



# 江蘇大學

## 计算机科学与通信工程学院

课 程	数据库系统概论课程设计
课 设 题 目	设计并实现一个点菜管理信息系统
学 生 姓 名	刘耀东
学 号	3140608037
专 业 班 级	软件工程 1402
指 导 教 师	邢玉萍
日 期	2016 年 7 月 5 日

一：需求分析

1. 信息要求

顾客的姓名、地址、电话等基本信息；菜的名字、单价；订单的订菜项数；订单细则的订菜数量、总价格；应收账款的应收金额等。

2. 处理要求

用户想要点哪些菜，可以一次点多种菜，也可以一次点同一种菜的多份，点完后提交订单；用户需要查看订单信息，在订单中可以看到点的菜的菜名、单价、购买数量、总价格等详细信息，并且可以选择某个订单付款，也可以把所有的订单一次性全部付款；当然也可以选中不想要的订单，点击“取消订单取消”；当选择付款后，提示订单的总价格，并可以输入实付款金额，及时算出找零金额。

3. 完整性要求

保证实体完整性、参照完整性、用户定义的完整性，从而保证系统中数据的正确性、相容性和有效性。

4. 数据字典

数据项

编号	数据项名	数据类型	长度
1	顾客号	CHAR	20
2	顾客名	CHAR	20
3	地址	CHAR	40
4	电话	CHAR	20
5	菜品号	CHAR	20
6	菜品名	CHAR	40
7	单价	FLOAT(1)	
8	订单号	CHAR	20
9	订菜项数	SMALLINT	
10	细则号	CHAR	20
11	订货数	SMALLINT	
12	价格	FLOAT(1)	
13	应收金额	FLOAT(1)	

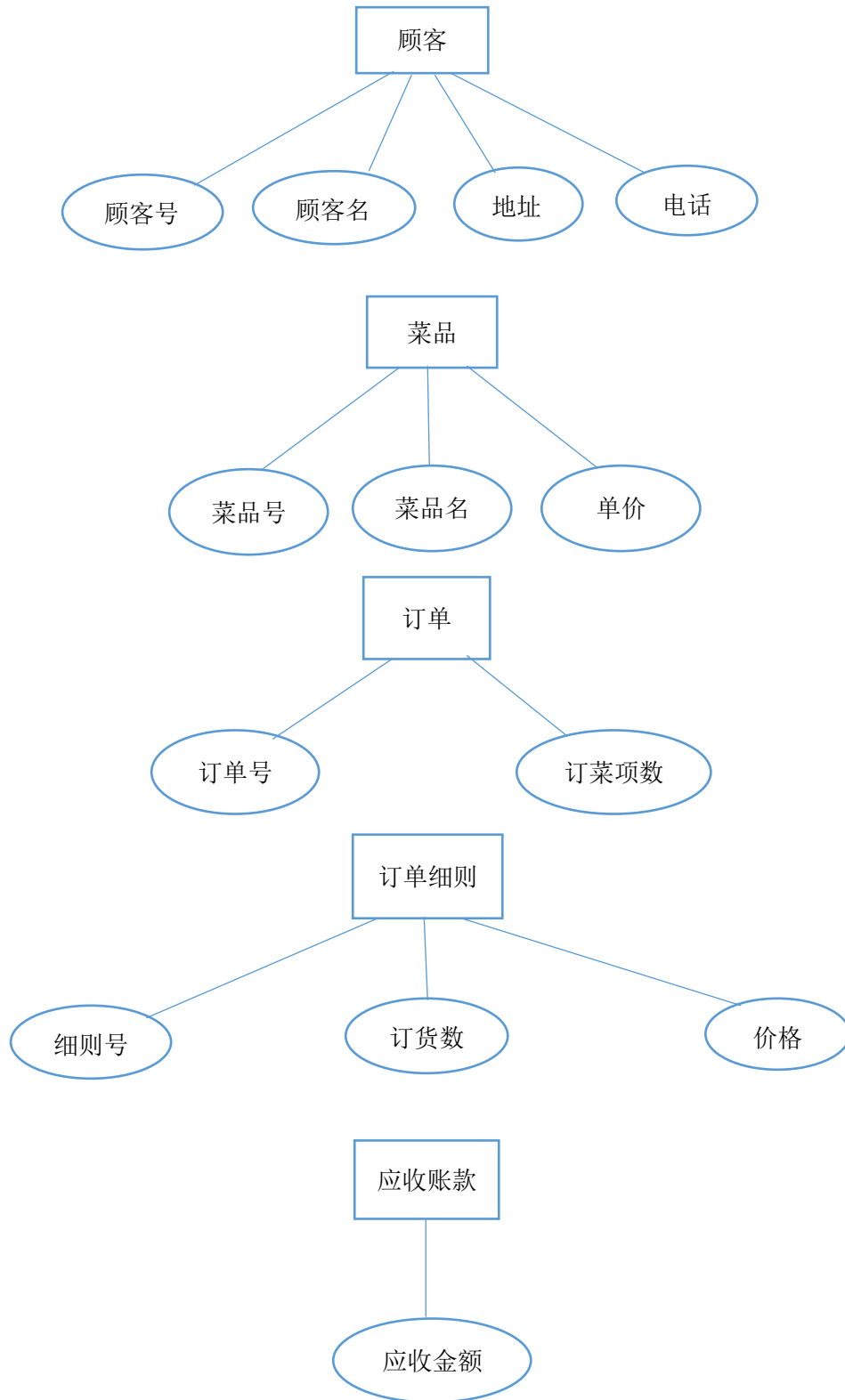
数据结构

数据结构名	组成
顾客	顾客号，顾客名，地址，电话
菜品	菜品号，菜品名，单价
订单	订单号，订菜项数
订单细则	细则号，订货数，价格
应收账款	应收金额

## 二：概念结构设计

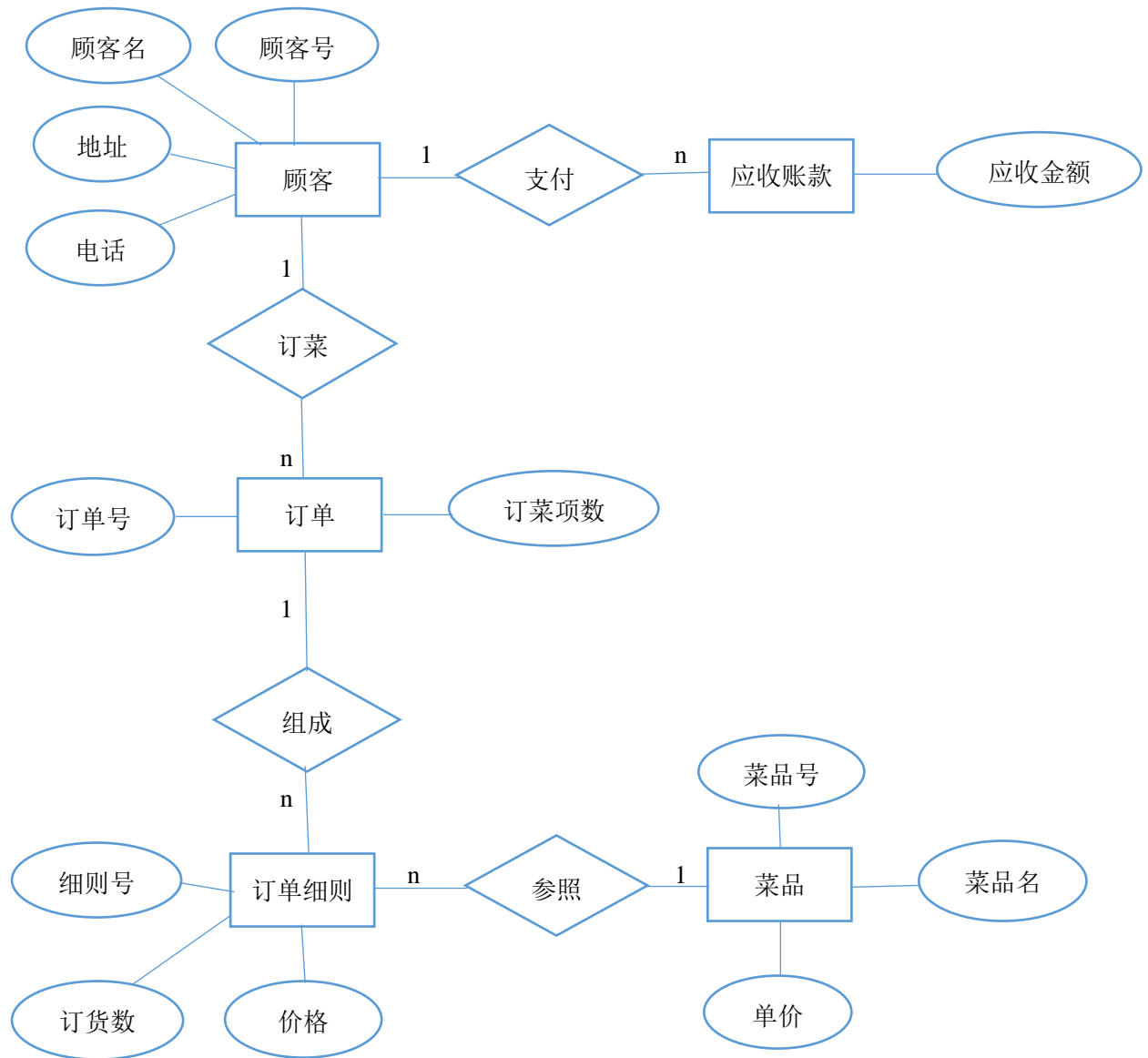
### E-R 模型

#### 1. 各个实体的 E-R 图



## 2. E-R 图的集成

根据上面各部分的实体图及其之间的相互联系，可以得到如下 E-R 图：



## 三：逻辑结构设计

### 1. 逻辑结构设计的任务

逻辑结构设计的任务就是把概念结构设计阶段设计好的基本 E-R 图转换为与选用数据库管理系统产品（本次使用 SQL Sever 数据库管理系统）所支持的数据模型相符合的逻辑结构。在这里也就是要将 E-R 图转换成关系数据模型，实际上就是要将实体型、实体的属性和实体型之间的联系转换为关系模式。

### 2. 转换

根据转换的一般原则可将上面的集成 E-R 图转换为下面的关系模式（关系的码用下划线标出，并转为了粗体）：

顾客：{顾客号，顾客名，地址，电话}

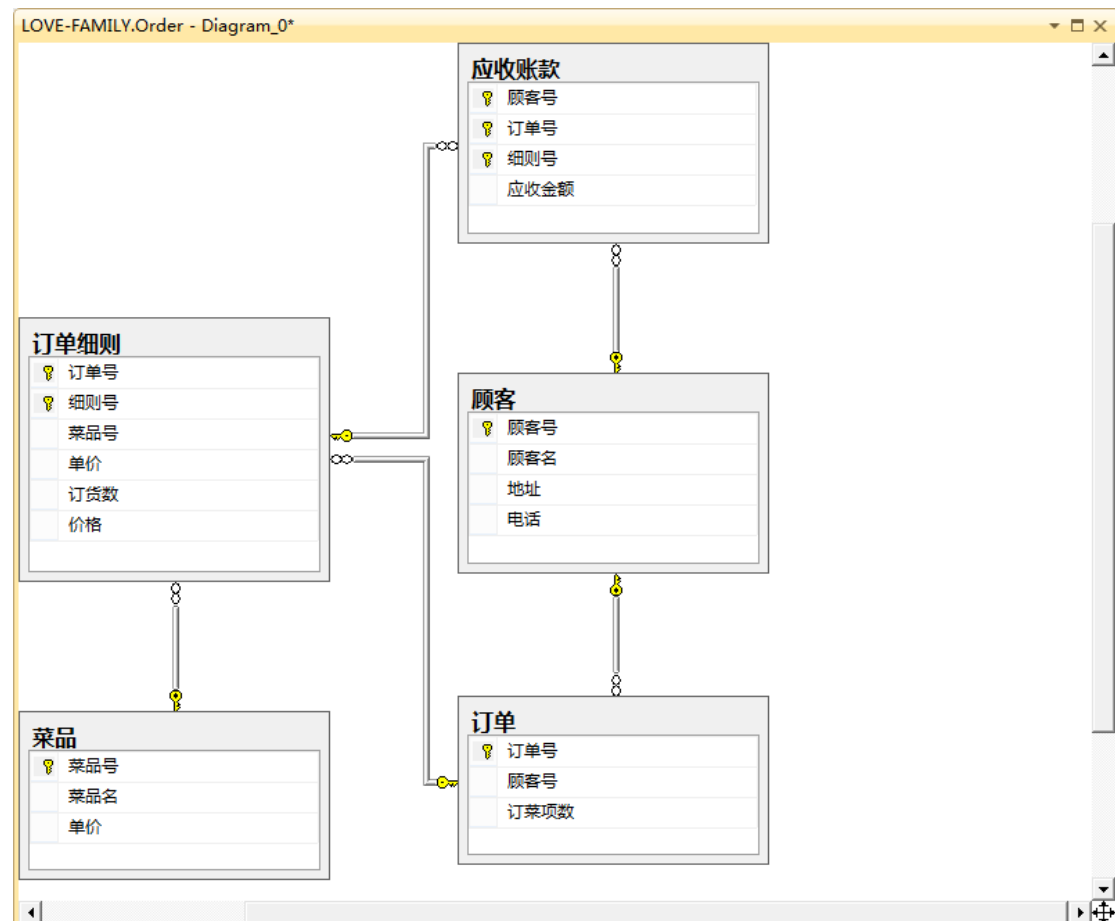
菜品：{菜品号，菜品名，单价}

订单：{订单号，顾客号，订菜项数}

订单细则：{订单号，细则号，菜品号，单价，订货数，价格}

应收账款：{顾客号，订单号，细则号，应收金额}

### 3. 数据库关系图



## 四：物理结构设计

数据库在物理设备上的存储结构与存取方法称为数据库的物理结构，它依赖于选定的数据库管理系统。为一个给定的逻辑数据模型选取一个最适合应用要求的物理结构的过程，就是数据库的物理设计。

数据库的物理设计通常分为两步：

- (1) 确定数据库的物理结构，在关系数据库中主要指存取方法和存储结构。
- (2) 对物理结构进行评价，评价的重点是时间和空间效率。

## 五：数据库的实施和维护

### 1. 顾客

--表的创建

```
CREATE TABLE 顾客
```

```
(
```

```
顾客号 CHAR(20),
```

```
顾客名 CHAR(20),
地址 CHAR(40),
电话 CHAR(20),
PRIMARY KEY(顾客号)
);
--插入数据
INSERT
INTO 顾客
VALUES('1','张三','江苏镇江江苏大学 A6','18852850000');
INSERT
INTO 顾客
VALUES('2','李四','江苏镇江江苏大学 B3','18852851111');
```

## 2. 菜品

--表的创建

```
CREATE TABLE 菜品
(
菜品号 CHAR(20),
菜品名 CHAR(40),
单价 FLOAT(1),
PRIMARY KEY(菜品号)
);
--插入数据
INSERT
INTO 菜品
VALUES('1','鸡腿套餐','21');
INSERT
INTO 菜品
VALUES('2','烤翅套餐','21');
INSERT
INTO 菜品
VALUES('3','手扒鸡套餐','30');
INSERT
INTO 菜品
VALUES('4','鸡肉卷套餐','20');
INSERT
INTO 菜品
VALUES('5','虾堡套餐','23');
INSERT
INTO 菜品
VALUES('6','小吃套餐','22');
```

## 3. 订单

--表的创建

```
CREATE TABLE 订单
(
```

```
订单号 CHAR(20),
顾客号 CHAR(20),
订菜项数 SMALLINT,
PRIMARY KEY(订单号),
FOREIGN KEY(顾客号) REFERENCES 顾客(顾客号)
);
```

#### 4. 订单细则

```
CREATE TABLE 订单细则
(
  订单号 CHAR(20),
  细则号 CHAR(20),
  菜品号 CHAR(20),
  单价 FLOAT(1),
  订货数 SMALLINT,
  价格 FLOAT(1),
  PRIMARY KEY(订单号,细则号),
  FOREIGN KEY(菜品号) REFERENCES 菜品(菜品号),
  FOREIGN KEY(订单号) REFERENCES 订单(订单号)
);
```

#### 5. 应收账款

--表的创建

```
CREATE TABLE 应收账款
(
  顾客号 CHAR(20),
  订单号 CHAR(20),
  细则号 CHAR(20),
  应收金额 FLOAT(1),
  PRIMARY KEY(顾客号,订单号,细则号),
  FOREIGN KEY(顾客号) REFERENCES 顾客(顾客号),
  FOREIGN KEY(订单号,细则号) REFERENCES 订单细则(订单号,细则号)
);
```

#### 6. 创建两个用户

(1) 张三

--创建登陆帐户 (create login)

```
CREATE LOGIN 张三 WITH PASSWORD='liuyaodong'
```

--为登陆账户创建数据库用户 (create user),在 Order 数据库中的 security 中的 user 下可以找到新创建的张三

```
CREATE USER 张三 FOR LOGIN 张三 WITH DEFAULT_SCHEMA=dbo
```

--授权

```
exec sp_addrolemember 'db_owner', '张三'
```

(2) 李四

--创建登陆帐户 (create login)

```
CREATE LOGIN 李四 WITH PASSWORD='liuyaodong'
```

--为登陆账户创建数据库用户（create user）,在 Order 数据库中的 security 中的 user 下可以找到新创建的张三

```
CREATE USER 李四 FOR LOGIN 李四 WITH DEFAULT_SCHEMA=dbo
```

--授权

```
exec sp_addrolemember 'db_owner', '李四'
```

## 六：应用程序的设计

### 1. 连接数据库类

```
package com.点菜;
```

```
import java.sql.*;
```

```
public class Connect {
```

```
    String JDriver = "com.microsoft.sqlserver.jdbc.SQLServerDriver";// SQL 数据库引擎
```

```
    String connectDB = "jdbc:sqlserver://127.0.0.1:1433;DatabaseName=Order";// 数据源
```

```
    static Connection con=null;
```

```
    public void connect(String user,String password){
```

```
        try {
```

```
            Class.forName(JDriver);// 加载数据库引擎，返回给定字符串名的类
```

```
        } catch (ClassNotFoundException e) {
```

```
            System.out.println("加载数据库引擎失败");
```

```
            System.exit(0);
```

```
        }
```

```
        System.out.println("数据库驱动成功");
```

```
        try {
```

```
            con=DriverManager.getConnection(connectDB, user, password);// 连接数据库对象
```

```
            System.out.println("连接数据库成功");
```

```
        } catch (SQLException e) {
```

```
            e.printStackTrace();
```

```
            System.out.println("数据库连接错误");
```

```
            System.exit(0);
```

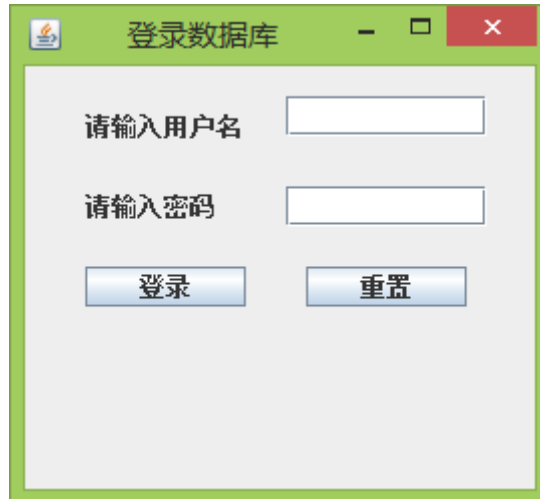
```
        }
```

```
    }
```

```
}
```

### 2. 登录窗口类





package com.点菜;

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.sql.*;

public class Login extends JFrame implements ActionListener{
    private JPanel jp=new JPanel();
    JLabel name=new JLabel("请输入用户名");
    JLabel password=new JLabel("请输入密码");
    private JLabel[] jl={ name,password};
    JButton login=new JButton("登录");
    JButton reset=new JButton("重置");
    private JButton[] jb={login,reset};
    public static JTextField jName=new JTextField();
    private JPasswordField jPassword=new JPasswordField();
    public Login(){
        jp.setLayout(null);
        for(int i=0;i<2;i++){
            jl[i].setBounds(30,20+40*i,180,20);
            jb[i].setBounds(30+110*i,100,80,20);
            jp.add(jl[i]);
            jp.add(jb[i]);
            jb[i].addActionListener(this);
        }
        jName.setBounds(130,15,100,20);
        jp.add(jName);
        jName.addActionListener(this);
        jPassword.setBounds(130,60,100,20);
        jp.add(jPassword);
        jPassword.setEchoChar('*');//设置密码框中的回显字符
```

```

jPassword.addActionListener(this);
this.add(jp);

this.setTitle("登录数据库");
this.setBounds(550,200,270,250);
this.setVisible(true);
this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

}

public void actionPerformed(ActionEvent e){
    if(e.getSource()==jName){//如果事件源为文本框
        jPassword.requestFocus();//切换输入焦点到密码框
    }else if(e.getSource()==jb[1]){//如果事件源为重置按钮
        //清空姓名文本框、密码框中的所有信息
        jName.setText("");
        jPassword.setText("");
        jName.requestFocus();//让输入焦点回到文本框
    }else{//如果事件源为登录按钮，则判断登录名和密码是否正确
        //判断用户名和密码是否匹配
        UserInformation u=new UserInformation();

        if(jName.getText().equals(u.user1)&&String.valueOf(jPassword.getPassword()).equals(u.password1)||jName.getText().equals(u.user2)&&String.valueOf(jPassword.getPassword()).equals(u.password2)){

            Connect con=new Connect();
            con.connect(jName.getText(), String.valueOf(jPassword.getPassword()));
            System.out.println("登录成功，欢迎您的到来！");
            this.dispose();
            new Function();

        }else{
            JOptionPane.showMessageDialog(this,"对不起，您的用户名或密码错误！","
            错误提示框",JOptionPane.INFORMATION_MESSAGE);
        }
    }
}
}

```

### 3. 功能选择窗口类



```
package com.点菜;

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.sql.*;

public class Function extends JFrame implements ActionListener{
    JPanel jp=new JPanel();
    JButton[] jb={new JButton("点菜"),new JButton("订单"),new JButton("退出")};
    JLabel jl=new JLabel("欢迎光临");

    public Function(){
        jp.setLayout(new FlowLayout());
        jl.setHorizontalAlignment(JLabel.CENTER);
        for(int i=0;i<jb.length;i++){
            jp.add(jb[i]);
            jb[i].addActionListener(this);
        }
        this.add(jl,BorderLayout.NORTH);
        this.add(jp, BorderLayout.CENTER);


        this.setTitle("WELCOME");
        this.setVisible(true);
//        this.pack();
        this.setBounds(500,300,400,110);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
    public void actionPerformed(ActionEvent e){
        if(e.getSource()==jb[0]){//点 “点菜” 按钮
            this.dispose();
            new Order();
        }else if(e.getSource()==jb[1]){//点 “订单” 按钮
            this.dispose();
            new Indent();
        }else if(e.getSource()==jb[2]){//点 “退出” 按钮
            // 关闭连接
            try {
//                Connect. rs.close();
            } catch (SQLException e1) {
                e1.printStackTrace();
            }
        }
    }
}
```

```

//          Connect.stmt.close();// 关闭命令对象连接
          Connect.con.close();// 关闭数据库连接
        } catch (SQLException e1) {
            e1.printStackTrace();
        }
        System.exit(0);
    }
}
}

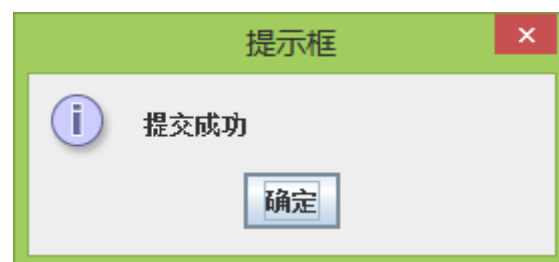
```

#### 4. 点菜窗口类



菜名	价格	数量
<input type="checkbox"/> 鸡腿套餐	21.0元	0
<input type="checkbox"/> 烤翅套餐	21.0元	0
<input type="checkbox"/> 手扒鸡套餐	30.0元	0
<input type="checkbox"/> 鸡肉卷套餐	20.0元	0
<input type="checkbox"/> 虾堡套餐	23.0元	0
<input type="checkbox"/> 小吃套餐	22.0元	0

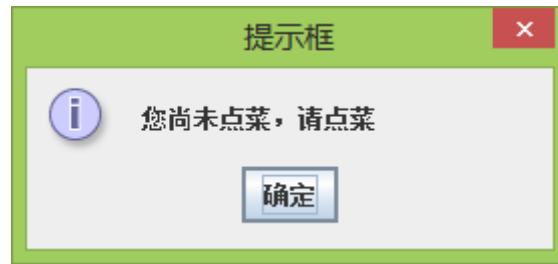
提交订单    返回



提示框

*i* 提交成功

确定



package com.点菜;

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.sql.*;

public class Order extends JFrame implements ActionListener{
    static int ddh=1;//订单号
    JPanel jp0=new JPanel();
    JPanel jp1=new JPanel();
    JCheckBox jcb1=new JCheckBox();
    JCheckBox jcb2=new JCheckBox();
    JCheckBox jcb3=new JCheckBox();
    JCheckBox jcb4=new JCheckBox();
    JCheckBox jcb5=new JCheckBox();
    JCheckBox jcb6=new JCheckBox();
    private JCheckBox jcb[]={jcb1,jcb2,jcb3,jcb4,jcb5,jcb6};
    JButton jb[]=new JButton[]{new JButton("提交订单"),new JButton("返回")};
    Statement stmt;
    ResultSet rs;
    Box vbox = Box.createVerticalBox();//创建纵向 Box 容器
    // SpinnerNumberModel          snm[]={new SpinnerNumberModel(0,0,30,1),new
    SpinnerNumberModel(0,0,30,1),
    //          new SpinnerNumberModel(0,0,30,1),new SpinnerNumberModel(0,0,30,1),
    //          new SpinnerNumberModel(0,0,30,1),new SpinnerNumberModel(0,0,30,1)
    // };
    JTextField jtf[]={new JTextField("0"),new JTextField("0"),new JTextField("0"),new
    JTextField("0"),
        new JTextField("0"),new JTextField("0")
    };

    public Order(){
        //      jp0.setLayout(new FlowLayout());
        //      jp1.setLayout(new FlowLayout());
        try {

            stmt=Connect.con.createStatement();
```

```

String sql="SELECT 订单号 FROM 订单 ORDER BY 订单号 DESC";
rs=stmt.executeQuery(sql);
//此处耗费了大量时间纠错
// while(rs.next()){
//     if(!rs.next()){
//         ddh=1;
//         System.out.println("6666666666");
//     }else{
//         String a=rs.getString(1);//此时 a 为“某数”
//         System.out.println(a+"3");
//         char c[]=a.toCharArray();
//         System.out.print(c[0]);
//         int b=c[0]-'0';//将字符型转为整数型
//         int b=Integer.valueOf(a);
//         ddh=b+1;
//     }

rs = stmt.executeQuery("SELECT * FROM 菜品");
for(int i=0;rs.next();i++){
    jcb[i].setText(rs.getString(2)+rs.getString(3)+"元");
//    jcb[i].setBounds(10,20+30*i, 100,30);
    vbox.add(Box.createVerticalStrut(20));
    vbox.add(jcb[i]);
    vbox.add(jtf[i]);
}
} catch (SQLException e) {
    e.printStackTrace();
}
jp0.add(vbox);
for(int i=0;i<jb.length;i++){
    jb[i].addActionListener(this);
    jp1.add(jb[i]);
}
for(int i=0;i<jtf.length;i++){
    jcb[i].addActionListener(this);
//    jtf[i].addActionListener(this);
}
this.add(jp0, BorderLayout.NORTH);
this.add(jp1, BorderLayout.CENTER);

this.setTitle("点菜");
this.setBounds(500,100,300,500);
this.setVisible(true);
this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

```

```
}
```

```
public void actionPerformed(ActionEvent e){
```

```
    int xzh=0;//细则号
```

```
    int dcxs=0;//订菜项数
```

```
    if(e.getSource()==jcb[0]){//复选框
```

```
        if(jcb[0].isSelected()){
            jtf[0].setText("1");
```

```
        }else{
            jtf[0].setText("0");
```

```
        }
```

```
    }else if(e.getSource()==jcb[1]){//复选框
```

```
        if(jcb[1].isSelected()){
            jtf[1].setText("1");
```

```
        }else{
            jtf[1].setText("0");
```

```
        }
```

```
    }else if(e.getSource()==jcb[2]){//复选框
```

```
        if(jcb[2].isSelected()){
            jtf[2].setText("1");
```

```
        }else{
            jtf[2].setText("0");
```

```
        }
```

```
    }else if(e.getSource()==jcb[3]){//复选框
```

```
        if(jcb[3].isSelected()){
            jtf[3].setText("1");
```

```
        }else{
            jtf[3].setText("0");
```

```
        }
```

```
    }else if(e.getSource()==jcb[4]){//复选框
```

```
        if(jcb[4].isSelected()){
            jtf[4].setText("1");
```

```
        }else{
            jtf[4].setText("0");
```

```
        }
```

```
    }else if(e.getSource()==jcb[5]){//复选框
```

```
        if(jcb[5].isSelected()){
            jtf[5].setText("1");
```

```
        }else{
            jtf[5].setText("0");
```

```
        }
```

```
    }else if(e.getSource()==jb[0]){//提交订单
```

```
    //    ddh++;
```

```

        for(int i=0;i<jcb.length;i++){
            if(jcb[i].isSelected()){
                dcxs++;
            }
        }
        for(int i=0;i<jcb.length;i++){
            if(Integer.parseInt(jtf[i].getText())>0){

                try {
//                    String ddhs=((Integer)ddh).toString();
//                    String xzhs=((Integer)xzh).toString();
                    PreparedStatement ps=Connect.con.prepareStatement("SELECT 顾
客号 FROM 顾客 WHERE 顾客名=?");//处理 SQL 命令，获取预编译语句对象
                    String u=Login.jName.getText();
                    ps.setString(1, u);//准备参数，插入数据
                    rs=ps.executeQuery();//执行 SQL 语句
//                    System.out.println(u);
//                    rs=stmt.executeQuery("SELECT 顾客号 FROM 顾客 WHERE
顾客名=u");

                    String gkh=null;//顾客号
                    while(rs.next()){
                        gkh=rs.getString(1);//顾客号
                    }
//                    System.out.println(gkh);
//                    stmt.executeUpdate("INSERT INTO 订 单
VALUES(ddh,gkh,dcxs)");

                    ps=Connect.con.prepareStatement("INSERT INTO 订 单
VALUES(?,?,?)");//处理 SQL 命令，获取预编译语句对象
                    ps.setInt(1, ddh);//准备参数，插入数据
                    ps.setString(2, gkh);
                    ps.setInt(3, dcxs);
                    ps.executeUpdate();//执行 SQL 语句
                    break;
                }catch (SQLException e1) {
                    e1.printStackTrace();
                }
            }
        }
        for(int i=0;i<jcb.length;i++){
            if(Integer.parseInt(jtf[i].getText())>0){
                xzh++;
                try{
//                    String cph=((Integer)(i+1)).toString();//菜品号
//                    rs = stmt.executeQuery("SELECT 单价 FROM 菜品 WHERE

```



```

菜品号=i+1");

        PreparedStatement ps=Connect.con.prepareStatement("SELECT 单
价 FROM 菜品 WHERE 菜品号=?");//处理 SQL 命令，获取预编译语句对象
        int cph=i+1;//菜品号
        ps.setInt(1, cph);//准备参数，插入数据
        rs=ps.executeQuery();//执行 SQL 语句
        float dj=0;
        while(rs.next()){
            dj=rs.getFloat(1);//单价
        }
        int dhs=Integer.parseInt(jtf[i].getText());//订货数
        float jg=dj*dhs;//价格
//        stmt.executeUpdate("INSERT INTO 订 单 细 则
VALUES(ddh,xzh,i+1,rs.getFloat(1),dhs,Integer.parseInt(rs.getString(1))*dhs)");
        ps=Connect.con.prepareStatement("INSERT INTO 订 单 细 则
VALUES(?,?,?,*,?)");//处理 SQL 命令，获取预编译语句对象
        ps.setInt(1, ddh);//准备参数，插入数据
        ps.setInt(2, xzh);
        ps.setInt(3, cph);
        ps.setFloat(4, dj);
        ps.setInt(5, dhs);
        ps.setFloat(6, jg);
        ps.executeUpdate();//执行 SQL 语句
        //提交成功框
//        JOptionPane.showMessageDialog(this," 提 交 成 功 "," 提 示 框
",JOptionPane.INFORMATION_MESSAGE);

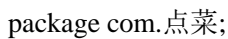
        } catch (SQLException e1) {
            e1.printStackTrace();
        }
    }

//        JOptionPane.showMessageDialog(this," 提 交 成 功 "," 提 示 框
",JOptionPane.INFORMATION_MESSAGE);
    }

    if(jcb[0].isSelected()||jcb[1].isSelected()||jcb[2].isSelected()||jcb[3].isSelected()||jcb[4].isSele
ted()||
        jcb[5].isSelected()){
        JOptionPane.showMessageDialog(this," 提 交 成 功 "," 提 示 框
",JOptionPane.INFORMATION_MESSAGE);
        ddh++;
    }else{
        JOptionPane.showMessageDialog(this," 您 尚 未 点 菜 ， 请 点 菜 "," 提 示 框
",JOptionPane.INFORMATION_MESSAGE);
    }

```

## 5. 订单信息窗口类



```

static JTable table;//由表格模型创建 table 对象
JScrollPane scrollPane;//将表格对象 table 封装进滚动窗格
JPanel jp=new JPanel();
TableListener tableListener=new TableListener();// 创建自定义的监听器的对象
tableListener
static float price=0;//应收款

public Indent(){
    PreparedStatement ps;
    ResultSet rs;
    try {
        ps = Connect.con.prepareStatement("SELECT 顾客号 FROM 顾客 WHERE 顾客名=?");
        //处理 SQL 命令，获取预编译语句对象
        String u=Login.jName.getText();
        ps.setString(1, u);//准备参数，插入数据
        rs=ps.executeQuery();//执行 SQL 语句
        String gkh=null;//顾客号
        while(rs.next()){
            gkh=rs.getString(1);//顾客号
        }
        String sql="SELECT 订单.订单号,顾客.顾客名,订菜项数,细则号,菜品名,菜品.
        单价,订货数,价格 FROM 顾客,菜品,订单,订单细则 WHERE 订单.订单号=订单细则.订单
        号 AND 订单.顾客号=顾客.顾客号 AND 订单细则.菜品号=菜品.菜品号 AND 顾客.顾客
        号=?";
        ps = Connect.con.prepareStatement(sql);
        ps.setString(1, gkh);
        rs=ps.executeQuery();
        String[][] data = new String[40][8];
        for(int i=0;rs.next();i++){
            for(int j=0;j<8;j++){
                //
                String
                data[i][j]={rs.getString(1),rs.getString(2),rs.getString(3),rs.getString(4),rs.getString(5),rs.getString
                (6),rs.getString(7),rs.getString(8)};
                data[i][j]=rs.getString(j+1);
            }
        }
        //
        for(int i=0;rs.next();i++){
            //
            data[i][]
            ={rs.getString(1),rs.getString(2),rs.getString(3),rs.getString(4),rs.getString(5),rs.getString(6),rs.ge
            tString(7),rs.getString(8)};
            //
        }
    }
}

```

```

        tableModel=new DefaultTableModel(data,head);
        table=new JTable(tableModel);
        scrollPane=new JScrollPane(table);
        table.createDefaultColumnsFromModel();
        table.getSelectionModel().addListSelectionListener(tableListener);
        table.getColumnModel().addColumnModelListener(tableListener);
        for(int i=0;i<jb.length;i++){
            jb[i].addActionListener(this);
            jp.add(jb[i]);
        }
        this.add(scrollPane,BorderLayout.NORTH);
        this.add(jp, BorderLayout.CENTER);

    } catch (SQLException e) {
        e.printStackTrace();
    }

    this.setTitle("订单信息");
    this.setVisible(true);
    this.setBounds(300,50,800,600);
    this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

}

public void actionPerformed(ActionEvent e){
    if(e.getSource()==jb[0]){//全部付款
        String gkh=null;
        String sql1="";
        String sql2="";
        PreparedStatement ps1,ps2;
        try {
            ps1 = Connect.con.prepareStatement("SELECT 顾客号 FROM 顾客
WHERE 顾客名=?");//处理 SQL 命令，获取预编译语句对象
            String u=Login.jName.getText();
            ps1.setString(1, u);//准备参数，插入数据
            ResultSet rs=ps1.executeQuery();//执行 SQL 语句
            while(rs.next()){
                gkh=rs.getString(1);//顾客号
            }
            sql1="SELECT 顾客号,订单.订单号,细则号,价格 FROM 订单,订单细则
WHERE 订单.订单号=订单细则.订单号 AND 顾客号=?";
            sql2="INSERT INTO 应收账款 VALUES(?,?,?,?)";

```

```

        ps1=Connect.con.prepareStatement(sql1);
        ps1.setString(1, gkh);
        rs=ps1.executeQuery();
        ps2=Connect.con.prepareStatement(sql2);
        while(rs.next()){
            ps2.setString(1, rs.getString(1));
            ps2.setString(2, rs.getString(2));
            ps2.setString(3, rs.getString(3));
            ps2.setFloat(4, rs.getFloat(4));
            ps2.executeUpdate();
        }
        ps1=Connect.con.prepareStatement("SELECT SUM(应收金额) FROM 应收
账款");

        rs=ps1.executeQuery();
        while(rs.next()){
            price=rs.getFloat(1);
        }
        //关闭连接
        rs.close();
        ps1.close();
        ps2.close();

        this.dispose();
        new Pay();

    } catch (SQLException e1) {
        e1.printStackTrace();
    }
}

```

```

} else if(e.getSource()==jb[1]){ //付款
    DefaultTableModel temp=(DefaultTableModel)table.getModel();//获得表格模型
    int r=temp.getRowCount();//获得表格的行数
    if(r>0){
        price=Float.parseFloat((String)table.getValueAt(tableListener.row, 7));
    }
    String gkh=null;
    String ddh=null;
    String xzh=null;
    ddh=(String)table.getValueAt(tableListener.row, 0);
    xzh=(String)table.getValueAt(tableListener.row, 3);
    try {

```

```

        PreparedStatement ps=Connect.con.prepareStatement("SELECT 顾客号
FROM 顾客 WHERE 顾客名=?");//处理 SQL 命令，获取预编译语句对象
        String u=Login.jName.getText();
        ps.setString(1, u);//准备参数，插入数据
        ResultSet rs=ps.executeQuery();//执行 SQL 语句
        while(rs.next()){
            gkh=rs.getString(1);//顾客号
        }
        String sql="INSERT INTO 应收账款 VALUES(?,?,?,?)";
        ps=Connect.con.prepareStatement(sql);
        ps.setString(1, gkh);
        ps.setString(2, ddh);
        ps.setString(3, xzh);
        ps.setFloat(4, price);
        ps.executeUpdate();

        this.dispose();
        new Pay();
    } catch (SQLException e1) {
        // TODO Auto-generated catch block
        e1.printStackTrace();
    }

}

}else if(e.getSource()==jb[2]){//取消订单
    DefaultTableModel temp=(DefaultTableModel)table.getModel();//获得表格模型
    int r=temp.getRowCount();//获得表格的行数
    String ddh=null;
    // String dcxs=null;
    String xzh=null;
    if(r>0){//检查行数是否大于 0，如果大于 0，删除指定行
        ddh=(String)table.getValueAt(tableListener.row, 0);//获取删除订单的订单
        // 号
        // dcxs=(String)table.getValueAt(tableListener.row, 2);
        xzh=(String)table.getValueAt(tableListener.row, 3);//获取删除订单的订单细
        // 则号

        temp.removeRow(tableListener.row);
    }
    // System.out.println(ddh);
    // System.out.println(xzh);

    try {
        //删除数据库中的订单细则表中的相关数据
        String sql="DELETE FROM 订单细则 WHERE 订单号=? AND 细则号

```

```

=?";

        PreparedStatement ps=Connect.con.prepareStatement(sql);
        ps.setString(1, ddh);
        ps.setString(2,xzh);
        ps.executeUpdate();
//        System.out.println("删除成功");
//        //将订单表中的相关订菜项数减 1
        sql="UPDATE 订单 SET 订菜项数=订菜项数-1 WHERE 订单号=?";
        ps=Connect.con.prepareStatement(sql);
        ps.setString(1, ddh);
        ps.executeUpdate();
//        System.out.println("更新成功");
        sql="DELETE FROM 订单 WHERE 订菜项数=0";//当订菜项数减为 0 时，
删除该订单

        ps=Connect.con.prepareStatement(sql);
        ps.executeUpdate();
    } catch (SQLException e1) {
        // TODO Auto-generated catch block
        e1.printStackTrace();
    }
    this.dispose();
    new Indent();

    }else if(e.getSource()==jb[3]){//返回
        this.dispose();
        new Function();
    }
}

}

```

//自定义的监听器内部类 TableListener，实现了 2 个监听器接口

```

class TableListener implements ListSelectionListener,TableColumnModelListener{
    int row=0;//定义行索引
    int col=0;//定义列索引
    //实现 ListSelectionListener 监听接口中的方法 valueChanged
    public void valueChanged(ListSelectionEvent e){//当选中发生变化时会调用该方法
        row=Indent.table.getSelectedRow();//获取当前选择的行索引
    }
    //实现 TableColumnModelListener 监听接口中的方法 columnSelectionChanged
    public void columnSelectionChanged(ListSelectionEvent e){//当选中列发生变化时

    }
}

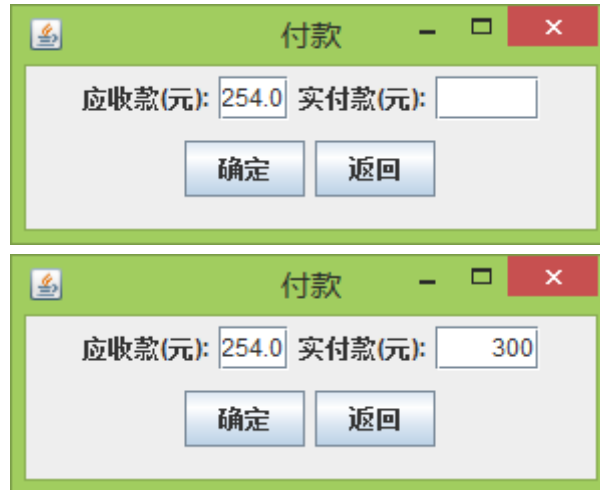
```

```

    public void columnMarginChanged(ChangeEvent ce){ }
    public void columnMoved(TableColumnModelEvent tcme){ }
    public void columnRemoved(TableColumnModelEvent tcme){ }
    public void columnAdded(TableColumnModelEvent tcme){ }
}

```

## 6. 付款窗口类



package com.点菜;

```

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.sql.*;

public class Pay extends JFrame implements ActionListener{
    JPanel jp0=new JPanel();
    JPanel jp1=new JPanel();
    JLabel jl[]={new JLabel("应收款(元):"),new JLabel("实付款(元):")};
    JTextField jtf[]={new JTextField(String.valueOf(Indent.price)),new
    JTextField("        ")};//必须有花括号中的内容
    JButton jb[]={new JButton("确定"),new JButton("返回")};
    FlowLayout fl=new FlowLayout();
    static float price2=0;//实付款
    public Pay(){
        jp0.setLayout(fl);
        jp1.setLayout(fl);
        for(int i=0;i<2;i++){
            //        jl[i].setBounds(30,20+40*i,100,20);
            //        jtf[i].setBounds(220,20+40*i,100,20);
            //        jb[i].setBounds(30+90*i,120,80,20);
            jp0.add(jl[i]);
            jp0.add(jtf[i]);
            jp1.add(jb[i]);
            jb[i].addActionListener(this);
        }
    }
}

```



```

    }
    jtf[1].addActionListener(this);
    this.add(jp0, BorderLayout.NORTH);
    this.add(jp1, BorderLayout.CENTER);
    this.setTitle("付款");
//    this.pack();
    this.setBounds(550,200,300,120);
    this.setVisible(true);
    this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    jtf[1].requestFocus();//获取输入焦点
}

public void actionPerformed(ActionEvent e){
    if(e.getSource()==jtf[1]){//实付款文本框
        price2=Float.parseFloat(jtf[1].getText());

    }else if(e.getSource()==jb[0]){//确定
        price2=Float.parseFloat(jtf[1].getText());
        this.dispose();
        new Change();

    }else if(e.getSource()==jb[1]){//返回

        try {
            String sql="DELETE FROM 应收账款";
            PreparedStatement ps=Connect.con.prepareStatement(sql);
            ps.executeUpdate();
        } catch (SQLException e1) {
            e1.printStackTrace();
        }
        this.dispose();
        new Function();
    }
}
}
}

```

## 7. 找零窗口类



```
package com.点菜;
```

```
import java.awt.*;
```

```
import java.awt.event.*;
```

```
import javax.swing.*;
```

```
import java.sql.*;
```

```
public class Change extends JFrame implements ActionListener{
```

```
    JPanel jp0=new JPanel();
```

```
    JPanel jp1=new JPanel();
```

```
    JLabel jl=new JLabel("找零(元):");
```

```
    JTextField jtf=new JTextField(String.valueOf(Pay.price2-Indent.price));
```

```
    JButton jb=new JButton("确定");
```

```
    public Change(){
```

```
        jp0.add(jl);
```

```
        jp0.add(jtf);
```

```
        jp1.add(jb);
```

```
        jb.addActionListener(this);
```

```
        this.add(jp0, BorderLayout.NORTH);
```

```
        this.add(jp1, BorderLayout.CENTER);
```

```
        this.setTitle("找零");
```

```
        this.setBounds(500,200,250,120);
```

```
        this.setVisible(true);
```

```
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
    }
```

```
    public void actionPerformed(ActionEvent e){
```

```
        if(e.getSource()==jb){//确定
```

```
            String ddh=null;
```

```
            String xxh=null;
```

```
            String sql1=null;
```

```
            String sql2=null;
```

```
            String sql3=null;
```

```
            PreparedStatement ps1,ps2,ps3,ps4;
```

```
            ResultSet rs;
```

```
            try {
```

```
                sql1="SELECT 订单号,细则号 FROM 应收账款";
```

```
                sql2="DELETE FROM 订单细则 WHERE 订单号=? AND 细则号=?";
```

```
                sql3="UPDATE 订单 SET 订菜项数=订菜项数-1 WHERE 订单号=?";
```

```
                ps1=Connect.con.prepareStatement(sql1);
```

```
                rs=ps1.executeQuery();
```

```
                ps2=Connect.con.prepareStatement(sql2);
```

```
                ps3=Connect.con.prepareStatement(sql3);
```

```

        ps4=Connect.con.prepareStatement("DELETE FROM 应收账款");
        ps4.executeUpdate();
        while(rs.next()){
            ps2.setString(1, rs.getString(1));
            ps2.setString(2, rs.getString(2));
            ps2.executeUpdate();
            ps3.setString(1, rs.getString(1));
            ps3.executeUpdate();
        }
        sql1="DELETE FROM 订单 WHERE 订菜项数=0";//当订菜项数减为 0
时，删除该订单

        ps1=Connect.con.prepareStatement(sql1);
        ps1.executeUpdate();

        //关闭连接
        rs.close();
        ps1.close();
        ps2.close();
        ps3.close();
        ps4.close();

        this.dispose();
        new Function();

    } catch (SQLException e1) {
        e1.printStackTrace();
    }

}
}
}

```

## 8. 用户信息类

package com.点菜;

```

public class UserInformation {
    String user1="张三";
    String password1="liuyaodong";
    String user2="李四";
    String password2="liuyaodong";
}

```

## 9. 主方法类

package com.点菜;

```
public class Main {  
    public static void main(String[] args) {  
        new Login();  
    }  
}
```