Laboratory Work

Subject: Java Technologies

Branch: B.Tech. (CE)

Semester: IV

Batch: A2

Student Roll No: CE030

Student Name: Paramar Lakhman

Department of Computer Engineering,

Faculty of Technology,

Dharmsinh Desai University, Nadiad – 387001.

Gujarat, INDIA.

LAB-6 JDBC, Generics

Q.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.

Ans.

Insert two records for student

```
// Press Shift twice to open the Search Everywhere dialog and type `show whitespaces`,
// then press Enter. You can now see whitespace characters in your code.
import java.sql.*;

public class Main {

    public static void main(String[] args) {
        // JDBC URL, username, and password of MySQL server
        String url = "jdbc:mysql://localhost:3306/lucky";
        String user = "root";
        String password = "";

        try {
            // Establish a connection
            Connection connection = DriverManager.getConnection(url, user, password);
            System.out.println("Connected to the database!");

            // Create a statement
            Statement statement = connection.createStatement();

            String query="INSERT INTO `student` (`id`, `firstname`, `lastname`, `brach`, `username`,
`password`) VALUES (NULL, 'lakhman', 'parmar', 'ce', 'lucky737', '123');";
            String query2="INSERT INTO `student` (`id`, `firstname`, `lastname`, `brach`, `username`,
`password`) VALUES (NULL, 'vaibhav', 'makvana', 'ce', 'vaibhav123', '123456');";
            String query3="INSERT INTO `student` (`id`, `firstname`, `lastname`, `brach`, `username`,
`password`) VALUES (NULL, 'shivansh', 'patel', 'ce', 'shivansh', '12abc56');";

            // Execute a query
         int x = statement.executeUpdate(query1);
         System.out.println("no of row affted is: "+x);

//            statement.close();
//            connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

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

```
// Press Shift twice to open the Search Everywhere dialog and type `show whitespaces`,
// then press Enter. You can now see whitespace characters in your code.
import java.sql.*;

public class Main {

    public static void main(String[] args) {
        // JDBC URL, username, and password of MySQL server
        String url = "jdbc:mysql://localhost:3306/lucky";
        String user = "root";
        String password = "";

        try {
            // Establish a connection
            Connection connection = DriverManager.getConnection(url, user, password);
            System.out.println("Connected to the database!");

            // Create a statement
            Statement statement = connection.createStatement();
            String query="INSERT INTO `student` (`id`, `firstname`, `lastname`, `brach`, `username`,
`password`) VALUES (NULL, 'lakhman', 'parmar', 'ce', 'lucky737', '123');";
            String query2="INSERT INTO `student` (`id`, `firstname`, `lastname`, `brach`, `username`,
`password`) VALUES (NULL, 'vaibhav', 'makvana', 'ce', 'vaibhav123', '123456');";
            String query3="INSERT INTO `student` (`id`, `firstname`, `lastname`, `brach`, `username`,
`password`) VALUES (NULL, 'shivansh', 'patel', 'ce', 'shivansh', '12abc56');";

            // Execute a query
//            int x = statement.executeUpdate(query3);
//            System.out.println("no of row affted is: "+x);
            String slectquery="select * from student";
//
//            // Execute a query
            ResultSet resultSet = statement.executeQuery(slectquery);
            resultSet.next();
            int id = resultSet.getInt("id");
            String firstnamename = resultSet.getString("firstname");
            String lastname=resultSet.getString("lastname");
            String branch=resultSet.getString("branch");
            String username=resultSet.getString("username");
            String pass=resultSet.getString("password");

            // Add more columns as needed
            System.out.println("id : "+ id +", name : "+firstnamename + " "  + lastname +", branch :" +
                    branch+ ", username : "+username+", pass : "+pass);
//
//            // Process the results
//            while (resultSet.next()) {
//                // Retrieve data by column name
//                int id = resultSet.getInt("id");
//                String firstnamename = resultSet.getString("firstname");
//                String lastname=resultSet.getString("lastname");
//                String branch=resultSet.getString("branch");
//                String username=resultSet.getString("username");
//                String pass=resultSet.getString("password");
//
//                // Add more columns as needed
//                System.out.println("id : "+ id +", name : "+firstnamename + " "  + lastname +", branch :" +
//                        branch+ ", username : "+username+", pass : "+pass);
//                // Print the results
//
//
//            }

            // Close resources
//            resultSet.close();
//            statement.close();
//            connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

Q.2

Using JDBC API and MySql database perform the following operations.

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**.

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.

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

2. **createMovie()**it will insert a new record for a movie
3. **deleteMovie(int MovieID)**it will delete a movie with given MovieID
4. **updateMovieTitle(String title, int id)**it will update the title of a movie with given id.
5. **findMovieById(int MovieId)**it will display all details of a movie with a given MovieId
6. **findAllMovie()**it will display all details of all movies

Ans.

```
import java.sql.*;

class Movies{


   private int id;
   private String title;
    private   String genre;
    private int yearOfRelease;

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

    public Movies() {

    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getTitle() {
        return title;
    }
    public void setTitle(String title) {
        this.title = title;
    }

    public String getGenre() {
        return genre;
    }

    public void setGenre(String genre) {
        this.genre = genre;
    }

    public int getyearOfRelease() {
        return yearOfRelease;
    }

    public void setyearOfRelease(int yearOfRelease) {
        this.yearOfRelease = yearOfRelease;
    }

    boolean createMovie(Connection connection) throws SQLException {
        String inserter="INSERT INTO `MOVIES` (`id`, `Title`, `Genre`, `YearOfRelease`) VALUES (NULL,
'"+this.title+"', '"+this.genre+"', '"+this.getyearOfRelease()+"');";
        Statement statement=connection.createStatement();


        return statement.executeUpdate(inserter)==1;
    }
    int deleteMovie(int MovieID,Connection connection) throws SQLException {

//        DELETE FROM `MOVIES` WHERE `MOVIES`.`id` = 1;
        String inserter="DELETE FROM `MOVIES` WHERE `MOVIES`.`id` = "+MovieID+";";
        Statement statement=connection.createStatement();
        return statement.executeUpdate(inserter);

    }
    boolean updateMovieTitle(String title,int MovieID,Connection connection) throws SQLException {
//        UPDATE `MOVIES` SET `Genre` = 'comedy,reallife' WHERE `MOVIES`.`id` = 2;
        String updater="UPDATE `MOVIES` SET `Title` = '"+title+"' WHERE `MOVIES`.`id` =
"+MovieID+";";
        Statement statement=connection.createStatement();
        return statement.execute(updater);
    }
    ResultSet findMovieById(int MovieId ,Connection connection) throws SQLException {
        String finder="SELECT * FROM `MOVIES` WHERE `id` = "+MovieId+" ";
        Statement statement=connection.createStatement();
        ResultSet resultSet = statement.executeQuery(finder);

        return resultSet;
    }
    ResultSet findAllMovie(Connection connection) throws SQLException {
        String finder="SELECT * FROM `MOVIES`  ";
        Statement statement=connection.createStatement();
        ResultSet resultSet = statement.executeQuery(finder);

        return resultSet;
    }
    void printallmovie(ResultSet resultSet) throws SQLException {
        resultSet.next();
        while (resultSet.next()) {
             int id=resultSet.getInt("id");
             String title=resultSet.getString("Title");

               String genre=resultSet.getString("Genre");
             int yearOfRelease=resultSet.getInt("YearOfRelease");
            System.out.println("id: "+id+", Title: "+title+", Genre: "+genre+", YearOfRelease:
"+yearOfRelease);
        }

    }
    void printmovie(ResultSet resultSet) throws SQLException {
        resultSet.next();

            int id=resultSet.getInt("id");
            String title=resultSet.getString("Title");

            String genre=resultSet.getString("Genre");
            int yearOfRelease=resultSet.getInt("YearOfRelease");
            System.out.println("id: "+id+", Title: "+title+", Genre: "+genre+", YearOfRelease:
"+yearOfRelease);


    }
```

```
}

public class Movie {

    public static void main(String[] args) {
        String url = "jdbc:mysql://localhost:3306/lucky";
        String user = "root";
        String password = "";

        try {
            Connection connection = DriverManager.getConnection(url, user, password);
            System.out.println("Connected to the database");

            Movies movieHandler = new Movies();

            // Insert a new movie
            Movies newMovie = new Movies(10, "Avengers", "Sci-Fi", 2014);
            boolean insertionResult = newMovie.createMovie(connection);
            System.out.println("Movie Inserted: " + insertionResult);

            // Update the title of a movie
            boolean updateResult = movieHandler.updateMovieTitle("Avengers Endgame", 10, connection);
            System.out.println("Movie Title Updated: " + updateResult);

            // Display details of a movie by ID
            ResultSet movieById = movieHandler.findMovieById(4, connection);
            System.out.println("Details of Movie with ID 4:");
            movieHandler.printmovie(movieById);

            // Display details of all movies
            ResultSet allMovies = movieHandler.findAllMovie(connection);
            System.out.println("\nDetails of All Movies:");
            movieHandler.printallmovie(allMovies);

            // Delete a movie by ID
            int movieIdToDelete = 4;
            int deleteResult = movieHandler.deleteMovie(movieIdToDelete, connection);
            System.out.println("\nMovie Deleted: " + (deleteResult > 0));

            // Display details of all movies after deletion
            ResultSet remainingMovies = movieHandler.findAllMovie(connection);
            System.out.println("\nDetails of Remaining Movies:");
            movieHandler.printallmovie(remainingMovies);

        } catch (SQLException e) {
            throw new RuntimeException(e);
        }
    }
}
```

output:

```
/Library/Java/JavaVirtualMachines/jdk-21.jdk/Contents/Home/bin/java -
javaagent:/Users/lakhman/Applications/IntelliJ IDEA Community
Edition.app/Contents/lib/idea_rt.jar=61367:/Users/lakhman/Applications/IntelliJ
IDEA Community Edition.app/Contents/bin -Dfile.encoding=UTF-8 -
Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath
/Users/lakhman/Desktop/sem 4
coding/java/jdbc/out/production/jdbc:/Users/lakhman/Downloads/mysql-
connector-j-8.2.0/mysql-connector-j-8.2.0.jar Movie
Connected to the database
Movie Inserted: true
Movie Title Updated: false
Details of Movie with ID 4:
id: 4, Title: IRON MAN, Genre: scifi, YearOfRelease: 2008

Details of All Movies:
id: 3, Title: ironman2, Genre: ironman3, YearOfRelease: 2008
id: 4, Title: IRON MAN, Genre: scifi, YearOfRelease: 2008
id: 7, Title: Avengers, Genre: Sci-Fi, YearOfRelease: 2014

Movie Deleted: true

Details of Remaining Movies:
id: 3, Title: ironman2, Genre: ironman3, YearOfRelease: 2008
id: 7, Title: Avengers, Genre: Sci-Fi, YearOfRelease: 2014

Process finished with exit code 0
```

Q.3

1. 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.

Ans.

```
class calc<T extends  Number>{
    T op1;
    T op2;

    public calc(T op1, T op2) {
        this.op1 = op1;
        this.op2 = op2;
    }

    public calc() {

    }

    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) {
        return op1.doubleValue()/op2.doubleValue();
    }

}
public class TestCalculator {
    public static void main(String[] args) {

        calc c1=new calc();
        c1.op1=34234;
        c1.op2=234.3545;
        System.out.println(c1.addition(354,3453.354342));
        System.out.println(c1.subtraction(c1.op1, c1.op2));
        System.out.println(c1.multiplication(c1.op1, c1.op2));
        System.out.println(c1.division(c1.op1, c1.op2));
    }
}
```

Output:

```
/Library/Java/JavaVirtualMachines/jdk-21.jdk/Contents/Home/bin/java -
javaagent:/Users/lakhman/Applications/IntelliJ IDEA Community
Edition.app/Contents/lib/idea_rt.jar=61454:/Users/lakhman/Applications/IntelliJ
IDEA Community Edition.app/Contents/bin -Dfile.encoding=UTF-8 -
Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath
/Users/lakhman/Desktop/sem 4
coding/java/jdbc/out/production/jdbc:/Users/lakhman/Downloads/mysql-
connector-j-8.2.0/mysql-connector-j-8.2.0.jar TestCalculator
3807.354342
33999.6455
8022891.953
146.07784360872097

Process finished with exit code 0
```

Q.4

1. 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.

Ans.

```
public class check {
    public static <T> boolean arrEquals(T[] arr1, T[] arr2){
        if (arr1.length != arr2.length) return false;
        for(int i=0;i<arr1.length;i++) {
            if (!arr1[i].equals(arr2[i])) return false;
        }
        return true;
    }
    public static void main(String[] args) {
        Integer[] intArray1 = {1, 2, 3, 4, 5};
        Integer[] intArray2 = {1, 2, 3, 4, 5};

        boolean intArraysEqual = arrEquals(intArray1, intArray2);
        System.out.println("Integer Arrays are equal: " + intArraysEqual);

        // Example with String arrays
        String[] strArray1 = {"apple", "orange", "banana"};
        String[] strArray2 = {"apple", "orange", "banana"};

        boolean strArraysEqual = arrEquals(strArray1, strArray2);
        System.out.println("String Arrays are equal: " + strArraysEqual);

        // Example with Double arrays
        Double[] doubleArray1 = {1.0, 2.0, 3.1};
        Double[] doubleArray2 = {1.0, 2.0, 3.0};

        boolean doubleArraysEqual = arrEquals(doubleArray1, doubleArray2);
        System.out.println("Double Arrays are equal: " + doubleArraysEqual);
    }

}
```

Output:

```
/Library/Java/JavaVirtualMachines/jdk-21.jdk/Contents/Home/bin/java -
javaagent:/Users/lakhman/Applications/IntelliJ IDEA Community
Edition.app/Contents/lib/idea_rt.jar=61632:/Users/lakhman/Applications/IntelliJ
IDEA Community Edition.app/Contents/bin -Dfile.encoding=UTF-8 -
Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath
/Users/lakhman/Desktop/sem 4
coding/java/jdbc/out/production/jdbc:/Users/lakhman/Downloads/mysql-
```

```
connector-j-8.2.0/mysql-connector-j-8.2.0.jar check
Integer Arrays are equal: true
String Arrays are equal: true
Double Arrays are equal: false

Process finished with exit code 0
```