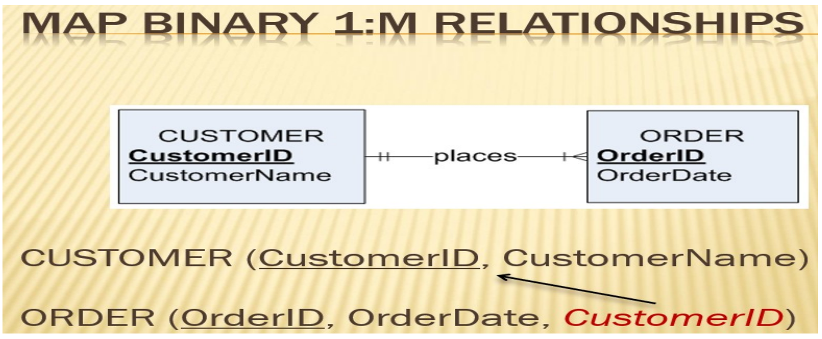
A simple demonstration application program

When a database contains multiple tables, the application program may need a more complicated logic to deal with the relevant database access requests – retrieval data and write data into the database persistently. These tables normally have foreign keys linked together.

Consider a simple case, customers place orders when they go to online shopping. We assume only two tables in the database – ***customers* and *orders***. One customer can place zero or more orders. This is a common scenario that you might be familiar when you studied the database course, which is a typical one to many relationship. First, we build a database named *Purchase*, with table names *Customers*, *Orders* under MySQL. Then we write a java application program to allow the user adding new customer, adding order to new or existing customer, search customer & associated orders they placed, display all customer and orders, as well as some statistics (depending on the requirement or definition).

(1) Database design and implementation (under MySQL)

Note: when mapping 1:m relationship, at the many side, you need to put *a foreign key.*


MAP BINARY 1:M RELATIONSHIPS
CUSTOMER
CustomerID
CustomerName
places
ORDER
OrderID
OrderDate
CUSTOMER (CustomerID, CustomerName)
ORDER (OrderID, OrderDate, CustomerID)

So tables in the database (MySQL) has tables:

Customer(<u>CustomerID</u>, CustomerName)

Order(<u>OrderID</u>, OrderDate, *CustomerID*)

Foreign key (CustomerID) references Customer

Sample data of tables

<u>Customers</u>

<u>CustomerID</u>	FirstName	LastName	Phone
1	Tony	Blair	0402839123
2	Michael	Li	0401777222
3	John	Smith	0445155666

<u>Orders</u>

<u>OrderID</u>	Price ($)	Shipper	*CustomerID*
A001	100	AustPost	1
A002	200	AustPost	1
A003	150	Toll	2
A004	160	Toll	2
A005	190	FedEX	3

Using MySQL Client command line to implement the database(sql script)

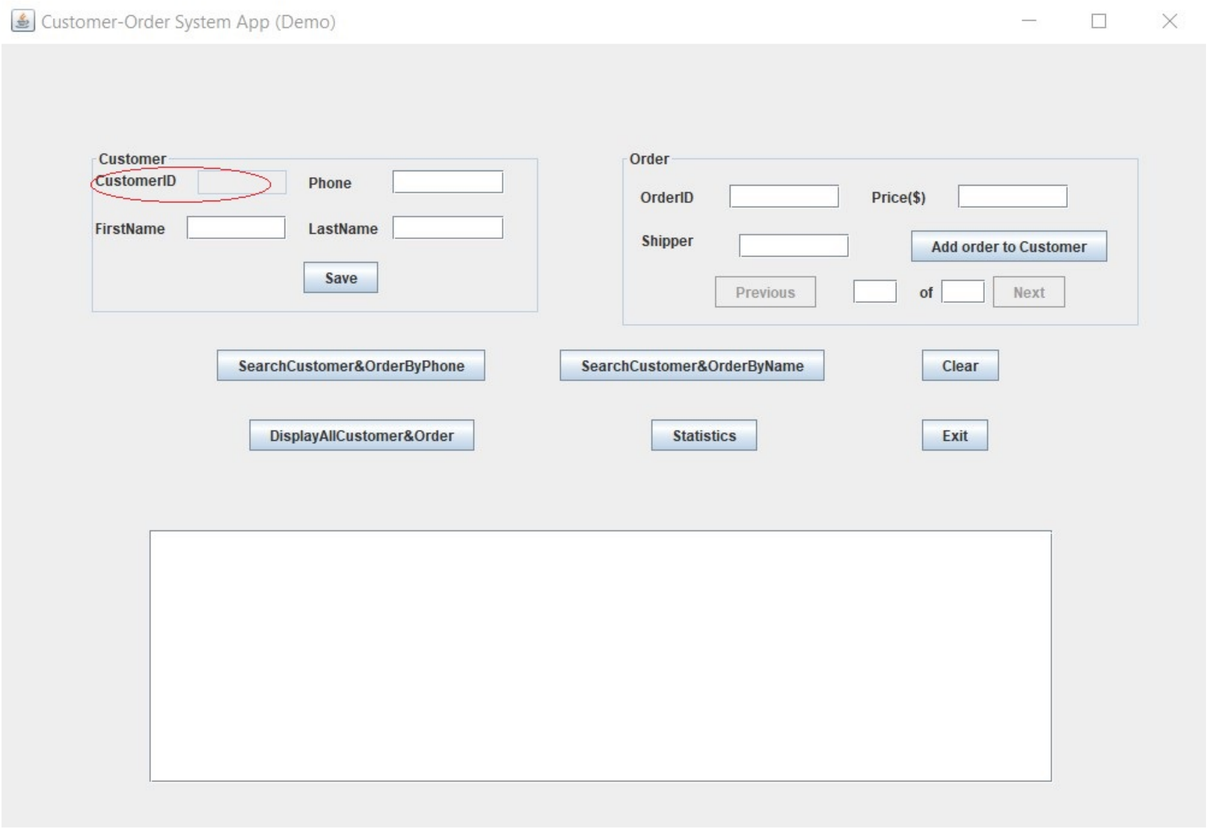
```
mysql>Create database Purchase;
mysql>Use Purchase;

mysql>CREATE TABLE CUSTOMERS
(
        CUSTOMERID INT NOT NULL AUTO_INCREMENT,
        FIRSTNAME VARCHAR (20) NOT NULL,
        LASTNAME VARCHAR (20) NOT NULL,
        PHONE VARCHAR (20) NOT NULL,
        PRIMARY KEY (CUSTOMERID)
);
mysql>INSERT INTO CUSTOMERS (FIRSTNAME,LASTNAME,PHONE) VALUES
        ('Tony','Blair','0402839123'),
        ('Michael','Li','0401777222'),
        ('John','Smith','0445155666');

mysql>CREATE TABLE ORDERS
(
        ORDERID VARCHAR(10) NOT NULL,
        PRICE INT NOT NULL,
        SHIPPER VARCHAR(20) NOT NULL,
        CUSTOMERID INT NOT NULL,
        PRIMARY KEY (ORDERID),
        FOREIGN KEY (CUSTOMERID) REFERENCES CUSTOMERS(CUSTOMERID)
);

mysql>INSERT INTO ORDERS (ORDERID,PRICE,SHIPPER,CUSTOMERID) VALUES
        ('A001',100,'AustPost',1),
        ('A002',200,'AustPost',1),
        ('A003',150,'Toll',2),
        ('A004',160,'Toll',2),
        ('A005',190,'FedEx',3);
```

(2) GUI of this application program



(3) Functionality descriptions

(i) To add a new customer, entering data in Customer panel, then click 'save'.
The entered data should be stored in the Customers table.

(ii) To add an order to Customer, for the new customer at the left panel, entering the corresponding data about the order (that customer placed) and then click the button 'Add order to customer'. For existing customer, search a customer first and then entering the order details and finally add it.

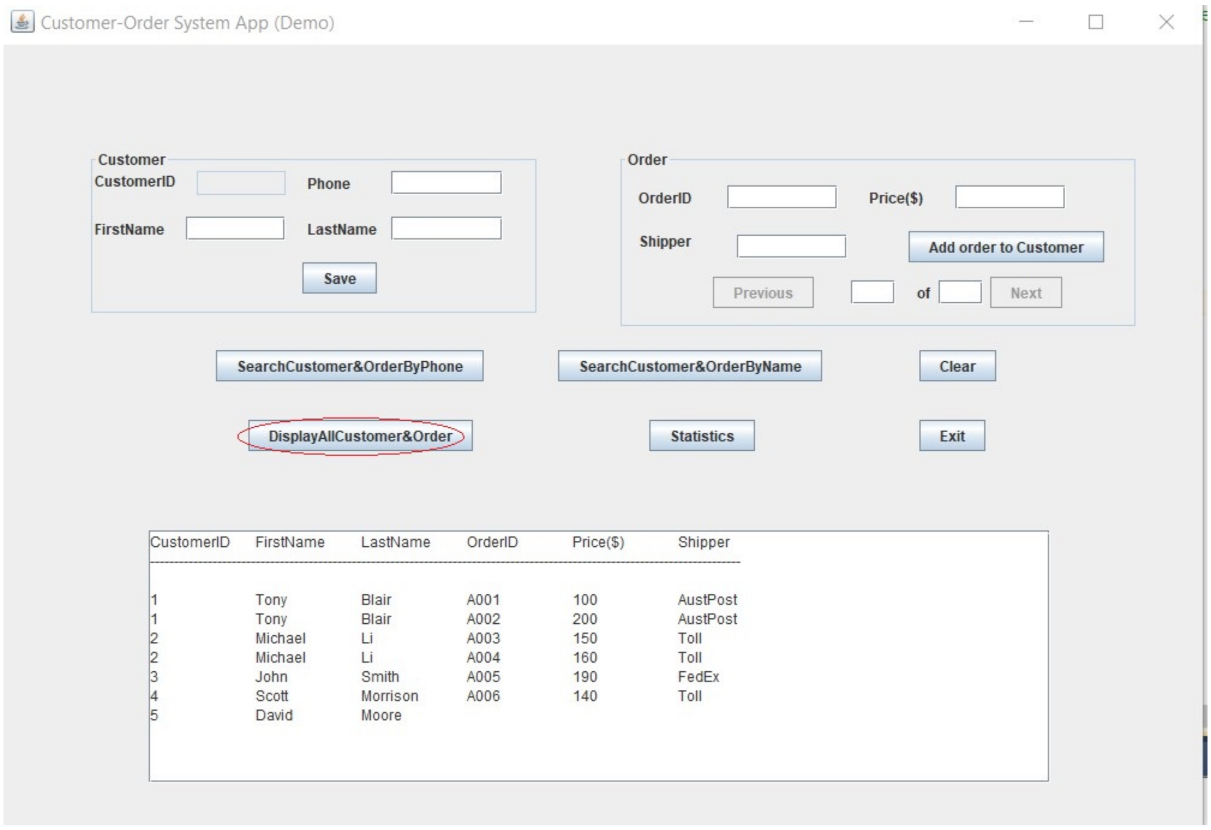
(iii) To search a customer and display the related orders who placed, enter the first and last name, or alternatively enter the phone then click the search button. If the customer has not placed an order, a message is displayed on the text area. If a customer placed multiple orders, they will be displayed at the right panel with the browsing functionalities -previous and next. If a customer is not existed, the error message will be displayed on the text area.

(iv) click 'DisplayAllCustomer&Order', all customers and their orders are displayed.

(v) click 'statistics' button in this case, it will display the customer and the numbers of order placed. (need to use SQL statement with group by)

Note: In my implementation, I used *left join* for joining two tables. The purpose is to display all customer and associated orders including those customers who have not placed the order yet. In SQL language, there are a number ways to join two tables.

(4)Tests and code analysis



```
CustomerID  FirstName  LastName  OrderID  Price($)  Shipper
------------------------------------------------------------

1           Tony       Blair     A001     100       AustPost
1           Tony       Blair     A002     200       AustPost
2           Michael    Li        A003     150       Toll
2           Michael    Li        A004     160       Toll
3           John       Smith     A005     190       FedEx
4           Scott      Morrison  A006     140       Toll
5           David      Moore
```

<u>Tests:</u>

(i)Click 'DisplayAllCustomer&Order' button

(ii)Click 'Statistics' button

(iii)Adding a new customer (without adding an order): David Moore, 0412333666

(iv)Adding a new customer (with adding an order): David Li 0412310888
A008 155 Toll

(v)input phone in the textField: 0402839123 and click 'SearchCustomer&OrderByPhone' Button.

(vi) input firstName, lastName textField: Michael Li and click 'SearchCustomer&OrderByName' button

(vii) input firstName, lastName textField: Michael Li and click 'SearchCustomer&OrderByName' button, and add a new order: A009, 175 AustPost

(viii) input firstName, lastName textField: Steven Johnson and click 'SearchCustomer&OrderByName' button

(ix) others

<u>Code analysis</u> (see the java source code)