

Javaweb结课作业

**学生信息管理系统**

**学生姓名： 徐奥**

**所在院系： 人工智能与计算机**

**所属专业： 软件工程**

**所在班级： 软件11808**

**个人学号： 1804230809**

**评阅老师： 黄俊杰**

**2021年 7月 1日**

摘 要

进行学生信息管理是学校的重要工作之一。传统的学生信息管理由于是手工操作，工作量大且容易出错。随着计算机和网络技术的迅速发展，越来越多的院校都拥有了自己的学生信息管理系统，而采用B/S架构的学生信息管理系统进行学生信息的管理则是其趋势。

Java web技术的基础就是核心Java平台，它有许多优点，例如：“编写一次、随处运行”的特性、方便存取数据库的JDBC API技术以及能够在Internet应用中保护数据的安全模式等等。J2EE体系结构提供中间层集成框架用来满足高可用性、高可靠性以及可扩展性的应用需求，为搭建具有可伸缩性、灵活性、易维护性的网络应用系统提供了良好的机制。

本文基于Java web平台技术、采用Browser/Server结构,设计并实现了该系统的简化版学生信息管理系统。

论文首先介绍了系统开发涉及的相关技术及开发平台，主要围绕学生信息管理系统实现的相关技术和系统的体系结构，介绍了学生信息管理系统的总体分析与设计，包括需求分析、功能模块分析和数据库设计。最后，利用Jsp+Servlet+JavaBean模式对简单学生信息管理子系统进行了实现，并作了详细描述，主要采用图片与代码结合的方式详细地叙述了系统功能的实现。

关键词：学生信息管理系统，Java web，Servlet，JSP

**目录**

[1绪论 6](#_Toc470900803)

[1.1 课题背景及现状 6](#_Toc470900804)

[1.2 开发环境及工具介绍 6](#_Toc470900805)

[1.2.1操作系统及配置 6](#_Toc470900806)

[1.2.2编译环境 6](#_Toc470900807)

[1.2.2.1）Eclipse Neon Release (4.6.0) 6](#_Toc470900808)

[1.2.2.2）JetBrains WebStorm 2016.2.4 6](#_Toc470900809)

[1.2.2.3）Apache Tomcat v8.0 7](#_Toc470900810)

[1.2.2.4）MySQL Workbench 6.3 CE 7](#_Toc470900811)

[1.3 编程语言 7](#_Toc470900812)

[2 需求分析 7](#_Toc470900813)

[2.1 面向人群 7](#_Toc470900816)

[2.2 总体功能描述 7](#_Toc470900817)

[2.3 功能描述 8](#_Toc470900818)

[2.3.1用户登录系统实现功能： 8](#_Toc470900819)

[2.3.2管理员登录系统实现功能 8](#_Toc470900820)

[3总体设计 8](#_Toc470900821)

[3.1 运行环境 8](#_Toc470900825)

[3.2 基本处理流程 8](#_Toc470900826)

[3.2.1用户登录流程图 8](#_Toc470900827)

[3.2.2子模块操作处理流程图 9](#_Toc470900828)

[3.3 模块结构 10](#_Toc470900829)

[3.3.1系统结构图 10](#_Toc470900830)

[3.3.2功能模块简介 10](#_Toc470900831)

[4 详细设计 11](#_Toc470900832)

[4.1 子管理模块设计 11](#_Toc470900838)

[4.1.1程序流程图 11](#_Toc470900839)

[4.1.2 算法描述 11](#_Toc470900840)

[4.2 关键类说明 12](#_Toc470900841)

[5 数据库设计 12](#_Toc470900842)

[5.1 概念结构设计 12](#_Toc470900848)

[5.2 逻辑结构设计 13](#_Toc470900849)

[5.3 物理结构设计 13](#_Toc470900852)

[6 编码与测试 14](#_Toc470900853)

[6.1 操作界面截图 14](#_Toc470900854)

[6.2 调试过程 17](#_Toc470900862)

[6.3 登录注册模块 17](#_Toc470900863)

[6.4 增删改查模块 18](#_Toc470900864)

[7 总结 23](#_Toc470900865)

[8 附录 23](#_Toc470900866)

[参考文献 29](#_Toc470900867)

1绪论

* 1. 课题背景及现状

随着信息技术在学生信息管理上的广泛应用，学生信息管理系统的实施在技术上已逐步成熟。学生信息的管理对于学校管理者来说至关重要，是学校不可缺少的数据资源。特别是近几年，学生数量的增多使学校的管理面临更大压力和挑战。目前大多数学校仍然采用传统的人工管理方式，对学生信息的管理还停留在纸介质的基础上，这既不易于规范化，管理效率也不高，浪费了大量人力物力财力。

作为计算机应用的一部分，使用计算机进行学生信息的管理，有着手工管理无法比拟的优点。本系统针对学校及学生的信息特点，可以对学生信息进行高效率、规范化的管理，大大减轻了人员的工作负担，并避免了人工管理时容易出现的错误。

* 1. 开发环境及工具介绍

**1.2.1操作系统及配置**

Windows 10 专业版 x64位，Pentium(R) Dual-Core CPU， 2GB

**1.2.2编译环境**

**1.2.2.1）Eclipse Neon Release (4.6.0)**

Eclipse是著名的跨平台开源集成开发环境（IDE）。最初主要用来Java语言开发，目前亦有人通过插件使其作为C++、Python、PHP等其他语言的开发工具。Eclipse的本身只是一个框架平台，但是众多插件的支持，使得Eclipse拥有较佳的灵活性，所以许多软件开发商以Eclipse为框架开发自己的IDE。

**1.2.2.2）JetBrains WebStorm 2016.2.4**

WebStorm 是jetbrains公司旗下一款JavaScript 开发工具。被广大中国JS开发者誉为“Web前端开发神器”、“最强大的HTML5编辑器”、“最智能的JavaScript IDE”等。与IntelliJ IDEA同源，继承了IntelliJ IDEA强大的JS部分的功能。

**1.2.2.3）Apache Tomcat v8.0**

Tomcat是Apache 软件基金会（Apache Software Foundation）的Jakarta 项目中的一个核心项目，由Apache、Sun 和其他一些公司及个人共同开发而成。由于有了Sun 的参与和支持，最新的Servlet 和JSP 规范总是能在Tomcat 中得到体现，Tomcat 5支持最新的Servlet 2.4 和JSP 2.0 规范。因为Tomcat 技术先进、性能稳定，而且免费，因而深受Java 爱好者的喜爱并得到了部分软件开发商的认可，成为目前比较流行的Web 应用服务器。

Tomcat 服务器是一个免费的开放源代码的Web 应用服务器，属于轻量级应用服务器，在中小型系统和并发访问用户不是很多的场合下被普遍使用，是开发和调试JSP 程序的首选。

**1.2.2.4）MySQL Workbench 6.3 CE**

MySQL是一个关系型数据库管理系统，由瑞典MySQL AB公司开发，目前属于 Oracle 旗下产品。MySQL 最流行的关系型数据库管理系统，在 WEB 应用方面MySQL是最好的 RDBMS应用软件之一。由于其社区版的性能卓越，搭配 PHP 和 Apache 可组成良好的开发环境。

* 1. 编程语言

Java，html，css，javascipt

2需求分析

2. 1. 面向人群

学校的学生信息管理人员以及老师和学生

* 1. 总体功能描述

本系统利用Java Web技术实现了学生信息管理系统，具有简单的学生信息管理功能。

实现了以下功能模块：

院系信息管理模块，学生信息管理模块，课程信息管理模块，成绩管理模块

并能根据登入用户的权限自动展示相关操作。

* 1. 功能描述

**2.3.1用户登录系统实现功能：**

1. 查看全部用户，用页面返回结果
2. 查看全部院系，用页面返回结果
3. 查看全部学生信息，用页面返回结果
4. 查看全部课程信息，课程平均分，课程不及格率，用页面返回结果
5. 查看全部成绩，用页面返回结果

**2.3.2管理员登录系统实现功能**

1）对用户进行增删改查，用页面返回处理以及查询结果

2）对院系进行增删改查，用页面返回处理以及查询结果

3）对学生进行增删改查，用页面返回处理以及查询结果

4）对课程进行增删改查，并可以查看平均分、不及格率和成绩排名，用页面返回处理以及查询结果

5）对成绩进行增删改查，用页面返回处理以及查询结果

3总体设计

3. 1. 运行环境

硬件环境：Windows 7 及以上，处理器：Intel Pentium及以上，内存：2G以上

浏览器：IE9 以上及Chrome，FireFox，Safari等现代浏览器

* 1. 基本处理流程

**3.2.1用户登录流程图**

登录流程图

**图3-2-1 用户登录流程图**

**3.2.2子模块操作处理流程图**

子模块操作流程图

**图3-2-2 子模块操作处理流程图**

* 1. 模块结构

3.3.1系统结构图

系统功能模块图

**图3-3-1 系统结构图**

3.3.2功能模块简介

用户管理模块：对用户的信息进行增删改查

院系管理模块：对院系的信息进行增删改查

班级管理模块：对班级的信息进行增删改查

学生管理模块：对学生的基本信息进行增删改查

课程管理模块：对课程进行增删改查，并能查看平均分，不及格率，成绩排名

4 详细设计

5. 1. 子管理模块设计

**4.1.1程序流程图**

用户操作流程图

**图4-1-1 程序流程图**

**4.1.2 算法描述**

1. 用户在前台操作，向后台发送带参数的GET请求
2. AdminDao.java捕捉到请求，根据类型分发到不同处理方法中
3. 处理方法中实例化XXXDao对象并调用DBUtils.java提供的方法连接和操作数据库
4. 用ajax向前台user.jsp或者admin.jsp发送操作后的数据
   1. 关键类说明

LoginServlet.java：负责处理用户登录操作

RegisterServlet.java：负责处理用户注册操作

AdminDao.java：一个Servlet文件，接收前台发送的操作指令

XXXDao.java：封装了关于该信息在数据库中增删改查的方法

DBUtils.java：封装了获取数据库连接和释放数据库连接的方法

Admin.jsp：管理员操作界面

User.jsp：用户操作界面

5 数据库设计

5. 1. 概念结构设计

E-R图

E-R图

**图5-1 E-R图**

本数据库有实体集：User（用户），Department（院系），Class（班级），Student（学生），Course（课程）

它们之间的联系有：1对1：Student和User

1对多：Department和Class，Class和Studen

多对多：Course和Student

* 1. 逻辑结构设计

**关系模式**（下划线为主键，#表示外键）

User(Username,Password,Level)

Department(Dno,Dname)

Class(Clno,Clname,#Dno)

Student(Sno,Sname,Ssex,Sage,#Clno)

Course(Cno,Cname,Cteacher,Ccredit)

SC(#Sno,#Cno,Grade)

* 1. 物理结构设计

**表5-3 表汇总**

|  |  |
| --- | --- |
| 表名 | 功能说明 |
| User | 账号 |
| Department | 院系 |
| Class | 班级 |
| Student | 学生 |
| Course | 课程 |
| SC | 成绩 |

**表5-4 User表（用户表）**

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型（精度范围） | 空/非空 | 约束条件 |
| Username | Char(12) | 非空 |  |
| Password | Char(12) | 空 |  |
| Level | Char(6) | 空 | 用户or管理员 |

**表5-5 Department表（院系表）**

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型（精度范围） | 空/非空 | 约束条件 |
| Dno | Char(12) | 非空 |  |
| Dname | Char(12) | 空 |  |

**表5-6 Class表（班级表）**

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型（精度范围） | 空/非空 | 约束条件 |
| Clno | Char(12) | 非空 |  |
| Clname | Char(12) | 空 |  |
| Dno | Char(12) | 空 |  |

**表5-7 student表（学生表）**

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型（精度范围） | 空/非空 | 约束条件 |
| Sno | Char(12) | 非空 |  |
| Sname | Char(8) | 空 |  |
| Ssex | Char(2) | 空 | ‘男’or‘女’ |
| Sage | Smallint(6) | 空 | >0 |
| Clno | Char(12) | 空 |  |

**表5-8 SC表（成绩表）**

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型（精度范围） | 空/非空 | 约束条件 |
| Sno | Char(12) | 非空 |  |
| Cno | Char(12) | 非空 |  |
| Grade | Smallint(6) | 空 | 0~100 |

6 编码与测试

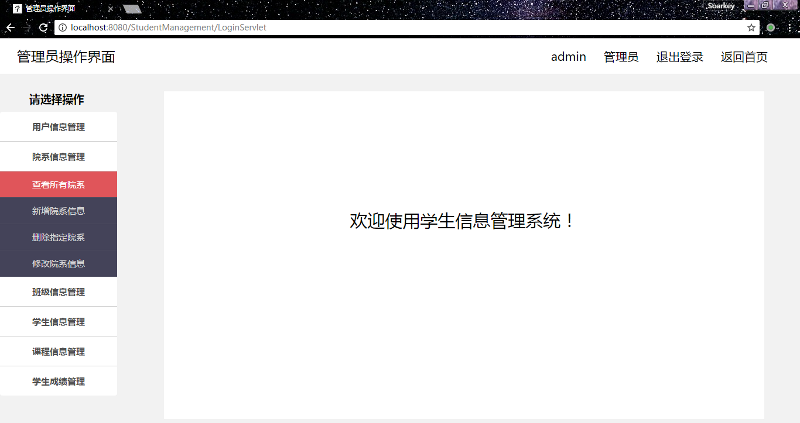
# 6.1 操作界面截图



**图6-1-1 登录界面**



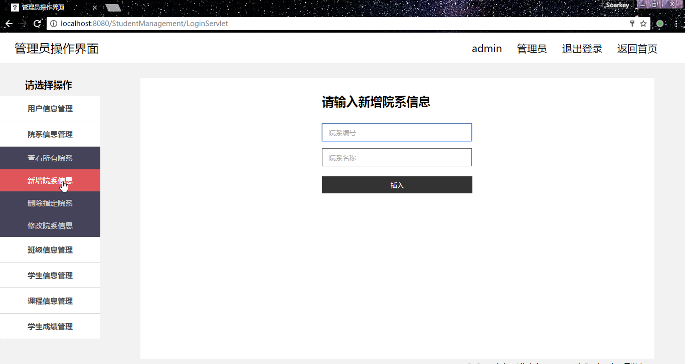
**图6-1-2 登录失败界面**



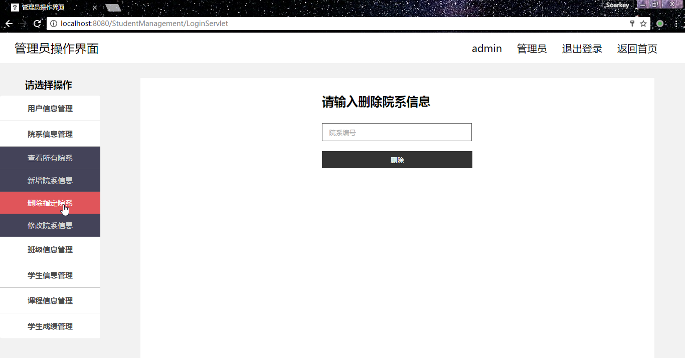
**图6-1-3 管理员操作界面**



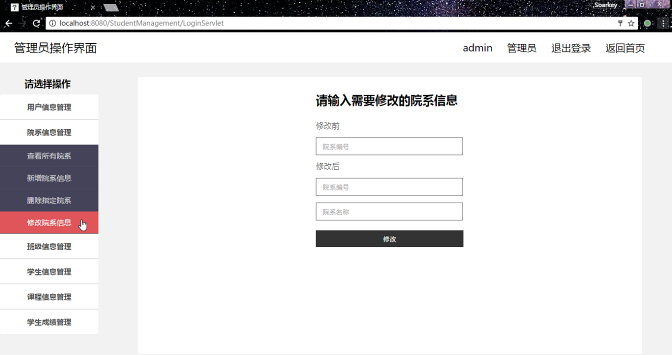
**图6-1-4 用户操作界面**



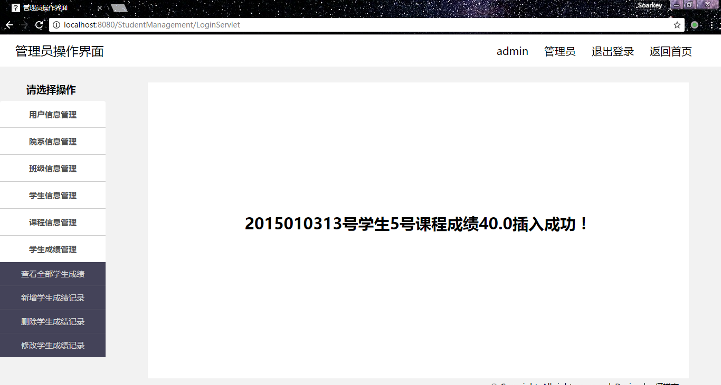
**图6-1-5 新增信息界面**



**图6-1-6 删除信息界面**



**图6-1-7 修改信息界面**



**图6-1-8 操作提示界面**



# 调试过程

我在调试这些功能时选择了按操作类型逐项调试，进行增删改查的调试，过程持续地比较久。在调试中出现了点击修改信息完成后，数据库中未进行更新的错误，后来经分析，是由于发送的请求里缺少了一些属性，导致操作失败。有的时候会出现用户注册信息为空也能注册成功并登录，后来给前台输入信息时加了js输入限制避免了此类错误。

# 登录注册模块

|  |
| --- |
| 1. public class LoginServlet extends HttpServlet { 2. private static final long serialVersionUID = 1L; 3. protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException { 4. request.setCharacterEncoding("utf-8"); 5. String username = request.getParameter("username"); 6. String password = request.getParameter("password"); 7. String level = null; 8. //实例化UserDao对象 9. UserDao userDao = new UserDao(); 10. User user = userDao.login(username, password); 11. //判断是否登录成功 12. if(user != null){//成功 13. level = user.getLevel(); 14. if(level.equals("用户")){ 15. request.getSession().setAttribute("user", user);//将用户对象放到session中 16. //转发到user.jsp中 17. request.getRequestDispatcher("user.jsp").forward(request, response); 18. } 19. else{ 20. request.getSession().setAttribute("admin", user);//将管理员对象放到session中 21. //转发到admin.jsp中 22. request.getRequestDispatcher("admin.jsp").forward(request, response); 23. } 24. }else {//失败 25. request.setAttribute("info"," 错误:用户名或密码错误！"); 26. request.getRequestDispatcher("message.jsp").forward(request, response); 27. } 28. } 29. } |

# 增删改查模块

|  |
| --- |
| 1. public class CourseDao { 2. // 获取所有课程的信息，用ArrayList返回 3. public ArrayList<Course> query\_all\_course() { 4. Connection conn = DBUtils.getConnection(); 5. String sql = "select \* from course order by cno;"; 6. ArrayList<Course> results = new ArrayList<Course>(); 7. try { 8. PreparedStatement ps = (PreparedStatement) conn.prepareStatement(sql); 9. ResultSet rs = ps.executeQuery(); 10. while (rs.next()) { 11. Course temp = new Course(); 12. temp.setCno(rs.getString("Cno")); 13. temp.setCname(rs.getString("Cname")); 14. temp.setCteacher(rs.getString("Cteacher")); 15. temp.setCcredit(rs.getInt("Ccredit")); 16. results.add(temp); 17. } 18. // 关闭资源 19. rs.close(); 20. ps.close(); 21. } catch (SQLException e) { 22. e.printStackTrace(); 23. } finally { 24. DBUtils.closeConnection(conn); 25. } 26. return results; 27. } 28. // 插入课程信息，返回一个int值表示状态,1：成功，0失败 29. public int insert\_course(String Cno, String Cname, String Cteacher, double Ccredit) { 30. Connection conn = DBUtils.getConnection(); 31. String sql = "insert into course values(?,?,?,?);"; 32. int flag = 0; 33. try { 34. PreparedStatement ps = (PreparedStatement) conn.prepareStatement(sql); 35. ps.setString(1, Cno); 36. ps.setString(2, Cname); 37. ps.setString(3, Cteacher); 38. ps.setDouble(4, Ccredit); 39. flag = ps.executeUpdate(); 40. ps.close(); 41. } catch (SQLException e) { 42. e.printStackTrace(); 43. } finally { 44. DBUtils.closeConnection(conn); 45. } 46. return flag; 47. } 48. // 删除课程信息，返回一个int值表示状态,1：成功，0失败 49. public int delete\_course(String Cno) { 50. Connection conn = DBUtils.getConnection(); 51. String sql = "delete from course where Cno = ?;"; 52. int flag = 0; 53. try { 54. PreparedStatement ps = (PreparedStatement) conn.prepareStatement(sql); 55. ps.setString(1, Cno); 56. flag = ps.executeUpdate(); 57. ps.close(); 58. } catch (SQLException e) { 59. e.printStackTrace(); 60. } finally { 61. DBUtils.closeConnection(conn); 62. } 63. return flag; 64. } 65. //修改课程信息，返回一个int值表示状态,1：成功，0失败 66. public int alter\_course(String cno,String after\_cno,String after\_cname,String after\_cteacher,double after\_ccredit) { 67. Connection conn = DBUtils.getConnection(); 68. String sql = "update course set cno = ?,cname = ?,cteacher = ?,ccredit = ? where cno = ?;"; 69. int flag = 0; 70. try { 71. PreparedStatement ps = (PreparedStatement) conn.prepareStatement(sql); 72. ps.setString(1, after\_cno); 73. ps.setString(2, after\_cname); 74. ps.setString(3, after\_cteacher); 75. ps.setDouble(4, after\_ccredit); 76. ps.setString(5, cno); 77. flag = ps.executeUpdate(); 78. ps.close(); 79. } catch (SQLException e) { 80. e.printStackTrace(); 81. }finally { 82. DBUtils.closeConnection(conn); 83. } 84. return flag; 85. } 86. // 查询课程平均分信息，返回一个ArrayLst集合 87. public ArrayList<Course\_avg> course\_avg() { 88. Connection conn = DBUtils.getConnection(); 89. String sql = "select sc.cno cno,cname,avg(grade) avg from course,sc where course.cno = sc.cno group by cno order by cno;"; 90. ResultSet result = null; 91. ArrayList<Course\_avg> course\_avg = new ArrayList<Course\_avg>(); 92. try { 93. PreparedStatement ps = (PreparedStatement) conn.prepareStatement(sql); 94. result = ps.executeQuery(); 95. while(result.next()){ 96. Course\_avg temp = new Course\_avg(); 97. temp.setCno(result.getString("Cno")); 98. temp.setCname(result.getString("Cname")); 99. temp.setAvg(result.getDouble("avg")); 100. course\_avg.add(temp); 101. } 102. ps.close(); 103. result.close(); 104. } catch (SQLException e) { 105. e.printStackTrace(); 106. } finally { 107. DBUtils.closeConnection(conn); 108. } 109. return course\_avg; 110. } 111. //查询课程不及格率，返回一个ArrayList集合 112. public ArrayList<Course\_fail\_rate> fail\_rate(){ 113. Connection conn = DBUtils.getConnection(); 114. String sql = "select cno,(select cname from course where cno = x.cno) cname,cast(100.0\*(select count(sno) from sc where grade < 60 and cno = x.cno)/(select count(sno) from sc where cno = x.cno) as decimal(18,2)) rate from sc x group by cno order by cno;"; 115. ArrayList<Course\_fail\_rate> fail\_rate = new ArrayList<Course\_fail\_rate>(); 116. try { 117. PreparedStatement ps = (PreparedStatement) conn.prepareStatement(sql); 118. ResultSet rs = ps.executeQuery(); 119. while(rs.next()){ 120. Course\_fail\_rate temp = new Course\_fail\_rate(); 121. temp.setCno(rs.getString("cno")); 122. temp.setCname(rs.getString("cname")); 123. temp.setFail\_rate(rs.getDouble("rate")); 124. fail\_rate.add(temp); 125. } 126. rs.close(); 127. ps.close(); 128. } catch (SQLException e) { 129. e.printStackTrace(); 130. } finally { 131. DBUtils.closeConnection(conn); 132. } 133. return fail\_rate; 134. } 135. //查询课程排名情况,返回一个ArrayList集合 136. public ArrayList<Course\_ranking> course\_ranking(String cno){ 137. Connection conn = DBUtils.getConnection(); 138. String sql = "select student.Sno Sno,Dname,Clname,Sname,Ssex,Sage,Grade from department,class,student,sc where student.sno = sc.sno and class.Clno = student.Clno and department.Dno = class.Dno and cno = '"+cno+"' order by grade desc;"; 139. ArrayList<Course\_ranking> course\_ranking = new ArrayList<Course\_ranking>(); 140. try { 141. PreparedStatement ps = (PreparedStatement) conn.prepareStatement(sql); 142. ResultSet rs = ps.executeQuery(); 143. while(rs.next()){ 144. Course\_ranking temp = new Course\_ranking(); 145. temp.setSno(rs.getString("Sno")); 146. temp.setDname(rs.getString("Dname")); 147. temp.setClname(rs.getString("Clname")); 148. temp.setSname(rs.getString("Sname")); 149. temp.setSsex(rs.getString("Ssex")); 150. temp.setSage(rs.getInt("Sage")); 151. temp.setGrade(rs.getDouble("Grade")); 152. course\_ranking.add(temp); 153. } 154. rs.close(); 155. ps.close(); 156. } catch (SQLException e) { 157. e.printStackTrace(); 158. } finally { 159. DBUtils.closeConnection(conn); 160. } 161. return course\_ranking; 162. } 163. } |

7 总结

本次项目遇到了许多问题，由于我对Servlet和Jsp以及前后台交互不熟悉导致前期进度相对缓慢，并且在代码上出现了很多重复的地方，导致了代码比较混乱，功能重复度太高，这个地方我还有很大的改进空间。在编程中，因为我采用了Ajax进行操作界面的异步刷新，而我不太会编写javascript代码，所有在这方面实现起来比较吃力，最后通过从图书馆借阅资料，百度解决方案，观看Java Web教学视频还是比较顺利地解决这个问题。其次，在本系统的界面设计上，我也琢磨了比较长的时间，经过不断地试错修正，利用所学的Html 5以及CSS 3设计出了一个较为美观的界面。

虽然在这次课程设计中，我遇到了许许多多的问题，但在这几天的时间里，我还是实现了学生信息管理系统的大致功能。虽然过程比较辛苦，但通过这次的学习与实践，我对Java Web编写动态页面更加熟悉了，也收获了很多解决此类问题的经验，相信面对以后的课程设计或者实际开发项目，我能更好地去处理解决它。

8 附录

1. **DBUtils.java 数据库连接类**

package utils;

import java.sql.DriverManager;

import java.sql.SQLException;

import com.mysql.jdbc.Connection;

public class DBUtils {

/\*\*

\* 获取数据库连接

\* @return Connection对象

\*/

public static Connection getConnection(){

String dbUserName = "sa";

String dbUserPasswd = "hellojava";

String dbURL = "jdbc:mysql://localhost:3306/studentinfomanagement?"

+ "user="+dbUserName+"&password="+dbUserPasswd+"&useUnicode=true&characterEncoding=UTF8";

Connection conn = null;

try {

Class.forName("com.mysql.jdbc.Driver");

conn = (Connection) DriverManager.getConnection(dbURL,dbUserName,dbUserPasswd);

} catch (ClassNotFoundException | SQLException e) {

e.printStackTrace();

}

return conn;

}

/\*\*

\* 关闭数据库连接

\* @param conn Connection对象

\*/

public static void closeConnection(Connection conn) {

//判断conn是否为空

if(conn != null){

try {

conn.close();//关闭数据库连接

} catch (SQLException e) {

e.printStackTrace();

}

}

}

}

1. **Department.java 院系操作类**

package dao;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import com.mysql.jdbc.Connection;

import com.mysql.jdbc.PreparedStatement;

import model.Department;

import utils.DBUtils;

public class DepartmentDao {

//查询所有的系信息，查询返回一个含值的ArrayList,当为空值的说明表中无数据元组

public ArrayList<Department> query\_all\_department() {

Connection conn = DBUtils.getConnection();

String sql = "select \* from department order by dno;";

ArrayList<Department> results = new ArrayList<Department>();

try {

PreparedStatement ps = (PreparedStatement) conn.prepareStatement(sql);

ResultSet rs = ps.executeQuery();

while (rs.next()){

Department temp = new Department();

temp.setDno(rs.getString("Dno"));

temp.setDname(rs.getString("Dname"));

results.add(temp);

}

rs.close();

ps.close();

} catch (SQLException e) {

e.printStackTrace();

} finally {

DBUtils.closeConnection(conn);

}

return results;

}

//插入院系信息，返回一个int值表示状态,1：成功，0失败

public int insert\_department(String dno,String dname){

Connection conn = DBUtils.getConnection();

String sql = "insert into department values(?,?);";

int flag = 0;

try {

PreparedStatement ps = (PreparedStatement) conn.prepareStatement(sql);

ps.setString(1, dno);

ps.setString(2, dname);

flag = ps.executeUpdate();

ps.close();

} catch (SQLException e) {

e.printStackTrace();

}finally {

DBUtils.closeConnection(conn);

}

return flag;

}

//删除院系信息，返回一个int值表示状态,1：成功，0失败

public int delete\_department(String dno) {

Connection conn = DBUtils.getConnection();

String sql = "delete from department where dno = ?;";

int flag = 0;

try {

PreparedStatement ps = (PreparedStatement) conn.prepareStatement(sql);

ps.setString(1, dno);

flag = ps.executeUpdate();

ps.close();

} catch (SQLException e) {

e.printStackTrace();

}finally {

DBUtils.closeConnection(conn);

}

return flag;

}

//修改院系信息，返回一个int值表示状态,1：成功，0失败

public int alter\_department(String dno,String after\_dno,String after\_dname) {

Connection conn = DBUtils.getConnection();

String sql = "update department set dno = ?,dname = ? where dno = ?;";

int flag = 0;

try {

PreparedStatement ps = (PreparedStatement) conn.prepareStatement(sql);

ps.setString(1, after\_dno);

ps.setString(2, after\_dname);

ps.setString(3, dno);

flag = ps.executeUpdate();

ps.close();

} catch (SQLException e) {

e.printStackTrace();

}finally {

DBUtils.closeConnection(conn);

}

return flag;

}

}

1. **Department.java 院系对象，封装了院系的属性**

package model;

import java.io.Serializable;

public class Department implements Serializable{

private static final long serialVersionUID = 1L;

private String Dno;//系编号

private String Dname;//系名

public String getDno() {

return Dno;

}

public void setDno(String dno) {

Dno = dno;

}

public String getDname() {

return Dname;

}

public void setDname(String dname) {

Dname = dname;

}

}

1. **数据库建表代码**

#建数据库

create database StudentInfoManagement;

use StudentInfoManagement;

#院系表

create table Department(

Dno char(12),

Dname char(12),

constraint primary key PK\_Department (Dno)

);

#班级表

create table Class(

Clno char(12),

Clname char(12),

Dno char(12),

constraint primary key PK\_Class (Clno),

constraint foreign key FK\_Class\_Department (Dno) references Department(Dno)

);

#学生表

create table Student(

Sno char(12),

Sname char(8),

Ssex char(2) check(Ssex in ('男','女')),

Sage smallint check(Sage > 0),

Clno char(12),

constraint primary key PK\_Student (Sno),

constraint foreign key FK\_Student\_Class (Clno) references Class(Clno)

);

#课程表

create table Course(

Cno char(12),

Cname char(12),

Cteacher char(8),

Ccredit smallint check(Ccredit > 0),

constraint primary key PK\_Course (Cno)

);

#选课表

create table SC(

Sno char(12),

Cno char(12),

Grade smallint check(Grade < 100 and Grade > 0),

constraint foreign key FK\_SC\_Student (Sno) references Student(Sno),

constraint foreign key FK\_SC\_Course (Cno) references Course(Cno),

constraint primary key PK\_SC (Sno,Cno)

);

#用户表

create table User(

username char(12),

password char(12) not null,

level char(6) check(level in ('用户','管理员')),

constraint primary key PK\_User (username)

);

参考文献

[1]软件开发技术联盟. 《JAVA WEB开发实战》.清华大学出版社. 2013

[2]丁振凡 .《Java语言程序设计》. 清华大学出版社. 2014

[3]王珊，萨师煊. 《数据库系统概论》.高等教育出版社. 2014