TRINITY INTERNATIONAL COLLEGE

(Tribhuvan University Affiliated)

Lab Assignment 4: Advance Java Programming

Submitted By: **Submitted to:**

Name:

Program: **B. Sc. (CSIT)**

Roll No:

Semester: seventh (7^{th})

Date:

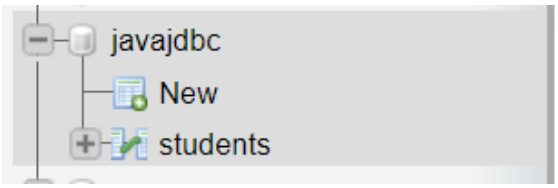
KATHMANDU, NEPAL
2020

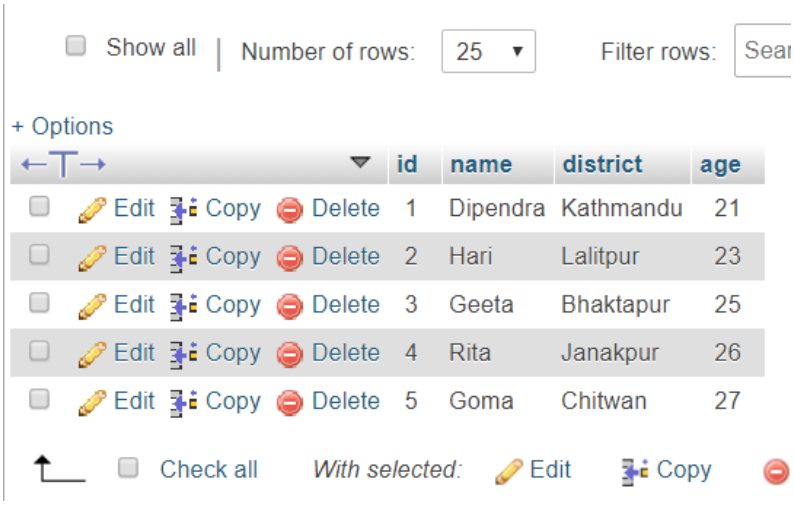
Unit 4: Database Connectivity

Here is my database structure.

SQL Query:

```
create database JavaJDBC;
use JavaJDBC;
CREATE TABLE students
 (
   id int NOT NULL AUTO_INCREMENT,
   name varchar(25) NOT NULL,
   district varchar(25),
   age int,
   PRIMARY KEY (id)
);
```





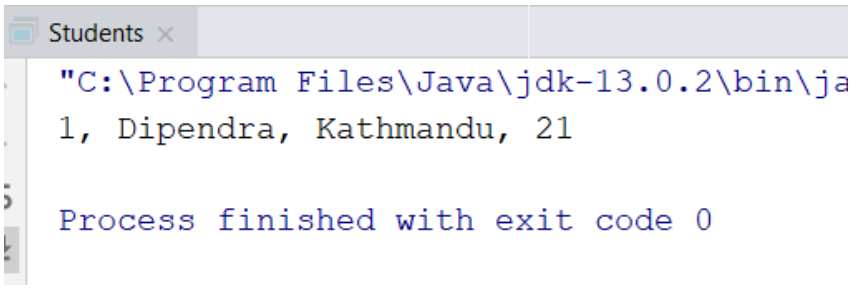
1) Write a Java program using JDBC to extract name of those students who live in Kathmandu district, assuming that the student table has four attributes (ID, name, district, and age). [2072]

⇨

Program

```
package Q01_ExtractStudentInformation;
import java.sql.*;
public class Students
{
    public static void main(String[] args) throws
                                                SQLException
    {
//******* Establishing Connection to the Database********

        String url = "jdbc:mariadb://localhost:3306
                                                /JavaJDBC";
        String username = "root";
        String password= "";
        Connection connection=DriverManager.getConnection
                                (url,username,password);

// ****************** Executing SQL Statement ***********
        Statement statement= connection.
                                        createStatement();

        String sql = "select * from students where
                                    district='kathmandu'";
        ResultSet resultSet = statement.
                                        executeQuery(sql);
        while (resultSet.next())
        {
            System.out.printf("%d, %s, %s, %d \n",
                    resultSet.getInt("id"),
                    resultSet.getString("name"),
                    resultSet.getString("district"),
                    resultSet.getInt("age")
            );
        }
        statement.close();
        connection.close();
    }
}
```

Output



2) Write a program to illustrate the process of executing SQL statements in JDBC? [2073, 2074]
3) Implement CRUD (Create/Insert, Read/Select, Update, Delete) operations for student table. Ask for user input where applicable.

⇨

Program

Create Operation

```
package Q02_Q03_SQLStatements;

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

public class Create {
    public static void main(String[] args) throws
                                                SQLException
     {
//******** Establishing Connection to the Database

        String url = "jdbc:mariadb://localhost:3306
                                                /JavaJDBC";
        String username = "root";
        String password = "";
        Connection connection=DriverManager.getConnection
                                (url, username, password);

//**Insert into students table according to user input**
        Statement statement= connection.
                                        createStatement();

        Scanner in = new Scanner(System.in);
        System.out.println("Enter Name:");
        String name = in.nextLine();
        System.out.println("Enter District:");
        String district = in.next();
        System.out.println("Enter Age:");
        int age = in.nextInt();

        String sql = String.format("Insert into students
                                (name,district,age) Values
                  ('"+name+"','"+district+"','"+age+"')");

        int rowInserted = statement.executeUpdate(sql);
        if(rowInserted>0)
          System.out.println("Row InsertedSuccessfully!");
        else
            System.out.println("Row Insertion Failed!!");
        statement.close();
        connection.close();
    }
}
```

output:

```
Create
"C:\Program Files\Java\jdk-13.0.2\bin\jav
Enter Name:
Lakhan
Enter District:
Kanchanpur
Enter Age:
26
Row Inserted Successfully!

Process finished with exit code 0
```

+ Options

	id	name	district	age
Edit Copy Delete	1	Dipendra	Kathmandu	21
Edit Copy Delete	2	Hari	Lalitpur	23
Edit Copy Delete	3	Geeta	Bhaktapur	25
Edit Copy Delete	4	Rita	Janakpur	26
Edit Copy Delete	5	Goma	Chitwan	27
Edit Copy Delete	9	Lakhan	Kanchanpur	26

Read Operation

```
package Q02_Q03_SQLStatements;

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

public class Read {

    public static void main(String[] args) throws
                                            SQLException
    {
//*** Establishing Connection to the Database ****

        String url = "jdbc:mariadb://localhost:3306/
                                              JavaJDBC";
        String username = "root";
        String password = "";
        Connection connection = DriverManager.
                                          getConnection
                          (url, username, password);

// ************** Creating SQL Statement******
        Statement statement = connection.
                                        createStatement();
```

```
// **** If user want to select but he/she don't know
//what values are  inside database table *******

        System.out.println("Do You want to see database
          table before making Selection Query??(Y/N):");

        Scanner in = new Scanner(System.in);
        char userChoice = in.next().charAt(0);

        if (userChoice == 'Y')
        {
            String sql = "select * from students";
            ResultSet resultSet = statement.
                                        executeQuery(sql);
            while (resultSet.next())
            {
                System.out.printf("%d, %s, %s, %s, \n",
                        resultSet.getInt("id"),
                        resultSet.getString("name"),
                       resultSet.getString("district"),
                        resultSet.getString("age")
                );
            }
        }

// ****** If user want to select and he/she don't know
//what values are  inside database table **************

        if(userChoice== 'N')
        {
            System.out.println("Enter the Id to select
                                        from database:");
            int id = in.nextInt();

            String sql = "select * from students where
                                          id = '"+id+"'";
            ResultSet resultSet = statement.
                                        executeQuery(sql);

            while (resultSet.next())
            {
                System.out.printf("%d, %s, %s, %s \n",
                        resultSet.getInt("id"),
                        resultSet.getString("name"),
                       resultSet.getString("district"),
                        resultSet.getString("age")

                );
            }
        }
        statement.close(); //statement has to be close.
        connection.close();
  }
}
```

Output

If user wish to see the database table details before executing 'Read' operation.

```
Read
"C:\Program Files\Java\jdk-13.0.2\bin\java.exe" -Didea.launcher.port=63265 "
Do You want to see database table before making Selection Query??(Y/N):
Y
1, Dipendra, Kathmandu, 21,
2, Hari, Lalitpur, 23,
3, Geeta, Bhaktapur, 25,
4, Rita, Janakpur, 26,
5, Goma, Chitwan, 27,
9, Lakhan, Kanchanpur, 26,
```

If user wants to execute 'Read' operation directly without seeing details in database table.

```
Read
"C:\Program Files\Java\jdk-13.0.2\bin\java.exe" -Didea.launcher.port=63
Do You want to see database table before making Selection Query??(Y/N):
N
Enter the Id to select from database:
1
1, Dipendra, Kathmandu, 21

Process finished with exit code 0
```

Update Operation

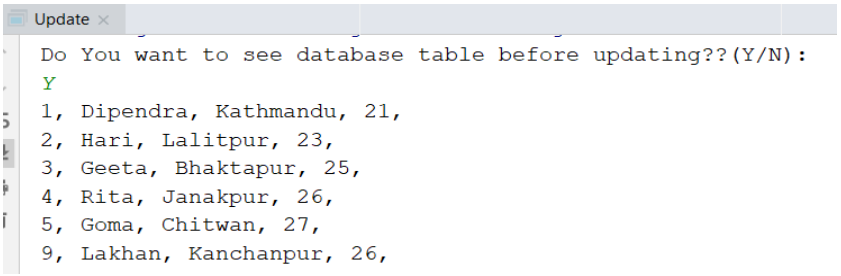
```
package Q02_Q03_SQLStatements;

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

public class Update {

    public static void main(String[] args) throws
                                                SQLException
    {
//******Establishing Connection to the Database ********

        String url "jdbc:mariadb://localhost:3306
                                                 /JavaJDBC";
        String username = "root";
        String password = "";

        Connection connection = DriverManager.
                  getConnection(url, username, password);



// *********** Creating SQL Statement *********
        Statement statement = connection.
```

```
                                        createStatement();

// ****** If user want to update but he/she don't know
//what values are  inside database table ******

        System.out.println("Do You want to see database
                          table before updating??(Y/N):");
        Scanner in = new Scanner(System.in);
        char userChoice = in.next().charAt(0);

        if (userChoice == 'Y')
        {
            String sql = "select * from students";
            ResultSet resultSet = statement.
                                        executeQuery(sql);
            while (resultSet.next())
           {
                System.out.printf("%d, %s, %s, %s, \n",
                        resultSet.getInt("id"),
                        resultSet.getString("name"),
                        resultSet.getString("district"),
                        resultSet.getString("age")
                );
            }

        }
//***If user knows what to update in database table*****

        if (userChoice == 'N')
        {

            System.out.println("Enter the Id to update
                                        into database:");
            int id = in.nextInt();

            System.out.println("Enter the Name to update
                                        into database:");
            String name = in.next();

            System.out.println("Enter the District to
                                  update into database:");
            String district = in.next();

            System.out.println("Enter the Age to update
                                        into database:");
            int age = in.nextInt();

            String sql = "update students set
                                 name = '" + name + "'
                                ,district= '" + district + "'
                                , age = '" + age + "
                                ' where id='" + id + "'";

           int rowUpdated = statement.executeUpdate(sql);
```

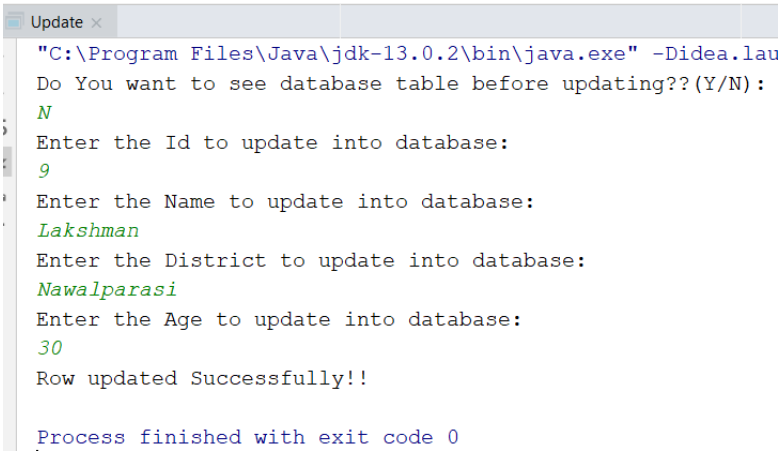
```
            if (rowUpdated > 0)
                System.out.println("Row updated
                                        Successfully!!");

              else
                 System.out.println("Row isn't Updated!!");
          }
          statement.close(); //statement has to be close.
          connection.close(); // connection is also closed.
      }
}
```

Output:

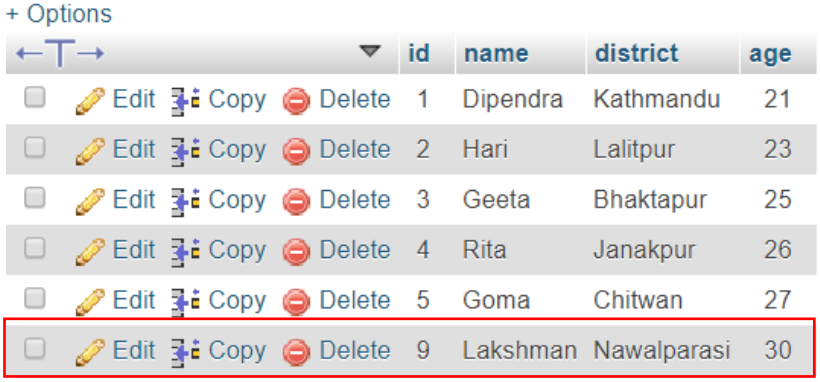
If user wish to see the database table details before executing 'Update' operation.



If user wants to execute 'Update' operation directly without seeing details in database table.



Record '9' was successfully updated.



Delete Operation

```
package Q02_Q03_SQLStatements;

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

public class Delete
{

        public static void main(String[] args) throws
                                                SQLException
        {
//******* Establishing Connection to the Database *****

            String url = "jdbc:MySQL://localhost:3306
                                                /JavaJDBC ";
            String username = "admin";
           String password= "admin";

            Connection connection = DriverManager.
                     getConnection(url,username,password);

// ******Creating SQL Statement **********************
            Statement statement= connection.
                                        createStatement();

// ****** If user want to select but he/she don't know
//what values are  inside database table
//***********************************

            System.out.println("Do You want to see
                             database table before making
                             Selection Query??(Y/N):");

            Scanner in = new Scanner(System.in);
            char userChoice = in.next().charAt(0);

            if (userChoice == 'Y')
            {
                String sql = "select * from students";
                ResultSet resultSet = statement.
                                        executeQuery(sql);
                while (resultSet.next())
                {
                    System.out.printf("%d, %s, %s, %s,\n",
                            resultSet.getInt("id"),
                            resultSet.getString("name"),

resultSet.getString("district"),
                            resultSet.getString("age")
                    );
                }

            }
```

```
// **** If user want to select but he/she don't know what
//values are inside database table ********************

            if(userChoice== 'N')
            {
                System.out.println("Enter the Id to Delete
                                        from database:");
                int id = in.nextInt();

                String sql = "delete from students where
                                       id= '" + id + "'";
                int rowDeleted = statement.
                                      executeUpdate(sql);

                if (rowDeleted > 0)
                    System.out.println("Row Deleted
                                         Successfully!!");

                else
                    System.out.println("Row isn't
                                           Deleted!!");
                statement.close();
                connection.close();
            }
        }
}
```

Output

If user wish to see the database table details before executing 'Delete' operation.

```
Delete
"C:\Program Files\Java\jdk-13.0.2\bin\java.exe" -Didea.launcher.port=49
Do You want to see database table before making Selection Query??(Y/N):
Y
1, Dipendra, Kathmandu, 21,
2, Hari, Lalitpur, 23,
3, Geeta, Bhaktapur, 25,
4, Rita, Janakpur, 26,
5, Goma, Chitwan, 27,
9, Lakshman, Nawalparasi, 30,

Process finished with exit code 0
```

If user wants to execute 'Delete' operation directly without seeing details in database table.

```
Delete
"C:\Program Files\Java\jdk-13.0.2\bin\java.exe" -Didea.launcher.port=495
Do You want to see database table before making Selection Query??(Y/N):
N
Enter the Id to Delete from database:
9
Row Deleted Successfully!!

Process finished with exit code 0
```

Record with id '9' was successfully removed.

4) Implement CRUD operations for student table using prepared statements. Ask for user input where applicable.

⇨

Program

Create Operation Using PreparedStatement

```
package Q04_CrudPreparedStatement;

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

public class CreatePreparedStatement
{
    public static void main(String[] args) throws
                                                SQLException
   {
        String url = "jdbc:mariadb://localhost:3306
                                                /JavaJDBC?";
        String username = "root";
        String password = "";
        Connection connection = DriverManager.
                          getConnection(url, username,
                                                password);
         Scanner in = new Scanner(System.in);

        System.out.println("Enter Name:");
        String name = in.next();

        System.out.println("Enter District:");
        String district = in.next();

        System.out.println("Enter Age:");
        int age = in.nextInt();

        //****** Sql prepared statement ************
        String sql = String.format("Insert into students
                                  (name,district,age)
                                    Values (?,?,?)");
```

```
PreparedStatement statement = connection.
                                    prepareStatement(sql);

        //****************** Binding Parameter *******
        statement.setString(1,name);
        statement.setString(2,district);
        statement.setInt(3,age);
        int rowInserted = statement.executeUpdate();

        if(rowInserted>0)
            System.out.println("Row Inserted
                                        Successfully!");
        else
            System.out.println("Row Insertion Failed!!");
        statement.close();
        connection.close();
    }
}
```

Output

```
CreatePreparedStatement
"C:\Program Files\Java\jdk-13.0.2\bin
Enter Name:
Ramesh
Enter District:
Kailali
Enter Age:
36
Row Inserted Successfully!

Process finished with exit code 0
```

+ Options

	id	name	district	age
Edit Copy Delete	1	Dipendra	Kathmandu	21
Edit Copy Delete	2	Hari	Lalitpur	23
Edit Copy Delete	3	Geeta	Bhaktapur	25
Edit Copy Delete	4	Rita	Janakpur	26
Edit Copy Delete	5	Goma	Chitwan	27
Edit Copy Delete	10	Ramesh	Kailali	36

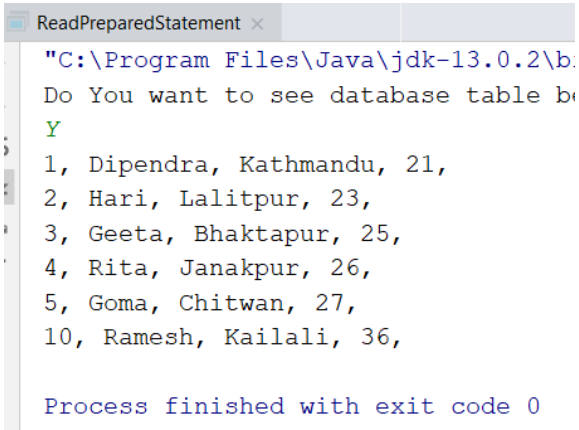
Read Operation Using PreparedStatement

```
package Q04_CrudPreparedStatement;

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

public class ReadPreparedStatement
{
        public static void main(String[] args) throws
                                                SQLException
          {

            String url = "jdbc:mariadb://localhost:3306/
                                                 JavaJDBC";
            String username = "root";
            String password = "";
            Connection connection = DriverManager.
                              getConnection(url, username,
                                                 password);

            System.out.println("Do You want to see
                              database table before making
                                 Selection Query??(Y/N):");
            Scanner in = new Scanner(System.in);
            char userChoice = in.next().charAt(0);

            if (userChoice == 'Y')
            {
                String sql = "select * from students";
                PreparedStatement statement = connection.
                                        prepareStatement(sql);
                ResultSet resultSet = statement.
                                             executeQuery();
                while (resultSet.next())
                {
                   System.out.printf("%d, %s, %s, %s, \n",
                            resultSet.getInt("id"),
                            resultSet.getString("name"),
                          resultSet.getString("district"),
                            resultSet.getString("age")
                     );
                }
                statement.close();
            }


            if(userChoice== 'N')
            {
                System.out.println("Enter the Id to select
                                           from database:");
                int id = in.nextInt();
```

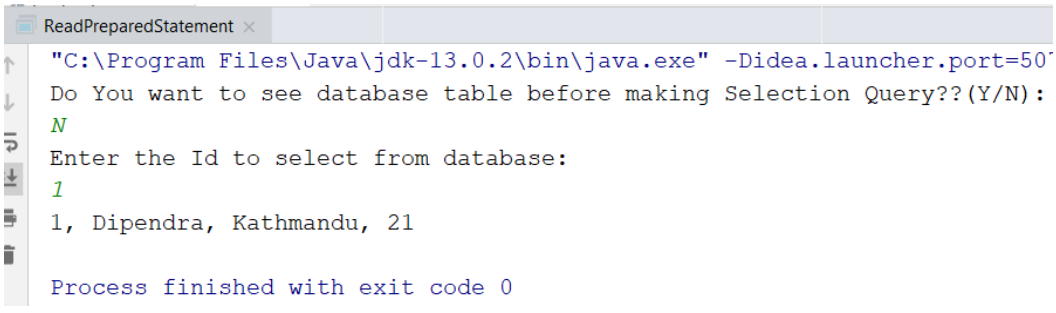
```
//***************** Sql Prepared Statement*************

                String sql = "select * from students where
                                                    id = ?";
                PreparedStatement statement = connection.
                                        prepareStatement(sql);
                statement.setInt(1,id);

                ResultSet resultSet = statement.
                                            executeQuery();

                while (resultSet.next())
                {
                    System.out.printf("%d, %s, %s, %s \n",
                            resultSet.getInt("id"),
                            resultSet.getString("name"),
                          resultSet.getString("district"),
                            resultSet.getString("age")
                    );
                }
                statement.close();
            }
            connection.close();
        }
}
```

Output

If user wish to see the database table details before executing 'Read' operation.



If user wants to execute 'Read' operation directly without seeing details in database table.



Update Operation Using PreparedStatement

```
package Q04_CrudPreparedStatement;

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

public class UpdatePreparedStatement
{
    public static void main(String[] args) throws
                                                SQLException
    {


        String url = "jdbc:mariadb://localhost:3306
                                                /JavaJDBC";
        String username = "root";
        String password = "";

        Connection connection = DriverManager.
                  getConnection(url, username, password);


        System.out.println("Do You want to see database
                        table before updating??(Y/N):");
        Scanner in = new Scanner(System.in);
        char userChoice = in.next().charAt(0);

        if (userChoice == 'Y')
        {
            String sql = "select * from students";
            PreparedStatement statement = connection.
                                    prepareStatement(sql);
            ResultSet resultSet = statement.
                                        executeQuery(sql);
            while (resultSet.next())
           {
                System.out.printf("%d, %s, %s, %s, \n",
                        resultSet.getInt("id"),
                        resultSet.getString("name"),
                        resultSet.getString("district"),
                        resultSet.getString("age")
                );
            }
            statement.close();
        }
         if (userChoice == 'N')
        {
             System.out.println("Enter the Id to update
                                        into database:");
            int id = in.nextInt();

            System.out.println("Enter the Name to update
                                          into database:");
            String name = in.next();
```

```
            System.out.println("Enter the District to
                                update into database:");
            String district = in.next();

            System.out.println("Enter the Age to update
                                        into database:");
            int age = in.nextInt();

             String sql = "update students set name =
                              ?,district= ?, age = ?
                                        where id= ?";
            PreparedStatement statement = connection.
                                   prepareStatement(sql);

            statement.setString(1,name);
            statement.setString(2,district);
            statement.setInt(3,age);
            statement.setInt(4,id);

            int rowUpdated = statement.executeUpdate();

            if (rowUpdated > 0)
                System.out.println("Row updated
                                        Successfully!!");
            else
                System.out.println("Row isn't
                                             Updated!!");
            statement.close();
        }
        connection.close();
    }
}
```

Output

If user wish to see the database table details before executing 'Update' operation.

```
UpdatePreparedStatement
"C:\Program Files\Java\jdk-13.0.2\bin\java.exe" -Didea.laur
Do You want to see database table before updating??(Y/N):
Y
1, Dipendra, Kathmandu, 21,
2, Hari, Lalitpur, 23,
3, Geeta, Bhaktapur, 25,
4, Rita, Janakpur, 26,
5, Goma, Chitwan, 27,
10, Ramesh, Kailali, 36,

Process finished with exit code 0
```

If user wants to execute 'Update' operation directly without seeing details in database table.

```
UpdatePreparedStatement
"C:\Program Files\Java\jdk-13.0.2\bin\java.exe" -Didea.lau
Do You want to see database table before updating??(Y/N):
N
Enter the Id to update into database:
10
Enter the Name to update into database:
Suresh
Enter the District to update into database:
Jajarkot
Enter the Age to update into database:
70
Row updated Successfully!!

Process finished with exit code 0
```

Record '10' was successfully updated.

+ Options

	id	name	district	age
Edit Copy Delete	1	Dipendra	Kathmandu	21
Edit Copy Delete	2	Hari	Lalitpur	23
Edit Copy Delete	3	Geeta	Bhaktapur	25
Edit Copy Delete	4	Rita	Janakpur	26
Edit Copy Delete	5	Goma	Chitwan	27
Edit Copy Delete	10	Suresh	Jajarkot	70

Delete Operation Using PreparedStatement

```
package Q04_CrudPreparedStatement;

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

public class DeletePreparedStatement
{
    public static void main(String[] args) throws
                                                SQLException
  {
        String url = "jdbc:mariadb://localhost:3306
                                                /JavaJDBC";
        String username = "root";
        String password = "";
        Connection connection = DriverManager.
                getConnection(url, username, password);
```

```
        System.out.println("Do You want to see database
                            table before making Selection
                                        Query??(Y/N):");
        Scanner in = new Scanner(System.in);
        char userChoice = in.next().charAt(0);

        if (userChoice == 'Y')
        {
            String sql = "select * from students";
            PreparedStatement statement = connection.
                                    prepareStatement(sql);
            ResultSet resultSet = statement.
                                        executeQuery();
            while (resultSet.next())
           {
                System.out.printf("%d, %s, %s, %s, \n",
                        resultSet.getInt("id"),
                        resultSet.getString("name"),
                        resultSet.getString("district"),
                        resultSet.getString("age")
                );
            }
        }
        if (userChoice == 'N')
        {
            System.out.println("Enter the Id to Delete
                                        from database:");
            int id = in.nextInt();

            String sql = "delete from students where id=
                                                    ?";
            PreparedStatement statement = connection.
                                    prepareStatement(sql);

            statement.setInt(1, id);

            int rowDeleted = statement.executeUpdate();

            if (rowDeleted > 0)
                System.out.println("Row Deleted
                                        Successfully!!");
            else
                System.out.println("Row isn't
                                            Deleted!!");
            statement.close();
            connection.close();
        }
    }
}
```

Output:

If user wish to see the database table details before executing 'Delete' operation.

```
DeletePreparedStatement
"C:\Program Files\Java\jdk-13.0.2\bin\java.exe" -Didea.launcher.port=51
Do You want to see database table before making Selection Query??(Y/N):
Y
1, Dipendra, Kathmandu, 21,
2, Hari, Lalitpur, 23,
3, Geeta, Bhaktapur, 25,
4, Rita, Janakpur, 26,
5, Goma, Chitwan, 27,
10, Suresh, Jajarkot, 70,

Process finished with exit code 0
```

If user wants to execute 'Delete' operation directly without seeing details in database table.

```
DeletePreparedStatement
"C:\Program Files\Java\jdk-13.0.2\bin\java.exe" -Didea.launcher.port=520
Do You want to see database table before making Selection Query??(Y/N):
N
Enter the Id to Delete from database:
10
Row Deleted Successfully!!

Process finished with exit code 0
```

Record with id '10' was successfully removed.

+ Options

	id	name	district	age
Edit Copy Delete	1	Dipendra	Kathmandu	21
Edit Copy Delete	2	Hari	Lalitpur	23
Edit Copy Delete	3	Geeta	Bhaktapur	25
Edit Copy Delete	4	Rita	Janakpur	26
Edit Copy Delete	5	Goma	Chitwan	27

Check all *With selected:* Edit Copy Delete

5) Implement CRUD operations for student table in Swing. Ask for user input where applicable.
⇨

Program

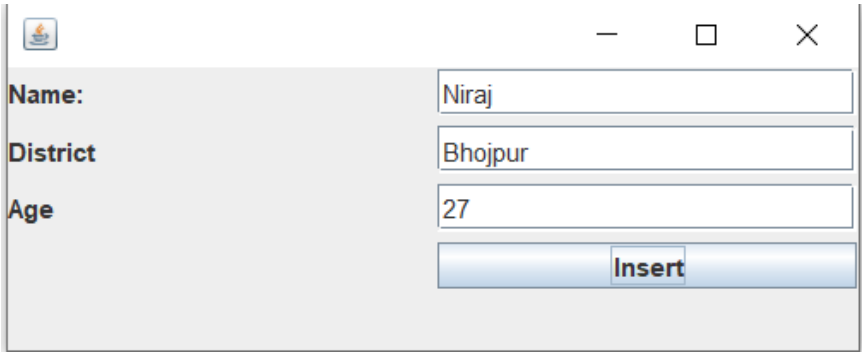
Insert Operation Using Swing

```
package Q05_CrudSwing;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

import static java.sql.Types.NULL;

public class CrudSwingInsert extends JFrame
{
    public static void main(String[] args)
    {
        CrudSwingInsert frame = new CrudSwingInsert();
        frame.setVisible(true);
    }

    String name,district;
    int age =  NULL;
    public CrudSwingInsert()
    {
        setLayout(new GridLayout(5,1,5,5));

        JLabel nameLabel = new JLabel("Name:");
        add(nameLabel);
        JTextField nameTextField = new JTextField(20);
        add(nameTextField);

        JLabel districtLabel = new JLabel("District");
        add(districtLabel);
        JTextField districtTextField = new JTextField(20);
        add(districtTextField);

        JLabel ageLabel = new JLabel("Age");
        add(ageLabel);
        JTextField ageTextField = new JTextField(20);
        add(ageTextField);

        JLabel blankLabel = new JLabel("");
        add(blankLabel);
        JButton insertButton = new JButton("Insert");
        add(insertButton);

        insertButton.addActionListener(new ActionListener()
```

```
        {
            @Override


        public void actionPerformed(ActionEvent actionEvent)
            {
                name = nameTextField.getText();
                district = districtTextField.getText();

                try
                  {
                     age = Integer.parseInt
                                    (ageTextField.getText());

                }
                catch (NumberFormatException e)
                {

                }
                if(name.equals("")|| district.equals("")||
                                                age == NULL)
                {

           JOptionPane.showMessageDialog(null,"Values aren't
                        acceptable!! Either fields are empty
                        or 'Age' is not an integer type.",
                        "Failure"
                                ,JOptionPane.ERROR_MESSAGE);
                }
                else                    {
                       InsertIntoDatabase();
                }
                }
        });

        pack();
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }

    public void InsertIntoDatabase()
    {

        try         {
            String url = "jdbc:mariadb://localhost:3306/
                                                JavaJDBC";
            String username = "root";
            String password = "";
            Connection connection = DriverManager.
                      getConnection(url, username, password);
            setLayout(new GridLayout(5,1,5,5));
            String sql = String.format("Insert into students
                       (name,district,age) Values (?,?,?)");
            PreparedStatement statement = connection.
                                        prepareStatement(sql);
```

```
            statement.setString(1,name);
            statement.setString(2,district);
            statement.setInt(3,age);
            int rowInserted = statement.executeUpdate();


            if(rowInserted>0)
                JOptionPane.showMessageDialog(null,"Row
                          Inserted Successfully","Success",
                           JOptionPane.INFORMATION_MESSAGE);
            else
                JOptionPane.showMessageDialog(null,"Row
                           Inserted Successfully","Failure"
                               ,JOptionPane.ERROR_MESSAGE);
            statement.close();
            connection.close();
        }
        catch (Exception e)
        {
            System.out.println("Error:"+ e.getMessage());
        }
    }
}
```

Output



+ Options

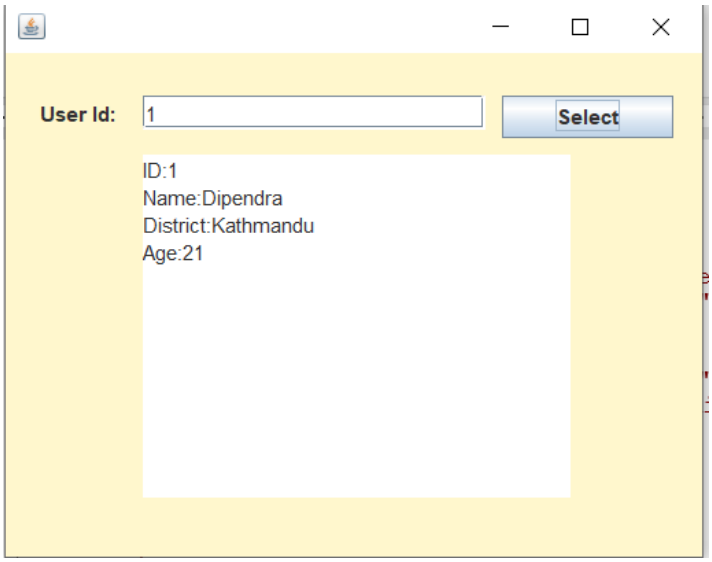
	id	name	district	age
Edit Copy Delete	1	Dipendra	Kathmandu	21
Edit Copy Delete	2	Hari	Lalitpur	23
Edit Copy Delete	3	Geeta	Bhaktapur	25
Edit Copy Delete	4	Rita	Janakpur	26
Edit Copy Delete	5	Goma	Chitwan	27
Edit Copy Delete	11	Niraj	Bhojpur	27

Select Operation Using Swing

```
package Q05_CrudSwing;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

public class CrudSwingSelect extends JFrame
{
    public static void main(String[] args)
    {
        CrudSwingSelect frame = new CrudSwingSelect();
        frame.setVisible(true);
        frame.setBounds(500,100,1000,500);
    }

    int id;
    JTextArea textArea;
    Container c;
    public CrudSwingSelect()
    {
         c = this.getContentPane();
         c.setLayout(null);

        //Setting the BackGround Color of the Frame.
        Color clr = new Color(255, 247, 205);
        c.setBackground(clr);

        //Adding 'UserName:' Label
        JLabel idLabel = new JLabel("User Id:");
        idLabel.setBounds(20,10,90,50);
        c.add(idLabel);

        //Adding name text field.
        JTextField idField = new JTextField();
        idField.setBounds(80,25,200,20);
        add(idField);


        //Adding 'Select' Button.
        JButton selectButton = new JButton("Select");
        selectButton.setBounds(290,25,100,25);
        add(selectButton);

        //Adding TextArea
        textArea = new JTextArea();
        textArea.setBounds(80,60,250,200);
        add(textArea);

        selectButton.addActionListener(new ActionListener()

        {
            @Override
            public void actionPerformed(ActionEvent actionEvent)
```

```
            {

                try
                {
                    id = Integer.parseInt(idField.getText());

                }
                catch (NumberFormatException e)
                {
                    JOptionPane.showMessageDialog(null,"Values aren't
                                acceptable!! 'ID' is not an integer
                            type.","Failure",JOptionPane.ERROR_MESSAGE);
                }

                 SelectFromDatabase();
            }
        });
        pack();
        setDefaultCloseOperation(EXIT_ON_CLOSE);

    }
    public void SelectFromDatabase()
    {
        try
        {   String url = "jdbc:mariadb://localhost:3306/JavaJDBC";
            String username = "root";
            String password = "";
            Connection connection = DriverManager.getConnection(url,
                                                    username, password);
            String sql = "select * from students where id = ?";
            PreparedStatement statement = connection.
                                                prepareStatement(sql);
            statement.setInt(1,id);
            ResultSet resultSet = statement.executeQuery();


            int id=0,age=0;
            String name="",district="";

            while (resultSet.next())
            {
                  id=resultSet.getInt("id");
                  name = resultSet.getString("name");
                  district = resultSet.getString("district");
                  age= resultSet.getInt("age");
            }
            textArea.setText("ID:" + id +"\nName:" +name +"\nDistrict:"
                                                +district+"\nAge:"+age);
            statement.close();
        }
        catch (Exception e)
        {
            System.out.println(e.getMessage());
        }
    }
}
```

Output



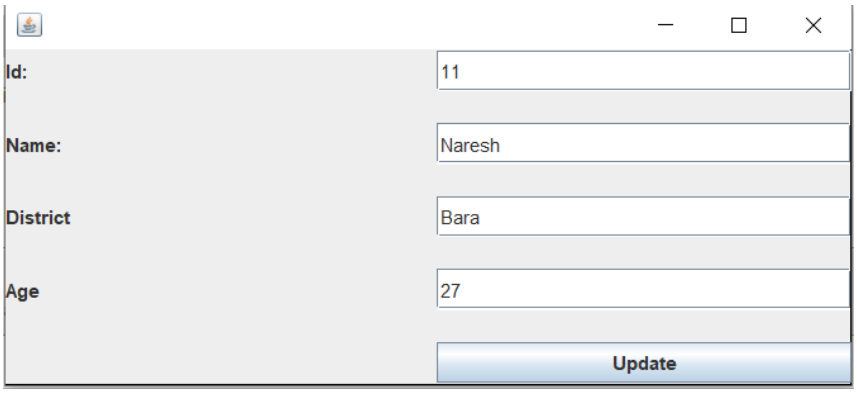
Update Operation Using Swing

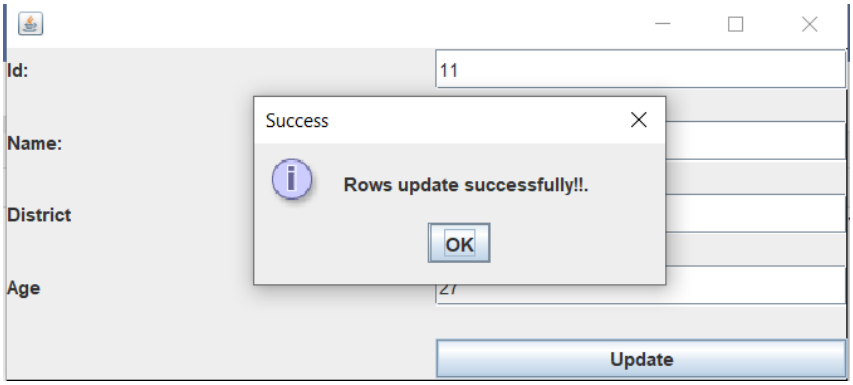
```
package Q05_CrudSwing;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class CrudSwingUpdate extends JFrame
{
    public static void main(String[] args)
    {
        CrudSwingUpdate frame = new CrudSwingUpdate();
        frame.setVisible(true);
        frame.setBounds(500,100,550,250);
    }

    public CrudSwingUpdate()
    {
        setLayout(new GridLayout(5,1,10,20));

        JLabel idLabel = new JLabel("Id:");
        add(idLabel);
        JTextField idTextField = new JTextField(20);
        add(idTextField);

        JLabel nameLabel = new JLabel("Name:");
        add(nameLabel);
        JTextField nameTextField = new JTextField(20);
        add(nameTextField);
```

```
        JLabel districtLabel = new JLabel("District");
        add(districtLabel);
        JTextField districtTextField = new
                                            JTextField(20);
        add(districtTextField);

        JLabel ageLabel = new JLabel("Age");
        add(ageLabel);
        JTextField ageTextField = new JTextField(20);
        add(ageTextField);

        JLabel blankLabel = new JLabel("");
        add(blankLabel);

        JButton updateButton = new JButton("Update");
        add(updateButton);

        updateButton.addActionListener(new
                                        ActionListener()
        {
            @Override
          public void actionPerformed(ActionEvent
                                            actionEvent)
            {
                try
                 {
                      int id = Integer.parseInt
                                  (idTextField.getText());
                      String  name = nameTextField.
                                                getText();
                      String district = districtTextField.
                                                getText();
                      int age = Integer.parseInt
                                   (ageTextField.getText());

                     UpdateDatabase(id,name,district,age);

                  }
                  catch (NumberFormatException e)
                  {

                JOptionPane.showMessageDialog(null,"Values
                      aren't acceptable!! Either Fields
                      are empty or type mismatched.
                   ","Failure",JOptionPane.ERROR_MESSAGE);
                  }
            }
        });
        pack();
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }
    public void UpdateDatabase(int id, String name,
                               String district, int age)
    {
```

```
        try
        {
            String url = "jdbc:mariadb://localhost:3306
                                                /JavaJDBC";
            String username = "root";
            String password = "";
            Connection connection = DriverManager.
                 getConnection(url, username, password);

            String sql = "update students set name =
                                ?,district= ?,age = ?
                                        where id= ?";
            PreparedStatement statement = connection.
                                prepareStatement(sql);

            statement.setString(1,name);
            statement.setString(2,district);
            statement.setInt(3,age);
            statement.setInt(4,id);

            int rowUpdated = statement.executeUpdate();

                if (rowUpdated > 0)
                JOptionPane.showMessageDialog(null,"Rows
                           update successfully!!.
                           ","Success",JOptionPane.
                                  INFORMATION_MESSAGE);

            else
                JOptionPane.showMessageDialog(null,"Rows
                    aren't updated!!","Failure"
                             ,JOptionPane.ERROR_MESSAGE);

            statement.close();
            connection.close();
        }
        catch(Exception e)
        {
            System.out.println(e.getMessage());
        }
    }
}
```

Output





+ Options

	id	name	district	age
Edit Copy Delete	1	Dipendra	Kathmandu	21
Edit Copy Delete	2	Hari	Lalitpur	23
Edit Copy Delete	3	Geeta	Bhaktapur	25
Edit Copy Delete	4	Rita	Janakpur	26
Edit Copy Delete	5	Goma	Chitwan	27
Edit Copy Delete	11	Naresh	Bara	27

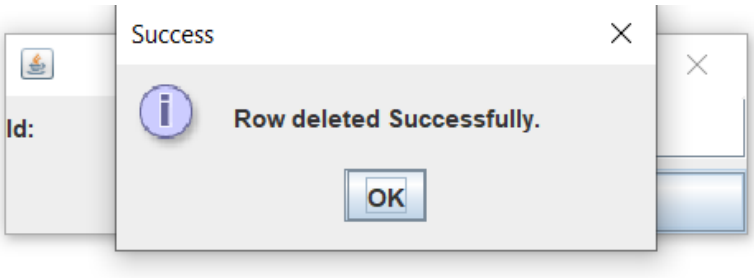
Check all *With selected:* Edit Copy

Delete Operation Using Swing

```
package Q05_CrudSwing;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class CrudSwingDelete extends JFrame
{
    public static void main(String[] args)
    {
        CrudSwingDelete frame = new CrudSwingDelete();
        frame.setVisible(true);
        frame.setBounds(500,100,500,100);
    }
    public CrudSwingDelete()
    {
        setLayout(new GridLayout(2,1,10,5));
        JLabel idLabel = new JLabel("Id:");
        add(idLabel);
        JTextField idTextField = new JTextField(20);
        add(idTextField);

        JLabel Label = new JLabel();
        add(Label);

        JButton deleteButton = new JButton("Delete");
        add(deleteButton);

        deleteButton.addActionListener(new ActionListener()
        {
            @Override
      public void actionPerformed(ActionEvent actionEvent)
            {
                try                    {
                    int id = Integer.parseInt
                                    (idTextField.getText());
                    DeleteFromDatabase(id);
                }
                catch (NumberFormatException e)
                {

                 JOptionPane.showMessageDialog(null,"Values
                        aren't acceptable!! Either fields
                        are empty or type mismatched.",
                      "Failure",JOptionPane.ERROR_MESSAGE);

                }
```

```
            }
        });
        pack();
        setDefaultCloseOperation(EXIT_ON_CLOSE);

    }
    public void DeleteFromDatabase(int id)
    {
        try
           {


            String url = "jdbc:mariadb://localhost:3306
                                                /JavaJDBC";
            String username = "root";
            String password = "";

            Connection connection = DriverManager.
                      getConnection(url, username, password);

            String sql = "delete from students where id= ?";
            PreparedStatement statement = connection.
                                       prepareStatement(sql);

            statement.setInt(1, id);

            int rowDeleted = statement.executeUpdate();


            if (rowDeleted > 0)
                JOptionPane.showMessageDialog(null,"Row
                                deleted Successfully."
               ,"Success",JOptionPane.INFORMATION_MESSAGE);

            else
                JOptionPane.showMessageDialog(null,"Row
                  can't be deleted.","Failure`
                                ",JOptionPane.ERROR_MESSAGE);

            statement.close();
            connection.close();

        }
        catch (Exception e)
        {

        }
    }
}
```

Output



+ Options

	id	name	district	age
Edit Copy Delete	1	Dipendra	Kathmandu	21
Edit Copy Delete	2	Hari	Lalitpur	23
Edit Copy Delete	3	Geeta	Bhaktapur	25
Edit Copy Delete	4	Rita	Janakpur	26
Edit Copy Delete	5	Goma	Chitwan	27

6) Implement account balance transfer operation (use transactions). Ask for user input where applicable

⇨

Program

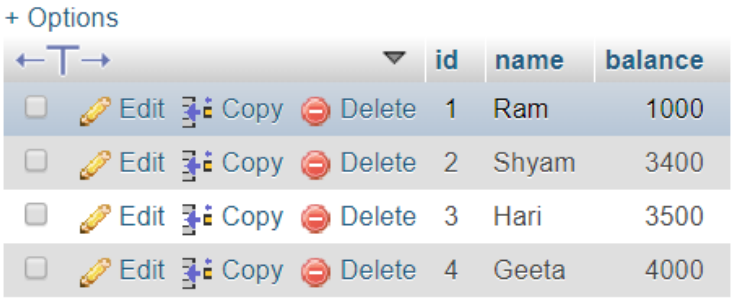
Deposit transaction

```
package Q06_Transaction;

import javax.swing.*;
import java.sql.*;
import java.util.*;

public class BalanceDeposit
{
    static Connection connection;
    public static void main(String[] args) throws Exception
{
        String url = "jdbc:mariadb://localhost:3306/Bank ";
        String username = "root";
        String password = "";
        connection = DriverManager.getConnection(url,
                                        username, password);

        Scanner in = new Scanner(System.in);
        System.out.println("**** Deposit *****");

        System.out.println("Enter id: ");
        int id = in.nextInt();

        System.out.println("Enter amount: ");
        double amount = in.nextDouble();

        try {
            String debitSql = "Update MyAccount set balance
                                = balance + ? where id= ?";
            PreparedStatement statement = connection.
                                prepareStatement(debitSql);
            statement.setDouble(1, amount);
            statement.setInt(2, id);
            statement.executeUpdate();
        }
        catch (SQLException e)
        {
            connection.rollback();
            System.out.println("Transaction Error!!");
        }
```

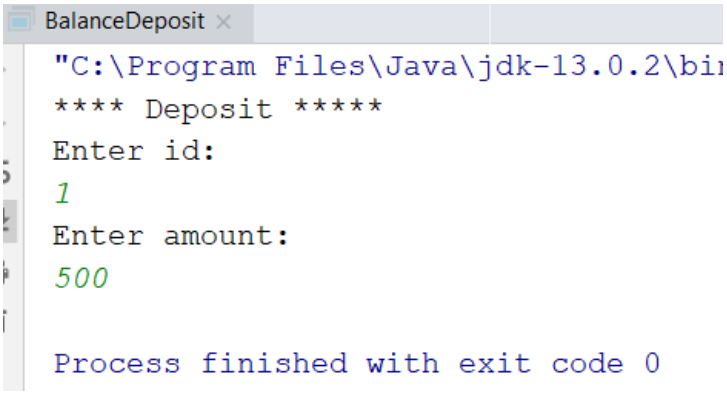
```
        }
}
```

Output

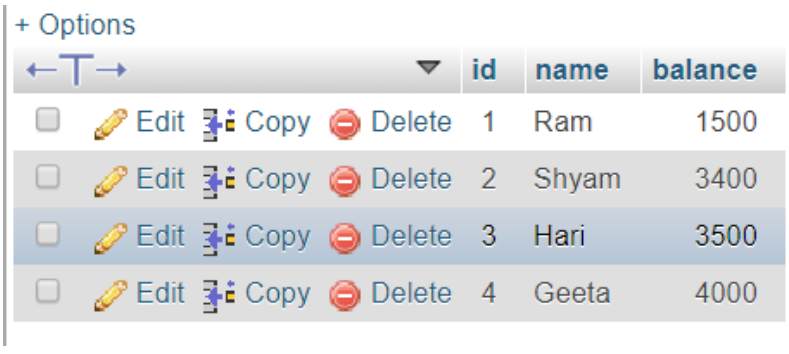
Database record before deposit transaction is performed



	id	name	balance
Edit Copy Delete	1	Ram	1000
Edit Copy Delete	2	Shyam	3400
Edit Copy Delete	3	Hari	3500
Edit Copy Delete	4	Geeta	4000



```
BalanceDeposit
"C:\Program Files\Java\jdk-13.0.2\bi
**** Deposit *****
Enter id:
1
Enter amount:
500

Process finished with exit code 0
```

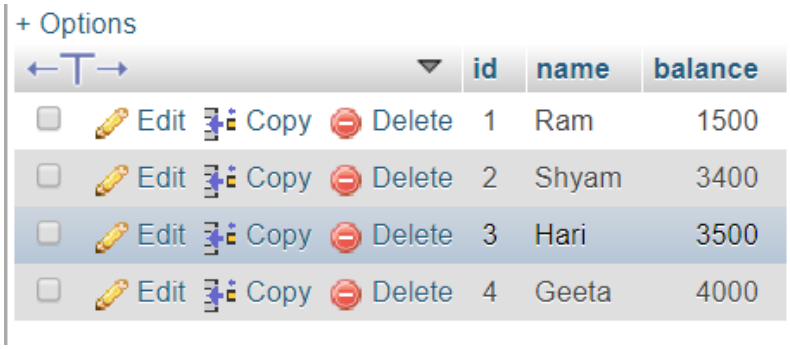


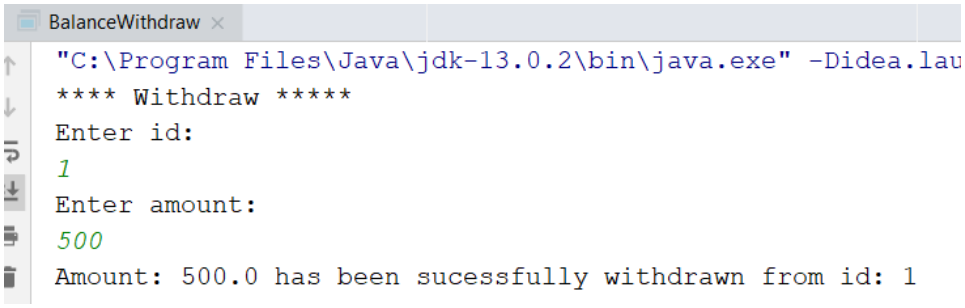
	id	name	balance
Edit Copy Delete	1	Ram	1500
Edit Copy Delete	2	Shyam	3400
Edit Copy Delete	3	Hari	3500
Edit Copy Delete	4	Geeta	4000

Withdraw Transaction

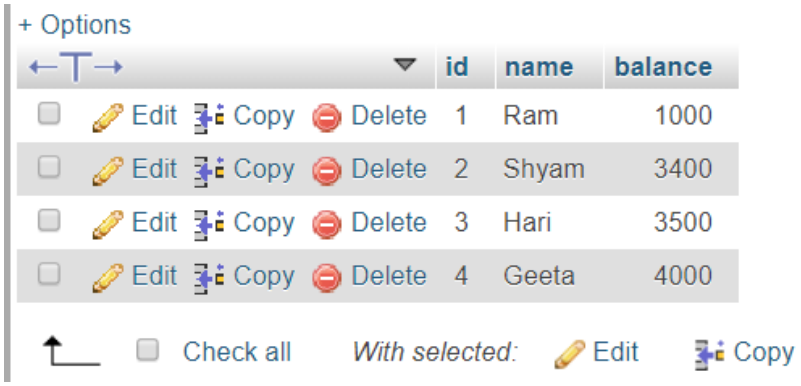
```
package Q06_Transaction;

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

public class BalanceWithdraw
{
    static Connection connection;
    public static void main(String[] args) throws Exception
   {
        String url = "jdbc:mariadb://localhost:3306/Bank";
        String username = "root";
        String password = "";
        connection = DriverManager.getConnection(url, username,
                                                          password);

        Scanner in = new Scanner(System.in);
        System.out.println("**** Withdraw *****");

        System.out.println("Enter id: ");
        int id = in.nextInt();

        System.out.println("Enter amount: ");
        double amount = in.nextDouble();

        try {
            String debitSql = "Update MyAccount set balance = balance
                                              - ? where id= ?";
            PreparedStatement statement = connection.
                                           prepareStatement(debitSql);
            statement.setDouble(1, amount);
            statement.setInt(2, id);
            statement.executeUpdate();

            connection.commit();
            System.out.println("Amount: " + amount + " has been
                                sucessfully withdrawn from id: "+ id);

        }
        catch (SQLException e)
        {
            connection.rollback();
            System.out.println("Transaction Error!!");
        }
    }
}
```

Database record before withdraw transaction is performed





After performing withdraw transaction.



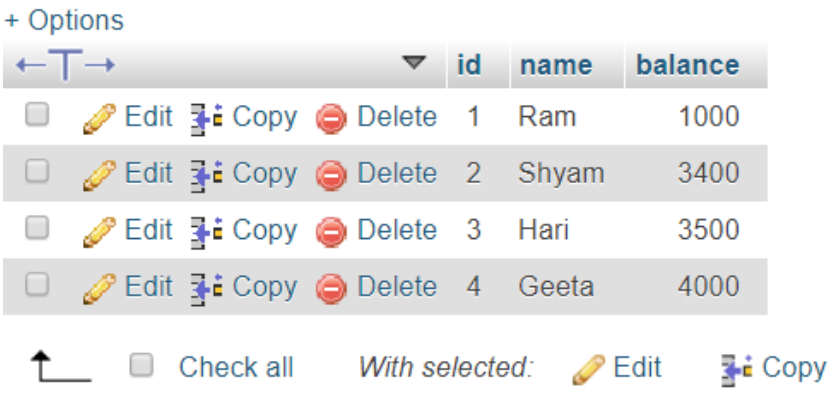
Transfer Transaction

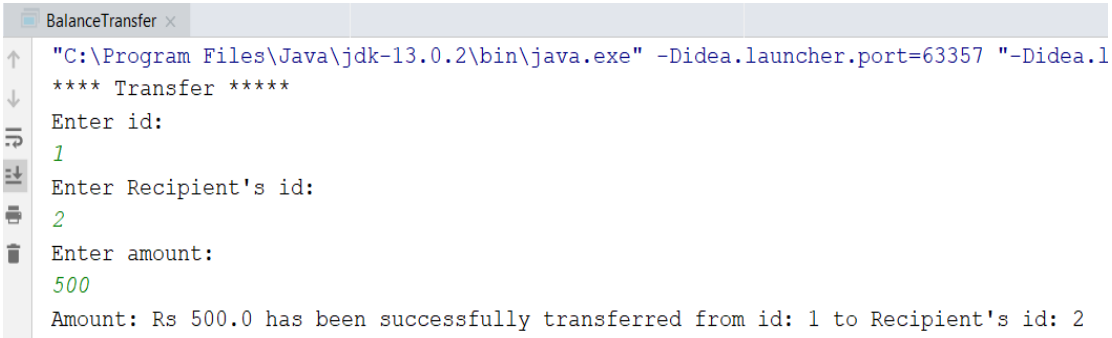
```
package Q06_Transaction;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import java.util.Scanner;

public class BalanceTransfer
{
    static Connection connection;
    public static void main(String[] args) throws Exception
      {
        String url = "jdbc:mariadb://localhost:3306/Bank";
        String username = "root";
        String password = "";
        connection = DriverManager.getConnection(url, username,
                                                        password);
        Scanner in = new Scanner(System.in);
        System.out.println("**** Transfer *****");

        System.out.println("Enter id: ");
        int id = in.nextInt();

        System.out.println("Enter Recipient's id: ");
        int recipientId = in.nextInt();

        System.out.println("Enter amount: ");
        double amount = in.nextDouble();
        try
         {
            String debitSql = "Update MyAccount set balance =
                                    balance - ? where id = ?";

            String creditSql = "Update MyAccount set balance =
                                   balance + ? where id = ?";

            PreparedStatement statement = connection.
                                        prepareStatement(debitSql);
            statement.setDouble(1, amount);
            statement.setInt(2, id);

            PreparedStatement statement1 = connection.
                                        prepareStatement(creditSql);
            statement1.setDouble(1, amount);
            statement1.setInt(2, recipientId);

            statement.executeUpdate();
            statement1.executeUpdate();
            connection.commit();
            System.out.println("Amount: Rs " + amount +" has been
```

```
                                    successfully transferred from
                                    id: " + id+ " to Recipient's id: "+
                                    recipientId);
            }



            catch (SQLException e)
            {
                connection.rollback();
                System.out.println("Transaction Error!!");
            }
        }
}
```

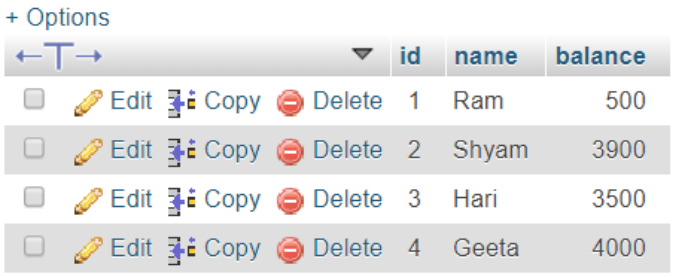
Output:

Database record before transfer transaction is performed





After performing withdraw transaction.


+ Options
id
name
balance
Edit Copy Delete 1 Ram 500
Edit Copy Delete 2 Shyam 3900
Edit Copy Delete 3 Hari 3500
Edit Copy Delete 4 Geeta 4000