**8. Database Connectivity:**

**Write a program to implement MySQL/Oracle database connectivity with any front end**

**language to implement Database navigation operations (add, delete, edit etc.)**

**final jdbc code:**

import java.sql.\*;

import java.util.Scanner;

public class Online {

public static void main(String[] args) {

try {

// Load MySQL JDBC Driver

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

// Establish connection

Connection c = DriverManager.getConnection("jdbc:mysql://localhost:3306/radhika", "root", "root");

// Create a statement

Statement sm = c.createStatement();

// Scanner for input

Scanner s = new Scanner(System.in);

int ch;

String sql;

do {

System.out.println("Enter Choice \n1. Insert \n2. Select \n3. Update \n4. Delete \n5. Exit ");

ch = s.nextInt();

s.nextLine(); // Consume the newline character

switch (ch) {

case 1:

sql = "INSERT INTO stud (id, name) VALUES (1, 'Ankita')";

sm.executeUpdate(sql);

System.out.println("Record is Inserted");

break;

case 2:

sql = "SELECT id, name FROM stud";

ResultSet rs = sm.executeQuery(sql);

while (rs.next()) {

String name1 = rs.getString("name");

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

System.out.print("ID: " + id);

System.out.println(", Name: " + name1);

}

break;

case 3:

sql = "UPDATE stud SET name='z' WHERE name='Ankita'";

sm.executeUpdate(sql);

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

break;

case 4:

sql = "DELETE FROM stud WHERE id=1";

sm.executeUpdate(sql);

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

break;

case 5:

System.out.println("Exiting...");

break;

default:

System.out.println("Invalid choice. Please enter a valid option.");

}

} while (ch != 5);

// Close resources

sm.close();

c.close();

s.close();

} catch (Exception e) {

e.printStackTrace();

}

}

}

**Theory**

**Loading the JDBC Driver:**

Class.forName("com.mysql.cj.jdbc.Driver"); loads the MySQL JDBC driver, which enables Java to connect to a MySQL database.

This step registers the MySQL driver with the DriverManager, making it available for establishing connections to the MySQL database.

**Establishing a Connection:**

DriverManager.getConnection("jdbc:mysql://localhost:3306/radhika", "root", "root"); establishes a connection to the MySQL database named radhika located on the local machine (localhost).

This method requires the database URL, username, and password as parameters.

The Connection object (c) is used for executing SQL statements.

**Creating a Statement:**

Statement sm = c.createStatement(); creates a Statement object that allows SQL commands to be executed directly.

Using this Statement object, various SQL commands (insert, select, update, delete) can be executed on the stud table.

**User Input and Choice Execution:**

The program uses a Scanner to accept user input for choosing which operation to perform.

Based on the user’s choice, SQL statements for each operation are executed.

Insert: Adds a new record to the stud table.

Select: Retrieves and displays all records from the stud table.

Update: Modifies a specific record in the stud table.

Delete: Removes a record from the stud table.

Exit: Ends the program loop.

**ResultSet:**

When selecting data (SELECT), ResultSet (rs) stores the retrieved records, allowing row-by-row access to the results.

The ResultSet provides methods (getString and getInt) to access each column in the query result.

Error Handling:

The try-catch block is used to handle exceptions that may occur during the database operations, such as SQLException.

**Closing Resources:**

After operations are complete, Statement, Connection, and Scanner resources are closed to free up memory and avoid potential memory leaks.