**《数据库原理及应用》**

**课程设计实验报告书**

**xx大学计算机科学与技术学院**

|  |  |
| --- | --- |
| 姓 名 | ×× |
| 专 业 | ×× |
| 班 级 | ×× |
| 学 号 | ×× |
| 指导教师 | ×× |

|  |  |
| --- | --- |
| 分 数 |  |

2022年 6 月 12 日

摘 要

在当今航空管理系统中，用户机票预定管理是一项非常重要的工作。使用人工方式管理各项材料信息不仅效率低下，而且容易出错，安全性很差。导致了管理费用增加，服务质量难以得到保证，从而影响企业的竞争力。只是现有绝大部分的大中型企业都迫切的需求一种效率高，误差小，安全保密的机票预定信息管理系统代替人工管理，从而为企业带来更大的效益。

该课题结合机票预定的要求，对Oracle 数据库、Java语言的学习和应用，完成了用户信息的增、删、改、查功能和机票、航空公司、航线的管理。系统运行结果证明，此机票预定信息系统可以满足管理者的简单日常管理工作。该系统的核心部分是数据库的生成，利用eclipse完成了管理平台的页面设计，构造所有的类库；使用Oracle数据库存储数据。

该系统的成功运行为医药企业带来信息准确、数据安全、节约人力、提高工作效率和质量、提高管理效率、降低错误发生几率等全方位的经济效益。

关键词：机票预购系统，数据库，窗口应用，Java，JFrame。

**目录**

[1 系统需求分析与设计 2](#_Toc75376030)

[1.1 用户需求分析 2](#_Toc75376031)

[1.1.1 用户需求 2](#_Toc75376032)

[1.1.2 系统功能需求 2](#_Toc75376033)

[1.1.3 系统性能需求 2](#_Toc75376034)

[1.1.4 数据分析 3](#_Toc75376035)

[1.2 功能模块图及分模块功能描述 6](#_Toc75376036)

[1.2.1 系统的功能模块图 6](#_Toc75376037)

[1.2.2 系统功能模块简介 6](#_Toc75376038)

[2 数据库设计 8](#_Toc75376039)

[2.1 数据库概念设计（E-R图） 8](#_Toc75376040)

[2.1 数据库逻辑设计（关系模型） 8](#_Toc75376041)

[3 系统实施 10](#_Toc75376042)

[3.1 建立数据库及其相关对象 10](#_Toc75376043)

[3.2 数据库连接 14](#_Toc75376044)

[3.3 主要模块实施 15](#_Toc75376045)

[3.3.1 登录模块的开发 15](#_Toc75376046)

[3.3.2 主菜单模块的开发 19](#_Toc75376047)

[3.3.3 用户管理模块的开发 23](#_Toc75376048)

[3.4 系统测试 29](#_Toc75376049)

[3.4.1 软件测试的对象 29](#_Toc75376050)

[3.4.2 软件测试的结果 32](#_Toc75376051)

[4 系统说明 33](#_Toc75376052)

[4.1 开发环境 33](#_Toc75376053)

[4.2 系统安装、配置与发布应用程序的步骤 33](#_Toc75376054)

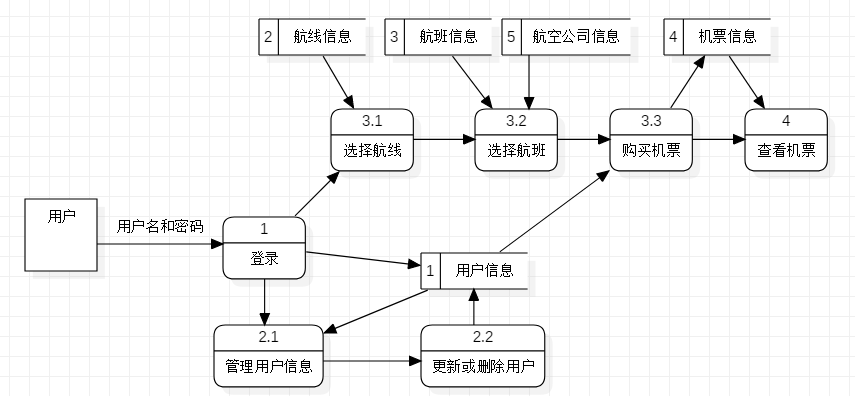
[总 结 34](#_Toc75376055)

[附录：部分源代码 35](#_Toc75376056)

1 系统需求分析与设计

1.1 用户需求分析

### 1.1.1 用户需求



用户通过用户名和密码登录系统进入主界面，在主界面可以进入用户信息管理界面管理用户信息；查询已购买的机票并输出在界面中；进入机票购买的界面购买机票。

### 1.1.2 系统功能需求

1. 用户信息管理  
   用户可以在窗口界面中对自己的用户信息进行修改，新增用户或删除当前用户。
2. 机票预购  
   用户可以在窗口界面中选择自己喜欢的航线，进而选择自己喜欢的航班，进而完成机票的预购。
3. 预购机票管理  
   用户可以在窗口界面中对自己预购的机票进行管理，包括对未付款机票的付款，机票的删除。

### 1.1.3 系统性能需求

1.系统响应时间需求

系统具有快速响应的特性，用户打开界面和提交事务的平均响应时间应低于1.5秒。用户进行实时查询业务操作的数据处理时间应低于5秒。（响应）

2.系统可靠性要求

系统应具有较高的稳定性

3.系统易用性要求

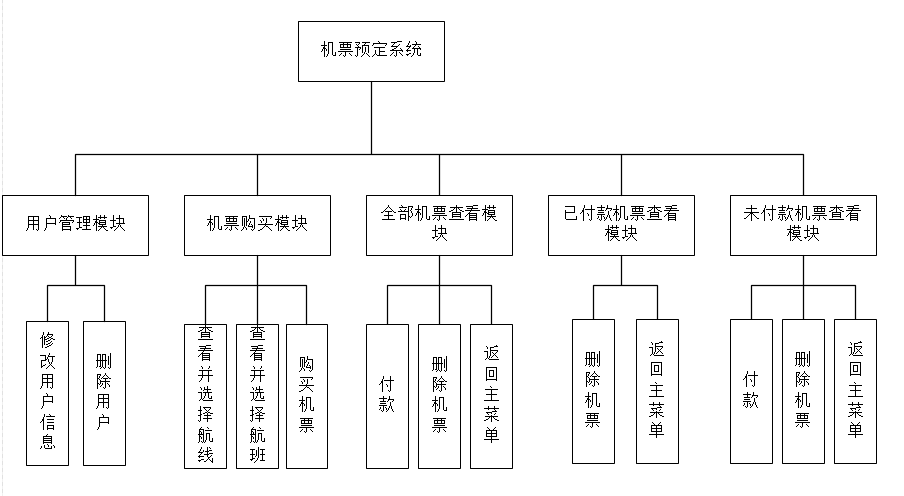
系统用户操作界面应操作简洁、易用。

### 1.1.4 数据分析

|  |  |
| --- | --- |
| **名称** | **描述** |
| 数据项名称 | 证件类型 |
| 别名 | ctype |
| 含义 | 用户使用的证件类型 |
| 类型 | 字符型 |
| 长度 | 20 |
| 取值及含义 | 字符，表示证件类型 |
| **名称** | **描述** |
| 数据项名称 | 证件号码 |
| 别名 | cdata |
| 含义 | 用户使用的证件的号码 |
| 类型 | 字符型 |
| 长度 | 20 |
| 取值及含义 | 任意字符，表示证件号码 |
| **名称** | **描述** |
| 数据项名称 | 用户名 |
| 别名 | cname |
| 含义 | 用户的名字 |
| 类型 | 字符型 |
| 长度 | 20 |
| 取值及含义 | 任意字符，表示用户姓名 |
| **名称** | **描述** |
| 数据项名称 | 联系方式 |
| 别名 | Tel |
| 含义 | 用户的电话 |
| 类型 | 字符型 |
| 长度 | 20 |
| 取值及含义 | 任意字符，用以存储不同格式的联系方式 |
| **名称** | **描述** |
| 数据项名称 | 用户密码 |
| 别名 | Pwd |
| 含义 | 用户登录的密码 |
| 类型 | 字符型 |
| 长度 | 20 |
| 取值及含义 | 任意字符，用户的密码 |
| **名称** | **描述** |
| 数据项名称 | 公司代码 |
| 别名 | Mcode |
| 含义 | 航空公司的公共代码 |
| 类型 | 字符型 |
| 长度 | 10 |
| 取值及含义 | {A-Z}{A-Z}，公司的代码 |
| **名称** | **描述** |
| 数据项名称 | 公司名称 |
| 别名 | Mname |
| 含义 | 航空公司的名称 |
| 类型 | 字符型 |
| 长度 | 20 |
| 取值及含义 | 任意字符，航空公司名称 |
| **名称** | **描述** |
| 数据项名称 | 航线名称 |
| 别名 | Rname |
| 含义 | 航线的名称 |
| 类型 | 字符型 |
| 长度 | 20 |
| 取值及含义 | 0-20，描述航线 |
| **名称** | **描述** |
| 数据项名称 | 航线种类 |
| 别名 | Rtype |
| 含义 | 航线的种类 |
| 类型 | 字符型 |
| 长度 | 20 |
| 取值及含义 | 任意字符，描述航线类型 |
| **名称** | **描述** |
| 数据项名称 | 航线长度 |
| 别名 | Rlength |
| 含义 | 航线的总长 |
| 类型 | Number |
| 长度 | 10 |
| 取值及含义 | {0-10}，航线长度 |
| **名称** | **描述** |
| 数据项名称 | 航班编号 |
| 别名 | Pid |
| 含义 | 航班的编号 |
| 类型 | 字符型 |
| 长度 | 8 |
| 取值及含义 | {A-Z}{A-Z}{0-9}，前两位为航空公司代码，后面为地区及其他信息代码 |
| **名称** | **描述** |
| 数据项名称 | 飞机名称 |
| 别名 | Pname |
| 含义 | 飞机的称呼名 |
| 类型 | 字符型 |
| 长度 | 50 |
| 取值及含义 | 任意字符，飞机的名字 |
| **名称** | **描述** |
| 数据项名称 | 飞机类型 |
| 别名 | Ptype |
| 含义 | 飞机的种类 |
| 类型 | 字符型 |
| 长度 | 10 |
| 取值及含义 | {A-Z}{0-9}，前两位为飞机生产商代码，后两位为飞机代号 |
| **名称** | **描述** |
| 数据项名称 | 机票当前预售情况 |
| 别名 | Tstate |
| 含义 | 机票当前是否预售 |
| 类型 | 字符型 |
| 长度 | 20 |
| 取值及含义 | 任意字符，机票预售与否 |
| **名称** | **描述** |
| 数据项名称 | 机票序号 |
| 别名 | Tid |
| 含义 | 机票的序号 |
| 类型 | Number |
| 长度 | 10 |
| 取值及含义 | {0-9}，表示当前机票是第几张 |
| **名称** | **描述** |
| 数据项名称 | 票价 |
| 别名 | price |
| 含义 | 机票的价钱 |
| 类型 | Number |
| 长度 | 5 |
| 取值及含义 | {0-9}，机票价格 |
| **名称** | **描述** |
| 数据项名称 | 折扣 |
| 别名 | discount |
| 含义 | 机票的折扣 |
| 类型 | Number |
| 长度 | 1 |
| 取值及含义 | 0-9，机票的折扣 |
| **名称** | **描述** |
| 数据项名称 | 机舱等级 |
| 别名 | plevel |
| 含义 | 机舱的等级 |
| 类型 | 字符型 |
| 长度 | 20 |
| 取值及含义 | 任意字符，描述机舱等级 |
| **名称** | **描述** |
| 数据项名称 | 经手业务员 |
| 别名 | Tmanager |
| 含义 | 处理的业务员 |
| 类型 | 字符型 |
| 长度 | 50 |
| 取值及含义 | 任意字符，业务员姓名 |
| **名称** | **描述** |
| 数据项名称 | 付款情况 |
| 别名 | cstate |
| 含义 | 当前机票得付款情况 |
| 类型 | 字符型 |
| 长度 | 20 |
| 取值及含义 | 任意字符，当前付款情况 |

1.2 功能模块图及分模块功能描述

### 1.2.1 系统的功能模块图



### 1.2.2 系统功能模块简介

**用户管理模块：**用户管理模块是用来修改用户数据，包括对用户证件类型，证件号码，用户密码，联系方式的修改，并且可以删除某用户的模块。

**机票购买模块：**机票购买模块是用来供用户预购机票的模块，包括查看当前航线，根据选择的航线选择航班，根据选择的航班预订机票。

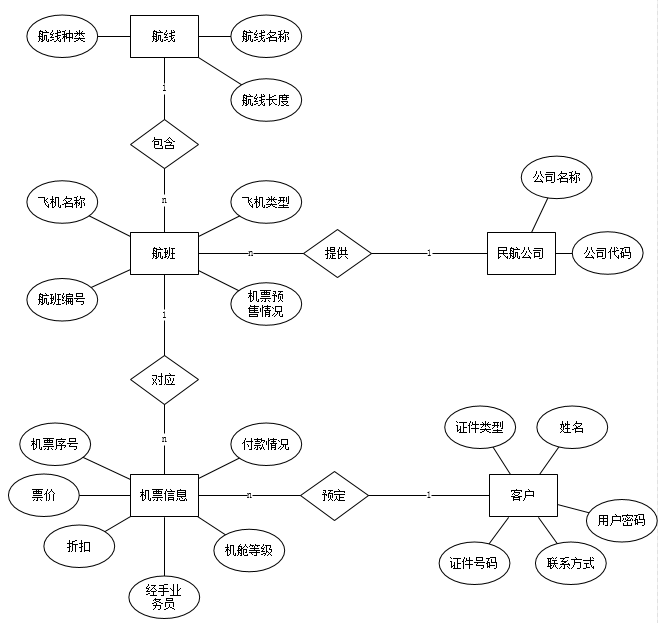
**全部机票查看模块：**全部机票查看模块是用来供用户查看机票的模块，包括查看用户预购的全部机票，为未付款的机票付款，删除机票。

**已付款机票查看模块：**已付款机票查看模块是用来供用户查看机票的模块，包括查看用户已付款的预购机票，删除机票。

**未付款机票查看模块：**未付款机票查看模块是用来供用户查看机票的模块，包括查看用户未付款的机票，为未付款的机票付款，删除机票。

2 数据库设计

2.1 数据库概念设计（E-R图）



2.1 数据库逻辑设计（关系模型）



**初始设计：**

航班（航班编号，公司代码，公司名称，航线名称，航线种类，飞机名称，飞机类型，机票当前预售情况）

机票（机票编号，航班编号，证件类型，证件号码，用户名，票价，折扣，机舱等级，经手业务员，付款情况）

客户（证件类型，证件号码，用户名，联系方式，用户密码）

**经过规范化后：**

航线（航线名称，航线种类，航线长度）

航空公司（公司代码，公司名称）

航班（航班编号，公司代码，航线名称，飞机名称，飞机类型，机票当前预售情况）

机票（机票编号，航班编号，证件类型，证件号码，票价，折扣，机舱等级，经手业务员，付款情况）

客户（证件类型，证件号码，用户名，联系方式，用户密码）

3 系统实施

3.1 建立数据库及其相关对象

数据库表目录：

|  |  |
| --- | --- |
| Company | 航空公司表 |
| Customer | 客户表 |
| Route | 航线表 |
| Plane | 航班表 |
| Ticket | 机票表 |

数据库结构表：

|  |  |  |
| --- | --- | --- |
| Company: 航空公司记录表  表说明：记录每一个航班对应的航空公司名称 | | |
| 字段 | 类型 Null 默认 | 注释 |
| MCODE | VARCHAR2(10) No null | 主键，航空公司的代码 |
| MNAME | VARCHAR2(20) Yes null | 航空公司的姓名 |
| Customer:客户表  表说明：记录每一位客户的信息 | | |
| 字段 | 类型 Null 默认 | 注释 |
| CTYPE | VARCHAR2(20) No null | 关联主键，客户的证件类型 |
| CDATA | VARCHAR2(20) No null | 关联主键，客户的证件号码 |
| CNAME | VARCHAR2(20) Yes null | 客户的姓名 |
| TEL | VARCHAR2(20) Yes null | 客户的联系方式 |
| PWD | VARCHAR2(10) Yes null | 客户的登录密码 |
| Route:航线表  表说明：记录每一条航线的信息 | | |
| 字段 | 类型 Null 默认 | 注释 |
| RNAME | VARCHAR2(20) No null | 主键，航线的名称 |
| RTYPE | VARCHAR2(20) Yes null | 航线的类型 |
| RLENGTH | NUMBER(10) Yes null | 航线的长度 |
| Plane:航班表  表说明：记录每一条航线所拥有的的航班 | | |
| 字段 | 类型 Null 默认 | 注释 |
| PID | VARCHAR2(8) No null | 主键，航班的序号 |
| MCODE | VARCHAR2(10) No null | 提供该航班的公司代码 |
| RNAME | VARCHAR2(20) No null | 该航班所飞的航线 |
| PNAME | VARCHAR2(50) Yes null | 飞该航班的飞机名称 |
| PTYPE | VARCHAR2(10) Yes null | 飞该航班的飞机型号 |
| TSTATE | VARCHAR2(20) Yes null | 该航班是否售票 |
| Ticket:机票表  表说明：记录用户预订的每一张机票 | | |
| 字段 | 类型 Null 默认 | 注释 |
| TID | NUMBER(10) No null | 主键，机票序号 |
| PID | VARCHAR2(8) No null | 航班序号 |
| CTYPE | VARCHAR2(20) Yes null | 客户的证件类型 |
| CDATA | VARCHAR2(20) Yes null | 客户的证件信息 |
| PRICE | NUMBER(5,2) Yes null | 机票的价格 |
| DISCOUNT | NUMBER(1) Yes null | 机票的折扣 |
| PLEVEL | VARCHAR2(20) Yes null | 机舱等级 |
| TMANAGER | VARCHAR2(50) Yes null | 经手业务员 |
| CSTATE | VARCHAR2(20) Yes null | 付款状态 |

/\*==============================================================\*/

/\* Table: "Company" \*/

/\*==============================================================\*/

create table "Company"

(

MCODE VARCHAR2(10) not null,

MNAME VARCHAR2(20),

constraint PK\_COMPANY primary key (MCODE)

);

/\*==============================================================\*/

/\* Table: "Customer" \*/

/\*==============================================================\*/

create table "Customer"

(

CTYPE VARCHAR2(20) not null,

CDATA VARCHAR2(20) not null,

CNAME VARCHAR2(20),

TEL VARCHAR2(20),

PWD VARCHAR2(10),

constraint PK\_CUSTOMER primary key (CTYPE, CDATA)

);

/\*==============================================================\*/

/\* Table: "Plane" \*/

/\*==============================================================\*/

create table "Plane"

(

PID VARCHAR2(8) not null,

MCODE VARCHAR2(10) not null,

RNAME VARCHAR2(20) not null,

PNAME VARCHAR2(50),

PTYPE VARCHAR2(10),

TSTATE VARCHAR2(20),

constraint PK\_PLANE primary key (PID)

);

/\*==============================================================\*/

/\* Index: "Include\_FK" \*/

/\*==============================================================\*/

create index "Include\_FK" on "Plane" (

RNAME ASC

);

/\*==============================================================\*/

/\* Index: "Provide\_FK" \*/

/\*==============================================================\*/

create index "Provide\_FK" on "Plane" (

MCODE ASC

);

/\*==============================================================\*/

/\* Table: "Route" \*/

/\*==============================================================\*/

create table "Route"

(

RNAME VARCHAR2(20) not null,

RTYPE VARCHAR2(20),

RLENGTH NUMBER(10),

constraint PK\_ROUTE primary key (RNAME)

);

/\*==============================================================\*/

/\* Table: "Ticket" \*/

/\*==============================================================\*/

create table "Ticket"

(

TID NUMBER(10) not null,

PID VARCHAR2(8) not null,

CTYPE VARCHAR2(20),

CDATA VARCHAR2(20),

PRICE NUMBER(5,2),

DISCOUNT NUMBER(1),

PLEVEL VARCHAR2(20),

TMANAGER VARCHAR2(50),

CSTATE VARCHAR2(20),

constraint PK\_TICKET primary key (TID)

);

/\*==============================================================\*/

/\* Index: "Corresponding\_FK" \*/

/\*==============================================================\*/

create index "Corresponding\_FK" on "Ticket" (

PID ASC

);

/\*==============================================================\*/

/\* Index: "Book\_FK" \*/

/\*==============================================================\*/

create index "Book\_FK" on "Ticket" (

CTYPE ASC,

CDATA ASC

);

alter table "Plane"

add constraint FK\_PLANE\_INCLUDE\_ROUTE foreign key (RNAME)

references "Route" (RNAME);

alter table "Plane"

add constraint FK\_PLANE\_PROVIDE\_COMPANY foreign key (MCODE)

references "Company" (MCODE);

alter table "Ticket"

add constraint FK\_TICKET\_BOOK\_CUSTOMER foreign key (CTYPE, CDATA)

references "Customer" (CTYPE, CDATA);

alter table "Ticket"

add constraint FK\_TICKET\_CORRESPON\_PLANE foreign key (PID)

references "Plane" (PID);

/\*创建用户信息更改触发器\*/

create or replace trigger ticket\_update after update on customer

for each row

begin

update ticket set ctype=:new.ctype,cdata=:new.cdata where (ctype=:old.ctype and cdata=:old.cdata);

end;

/

/\*创建机票信息删除触发器\*/

create or replace trigger ticket\_delete after delete on customer

for each row

begin

delete from ticket where (ctype=:old.ctype and cdata=:old.cdata);

end;

/

/\*创建机票序号序列\*/

create sequence ticket\_seq

maxvalue 99999999

start with 1

increment by 1

cache 50;

3.2 数据库连接

使用JDBC进行数据库连接

**package** jdbcutils;

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.SQLException;

//数据库连接类

**public** **class** JdbcConn {

/\*定义连接用变量，使用私用变量，用于保护，

\* 提高安全性，

\* 但方法是静态类型，要调用这个变量，

\* 所以也需要定义为static类型\*/

**private** **static** Connection *conn* = **null**; //定义一个Connection对象

**private** **static** String *url* = "jdbc:oracle:thin:@localhost:1521:xe";

**private** **static** String *user* = "ticket";

**private** **static** String *password* = "ticket";

/\*注册驱动

\* 放在静态代码块中

\* 当类加载到内容中，该代码块就会自动运行 \*/

**static** {

**try** {

Class.*forName*("oracle.jdbc.OracleDriver");

} **catch** (ClassNotFoundException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

//要用类名且不产生对象，直接访问的方法需要定义成static

**public** **static** Connection getConnection(){

**try** {

*conn* = DriverManager.*getConnection*(*url*,*user*,*password*);

**return** *conn*;

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

**return** **null**;

}

}

**public** **static** **void** closeConnection(){

**if**(*conn* != **null**) {

**try** {

*conn*.close();

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

}

3.3 主要模块实施

### 3.3.1 登录模块的开发

**package** frame;

**import** java.awt.BorderLayout;

**import** java.awt.EventQueue;

**import** javax.swing.JFrame;

**import** javax.swing.JPanel;

**import** javax.swing.border.EmptyBorder;

**import** classes.Customer;

**import** javadao.CustomerDao;

**import** javax.swing.JLabel;

**import** javax.swing.JOptionPane;

**import** javax.swing.JPasswordField;

**import** javax.swing.JTextField;

**import** javax.swing.JButton;

**import** java.awt.event.ActionListener;

**import** java.awt.event.ActionEvent;

**import** javax.swing.JToolBar;

**import** javax.swing.JList;

**import** javax.swing.JComboBox;

**import** javax.swing.DefaultComboBoxModel;

**public** **class** LoginFrame **extends** JFrame {

**private** JPanel contentPane;

**private** JPasswordField passPwd;

**private** JTextField textCustomeriddata;

**private** JTextField textCustomeridtype;

/\*\*

\* Launch the application.

\*/

**public** **static** **void** main(String[] args) {

EventQueue.*invokeLater*(**new** Runnable() {

**public** **void** run() {

**try** {

LoginFrame frame = **new** LoginFrame();

frame.setVisible(**true**);

} **catch** (Exception e) {

e.printStackTrace();

}

}

});

}

/\*\*

\* Create the frame.

\*/

**public** LoginFrame() {

setTitle("\u767B\u5F55\u7A97\u4F53");

setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

setBounds(800, 300, 531, 360);

contentPane = **new** JPanel();

contentPane.setBorder(**new** EmptyBorder(5, 5, 5, 5));

setContentPane(contentPane);

contentPane.setLayout(**null**);

JLabel ctype = **new** JLabel("\u8BC1\u4EF6\u7C7B\u578B");

ctype.setBounds(98, 65, 72, 18);

contentPane.add(ctype);

textCustomeridtype = **new** JTextField();

textCustomeridtype.setColumns(10);

textCustomeridtype.setBounds(206, 64, 170, 24);

contentPane.add(textCustomeridtype);

JLabel cdata = **new** JLabel("\u8BC1\u4EF6\u53F7");

cdata.setBounds(98, 115, 72, 18);

contentPane.add(cdata);

JLabel pwd = **new** JLabel("\u5BC6 \u7801");

pwd.setBounds(98, 169, 72, 18);

contentPane.add(pwd);

passPwd = **new** JPasswordField();

passPwd.setBounds(206, 166, 170, 24);

contentPane.add(passPwd);

textCustomeriddata = **new** JTextField();

textCustomeriddata.setBounds(206, 112, 170, 24);

contentPane.add(textCustomeriddata);

textCustomeriddata.setColumns(10);

JButton btnOK = **new** JButton("\u767B\u5F55");

btnOK.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

CustomerDao u = **new** CustomerDao();

**if**(textCustomeridtype.getText().isEmpty()) {

JOptionPane.*showMessageDialog*(getContentPane(), "证件类型不能为空","提示信息",JOptionPane.***ERROR\_MESSAGE***);

}**else** **if**(textCustomeriddata.getText().isEmpty()) {

JOptionPane.*showMessageDialog*(getContentPane(), "证件号不能为空","提示信息",JOptionPane.***ERROR\_MESSAGE***);

} **else** **if**(**new** String(passPwd.getPassword()).isEmpty()) {

JOptionPane.*showMessageDialog*(getContentPane(), "密码不能为空","提示信息",JOptionPane.***ERROR\_MESSAGE***);

} **else** **if**(u.findById(textCustomeridtype.getText().trim(), textCustomeriddata.getText().trim(), **new** String(passPwd.~~getText~~()).trim())!= **null**) {

Customer c = u.findById(textCustomeridtype.getText().trim(),textCustomeriddata.getText().trim(), **new** String(passPwd.~~getText~~()).trim());

MainFrame mainFrame = **new** MainFrame(c,c.getCname());

mainFrame.*loginname* = c.getCname();

mainFrame.setTitle("主窗体 欢迎您："+mainFrame.*loginname*);

mainFrame.setVisible(**true**);

setVisible(**false**);

} **else** {

JOptionPane.*showMessageDialog*(getContentPane(), "用户名或密码错误","提示信息",JOptionPane.***ERROR\_MESSAGE***);

}

}

});

btnOK.setBounds(100, 251, 113, 27);

contentPane.add(btnOK);

JButton btnCancel = **new** JButton("\u53D6\u6D88");

btnCancel.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

textCustomeriddata.setText("");

passPwd.setText("");

}

});

btnCancel.setBounds(324, 251, 113, 27);

contentPane.add(btnCancel);

}

}

结果如下：



### 3.3.2 主菜单模块的开发

**package** frame;

**import** java.awt.BorderLayout;

**import** java.awt.EventQueue;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.ActionListener;

**import** java.util.Vector;

**import** javax.swing.JFrame;

**import** javax.swing.JPanel;

**import** javax.swing.border.EmptyBorder;

**import** classes.Customer;

**import** javadao.TicketDao;

**import** javax.swing.JMenuBar;

**import** javax.swing.JMenu;

**import** javax.swing.JMenuItem;

**import** javax.swing.JLabel;

**import** java.awt.Font;

**public** **class** MainFrame **extends** JFrame {

**public** **static** String *loginname*;

**public** **static** Customer *u*;

**private** JPanel contentPane;

/\*\*

\* Create the frame.

\*/

**public** MainFrame(Customer u,String name) {

**this**.*u* = u;

**this**.*loginname* = name;

setTitle("主窗体 欢迎您："+*loginname*);

setBounds(800, 300, 692, 489);

JMenuBar menuBar = **new** JMenuBar();

setJMenuBar(menuBar);

JMenu mnNewMenu = **new** JMenu("\u7528\u6237\u7BA1\u7406");

menuBar.add(mnNewMenu);

/\*

\* mnFindUser.addActionListener(new ActionListener() {

\*

\* @Override public void actionPerformed(ActionEvent e) { // **TODO** Auto-generated

\* method stub FindUsers finduser = new FindUsers(); finduser.setVisible(true);

\* }

\*

\* });

\*/

JMenuItem mnUpdateUser = **new** JMenuItem("\u4FE1\u606F\u66F4\u65B0");

mnNewMenu.add(mnUpdateUser);

mnUpdateUser.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent arg0) {

// **TODO** Auto-generated method stub

ManageUsers manageUser = **new** ManageUsers(u);

manageUser.setVisible(**true**);

setVisible(**false**);

}

});

JMenuItem mnRelogin = **new** JMenuItem("\u91CD\u65B0\u767B\u5F55");

mnNewMenu.add(mnRelogin);

mnRelogin.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent arg0) {

// **TODO** Auto-generated method stub

LoginFrame loginframe = **new** LoginFrame();

loginframe.setVisible(**true**);

setVisible(**false**);

}

});

JMenu mnTicket = **new** JMenu("\u673A\u7968\u7BA1\u7406");

menuBar.add(mnTicket);

JMenuItem mnFindHave = **new** JMenuItem("\u67E5\u8BE2\u5DF2\u8D2D\u4E70\u673A\u7968");

mnTicket.add(mnFindHave);

mnFindHave.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent arg0) {

// **TODO** Auto-generated method stub

TicketDao tdao = **new** TicketDao();

Vector columns = tdao.findColumnNames();

Vector rowData = tdao.findTicket("已付款");

TicketBuyFrame ticketbuyframe = **new** TicketBuyFrame(columns,rowData,u);

ticketbuyframe.setVisible(**true**);

setVisible(**false**);

}

});

JMenuItem mnFindNot = **new** JMenuItem("\u67E5\u8BE2\u672A\u4ED8\u6B3E\u673A\u7968");

mnTicket.add(mnFindNot);

mnFindNot.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent arg0) {

// **TODO** Auto-generated method stub

TicketDao tdao = **new** TicketDao();

Vector columns = tdao.findColumnNames();

Vector rowData = tdao.findTicket("未付款");

TicketNotBuyFrame ticketnotbuyframe = **new** TicketNotBuyFrame(columns,rowData,u);

ticketnotbuyframe.setVisible(**true**);

setVisible(**false**);

}

});

JMenuItem mnFindAll = **new** JMenuItem("\u67E5\u8BE2\u6240\u6709\u673A\u7968");

mnTicket.add(mnFindAll);

mnFindAll.addActionListener(**new** ActionListener() {

@Override

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

// **TODO** Auto-generated method stub

TicketDao tdao = **new** TicketDao();

Vector columns = tdao.findColumnNames();

Vector rowData = tdao.findAll();

AllTicketFrame all = **new** AllTicketFrame(columns,rowData,u);

all.setVisible(**true**);

setVisible(**false**);

}

});

JMenuItem mnBuy = **new** JMenuItem("\u8D2D\u4E70\u673A\u7968");

mnTicket.add(mnBuy);

mnBuy.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent arg0) {

// **TODO** Auto-generated method stub

AirLaneFrame airlaneframe = **new** AirLaneFrame(MainFrame.*u*);

airlaneframe.setVisible(**true**);

setVisible(**false**);

}

});

contentPane = **new** JPanel();

contentPane.setBorder(**new** EmptyBorder(5, 5, 5, 5));

setContentPane(contentPane);

contentPane.setLayout(**null**);

JLabel lblNewLabel = **new** JLabel("\u6B22\u8FCE\u4F7F\u7528\u672C\u7CFB\u7EDF");

lblNewLabel.setFont(**new** Font("宋体", Font.***PLAIN***, 25));

lblNewLabel.setBounds(188, 145, 224, 30);

contentPane.add(lblNewLabel);

JLabel lblNewLabel\_1 = **new** JLabel("\u8BF7\u70B9\u51FB\u4E0A\u65B9\u6309\u94AE\u8FDB\u884C\u64CD\u4F5C");

lblNewLabel\_1.setFont(**new** Font("宋体", Font.***PLAIN***, 25));

lblNewLabel\_1.setBounds(188, 199, 294, 45);

contentPane.add(lblNewLabel\_1);

JLabel lblNewLabel\_2 = **new** JLabel("\u795D\u60A8\u4F7F\u7528\u6109\u5FEB");

lblNewLabel\_2.setFont(**new** Font("宋体", Font.***PLAIN***, 25));

lblNewLabel\_2.setBounds(188, 257, 203, 46);

contentPane.add(lblNewLabel\_2);

}

}

结果如下：



### 3.3.3 用户管理模块的开发

**package** frame;

**import** java.awt.BorderLayout;

**import** java.awt.EventQueue;

**import** javax.swing.JFrame;

**import** javax.swing.JPanel;

**import** javax.swing.border.EmptyBorder;

**import** classes.Customer;

**import** javadao.CustomerDao;

**import** javax.swing.JLabel;

**import** javax.swing.JOptionPane;

**import** javax.swing.JTextField;

**import** javax.swing.JPasswordField;

**import** javax.swing.JButton;

**import** javax.swing.JSeparator;

**import** java.awt.event.ActionListener;

**import** java.awt.event.ActionEvent;

**public** **class** ManageUsers **extends** JFrame {

**private** Customer u;

**private** JPanel contentPane;

**private** JTextField CID;

**private** JPasswordField Cpwd;

**private** JTextField CType;

**private** JTextField CId;

**private** JTextField CName;

**private** JTextField CTel;

**private** JTextField CPwd;

**private** JTextField Ctype;

/\*\*

\* Launch the application.

\*/

/\*

\* public static void main(String[] args) { EventQueue.invokeLater(new

\* Runnable() { public void run() { try { ManageUsers frame = new ManageUsers();

\* frame.setVisible(true); } catch (Exception e) { e.printStackTrace(); } } });

\* }

\*/

/\*\*

\* Create the frame.

\*/

**public** ManageUsers(Customer u) {

**this**.u = u;

setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

setBounds(800, 300, 746, 497);

contentPane = **new** JPanel();

contentPane.setBorder(**new** EmptyBorder(5, 5, 5, 5));

setContentPane(contentPane);

contentPane.setLayout(**null**);

JLabel ID = **new** JLabel("\u8BC1\u4EF6\u53F7");

ID.setBounds(195, 42, 72, 18);

contentPane.add(ID);

Ctype = **new** JTextField();

Ctype.setColumns(10);

Ctype.setBounds(84, 40, 86, 24);

contentPane.add(Ctype);

JLabel type = **new** JLabel("\u8BC1\u4EF6\u7C7B\u578B");

type.setBounds(28, 41, 72, 18);

contentPane.add(type);

CID = **new** JTextField();

CID.setBounds(250, 40, 86, 24);

contentPane.add(CID);

CID.setColumns(10);

JLabel pwd = **new** JLabel("\u5BC6\u7801");

pwd.setBounds(356, 44, 72, 18);

contentPane.add(pwd);

Cpwd = **new** JPasswordField();

Cpwd.setBounds(398, 40, 86, 23);

contentPane.add(Cpwd);

JButton btnConfirm = **new** JButton("\u786E\u8BA4");

btnConfirm.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

Customer u = **new** Customer(**null**, **null**, **null**, **null**, **null**);

CustomerDao udao = **new** CustomerDao();

**if**(!CID.getText().trim().isEmpty() && !**new** String(Cpwd.getPassword()).trim().isEmpty()) {

**if**(udao.findById(Ctype.getText().trim(), CID.getText().trim(), Cpwd.~~getText~~().trim())!=**null**) {

u = udao.findById(Ctype.getText().trim(), CID.getText().trim(), **new** String(Cpwd.~~getText~~()).trim());

CTel.setText(u.getTel());

CPwd.setText(u.getPwd());

CType.setText(u.getCtype());

CId.setText(u.getCdata());

CName.setText(u.getCname());

} **else** {

JOptionPane.*showMessageDialog*(getContentPane(),"请检查输入内容");

}

} **else** **if**(CID.getText().trim().isEmpty()){

JOptionPane.*showMessageDialog*(getContentPane(),"请输入证件号码");

} **else** **if**(**new** String(Cpwd.getPassword()).isEmpty()){

JOptionPane.*showMessageDialog*(getContentPane(), "请输入密码");

}

}

});

btnConfirm.setBounds(554, 38, 113, 27);

contentPane.add(btnConfirm);

JSeparator separator = **new** JSeparator();

separator.setBounds(28, 96, 675, 10);

contentPane.add(separator);

JLabel C\_type = **new** JLabel("\u8BC1\u4EF6\u7C7B\u578B");

C\_type.setBounds(124, 148, 72, 18);

contentPane.add(C\_type);

CType = **new** JTextField();

CType.setBounds(228, 145, 86, 24);

contentPane.add(CType);

CType.setColumns(10);

JLabel C\_name = **new** JLabel("\u7528\u6237\u540D");

C\_name.setBounds(124, 222, 72, 18);

contentPane.add(C\_name);

CId = **new** JTextField();

CId.setBounds(525, 145, 86, 24);

contentPane.add(CId);

CId.setColumns(10);

JLabel C\_tel = **new** JLabel("\u8054\u7CFB\u65B9\u5F0F");

C\_tel.setBounds(124, 285, 72, 18);

contentPane.add(C\_tel);

CName = **new** JTextField();

CName.setBounds(228, 219, 86, 24);

contentPane.add(CName);

CName.setColumns(10);

JLabel C\_id = **new** JLabel("\u8BC1\u4EF6\u53F7\u7801");

C\_id.setBounds(439, 148, 72, 18);

contentPane.add(C\_id);

CTel = **new** JTextField();

CTel.setBounds(228, 282, 86, 24);

contentPane.add(CTel);

CTel.setColumns(10);

JLabel C\_pwd = **new** JLabel("\u5BC6 \u7801");

C\_pwd.setBounds(439, 222, 72, 18);

contentPane.add(C\_pwd);

CPwd = **new** JTextField();

CPwd.setBounds(525, 219, 86, 24);

contentPane.add(CPwd);

CPwd.setColumns(10);

JSeparator separator\_1 = **new** JSeparator();

separator\_1.setBounds(28, 330, 675, 10);

contentPane.add(separator\_1);

JButton btnUpdate = **new** JButton("\u4FEE\u6539\u4FE1\u606F");

btnUpdate.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

CustomerDao udao = **new** CustomerDao();

Customer u = **new** Customer(**null**, **null**, **null**, **null**, **null**);

u.setCdata(CId.getText());

u.setCtype(CType.getText());

u.setCname(CName.getText());

u.setTel(CTel.getText());

u.setPwd(CPwd.getText());

**if**(!CTel.getText().isEmpty()) {

**if**(udao.doUpdate1(u) > 0) {

JOptionPane.*showMessageDialog*(getContentPane(), "用户修改成功");

} **else** **if**(udao.doUpdate1(u) == -5) {

JOptionPane.*showMessageDialog*(getContentPane(), "该用户已存在");

} **else** **if**(udao.doUpdate1(u) == -4) {

JOptionPane.*showMessageDialog*(getContentPane(), "存在不能为空的信息");

} **else** {

**if**(udao.doInsert(u)) {

JOptionPane.*showMessageDialog*(getContentPane(), "新建用户成功");

} **else** {

JOptionPane.*showMessageDialog*(getContentPane(), "用户修改失败");

}

}

}

}

});

btnUpdate.setBounds(129, 371, 113, 27);

contentPane.add(btnUpdate);

JButton btnDelete = **new** JButton("\u5220\u9664\u4FE1\u606F");

btnDelete.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

CustomerDao udao = **new** CustomerDao();

Customer u = **new** Customer(**null**, **null**, **null**, **null**, **null**);

u.setCname(CTel.getText());

**if**(!CTel.getText().isEmpty()) {

**if**(udao.doDelete(u)) {

JOptionPane.*showMessageDialog*(getContentPane(), "用户删除成功");

CType.setText("");

CPwd.setText("");

CTel.setText("");

CId.setText("");

CName.setText("");

CID.setText("");

Cpwd.setText("");

**if**(udao.findAll() == **null**) {

LoginFrame frame = **new** LoginFrame();

frame.setVisible(**true**);

setVisible(**false**);

}

} **else** {

JOptionPane.*showMessageDialog*(getContentPane(), "用户删除失败");

}

}

}

});

btnDelete.setBounds(314, 371, 113, 27);

contentPane.add(btnDelete);

JButton btnBack = **new** JButton("\u8FD4\u56DE\u4E3B\u83DC\u5355");

btnBack.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

MainFrame mainFrame = **new** MainFrame(u,u.getCname());

mainFrame.setVisible(**true**);

setVisible(**false**);

}

});

btnBack.setBounds(498, 371, 113, 27);

contentPane.add(btnBack);

}

}

结果如下：



3.4 系统测试

### 3.4.1 软件测试的对象

以用户的相关操作模块CustomerDao.java为测试对象，设计测试代码TestCustomerDao.java如下：

**package** test;

**import** java.util.Vector;

**import** javax.swing.JOptionPane;

**import** org.junit.Test;

**import** classes.Customer;

**import** javadao.CustomerDao;

**public** **class** TestCustomerDao {

@Test

@SuppressWarnings("rawtypes")

**public** **void** textdoInsert(){

Customer u = **new** Customer("身份证","123456","张三","1234567","000000");

CustomerDao udao = **new** CustomerDao();

**if**(udao.doInsert(u)) {

System.***out***.println("插入成功");

} **else** {

System.***out***.println("插入失败");

};

}

@Test

**public** **void** textdoUpdate(){

Customer u = **new** Customer("身份证","1234567","李四","1234567","000000");

u.setTel("123456");

CustomerDao udao = **new** CustomerDao();

**if**(udao.doUpdate(u)) {

System.***out***.println("更新成功");

} **else** {

System.***out***.println("更新失败");

};

}

@Test

**public** **void** textdoUpdate1() {

Customer u = **new** Customer("身份证","1234567","李四","123456","000000");

u.setPwd("123456");

CustomerDao udao = **new** CustomerDao();

**if**(udao.doUpdate1(u) > 0) {

System.***out***.println("用户修改成功");

} **else** **if**(udao.doUpdate1(u) == -5) {

System.***out***.println("该用户已存在");

} **else** **if**(udao.doUpdate1(u) == -4) {

System.***out***.println("存在不能为空的信息");

} **else** {

System.***out***.println("用户修改失败");

}

}

@Test

**public** **void** textdoDelete(){

Customer u = **new** Customer(**null**, **null**, **null**, **null**, **null**);

u.setCtype("身份证");

u.setCdata("1234567");

CustomerDao udao = **new** CustomerDao();

**if**(udao.doDelete(u)) {

System.***out***.println("删除成功");

} **else** {

System.***out***.println("删除失败");

};

}

@SuppressWarnings("rawtypes")

@Test

**public** **void** textfindAll(){

CustomerDao udao = **new** CustomerDao();

Vector rowData = **new** Vector();

rowData = udao.findAll();

System.***out***.println(rowData);

}

@SuppressWarnings("rawtypes")

@Test

**public** **void** textfindByName(){

CustomerDao udao = **new** CustomerDao();

String name = "张";

Vector rowData = **new** Vector();

rowData = udao.findByName(name);

System.***out***.println(rowData);

}

@Test

**public** **void** textfindById(){

CustomerDao udao = **new** CustomerDao();

Customer u = **new** Customer(**null**, **null**, **null**, **null**, **null**);

String ctype = "身份证";

String cdata = "123456";

String pwd = "000000";

u = udao.findById(ctype,cdata,pwd);

System.***out***.println(u);

}

@Test

**public** **void** textfindColumnNames(){

CustomerDao udao = **new** CustomerDao();

Vector colNames = **new** Vector();

colNames = udao.findColumnNames();

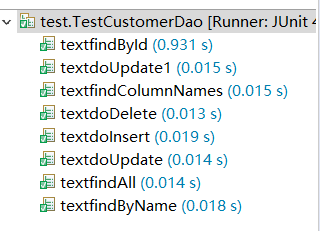
System.***out***.println(colNames);

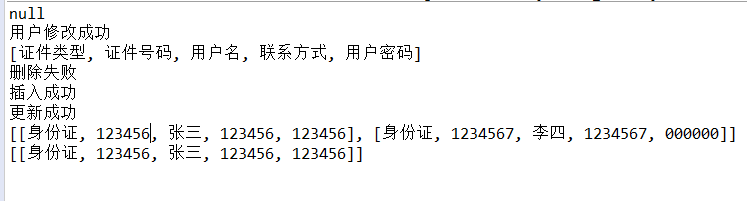
}

}

### 3.4.2 软件测试的结果

测试程序运行顺序和结果如下：





4 系统说明

## 4.1 开发环境

本系统的是在Java平台下开发的，系统的后台数据库为oracle 数据库；因此在使用本系统前，应先安装Java框架和oracle数据库，否则该系统无法运行。

## 4.2 系统安装、配置与发布应用程序的步骤

系统的原文件直接复制到机器上后，在数据库管理系统上建立一个名为ticket的数据库后，系统便可以运行。

总 结

在整个课程设计的工作中，自己自行学习了java与数据库的连接方法，以及如何使用java绘制窗体，设计窗口运行程序，使自己对java应用有了进一步的了解，提升了自己的java设计水平。

在本次课程设计中，完成了一个简单的机票预订系统，解决了java与数据库的连接问题，用java绘制窗口，编写窗口应用程序的问题。但其中仍有未解决的问题，例如对整体框架的完善和窗口之间的关联性。日后会对系统的框架进行进一步的完善，以提高系统的可扩展性和可维护性。

通过本次课程设计，也体会到了软件开发过程的艰辛以及辛苦，但也体会到了程序与程序之间的奇妙关系，对编程的兴趣有了进一步的提高。

附录：部分源代码

**CustomerDao.java**

**package** javadao;

**import** java.sql.Connection;

**import** java.sql.PreparedStatement;

**import** java.sql.ResultSet;

**import** java.sql.ResultSetMetaData;

**import** java.sql.SQLException;

**import** java.util.Vector;

**import** classes.Customer;

**import** jdbcutils.JdbcConn;

/\*

\* Customer类的存取

\* \*/

**public** **class** CustomerDao {

**private** Connection conn;

**private** PreparedStatement pstmt; //执行数据库操作

**private** ResultSet rs; //存放数据库返回的结果

**private** ResultSetMetaData rsmd; //存放数据库中的源数据

/\*

\* 执行插入操作

\* \*/

**public** **boolean** doInsert(Customer u) {

conn = JdbcConn.*getConnection*();

**try** {

String sql = "insert into \"Customer\" values(?,?,?,?,?)"; //利用PowerDesigned生成的sql语句将表名用了""

pstmt = conn.prepareStatement(sql);

pstmt.setString(1,u.getCtype());

pstmt.setString(2,u.getCdata());

pstmt.setString(3,u.getCname());

pstmt.setString(4, u.getTel());

pstmt.setString(5, u.getPwd());

**int** result = pstmt.executeUpdate();

**return** result > 0;

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

**return** **false**;

} **finally** {

JdbcConn.*closeConnection*();

}

}

/\*

\* 执行更新操作

\* \*/

**public** **boolean** doUpdate(Customer u) {

conn = JdbcConn.*getConnection*();

**try** {

String sql = "update \"Customer\" set tel = ? where (ctype = ? and cdata = ?)";

pstmt = conn.prepareStatement(sql);

pstmt.setString(1, u.getTel());

pstmt.setString(2,u.getCtype());

pstmt.setString(3,u.getCdata());

**int** result = pstmt.executeUpdate();

**return** result > 0;

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

**return** **false**;

} **finally** {

JdbcConn.*closeConnection*();

}

}

/\*

\* 执行更新操作1

\* \*/

@SuppressWarnings("finally")

**public** **int** doUpdate1(Customer u) {

conn = JdbcConn.*getConnection*();

**try** {

String sql = "update \"Customer\" set ctype=?,cdata=?,tel=?,pwd=? where cname=? ";

pstmt = conn.prepareStatement(sql);

pstmt.setString(1, u.getCtype());

pstmt.setString(2, u.getCdata());

pstmt.setString(3, u.getTel());

pstmt.setString(4, u.getPwd());

pstmt.setString(5, u.getCname());

**int** result = pstmt.executeUpdate();

**return** result;

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

//e.printStackTrace();

System.***out***.println(e.getMessage());

**if**(e.getMessage().substring(0,9).equals("ORA-00001")) {

**return** -5 ;

} **else** **if**(e.getMessage().substring(0,9).equals("ORA-01407")) {

**return** -4 ;

} **else** {

**return** 0;

}

} **finally** {

JdbcConn.*closeConnection*();

}

}

/\*

\* 执行删除操作

\* \*/

**public** **boolean** doDelete(Customer u) {

conn = JdbcConn.*getConnection*();

**try** {

String sql = "delete from \"Customer\" where Ctype = ? and Cdata = ?";

pstmt = conn.prepareStatement(sql);

pstmt.setString(1,u.getCtype());

pstmt.setString(2,u.getCdata());

**int** result = pstmt.executeUpdate();

**return** result > 0;

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

**return** **false**;

} **finally** {

JdbcConn.*closeConnection*();

}

}

/\*

\* 执行查找操作(查找全部)

\* \*/

@SuppressWarnings({ "rawtypes", "unchecked" }) //消除警告

**public** Vector findAll() {

conn = JdbcConn.*getConnection*();

Vector rowData = **new** Vector();

**try** {

String sql = "select \* from \"Customer\" order by cdata";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

**while**(rs.next()) {

Vector row = **new** Vector();

row.add(rs.getString(1));

row.add(rs.getString(2));

row.add(rs.getString(3));

row.add(rs.getString(4));

row.add(rs.getString(5));

rowData.add(row);

}

**return** rowData;

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

**return** **null**;

} **finally** {

JdbcConn.*closeConnection*();

}

}

/\*

\* 执行查找操作(按姓名模糊查询)

\* \*/

@SuppressWarnings({ "rawtypes", "unchecked" })

**public** Vector findByName(String name) { //Vector类实现了一个可增长的对象数组。

conn = JdbcConn.*getConnection*();

Vector rowData = **new** Vector();

**try** {

String sql = "select \* from \"Customer\" where cname like ? ";

pstmt = conn.prepareStatement(sql);

pstmt.setString(1, "%"+name+"%");

rs = pstmt.executeQuery();

**while**(rs.next()) {

Vector row = **new** Vector();

row.add(rs.getString(1));

row.add(rs.getString(2));

row.add(rs.getString(3));

row.add(rs.getString(4));

row.add(rs.getString(5));

rowData.add(row);

}

**return** rowData;

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

**return** **null**;

} **finally** {

JdbcConn.*closeConnection*();

}

}

/\*

\* 执行查找操作(按用户证件类型及证件号和用户密码精确查询)

\* \*/

**public** Customer findById(String ctype, String cdata,String pwd) {

conn = JdbcConn.*getConnection*();

Customer u = **null**;

**try** {

String sql = "select \* from \"Customer\" where ctype = ? and cdata = ? and pwd = ? ";

pstmt = conn.prepareStatement(sql);

pstmt.setString(1, ctype);

pstmt.setString(2, cdata);

pstmt.setString(3, pwd);

rs = pstmt.executeQuery();

**if**(rs.next()) {

u = **new** Customer(rs.getString(1),rs.getString(2),rs.getString(3),rs.getString(4),rs.getString(5));

}

**return** u;

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

**return** **null**;

} **finally** {

JdbcConn.*closeConnection*();

}

}

/\*

\* 得到表的列名

\* \*/

@SuppressWarnings({ "rawtypes", "unchecked" })

**public** Vector findColumnNames() {

conn = JdbcConn.*getConnection*();

Vector columnNames = **new** Vector();

**try** {

String sql = "select ctype 证件类型,cdata 证件号码,cname 用户名,tel 联系方式, pwd 用户密码 from \"Customer\"";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

rsmd = rs.getMetaData();//取出源数据

**int** colCount = rsmd.getColumnCount();//得到列的个数

**for**(**int** i = 1;i <= colCount;i++) {

columnNames.add(rsmd.getColumnName(i));

}

**return** columnNames;

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

**return** **null**;

} **finally** {

JdbcConn.*closeConnection*();

}

}

}

**TicketDao.java**

package javadao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.ResultSetMetaData;

import java.sql.SQLException;

import java.util.Vector;

import classes.Customer;

import classes.Ticket;

import jdbcutils.JdbcConn;

public class TicketDao {

private Connection conn;

private PreparedStatement pstmt; //执行数据库操作

private ResultSet rs; //存放数据库返回的结果

private ResultSetMetaData rsmd; //存放数据库中的源数据

/\*

\* 执行插入操作

\* \*/

public int doInsert(Ticket u) {

conn = JdbcConn.getConnection();

try {

String sql = "insert into ticket values(ticket\_seq.nextval,?,?,?,?,?,?,?,?)";

pstmt = conn.prepareStatement(sql);

pstmt.setString(1,u.getPid());

pstmt.setString(2,u.getCtype());

pstmt.setString(3, u.getCdata());

pstmt.setInt(4, u.getPrice());

pstmt.setInt(5, u.getDiscount());

pstmt.setString(6, u.getPlevel());

pstmt.setString(7, u.getTmanager());

pstmt.setString(8, u.getCstate());

int result = pstmt.executeUpdate();

return result;

} catch (SQLException e) {

// TODO Auto-generated catch block

//e.printStackTrace();

if(e.getMessage().substring(0, 9).equals("ORA-00001")) {

return -2;

} else {

return 0;

}

} finally {

JdbcConn.closeConnection();

}

}

/\*

\* 执行查找操作(查找全部)

\* \*/

@SuppressWarnings({ "unchecked", "rawtypes" })

public Vector findAll() {

conn = JdbcConn.getConnection();

Vector rowData = new Vector();

try {

String sql = "select customer.cname,ticket.\* from ticket join customer on (ticket.CDATA=customer.CDATA and ticket.CTYPE=customer.CTYPE)";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

while(rs.next()) {

Vector row = new Vector();

row.add(rs.getString(1));

row.add(rs.getInt(2));

row.add(rs.getString(3));

row.add(rs.getString(4));

row.add(rs.getString(5));

row.add(rs.getInt(6));

row.add(rs.getInt(7));

row.add(rs.getString(8));

row.add(rs.getString(9));

row.add(rs.getString(10));

rowData.add(row);

}

return rowData;

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

return null;

} finally {

JdbcConn.closeConnection();

}

}

/\*

\* 执行删除操作

\* \*/

public boolean doDelete(Ticket u) {

conn = JdbcConn.getConnection();

try {

String sql = "delete from Ticket where tid=?";

pstmt = conn.prepareStatement(sql);

pstmt.setInt(1,u.getTid());

int result = pstmt.executeUpdate();

return result > 0;

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

return false;

} finally {

JdbcConn.closeConnection();

}

}

/\*

\* 执行查找操作(查找未付款的机票)

\* \*/

@SuppressWarnings({ "unchecked", "rawtypes" })

public Vector findTicket(String state) {

conn = JdbcConn.getConnection();

Vector rowData = new Vector();

try {

String sql = "select customer.cname,ticket.\* from ticket join customer on (ticket.CDATA=customer.CDATA and ticket.CTYPE=customer.CTYPE) where ticket.cstate=?";

pstmt = conn.prepareStatement(sql);

pstmt.setString(1, state);

rs = pstmt.executeQuery();

while(rs.next()) {

Vector row = new Vector();

row.add(rs.getString(1));

row.add(rs.getInt(2));

row.add(rs.getString(3));

row.add(rs.getString(4));

row.add(rs.getString(5));

row.add(rs.getInt(6));

row.add(rs.getInt(7));

row.add(rs.getString(8));

row.add(rs.getString(9));

row.add(rs.getString(10));

rowData.add(row);

}

return rowData;

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

return null;

} finally {

JdbcConn.closeConnection();

}

}

/\*

\* 执行查找操作(查找机票)

\* \*/

@SuppressWarnings({ "unchecked", "rawtypes" })

public Ticket findTicket(int id) {

conn = JdbcConn.getConnection();

Ticket u = new Ticket();

try {

String sql = "select \* from ticket where tid = ?";

pstmt = conn.prepareStatement(sql);

pstmt.setInt(1, id);

rs = pstmt.executeQuery();

if(rs.next()) {

u = new Ticket(rs.getInt(1),rs.getString(2),rs.getString(3),rs.getString(4),rs.getInt(5),rs.getInt(6),rs.getString(7),rs.getString(8),rs.getString(9));

}

return u;

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

return null;

} finally {

JdbcConn.closeConnection();

}

}

/\*

\* 得到表的列名

\* \*/

public Vector findColumnNames() {

conn = JdbcConn.getConnection();

Vector columnNames = new Vector();

try {

String sql = "select customer.cname 用户名,ticket.tid 机票序号,ticket.pid 航班编号,ticket.ctype 证件类型,ticket.cdata 证件号码,ticket.price 票价,ticket.discount 折扣,ticket.plevel 机舱等级,ticket.tmanager 经手业务员,ticket.cstate 付款情况 from ticket,customer";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

rsmd = rs.getMetaData();//取出源数据

int colCount = rsmd.getColumnCount();//得到列的个数

for(int i = 1;i <= colCount;i++) {

columnNames.add(rsmd.getColumnName(i));

}

return columnNames;

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

return null;

} finally {

JdbcConn.closeConnection();

}

}

/\*

\* 执行更新操作

\* \*/

public boolean doUpdate(String state,int id) {

conn = JdbcConn.getConnection();

try {

String sql = "update ticket set cstate = ? where tid = ?";

pstmt = conn.prepareStatement(sql);

pstmt.setString(1, state);

pstmt.setInt(2, id);

int result = pstmt.executeUpdate();

return result > 0;

} catch (SQLException e) {

// TODO Auto-generated catch block

//e.printStackTrace();

return false;

} finally {

JdbcConn.closeConnection();

}

}

}

**FlightDao.java**

package javadao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.ResultSetMetaData;

import java.sql.SQLException;

import java.util.Vector;

import classes.Flight;

import jdbcutils.JdbcConn;

public class FlightDao {

private Connection conn;

private PreparedStatement pstmt; //执行数据库操作

private ResultSet rs; //存放数据库返回的结果

private ResultSetMetaData rsmd; //存放数据库中的源数据

@SuppressWarnings({ "rawtypes", "unchecked" })

public Vector findColumnNames() {

conn = JdbcConn.getConnection();

Vector columnNames = new Vector();

try {

String sql = "select company.mname 航空公司名称,plane.pid 航班号,plane.mcode 航空公司代码,plane.rname 航线名称 ,plane.pname 飞机名称,plane.ptype 飞机类型,plane.tstate 当前预售情况 from plane,company";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

rsmd = rs.getMetaData();//取出源数据

int colCount = rsmd.getColumnCount();//得到列的个数

for(int i = 1;i <= colCount;i++) {

columnNames.add(rsmd.getColumnName(i));

}

return columnNames;

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

return null;

} finally {

JdbcConn.closeConnection();

}

}

/\*

\* 执行查找操作(查找全部)

\* \*/

@SuppressWarnings({ "rawtypes", "unchecked" })

public Vector findAll() {

conn = JdbcConn.getConnection();

Vector rowData = new Vector();

try {

String sql = "select company.mname, plane.\* from plane join company on plane.mcode=company.mcode";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

while(rs.next()) {

Vector row = new Vector();

row.add(rs.getString(1));

row.add(rs.getString(2));

row.add(rs.getString(3));

row.add(rs.getString(4));

row.add(rs.getString(5));

row.add(rs.getString(6));

row.add(rs.getString(7));

rowData.add(row);

}

return rowData;

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

return null;

} finally {

JdbcConn.closeConnection();

}

}

/\*

\* 执行查找操作(按航班号精确查询)

\* \*/

public Flight findById(String id) {

conn = JdbcConn.getConnection();

Flight u = null;

try {

String sql = "select \* from plane where pid = ? ";

pstmt = conn.prepareStatement(sql);

pstmt.setString(1, id);

rs = pstmt.executeQuery();

if(rs.next()) {

u = new Flight(rs.getString(2),rs.getString(1),rs.getString(3),rs.getString(4),rs.getString(5),rs.getString(6));

}

return u;

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

return null;

} finally {

JdbcConn.closeConnection();

}

}

/\*

\* 执行查找操作(按航线精确查询)

\* \*/

public Vector findByRoute(String route) {

conn = JdbcConn.getConnection();

Vector rowData = new Vector();

try {

String sql = "select company.mname, plane.\* from plane join company on plane.mcode=company.mcode where rname = ? ";

pstmt = conn.prepareStatement(sql);

pstmt.setString(1, route);

rs = pstmt.executeQuery();

while(rs.next()) {

Vector row = new Vector();

row.add(rs.getString(1));

row.add(rs.getString(2));

row.add(rs.getString(3));

row.add(rs.getString(4));

row.add(rs.getString(5));

row.add(rs.getString(6));

row.add(rs.getString(7));

rowData.add(row);

}

return rowData;

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

return null;

} finally {

JdbcConn.closeConnection();

}

}

}

**AirlaneDao.java**

package javadao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.ResultSetMetaData;

import java.sql.SQLException;

import java.util.Vector;

import classes.Airlane;

import classes.Flight;

import jdbcutils.JdbcConn;

public class AirlaneDao {

private Connection conn;

private PreparedStatement pstmt; //执行数据库操作

private ResultSet rs; //存放数据库返回的结果

private ResultSetMetaData rsmd; //存放数据库中的源数据

/\*

\* 得到表的列名

\* \*/

@SuppressWarnings({ "rawtypes", "unchecked" })

public Vector findColumnNames() {

conn = JdbcConn.getConnection();

Vector columnNames = new Vector();

try {

String sql = "select rname 航线名称,rtype 航线类型,rlength 航线长度 from route";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

rsmd = rs.getMetaData();//取出源数据

int colCount = rsmd.getColumnCount();//得到列的个数

for(int i = 1;i <= colCount;i++) {

columnNames.add(rsmd.getColumnName(i));

}

return columnNames;

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

return null;

} finally {

JdbcConn.closeConnection();

}

}

/\*

\* 执行查找操作(查找全部)

\* \*/

@SuppressWarnings({ "rawtypes", "unchecked" })

public Vector findAll() {

conn = JdbcConn.getConnection();

Vector rowData = new Vector();

try {

String sql = "select \* from route";

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

while(rs.next()) {

Vector row = new Vector();

row.add(rs.getString(1));

row.add(rs.getString(2));

row.add(rs.getInt(3));

rowData.add(row);

}

return rowData;

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

return null;

} finally {

JdbcConn.closeConnection();

}

}

/\*

\* 执行查找操作(按航班号精确查询)

\* \*/

public Airlane findByRoute(String id) {

conn = JdbcConn.getConnection();

Airlane u = null;

try {

String sql = "select \* from route where rname = ? ";

pstmt = conn.prepareStatement(sql);

pstmt.setString(1, id);

rs = pstmt.executeQuery();

if(rs.next()) {

u = new Airlane(rs.getString(1),rs.getString(2),rs.getInt(3));

}

return u;

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

return null;

} finally {

JdbcConn.closeConnection();

}

}

}

**AllTicketFrame.java**

package frame;

import java.awt.BorderLayout;

import java.awt.EventQueue;

import java.util.Vector;

import javax.swing.JFrame;

import javax.swing.JPanel;

import javax.swing.border.EmptyBorder;

import classes.Customer;

import classes.Ticket;

import javadao.TicketDao;

import javax.swing.JScrollPane;

import javax.swing.JTable;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JTextField;

import javax.swing.JButton;

import java.awt.event.ActionListener;

import java.awt.event.ActionEvent;

public class AllTicketFrame extends JFrame {

private static Vector columns,rowData;

public static Customer u;

private JPanel contentPane;

private JTable table;

private JTextField tTid;

/\*\*

\* Create the frame.

\*/

public AllTicketFrame(Vector columns,Vector rowData,Customer u) {

this.columns = columns;

this.rowData = rowData;

this.u = u;

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setBounds(800, 300, 751, 450);

contentPane = new JPanel();

contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));

setContentPane(contentPane);

contentPane.setLayout(null);

JScrollPane scrollPane = new JScrollPane();

scrollPane.setBounds(14, 13, 705, 286);

contentPane.add(scrollPane);

table = new JTable(rowData,columns);

scrollPane.setViewportView(table);

JLabel lblNewLabel = new JLabel("\u8BF7\u8F93\u5165\u673A\u7968\u5E8F\u53F7");

lblNewLabel.setBounds(37, 345, 113, 18);

contentPane.add(lblNewLabel);

tTid = new JTextField();

tTid.setBounds(179, 342, 86, 24);

contentPane.add(tTid);

tTid.setColumns(10);

JButton btnBuy = new JButton("\u4ED8\u6B3E");

btnBuy.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if(!tTid.getText().trim().isEmpty()) {

TicketDao tdao = new TicketDao();

if(tdao.doUpdate("已付款", Integer.valueOf(tTid.getText()))) {

JOptionPane.showMessageDialog(getContentPane(), "付款成功");

AllTicketFrame.rowData = tdao.findAll();

table = new JTable(AllTicketFrame.rowData,columns);

scrollPane.setViewportView(table);

} else {

JOptionPane.showMessageDialog(getContentPane(), "付款失败");

}

}

}

});

btnBuy.setBounds(316, 341, 113, 27);

contentPane.add(btnBuy);

JButton btnDelete = new JButton("\u5220\u9664");

btnDelete.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if(!tTid.getText().trim().isEmpty()) {

TicketDao tdao = new TicketDao();

Ticket t = tdao.findTicket(Integer.valueOf(tTid.getText()));

if(t!=null) {

if(tdao.doDelete(t)) {

JOptionPane.showMessageDialog(getContentPane(), "删除成功");

AllTicketFrame.rowData = tdao.findAll();

table = new JTable(AllTicketFrame.rowData,columns);

scrollPane.setViewportView(table);

} else {

JOptionPane.showMessageDialog(getContentPane(), "删除失败");

}

}

}

}

});

btnDelete.setBounds(463, 341, 113, 27);

contentPane.add(btnDelete);

JButton btnBack = new JButton("\u8FD4\u56DE\u4E3B\u83DC\u5355");

btnBack.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

MainFrame mainFrame = new MainFrame(u,u.getName());

mainFrame.setVisible(true);

setVisible(false);

}

});

btnBack.setBounds(606, 341, 113, 27);

contentPane.add(btnBack);

}

}