Subject: Java Technologies

Branch: B.Tech. (CE) Semester: IV

Batch: **A4**

Student Roll No: **CE079**

Student Name: **DOBARIYA VRUND GHANSHYAMBHAI**

Department of Computer Engineering,

Faculty of Technology,

Dharmsinh Desai University, Nadiad – 387001.

# LAB-6
# Topics: JDBC, Generics

1. Write a Java application to perform operations for student information like (id[Primary key, Auto increment], firstName, lastName, branch, username and password) from a database using JDBC. • Insert two records for student

• Practice the use of the following methods of the ResultSet interface: absolute(), afterLast(), beforeFirst(), first(), isFirst(), isLast(), last(), previous(), next(), relative().

**ManageStudentDetails**

```
import java.sql.*;
import java.util.Scanner;

public class ManageStudentDetails {
 public static void main(String[] a) {

 Scanner sc = new Scanner(System.in);

 //Creating connection object
 try(Connection c =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl","VGD"
, "vrund3626")) {

 Statement s1 =
c.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_UPDATABLE);
 ResultSet rs = s1.executeQuery("SELECT ID, FIRSTNAME,  LASTNAME,
BRANCH, USERNAME, PASSWORD FROM VGD.STUDENT");

 System.out.println(rs.isBeforeFirst());

 while(rs.next()) {

 if(rs.isFirst()==true)
 System.out.println("Details of the first student");

 if(rs.isLast()==true)
 System.out.println("Details of the last student");

 System.out.println(rs.getString(1));
 System.out.println(rs.getString(2));
 System.out.println(rs.getString(3));
 System.out.println(rs.getString(4));
 System.out.println(rs.getString(5));
 System.out.println("------------------------");
rs.updateString(2,"123");
 }

 System.out.println(rs.isAfterLast());

 rs.first();
 System.out.println(rs.isFirst());
 rs.last();
 System.out.println(rs.isLast());
 PreparedStatement s2 = c.prepareStatement("INSERT INTO
STUDENT(ID, FIRSTNAME, LASTNAME, BRANCH, USERNAME, PASSWORD)
```

```
VALUES(?,?,?,?,?,?)");


 for(int i=0;i<2;i++) {

 s2.setInt(1,sc.nextInt());
 s2.setString(2,sc.next());
 s2.setString(3,sc.next());
 s2.setString(4,sc.next());
 s2.setString(5,sc.next());
 s2.setString(6,sc.next());
 s2.executeUpdate();
 }

 rs = s1.executeQuery("SELECT * FROM STUDENT");

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

**Input:**

```
5
abc
abc
abc
abc
abc
6
xyz
xyz
zzz
zzz
zzz
```

**Output:**

```
true
Details of the first student
1
jack
sparrow
CE
pirat
-------------------------
2
tony
stark
EC
ironman
-------------------------
3
tom
hardy
```

```
ME
venom
-----------------------
Details of the last student
4
John
Doe
IT
JDK
-----------------------
true
true
true
```

2. Using JDBC API and MySql database perform the following operations.

I.create a table MOVIES with following columns in the database:
**Id** of type **INTEGER AUTO INCREMENT**,
**Title** of type **VARCHAR (50)**,
**Genre** of type **VARCHAR (50)**,
**YearOfRelease** of type **INTEGER**.

II. Define **Movie** class with following data members
private Integer id;
private String title;
private String genre;
private Integer yearOfRelease;

Create getters and setters for the mentioned data members.

III. Define following methods in a class, test the methods according to user input

A. **createMovie(Movie m)**- it will insert a new record for a movie.
B. **deleteMovie(int MovieID)**- it will delete a movie with given MovieID
C. **updateMovieTitle(String title, int id)**- it will update the title of a movie with given id. D. **findMovieById(int MovieId)**- it will display all details of a movie with a given MovieId E. **findAllMovie()**- it will display all details of all movies

### Movie.java

```
public class Movie {
 private int id;
 private String genre;
 private String title;
 private int yearOfRelease;

 public Movie(int id, String title, String genre, int yearOfRelease)
{
 this.id = id;
 this.genre = genre;
 this.title = title;
 this.yearOfRelease = yearOfRelease;
 }

 public int getId() {
 return id;
 }
 public String getGenre() {
```

```
 return genre;
 }
 public String getTitle() {
 return title;
 }
 public int getYearOfRelease() {
 return yearOfRelease;
 }
}
```

**ManageMovies.java**

```
import java.sql.*;

public class ManageMovies {
 Connection c;
 ManageMovies() throws SQLException {
 this.c =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl","
VG D","vrund3626");
 }
 void createMovie(Movie m) throws SQLException {
 int id = m.getId();
 String title = m.getTitle();
 String genre = m.getGenre();
 int yearOfRelease = m.getYearOfRelease();
 Statement s = c.createStatement();
 String insQuery = "INSERT INTO MOVIES(ID, TITLE, GENRE,
YEAROFRELEASE) VALUES("+ id +",'"+ title + "','" + genre +"',"
+  yearOfRelease + ")";
// System.out.println(insQuery);
 s.executeUpdate(insQuery);
 }

 void deleteMovie(int movieId) throws SQLException {
Statement s = c.createStatement();
 String delQuery = "DELETE FROM MOVIES WHERE ID=" + movieId;
s.executeUpdate(delQuery);
 System.out.println("\n\nMovie with id= "+movieId+" is
deleted");
 }

 void updateMovieTitle(String title, int id) throws SQLException {
Statement s = c.createStatement();
 String upQuery = "UPDATE MOVIES SET TITLE ='" + title + "'
WHERE ID=" + id;
 s.executeUpdate(upQuery);
 System.out.println("");
 System.out.print("\n"+"Movie title updated for id= "+id);  }

 void findMovieById(int id) throws SQLException {
 Statement s = c.createStatement();
```

```
 String Query = "SELECT ID,TITLE,GENRE,YEAROFRELEASE FROM MOVIES
WHERE ID=" + id;
 s.executeQuery(Query);
 ResultSet rs = s.executeQuery(Query);
 if(rs==null) {
 System.out.println("Movie not found");
 return;
 }
 rs.next();
 System.out.println("\n");
 System.out.println("found the movie with id "+id);
System.out.print(rs.getInt(1)+" ");
 System.out.print(rs.getString(2)+" ");
 System.out.print(rs.getString(3)+" ");
 System.out.print(rs.getString(4)+" ");
 }

 void findAllMovie() throws SQLException {
 Statement s = c.createStatement();
 String Query = "SELECT ID,TITLE,GENRE,YEAROFRELEASE FROM
MOVIES";
 ResultSet rs = s.executeQuery(Query);
 while(rs.next()) {
 System.out.println("");
 System.out.print(rs.getInt(1)+" ");
      System.out.print(rs.getString(2)+"      ");
System.out.print(rs.getString(3)+"            ");
System.out.println(rs.getString(4));
 System.out.print("----------------------");  }
 }

}
```

**Driver.java**

```
import java.sql.SQLException;

public class Driver {
 public static void main(String[] a) throws SQLException {
Movie m1 = new Movie(1,"Oppenheimer","Thriller",2023);  Movie m2
= new Movie(2,"12th fail","Drama",2023);  Movie m3 = new
Movie(3,"Marry Christmas","Thriller",2024);  Movie m4 = new
Movie(4,"Dunki","Comedy",2023);  ManageMovies mm = new
ManageMovies();
 mm.createMovie(m1);
 mm.createMovie(m2);
 mm.createMovie(m3);
 mm.createMovie(m4);
 mm.findAllMovie();
 mm.findMovieById(4);
 mm.updateMovieTitle("12th Fail", 2);
 mm.deleteMovie(4);
 mm.findAllMovie();
```

```
 }
}
```

**Output:**

```
1 Oppenheimer Thriller 2023
----------------------
2 12th Fail Drama 2023
----------------------
3 Marry Christmas Thriller 2024
----------------------
4 Dunki Comedy 2023
----------------------

found the movie with id 4
4 Dunki Comedy 2023

Movie title updated for id= 2

Movie with id= 4 is deleted

1 Oppenheimer Thriller 2023
----------------------
2 12th Fail Drama 2023
----------------------
3 Marry Christmas Thriller 2024
----------------------
Process finished with exit code 0
```

3.Create a Generic class Calculator which can perform addition, subtraction, multiplication and division. Make sure that Calculator class works for Numeric values only. Write an appropriate main method in TestCalculator class.

**Calculator.java**

```
public class Calculator<T extends Number> {
 double addition(T op1, T op2) {
 return op1.doubleValue() + op2.doubleValue();
 }

 double subtraction(T op1, T op2) {
 return op1.doubleValue() - op2.doubleValue();
 }

 double multiplication(T op1, T op2) {
 return op1.doubleValue() * op2.doubleValue();
 }

 double division(T op1, T op2) throws Exception {
 if(op2.doubleValue()==0)
 throw new Exception("divide by zero!");
 return op1.doubleValue()/op2.doubleValue();
 }
```

```
}
```

**CalculatorTester.java**

```
public class CalculatorTester {
 public static void main(String[] abc) {
 Calculator<Integer> c = new Calculator<Integer>();
Integer op1 = new Integer(23);
 Integer op2 = new Integer(2);

 System.out.println(c.addition(op1,op2));
System.out.println(c.subtraction(op1,op2));
System.out.println(c.multiplication(op1,op2));  try {
 System.out.println(c.division(op1, op2));  }
 catch (Exception e) {
 System.out.println(e.getMessage());  }

 Calculator<Double> d = new Calculator<Double>();
Double a = new Double(345.3);
 Double b = new Double(45);

 System.out.println(d.addition(a,b));
System.out.println(d.subtraction(a,b));
System.out.println(d.multiplication(a,b));  try {
 System.out.println(d.division(a,b));  }
 catch (Exception e) {
 System.out.println(e.getMessage());  }

 }
}
```

**Output:**

```
25.0

21.0

46.0

11.5

390.3

300.3

15538.5

7.673333333333334
```

4.Write a Java program to create a generic method that takes two arrays of T type and checks if they have the same elements in the same order.

**CheckSameArr.java**

```
public class CheckSameArr<T> {
 T[] a;
 T[] b;
```

```
 public CheckSameArr(T[] a, T[] b) {
 this.a = a;
 this.b = b;
 boolean f=true;
 if(a.length != b.length)
 f=false;
 for(int i=0;i<Math.min(a.length,b.length);i++) {
if(!a[i].equals(b[i]))
 f=false;
 }
 if(!f)
 System.out.println("Not same");
 else
 System.out.println("Same");
 }
}
```

**Driver2.java**

```
public class Driver2 {
 public static void main(String[] ae) {
 Integer[] a = new Integer[3];
 Integer[] b = new Integer[3];
 a[0]=new Integer(1); b[0]= new Integer(1);
 a[1]=new Integer(2); b[1]= new Integer(2);
 a[2]=new Integer(999); b[2]= new Integer(999);
 CheckSameArr<Integer> ck = new CheckSameArr<>(a,b);  }
}
```

**Output:**

```
Same
```