**目录**

[第一章 概述 3](#_Toc15962)

[§1.1 项目背景 3](#_Toc19786)

[§1.2 项目意义 3](#_Toc29520)

[§1.3 开发环境 4](#_Toc24106)

[第二章 需求分析 4](#_Toc18056)

[§2.1 信息要求 4](#_Toc5020)

[§2.2处理要求 5](#_Toc22278)

[§2.3安全性与完整性要求 6](#_Toc5442)

[第三章 概念结构设计 6](#_Toc26578)

[第四章 逻辑结构设计 7](#_Toc5196)

[§4.1 由E-R图得出关系表 7](#_Toc32400)

[§4.2 由关系表构成数据库 7](#_Toc7192)

[第五章 主要的模块设计以及代码 8](#_Toc27700)

[§5.1 主要的模块设计 8](#_Toc2225)

[5.1.1 用户管理与用户登录功能 8](#_Toc14327)

[5.1.2教师模块 9](#_Toc23135)

[5.1.3学生模块 10](#_Toc26412)

[§5.2具体代码设计 10](#_Toc5800)

[5.2.1数据库的建立 10](#_Toc12471)

[5.2.2登录界面功能设计 11](#_Toc10870)

[5.2.2教师端的功能设计 12](#_Toc24872)

[5.2.3学生端的功能设计 15](#_Toc7734)

[第六章 总结 18](#_Toc7373)

[第七章 参考文献 18](#_Toc13361)

# 第一章 概述

§1.1 项目背景

随着高校规模的不断扩大，有关学生的各种信息量也成倍增长，各类学生的统计分析工作也越来越困难，面对如此繁杂的工作，我们需要为高校学生信息管理提供一种更加高效实用的管理手段，为学生信息的存储、计算、统计、分析、交流提供一种更加安全快捷的信息平台，并且能够减少大量的人工操作，以及在人工操作中由于人为因素而引起的数据错误，保证学生信息数据的安全性和完整性，使学生管理人员能够轻松,正确无误地完成各项工作，为学生管理工作服务。学生信息管理系统是一个教育单位不可缺少的部分,它的内容对于学校的决策者和管理者来说都至关重要,所以学生信息管理系统应该能够为用户提供充足的信息和快捷的查询手段。但一直以来人们使用传统人工的方式管理文件档案,这种管理方式存在着许多缺点,如:效率低、保密性差,另外时间一长,将产生大量的文件和数据,这对于查找、更新和维护都带来了不少的困难。

随着科学技术的不断提高,计算机科学日渐成熟,其强大的功能已为人们深刻认识,它已进入人类社会的各个领域并发挥着越来越重要的作用。作为计算机应用的一部分,使用计算机对学生信息进行管理,具有手工管理所无法比拟的优点。例如:检索迅速、查找方便、可靠性高、存储量大、保密性好、寿命长、成本低等。这些优点能够极大地提高学生信息管理的效率,也是企业的科学化、正规化管理,与世界接轨的重要条件。

§1.2 项目意义

1、保证信息的准确性和时效性，随时通过计算机对系统进行管理，使得学生信息处于最新状态，保证统计数据和分析的准确。

2、减轻劳动强度、提高工作效率、增加学生管理的透明度。高校学生管理信息系统可以发挥计算机的强大功能，让管理人员从大量繁琐的手工劳动中解放出来，将单调枯燥的纸张涂写变成灵活的电子信息操作。并充分利用网络优势,加快学校内部学生信息和文档的发布、传送和获取，改善和优化学生工作，使各个部门之间的工作联系紧密、井然有序、清楚明晰，大大减轻工作负担，提高工作效率，增加学生管理的透明度。

3、降低管理成本，由于采用本系统的原因，使辅导员管理工作一改往日埋在各种表格中的被动局面，基本实现了无纸化办公，根除了很多日常管理中的浪费现象，体现出现代化学生管理的优势。

4、规范化管理，由于采用了计算机统计分析学生信息，一些原始信息在录入时就必须要求准确录入，这就要求管理人员在提供信息时要准确，同时要求在录入时也要十分认真，这样做是为了将错误降到最低，提高系统工作的准确率，从而实现学生管理工作规范化。

§1.3 开发环境

1、软件环境：Eclipse、MySQL、Navicatformysql

2、开发工具：Eclipse 、tomcat7。4 、MySQL 、jdk8-64

# 第二章 需求分析

§2.1 信息要求

用户需要从数据库中获得信息的内容与性质。数据库中需要存储哪些数据。

设计内容：设计开发一个简单的“学生选课系统”，并实现基本的选课功能，询功能。

（1）实现学生基本信息的录入、修改、删除等操作，该部分数据库需要存储学生信息包括班号、学号、性别等；

（2）实现课程信息的录入、修改、删除等功能，该部分数据库需要存储课程信息包括课程号、课程名、学分等；

（3）实现选课功能，每个学生有选修学分限制，超出可选学分限制会提示，该部分所需存储数据在课程信息录入已有，无需重复存储；

（4）方便的对学生选课情况进行查询，可以根据学号、姓名、班级、课程名等多种方式查询；

（5）有汇总功能，能对每个学生的选修学分汇总，并对选修学分不足的学生进行筛选，该部分存储的数据是学生选课的学分。

**详细信息要求**

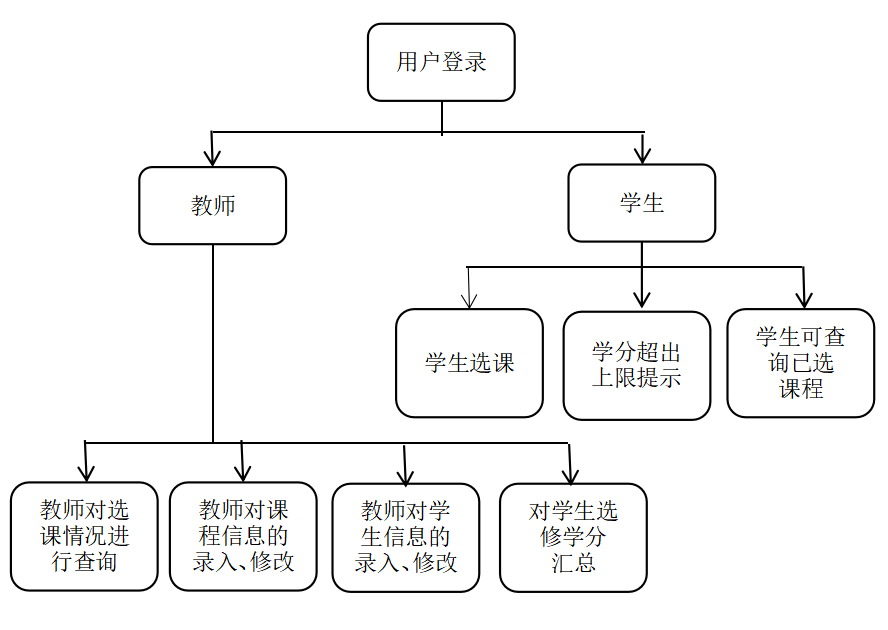
对系统功能分析，应包含以下功能：

1. 身份验证，用以用户登录，分别有教师用户与学生用户；
2. 教师用户可以录入、修改、删除学生信息、课程信息；
3. 教师用户可以按编号、名称、等多种方式查询学生信息、课程信息；
4. 教师用户可以统计学分，并可筛选出选修学分不够的同学；
5. 学生用户可以查看可选科目、并选课；
6. 当学分超出一个学生学分上限时，发出提示，并且不能继续选课；

学生选课结束，可以查看已选课程；

§2.2处理要求

用户需要完成什么处理功能，对处理的响应时间有什么要求。（给出功能模块图）



用户验证登录模块

用户根据自己身份，选择相应的登录选项（教师或学生）登录，与数据库中数据相验证，若用户存在，则跳转到相应页面；若用户不存在，则提示密码或用户名错误，跳回登录页面。

教师对课程信息的操作模块

教师可以添加课程、删除课程、修改课程，查询课程。修改、删除、添加模块应有修改成功的提示，查询页面应有分页显示功能。

教师对学生信息的操作模块

教师可以添加、修改，删除、查询学生信息。对学生信息的修改、删除、添加操作应该有修改成功的提示，对学生的查询应该可以分页显示。

模糊查询功能

教师用户可以根据某一个数据项查询学生或课程信息，或是查询含有某一字符的相关元组。

学分控制模块

本模块可统计选课学分不足或是学分超出上限的学生。

学生选课模块

学生可查看所有课程，可选择选课人数未满课程。选课前要有确认选项。选课后不能更改。成功选课后，选课对该学生关闭。该学生只能看到已选课程以及课程信息。

§2.3安全性与完整性要求

**安全性要求**

学生选课管理系统是直接面对使用人员的，这就要求系统能够提供良好的用户接,易用的人机交互界面.要实现这一点，就要求系统应该尽量使用用户熟悉的术语和中文信息的界面；针对用户可能出现的使用问题，要提供足够的在线帮助,缩短用户对系统熟悉的过程。

学生信息管理系统中涉及到的数据是学生的重要的信息，系统要提供方便的手段供系统维护人员进行数据的备份,日常的安全管理,系统意外崩溃时数据的恢复等工作。

**完整性要求**

学生选课管理系统的使用对象主要是教师、学生。基于方便性，在学生作为用户注册时需要完善自己的信息，将自己的信息填写完整，以便于教师在日后查询相关信息可以依据相关字段进行查询。对于教师，教师需要写明自己课程名称、学分等相关信息，课程信息的完整性才能够方便学生在进行选课的时候做出正确的选择。

# 第三章 概念结构设计

本系统的E-R图如图图3.1所示。

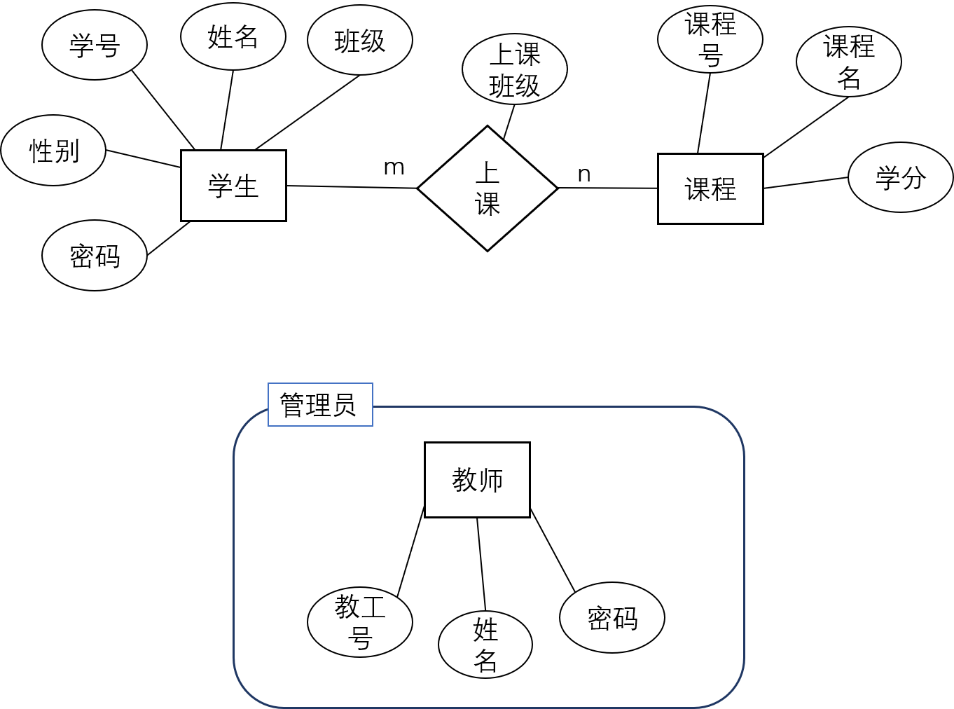


图3.1 学生选课系统E-R图

# 第四章 逻辑结构设计

§4.1 由E-R图得出关系表

1.学生信息表student(Sno ,Sname ,Spassword ,Sclass ,Ssex)

2.课程信息表course(Cno ,Cname ,Ccredit)

3.教师信息表teacher(Tno ,Tname ,Tpassword)

4.选课信息表sc(Sno ,Cno ,classroom)

§4.2 由关系表构成数据库

数据库中各表及其相应字段分别描述如下：系统数据库（SCDB）中主要包含以下四个数据表，各表及其相应字段分别描述如下：

1．学生信息表（student）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名 | 类型 | 长度 | 不是null | 其他 |
| SNo | int | 11 | √ | 主键 |
| Sname | varchar | 30 | √ |  |
| Spassword | varchar | 30 | √ |  |
| Sclass | varchar | 30 |  |  |
| Ssex | varchar | 10 |  |  |

表4.1 学生信息表

2.课程信息表（course）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名 | 类型 | 长度 | 不是null | 其他 |
| CNo | int | 11 | √ | 主键 |
| Cname | varchar | 30 | √ |  |
| Ccredit | varchar | 5 | √ | 默认0 |

表4.2 课程信息表

3.教师信息表（teacher）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名 | 类型 | 长度 | 不是null | 其他 |
| TNo | int | 11 | √ | 主键 |
| Tname | varchar | 30 |  |  |
| Tpassword | varchar | 30 |  |  |

表4.3 教师信息表

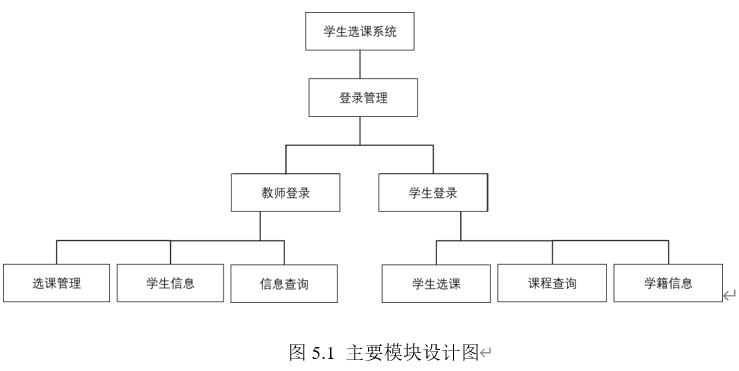
4、选课信息表（sc)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名 | 类型 | 长度 | 不是null | 其他 |
| SNo | int | 11 | √ | 外键 |
| CNo | varchar | 11 | √ | 外键 |
| classroom | varchar | 11 |  |  |

表4.4 选课信息表

# 第五章 主要的模块设计以及代码

§5.1 主要的模块设计



### 5.1.1 用户管理与用户登录功能

1、教师可根据需要添加、删除学生用户，并可对已有的学生信息进行修改操作；

2、在添加新的学生用户时，可以设置学生用户的密码；

3、各种不同身份的用户登录系统后，可享有不同的系统操作权限。

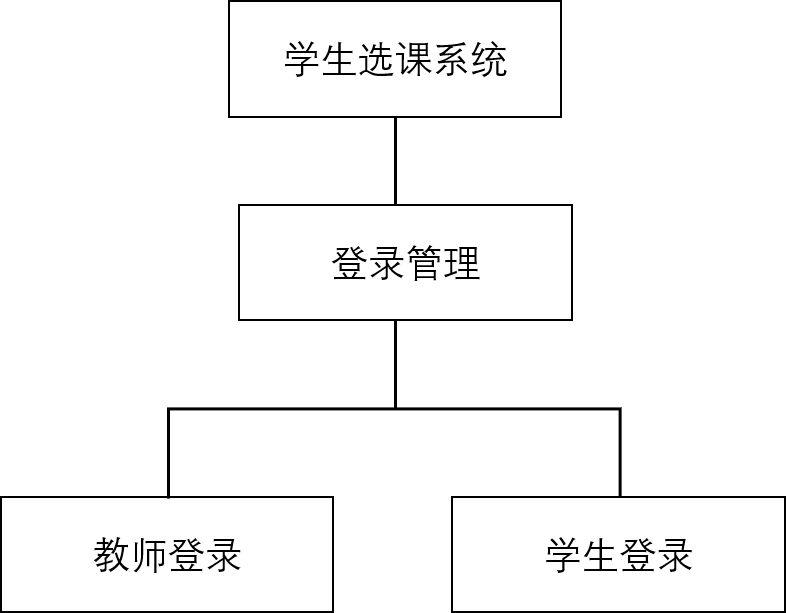


图5.1.1 登录层次图

### 5.1.2教师模块

（1）选课管理

1.查看课程信息

2.添加课程

3.修改课程

4.删除课程

（2）学生信息

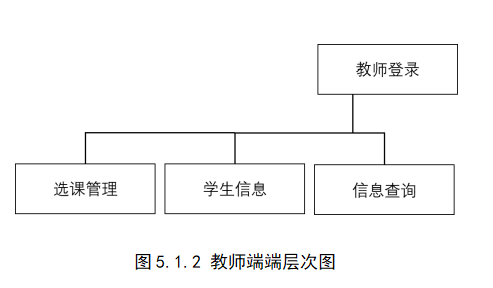
1.查看学生信息

2.修改学生信息

3.删除学生信息

4.筛选符合条件的学生

5.添加学生（新增学生用户）



（3）信息查询

1.根据学号查询

2.根据姓名查询

3.根据班级查询

4.根据课程号查询

5.根据课程名查询

6.根据上课教室查询

7.修改课室

### 5.1.3学生模块

（1）学生选课

（2）课程查询

1.查看已选课程信息

2.已选学分超额提示

3.取消选课

（3）学籍信息

1.查询个人信息

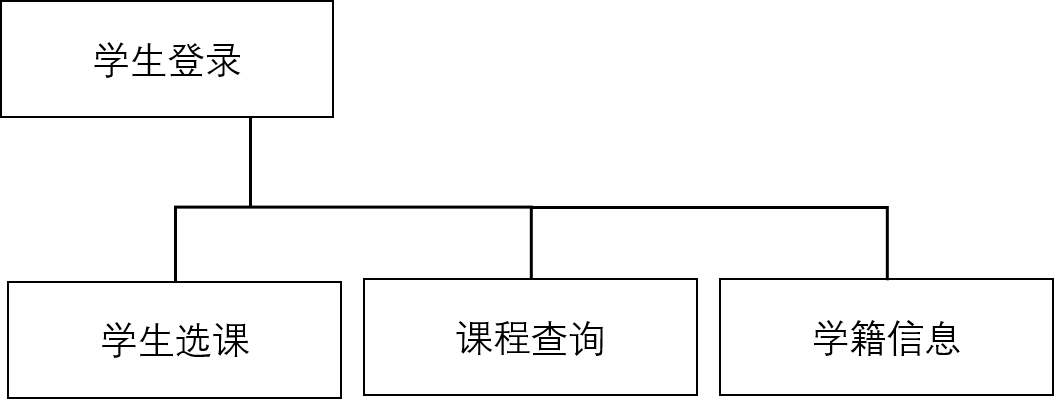


图5.1.3学生端层次图

**§5.2具体代码设计**

### 5.2.1数据库的建立

#### 1.创建表所用的SQL语句

（1）教师信息表创建：

CREATE TABLE `teacher` (

`TNo` int(11) NOT NULL AUTO\_INCREMENT,

`Tname` varchar(30) DEFAULT NULL,

`Tpassword` varchar(30) DEFAULT NULL,

PRIMARY KEY (`TNo`)

)

（2）课程信息表创建：

CREATE TABLE `course` (

`CNo` int(11) NOT NULL AUTO\_INCREMENT,

`Cname` varchar(30) NOT NULL,

`Ccredit` float(5,1) DEFAULT '0.0',

PRIMARY KEY (`CNo`)

)

（3）学生信息表创建：

CREATE TABLE `student` (

`SNo` int(11) NOT NULL AUTO\_INCREMENT,

`Sname` varchar(30) NOT NULL,

`Spassword` varchar(30) NOT NULL,

`Sclass` varchar(30) DEFAULT '',

`Ssex` varchar(10) DEFAULT '',

PRIMARY KEY (`SNo`)

)

（4）选课信息表创建：

CREATE TABLE `sc` (

`SNo` int(11) NOT NULL,

`CNo` int(11) NOT NULL,

`classroom` varchar(11) DEFAULT '0.0',

PRIMARY KEY (`SNo`,`CNo`)

)

#### 2.数据库的连接与获取数据库

public class Conn {

Connection conn = null;

public static final String DBDRIVER="com.mysql.jdbc.Driver";

public static final String DBURL="jdbc:mysql://localhost:3306/db\_select\_course?useUnicode=true&characterEncoding=utf8";

//指定连接本地数据库中的的db\_select\_course数据库

public static final String DBUSER="root";//账户

public static final String DBPASS="root";//密码

//获取连接

public Connection getConn()

{

try{

Class.forName(DBDRIVER);

conn = DriverManager.getConnection(DBURL,DBUSER,DBPASS);

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

return conn;

}

}

### 5.2.2登录界面功能设计

#### 1.登录功能

学生信息管理系统前台登录模块只有一个登录窗口分为学生、老师、两个级别登录,系统在后台程序有自动限制设置,可以自动识别登录者的限制.其系统登录模块算法如下：

1.输入用户名和密码;

2.判断用户名和密码是否匹配

（1）若匹配，则进入系统

（2）若提示输入信息错误,则返回登录界面重新登录；

#### 2.具体代码

throws ServletException, IOException {

response.setContentType("text/html;charset=utf-8");

HttpSession session = request.getSession();

PrintWriter out = response.getWriter();

//response.setCharacterEncoding("utf-8");

//response.setHeader("content-type","text/html;charset=utf-8");

try{

int username = Integer.parseInt(request.getParameter("username"));

String password = request.getParameter("password");

String type = null;

if(request.getParameter("type")!=null)

type = request.getParameter("type");

if(type.equals("tea")){//判断是否为老师

Teacher tea = new Teacher(username,null,password);

DaoTea logint = new DaoTea();

tea = logint.loginTea(tea);

if(tea!=null){

session.setAttribute("teacher", tea);

response.sendRedirect("/Ten/tea/teacher.jsp");

}else{

out.println("T用户名或密码不正确，请核对后重试!!");

response.setHeader("refresh","2;url=/Ten/tea/teaLogin.jsp");

}

}else if(type.equals("stu")){ //判断是否为学生

Student stu = new Student(username,null,password,null,null,0);

out.println(stu.getSname()+"::"+stu.getSpassword()+"<br>");

out.println(stu.getSname()+"::"+stu.getSpassword()+"<br>");

DaoStu logins = new DaoStu();

stu = logins.loginStu(stu);

if(stu!=null){

session.setAttribute("student", stu);

response.sendRedirect("/Ten/stu/student.jsp");

}else{

out.println("S用户名或密码不正确，请核对后重试!!");

response.setHeader("refresh","2;url=/Ten/Login.jsp");

}

}else{

out.println("错误！请登录！");

response.setHeader("refresh","2;url=/Ten/Login.jsp");

}

}catch(Exception e){ e.printStackTrace();response.sendRedirect("/Ten/Login.jsp");}

out.flush();

out.close();

}

### 5.2.2教师端的功能设计

#### 1.添加学生信息功能

学生信息管理系统对学生信息的管理非常重要,其中对信息的录入是系统最为关键的地方,以往我们管理学生信息的时候都是手工操作,而随社会不断发展的今天计算机的应用已全部取带了手工操作的方法,利用计算机可以方便的录入各类信息,进行高效的管理.学生信息的录入就是利用计算机通过程序读入数据库,录入学生信息模块算法描述如下：

1)教师登录后,录入学生信息模块；

2)确认后,学生信息添加成功.

#### 2.添加新课程功能

学生选课模块主要是给学生选择课程的功能,本功能主要由教师在后台发布自己的课程信息进入数据库,然后学生登录系统,可以看见所有发布的课程信息,然后学生再进行选择,学生选择的课程信息会自动进入后台数据库,其学生选课模块算法描述如下：

1)教师成功登录；

2)点击添加新课程；

3)输入课程号；

4)输入课程名；

5)点击提交按钮,完成添加新课程操作.

#### 3.修改学生信息功能

此信息修改模块主要是给学生和教师建立信息库,方便教师查询操作,信息修改模块算法描述如下：

1)教师成功登录；

2)点击修改学生信息；

3)输入学生的编号；

4)输入正确,进入修改页面进行相关修改；

#### 4.修改课程信息功能

此信息修改模块主要是给课程和教师建立信息库,方便教师查询操作,信息修改模块算法描述如下：

1)教师成功登录；

2)点击修改课程信息；

3)输入课程的编号；

4)输入正确,进入修改页面进行相关修改;

#### 5.具体代码

public class DaoQuery {

/\*

\* 根据学号查询选课情况

\* \*/

public List<QueryResult> selectSno(int sno){

List<QueryResult> list = null;

try{

list = new ArrayList<QueryResult>();

String sql = "select student.sno,sname,sclass,course.cno,cname,classroom from student,sc,course " +

"where student.sno=sc.sno and sc.cno=course.cno and student.sno=?;";

Connection conn = new Conn().getConn();

PreparedStatement pst = conn.prepareStatement(sql);

pst.setInt(1, sno);

ResultSet rs = pst.executeQuery();

while(rs.next()){

QueryResult qrs = new QueryResult();

qrs.setSNo(rs.getInt("sno"));

qrs.setCNo(rs.getInt("cno"));

qrs.setSname(rs.getString("sname"));

qrs.setCname(rs.getString("cname"));

qrs.setSclass(rs.getString("sclass"));

qrs.setClassroom(rs.getString("classroom"));

list.add(qrs);

}

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

return list;

}

/\*

\* 根据学生姓名查询选课情况

\* \*/

public List<QueryResult> selectSname(String name){

List<QueryResult> list = null;

try{

list = new ArrayList<QueryResult>();

String sql = "select student.sno,sname,sclass,course.cno,cname,classroom from student,sc,course " +

"where student.sno=sc.sno and sc.cno=course.cno and student.sname=?;";

Connection conn = new Conn().getConn();

PreparedStatement pst = conn.prepareStatement(sql);

pst.setString(1, name);

ResultSet rs = pst.executeQuery();

while(rs.next()){

QueryResult qrs = new QueryResult();

qrs.setSNo(rs.getInt("sno"));

qrs.setCNo(rs.getInt("cno"));

qrs.setSname(rs.getString("sname"));

qrs.setCname(rs.getString("cname"));

qrs.setSclass(rs.getString("sclass"));

qrs.setClassroom(rs.getString("classroom"));

list.add(qrs);

}

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

return list;

}

/\*

\* 根据班级查询选课情况

\* \*/

public List<QueryResult> selectSclass(String \_class){

List<QueryResult> list = null;

try{

list = new ArrayList<QueryResult>();

String sql = "select student.sno,sname,sclass,course.cno,cname,classroom from student,sc,course " +

"where student.sno=sc.sno and sc.cno=course.cno and student.sclass=?;";

Connection conn = new Conn().getConn();

PreparedStatement pst = conn.prepareStatement(sql);

pst.setString(1, \_class);

ResultSet rs = pst.executeQuery();

while(rs.next()){

QueryResult qrs = new QueryResult();

qrs.setSNo(rs.getInt("sno"));

qrs.setCNo(rs.getInt("cno"));

qrs.setSname(rs.getString("sname"));

qrs.setCname(rs.getString("cname"));

qrs.setSclass(rs.getString("sclass"));

qrs.setClassroom(rs.getString("classroom"));

list.add(qrs);

}

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

return list;

}

/\*

\* 根据课程号查询选课情况

\* \*/

public List<QueryResult> selectCno(int cno){

List<QueryResult> list = null;

try{

list = new ArrayList<QueryResult>();

String sql = "select student.sno,sname,sclass,course.cno,cname,classroom from student,sc,course " +

"where student.sno=sc.sno and sc.cno=course.cno and course.cno=?;";

Connection conn = new Conn().getConn();

PreparedStatement pst = conn.prepareStatement(sql);

pst.setInt(1, cno);

ResultSet rs = pst.executeQuery();

while(rs.next()){

QueryResult qrs = new QueryResult();

qrs.setSNo(rs.getInt("sno"));

qrs.setCNo(rs.getInt("cno"));

qrs.setSname(rs.getString("sname"));

qrs.setCname(rs.getString("cname"));

qrs.setSclass(rs.getString("sclass"));

qrs.setClassroom(rs.getString("classroom"));

list.add(qrs);

}

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

return list;

}

/\*

\* 根据课程名查询选课情况

\* \*/

public List<QueryResult> selectCname(String cname){

List<QueryResult> list = null;

try{

list = new ArrayList<QueryResult>();

String sql = "select student.sno,sname,sclass,course.cno,cname,classroom from student,sc,course " +

"where student.sno=sc.sno and sc.cno=course.cno and course.cname=?;";

Connection conn = new Conn().getConn();

PreparedStatement pst = conn.prepareStatement(sql);

pst.setString(1, cname);

ResultSet rs = pst.executeQuery();

while(rs.next()){

QueryResult qrs = new QueryResult();

qrs.setSNo(rs.getInt("sno"));

qrs.setCNo(rs.getInt("cno"));

qrs.setSname(rs.getString("sname"));

qrs.setCname(rs.getString("cname"));

qrs.setSclass(rs.getString("sclass"));

qrs.setClassroom(rs.getString("classroom"));

list.add(qrs);

}

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

return list;

}

/\*

\* 根据上课教室查询选课情况

\* \*/

public List<QueryResult> selectRoom(String room){

List<QueryResult> list = null;

try{

list = new ArrayList<QueryResult>();

String sql = "select student.sno,sname,sclass,course.cno,cname,classroom from student,sc,course " +

"where student.sno=sc.sno and sc.cno=course.cno and classroom=?;";

Connection conn = new Conn().getConn();

PreparedStatement pst = conn.prepareStatement(sql);

pst.setString(1, room);

ResultSet rs = pst.executeQuery();

while(rs.next()){

QueryResult qrs = new QueryResult();

qrs.setSNo(rs.getInt("sno"));

qrs.setCNo(rs.getInt("cno"));

qrs.setSname(rs.getString("sname"));

qrs.setCname(rs.getString("cname"));

qrs.setSclass(rs.getString("sclass"));

qrs.setClassroom(rs.getString("classroom"));

list.add(qrs);

}

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

return list;

}

### 5.2.3学生端的功能设计

#### 1.学生选课功能

学生可以进行网上选课,在每学期快要结束时,学校都会把下学期要修的课程放到学校网站上,供同学们进行选课,这样节省了大家的宝贵时间,也方便了老师的管理,方便又实用。学生选课功能算法描述如下：

1)学生成功登录；

2)点击选定课程按钮,进入选定课程的页面；

3)在页面中点击添加课程按钮；

4)点击想要选择的课程的复选框,进行选择；

5)然后点击提交按钮；

6)课程选择成功.

#### 2.查看学生自身学籍功能

学生信息管理系统方便了学生查看自己的信息,如果学校由于疏忽输入了错误信息,自己可以查看并且及时通知学校教师,及时修改,减少不必要的麻烦。算法描述如下：

1)学生成功登号；

2)点击查询学生信息；

3)输入学生的编信息功能

4)输入正确,进入信息查询页面进行相关修改.

#### 3.查看已选课程功能

学生可以查看本学期已选课程,并且可以方便的知道所选课程的教课老师和上课地点等,这样方便了学校专员,方便了老师和我们学生自己,需要的时候从网上下载课程表,随时看都可以。查看已选课程算法描述如下：

1)学生成功登录；

2)点击选定课程；

3)右边显示出本学期已选定的课程信息；

4)如果想继续选课,点击添加课程即可；

#### 4.删除已选课程功能

学生可以进行网上选课,就可以删除已选课程,比如说不小心选错了某一门课程,就必须在规定的时间里删除掉,然后重新选择要修的课程,因为有时粗心是不可避免的,所以有删除的功能是必须,这样既方便了老师的管理,又方便了自己。

1)学生成功登录；

2)选定课程；

3)右边显示出本学期已选定的课程信息,选择想要删除的课程；

4)单击课程后面的删除按钮即可；

#### 5.具体代码

public class DaoStu {

/\*\*

\* 学生登录

\* \*\*/

public Student loginStu(Student stu){

Student stu1 = null;

String sql\_loginS="select \* from student where SNo=? and Spassword=?;";

try{

Connection conn = new Conn().getConn();

PreparedStatement pst = conn.prepareStatement(sql\_loginS);

pst.setInt(1, stu.getSNo());

pst.setString(2,stu.getSpassword());

ResultSet rs = pst.executeQuery();

if(rs.next()){

stu1 = new Student(rs.getInt(1),rs.getString(2),rs.getString(3),rs.getString(4),rs.getString(5),0);

System.out.println("student login select over");

}

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

return stu1;

}

/\*

\* 查询所有学生

\* \*/

public List<Student> selectStu(){

List<Student> list = null;

try{

String sql\_select = "select SNo,Sname,Spassword,Sclass,Ssex from student;";//查询全部学生信息+学分学分;//查询所有的sql语句

Connection conn = new Conn().getConn();

PreparedStatement pst = conn.prepareStatement(sql\_select);

ResultSet rs = pst.executeQuery();

list = new ArrayList<Student>();

DaoCou scredit = new DaoCou();

while(rs.next()){

Student stu = new Student(rs.getInt("SNo"),rs.getString("Sname"),rs.getString("Spassword"),rs.getString("Sclass")

,rs.getString("Ssex"), scredit.selectScre(rs.getInt("SNo")));

list.add(stu);

}

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

return list;

}

/\*

\* 查询学生总学分\_某位学生

\* \*/

public Student selectScre(Student stu){

ResultSet rs = null;

try{

String sql\_insert = "select student.sno as sno,sum(course.ccredit) as scredit from student,sc,course" +

" where student.sno=sc.sno" +

" and sc.cno=course.cno" +

" and student.sno=?";//查询某位同学学分

Connection conn = new Conn().getConn();

PreparedStatement pst = conn.prepareStatement(sql\_insert);

pst.setInt(1,stu.getSNo());

rs = pst.executeQuery();

if(rs.next()){

System.out.println("Student\_id"+stu.getSNo()+"select one over!");

stu = new Student(rs.getInt("sno"),null,null,null,null,rs.getFloat("scredit"));

}

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

return stu;

}

/\*

\* 查询某条个学生

\* \*/

public Student selectStu(int id){

Student Stu = null;

try{

String sql\_selectone = "select \* from student where SNo=?";//查询某一条课程

Connection conn = new Conn().getConn();

PreparedStatement pst = conn.prepareStatement(sql\_selectone);

pst.setInt(1, id);

ResultSet rs = pst.executeQuery();

if(rs.next()){

Stu = new Student(rs.getInt("SNo"),rs.getString("Sname"),rs.getString("Spassword"),rs.getString("Sclass")

,rs.getString("Ssex"),0);

}

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

return Stu;

}

/\*

\* 修改某个学生信息

\* \*/

public int updateStu(Student stu){

int rs = 0;

try{

String sql\_update = "update Student set Sname=?,Spassword=?,Sclass=?,Ssex=? where SNo=?;";//修改一条学生信息

Connection conn = new Conn().getConn();

PreparedStatement pst = conn.prepareStatement(sql\_update);

pst.setString(1, stu.getSname());

pst.setString(2, stu.getSpassword());

pst.setString(3, stu.getSclass());

pst.setString(4, stu.getSsex());

pst.setInt(5, stu.getSNo());

rs = pst.executeUpdate();

if(rs!=0){

System.out.println("Student\_id"+stu.getSNo()+"update over!");

}

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

return rs;

}

/\*

\* 删除某个学生信息

\* \*/

public int deleteStu(int id){

int rs = 0;

try{

String sql\_delete = "delete from student where SNo=?;";//删除一条课程信息

Connection conn = new Conn().getConn();

PreparedStatement pst = conn.prepareStatement(sql\_delete);

pst.setInt(1, id);

rs = pst.executeUpdate();

if(rs!=0){

System.out.println("Student\_id"+id+"delete over!");

}

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

return rs;

}

/\*

\* 添加一位学生信息

\* \*/

public int insertStu(Student stu){

int rs = 0;

try{

String sql\_insert = "insert into student(Sname,Spassword,Sclass,Ssex) values(?,?,?,?);";//添加一条课程信息

Connection conn = new Conn().getConn();

PreparedStatement pst = conn.prepareStatement(sql\_insert);

pst.setString(1,stu.getSname());

pst.setString(2, stu.getSpassword());

pst.setString(3, stu.getSclass());

pst.setString(4, stu.getSsex());

rs = pst.executeUpdate();

if(rs!=0){

System.out.println("Student\_id"+stu.getSNo()+"insert over!");

}

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

return rs;

}

}

# 第六章 总结

这次的数据库课程设计，老师也给了充裕的时间让我们去准备，我们组这次要完成的项目是学生选课系统。在这次项目的数据库设计中，由于我们大二下学期学过数据库基本原理，但是我们的主要目标是利用课程中学到数据库知识和技术较好开发设计出数据库应用系统，去解决各行各业信息化处理要求。对我们来说还是有一定难度和挑战的。因此在前期我们一直在网上找视频自学，并确定好了选用java语言。为了方便和美观，我们最终决定了用前端加后端的路线来实现。因为考虑到每位组员都会有不同的意见和建议，所以创建表的过程我们是以组为单位一起进行的。先把自己能想到的需要用到的数据表画出来，并根据需求暂时给出数据表的几个大字段，再归结成大字段，并在之后的进一步分析中再将其一一分解。照这样的方法依次创建了项目中会用到的一些主要的数据表，包括学生信息表，课程信息表，教师信息表，选课信息表，然后再给每个表添加一些属性，并设置其数据类型、长度，最重要的是选一个属性作为主键，最后再参照第一范式，第二范式和第三范式对已经创建的表进行拆分。

完成好数据库后，我们开始分配任务，按照“web界面->tomcat服务器->Java操作代码->jdbc->数据库”的顺序，完善我们的系统，我们的问题主要是在后端的编写上面，因为有些组员到实际开发过程中可能会出现一些问题，所以有时会出现自行修改了数据库的部分表结构。就会导致出现了不同的数据库，如果不及时通知后端的话，很有可能后端完成的部分接口都要重写，这就加大了后端的工作任务，进而会降低小组整体的工作效率。因此我们经常遇到问题就一起讨论，最终决定好解决问题的方法后，再去改写后端。

通过这次课程设计，感受到了一个小组的力量，在设计中一起探讨问题,及时的发现问题把设计顺利的进行下去。同时巩固我们对数据库基本原理和基础理论理解，掌握数据库应用系统设计开发基本方法，进一步提高我们综合运用所学知识能力。

# 第七章 参考文献

[1] 陈刚．MyEclipse从入门到精通［Ｍ］．北京：清华大学出版社, 2005年4月.

[2] 孙卫琴. Tomcat与Java Web开发技术详解[M]. 第2版. 电子工业出版社, 2009年01月.

[3] 刘京华 等. Java Web整合开发王者归来[M]. 清华大学出版社, 2010年01月.

[4] 萨师煊，王珊.《数据库系统概论》.高等教育出版社,2002年2月.

[5] 温谦. HTML+CSS网页设计与布局从入门到精通[M].人民邮电出版社, 2008年08月.