255), "

+ " age INTEGER, "

+ " PRIMARY KEY ( id ))";

stmt.executeUpdate(createTableSQL);

System.out.println("Table created successfully...");

// Insert data into table

String insertSQL = "INSERT INTO Employees (id, name, age) VALUES (1, 'John Doe', 30)";

stmt.executeUpdate(insertSQL);

insertSQL = "INSERT INTO Employees (id, name, age) VALUES (2, 'Jane Smith', 25)";

stmt.executeUpdate(insertSQL);

System.out.println("Records inserted successfully...");

// Select and display data from the table

String selectSQL = "SELECT id, name, age FROM Employees";

ResultSet rs = stmt.executeQuery(selectSQL);

System.out.println("Data from Employees table:");

while (rs.next()) {

int id = rs.getInt("id");

String name = rs.getString("name");

int age = rs.getInt("age");

// Display the retrieved data

System.out.println("ID: " + id + ", Name: " + name + ", Age: " + age);

}

// Update data in the table

String updateSQL = "UPDATE Employees SET age = 35 WHERE id = 1";

stmt.executeUpdate(updateSQL);

System.out.println("Record updated successfully...");

// Delete data from the table

String deleteSQL = "DELETE FROM Employees WHERE id = 2";

stmt.executeUpdate(deleteSQL);

System.out.println("Record deleted successfully...");

// Clean-up environment

rs.close();

stmt.close();

conn.close();

} catch (Exception e) {

e.printStackTrace();

} finally {

try {

if (stmt != null) stmt.close();

if (conn != null) conn.close();

} catch (Exception e) {

e.printStackTrace();

}

}

}

}

**Ex 11.**

**Write a Java Program to create the table using JDBC**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.Statement;

public class CreateTableExample {

// MySQL database credentials

static final String JDBC\_DRIVER = "com.mysql.cj.jdbc.Driver";

static final String DB\_URL = "jdbc:mysql://localhost:3306/your\_database\_name"; // Replace with your database name

static final String USER = "your\_username"; // Replace with your MySQL username

static final String PASS = "your\_password"; // Replace with your MySQL password

public static void main(String[] args) {

Connection conn = null;

Statement stmt = null;

try {

// Step 1: Register JDBC driver

Class.forName(JDBC\_DRIVER);

// Step 2: Open a connection

System.out.println("Connecting to database...");

conn = DriverManager.getConnection(DB\_URL, USER, PASS);

// Step 3: Execute a query to create the table

System.out.println("Creating table in the database...");

stmt = conn.createStatement();

String sql = "CREATE TABLE Employees " +

"(id INT NOT NULL, " +

" name VARCHAR(255), " +

" age INT, " +

" PRIMARY KEY ( id ))";

stmt.executeUpdate(sql);

System.out.println("Table 'Employees' created successfully...");

} catch (Exception e) {

e.printStackTrace();

} finally {

try {

// Step 4: Clean-up environment

if (stmt != null) stmt.close();

if (conn != null) conn.close();

} catch (Exception e) {

e.printStackTrace();

}

}

}

}

**Ex 12**

**Write a Java Program to implement Remote Method Invocation.**

**Steps to implement RMI:**

1. Create a remote interface that defines the methods that can be called remotely.
2. Implement the remote interface on the server side.
3. Create a client that will invoke the remote methods.
4. Set up the RMI registry to register the server.

**Step-by-Step RMI**

1. **Create the Remote Interface**: The interface should extend java.rmi.Remote, and each method should throw java.rmi.RemoteException.

import java.rmi.Remote;

import java.rmi.RemoteException;

// Remote interface

public interface Hello extends Remote {

String sayHello() throws RemoteException;

}

**2. Implement the Remote Interface (Server Implementation)**: The server class implements the remote interface and extends UnicastRemoteObject.

import java.rmi.RemoteException;

import java.rmi.server.UnicastRemoteObject;

// Remote object implementation class

public class HelloImpl extends UnicastRemoteObject implements Hello {

// Constructor that throws RemoteException

public HelloImpl() throws RemoteException {

super();

}

// Implementation of the remote method

@Override

public String sayHello() throws RemoteException {

return "Hello, RMI World!";

}

}

3. **Create the Server Program**: The server program registers the remote object in the RMI registry.

import java.rmi.Naming;

import java.rmi.registry.LocateRegistry;

public class RMIServer {

public static void main(String[] args) {

try {

// Create and export a remote object

HelloImpl obj = new HelloImpl();

// Start the RMI registry on port 1099

LocateRegistry.createRegistry(1099);

// Bind the remote object in the registry with a name "Hello"

Naming.rebind("rmi://localhost:1099/Hello", obj);

System.out.println("RMI Server is ready...");

} catch (Exception e) {

e.printStackTrace();

}

}

}

4. **Create the Client Program**: The client looks up the remote object and invokes the remote method.

import java.rmi.Naming;

public class RMIClient {

public static void main(String[] args) {

try {

// Lookup the remote object in the RMI registry

Hello obj = (Hello) Naming.lookup("rmi://localhost:1099/Hello");

// Call the remote method and print the result

String message = obj.sayHello();

System.out.println("Message from server: " + message);

} catch (Exception e) {

e.printStackTrace();

}

}

}

**Steps to Run the RMI Program:**

1. **Compile all the Java classes**:

javac Hello.java HelloImpl.java RMIServer.java RMIClient.java

2. **Start the RMI registry**: Open a terminal and run the following command to start the RMI registry.

rmiregistry

3. **Run the server**: In a new terminal, run the server program:

java RMIServer

4. **Run the client**: In another terminal, run the client program:

java RMIClient

**Output:**

When the client program is run, it will print:

Message from server: Hello, RMI World!