

课程设计实践报告

****

**题目学生管理系统的设计与实现**

学生姓名 施苏玲

学 号 201983290418

学 院 计算机与软件学院

专 业 网络工程

指导教师 徐昕

**2021年6月13日**

1绪论

1.1设计背景

学生信息管理是学校管理中异常重要的一个环节，作为学校，除了育人，就是育知，学生信息管理的计算机化是整个学校教务管理中的重要一部分，能否实现这一步关系到学校办学整体效率的高低，由于它的重要性，学生信息管理系统的开发与应用就逐渐提入议程，并占着越来越重要的份量。

运用学生信息管理系统可以减轻学院教学人员的工作量，缩小开支，提高工作效率与准确率，能够合理安排时间，能够尽快的知道自己的考试成绩，投入新的课程的学习或复习这次没有考过的课程。而学生信息管理系统的应用也为今天的民办教育在未来市场的竞争力有所提高。从大的方向说，就是为了加速我国四化建设的发展，实现全部的自动化，使我国发展成为通讯网络化，决策科学化，办公自动化的国家。

在现代，高科技的飞跃发展，人们工作习惯的改变，特别是电脑的大量普及，人们生活节奏越来越快，怎样提高工作效率是人们首先考虑的问题。学生信息管理是一个非常繁琐与复杂的一项工作，一个原因就是工作量大不好管。对于一个学校而言，更应该运用一些本地资源，提高管理的力度，对学生负责，对国家负责。

1.2设计概述

本次设计主要目的是实现一个学生管理系统，系统主要功能要求如下：

1.学生管理：包括选课模块、学生课程表模块和成绩查询模块（可以查询本课程的最高成绩、平均成绩、最低成绩和名次）。

2.教师管理：课程设置申请模块，可以增加新的课程信息（课程号，课程名称、学分课时、课程简介等信息），删除旧的课程信息，修改已有的课程信息；课程管理模块，可以实现期末成绩的录入和学生名单的打印。

3.教务处管理：包括课程设置审批模块和教学安排模块。课程设置审批模块可以根据申请内容通过审批或拒绝申请。教学安排模块分为，选课结果管理，成绩单（学号、学生姓名、课程名称、成绩等信息）生成，并能统计选课人数、最高成绩、最低成绩、平均成绩和及格人数。

1.3开发系统的选用与介绍

全新一代 SQL Server 2019 为用户带来更多全新体验,，独特的产品优势定能使用户更加获益良多。企业版是全功能版本，而其它两个版本则分别面向工作组和中小企业，所支持的机器规模和扩展数据库功能都不一样，价格方面是根据处理器核心数量而定。

相较与之前版本，SQL Server2019具有以下优势：

（1）安全性和高可用性：

提高服务器正常运行时间并加强数据保护，无需浪费时间和金钱即可实现服务器到云端的扩展。

（2）企业安全性及合规管理：

内置的安全性功能及IT管理功能，能够在极大程度上帮助企业提高安全性能级别并实现合规管理。

（3）安心使用：

得益于卓越的服务和技术支持、大量值得信赖的合作伙伴、以及丰富的免费工具，用户可以放心使用。

（4）超快的性能：

在业界首屈一指的基准测试程序的支持下 ，用户可获得突破性的、可预测的性能。

（5）快速的数据发现：

通过快速的数据探索和数据可视化对成堆的数据进行细致深入的研究，从而能够引导企业提出更为深刻的商业洞见。

（6）可扩展的托管式自助[商业智能服务](https://baike.baidu.com/item/%E5%95%86%E4%B8%9A%E6%99%BA%E8%83%BD%E6%9C%8D%E5%8A%A1" \t "https://baike.baidu.com/item/Microsoft%20SQL%20Server%202012/_blank)：

通过托管式自主商业智能、IT 面板及 SharePoint 之间的协作，为整个商业机构提供可访问的智能服务。

（7）可靠、一致的数据：

针对所有业务数据提供一个全方位的视图，并通过整合、净化、管理帮助确保数据置信度。

（8）全方位的数据仓库解决方案：

凭借全方位数据仓库解决方案，以低成本向用户提供大规模的数据容量，能够实现较强的灵活性和可伸缩性。

（9）根据需要进行扩展：

通过灵活的部署选项，根据用户需要实现从服务器到云的扩展。

（10）解决方案的实现更为迅速：

通过一体机和私有云/公共云产品，降低解决方案的复杂度并有效缩短其实现时间。

（11）工作效率得到优化提高：

通过常见的工具，针对在服务器端和云端的IT人员及开发人员的工作效率进行优化。

（12）随心所欲扩展任意数据：

通过易于扩展的开发技术，可以在服务器或云端对数据进行任意扩展。

2 系统需求分析及相关技术介绍

2.1需求分析

用户的需求有课程的显示提供、保存、更新、查询，成绩的查询、更新、添加方面。通过系统功能分析，针对学生管理系统，应有如下功能需求：

1. 学生的需求：网上选择课程、选课情况查询、成绩查询、个人基本信息查询；
2. 老师的需求：老师能够对课程进行增删改查操作；
3. 管理员的需求：能够审批课程、统计查询所有课程和选课情况。
4. 本系统主要实现学生信息管理、课程信息管理和选课查询。

学生信息包括学号、性别、姓名、专业；

课程信息包括课程号、课程名、学时、学分、开课学期、课程简介、成绩、审批状态；

选课信息包括学号、课程号、成绩；

课程成绩统计表信息包括课程号、课程名、选课人数最高成绩、最低成绩、平均成绩、及格人数。

2.2 可行性分析

随着互联网的蓬勃发展，越来越多的机构将科研管理和Internet融合到一起，以便各高校管理。在这样的社会背景下，基于Windows和SQL server 2019，运用Java语言开发的学生管理系统。科学技术不断提高，计算机科学日渐程顺，因此技术可行。目标群体为高校的老师、管理员、学生，该系统将三者有机的结合在一起，有效的提高管理水平和效率。系统界面友好，操作简单，系统要求配置不高，应社会需求，有较好的社会前景。

2.3 系统运行环境

本系统运行环境为windows 10，开发平台为SQL server 2019和eclipse。

3 系统总体设计

根据需求分析，总共可分为四个模块，登录模块、老师模块、管理员模块、学生模块。然后根据选课流程和题目要求，安排每个模块工作内容和工作顺序，指定系统的工作流程。3.1 系统功能结构设计

通过上述功能需求分析，因为存在多种用户类型，所以需要增加用户登录的功能。老师有对课程进行增删改查的功能，管理员统计课程和选课情况，学生功能为选课和查询信息。系统功能模块划分如图3-1所示。

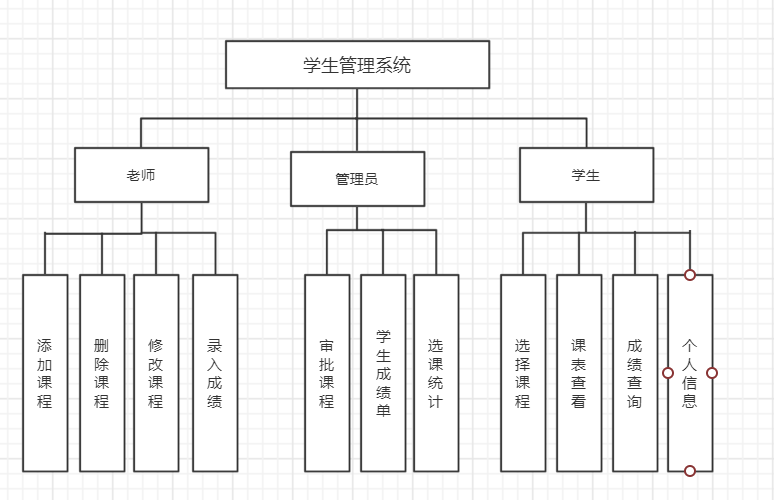


图3-1 系统功能结构图

3.2 系统功能流程设计

3.2.1删除课程功能

删除课程功能流程图如图3-2所示。

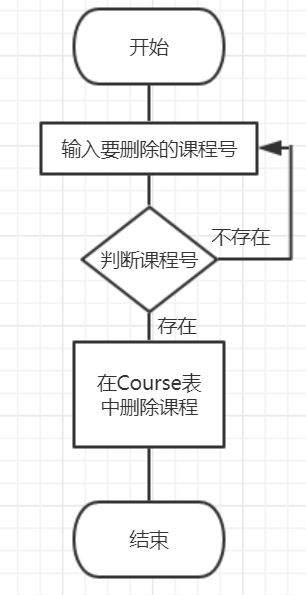


图3-2 删除课程功能流程图

3.2.2修改课程功能

修改课程功能流程图如图3-3所示。

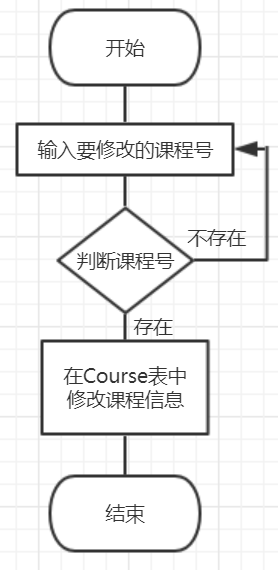


图3-3 修改课程功能流程图

3.2.3系统选课功能

系统选课流程如图3-4所示。

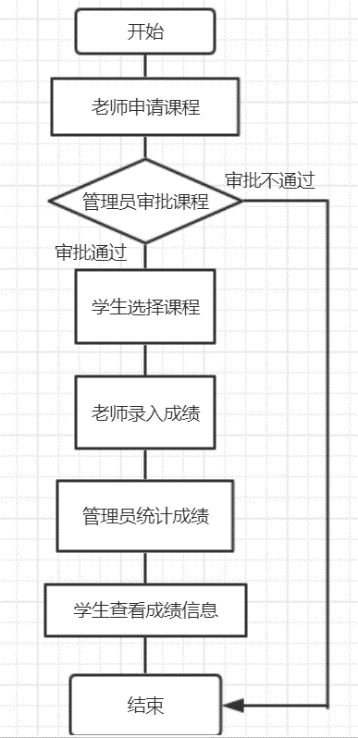


图3-4 系统选课流程图

3.3 数据库设计

设计一个选课数据库，能够实现学生自主选课功能，并能实现学生信息、课程信息、学生选课信息、学生成绩信息等的创建、更新、删除、修改，数据库完整性检查、冲突检查、查询优化以及数据库角色、权限的分配，实现数据库的备份等事务处理以保证数据库的完整性、一致性、安全性、可靠性。

（1）Student（学生表）

Student（学号，姓名，专业，性别），学号为主键，唯一且非空。姓名等三个普通属性，姓名和专业名不能为空。

学生表如表3-1所示。

表3-1 学生表结构说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 非空 | 主键 | 含义 |
| 学号 | varchar(10) | YES | YES | 学号 |
| 姓名 | varchar(10) | YES | NO | 姓名 |
| 专业名 | varchar(10) | YES | NO | 专业名 |
| 性别 | varchar(10) | NO | NO | 性别 |

（2）Course（课程表）

Course（课程号，课程名，学时，学分，开课学期，课程简介，是否通过审批），课程号为主键，唯一且非空。课程名等属性皆为普通属性，课程名不能为空。

课程表如表3-2所示。

表3-2课程表结构说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 非空 | 主键 | 含义 |
| 课程号 | varchar(10) | YES | YES | 课程号 |
| 课程名 | varchar(10) | YES | NO | 课程名 |
| 学分 | int | NO | NO | 学分 |
| 课时 | int | NO | NO | 课时 |
| 开课学期 | int | NO | NO | 开课学期 |
| 课程简介 | ntext | NO | NO | 课程简介 |
| 是否通过审批 | bit | NO | NO | 是否通过审批 |

（3）Stu\_Course（学生选课表）

Stu\_Course（学号，课程号，成绩），学号和课程号为主键，并且表Student和表Course为该表的主键表。学号和课程号，不可为空，且组合唯一，成绩可以为空。

学生选课表如表3-3所示。

表3-3学生选课表结构说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 非空 | 主键 | 含义 |
| 学号 | varchar(10) | YES | YES | 学号 |
| 课程号 | varchar(10) | YES | YES | 课程号 |
| 成绩 | int | NO | NO | 成绩 |

1. Grade\_count名（课程成绩统计表）

Grade\_count（课程号，课程名，选课人数，最高成绩，最低成绩，平均成绩，及格人数），该表以Course表和Stu\_Course表为主键表，课程、成绩不可超过两表范围。

学生选课表如表3-4所示。

表3-4课程成绩表结构说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 非空 | 主键 | 含义 |
| 课程号 | varchar(10) | YES | YES | 课程号 |
| 课程名 | varchar(10) | YES | NO | 课程名 |
| 选课人数 | int | NO | NO | 选课人数 |
| 最高成绩 | int | NO | NO | 最高成绩 |
| 最低成绩 | int | NO | NO | 最低成绩 |
| 平均成绩 | float | NO | NO | 平均成绩 |
| 及格人数 | int | NO | NO | 及格人数 |

（5）Login（登录表）

Login（用户名，密码）用户名为主键，两个属性皆不可为空。

登录表如表3-5所示。

表3-5登录表结构说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 非空 | 主键 | 含义 |
| 用户名 | varchar(10) | YES | YES | 用户名 |
| 密码 | varchar(10) | YES | NO | 密码 |

4系统详细设计

4.1 登录模块详细设计

首先在SQLsever2019上创建数据库XUANKE，接着在eclipse平台进行连接数据库操作，连接XUANKE。在数据库XUANKE中创建一个loging登录表，包含用户名和密码。在eclipse平台创建一个Windows应用程序项目，命名为学生选课管理系统。利用可视化编程技术设计登录界面。

4.2 老师模块详细设计

老师通过用户名/密码验证后，进入老师操作界面。界面设计四个Jbutton控件，分别命名为增加课程、修改课程、删除课程、录入成绩。设置事件监听，点击按钮，即可跳出相应的操作界面。增加、删除、修改课程部分，与数据库中Course表相连，进行插入，更新操作。录入成绩部分与数据库中Stu\_Course表相连，进行插入操作。

4.3 教务处管理员模块详细设计

管理员通过用户名/密码验证后，进入管理员操作界面。界面设计三个Jbutton控件，分别命名为审批课程、学生成绩单、选课统计。设置事件监听，点击按钮，即可跳出相应的操作界面。审批课程功能，与数据库中Course表相连，对课程属性中的审批状态进行修改。学生成绩单部分与数据库中Stu\_Course表相连，进行查询操作。

4.4 学生模块详细设计

管理员通过用户名/密码验证后，进入管理员操作界面。界面设计四个Jbutton控件，分别命名为选择课程、课表查看、成绩查询、个人信息。设置事件监听，点击按钮，即可跳出相应的操作界面。选择课程功能，与数据库中Stu\_Course表相连，进行查询、插入操作。课表查看部分对数据库中Stu\_Course表和Course表，进行嵌套查询，实现功能并将结果利用Jtable控件显示。成绩查询功能，对Grade\_count表信息进行查询，并将结果利用Jtable控件显示。个人信息部分对Student表信息进行查询，并将结果利用Jtable控件显示。

5系统实现

5.1 登录模块实现

打开登录界面，输入用户名和密码，账户2开头为学生账户，1开头为教师账户，0开头为管理员账户。选择身份进入系统。如果密码错误或者身份选择错误，不能进入系统。

登录实现界面如图5-1所示。



图5-1登录功能实现界面

登录功能的核心代码如下：

String name=textField.getText().trim();

String password=textField\_1.~~getText~~().trim();

**if**(name.equals("")||password.equals(""))//如果没输用户名或密码,则提示对不起,请输入用户名或密码

{

JOptionPane.*showMessageDialog*(frame,"对不起,请输入用户名或密码");

}**else**{//如果都有数据了就开始连接数据库验证

**try**{

**boolean**isMatch=**false**;

String password2=**null**;

String name\_label=(String)textField.getText();

String password\_lable=textField\_1.~~getText~~();

//数据库连接，执行sql语句

Connection connection=ConnectionFactory.*getConnection*();

Statement statement=connection.createStatement();

ResultSet resultSet=statement.executeQuery("SELECT \* FROM Login WHERE 用户名='"+name\_label+"';");

**if**(resultSet.next()){

String name1=(String)resultSet.getString("用户名");

String sql = "SELECT 密码 FROM Login WHERE 用户='"+name\_label+"';";

ResultSet set = statement.executeQuery(sql);

**if**(set.next()){

password2 = set.getString("密码").trim();

}

**if**(name1.equals(name\_label)){

**if**(password2.equals(password\_lable)){ **if**(Login.*flag1*==2&&name1.substring(0,1).equals("2")){

//name1为工号

*id*=name1;

Student student1= **new**Student();

student1.open();

setVisible(**false**);

System.***out***.println("111");

}

**elseif**(Login.*flag1*==1 &&name1.substring(0, 1).equals("1")){

//name1为工号

*id*=name1;

Teacher teacher= **new**Teacher();

teacher.open();

setVisible(**false**);

}**elseif**(Login.*flag1*==3 &&name1.substring(0, 1).equals("0")){

*id*=name;

Administrator administrator=**new**Administrator();

administrator.open();

}**else** {

JOptionPane.*showMessageDialog*(frame,"未选择身份或身份勾选错误!");}

}**else**{

JOptionPane.*showMessageDialog*(frame,"对不起,密码错误,请重新输入,登陆失败");}}

}

**else**{//如果没有查找到用户名就提示

JOptionPane.*showMessageDialog*(frame,"用户名不存在,请重新输入,错误!");}

}

**catch**(Exception ee){}

}

}

5.2 老师模块实现

老师进入系统，有四项功能，添加课程、修改课程、删除课程、录入成绩。

（1）老师功能的实现界面如图5-2所示。



图5-2 老师实现界面

（2）选择增加课程按钮，可进入增加界面，输入要增加的课程信息，若该课程已经存在，则会提示错误，然后可以重新输入信息。增加课程实现界面如图5-3所示。



图5-3 增加课程实现界面

增加课程功能的核心代码如下：

Statement statement=connection.createStatement();

ResultSet resultSet1=statement.executeQuery("SELECT \* FROM Course WHERE 课程号='"+textField.getText()+"';");

**if**(resultSet1.next()==**true**)

{

System.***out***.println(textField\_1.getText());

JOptionPane.*showMessageDialog*(frame,"该课程已存在,请重新输入!");

}**else** {

String kch=textField.getText();

String kcm=textField\_1.getText();

String xf1=textField\_2.getText();

**int**xf=Integer.*parseInt*(xf1);

String ks1=textField\_3.getText();

**int**ks=Integer.*parseInt*(ks1);

String kkxq=textField\_4.getText();

String kcjj=textField\_5.getText(); **if**(kch.equals("")||kcm.equals("")||xf1.equals("")||ks1.equals("")||kkxq.equals("")||kcjj.equals("")) {

JOptionPane.*showMessageDialog*(frame,"您还有必填项没有填写！");

}**else**{

String sql="INSERT INTO Course VALUES('"+kch+"','" +kcm+ "','" +xf+"','"+ks+"','"+kkxq+"','"+kcjj+"','False')";

PreparedStatement pst=**null**;

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

pst = connection.prepareStatement(sql);

pst.executeUpdate();

JOptionPane.*showMessageDialog*(frame,"添加成功!");//添加成功

（3）选择删除课程按钮，可进入删除界面，输入要删除的课程号，若该课程不存在，则会提示错误，然后可以重新输入要删除的课程号。删除课程实现界面如图5-4所示。

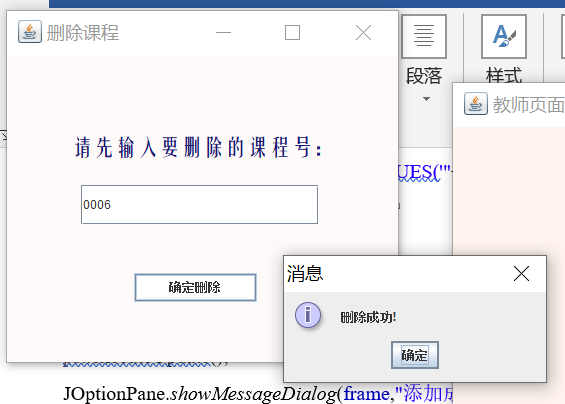


图5-4 删除课程实现界面

删除课程功能的核心代码如下：

Statement statement=connection.createStatement();

Statement statement2=connection.createStatement();

ResultSet resultSet2=statement2.executeQuery("SELECT \*FROM Stu\_Course WHERE 课程号='"+textField\_1.getText()+"';");

**if**(resultSet2.next()) {

String sql="DELETE FROM Stu\_Course WHERE 课程号

='"+textField\_1.getText()+"'";

PreparedStatement pst=**null**;

pst = connection.prepareStatement(sql);

pst.executeUpdate();

}

ResultSet resultSet1=statement.executeQuery("SELECT \*FROM Course WHERE 课程号='"+textField\_1.getText()+"';");

**if**(textField\_1.equals("")){

JOptionPane.*showMessageDialog*(frame,"您未填写要删除的课程号！");

}**elseif**(!resultSet1.next()){

JOptionPane.*showMessageDialog*(frame,"没有该课程,请重新输入！");

}**else**{

String sql="DELETE FROM Course WHERE 课程号='"+textField\_1.getText()+"'";

PreparedStatement pst=**null**;

pst = connection.prepareStatement(sql);

pst.executeUpdate();

JOptionPane.*showMessageDialog*(frame,"删除成功!");//删除成功

（4）选择修改课程按钮，可进入修改界面，输入要修改的课程号，重新编辑课程信息。若该课程不存在，则会提示错误，然后可以重新输入要修改的课程号。修改课程实现界面如图5-5、5-6所示。

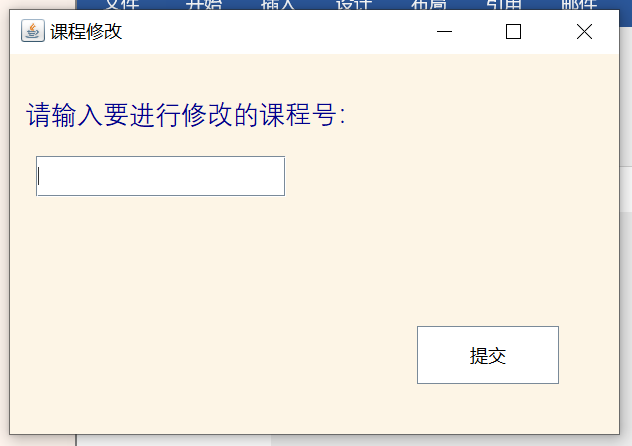


图5-5 修改课程实现界面1



图5-6 修改课程实现界面2

修改课程功能的核心代码如下：

String kcm\_label=textField.getText();

**int**xf\_label=Integer.*parseInt*(textField\_1.getText());

**int**ks\_label=Integer.*parseInt*(textField\_2.getText());

**int**kkxq\_label=Integer.*parseInt*(textField\_3.getText());

String kcjj\_label=textField\_4.getText();

Connection connection=ConnectionFactory.*getConnection*();

Statement statement=connection.createStatement();

String sql="UPDATE Course SET 课程名='"+kcm\_label+"',学分='"+xf\_label+"',课时='"+ks\_label+"',开课学期='"+kkxq\_label+"',课程简介='"+kcjj\_label+"' WHERE 课程号='"+*ModCourseNum1*+"';";

statement.executeUpdate(sql);

JOptionPane.*showMessageDialog*(frame,"修改成功！");

（5）选择录入成绩按钮，可进入成绩录入界面，填写对应的学号课程号以及成绩，也可对已有成绩进行修改。录入完毕后刷新成绩，即可显示最新成绩信息。录入成绩实现界面如图5-7所示。



图5-7 录入成绩实现界面

录入成绩功能的核心代码如下：

String xh=textField.getText();

String kch=textField\_1.getText();

String cj1=textField\_2.getText();

**int**cj=Integer.*parseInt*(cj1);

**try**{

Connection conn=ConnectionFactory.*getConnection*();

String sql1="update Stu\_Course SET 成绩='"+cj+"' WHERE 学号='"+xh+"' AND 课程号='"+kch+"'";

PreparedStatement pst=**null**;

pst=conn.prepareStatement(sql1);

pst.executeUpdate();

JOptionPane.*showMessageDialog*(frame,"提交成功!");

}**catch**(SQLException sqle){

JOptionPane.*showMessageDialog*(**null**,"数据操作错误","错误

",JOptionPane.***ERROR\_MESSAGE***);}

5.3 教务处管理员模块实现

教务处管理员进入系统，有两项功能，选课审批、学生成绩单。

（1）教务处管理员功能的实现界面如图5-8所示。

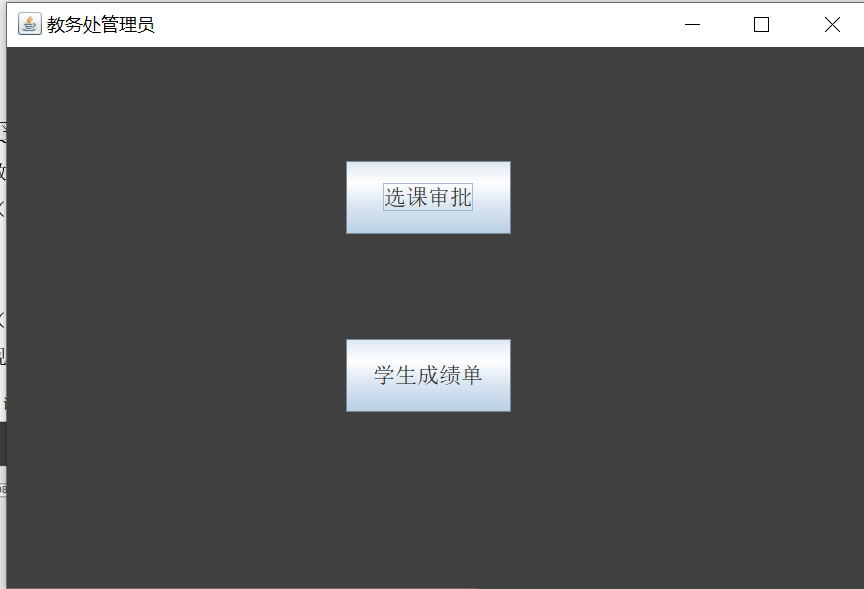


图5-8 教务处管理员实现界面

（2）选择选课审批按钮，进入审批界面，对老师申请的课程进行审批。审批课程功能的实现界面如图5-9所示。



图5-9 选课审批功能实现界面

选课审批功能的核心代码如下：

String kch=textField.getText();

**try**{

Connection conn=ConnectionFactory.*getConnection*();

String sql="UPDATE Course SET 是否通过审批='"+*flag*+"' WHERE 课程号='"+kch+"'";

PreparedStatement pstm=conn.prepareStatement(sql);

//执行

pstm.executeUpdate();

JOptionPane.*showMessageDialog*(frame,"操作成功!");

}**catch**(SQLException sqle){

}

（3）选择学生成绩单按钮，即可查看所有学生的成绩。学生成绩单功能的实现界面如图5-10所示。



图5-10 学生成绩单功能实现界面

显示学生成绩单功能的核心代码如下：

String sql="select \* from Stu\_Course";

PreparedStatement pstm=conn.prepareStatement(sql);

//执行查询

ResultSet rs=pstm.executeQuery();

//计算有多少条记录

**int**count=0;

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

count++;

}

rs=pstm.executeQuery();

//将查询获得的记录数据，转换成适合生成JTable的数据形式

Object[][] info=**new** Object[count][7];

count=0;

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

info[count][1]=rs.getString(1);//学号

info[count][2]=rs.getString(2);

info[count][3]=Integer.*toString*(rs.getInt(3));

//根据学号显示学生的姓名

String sql1="SELECT \* FROM Student WHERE 学号='"+info[count][1]+"';";

PreparedStatement pstm1=conn.prepareStatement(sql1);

//执行查询

ResultSet rs1=pstm1.executeQuery();

**if**(rs1.next()){

info[count][0]=rs1.getString(2);}

count++;

}

// 定义表头

String[] title={"学生姓名","学号","课程名称","成绩"};

// 创建JTable

**this**.tabDemo = **new** JTable(info,title);

// 显示表头

**this**.jth = **this**.tabDemo.getTableHeader();

**this**.scpDemo.setViewportView(tabDemo);

}**catch**(SQLException sqle){

JOptionPane.*showMessageDialog*(**null**,"数据操作错误","错误",JOptionPane.***ERROR\_MESSAGE***);

}

5.4 学生模块实现

学生进入系统，有四项功能，选择课程、课表查看、成绩查询、个人信息。

（1）学生功能的实现界面如图5-12所示。

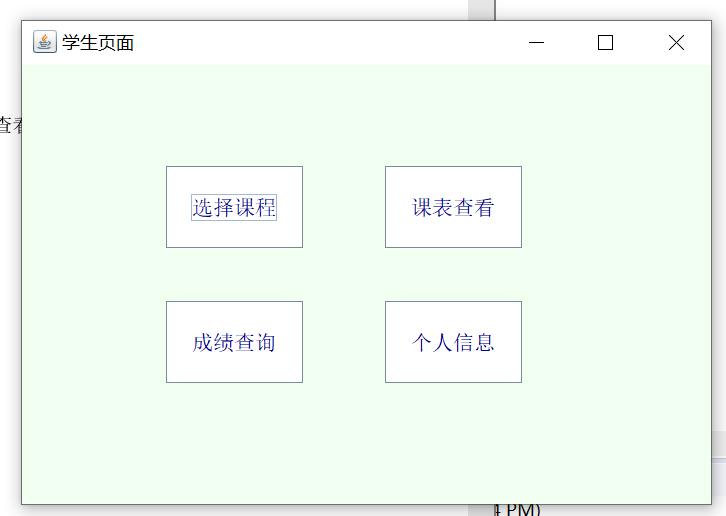


图5-12 学生实现界面

（2）学生点击选择课程按钮，就可看到已通过审批的课程，输入想要选择的课程的课程号，再确定选课，即可完成选课。选择课程功能的实现界面如图5-13所示。

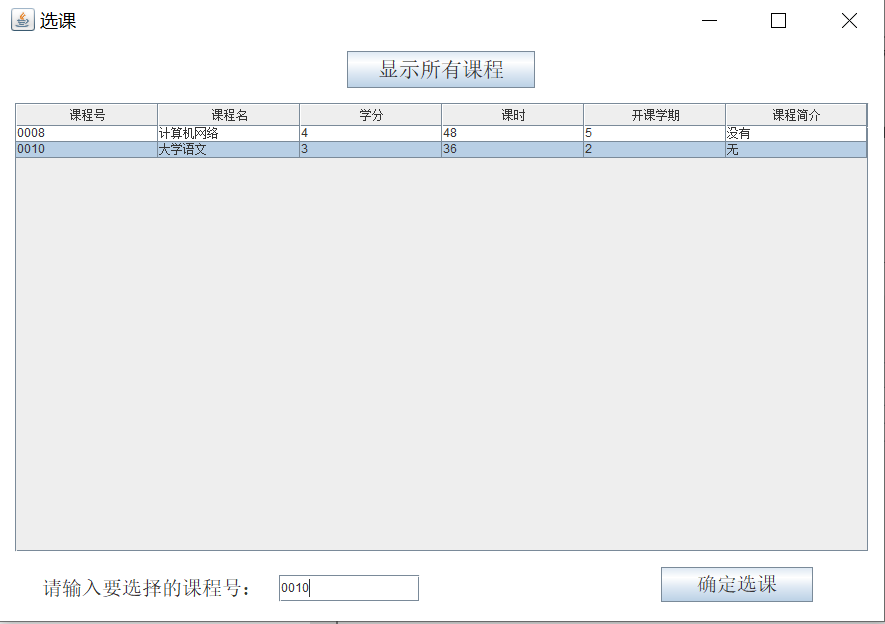


图5-13 选择课程实现界面

选课功能的核心代码如下：

Statement statement=connection.createStatement();

Statement statement1=connection.createStatement();

String sql="INSERT INTO Stu\_Course(学号,课程号) VALUES('"+*StuId\_Choose*+"','"+kch+"')";

String sql2="select \* from Course where 课程号='"+kch+"'";

PreparedStatement pstm=connection.prepareStatement(sql2);

ResultSet rs = pstm.executeQuery();

rs=pstm.executeQuery();

String CName =**null**;

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

CName=rs.getString(2);//将课程号存放在cno数组内

}

String sql1="INSERT INTO Grade\_count(课程号,课程名) VALUES('"+kch+"','"+CName+"')";

statement.executeUpdate(sql);

statement1.executeUpdate(sql1);

JOptionPane.*showMessageDialog*(frame,"添加成功!");

（3）学生点击课表查看按钮，就可看到已选上需要上课的的课程。课表查看功能的实现界面如图5-14所示。

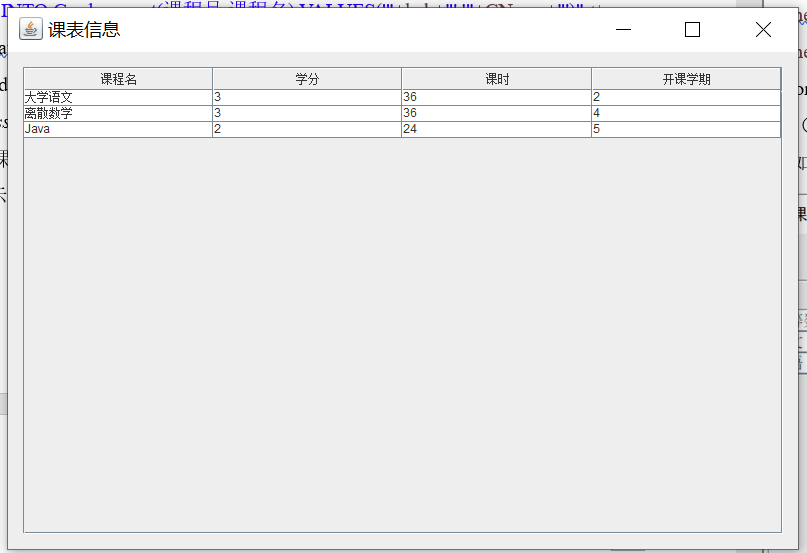


图5-14 课表查看实现界面

课表查看功能的核心代码如下：

String sql="select \* from Stu\_Course where 学号='"+*StuId*+"'";

PreparedStatement pstm=conn.prepareStatement(sql);

// 执行查询学号为StuId的课程

ResultSet rs=pstm.executeQuery();

rs=pstm.executeQuery();

String[] cno = **new** String[100];

// 计算学号为StuId有多少条课程记录

**int**count=0;**int**j=0;

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

count++;

cno[j++]=rs.getString(2);//将课程存放在cno数组内

}

//赋值i，进行循环

**int**i=count;

**int**Ccount=count;

Object[][] info1=**new** Object[Ccount][4];

Ccount=0;

**while**((i--)>0) {//循环，查询cno数组内每个课程号的信息

String sql1="select \* from Course where 课程号='"+cno[i]+"'";

PreparedStatement pst=conn.prepareStatement(sql1);

ResultSet rs1=pst.executeQuery();

} //将信息存入info1中

（4）学生点击成绩查询按钮，可看到课程成绩详情和我的成绩排名。点击课程成绩，可以看到每门课程的详细成绩。点击查看排名，可更直观的看出学生自己成绩如何。成绩查询功能的实现界面如图5-15、5-16所示。

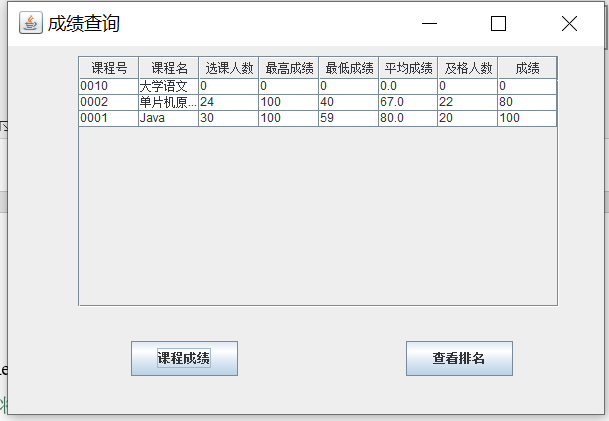


图5-15 成绩查询实现界面1



图5-16 成绩查询实现界面2

成绩查询功能的核心代码如下：

String sql="select \* from Stu\_Course where 学号='"+*StuId*+"'";

PreparedStatement pstm=conn.prepareStatement(sql);

// 执行查询学号为StuId的课程

ResultSet rs=pstm.executeQuery();

String[] cno = **new** String[100];

// 计算学号为StuId有多少条课程记录

**int**count=0;**int**j=0;

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

count++;

cno[j++]=rs.getString(2);//将课程号存放在cno数组内

}

//赋值i，进行循环

**int**i=count;

**int**Ccount=count;

Object[][] info1=**new** Object[Ccount][8];

Ccount=0;

**while**((i--)>0) {//循环，查询cno数组内每个课程号的信息

String sql1="select \* FROM Grade\_count where 课程号='"+cno[i]+"'";

Statement pst=conn.createStatement();

ResultSet rs1=pst.executeQuery(sql1);

//将信息存入info1中

**while**(rs1.next()){

info1[Ccount][0]=rs1.getString(1);

info1[Ccount][1]=rs1.getString(2);

info1[Ccount][2]=Integer.*toString*(rs1.getInt(3));

info1[Ccount][3]=Integer.*toString*(rs1.getInt(4));

info1[Ccount][4]=Integer.*toString*(rs1.getInt(5));

info1[Ccount][5]=rs1.getFloat(6);

info1[Ccount][6]=Integer.*toString*(rs1.getInt(7));

String sql2="select \* from Stu\_Course where 学号='"+*StuId*+"'AND 课程号='"+cno[i]+"'";

PreparedStatement pstm2=conn.prepareStatement(sql2);

// 执行查询学号为StuId的课程

ResultSet rs2=pstm2.executeQuery();

**if**(rs2.next()) {

info1[Ccount][7]=Integer.*toString*(rs2.getInt(3));}

Ccount++;

}}

排名部分：

**while**((i--)>0) {//循环，查询cno数组内每个课程号的信息

String sql1="select \* FROM Stu\_Course where 课程号='"+cno[i]+"' ORDER BY 成绩 DESC";

Statement pst=conn.createStatement();

ResultSet rs1=pst.executeQuery(sql1);

Ccount=0;

**while**(rs1.next()){

info1[Ccount][0]=rs1.getString(1);

System.***out***.println(info1[Ccount][0]);

info1[Ccount][1]=rs1.getString(2);

info1[Ccount][2]=Integer.*toString*(rs1.getInt(3));

Ccount++;

}

**for**(**int**n=0;n<Ccount;n++) {

**if**(info1[n][0].equals(*StuId*)) {

rank[m++]=n+1;}

}

}

String sql2="select \* from Stu\_Course where 学号='"+*StuId*+"'";

Statement pst2=conn.createStatement();

ResultSet rs2=pst2.executeQuery(sql2);

**while**(rs2.next()) {

info2[Ccount1][0]=rs2.getString(1);

info2[Ccount1][1]=rs2.getString(2);

info2[Ccount1][2]=Integer.*toString*(rs2.getInt(3));

info2[Ccount1][3]=rank[--m];

Ccount1++;

}

（5）学生点击个人信息按钮，可看到自己的一些基本信息，包含学号、专业等。个人信息查询功能的实现界面如图5-17所示。

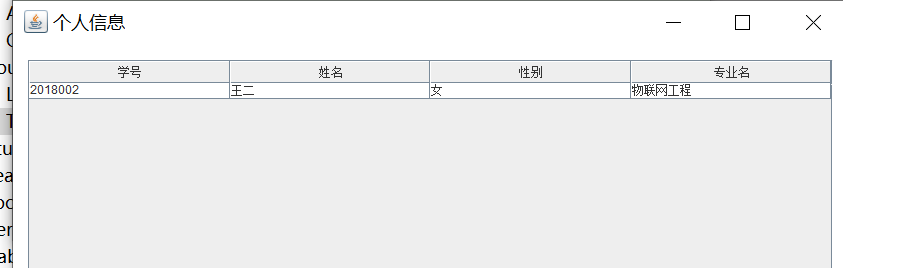


图5-17 个人信息查询功能实现界面

个人信息查询功能的核心代码如下：

String sql="select \* from Student where 学号='"+*StuId*+"'";

PreparedStatement pstm=conn.prepareStatement(sql);

// 执行查询学号为StuId的信息

ResultSet rs=pstm.executeQuery();

6 总结

通过这次Java课程设计的实践训练，我进一步了解了Java语言，熟悉并掌握了管理系统的设计的流程，而且提高了软件开发的实际能力，培养了自己工程设计能力和综合分析、解决问题的能力。在实训过程中，通过与同学的交流，我学习和实践了分析和设计学生管理系统的各种知识，包括系统的数据库设计、实施以及系统的分析、设计及编码和测试方面的知识。在深入理解课本理论知识的同时，加深掌握了Java语言的使用。课程设计任务最终圆满完成任务，本系统实现了关键的功能模块，在功能上基本满足了用户的需求，但是整个系统存在不完善的地方，如界面不太美观，操作性不太友好等。这也体现了自己能力的不足之处，在以后的学习以及工作中也是有待加强的一部分。总之，通过这次课程设计实训，我收获了许多。