Task-3

1.Set up a local database (MySQL, PostgreSQL, or any other suitable database) and establish a connection using jdbc.

Establish the connection using jdbc:

```
import java.sql.*;

public class connection{


Connection con = null;

public static Connection connectDB()
{
  try
  {

    // Step 2: involve among 7 in Connection
    // class i.e Load and register drivers

    // 2(a) Loading drivers using forName() method
    // Here, the name of the database is mysql
    Class.forName("com.mysql.jdbc.Driver");

    // 2(b) Registering drivers using DriverManager
    Connection con = DriverManager.getConnection(
      "jdbc:mysql://localhost:3306/database",
      "root", "1234");

    // Root is the username, and
    // 1234 is the password

    // Here, the object of Connection class is return
    // which further used in main class
    return con;
  }

  // Here, the exceptions is handle by Catch block
  catch (SQLException | ClassNotFoundException e)
  {

    // Print the exceptions
    System.out.println(e);

    return null;
  }
}
}
```

2.Create a table to store student records and insert sample data.

```
CREATE TABLE students (
  student_id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(255) NOT NULL,
  age INT NOT NULL,
  grade CHAR(1) NOT NULL
```

```
);

import java.io.*;
import java.sql.*;

public class Database {

    // url that points to mysql database, 'db' is database
    // name
    static final String url
        = "jdbc:mysql://localhost:3306/db";

    public static void main(String[] args)
        throws ClassNotFoundException
    {
        try {
            // this Class.forName() method is user for
            // driver registration with name of the driver
            // as argument

            // we have used MySQL driver
            Class.forName("com.mysql.jdbc.Driver");

            // getConnection() establishes a connection. It
            // takes url that points to your database,
            // username and password of MySQL connections as
            // arguments
            Connection conn = DriverManager.getConnection(
                url, "root", "1234");
}

        // create.Statement() creates statement object
        // which is responsible for executing queries on
        // table
        Statement stmt = conn.createStatement();

        // executeUpdate() is used for INSERT, UPDATE,
        // DELETE statements.It returns number of rows
        // affected by the execution of the statement
        int result = stmt.executeUpdate(
            "insert into student(Id,name,number) values('1','rachel','45')");

        // if result is greater than 0, it means values
        // has been added
        if (result > 0)
            System.out.println("successfully inserted");

        else
            System.out.println(
                "unsucessful insertion ");

        // closing connection
        conn.close();
    }

    catch (SQLException e) {
        System.out.println(e);
    }
}
```

```
}
```

3.Write Java code to perform CRUD operations (Create, Read, Update, Delete) on the database.

```
create database SampleDB;

use SampleDB;

CREATE TABLE `users` (
   `user_id` int(11) NOT NULL AUTO_INCREMENT,
   `username` varchar(45) NOT NULL,
   `password` varchar(45) NOT NULL,
   `fullname` varchar(45) NOT NULL,
   `email` varchar(45) NOT NULL,
   PRIMARY KEY (`user_id`)
};
String sql = "INSERT INTO Users (username, password, fullname, email) VALUES (?, ?, ?, ?)";

PreparedStatement statement = conn.prepareStatement(sql);
statement.setString(1, "bill");
statement.setString(2, "secretpass");
statement.setString(3, "Bill Gates");
statement.setString(4, "bill.gates@microsoft.com");

int rowsInserted = statement.executeUpdate();
if (rowsInserted > 0) {
   System.out.println("A new user was inserted successfully!");
}
String sql = "UPDATE Users SET password=?, fullname=?, email=? WHERE username=?";

PreparedStatement statement = conn.prepareStatement(sql);
statement.setString(1, "123456789");
statement.setString(2, "William Henry Bill Gates");
statement.setString(3, "bill.gates@microsoft.com");
statement.setString(4, "bill");

int rowsUpdated = statement.executeUpdate();
if (rowsUpdated > 0) {
    System.out.println("An existing user was updated successfully!");
}
String sql = "DELETE FROM Users WHERE username=?";

PreparedStatement statement = conn.prepareStatement(sql);
statement.setString(1, "bill");

int rowsDeleted = statement.executeUpdate();
if (rowsDeleted > 0) {
    System.out.println("A user was deleted successfully!");
}
```