JAVA AWT BASED- FOREX TRADING DATABASE SYSTEM -SQL CONNECTIVITY USING JDBC

A

Report

Submitted in partial fulfilment of the

BE 1V SEMESTER DATABASE MANAGEMENT SYSTEM

INFORMATION TECHNOLOGY

By

K.HEMA <1602-18-737-071>

Under the guidance of

B.Leelavathy

Department of Information Technology
Vasavi College of Engineering (Autonomous)
(Affiliated to Osmania University)
Ibrahimbagh, Hyderabad-31

BONAFIDE CERTIFICATE

This is to certify that the project report titled "**Forex Trading Database System**" project work of **Mrs.K.Hema** bearing Roll.no:**1602-18-737-071** who carried out this project under the guidance of **B.Leelavathy** during IV semester for the academic year **2019-2020.**

External Examiner

Internal Examiner

B.Leelavathy
Assistant professor
Department of Information Technology

DBMS Assignment 2
Title: Forex Trading Database System

ABSTRACT

This is project "**Forex Trading database systems"** is:It is a global decent.The foreign exchange (forex) market is the largest and most sophisticated market in the world for currency exchange. Forex trading takes place not on a centralized exchange as in the case of options, stock or futures, but through a wide variety of fx brokers. Nonetheless, money transfer comparison websites offer the most comprehensive and useful information you can find on the web. They do the legwork for you: they research the trends and the market, they compare exchange rates and brokers and list the best results based on your instructions. With all the information gathered, we only have to choose the best exchange rates.

This project help us to know how forex trading happens using database systems.

K.Hema
1602-18737-071

DBMS Assignment 2
Title: Forex Trading Database System

INTRODUCTION

A)REQUIREMENT ANALYSIS FOR FOREX TRADING DATABASE SYSTEM

List of Tables:

- Login
- Account
- transactions
- trade
- request_transaction
- does_trading
- customers_account

K.Hema
1602-18737-071

DBMS Assignment 2
Title: Forex Trading Database System

List of attributes with their domain types:

ENTITY	ATTRIBUTES	DOMAIN
Login	Customer id user name password customer name contact address	NUMBER(10) VARCHAR(15) NUMBER(10) VARCHAR(15) NUMBER(10) NUMBER(10)
does_trading	since	NUMBER(10)
Customers_account	since	NUMBER(10)
Transaction	Transaction id currency of	NUMBER(15) VARCHAR(15)
Account	Account id account type balance balance type	NUMBER(15) VARCHAR(15) NUMBER(20) VARCHAR(20)
Trade	Trading id adminstrator contact currency to address	NUMBER(10) VARCHAR(20) NUMBER(10) VARCHAR(15) NUMBER(15)
Request_Transaction	date	NUMBER(10)

K.Hema
1602-18737-071

B)AIM OF THE PROJECT

To create a Java GUI based form for the project Forex Trading Database Management System which takes the values like : Trade ID, contact ID, address,currencyto, administrator from the customer. These are the values to be updated in the database using JDBC connectivity .The values

entered (insertion, deletion, updation) by the user for the respective table in GUI should be updated in the database using JDBC.

C) ARCHITECTURE AND TECHNOLOGY USED

Java Eclipse, Oracle 11g Database, Java SE version 7, SQL*Plus.

SQL PLUS iis the most basic Oracle Database utility with a basic command-line interface, commonly used by users,administrators and programmers.

The interface of SQL Plus is used for creating the database. DDL and DML commands are implemented for operations being executed. The details of students, their logins, quiz, score are stored in the form of tables in the database.

Eclipse is an integrated development environment(IDE) used in computer programming. It contains a base workspace and an extensible plug-in system for customizing the environment. Eclipse is written mostly in java and its primary use is for developing java applicatons, but it may also be used to develop applications in other programming languages via plug-ins, including Erlang, Javascripts etc.

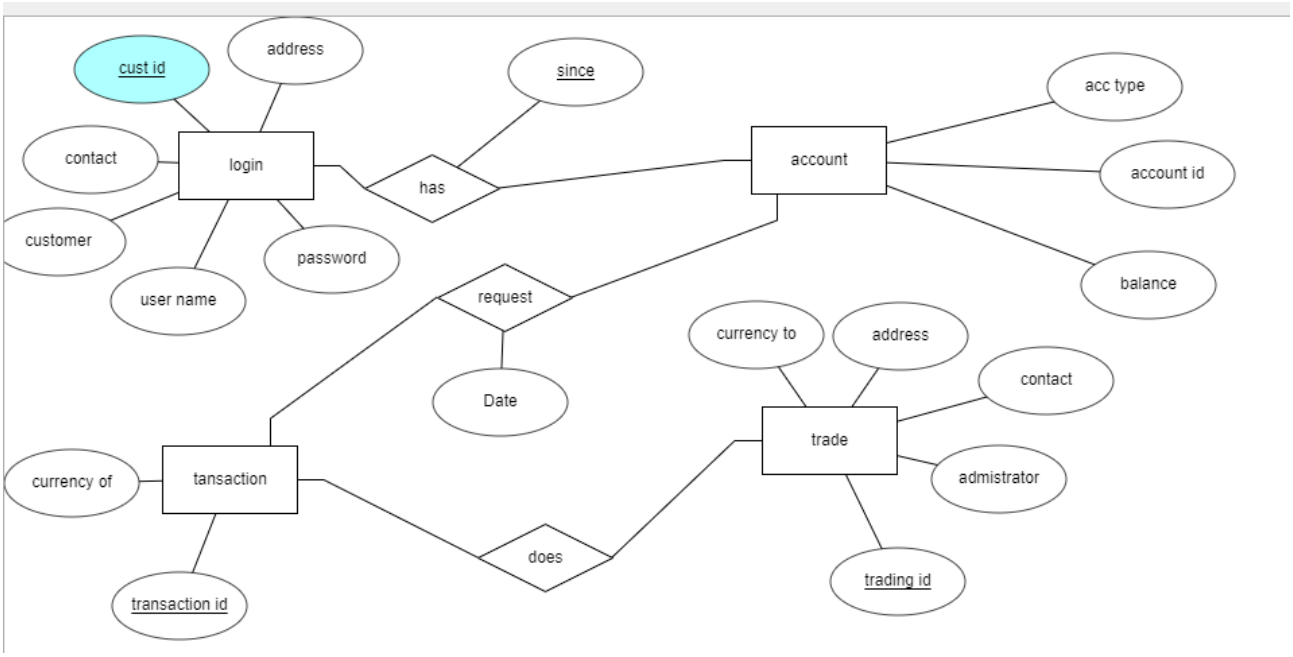
The front end application code is written in “Java” using eclipse. The portal for front end application is designed through Eclipse, runs and has the capacity to connect with the database which has data inserted using SQL.

Java AWT (Abstract Window Toolkit) is an API to develop GUI or window based applications in java.

Java AWT components are platform-dependent i.e. components are displayed according to the view of operating system. AWT is heavyweight i.e. its components are using the resources of OS.

The java.awt package provides classes for AWT API such as TextField, Label, TextArea, RadioButton, CheckBox, Choice, List etc.

D)ER Diagram:


cust id
address
since
acc type
contact
login
has
account
account id
customer
password
balance
user name
request
currency to
address
contact
Date
trade
currency of
tansaction
admistrator
does
trading id
transaction id

JAVA-SQL Connectivity using JDBC

Java Database Connectivity (JDBC) is an application programming interface (API) for the programming language Java, which defines how a client may access a database. It is a Java-based data access technology used for Java database connectivity. It is part of the Java Standard Edition platform, from Oracle Corporation. It provides methods to query and update data in a database and is oriented towards relational databases.

JDBC Connectivity

```
private void connToDb(){ try {
Class.forName(“oracle.jdbc.driver.OracleDriver”);
connection = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1522:xe","hema","vasavi  ");
statement = connection.createStatement();
}
 catch (SQLException connectException)
 {
 System.out.println(connectException.getMessage()); System.out.println(connectException.getSQLState());
System.out.println(connectException.getErrorCode());
System.exit(1);
 }
 catch (Exception e)
{
 System.err.println("Unable to find and load driver");
 System.exit(1);
 }
 }
```

Thus, the connection from Java to Oracle database is performed and therefore, can be used for updating tables in the database directly.

E) IMPLEMENTATION

Below is the code for the table Trade

Insert Trade:

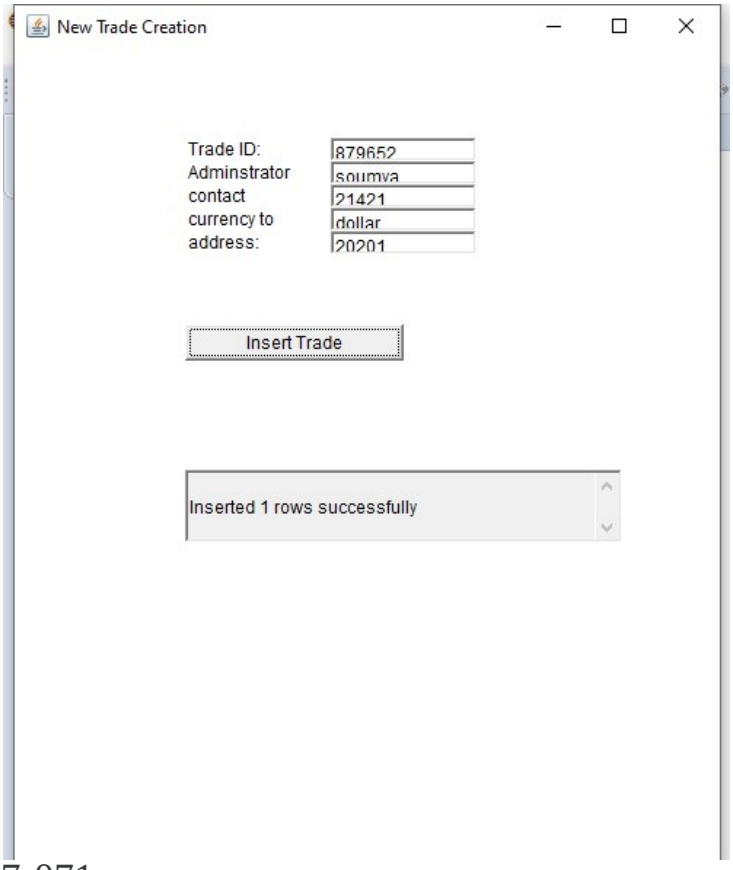
```
import java.awt.*;
import java.awt.event.*
import java.sql.*;
public class InsertTrade extends Frame
{
	Button insertTradeButton;
	TextField tridText, contactText, administratorText, currencytoText,addressText;
	TextArea errorText;
	Connection connection;
	Statement statement;
	public InsertTrade()
	{
		try
		{
			Class.forName("oracle.jdbc.driver.OracleDriver");
		}
		catch (Exception e)
		{
			System.err.println("Unable to find and load driver");
			System.exit(1);
		}
		connectToDB();
	}

	public void connectToDB()
  {
		try
```

```
                {
                 connection =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","hema","vasavi");
                 statement = connection.createStatement();


                }
        catch (SQLException connectException)
                {
                 System.out.println(connectException.getMessage());
                 System.out.println(connectException.getSQLState());
                 System.out.println(connectException.getErrorCode());


                 System.exit(1);
                }
   }
        public void buildGUI()
        {
                //Handle Insert Account Button
                insertTradeButton = new Button("Insert Trade");
                insertTradeButton.addActionListener(new ActionListener()
                {
                        public void actionPerformed(ActionEvent e)
                        {
                                try
                                {
                                  //String query = "INSERT INTO sailors (SID,SNAME, RATING, AGE) VALUES
(2,'Divya',7,20)";
                                        String query= "INSERT INTO trade VALUES("+"'" + tridText.getText() +
"','" + administratorText.getText() + "','" + contactText.getText() + "','" + currencytoText.getText() + "','" +
addressText.getText() +  "'" +")";
                                  int i = statement.executeUpdate(query);
```

```
                                    errorText.append("\nInserted " + i + " rows successfully");
                                }
                                catch (SQLException insertException)
                                {Frame f=new Frame();
OptionPane.showmessagedialog(f,”enter correct values”);
                                    displaySQLErrors(insertException);
                                }
                            }
                        });
                    tridText = new TextField(15);
                      administratorText = new TextField(15);
                      contactText = new TextField(100);


            currencytoText = new TextField(100);
            addressText = new TextField(100);
errorText = new TextArea(10, 40);
                      errorText.setEditable(false);


                      Panel first = new Panel();
                      first.setLayout(new GridLayout(6,3));
                      first.add(new Label("Trade ID:"));
                      first.add(tridText);
                      first.add(new Label("Adminstrator"));
                      first.add(administratorText);
                      first.add(new Label("contact"));
                      first.add(contactText);
                      first.add(new Label("currency to"));
                      first.add(currencytoText);
                      first.add(new Label("address:"));
                      first.add(addressText);
```

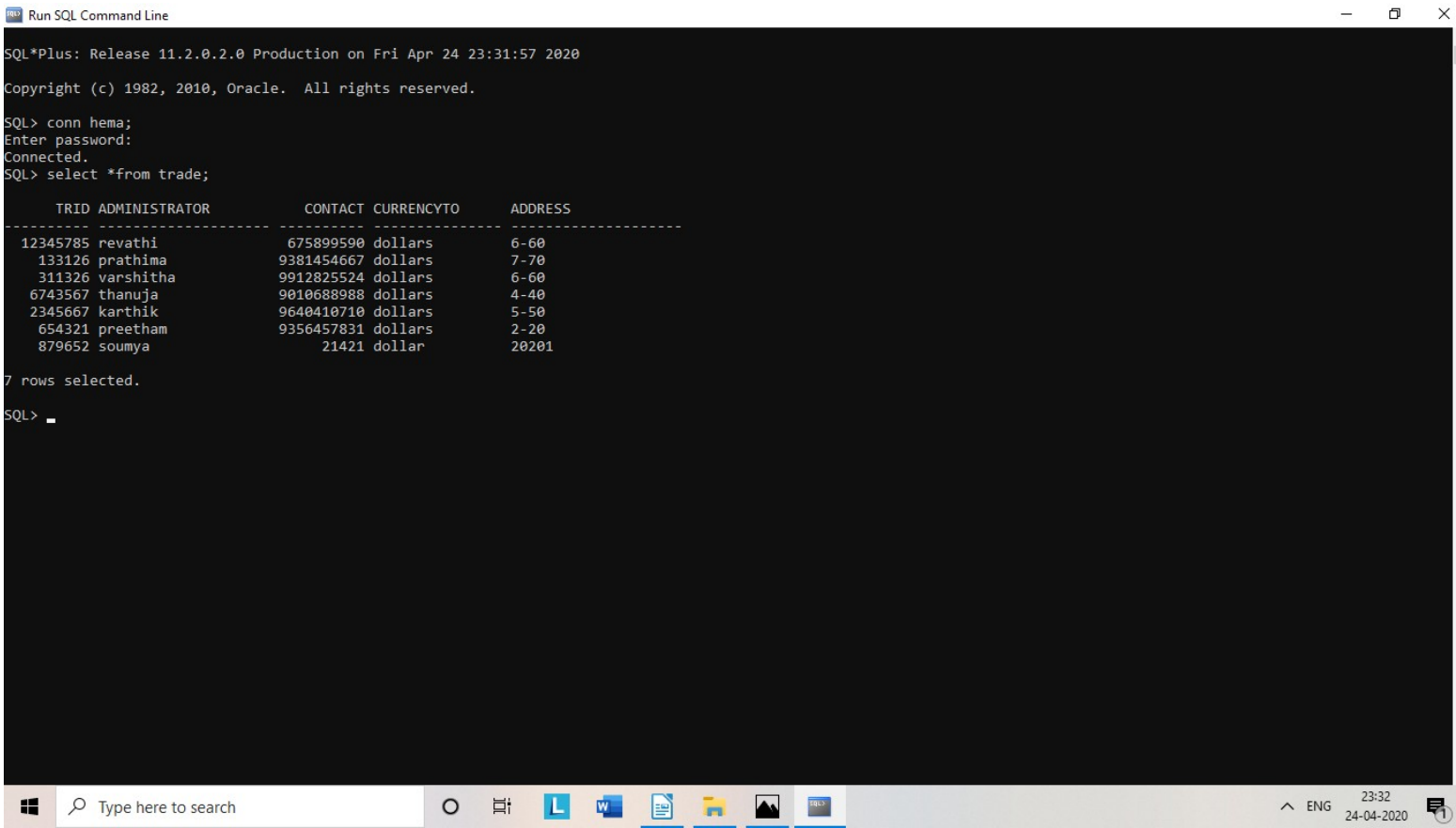
```
                    first.setBounds(125,90,200,100);


                    Panel second = new Panel(new GridLayout(4,1));
                    second.add(insertTradeButton);
      second.setBounds(125,220,150,100);
          Panel third = new Panel(new GridLayout(4,1))
third.add(errorText);
                    third.setBounds(125,320,300,200);
           setLayout(null);
                    add(second);
                    add(third);


                    setTitle("New Trade Creation");
                    setSize(500, 600);
                    setVisible(true);


}
private void displaySQLErrors(SQLException e)
          {
                    errorText.append("\nSQLException: " + e.getMessage() + "\n");
                    errorText.append("SQLState:     " + e.getSQLState() + "\n");
                    errorText.append("VendorError:  " + e.getErrorCode() + "\n");
          }
         public static void main(String[] args)
          {
                    InsertTrade itrade = new InsertTrade();


                    itrade.addWindowListener(new WindowAdapter(){
                      public void windowClosing(WindowEvent e)
           {
```

```
                System.exit(0);
            }
        });
        itrade.buildGUI()
```



DBMS Assignment 2

Title: Forex Trading Database System



UPDATE TRADE:

```
import java.awt.*;
import java.awt.event.*;
import java.sql.*;
public class UpdateTrade extends Frame
{
        Button updateTradeButton;
        List TradeIDList;
        TextField tridText, administratorText,contactText, currencyToText,addressText;
        TextArea errorText;
        Connection connection;
        Statement statement;
        ResultSet rs;
        public UpdateTrade()
        {
                try
                {
                        Class.forName("oracle.jdbc.driver.OracleDriver");
                }
                catch (Exception e)
                {
                        System.err.println("Unable to find and load driver");
                        System.exit(1);
                }
                connectToDB();
        }
       public void connectToDB()
  {
```

```
                    try
                    {
                     connection =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","hema","vasavi");
                     statement = connection.createStatement();


                    }
                    catch (SQLException connectException)
                    {
                     System.out.println(connectException.getMessage());
                     System.out.println(connectException.getSQLState());
                     System.out.println(connectException.getErrorCode());
System.exit(1);
                    }
   }
        private void loadTrade()
        {
                    try
                    {
                     rs = statement.executeQuery("SELECT TRID FROM TRADE");
                     while (rs.next())
                     {
                              TradeIDList.add(rs.getString("TRID"));
                     }
                    }
                    catch (SQLException e)
                    {
                     displaySQLErrors(e);
                    }
        }
```

```
public void buildGUI()
{
   TradeIDList = new List(10);
        loadTrade();
        add(TradeIDList);

        //When a list item is selected populate the text fields
        TradeIDList.addItemListener(new ItemListener()
        {
                public void itemStateChanged(ItemEvent e)
                {
                        try{
                                rs = statement.executeQuery("SELECT * FROM TRADE where TRID
='"+TradeIDList.getSelectedItem()+"'");
                                rs.next();
                                tridText.setText(rs.getString("TRID"));
                                administratorText.setText(rs.getString("ADMINISTRATOR"));
                                contactText.setText(rs.getString("CONTACT"));
                                currencyToText.setText(rs.getString("CURRENCYTO"));
                                addressText.setText(rs.getString("ADDRESS"));




                        }
                        catch (SQLException selectException)
                        {
                                displaySQLErrors(selectException);
                        }
                }
        });
```

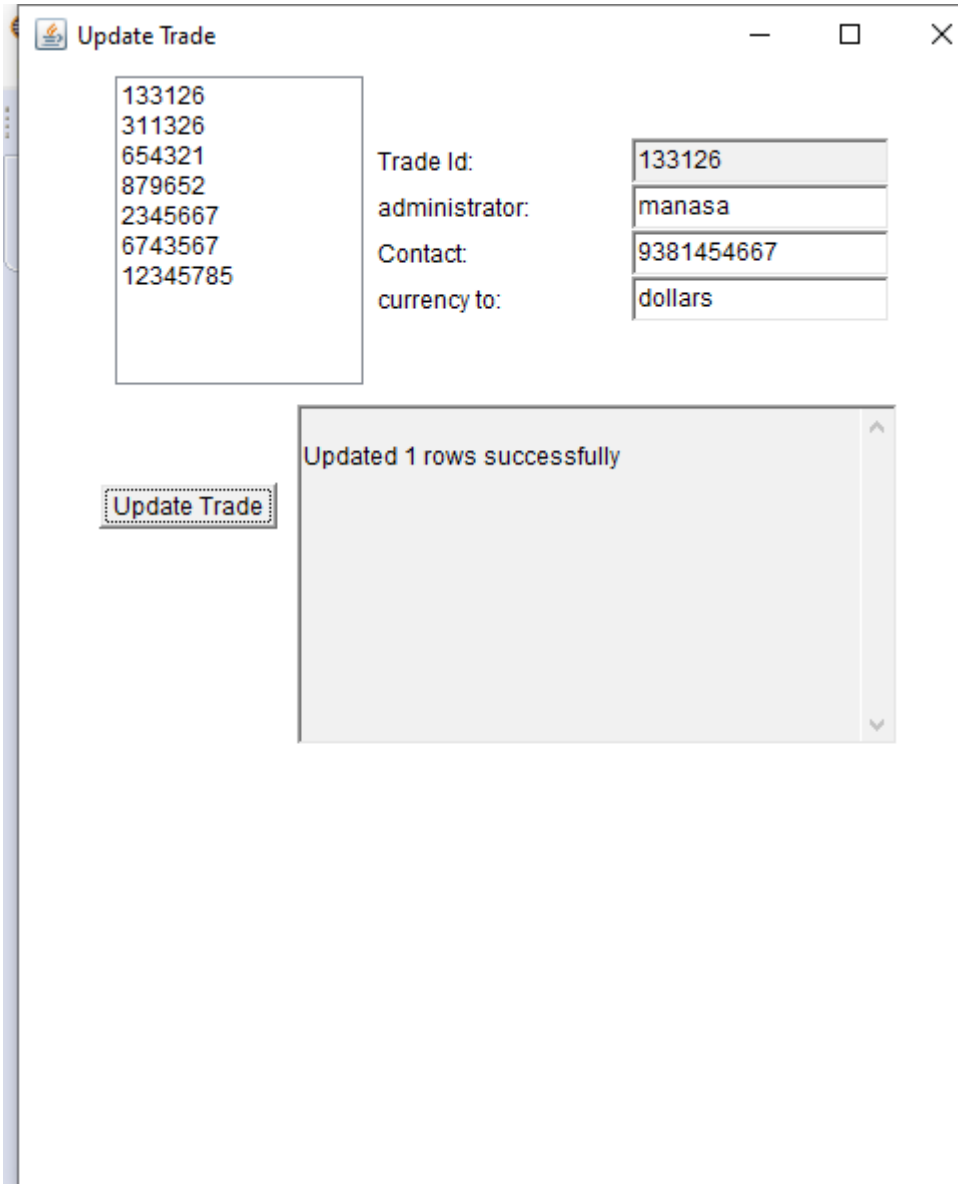
```
//Handle Update Sailor Button+"");

updateTradeButton = new Button("Update Trade");
updateTradeButton.addActionListener(new ActionListener()
{
	public void actionPerformed(ActionEvent e)
	{
		try
		{
		{
			Statement statement = connection.createStatement();
			int i = statement.executeUpdate("UPDATE TRADE "
			+ "SET administrator='" + administratorText.getText() +"', "
					+ "contact='" + contactText.getText() + "', " +
			 "currencyto ='" + currencyToText.getText() +"',"

					+ "address='" + addressText.getText() +
					"'where trid='" + TradeIDList.getSelectedItem()+"");
						errorText.append("\nUpdated " + i + " rows
successfully");
			TradeIDList.removeAll();
			loadTrade();
		}
		catch (SQLException insertException)
		{
			displaySQLErrors(insertException);
		}
	}
});
```

```
tridText = new TextField(15);
tridText.setEditable(false);
                administratorText = new TextField(15);
                contactText = new TextField(15);
                {
                currencyToText = new TextField(15);
                addressText = new TextField(15);

                errorText = new TextArea(10, 40);
                errorText.setEditable(false);

                Panel first = new Panel();
                first.setLayout(new GridLayout(4, 2));
                first.add(new Label("Trade Id:"));

                first.add(tridText);
                first.add(new Label("administrator:"));
                first.add(administratorText);
                first.add(new Label("Contact:"));
                first.add(contactText);
                first.add(new Label("currency to:"));
                first.add(currencyToText);

                Panel second = new Panel(new GridLayout(4, 1));
                second.add(updateTradeButton);

                Panel third = new Panel();
                third.add(errorText);

                add(first);
```

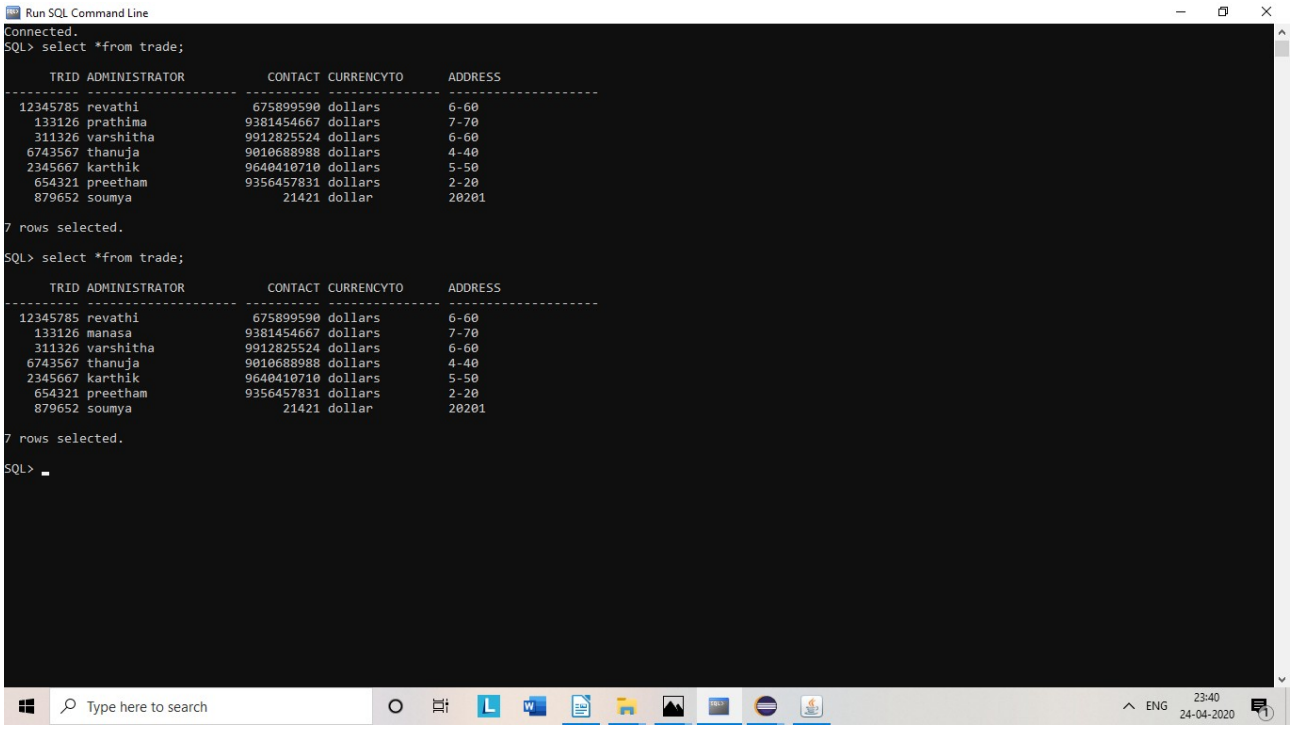
```
            add(second);
            add(third);

            setTitle("Update Trade");

            setSize(500, 600);
            setLayout(new FlowLayout());
            setVisible(true);

    }

    private void displaySQLErrors(SQLException e)
    {
            errorText.append("\nSQLException: " + e.getMessage() + "\n");
            errorText.append("SQLState:     " + e.getSQLState() + "\n");
            errorText.append("VendorError:  " + e.getErrorCode() + "\n");
    }
    public static void main(String[] args)
    {
            UpdateTrade uTrade = new UpdateTrade();

            uTrade.addWindowListener(new WindowAdapter(){
              public void windowClosing(WindowEvent e)
              {
                      System.exit(0);
              }
            });

            uTrade.buildGUI();
    }
```

}

{


Update Trade
133126
311326
654321
879652
2345667
6743567
12345785
Trade Id:
133126
administrator:
manasa
Contact:
9381454667
currency to:
dollars
Updated 1 rows successfully
Update Trade

{


Run SQL Command Line
Connected.
SQL> select *from trade;
TRID ADMINISTRATOR CONTACT CURRENCYTO ADDRESS
12345785 revathi 675899590 dollars 6-60
133126 prathima 9381454667 dollars 7-70
311326 varshitha 9912825524 dollars 6-60
6743567 thanuja 9010688988 dollars 4-40
2345667 karthik 9640410710 dollars 5-50
654321 preetham 9356457831 dollars 2-20
879652 soumya 21421 dollar 20201
7 rows selected.
SQL> select *from trade;
TRID ADMINISTRATOR CONTACT CURRENCYTO ADDRESS
12345785 revathi 675899590 dollars 6-60
133126 manasa 9381454667 dollars 7-70
311326 varshitha 9912825524 dollars 6-60
6743567 thanuja 9010688988 dollars 4-40
2345667 karthik 9640410710 dollars 5-50
654321 preetham 9356457831 dollars 2-20
879652 soumya 21421 dollar 20201
7 rows selected.
SQL>
Type here to search
ENG
23:40
24-04-2020

DELETE TRADE:

```
import java.awt.*;
import java.awt.event.*;
import java.sql.*;
public class DeleteTrade extends Frame
{
	Button deleteTradeButton;

	List TradeIDList;
	TextField tridText, administratorText, contactText, currencytoText,addressText;
	TextArea errorText;
	Connection connection;
	Statement statement;
	ResultSet rs;

	public DeleteTrade() {
		try
		{
			Class.forName("oracle.jdbc.driver.OracleDriver");
		}
		catch (Exception e)
		{
			System.err.println("Unable to find and load driver");
			System.exit(1);
		}
		connectToDB();
	}

	public void connectToDB()
```

```
   {
                    try
                    {
                     connection =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","hema","vasavi");
                     statement = connection.createStatement();


                    }
                    catch (SQLException connectException)
{
                    {
                     System.out.println(connectException.getMessage());
                     System.out.println(connectException.getSQLState());
                     System.out.println(connectException.getErrorCode());
                     System.exit(1);
                    }
   }


          private void loadTrade()
          {
                    try
                    {
                     rs = statement.executeQuery("SELECT * FROM TRADE");
                     while (rs.next())
                     {
                              TradeIDList.add(rs.getString("TRID"));
                     }
                    }
                    catch (SQLException e)
                    {
                     displaySQLErrors(e);
```

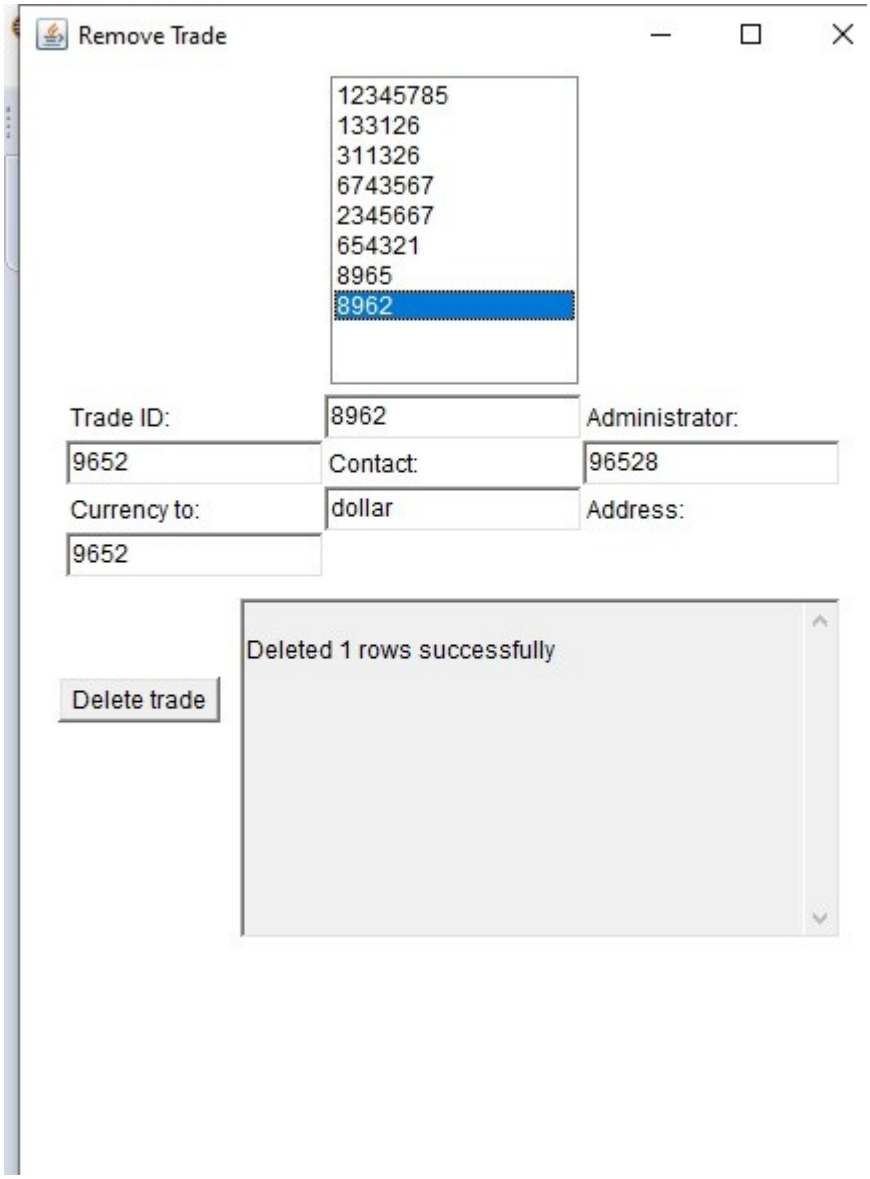
```
            }
    }

    public void buildGUI()
    {
       TradeIDList = new List(10);
            loadTrade();
            add(TradeIDList);
            {
            //When a list item is selected populate the text fields
            TradeIDList.addItemListener(new ItemListener()
            {
                    public void itemStateChanged(ItemEvent e)
                    {
                            try
                            {
                                    rs = statement.executeQuery("SELECT * FROM trade");
                                    while (rs.next())
                                    {
                                            if (rs.getString("TRID").equals(TradeIDList.getSelectedItem()))
                                            break;
                                    }
                                    if (!rs.isAfterLast())
                                    {
                                            tridText.setText(rs.getString("TRID"));
                                            administratorText.setText(rs.getString("ADMINISTRATOR"));
                                            contactText.setText(rs.getString("CONTACT"));
                                            currencytoText.setText(rs.getString("CURRENCYTO"));
                                            addressText.setText(rs.getString("ADDRESS"));
```

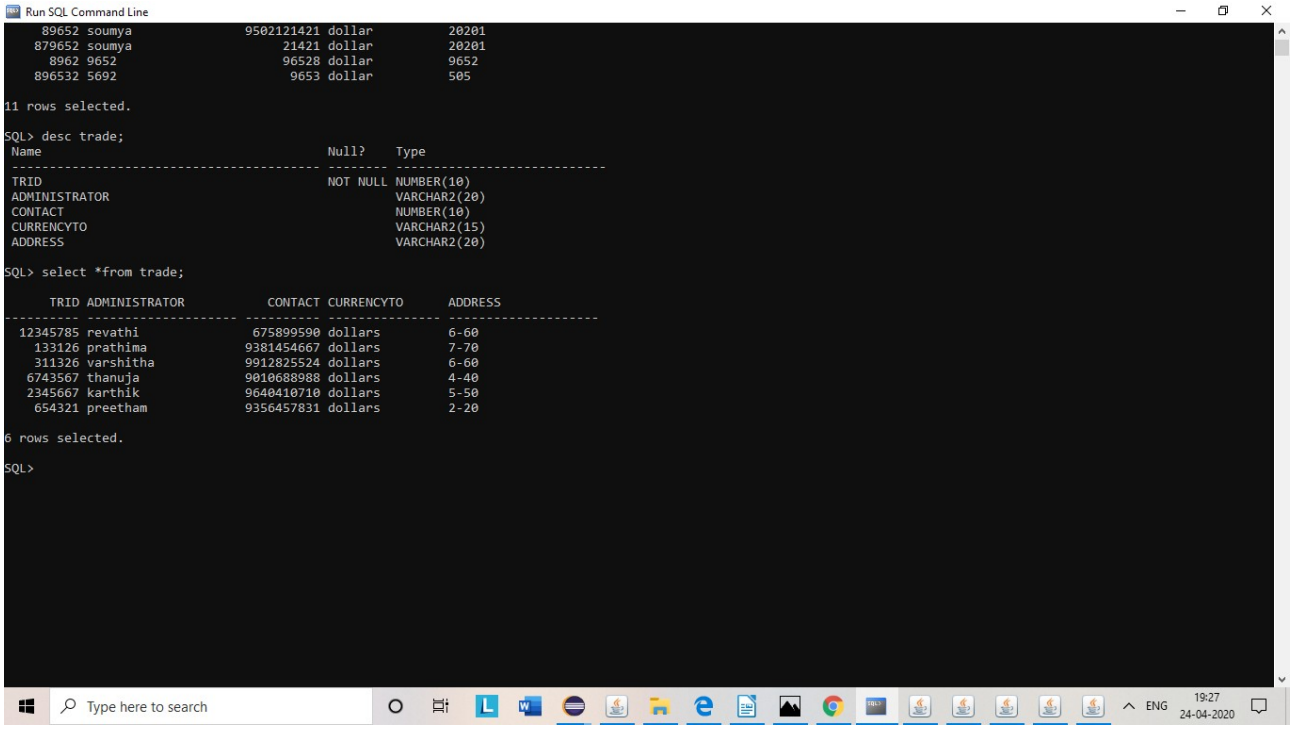
```
                                        }
                                }
                                catch (SQLException selectException)
                                {
                                        displaySQLErrors(selectException);
                                }
                        }
                });
{
                //Handle Delete Sailor Button
                deleteTradeButton = new Button("Delete trade");
                deleteTradeButton.addActionListener(new ActionListener()
                {
                        public void actionPerformed(ActionEvent e)
                        {
                                try
                                {
                                        Statement statement = connection.createStatement();
                                        int i = statement.executeUpdate("DELETE FROM trade WHERE trid =
"'+TradeIDList.getSelectedItem()+"' and administrator='"+ administratorText.getText()+"' and
contact='"+contactText.getText()+"' and currencyTo='"+ currencytoText.getText()+"' and address='"+
addressText.getText()+"'");
                                        errorText.append("\nDeleted "+i+ "rows successfully");
                                        tridText.setText(null);
                                        administratorText.setText(null);
                                        contactText.setText(null);
                                        currencytoText.setText(null);
                                        addressText.setText(null);
                                        TradeIDList.removeAll();
                                        loadTrade();
                                }
```

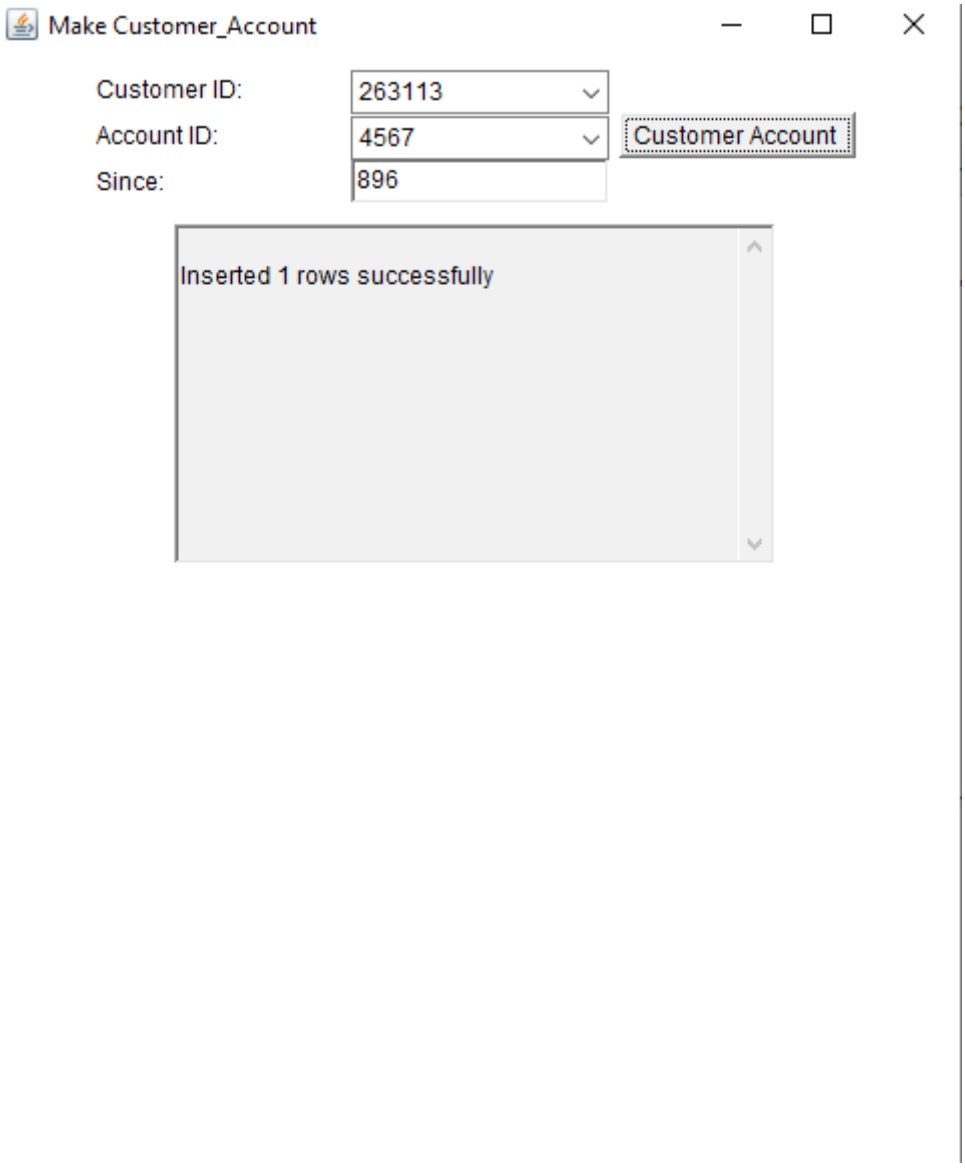
```
                                catch (SQLException insertException)
                                {
                                        displaySQLErrors(insertException);
                                }
                        }
                });
                {                       tridText = new TextField(15);
                administratorText = new TextField(15);
                contactText = new TextField(15);
        currencytoText = new TextField(15);
        addressText = new TextField(15);

                errorText = new TextArea(10, 40);
                errorText.setEditable(false);

                Panel first = new Panel();
                first.setLayout(new GridLayout(4, 2));
                first.add(new Label("Trade ID:"));
                first.add(tridText);
                first.add(new Label("Administrator:"));
                first.add(administratorText);
                first.add(new Label("Contact:"));
                first.add(contactText);
                first.add(new Label("Currency to:"));
                first.add(currencytoText);
                first.add(new Label("Address:"));
                first.add(addressText);


                Panel second = new Panel(new GridLayout(4, 1));
```

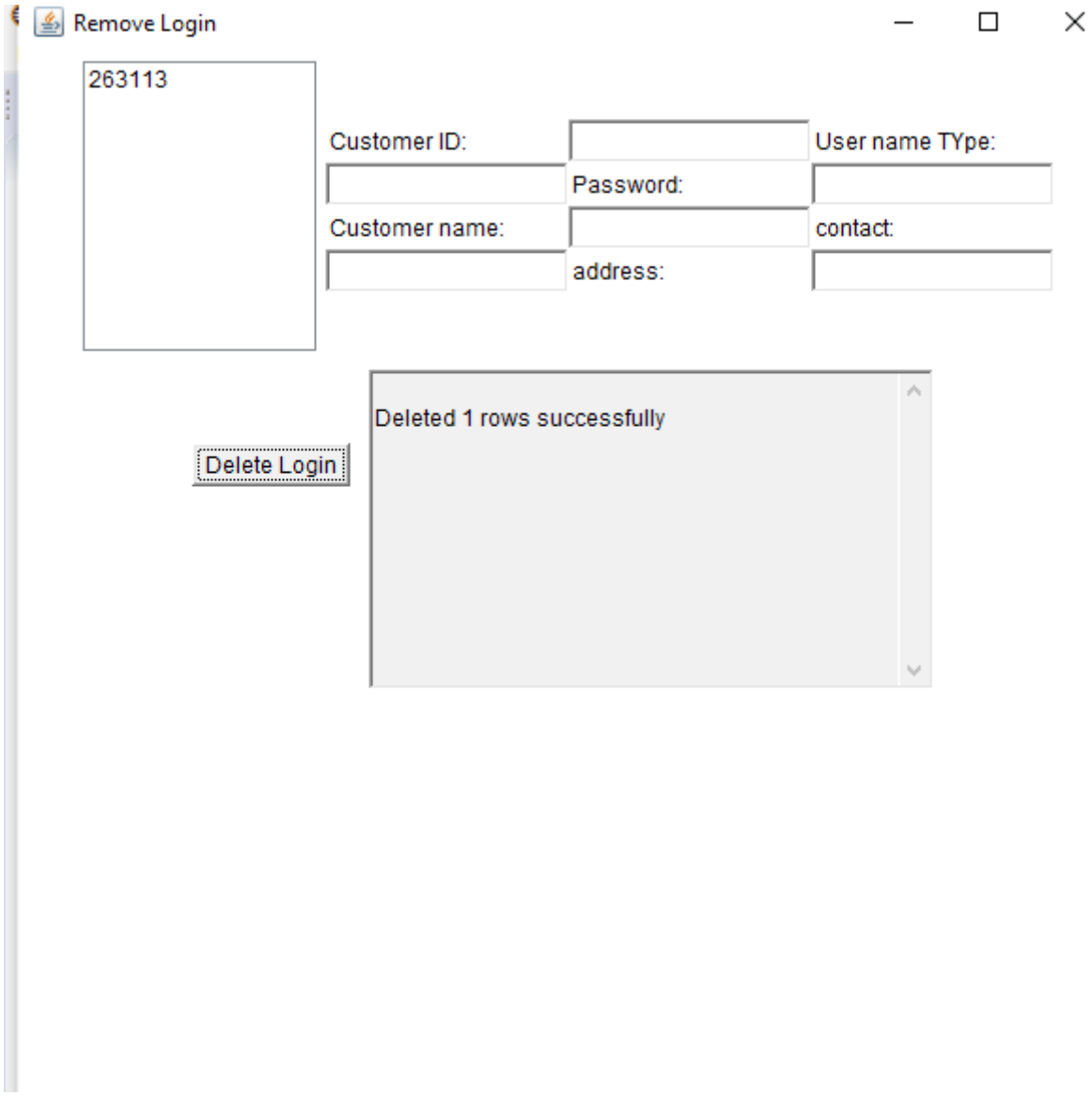
```
                second.add(deleteTradeButton);


                Panel third = new Panel();
                third.add(errorText);
                add(first);
                add(second);
        {
        add(third);


                setTitle("Remove Trade");
                setSize(450, 600);
                setLayout(new FlowLayout());
                setVisible(true);


        }
private void displaySQLErrors(SQLException e)
        {
                errorText.append("\nSQLException: " + e.getMessage() + "\n");
                errorText.append("SQLState:     " + e.getSQLState() + "\n");
                errorText.append("VendorError:  " + e.getErrorCode() + "\n");
        }
public static void main(String[] args)
        {
                DeleteTrade dtrade = new DeleteTrade();


                dtrade.addWindowListener(new WindowAdapter(){
                  public void windowClosing(WindowEvent e)
                  {
                        System.exit(0);
                  }
```

```
        });


        dtrade.buildGUI();
    }
}
```



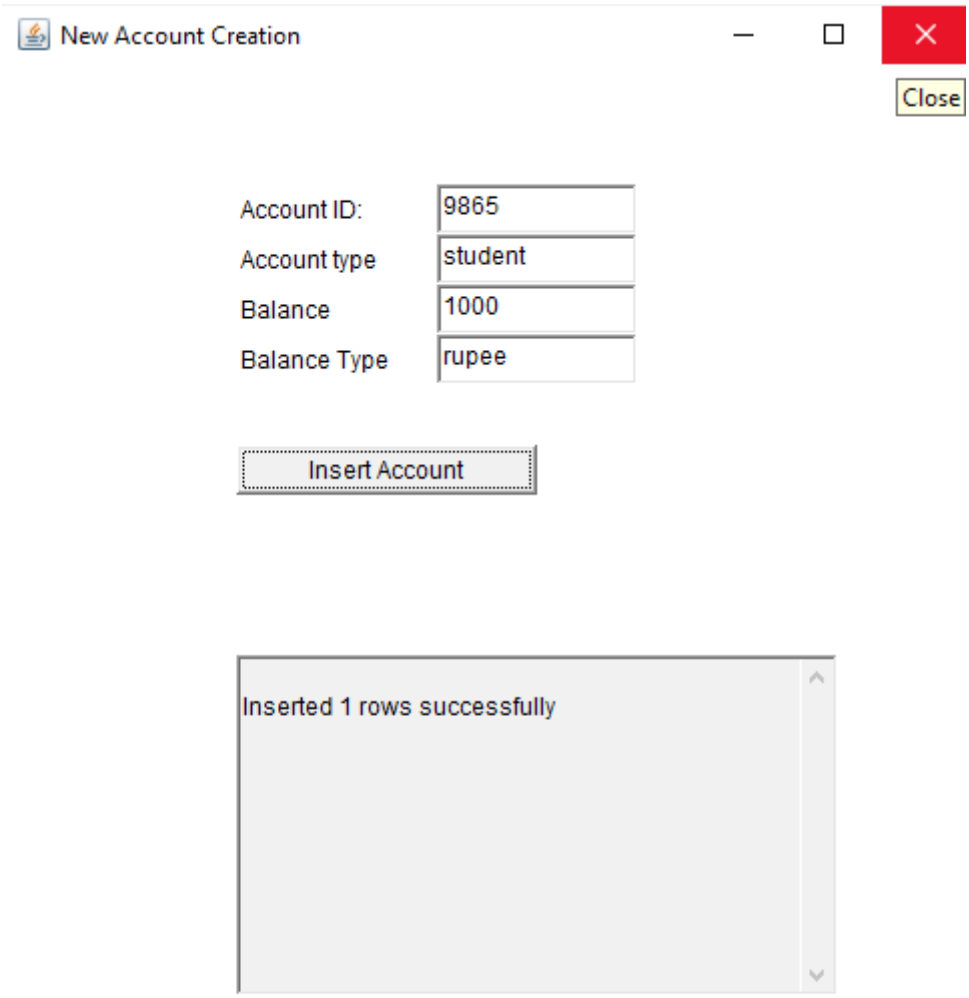

Run SQL Command Line
89652 soumya 9502121421 dollar 20201
879652 soumya 21421 dollar 20201
8962 9652 96528 dollar 9652
896532 5692 9653 dollar 505
11 rows selected.
SQL> desc trade;
Name Null? Type
TRID NOT NULL NUMBER(10)
ADMINISTRATOR VARCHAR2(20)
CONTACT NUMBER(10)
CURRENCYTO VARCHAR2(15)
ADDRESS VARCHAR2(20)
SQL> select *from trade;
TRID ADMINISTRATOR CONTACT CURRENCYTO ADDRESS
12345785 revathi 675899590 dollars 6-60
133126 prathima 9381454667 dollars 7-70
311326 varshitha 9912825524 dollars 6-60
6743567 thanuja 9010688988 dollars 4-40
2345667 karthik 9640410710 dollars 5-50
654321 preetham 9356457831 dollars 2-20
6 rows selected.
SQL>
Type here to search
ENG
19:27
24-04-2020

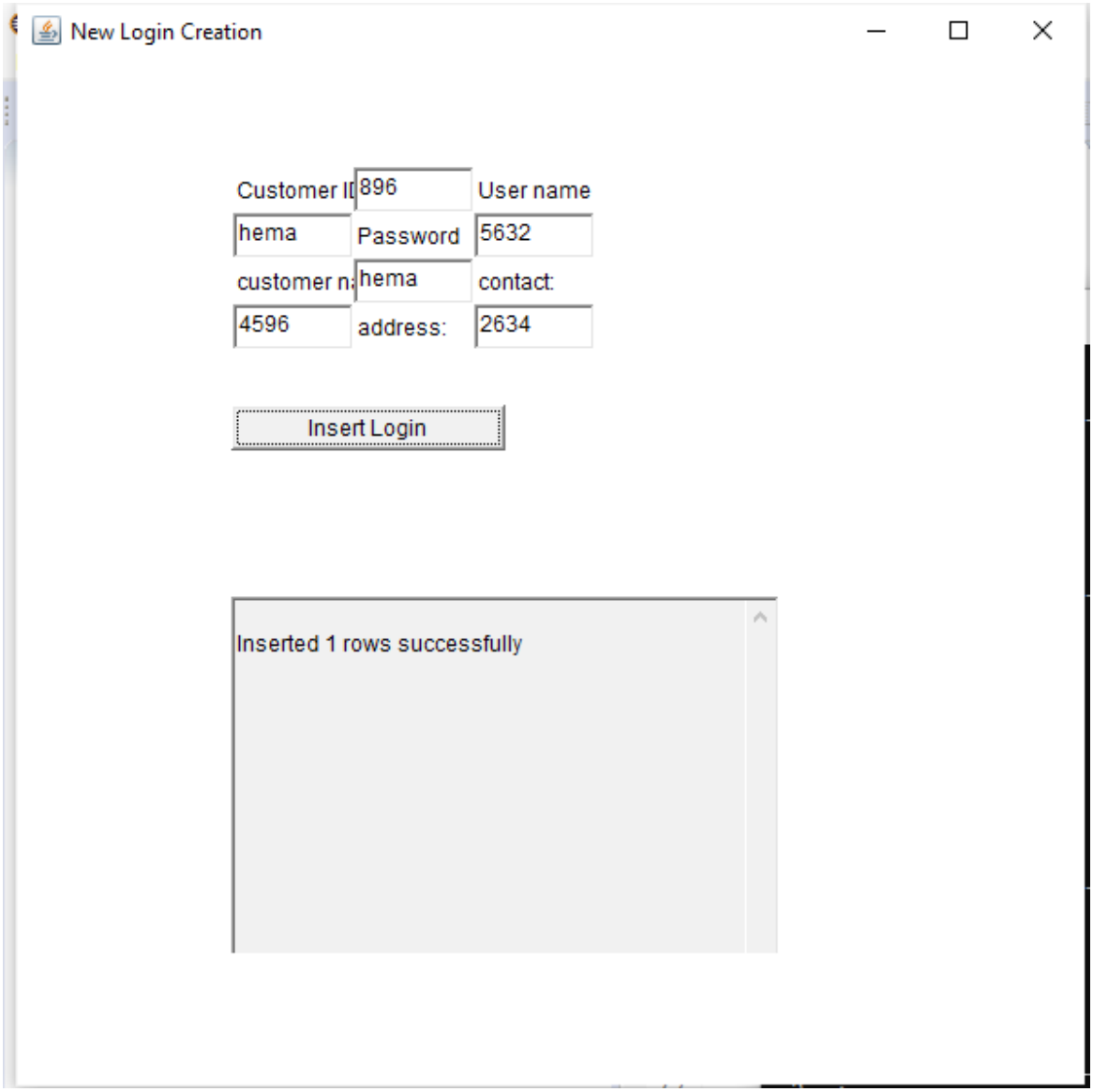

Make Customer_Account
Customer ID:
263113
Account ID:
4567
Customer Account
Since:
896
Inserted 1 rows successfully

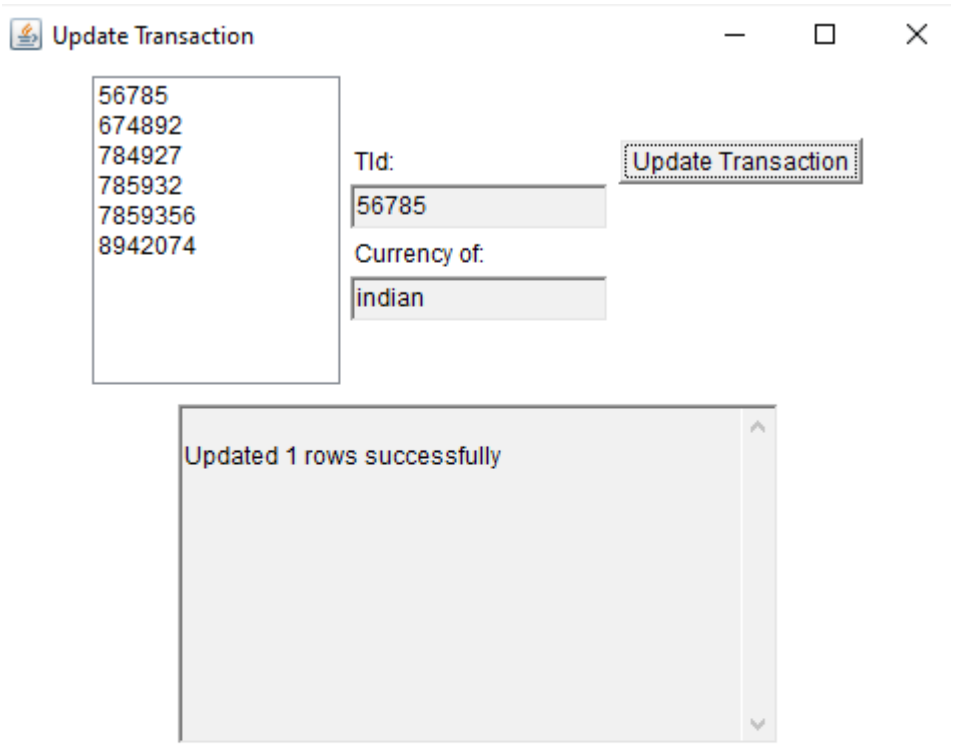

Remove Login
263113
Customer ID:
User name TYpe:
Password:
Customer name:
contact:
address:
Deleted 1 rows successfully
Delete Login

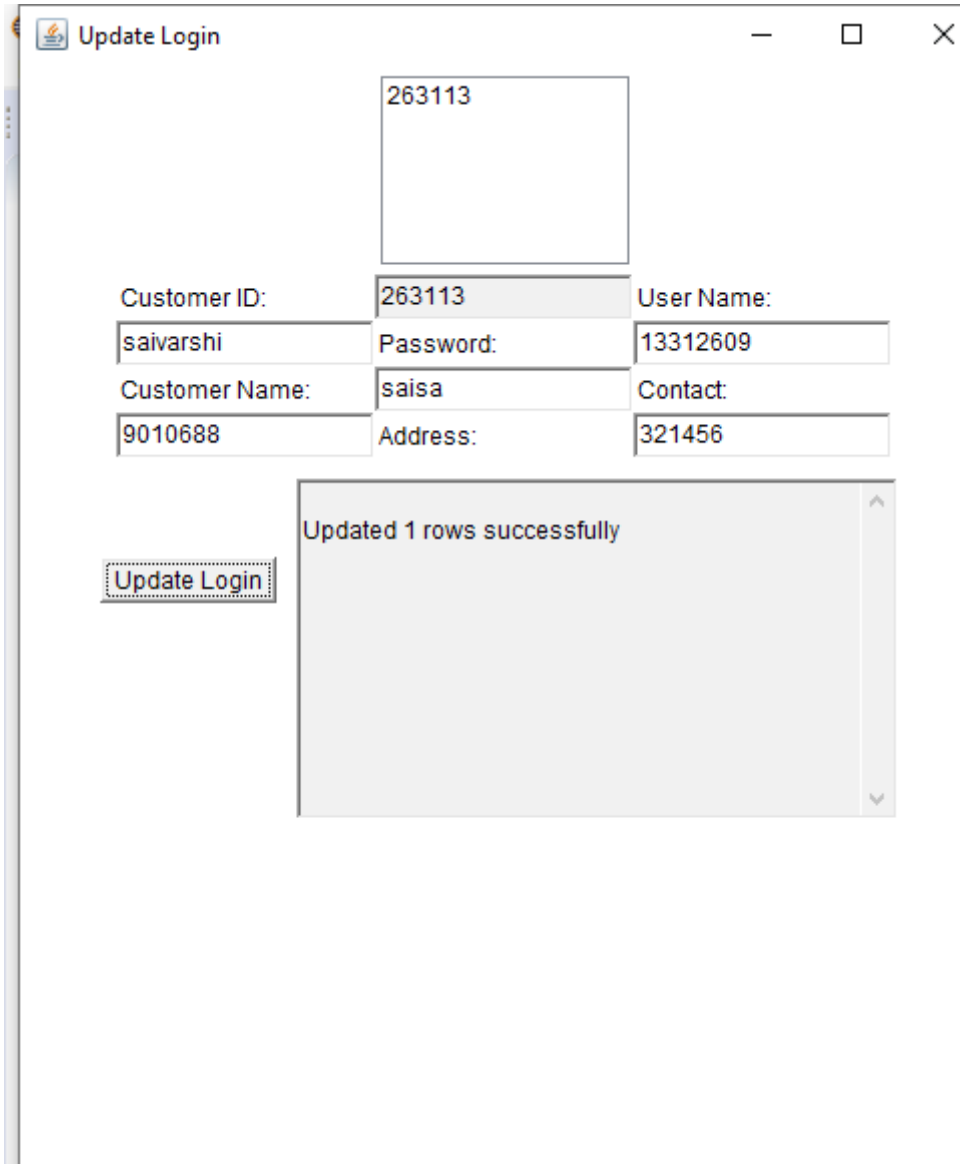
DBMS Assignment 2
Title: Forex Trading Database System

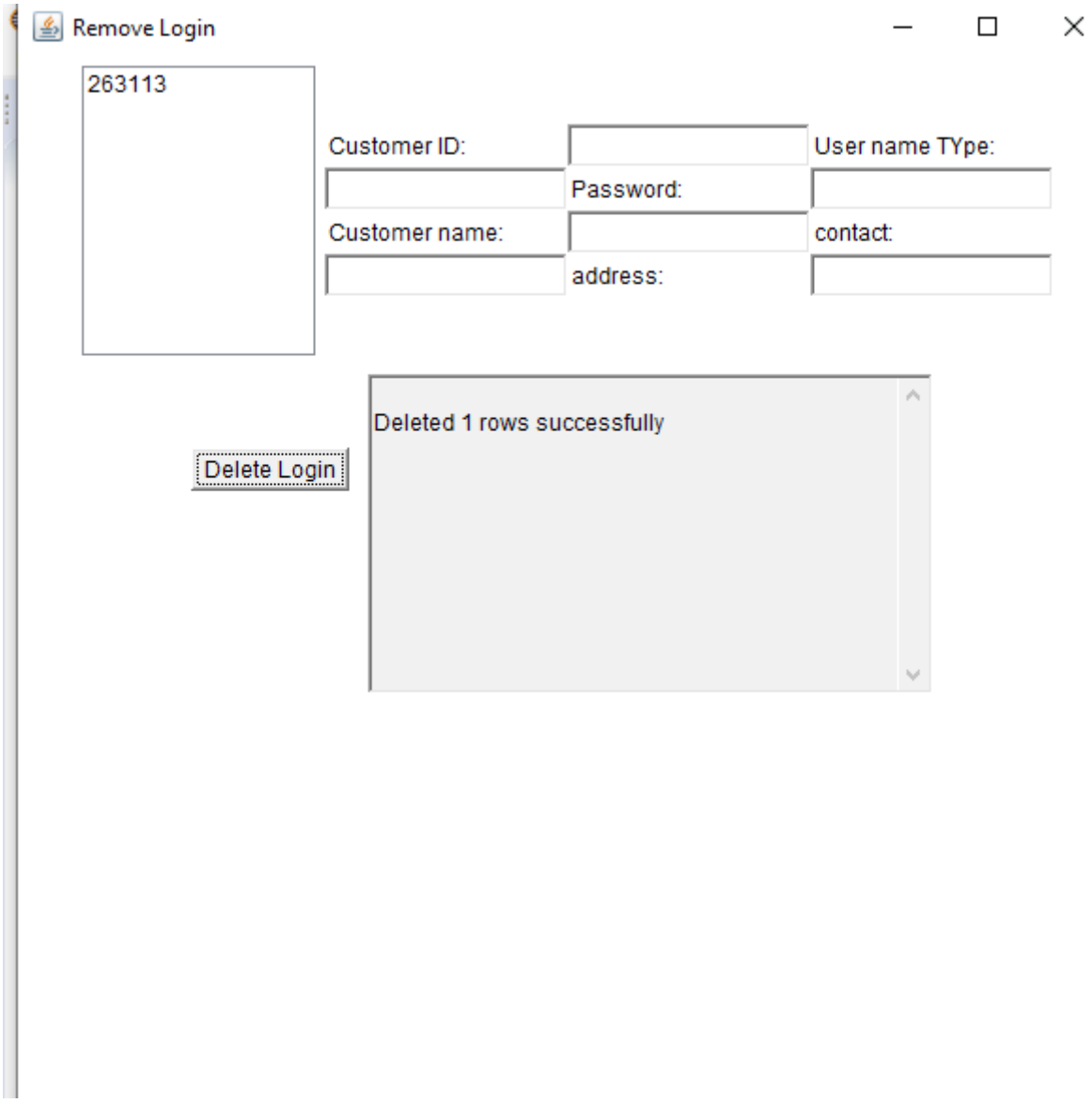
K.Hema
1602-18737-071

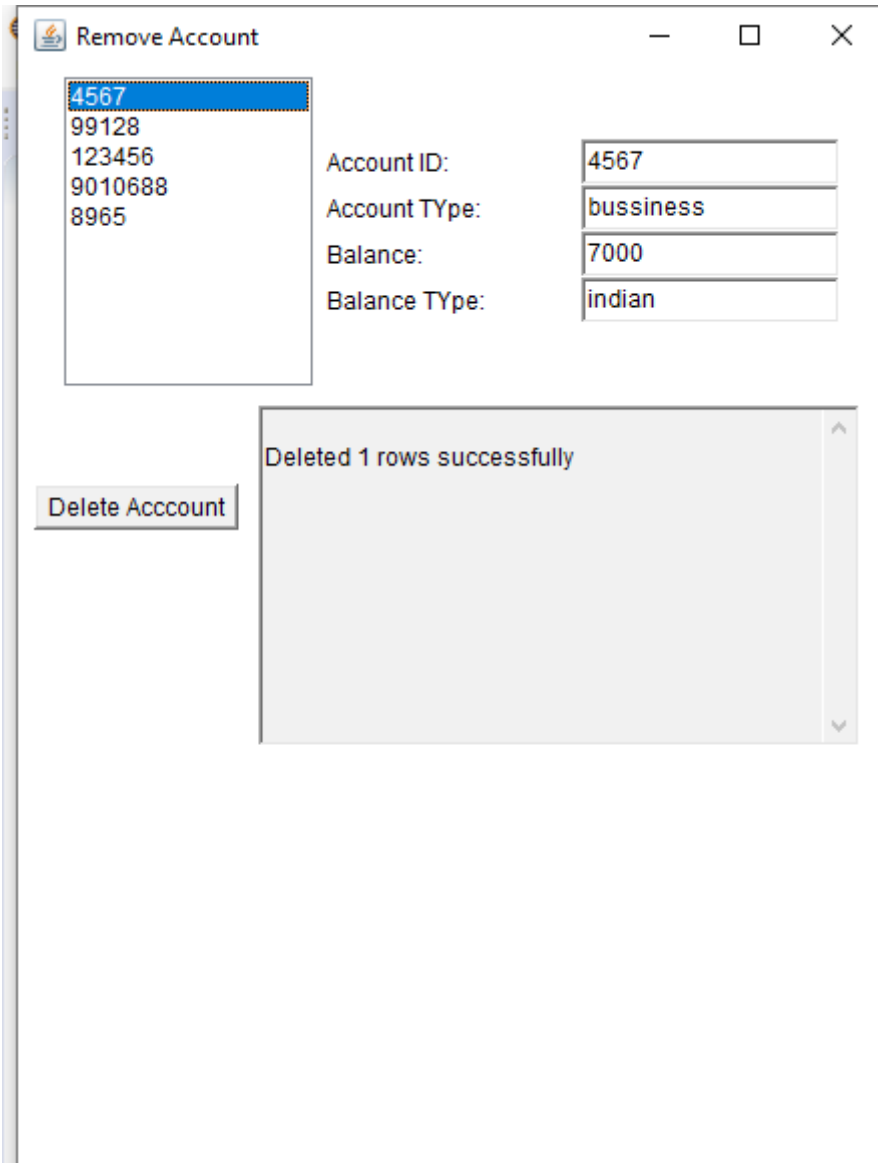

New Account Creation
Close
Account ID:
9865
Account type
student
Balance
1000
Balance Type
rupee
Insert Account
Inserted 1 rows successfully


New Login Creation
Customer ID
896
User name
hema
Password
5632
customer n
hema
contact:
4596
address:
2634
Insert Login
Inserted 1 rows successfully


Update Transaction
56785
674892
784927
785932
7859356
8942074
Tld:
56785
Currency of:
indian
Update Transaction
Updated 1 rows successfully


Update Login
263113
Customer ID:
263113
User Name:
saivarshi
Password:
13312609
Customer Name:
saisa
Contact:
9010688
Address:
321456
Update Login
Updated 1 rows successfully


Remove Login
263113
Customer ID:
User name TYpe:
Password:
Customer name:
contact:
address:
Deleted 1 rows successfully
Delete Login

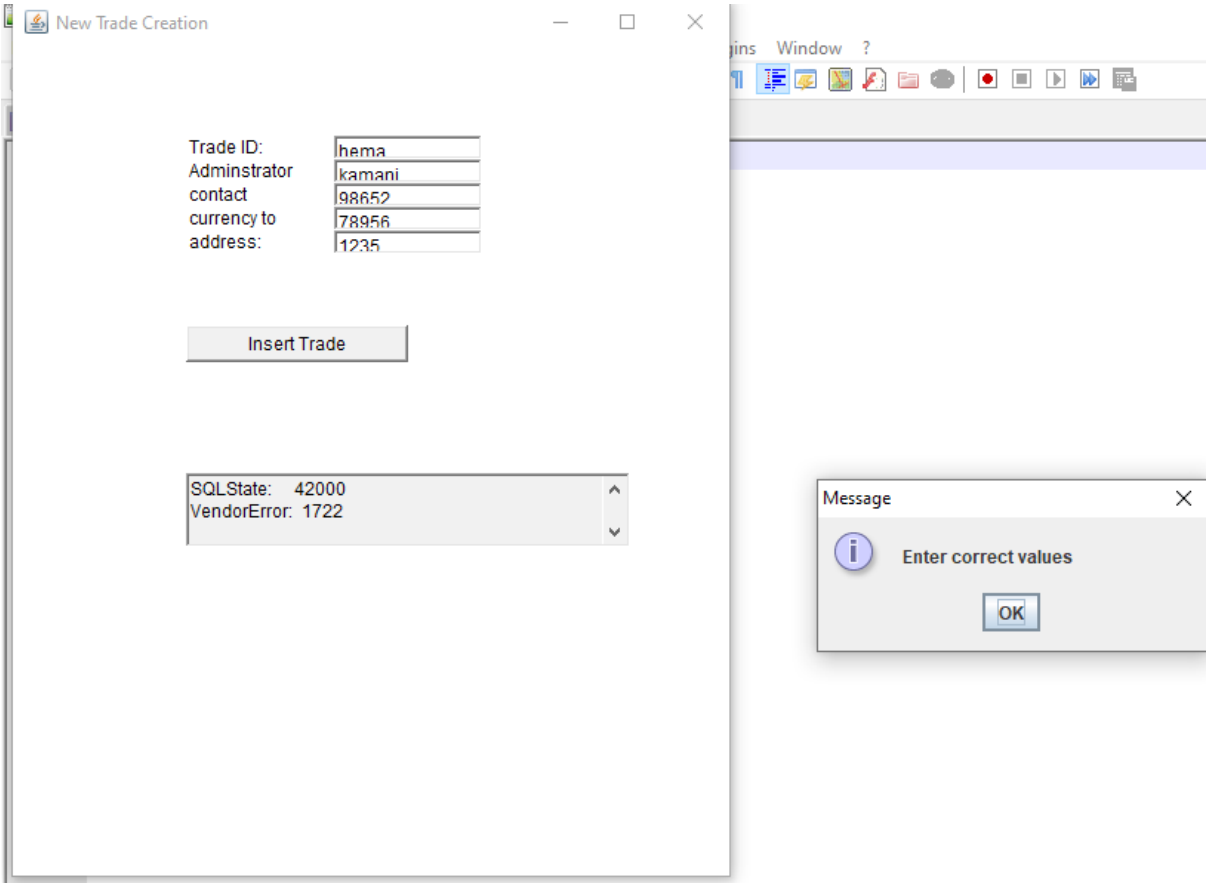

Remove Account
4567
99128
123456
9010688
8965
Account ID:
4567
Account TYpe:
bussiness
Balance:
7000
Balance TYpe:
indian
Deleted 1 rows successfully
Delete Acccount

F) Testing

The code written for building GUI and connecting with database ensures that the values entered by the user are of correct data types. It prompts an error message if the values entered are not of the specified data types.

Example

In this example the domain of the marks is number , whereas the user entered characters. So it prompted ann error message



RESULT

.1. Connection with database is established

2. The values given for tables in the GUI components by the user are saved in the database.

REFERENCES

https://docs.oracle.com/javase/7/docs/api/

https://www.geeksforgeeks.org/establishing-jdbc-connection-in-java/