软件学院

《Java程序设计》

课程设计报告

**题目：**

**班级：**

**姓名：**

**学号：**

**2016年1月7日**

# **摘 要**

随着信息技术的发展，计算机已被广泛的应用于社会的各个领域，成为推动社会发展的技术动力。而在计算机应用中，软件的作用十分突出，软件已经发展成为信息技术的核心，主导着信息产品的开发和信息技术市场的进一步的开拓。软件产业已成为社会信息化进程中的一个战略性产业。在软件技术的应用中软件的开发技术尤其是应用型软件产品的开发技术成了重中之重。不断开发适应用户需求、市场需要的新型软件产品。

我们本次的Java课程设计的最终目标是实现一个酒店商管理系统，使其具有一般酒店管理系统所具有的功能，可以实现管理员日志、管理员添加、管理员信息修改、商品信息查询、商品信息修改、商品添加、商品删除、商品分配等功能。因为Java是一门很优秀的编程语言，具有面向对象、与平台无关、安全、稳定和多线程等特点，所以用Java编程为我们本次程序的编写提供了方便。在设计中我们主要进行了数据库及类的设计，把整系统分成了18个类和3个表。通过应用awt、swing、sql、util等包，合理的添加了按钮、菜单、文本框、文本区、表格等组件类，还包含窗口、面板等容器类，经过设计者合理的布局，完成了窗体的设计，并通过建立JTDS桥接器连接到数据库后，向数据库发送SQL语句实现程序与数据库中表间的消息通信，从而实现了相应的酒店管理系统的功能。

本文主要分为绪论、系统功能介绍、数据库需求分析、系统功能实现、结论、致谢、参考文献7大块。在绪论中介绍了本次课设选题的缘由及思想。系统功能介绍中详细介绍了本图书管理系统的整体思想及其具体功能。在数据库需求分析中详细说明了数据库的需求分析、概念设计及表的设计。在系统功能实现中主要通过抓图来体现设计的结果。总之，我们的整体设计思想均在报告中体现。

关键词： 酒店商管理系统，Java程序设计语言，数据库

# **目 录**

[**摘 要** II](#_Toc439951785)

[**目 录** III](#_Toc439951786)

[**第1章 绪 论** 4](#_Toc439951787)

[1.1 研究背景和意义 4](#_Toc439951788)

[1.1.1 本课题研究背景 4](#_Toc439951789)

[1.1.2 本课题研究意义 5](#_Toc439951790)

[**第2章 系统功能介绍** 6](#_Toc439951791)

[**2.1系统功能总框图** 6](#_Toc439951792)

[**2.2 模块功能介绍** 6](#_Toc439951793)

[2.2.1 登陆模块功能介绍 6](#_Toc439951794)

[2.2.2 前台管理模块功能介绍 6](#_Toc439951795)

[2.2.3 后台管理模块功能介绍 7](#_Toc439951796)

[2.2.4材料管理模块功能介绍 7](#_Toc439951797)

[2.2.5 系统安全模块功能介绍 7](#_Toc439951798)

[**第3章 数据库设计** 9](#_Toc439951799)

[**3.1 数据库需求分析** 9](#_Toc439951800)

[**3.2 数据库概念结构设计** 9](#_Toc439951801)

[**3.3数据库表的设计** 10](#_Toc439951802)

[**第4章 系统功能实现** 12](#_Toc439951803)

[**4.1 管理员模块功能实现** 12](#_Toc439951804)

[4.1.1 管理员登录 13](#_Toc439951805)

[**4.2 管理人员模块功能实现** 13](#_Toc439951806)

[4.2.1 管理日志 13](#_Toc439951807)

[4.2.2 管理员添加 14](#_Toc439951808)

[4.2.3 管理员信息修改 17](#_Toc439951809)

[4.2.4 管理员ID查询 19](#_Toc439951810)

[4.2.5 管理员组合查询 22](#_Toc439951811)

[4.2.6 代理商信息查询 25](#_Toc439951812)

[**4.3 材料管理模块功能实现** 27](#_Toc439951813)

[4.3.1 材料信息查询 27](#_Toc439951814)

[4.3.2 材料添加 31](#_Toc439951815)

[4.3.3 材料信息修改 33](#_Toc439951816)

[4.3.4 材料删除 35](#_Toc439951817)

[**4.4 功能列表模块功能实现** 36](#_Toc439951818)

[4.4.1 主面板 36](#_Toc439951819)

[4.4.2 台号管理 39](#_Toc439951820)

[4.4.3 菜系管理 40](#_Toc439951821)

[4.4.4 菜品管理 41](#_Toc439951822)

[**结 论** 44](#_Toc439951823)

[**致 谢** 45](#_Toc439951824)

# **第1章 绪 论**

## 1.1 研究背景和意义

### 1.1.1 本课题研究背景

随着企业的管理意识在不断增强，管理手段也在不断更新。现代化管理中不可或缺的就是信息化集成管理系统，它是依靠计算机为手段，通过网络来连接各单位各部门。企业信息管理系统它不是结果，而是企业管理中的过程控制。它可以实现信息共享，数据提炼，让所有工作繁琐的问题交给计算机。

企业管理信息系统的首要任务就是让企业能稳定的在一定环境下工作，所以

稳定问题是关键，怎么稳定呢，体现在软件实用性上。软件在使用过程中符合我们酒店管理流程，符合使用人风格界面。人性化是系统存在的根本。

目前来说，国内的酒店管理信息系统都相应的存在这样那样的问题。管理水平不够，才是最关键的问题。我们国内做的系统往往追求软体界面，操作性及硬件配置条件。忽略了开发价值等问题。即使开发出来的系统都是完全按照酒店行业流程，技术先进性也许也会忽略。所以现阶段来说我们开发的系统都是用流行的开发模式，较为理想开发环境，有着强大的硬件支持。光是先进开发理念还是不够的所以我们选择面向用户，调研工作是比较庞大的，根据顾客服务理念是第一的，最终我们开发的系统运行在各个酒店，所以我们就必须从调研开始搞清楚酒店到底需要什么，我们能给酒店带来什么。

21 世纪，全球网络化，科技在突飞猛进。计算机应经普及到我们社会和经济电子科技大学硕士学位论文生活中的各个领域。人们进行信息交流的深度与广度不断增加，这使得我们的服务、管理也相应该跟得上时代的步伐去前进。电子商务在此得到极大地普及与发展。

酒店客房业在电子商务中还处于形成的初期，传统的酒店客房业中，大都是以实际门面的形式来进行宣传和交流的。作为网络普及化的社会，这还缺少了一些更为现代化的元素。酒店客房业是一个个性化、多样化的服务业，而电子商务择最能体现个性化、多样化服务的商务方式。而且现在大部分电子商务都体现为向大企业服务，结合酒店客房这个行业的特点，最终还发展为同小区小店服务的模式。以最贴切的方式，最灵便的服务进入个个网络社区，进入人们的生活。

针对大背景下的问题与方向，可以察觉到在酒店客房服务这个大行业中尚有可为的地方。提供适合不同人群，最贴切的服务，通过网络平台，去体现服务质量，去创造效益。

应当抓住中小店铺，提供区域化酒店客房服务这一理念，为他们提供最贴切的服务，把它体现在电子商务服务上来，融合到我们软件开发的产品中，去建立起特色鲜明的酒店客房预订网站，来获得最大的经济效益。

综上所述，本论文研究并实现一套酒店管理系统是符合现在的实际情况的选择，通过研究并吸取国内外相关经验教训来进行开发具有一定的必要。

### 1.1.2 本课题研究意义

酒店的前台管理、楼层控制、总台管理构成了整个酒店的业务体系。在传统行业影响下我们都采取人工化的信息交换这样的方式如何能满足业务不断增长的当今酒店业呢

在酒店业铺天盖地的发展着，竞争也随着覆盖率增加也越演越烈。一家大中型酒店的服务质量及管理水平是它能生存及发展的必要条件。人工干预导致工作的混乱及错误，导致客户满意度的下滑。最终导致业绩下滑甚至于威胁到酒店生存。在这样的环境下，酒店业就被迫寻求其发展，信息化建设这样才能提升酒店的竞争力。这就需要我们用计算机来实现酒店的管理理念。

酒店管理信息系统在能对酒店决策者们带来新的机遇和挑战，因为数据的使用是价值的体现，不断发展的网络技术是平台。因此以顾客服务为中心的管理模式就在计算机网络中过程控制中体现出来。

# **第2章 系统功能介绍**

## **2.1系统功能总框图**

酒店管理系统功能总框图，如下图所示。

酒店管理系统

系统安全

前台服务

后台管理

登录

材料管理

用户管理

修改密码

开台点菜

智能化获取菜品

自动结账

菜系管理

台号管理

菜品管理

编号查询

材料添加

材料统计

## **2.2 模块功能介绍**

### 2.2.1 登陆模块功能介绍

此模块主要是保护系统数据，不被管理员和代理商之外的人改动，成功登陆后可以实现进入各个模块如后台管理、材料管理、系统安排以及退出的功能。

### 2.2.2 前台管理模块功能介绍

1.开台点菜

此模块可以清楚的显示出用户点菜的信息，包括菜品编号，菜品助记码，菜品名，菜品数量，操作日期，总额。管理人员可以通过此模块实现的功能清楚了解用户点菜的详细信息，从而给管理人员带来了很多方便。

2. 智能化获取菜品

此模块功能是通过自动终端获取菜品信息。可以通过菜品的编号，菜品助记码或菜品名称查询菜品状况。

3.自动结账

用户只需要选中结账的台号，系统就会自动为选中的台号计算消费金额，并且用户输入实收金额后，系统还会自动计算出需要找零的金额。

### 2.2.3 后台管理模块功能介绍

1.台号管理

此模块主要的实现功能是添加新的桌位信息，即当有新的桌位被占用时，管理员可以通过此功能实现添加台号信息。其中信息分别包括序号、台号、座位数，同时可有两项功能在此处实现，即添加以及删除。

2.菜系管理

此模块可以选择用户所需食物所属菜系。同时可有两项功能在此处实现，即添加以及删除。如点击添加则出现相应菜系的菜品。其中信息分别包括序号、菜系编号、菜系名称

3.菜品管理

此模块主要的实现功能是用户进行菜品的选择。显示列表信息有序号，编号，名称，助记码，菜系，单位，单价。同时可有两项功能在此处实现，即添加以及删除。添加是信息有编号，菜品名称，单位（盘），助记码，菜系，单价。

### 2.2.4材料管理模块功能介绍

1.材料统计

此模块主要的实现功能是对酒店所用食材的统计。通过全查进行获得，如编号，名称，数量，单价，消费金额。并且还可以通过点击表格进行删除。

2.材料添加

此模块主要的实现功能是对酒店所用食材的添加。在相应文本区域写入信息，点击添加按钮即可完成。

3.编号查询

此模块主要的实现功能是对所余素材的查询。输入食材编号即可查询，便于酒店进货处理。

### 2.2.5 系统安全模块功能介绍

1.修改密码

此模块主要用于管理员对用户账户密码和用户对登录密码的修改及系统密码的修改。

2.系统安全

此模块主要用于维护系统运行稳定、安全可靠以及系统的管理，例如系统密码。

# **第3章 数据库设计**

在此酒店管理系统中，使用到了一个重要的连接，即与数据库 s相连。数据库中存有酒店中所有信息，包括用户信息、管理员添加、删除与修改记录。所有与管理有关的数据皆在其中，是保证系统能够正常实现各种功能的一架桥。

## **3.1 数据库需求分析**

⒈ tb\_sort表，即菜系表，记录了菜系编号、菜系名称。

⒉ tb\_user表，即用户表，记录了用户号、用户名、密码、性别、出生日期、id卡、消费日期。

⒊ tb\_menu表，即菜品表，记录了菜品名、菜品编号、数量、助记码、单价、单位。

4．tb\_buy表，即消费项目表，记录了序号、菜品名、菜品编号、数量、助记码、单价、单位、总价。

5．tb\_deskuse表，即消费单表，记录了编号、台号、座位数。

6. tb\_desk表，即台号表，记录了台号、座位数。

## **3.2 数据库概念结构设计**

酒店管理系统E-R图，如下图所示。

n

用户

管理

n

m

n

m

安排

管理员

点取

餐台

m

m

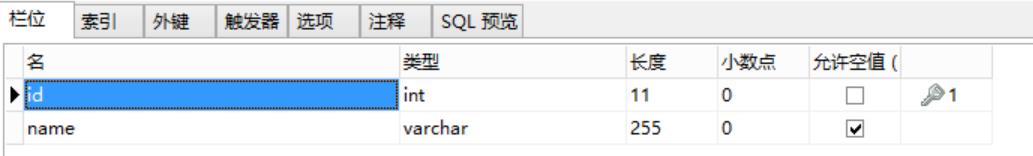
n

记录

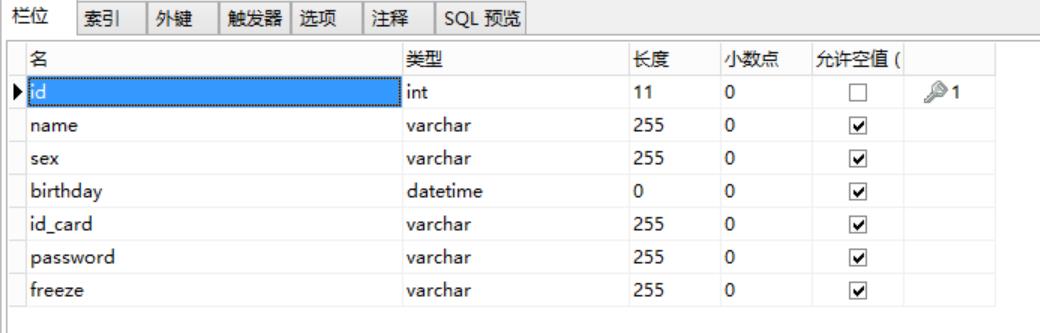
菜品

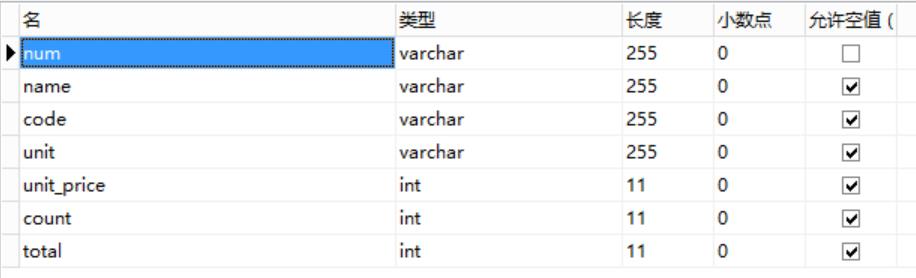
## **3.3数据库表的设计**

1. tb\_sort表中记录菜系信息便于以后可以查找。如下图所示

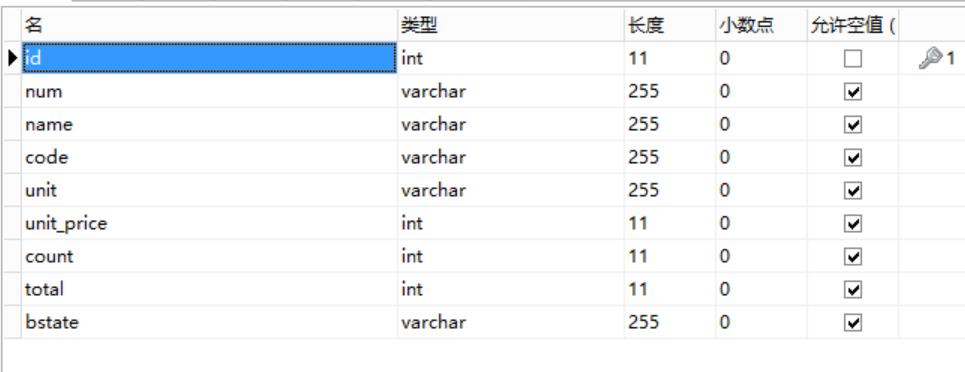


⒉ tb\_user表中记录了可以使管理员和代理商进入该系统的记录了用户号、用户名、密码、性别、出生日期、id卡、消费日期。如下图所示。



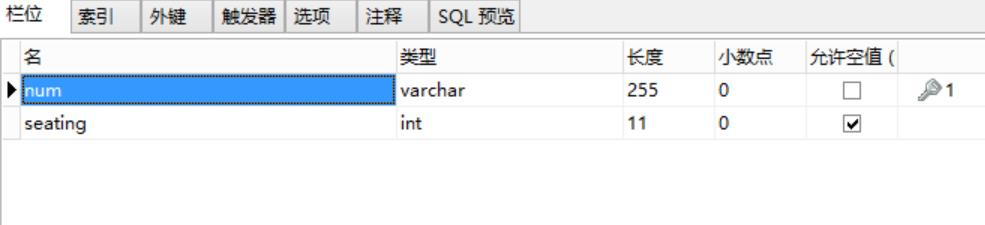
⒊ tb\_menu表中记录了菜品信息，如下图所示。

4．tb\_buy表中记录了消费单信息



5．tb\_deskuse表中记录了消费项目信息

6．tb\_desk表中记录了台号信息



# **第4章 系统功能实现**

## **4.1 管理员模块功能实现**

此模块是整个系统最主要的部分。登录界面如下图。



主要代码如下：

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("正在执行登录操作");

udi = new UserDAOImp();

String uname = jTextField1.getText().trim();

char ch[] = jPasswordField1.getPassword();

String password = new String(ch);

System.out.println("gui中输入的用户名" + uname + "\t密码是" + password);

if (udi.isExist(uname, password)) {

System.out.println("登录成功！");

this.setVisible(false);

MainJFrame ij = new MainJFrame();

ij.setVisible(true);

} else {

System.out.println("用户不存在请重新输入");

jTextField1.setText("");

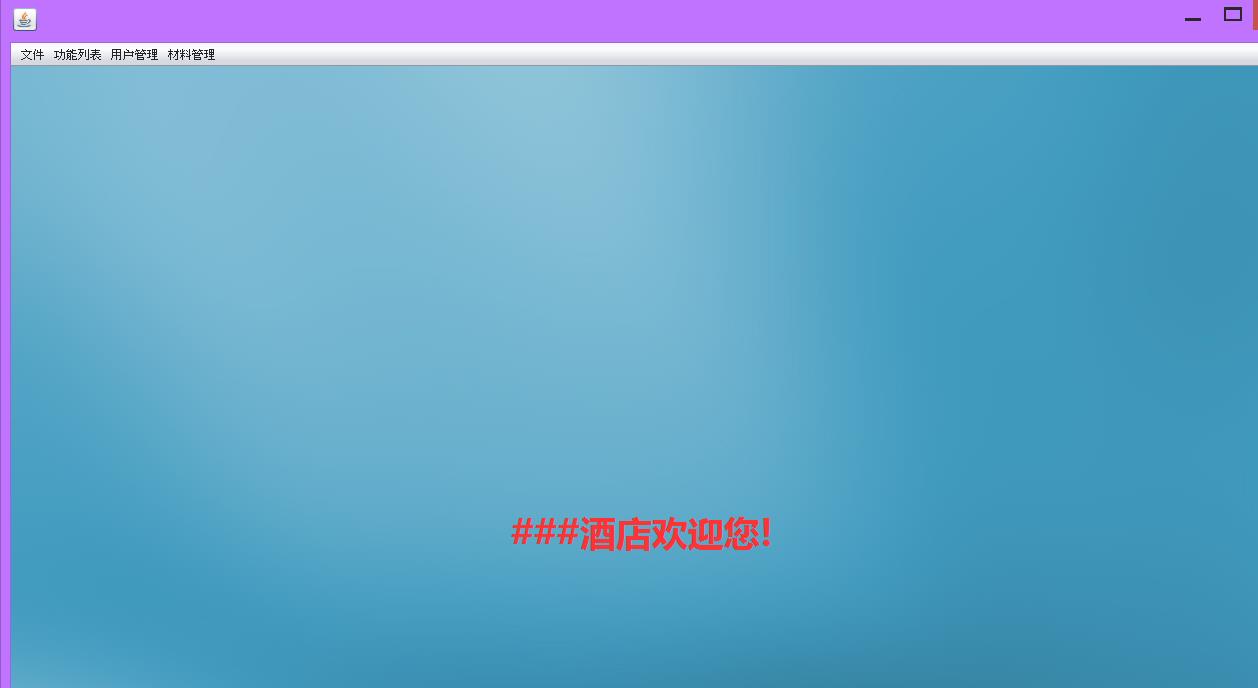
jPasswordField1.setText("");

}

}

### 4.1.1 管理员登录

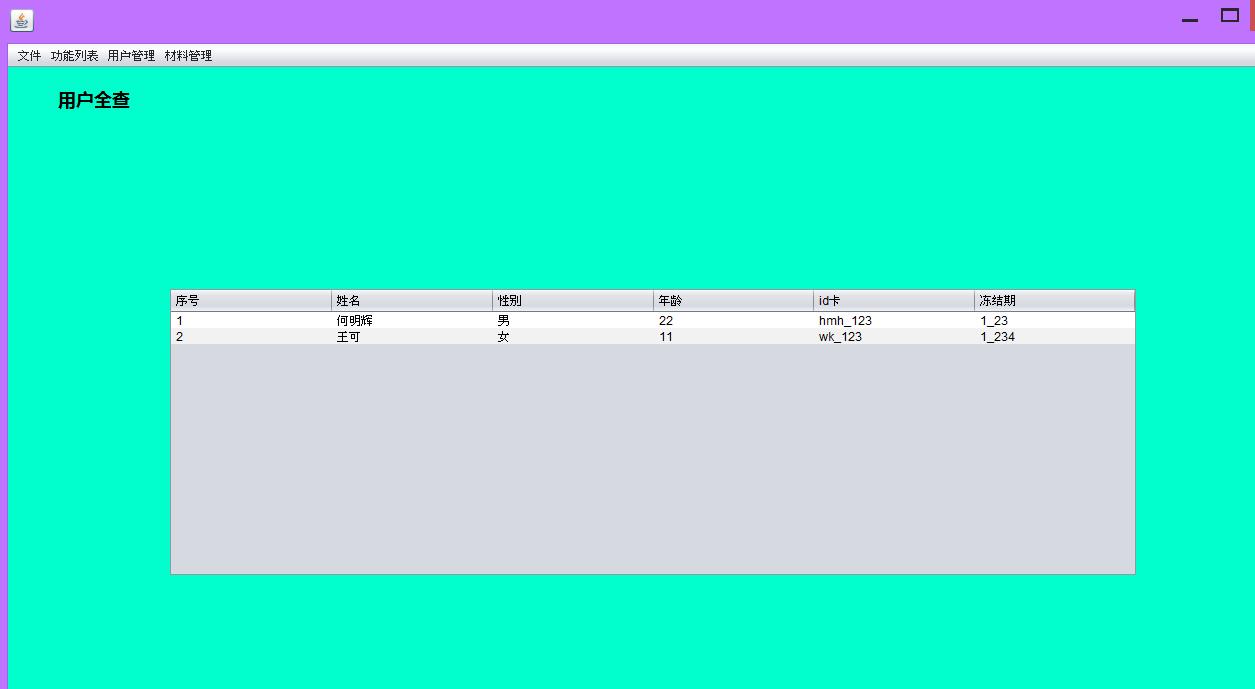
管理员登录后界面如下图



## **4.2 管理人员模块功能实现**

### [4.2.1](#_Toc233984926) 用户查询

此模块通过顶层容器JFrame，再添加普通容器 JScrollPane、Jpanel，然后通过绝对布局创建界面，最后主要通过list利用Iterator遍历在连接好的数据库中查找表tb\_user中的内容，从而实现了可以清楚的显示出商品分配的功能。管理日志界面如下图所示。

v

主要代码如下：

public List<User> selectAll() {

List<User> list = new ArrayList();

try {

con = db.getCon();

if (con == null) {

System.out.print("数据库连接失败");

}

stmt = con.createStatement();

String sql = "select \* from tb\_user";

rs = stmt.executeQuery(sql);

while (rs.next()) {

System.out.println("有一行新纪录"); //uid

int uid = rs.getInt(1);

String uname = rs.getString(2);

String sex = rs.getString(3);

int age = rs.getInt(4);

String id\_card = rs.getString(5);

String password = rs.getString(6);

String freeze = rs.getString(7);

User u = new User();

u.setId(uid);

u.setName(uname);

u.setSex(sex);

u.setAge(age);

u.setId\_card(id\_card);

u.setPassword(password);

u.setFreeze(freeze);

list.add(u);

}

} catch (SQLException e) {

e.printStackTrace();

}

return list;

}

### [4.2.2 用户添加](#_Toc233984926)

该功能主要实现管理员的添加，以便于让实行对管理员的管理。按照标签提示输入信息，然后点击添加按钮。管理员添加窗口如下图所示。



主要代码如下：

public boolean insert(User u) {

boolean flag = true;

try {

con = db.getCon();

if (con == null) {

System.out.print("数据库连接失败");

}

//pstmt

String sql = "insert into tb\_user(name,sex,age,id\_card,password,freeze) values(?,?,?,?,?,?)";

pstmt = con.prepareStatement(sql);

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

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

pstmt.setInt(3, u.getAge());

pstmt.setString(4, u.getId\_card());

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

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

int state = pstmt.executeUpdate();

if (state > 0) {

System.out.print("dao中插入用户成功");

flag = true;

}

} catch (SQLException e) {

e.printStackTrace();

}

return flag;

}

private void jButton26ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("gui正在执行添加用户操作");

udi = new UserDAOImp();

//1 获取组件的信息 并进行类型转换

String uname = jTextField20.getText().trim();

char ch[] = jPasswordField1.getPassword();

String password = new String(ch);

String sex = "";

if (jRadioButton1.isSelected()) {

sex = "男";

}

if (jRadioButton2.isSelected()) {

sex = "女";

}

int age = 0;

try {

String agestr = jTextField21.getText(); //字符串

age = Integer.parseInt(agestr);

} catch (NumberFormatException e) {

e.printStackTrace();

}

String idk = jTextField22.getText().trim();

String freeze = jTextField23.getText().trim();

//如果属性超过三个，建议使用对象来封装传递数据

User u = new User();

u.setName(uname);

u.setAge(age);

u.setPassword(password);

u.setSex(sex);

u.setId\_card(idk);

u.setFreeze(freeze);

//2、调用数据访问层方法

boolean result = udi.insert(u);

if (result) {

System.out.println("gui添加用户成功");

this.setVisible(false);

//系统主界面

} else {

System.out.println("gui添加用户失败");

}

private void jButton27ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("gui正在执行添加重置用户操作");

jTextField21.setText("");

jTextField20.setText("");

buttonGroup1.clearSelection();

jPasswordField1.setText("");

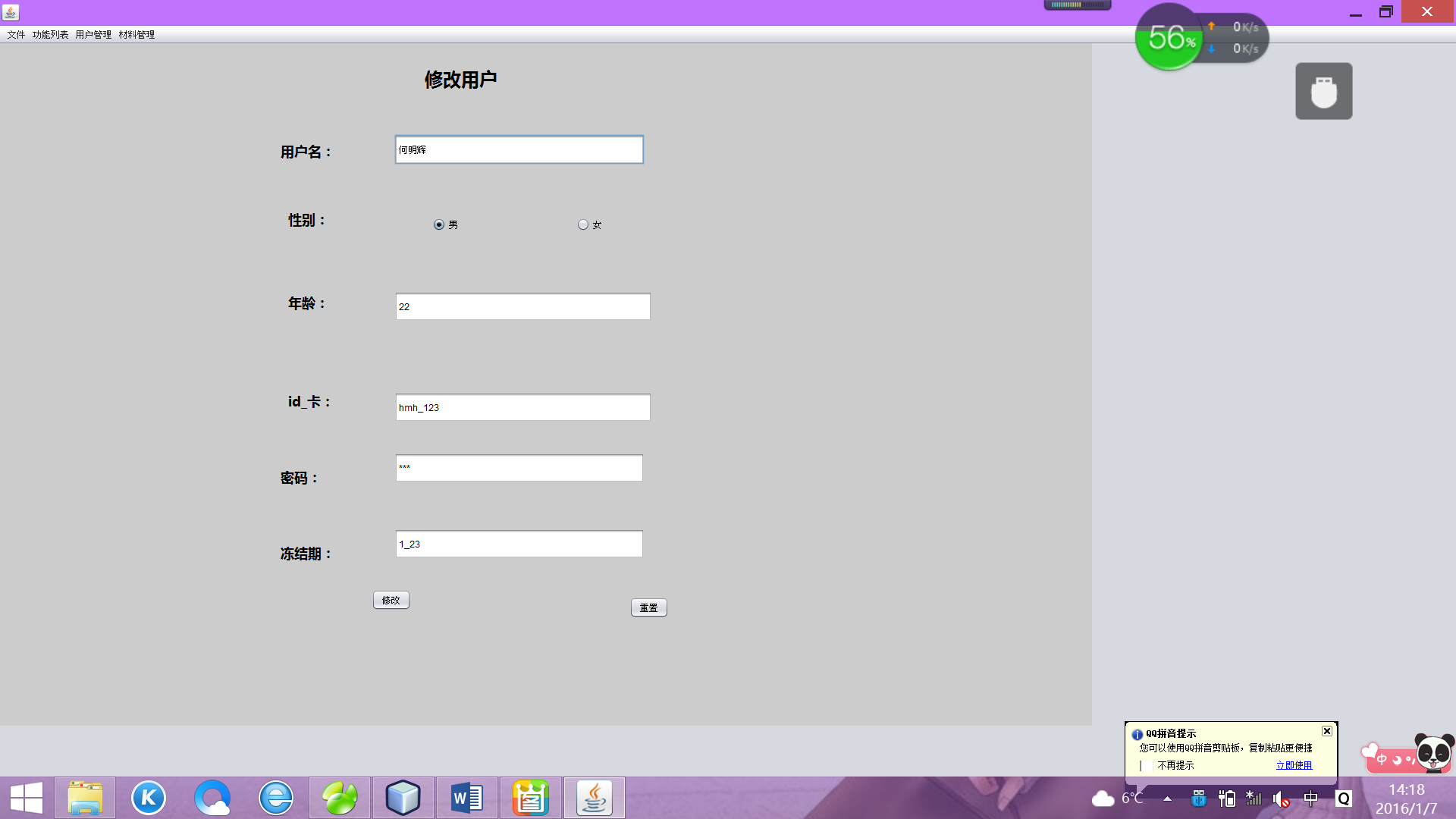
jTextField22.setText("");

jTextField23.setText("");

}

### [4.2.3 用户信息修改](#_Toc233984926)

该功能是根据用户名查找出管理员资料，对管理员需要修改的资料进行重定义，管理员信息修改下图所示。



主要代码如下：

public boolean updateUser(int id, User u) {

boolean flag = false;

try {

con = db.getCon();

String sql = "Update tb\_user set username=?,password=?,sex=?,age=?,tel=?,address=?,attribute=? where id=?";

pstmt = con.prepareStatement(sql);

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

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

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

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

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

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

pstmt.setInt(7, u.getAttribute());

pstmt.setInt(8, id);

int i = pstmt.executeUpdate();

if (i > 0) {

flag = true;

} else {

flag = false;

}

if (pstmt != null) {

pstmt.close();

}

if (con != null) {

con.close();

}

} catch (SQLException ex) {

ex.printStackTrace();

}

return flag;

}

private void jButton26ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("gui正在执行添加用户操作");

udi = new UserDAOImp();

//1 获取组件的信息 并进行类型转换

String uname = jTextField20.getText().trim();

char ch[] = jPasswordField1.getPassword();

String password = new String(ch);

String sex = "";

if (jRadioButton1.isSelected()) {

sex = "男";

}

if (jRadioButton2.isSelected()) {

sex = "女";

}

int age = 0;

try {

String agestr = jTextField21.getText(); //字符串

age = Integer.parseInt(agestr);

} catch (NumberFormatException e) {

e.printStackTrace();

}

String idk = jTextField22.getText().trim();

String freeze = jTextField23.getText().trim();

//如果属性超过三个，建议使用对象来封装传递数据

User u = new User();

u.setName(uname);

u.setAge(age);

u.setPassword(password);

u.setSex(sex);

u.setId\_card(idk);

u.setFreeze(freeze);

//2、调用数据访问层方法

boolean result = udi.insert(u);

if (result) {

System.out.println("gui添加用户成功");

this.setVisible(false);

//系统主界面

} else {

System.out.println("gui添加用户失败");

}

}

private void jButton27ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("gui正在执行添加重置用户操作");

jTextField21.setText("");

jTextField20.setText("");

buttonGroup1.clearSelection();

jPasswordField1.setText("");

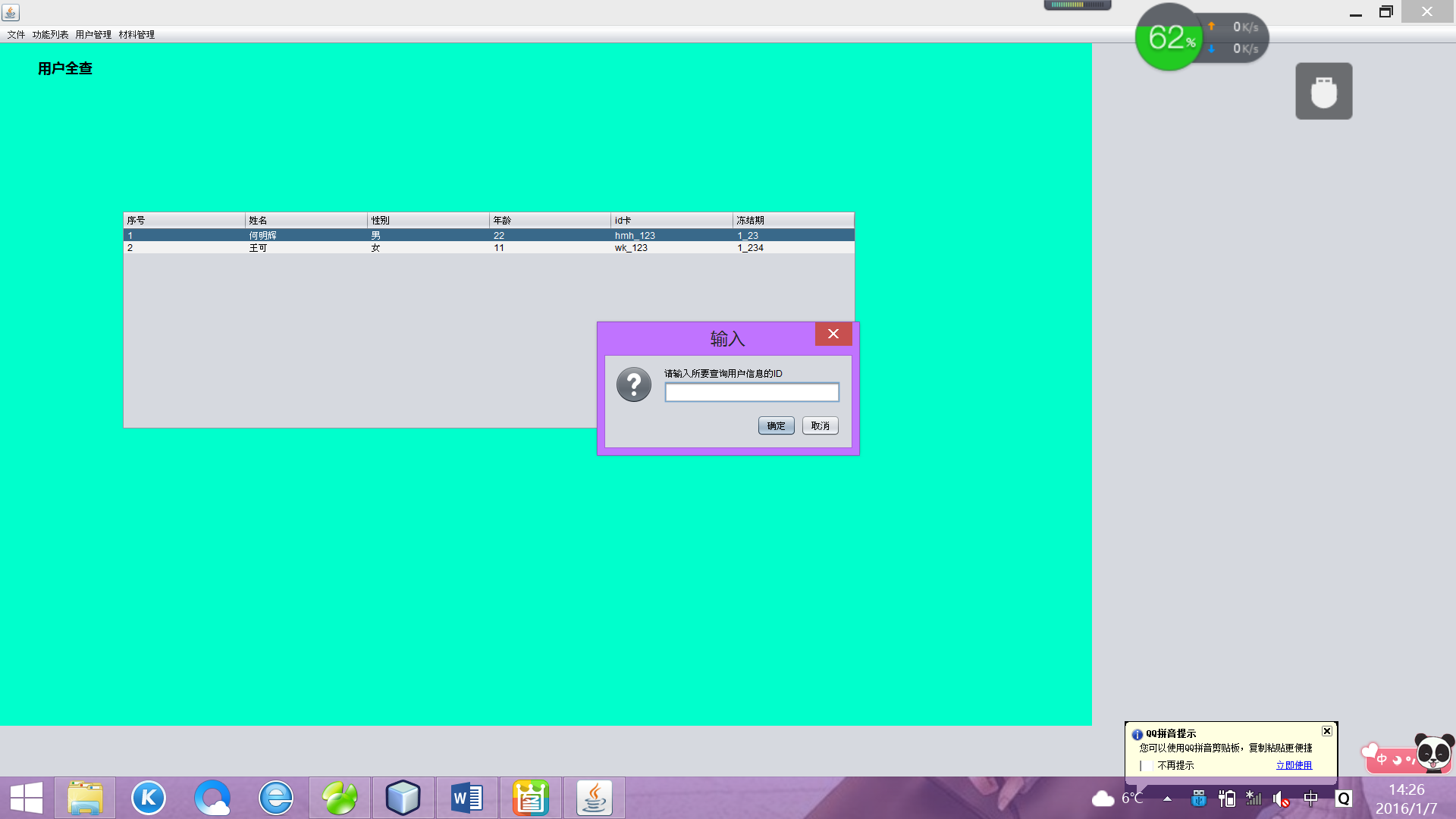
jTextField22.setText("");

jTextField23.setText("");

}

### 4.2.4 用户ID查询

该功能是对数据库中所有管理员id查询。管理员id查询如下图所示。



主要代码如下：

private void jMenuItem12ActionPerformed(java.awt.event.ActionEvent evt) {

String inputValue = JOptionPane.showInputDialog("请输入所要查询用户信息的ID");

input = new Integer(inputValue);

udi = new UserDAOImp();

User user = udi.selectById(input);

if (user != null) {

jPanel1.setVisible(false);

jPanel2.setVisible(false);

jPanel3.setVisible(false);

jPanel6.setVisible(false);

jPanel7.setVisible(false);

jPanel8.setVisible(false);

jPanel9.setVisible(true);

jPanel10.setVisible(false);

jPanel11.setVisible(false);

jPanel12.setVisible(false);

jLabel38.setText("ID查询");

jTextField28.setVisible(false);

jLabel53.setVisible(false);

jTextField29.setVisible(false);

jLabel54.setVisible(false);

jTextField30.setVisible(false);

jLabel55.setVisible(false);

jButton30.setVisible(false);

jButton31.setVisible(false);

jButton35.setVisible(false);

jButton36.setVisible(false);

jButton37.setVisible(false);

jButton38.setVisible(false);

jButton39.setVisible(false);

jTable9.setModel(new javax.swing.table.DefaultTableModel(

this.getDataByID(),

new String[]{

"序号", "姓名", "性别", "年龄", "id卡", "冻结期"

}

));

} else {

JOptionPane.showMessageDialog(null, "根据ID查无此人", "错误提示", JOptionPane.ERROR\_MESSAGE);

}

}

public User selectById(int id) {

User u = null;

try {

con = db.getCon();

if (con == null) {

System.out.println("UserDaoImp中数据库连接失败");

}

String sql = "select \* from tb\_user where id=?";

pstmt = con.prepareStatement(sql);

pstmt.setInt(1, id);

rs = pstmt.executeQuery();// 执行sql语句

System.out.println("id\t用户名\t性别\t年龄\tId卡\t密码\t冻结");

while (rs.next()) {

System.out.println("有一行新纪录");

int uid = rs.getInt(1);

String uname = rs.getString(2);

String sex = rs.getString(3);

int age = rs.getInt(4);

String id\_card = rs.getString(5);

String password = rs.getString(6);

String freeze = rs.getString(7);

System.out.println(uid + "\t" + uname + "\t" + sex + "\t"

+ age+ "\t"+id\_card+ "\t"+ password + "\t" + freeze);

u = new User();

u.setId(uid);

u.setName(uname);

u.setSex(sex);

u.setAge(age);

u.setId\_card(id\_card);

u.setPassword(password);

u.setFreeze(freeze);

}

rs.close();

pstmt.close();

con.close();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return u;

}

public Object[][] getDataByID() {

//从数据库中获取数据

udi = new UserDAOImp();

User user = udi.selectById(input);

//list.size();

list = new ArrayList<User>();

list.add(user);

Object[][] dataId = null;

dataId = new Object[list.size()][6];

int count = 1;

if (user != null) {

for (int i = 0; i < list.size(); i++) {

//处理好一行一行的数据

User u = list.get(i);

dataId[i][0] = count++;

dataId[i][1] = u.getName();

dataId[i][2] = u.getSex();

dataId[i][3] = u.getAge();

dataId[i][4] = u.getId\_card();

dataId[i][5] = u.getFreeze();

}

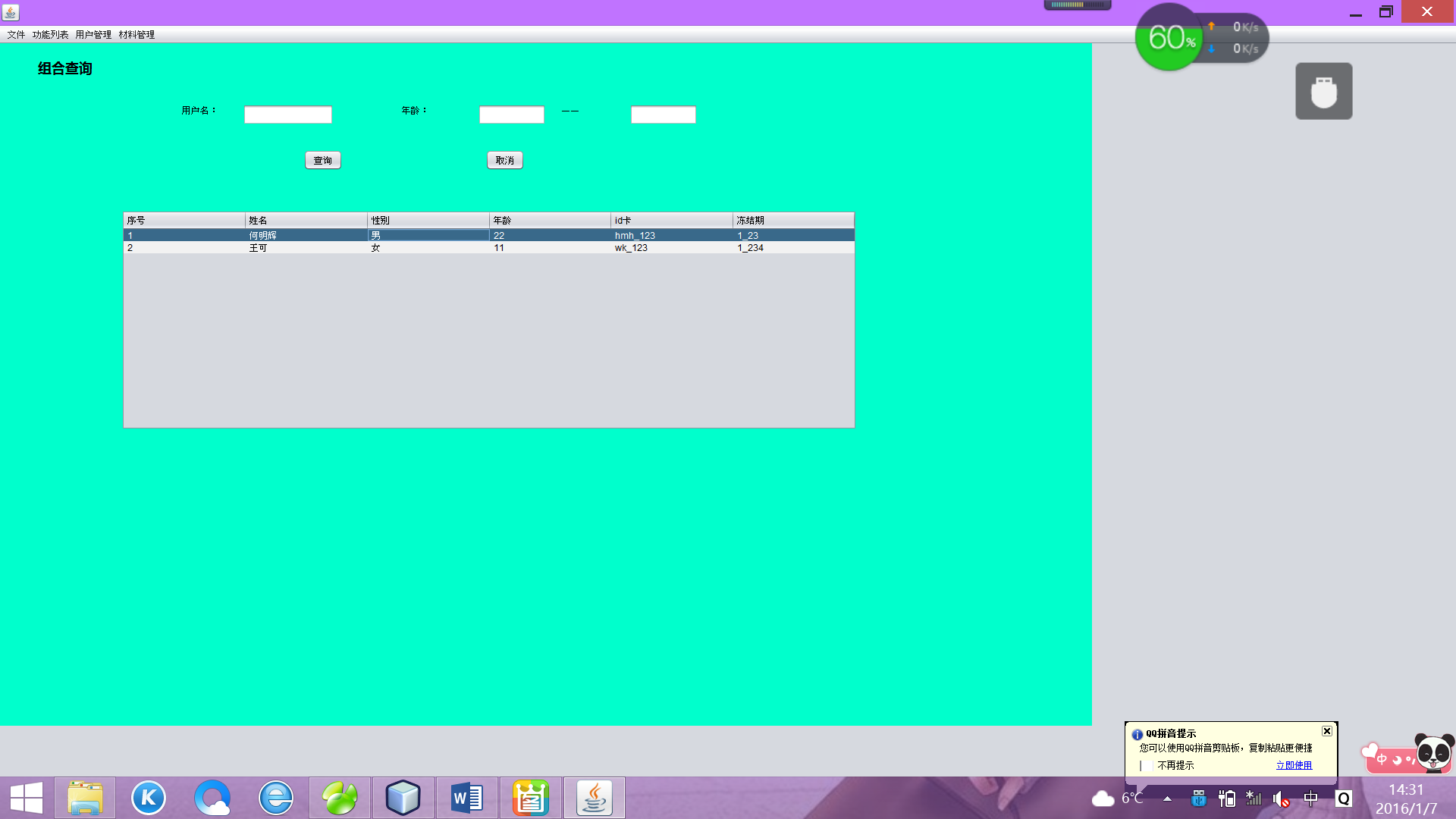
}

return dataId;

}

### [4.2.5 用户组合查询](#_Toc233984928)

该功能是对数据库中所有管理员id查询。管理员id查询如下图所示。



主要代码如下：

public List<User> selectZuHeB(String name, int age1, int age2) {

List<User> list = new ArrayList<User>();

try {

con = db.getCon();

if (con == null) {

System.out.println("UserDaoImp中数据库连接失败");

}

String sql = "select \* from tb\_user where name like ? and age between ? and ?";

pstmt = con.prepareStatement(sql);

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

pstmt.setInt(2, age1);

pstmt.setInt(3, age2);

rs = pstmt.executeQuery();// 执行sql语句

System.out.println("id\t用户名\t性别\t年龄\tId卡\t密码\t冻结");

while (rs.next()) {

int uid = rs.getInt(1);

String uname = rs.getString(2);

String sex = rs.getString(3);

int age = rs.getInt(4);

String id\_card = rs.getString(5);

String password = rs.getString(6);

String freeze = rs.getString(7);

System.out.println(uid + "\t" + uname + "\t" + sex + "\t"

+ age+ "\t"+id\_card+ "\t"+ password + "\t" + freeze);

User u = new User();

u.setId(uid);

u.setName(uname);

u.setSex(sex);

u.setAge(age);

u.setId\_card(id\_card);

u.setPassword(password);

u.setFreeze(freeze);

list.add(u);

}

rs.close();

pstmt.close();

con.close();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return list;

}

private void jButton30ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("gui正在执行组合查询用户操作");

udi = new UserDAOImp();

name = jTextField28.getText().trim();

age1 = new Integer(jTextField29.getText().trim());

age2 = new Integer(jTextField30.getText().trim());

list = udi.selectZuHeB(name, age1, age2);

if (list.size() > 0) {

jPanel1.setVisible(false);

jPanel2.setVisible(false);

jPanel3.setVisible(false);

jPanel6.setVisible(false);

jPanel7.setVisible(false);

jPanel8.setVisible(false);

jPanel9.setVisible(true);

jPanel10.setVisible(false);

jPanel11.setVisible(false);

jPanel12.setVisible(false);

jTable9.setModel(new javax.swing.table.DefaultTableModel(

this.getDataZuHeB(),

new String[]{

"序号", "姓名", "年龄", "性别", "兴趣爱好"

}

));

} else {

JOptionPane.showMessageDialog(null, "根据姓名和年龄查无此人", "错误提示", JOptionPane.ERROR\_MESSAGE);

}

}

public Object[][] getDataZuHeB() {

//从数据库中获取数据

udi = new UserDAOImp();

list = new ArrayList<User>();

list = udi.selectZuHeB(name, age1, age2);

Object[][] dataZuHeB = null;

dataZuHeB = new Object[list.size()][6];

int count = 1;

if (list != null) {

for (int i = 0; i < list.size(); i++) {

//处理好一行一行的数据

User u = list.get(i);

dataZuHeB[i][0] = count++;

dataZuHeB[i][1] = u.getName();

dataZuHeB[i][2] = u.getSex();

dataZuHeB[i][3] = u.getAge();

dataZuHeB[i][4] = u.getId\_card();

dataZuHeB[i][5] = u.getFreeze();

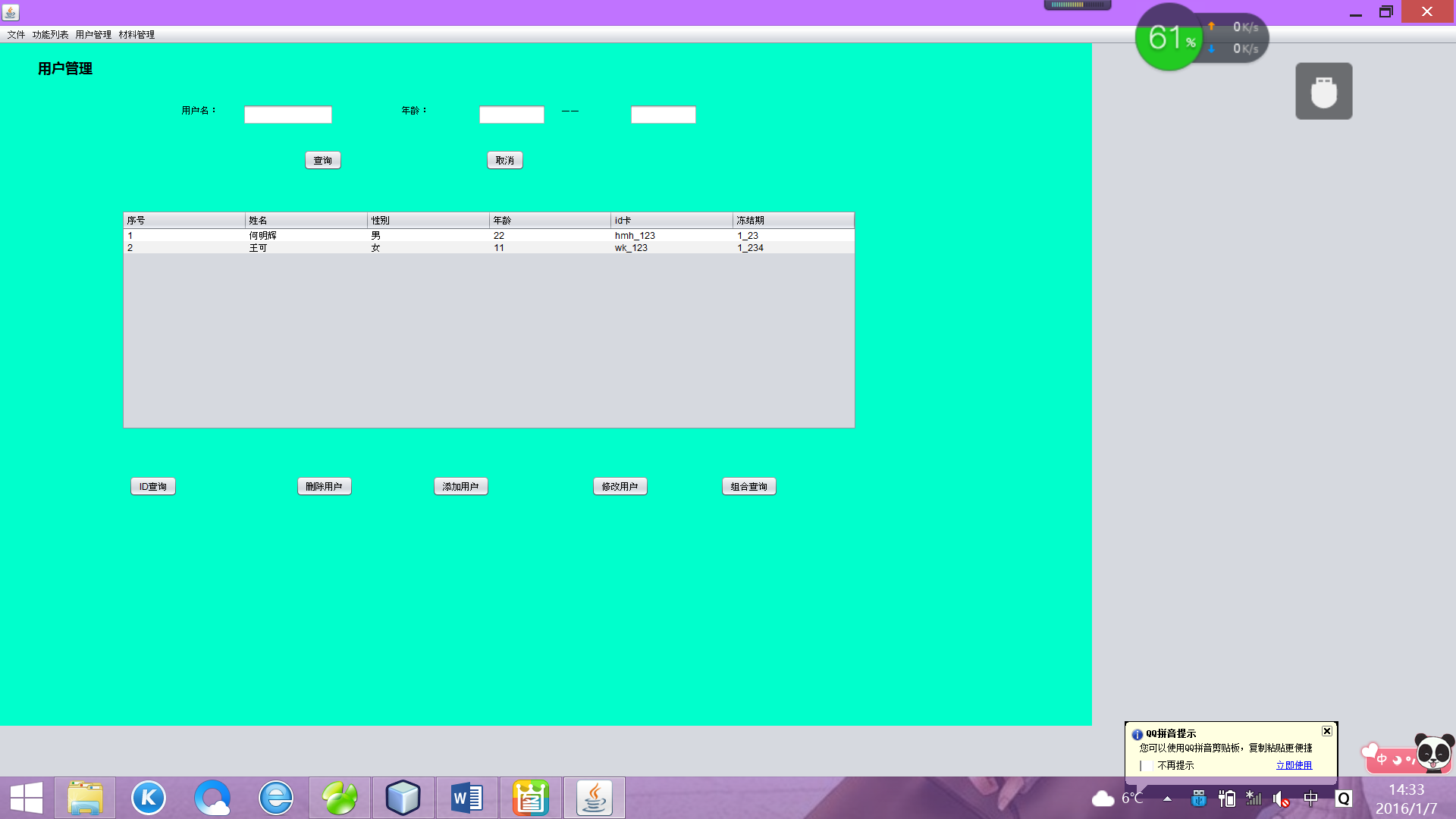
}

}

return dataZuHeB;

}

### [4.2.6 管理员的用户管理](#_Toc233984928)



主要代码如下：

private void jButton24ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("用户管理按钮");

jPanel1.setVisible(false);

jPanel2.setVisible(false);

jPanel3.setVisible(false);

jPanel6.setVisible(false);

jPanel7.setVisible(false);

jPanel8.setVisible(false);

jPanel9.setVisible(true);

jPanel10.setVisible(false);

jPanel11.setVisible(false);

jPanel12.setVisible(false);

jLabel38.setText("用户管理");

jTextField28.setVisible(true);

jLabel53.setVisible(true);

jTextField29.setVisible(true);

jLabel54.setVisible(true);

jTextField30.setVisible(true);

jLabel55.setVisible(true);

jButton30.setVisible(true);

jButton31.setVisible(true);

jButton35.setVisible(true);

jButton36.setVisible(true);

jButton37.setVisible(true);

jButton38.setVisible(true);

jButton39.setVisible(true);

jTable9.setModel(new javax.swing.table.DefaultTableModel(

this.getData(),

new String[]{

"序号", "姓名", "性别", "年龄", "id卡", "冻结期"

}

));

}

## [**4.3 材料管理模块功能实现**](#_Toc233984927)

### [4.3.1 材料信息查询](#_Toc233984928)

该功能是对数据库中所有材料信息进行查询。材料信息查询如下图所示。



主要代码如下：

public List<Goods> selectAll() {

List<Goods> list = new ArrayList();

try {

con = db.getCon();

if (con == null) {

System.out.print("数据库连接失败");

}

stmt = con.createStatement();

String sql = "select \* from tb\_goods";

rs = stmt.executeQuery(sql);

while (rs.next()) {

System.out.println("有一行新纪录"); //uid

String gnum=rs.getString(1);

String gname=rs.getString(2);

int ggnum=rs.getInt(3);

int gmoney=rs.getInt(4);

int gtotal=rs.getInt(5);

Goods g = new Goods();

g.setNum(gnum);

g.setName(gname);

g.setGnum(ggnum);

g.setGumoney(gmoney);

g.setGtotal(gtotal);

list.add(g);

}

} catch (SQLException e) {

e.printStackTrace();

}

return list;

}

private void jMenuItem3ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("材料全查");

jPanel1.setVisible(false);

jPanel2.setVisible(false);

jPanel3.setVisible(true);

jPanel6.setVisible(false);

jPanel7.setVisible(false);

jPanel8.setVisible(false);

jPanel9.setVisible(false);

jPanel10.setVisible(false);

jPanel11.setVisible(false);

jPanel12.setVisible(false);

jTable3.setModel(new javax.swing.table.DefaultTableModel(

this.getDataGoods(),

new String[]{

"序号", "材料编号", "材料名称", "材料数量", "单价", "总价"

}

));

}

public Object[][] getDataGoods() //list 中的数据转换为

{

//从数据库中获取数据

gdi = new GoodsDaoImp();

listg = gdi.selectAll();

//list.size();

System.out.println(listg.size());

Object[][] dataGoods = new Object[listg.size()][6];

int count = 1;

for (int i = 0; i < listg.size(); i++) {

//处理好一行一行的数据

Goods g = listg.get(i);

dataGoods[i][0] = count++;

dataGoods[i][1] = g.getNum();

dataGoods[i][2] = g.getName();

dataGoods[i][3] = g.getGnum();

dataGoods[i][4] = g.getGumoney();

dataGoods[i][5] = g.getGtotal();

}

return dataGoods;

} Private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try {

System.out.println("执行下拉框事件");

String selectitem = (String) jComboBox1.getSelectedItem();

System.out.println("选中的事件是" + selectitem);

if (selectitem.equals("用户号")) {

u = udi.getUserInfoById(Integer.parseInt(jTextField6.getText().trim()));

}

if (selectitem.equals("用户名")) {

u = udi.getUserInfoByName(jTextField6.getText().trim());

}

if (selectitem.equals("地址")) {

u = udi.getUserInfoByAddress(jTextField6.getText().trim());

}

} catch (NumberFormatException e) {

JOptionPane.showMessageDialog(this, "警告：请输入正确数据！");

}

}

private void jButton10ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

//查找按钮处理

jComboBox1ActionPerformed(evt);

boolean flag = udi.isExist(u.getUsername(), u.getPassword());

if (flag) {

System.out.println("查找成功！");

if (u.getAttribute() == 0) {

DefaultTableModel newmodel = new DefaultTableModel(new Object[][]{{u.getId(), u.getUsername(), u.getSex(), u.getAge(), u.getTel(), u.getAddress()}},

new String[]{"用户号", "姓名", "性别", "年龄", "联系方式", "地址"});

jTable1.setModel(newmodel);

} else {

JOptionPane.showMessageDialog(this, "查找失败，不存在该代理商！");

}

} else {

JOptionPane.showMessageDialog(this, "查找失败，不存在该代理商！");

}

}

row3 = jTable6.getSelectedRow();

System.out.println("表格点击第" + (row3 + 1) + "行");

if (row3 >= 0) {

System.out.println("表格点击行可进行操作");

Object[] options = {"删除", "取消"};

int response3 = JOptionPane.showOptionDialog(this, "这是个选项对话框，用户可以选择自己的按钮的个数", "选项对话框标题", JOptionPane.YES\_OPTION, JOptionPane.QUESTION\_MESSAGE, null, options, options[0]);

if (response3 == 0) {

System.out.println("您按下了删除按钮 ");

deletedesk = listd.get(row3);

ddi = new DeskDaoImp();

boolean flag3 = ddi.delect(deletedesk.getNum());

if (flag3) {

System.out.println("gui删除成功");

jTable6.setModel(new javax.swing.table.DefaultTableModel(

this.getDataDesk(),

new String[]{

"序号", "台号", "座位数"

}

));

} else {

System.out.println("gui删除不成功");

}

} else if (response3 == 1) {

System.out.println("您按下了取消按钮 ");

}

}

### [4.3.2 材料添加](#_Toc233984929)

该功能是有管理员进行添加新的材料的资料。材料添加如下图所示。



主要代码如下：

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("材料添加");

String gnum = jTextField1.getText();

String gname = jTextField2.getText();

int ggnum = 0;

try {

String seatingstr = jTextField4.getText().trim(); //字符串

ggnum = Integer.parseInt(seatingstr);

} catch (NumberFormatException e) {

e.printStackTrace();

}

int gmoney = 0;

try {

String seatingstr = jTextField11.getText().trim(); //字符串

gmoney = Integer.parseInt(seatingstr);

} catch (NumberFormatException e) {

e.printStackTrace();

}

jTextField13.setText(gmoney \* ggnum + "");

int gtotal = 0;

try {

String seatingstr = jTextField13.getText().trim(); //字符串

gtotal = Integer.parseInt(seatingstr);

} catch (NumberFormatException e) {

e.printStackTrace();

}

Goods g = new Goods();

g.setNum(gnum);

g.setName(gname);

g.setGnum(ggnum);

g.setGumoney(gmoney);

g.setGtotal(gtotal);

//2、调用数据访问层方法

boolean result = gdi.insert(g);

if (result) {

System.out.println("gui添加材料成功");

jTable3.setModel(new javax.swing.table.DefaultTableModel(

this.getDataGoods(),

new String[]{

"序号", "材料编号", "材料名称", "材料数量", "单价", "总价"

}

));

//系统主界面

} else {

System.out.println("gui添加材料失败");

}

}

public boolean insert(Goods g) {

boolean flag = true;

try {

con = db.getCon();

if (con == null) {

System.out.print("数据库连接失败");

}

//pstmt

String sql = "insert into tb\_goods(num,name,gnum,gumoney,gtotal) values(?,?,?,?,?)";

pstmt = con.prepareStatement(sql);

pstmt.setString(1, g.getNum());

pstmt.setString(2, g.getName());

pstmt.setInt(3, g.getGnum());

pstmt.setInt(4, g.getGumoney());

pstmt.setInt(5, g.getGtotal());

int state = pstmt.executeUpdate();

if (state > 0) {

System.out.print("dao中插入用户成功");

flag = true;

}

} catch (SQLException e) {

e.printStackTrace();

}

return flag;

}

### [4.3.3 材料信息修改](#_Toc233984930)

该功能是对需要修改的材料的资料进行重新定义，材料信息修改如下图所示。



主要代码如下：

private void jButton21ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("材料修改");

gdi = new GoodsDaoImp();

String nnum = listg.get(row5).getNum();

System.out.println("获取行号");

String gnum = jTextField32.getText();

String gname = jTextField33.getText();

int ggnum = 0;

try {

String seatingstr = jTextField34.getText().trim(); //字符串

ggnum = Integer.parseInt(seatingstr);

} catch (NumberFormatException e) {

e.printStackTrace();

}

int gmoney = 0;

try {

String seatingstr = jTextField35.getText().trim(); //字符串

gmoney = Integer.parseInt(seatingstr);

} catch (NumberFormatException e) {

e.printStackTrace();

}

jTextField36.setText(gmoney \* ggnum + "");

int gtotal = 0;

try {

String seatingstr = jTextField36.getText().trim(); //字符串

gtotal = Integer.parseInt(seatingstr);

} catch (NumberFormatException e) {

e.printStackTrace();

}

Goods g = new Goods();

g.setNum(gnum);

g.setName(gname);

g.setGnum(ggnum);

g.setGumoney(gmoney);

g.setGtotal(gtotal);

boolean result = gdi.update(g, nnum);

if (result) {

jTable3.setModel(new javax.swing.table.DefaultTableModel(

this.getDataGoods(),

new String[]{

"序号", "材料编号", "材料名称", "材料数量", "单价", "总价"

}

));

System.out.println("gui修改材料成功");

jPanel1.setVisible(false);

jPanel2.setVisible(false);

jPanel3.setVisible(true);

jPanel6.setVisible(false);

jPanel7.setVisible(false);

jPanel8.setVisible(false);

jPanel9.setVisible(false);

jPanel10.setVisible(false);

jPanel12.setVisible(false);

//系统主界面

} else {

System.out.println("gui修改材料失败");

}

public boolean update(Goods goods, String num) {

boolean flag = false;

try {

con = db.getCon();

if (con == null) {

System.out.println("UserDaoImp中数据库连接失败");

}

String sql = "update tb\_goods set num=?,name=?,gnum=?,gumoney=?,gtotal=? where num=?";

pstmt = con.prepareStatement(sql);

pstmt.setString(1, goods.getNum());

pstmt.setString(2, goods.getName());

pstmt.setInt(3, goods.getGnum());

pstmt.setInt(4, goods.getGumoney());

pstmt.setInt(5, goods.getGtotal());

pstmt.setString(6, num);

int count = pstmt.executeUpdate();

if (count > 0) {

flag = true;

System.out.println("成功修改" + count + "条数据！");

}

pstmt.close();

con.close();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

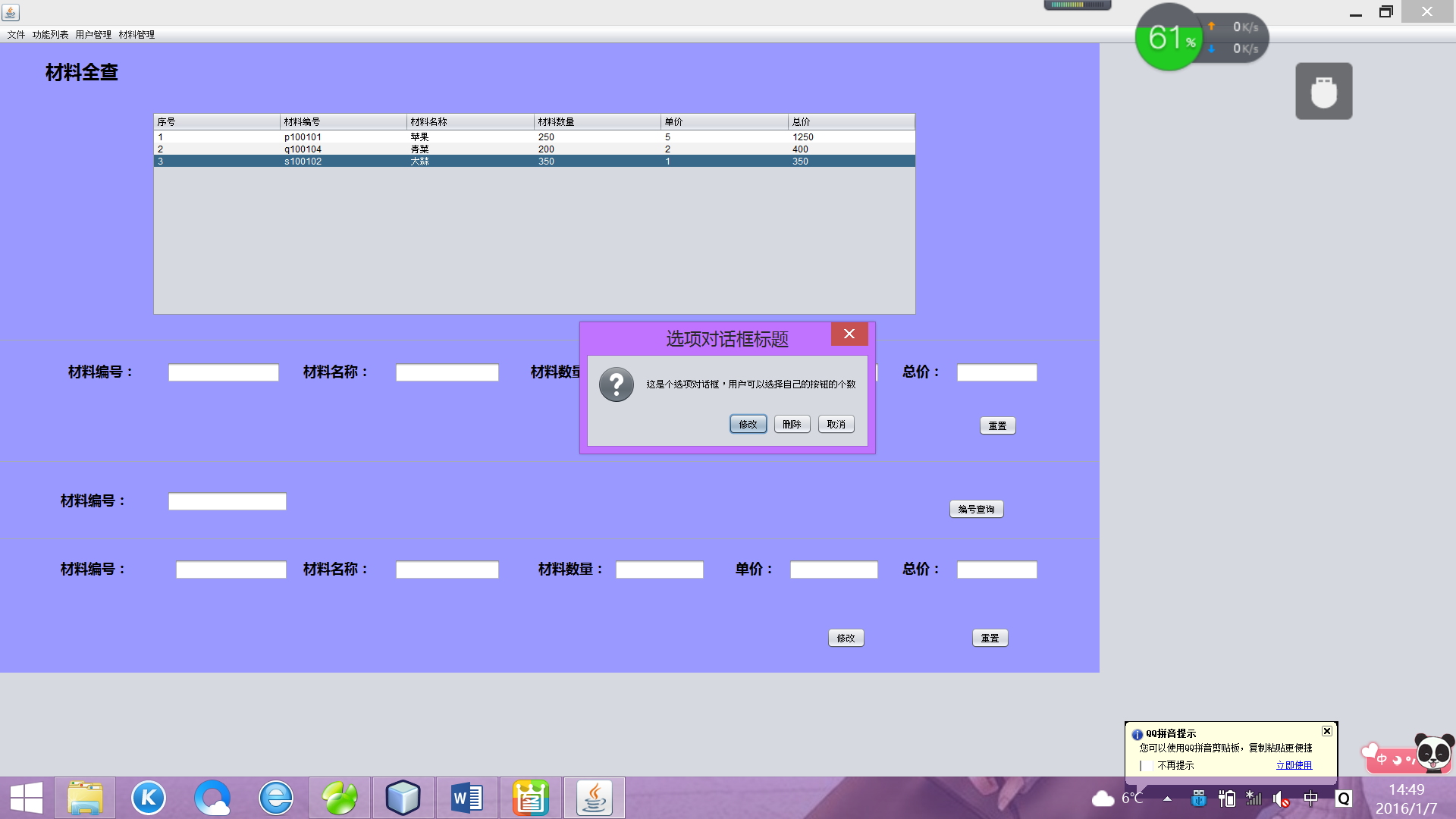
}

return flag;

}

### [4.3.4 材料删除](#_Toc233984931)

该功能是删除数据库中已有材料的信息。材料删除如下图所示。



主要代码如下：

public boolean delect(String num) {

boolean flag = false;

try {// 1、连接

con = db.getCon();

if (con == null) {

System.out.println("UserDaoImp中数据库连接失败");

}

String sql = "delete from tb\_goods where num=?";

pstmt = con.prepareStatement(sql);

pstmt.setString(1, num);

int state = pstmt.executeUpdate();

if (state > 0) {

System.out.println("在dao中删除一行记录");

flag = true;

}

pstmt.close();

con.close();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return flag;

}

## [**4.4 功能列表模块功能实现**](#_Toc233984932)

### [4.4.1 主面板](#_Toc233984933)

该功能是实现对数据库中顾客订单查询。顾客订单查询如下图所示。



主要代码如下：

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("正在执行开单操作");

str = "NEW";

mdi = new MenuDaoImp();

String fnum = jTextField3.getText().trim();

String jc5 = (String) jComboBox5.getSelectedItem();

String jc1 = (String) jComboBox1.getSelectedItem();

String jc2 = (String) jComboBox2.getSelectedItem();

int funit = 0;

try {

String seatingstr = jTextField5.getText().trim(); //字符串

funit = Integer.parseInt(seatingstr);

} catch (NumberFormatException e) {

e.printStackTrace();

}

Buy b = new Buy();

b.setNum(jc5);

b.setName(mdi.selectByNum(jc5).getName());

b.setCode(mdi.selectByNum(jc5).getCode());

b.setUnit(mdi.selectByNum(jc5).getUnit());

b.setUnit\_price(mdi.selectByNum(jc5).getUnit\_price());

b.setCount(funit);

b.setTotal(funit \* mdi.selectByNum(jc5).getUnit\_price());

b.setBstate(str);

DeskUse du = new DeskUse();

du.setNum(jc1);

du.setSeating(0);

dudi = new DeskUserDaoImp();

boolean result1 = dudi.insert(du);

//2、调用数据访问层方法

bdi = new BuyDaoImp();

boolean result = bdi.insert(b);

if (result) {

System.out.println("gui添加菜系成功");

jTable1.setModel(new javax.swing.table.DefaultTableModel(

this.getDataForm(),

new String[]{

"", "序号", "商品编号", "商品名称", "单位", "数量", "单价", "金额"

}

));

jTable2.setModel(new javax.swing.table.DefaultTableModel(

this.getDataForm1(),

new String[]{

"序号", "台号", "开台时间"

}

));

//系统主界面

} else {

System.out.println("gui添加菜系失败");

}

}

private void jButton16ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("正在执行结账操作");

int xmoney = 0;

try {

String seatingstr = jTextField17.getText().trim(); //字符串

xmoney = Integer.parseInt(seatingstr);

} catch (NumberFormatException e) {

e.printStackTrace();

}

int smoney = 0;

try {

String seatingstr = jTextField18.getText().trim(); //字符串

smoney = Integer.parseInt(seatingstr);

} catch (NumberFormatException e) {

e.printStackTrace();

}

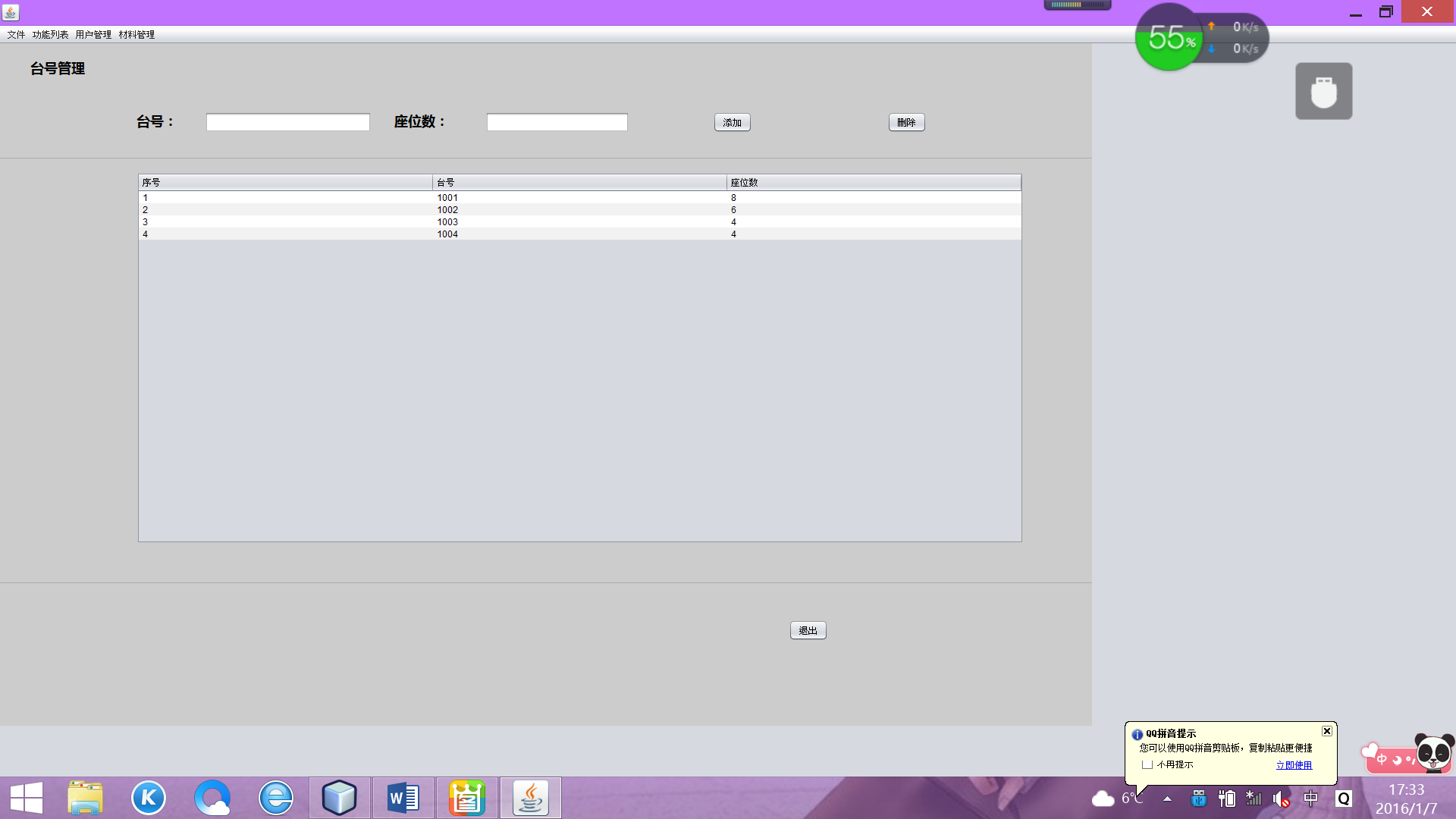
int rr = smoney - xmoney;

jTextField19.setText(rr + "");

}

### [4.4.2 台号管理](#_Toc233984934)

该功能用于向数据库中台号查询。结果如下图所示。



主要代码如下：

private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("正在执行台号添加操作");

ddi = new DeskDaoImp();

String dnum = jTextField6.getText().trim();

int seating = 0;

try {

String seatingstr = jTextField7.getText().trim(); //字符串

seating = Integer.parseInt(seatingstr);

} catch (NumberFormatException e) {

e.printStackTrace();

}

Desk d = new Desk();

d.setNum(dnum);

d.setSeating(seating);

//2、调用数据访问层方法

boolean result = ddi.insert(d);

if (result) {

System.out.println("gui添加菜系成功");

jTable6.setModel(new javax.swing.table.DefaultTableModel(

this.getDataDesk(),

new String[]{

"序号", "台号", "座位数"

}

));

//系统主界面

} else {

System.out.println("gui添加菜系失败");

}

}

private void jButton8ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("正在执行台号删除操作");

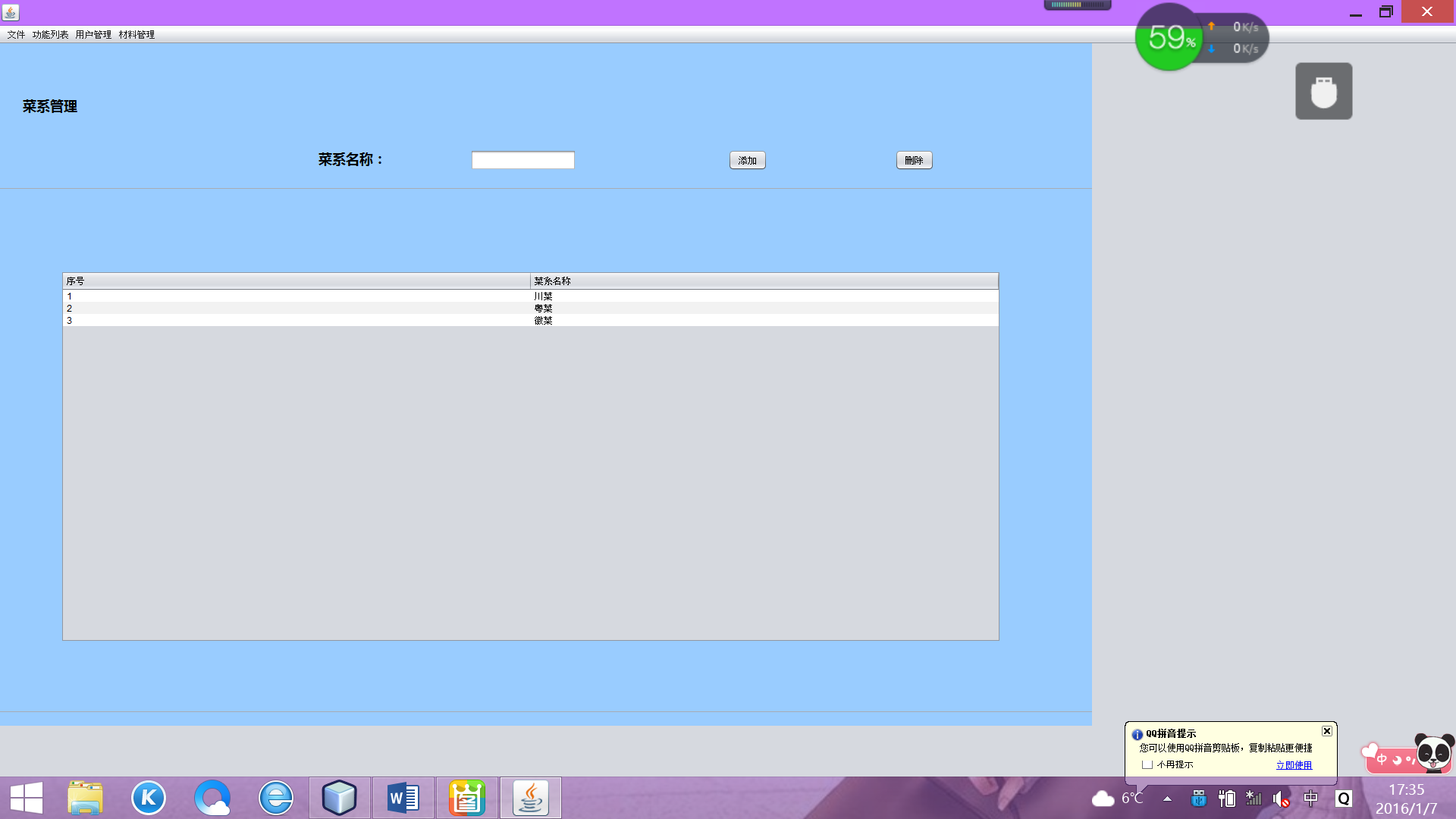
jTextField6.setText("");

jTextField7.setText("");

}

### [4.4.3 菜系管理](#_Toc233984934)

该功能用于对菜系查询。菜系信息修改如下图所示。



主要代码如下：

private void jButton10ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("正在执行菜系添加操作");

sdi = new SortDaoImp();

String sname = jTextField8.getText().trim();

Sort s = new Sort();

s.setName(sname);

//2、调用数据访问层方法

boolean result = sdi.insert(s);

if (result) {

System.out.println("gui添加菜系成功");

jTable7.setModel(new javax.swing.table.DefaultTableModel(

this.getDataSort(),

new String[]{

"序号", "菜系名称"

}

));

//系统主界面

} else {

System.out.println("gui添加菜系失败");

}

}

private void jButton11ActionPerformed(java.awt.event.ActionEvent evt) {

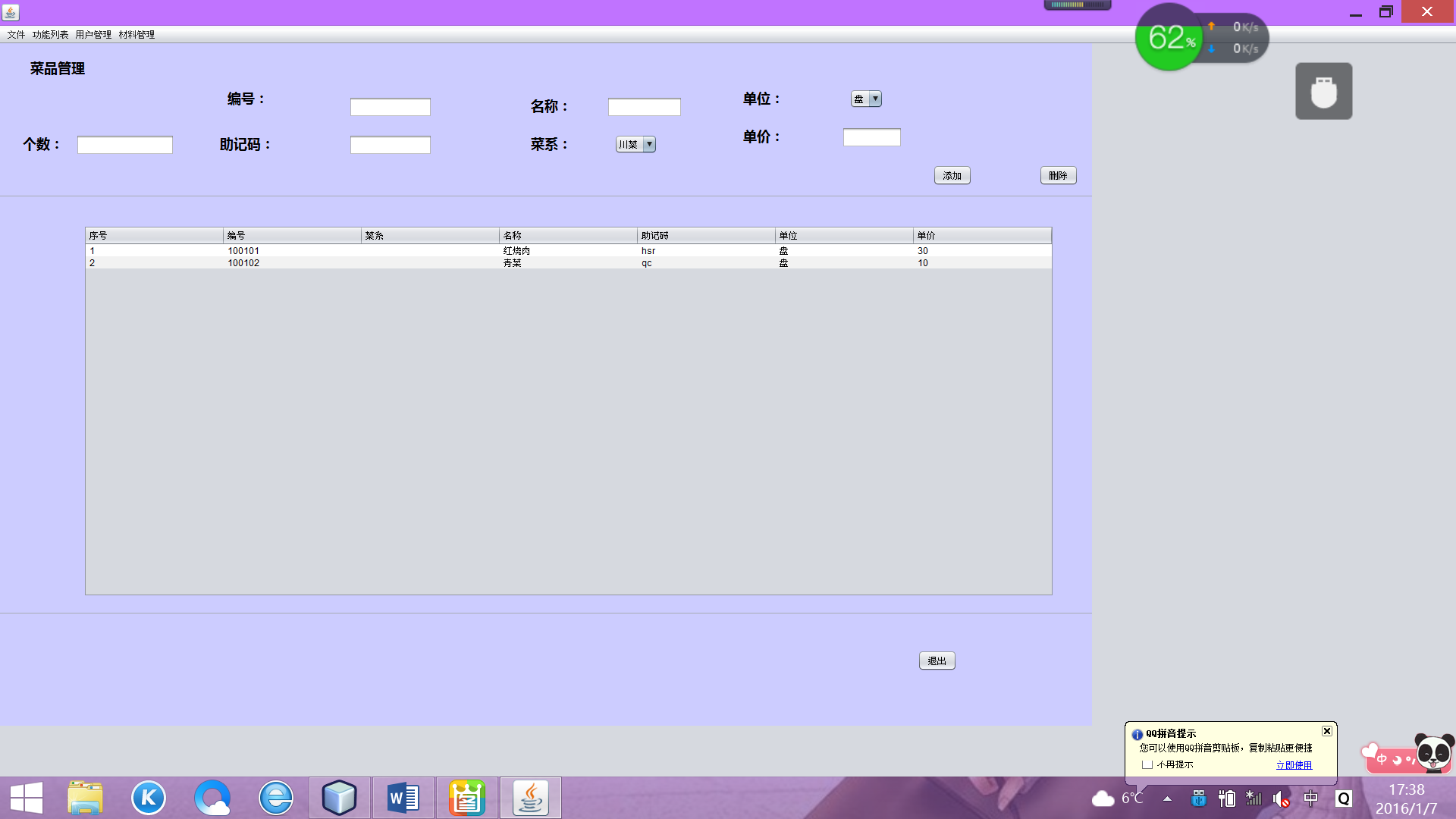
System.out.println("正在执行菜系删除操作");

jTextField8.setText("");

}

### [4.4.4 菜品管理](#_Toc233984934)

该功能用于对数据库中已有的菜品查询，菜品删除如下图所示。



主要代码如下：

private void jButton13ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("正在执行菜品添加操作");

mdi = new MenuDaoImp();

String mnum = jTextField9.getText().trim();

String mname = jTextField10.getText().trim();

int price = 0;

try {

String pricestr = jTextField14.getText().trim(); //字符串

price = Integer.parseInt(pricestr);

} catch (NumberFormatException e) {

e.printStackTrace();

}

int number = 0;

try {

String pricestr = jTextField37.getText().trim(); //字符串

number = Integer.parseInt(pricestr);

} catch (NumberFormatException e) {

e.printStackTrace();

}

String jc3 = (String) jComboBox3.getSelectedItem();

String mcode = jTextField12.getText().trim();

jc4 = (String) jComboBox4.getSelectedItem();

int np = number \* price;

Menu m = new Menu();

m.setNum(mnum);

m.setCode(mcode);

m.setName(mname);

m.setUnit(jc3);

m.setUnit\_price(price);

m.setCount(number);

m.setTotal(np);

//2、调用数据访问层方法

boolean result = mdi.insert(m);

if (result) {

System.out.println("gui添加菜品成功");

jTable8.setModel(new javax.swing.table.DefaultTableModel(

this.getDataMenu(),

new String[]{

"序号", "编号", "菜系", "名称", "助记码", "单位", "单价"

}

));

//系统主界面

} else {

System.out.println("gui添加菜品失败");

}

}

private void jButton14ActionPerformed(java.awt.event.ActionEvent evt) {

System.out.println("正在执行菜品删除操作");

jTextField9.setText("");

jTextField10.setText("");

jTextField12.setText("");

jTextField14.setText("");

}

# [**结 论**](#_Toc233984939)

一周的java课程设计已经结束了，在这次课程设计中我学习了知识，努力完成了项目。虽然只是短短的一周时间，可是在这周我们把课堂里老师讲的知识用到实践中去了，感到非常高兴。在这次课设中我们不仅学到了知识，还锻练了我的动手的能力，在这次课设中我们真正体会到了实践的重要性，就如古语所说的“纸上得来终觉浅，绝知此事要躬行”，这对我以后的工作有很大的帮助。

制定设计题目很关键，我对酒店感兴趣，最终选定了酒店管理系统。之后在收集资料的过程中，我上网查阅资料，了解了酒店管理的相关资料，纷繁复杂，通过学习的理论知识和广泛地资料分析、制定了需求分析。

在程序中我主要用了awt包和swing包，界面主要用JFrame、JPanel来实现，用绝对布局设置。这个项目不但成功的连接了数据库，也成功实现了很多功能：在管理员方面实现了管理员登陆、用户删除、用户添加、用户信息修改法四个功能，在材料管理方面实现了材料添加、材料修改、材料查询、材料删除四个功能，在功能列表方面实现了台号管理、菜系管理、菜品管理、主面板显示订餐信息四个功能。在众多功能中刚开始时各项查询方面出了问题，但最后在自己思考、询问同学、咨询老师的帮助下，终于改正了错误，使此功能终于可以正常使用了。

这次的课程设计可以看作是一次理论与实践相结合的桥梁，通过这次的课程设计，我学习到了许多的知识，也认识到了自己目前的不足，那就是缺乏相应的知识与经验，以及动手能力不足，所以在运用和操作方面都不是那么的得心应手。但是，经过这段时间对相关书籍的阅读和分析，顺利的完成了设计，我还明白了在编写程序的时候，应该尽量使界面简洁大方，布局统一。变量类型的定义，一定要够用就好，这样程序就可以尽可能的减少对系统资源的占用。在设计时也免不了存在着一些不足，所以在今后的学习中我会努力取得更大的进步，对于我不足的地方希望老师能够及时给予批评，以便我在今后的学习或工作中能够及时的改正。总之，这次课程设计为我提供了与众不同的学习方法，在书本中面对现实，为我将来在社会上立足提供了良好的前提。

课程设计虽然结束了，但我感觉要想把课程设计做好，那必须把基础知识学透彻。人生有数，学无止境，学习生活是无穷尽的，只有掌握更多的、更新的知识才能使自己始终立于不败之地。如果只为现在所取得的成绩而沉浸于喜悦、骄傲之中的话，那么，我想那是不会前进的，只能停留在原来的地方，甚至还会倒退。所以我应该从三周的课程设计中所暴露出的问题入手，不断完善自己的欠缺和不足。相信此次课程设计为我以后继续从事计算机工作打下了扎实的基础，也明确了以后的学习方向。虽然一周的时间很短暂，但是我相信在这次课程设计中学到的东西会是我永远的珍藏！

# [**致 谢**](#_Toc233984940)

历时一个星期的课程设计已经结束，我通用不懈的努力终于顺利地完成了课程设计的任务。在喜悦的同时，我也要向曾经在设计过程中帮助过我的朱老师和同学表示感谢！

在整个课程设计中朱老师对我们耐心指导和帮助才使我们能够顺利完成此次课程设计任务。老师每天都冒着严寒来到实训楼，给我们上课，尤其是她严谨的工作态度锐意创新的精神，使我们受益匪浅，在此特别向朱老师表示深深的感谢和由衷的敬意。

感谢学校提供给我们这次课程设计机会，使我们有了良好的环境学习Java程序设计和 数据库知识，使我们能够把学到的知识运用到实践中。一周的实训期间我们在实训中遇到了许多困难，当被这些问题困扰的时候，我们就向老师和同学寻求帮助，老师很认真的为我们解答并告诉我们是哪里出了问题导致了错误，通过老师的细心指导使我们在Java和 数据库设计方面的知识得到了扩充，懂得了许多以前不知道的知识，我们也在老师的细心指导中体会到了老师对学生那种从内心发出的关爱，让我们非常感动。

在这段时，同学们也给了我们不少的帮助，虽然大家都很忙，但是当我有问题时，大家还是细心地帮我解决所遇到的问题，给予我及时的帮助，在这里非常感谢所有的同学们。在老师及同学的指导和帮助下，我基本完成了此次课程设计的任务。可以说这次的成绩不只属于我自己，更多的属于老师的帮助。至此，我要向老师说一声老师：“您辛苦了”。我相信我们以后的路会走的更顺畅，不管以后遇到了怎样的困难，我都会想起老师那肯定我们时的目光，我能为我们能有这样的老师而倍感自豪。

最后再次感谢在课程设计过程中给予我帮助的老师和同学们，他们的帮助不仅使我开阔了视野，拓宽了思路，增长了学识，为我今后的工作和学习打下了牢固的基础，也使我增强了对系统开发的兴趣，特别是Java语言的系统开发。在这里再一次表示衷心的感谢。