**应用集成原理与工具集成实验报告**

**姓名：宋琦敏 学号：2014317200325 班级：计科1403**

1. **实验环境：**

Window8.1操作系统、myeclipse集成开发环境、wampserver数据库操作集成开发环境

1. **实验目的：**

深入理解数据集成的原理

掌握中间件的数据集成方法

了解数据集成的技术与标准（jdbc、xml）

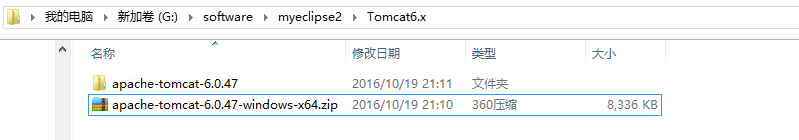
1. **实验内容：**

基于MVC模式的Java Web 开发应用程序，实现四个后台数据库的学生信息管理系统，并能在前台界面正确对数据库进行“增删改查”的功能，要求前端对数据库的操作采用基于中间件封装的技术，用户透明访问数据库，即用户看不见后台不同数据库的异构性。（本实验实现了两个后台数据库）

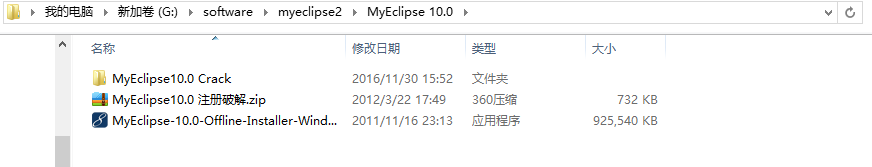
1. **实验步骤：**

**第一部分：连接两个数据库方法一**

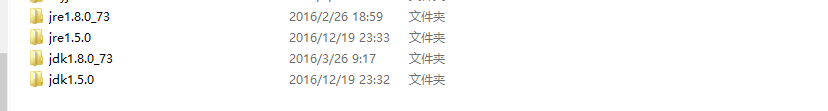
1. 环境安装与配置
2. 安装相应软件
3. 下载安装tomcat6.0。下载好压缩包后，直接解压至某一目录下目录不能包含中文名



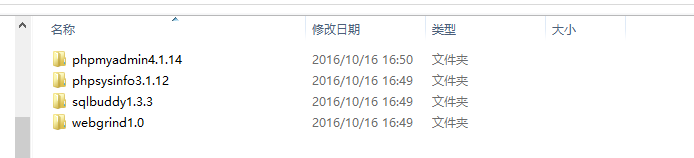
1. 下载用于java web 开发javaEE版的myeclipse继承开发环境。下载压缩包后，进行解压。在这里下载的是Myeclipse10.0，破解版，但试用期过后需要手动进行破解（网上有相关教程）



1. 下载jdk,双击exe进行安装，安装后配置环境变量。在这里因为本人之前已安装过jdk所以无需再进行下载安装。



1. 下载wampserver，用于数据库的操作，下载相应安装包并解压。下载该软件主要用于数据库的导入，数据库的可视化显示。因为wampserver是集成安装环境，即在windows下的apache、php、mysql的服务器软件，在这里只有要其中的mysql软件的功能。



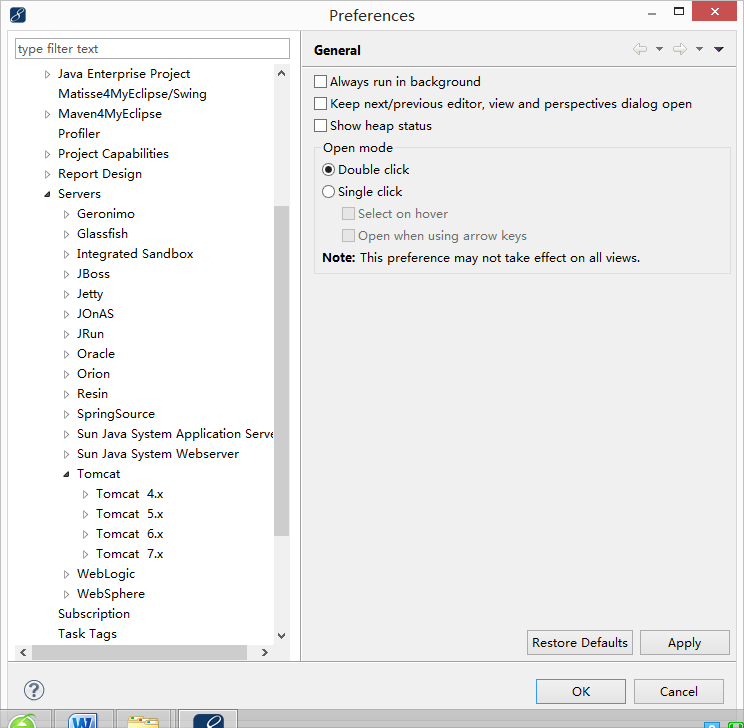
Mysql所用的版本为4.1.14

1. 下在jdbc驱动，无需解压

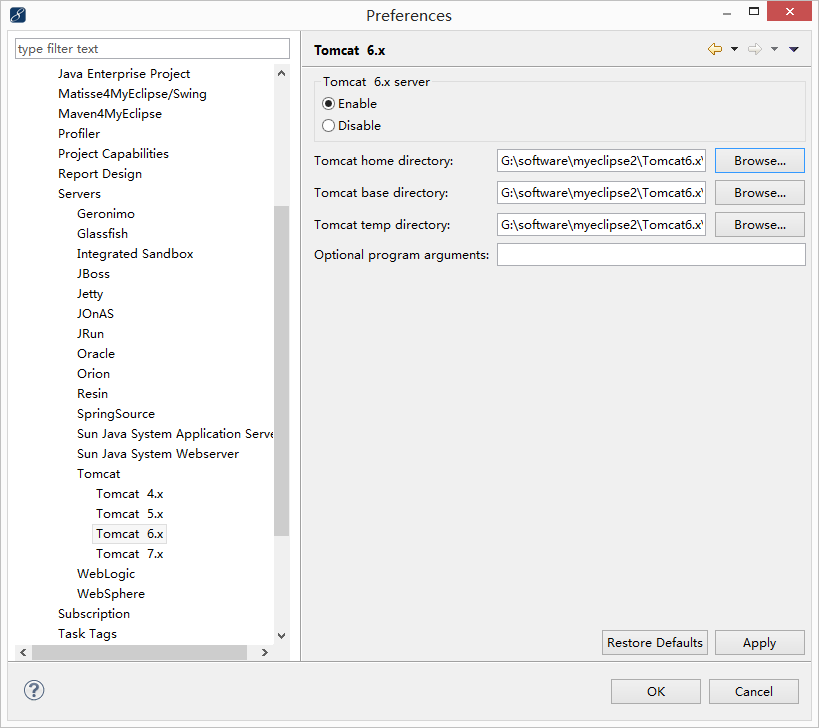


1. 环境配置
2. Myeclipse10.0配置tomcat6

进入windows->preference->myeclipse->servers->tomcat,选择tomcat6.X

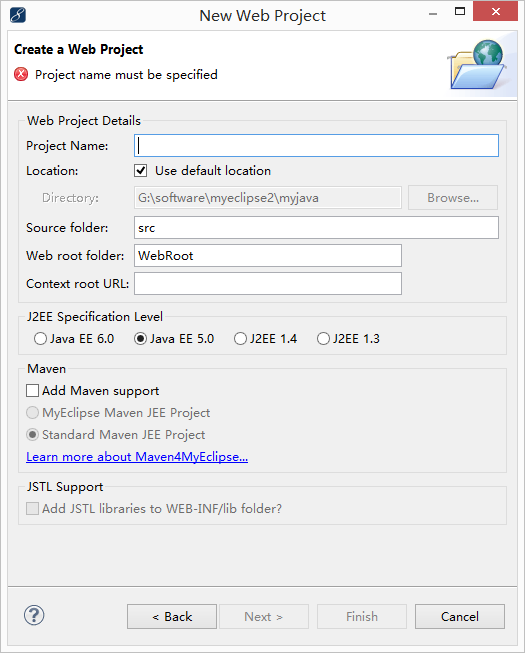


将上步下载压缩后的apache-tomcat-6.0.47导入进来，选择enable选项



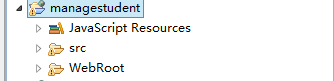
1. 创建java web工程

选择Myeclipse->files->new->web project



填写工程名，在这里工程名为“managestudent”

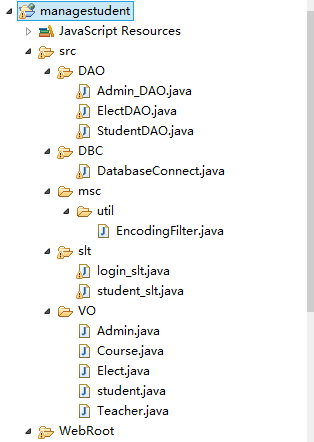
成功创建工程后会显示如下：

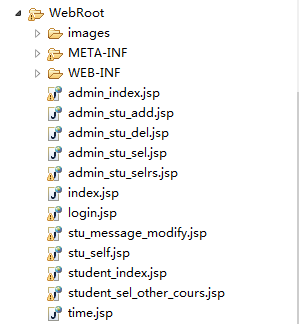


在WebRoot文件夹下可以创建jsp文件

在src文件价夹下可以继续创建包树，包含不同功能的文件

本工程所需要的所有文件如下：



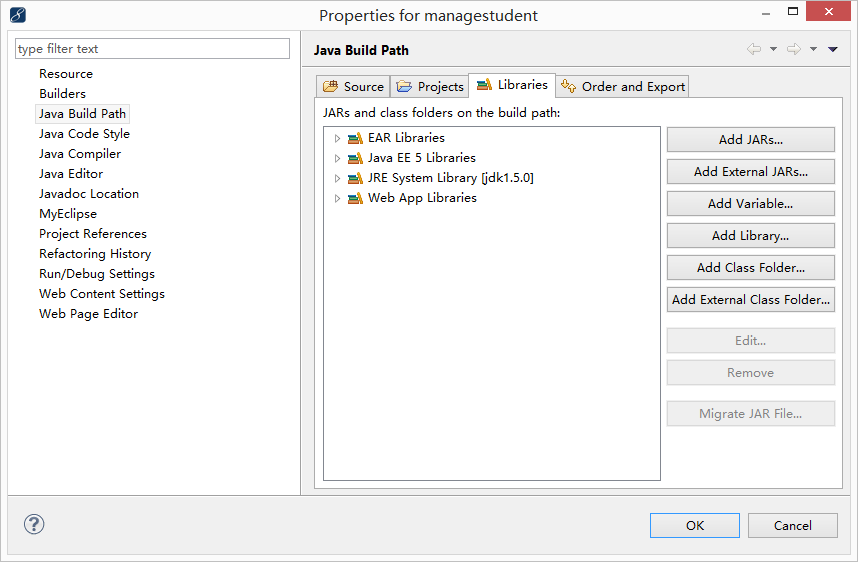


1. 因为本程序在连接两个数据库时，选择用

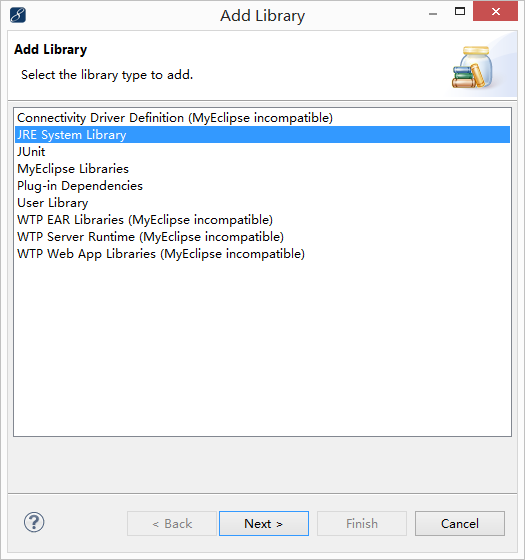


语句执行，但lookup支持的jre版本较低，所以更改工程的jre为较低版本。步骤如此：

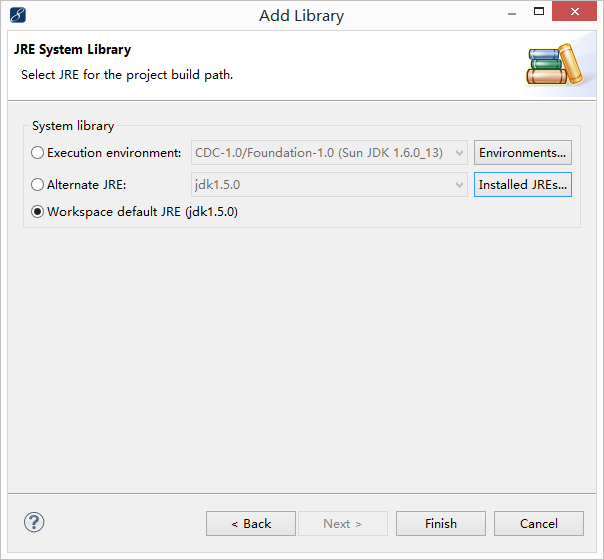
工程名(右键单击)->properties->java build path



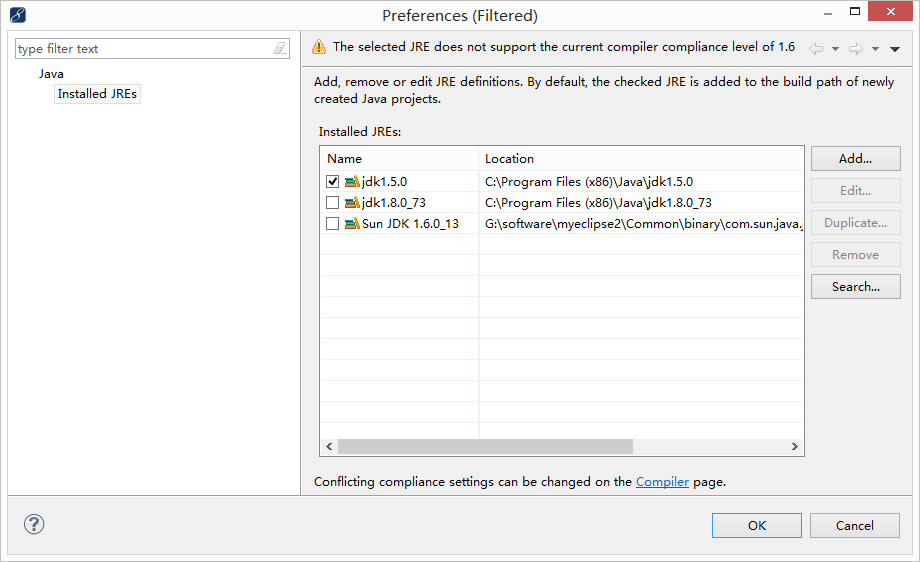
选择libraries->add library



选择下一步next



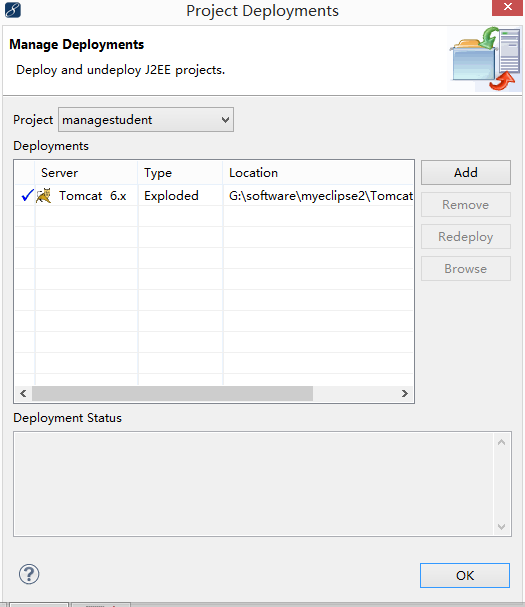
选择installed JREs



选择jdk1.5.0版本，点击ok即可

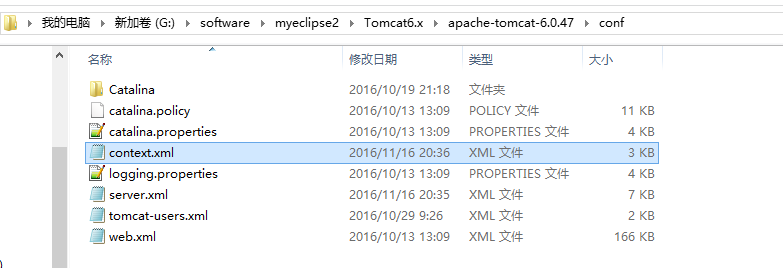
1. 发布工程到服务器

点击myeclipse界面中的图标



在project处选择要发布的工程名（这里是managestudent），在deployments处选择add添加发布的服务器（这里是之前配置过的tomcat6.x）

1. 程序设计（核心代码）
2. 两个数据库的连接—采用xml文件配置连接两个数据库
3. 配置tomcat安装目录下conf文件夹下的context.xml文件



在<Context></Context>中添加如下两段代码，分别代表两个不同的数据库

url="jdbc:mysql://localhost:3306/student?useUnicode=true&amp;characterEncoding=UTF-8"

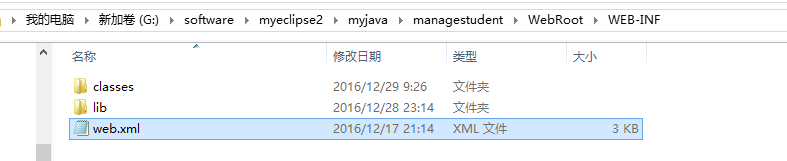
url="jdbc:mysql://localhost:3306/jike1?useUnicode=true&amp;characterEncoding=UTF-8"

|  |
| --- |
| <Resource name="jdbc/mysqla"  auth="Container"  type="javax.sql.DataSource"  driverClassName="com.mysql.jdbc.Driver" url="jdbc:mysql://localhost:3306/student?useUnicode=true&amp;characterEncoding=UTF-8"  username="root"  password=""  maxActive="100"  maxIdle="2"  maxWait="10000" />  <ResourceParams name="jdbc/mysqla">  <parameter>  <name>removeAbandoned</name>  <value>true</value>  </parameter>  <parameter>  <name>removeAbandonedTimeout</name>  <value>60</value>  </parameter>  <parameter>  <name>logAbandoned</name>  <value>true</value>  </parameter>  </ResourceParams> |

|  |
| --- |
| <Resource name="jdbc/mysqlb"  auth="Container"  type="javax.sql.DataSource"  driverClassName="com.mysql.jdbc.Driver" url="jdbc:mysql://localhost:3306/jike1?useUnicode=true&amp;characterEncoding=UTF-8"  username="root"  password=""  maxActive="100"  maxIdle="2"  maxWait="10000" />  <ResourceParams name="jdbc/mysqlb">  <parameter>  <name>removeAbandoned</name>  <value>true</value>  </parameter>  <parameter>  <name>removeAbandonedTimeout</name>  <value>60</value>  </parameter>  <parameter>  <name>logAbandoned</name>  <value>true</value>  </parameter>  </ResourceParams> |

1. 配置工程下的文件

打开mamagestudent->Webroot->WEB-INF->web.xml文件



在<web-app version=…….></web-app>标签下，增加如下代码

|  |
| --- |
| <!-- 数据源 -->  <resource-ref>  <description>DB Connection</description>  <res-ref-name>jdbc/mysqla</res-ref-name>  <res-type>javax.sql.DataSource</res-type>  <res-auth>Container</res-auth>  </resource-ref>  <resource-ref>  <description>DB Connection</description>  <res-ref-name>jdbc/mysqlb</res-ref-name>  <res-type>javax.sql.DataSource</res-type>  <res-auth>Container</res-auth>  </resource-ref> |

1. 实现数据库连接的java程序

|  |
| --- |
| package DBC;  import java.io.IOException;  import java.io.PrintWriter;  import java.sql.Connection;  import java.sql.DriverManager;  import javax.servlet.ServletException;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import javax.servlet.http.HttpSession;  import javax.servlet.ServletRequest;  import java.sql.\*;  import javax.sql.\*;  import javax.naming.\*;  import javax.sql.DataSource;  public class DatabaseConnect extends HttpServlet  {  private static String username;  private static String password;  private static String kind;  private static final String driver="com.mysql.jdbc.Driver";  private static Connection conn=null;  static  {  try{  Class.forName(driver);  }catch(Exception ex)  {  ex.printStackTrace();  }  }  public static Connection getConnection(char x) throws Exception {  /\*if(conn==null)  {  conn=DriverManager.getConnection(url, username, password);  return conn;  }  return conn;\*/  //只有写成如下代码才可以实现多次查询  if(x=='a'){  DataSource ds = null;  InitialContext ctx=new InitialContext(); ds=(DataSource)ctx.lookup("java:comp/env/jdbc/mysqla");  try {  conn = ds.getConnection()；  } catch (Exception e) {  System.out.println("vv"+e.getMessage());  }  }  else if(x=='b')  {  DataSource ds = null;  InitialContext ctx=new InitialContext(); ds=(DataSource)ctx.lookup("java:comp/env/jdbc/mysqlb");  try {  conn = ds.getConnection();  } catch (Exception e) {  System.out.println("vv"+e.getMessage());  }  }  return conn ;  }  public void close() {  try {  this.conn.close() ;  }  catch (Exception e) {  }  }  } |

说明：

因为本工程实现两个数据库的连接，分别代表计科1班，和计科3班，x变量标识不同的班级同学，当在登录界面是计科1班同学登录时，即连接数据库mysqlb,当时计科3班同学登录时，即连接数据库mysqla

public static Connection getConnection(char x)用于连接数据库的getConnection(char x)函数中的参数，即需要判断连接哪个数据库。

在登录界面如果是管理员身份登录，两个数据库都需要连接，当对数据库操作时调用两次不同getConnection()函数

1. 登录界面设计
2. Servlet核心代码：用于判断登录用户的身份是“管理员”、“计科1班学生”、“计科3班学生”

|  |
| --- |
| **Login\_slt.java(servlet判断登录用户身份)** |
| **package** slt;  **import** java.io.IOException;  **import** java.io.PrintWriter;  **import** java.text.DateFormat;  **import** java.util.Date;  **import** javax.servlet.ServletException;  **import** javax.servlet.http.HttpServlet;  **import** javax.servlet.http.HttpServletRequest;  **import** javax.servlet.http.HttpServletResponse;  **import** javax.servlet.http.HttpSession;  **import** VO.\*;  **import** DAO.\*;  **import** java.util.ArrayList;  **public** **class** login\_slt **extends** HttpServlet {  **public** login\_slt() {  **super**();  }  **public** **void** destroy() {  **super**.destroy();  }  **public** **void** doPost(HttpServletRequest request, HttpServletResponse response)**throws** ServletException, IOException {  PrintWriter out = response.getWriter();  response.setContentType("text/html");  request.setCharacterEncoding("utf-8");  response.setCharacterEncoding("utf-8");    String username=request.getParameter("username");  String password=request.getParameter("password");  String kind=request.getParameter("kind");    **if**(kind.equals("admin")){  Admin\_DAO udao=**new** Admin\_DAO();  Admin adm=**null**;  ArrayList<Admin> list=udao.getAlladmin();  **if**(list != **null**)  {  **int** count=0;  **for**(**int** i=0;i<list.size();i++)  {  adm=list.get(i);  count++;  String name=adm.getadmname();  String pass=adm.getadmpassword(); **if**(username.trim().equals(name.trim())&&(password.trim().equals(pass.trim())))//trim（）删除空格  {  HttpSession session;  session=request.getSession(**true**); session.setAttribute("username",adm.getadmname());  session.setAttribute("password",adm.getadmpassword());  response.sendRedirect("admin\_index.jsp");  **break**;  }  **if**(count==list.size()){  response.setHeader("refresh","2;URL=login.jsp") ;  out.print("输入有误！请重新登录......</font>");  }  }  }  }  **else** **if**(kind.equals("student")){  StudentDAO udao=**new** StudentDAO();  student stu=**null**;  ArrayList<student> list=**null**;  System.*out*.println("@@@");  System.*out*.println(username.indexOf("003"));  System.*out*.println(username.indexOf("001"));  **if**(username.indexOf("003")>0)  {  System.*out*.println("@012");  list=udao.getAllstudent('a');  }  **else** **if**(username.indexOf("001")>0)  {  list=udao.getAllstudent('b');  }  System.*out*.println("%123");    **if**(list != **null**)  {  **int** count=0;  System.*out*.println("2");  **for**(**int** i=0;i<list.size();i++)  {  stu=list.get(i);  count++;  String name=stu.getId();  String pass=stu.getPassword();  System.*out*.println("4");  **if**(username.trim().equals(name.trim())&&(password.trim().equals(pass.trim())))//trim（）删除空格  {  ;  HttpSession session;  session=request.getSession(**true**);  session.setAttribute("username",stu.getId());  session.setAttribute("password",stu.getPassword());  response.sendRedirect("student\_index.jsp");  **break**;  }  **if**(count==list.size()){    response.setHeader("refresh","2;URL=login.jsp") ;  out.print("输入有误！请重新登录......</font>");  }  }      }  }  out.flush();  out.close();  }  } |

1. 管理员登录判断java程序（操作数据库）

|  |
| --- |
| **Admin\_DAO.java（判断管理员是否正确）查询数据库** |
| **package** DAO;  **import** java.sql.\*;  **import** java.util.ArrayList;  **import** DBC.DatabaseConnect;  **import** VO.Admin;  **import** VO.\*;  **public** **class** Admin\_DAO {  **public** ArrayList<Admin> getAlladmin()  {  Connection conn=**null**;  PreparedStatement stmt=**null**;  ResultSet rs=**null**;  ArrayList<Admin> list=**new** ArrayList<Admin>();//学生信息集合  **try**  {  conn=DatabaseConnect.*getConnection*('a');  String sql= "select admname,admpassword from admin;";//SQL语句  stmt=conn.prepareStatement(sql);  rs=stmt.executeQuery();  **while**(rs.next())  {  Admin adm=**new** Admin();  adm.setadmname(rs.getString("admname"));  adm.setadpassword(rs.getString("admpassword"));  list.add(adm);//把一个学生加入语句  }  **return** list;//返回集合  }  **catch**(Exception ex)  {  ex.printStackTrace();  **return** **null**;  }  **finally**  {  //释放数据集对象  **if**(rs!=**null**)  {  **try**{  rs.close();  rs=**null**;  }  **catch**(Exception ex)  {  ex.printStackTrace();  }  }  //释放语句对象  **if**(stmt!=**null**)  {  **try**{  stmt.close();  stmt=**null**;  }  **catch**(Exception ex)  {  ex.printStackTrace();  }  }  }  }  } |

1. 学生登录判断（计科1班、计科3班）java程序（操作数据库）

|  |
| --- |
| **Student\_DAO.java(部分代码：和登录相关)查询数据库** |
| **package** DAO;  **import** java.sql.\*;  **import** java.util.ArrayList;  **import** java.util.Vector;  **import** java.sql.Connection;  **import** java.sql.PreparedStatement;  **import** java.sql.ResultSet;  **import** VO.\*;  **import** DBC.DatabaseConnect;  **public** **class** StudentDAO {  **public** ArrayList<student> getAllstudent(**char** x)  {  Connection conn=**null**;  PreparedStatement stmt=**null**;  ResultSet rs=**null**;  ArrayList<student> list=**new** ArrayList<student>();//学生信息集合  **try**  {  **if**(x=='a'){  conn=DatabaseConnect.*getConnection*('a');  String sql= "select `stu-id`,`name`,`sex`,`nation`,`birth`,`clas`,`enrollment\_date`,`hobby`,`password` from `student`;";//SQL语句  System.*out*.println(sql);  stmt=conn.prepareStatement(sql);  rs=stmt.executeQuery();  **while**(rs.next())  {  student stu=**new** student();  stu.setId(rs.getString("stu-id"));  stu.setPassword(rs.getString("password"));  list.add(stu);//把一个学生加入语句  }  }  **else**{  conn=DatabaseConnect.*getConnection*('b');  String sql= "select `sid`,`password` from `student`;";//SQL语句  System.*out*.println(sql);  stmt=conn.prepareStatement(sql);  rs=stmt.executeQuery();  **while**(rs.next())  {  student stu=**new** student();  stu.setId(rs.getString("sid"));  stu.setPassword(rs.getString("password"));  list.add(stu);//把一个学生加入语句  }  }  **return** list;//返回集合  }  **catch**(Exception ex)  {  ex.printStackTrace();  **return** **null**;  }  **finally**  {  //释放数据集对象  **if**(rs!=**null**)  {  **try**{  rs.close();  rs=**null**;  }  **catch**(Exception ex)  {  ex.printStackTrace();  }  }  //释放语句对象  **if**(stmt!=**null**)  {  **try**{  stmt.close();  stmt=**null**;  }  **catch**(Exception ex)  {  ex.printStackTrace();  }  }  }  }  ……………….  } |

1. 界面显示

登录界面

用户名：管理员身份为昵称

学生身份为学号



管理员登录界面

（管理员：大米）



（欢迎你：XXXX）根据登录的不同身份动态显示

学生登录界面（计科1班）

欢迎你：显示1班学生登录的学号



学生登录界面（计科3班）

欢迎你：显示3班学生登录的学号



1. 管理员操作
2. 功能操作界面核心代码

|  |
| --- |
| **Admin\_index.jsp(部分代码)** |
| <form name=*"form1"* method=*"post"* action=*""* >  <table width=*"97%"*height=*"80%"* align=*"center"* cellpadding=*"0"* cellspacing=*"0"*background=*"images/14.jpg"*>  <tr>  <td>欢迎进入计科专业学生信息系统</td>  </tr>  <tr>  <td><table width=*"100%"* cellspacing=*"0"* cellpadding=*"0"*>  <tr>  <td width=*"37%"*>欢迎你:<%out.println(username); %></td>  <td width=*"63%"*><%@include file=*"time.jsp"* %>&nbsp;</td>  </tr>  </table></td>  </tr>  <tr>  <td><table width=*"100%"* cellspacing=*"0"* cellpadding=*"0"*>  <tr>  <td width=*"25%"*><a href=*"admin\_stu\_sel.jsp"* target=*"main"*>查询学生信息</td>  <td width=*"24%"*><a href=*"admin\_stu\_add.jsp"* target=*"main"*>增加学生信息</td>  <td width=*"23%"*><a href=*"admin\_stu\_del.jsp"* target=*"main"*>删除学生信息</td>  <td width=*"7%"*><a href=*"login.jsp"* target=*"main"*>退出</td>  </tr>  </table></td>  </tr>  <tr>  <td>&nbsp;</td>  </tr>  </table>  </form> |

1. 查询学生信息

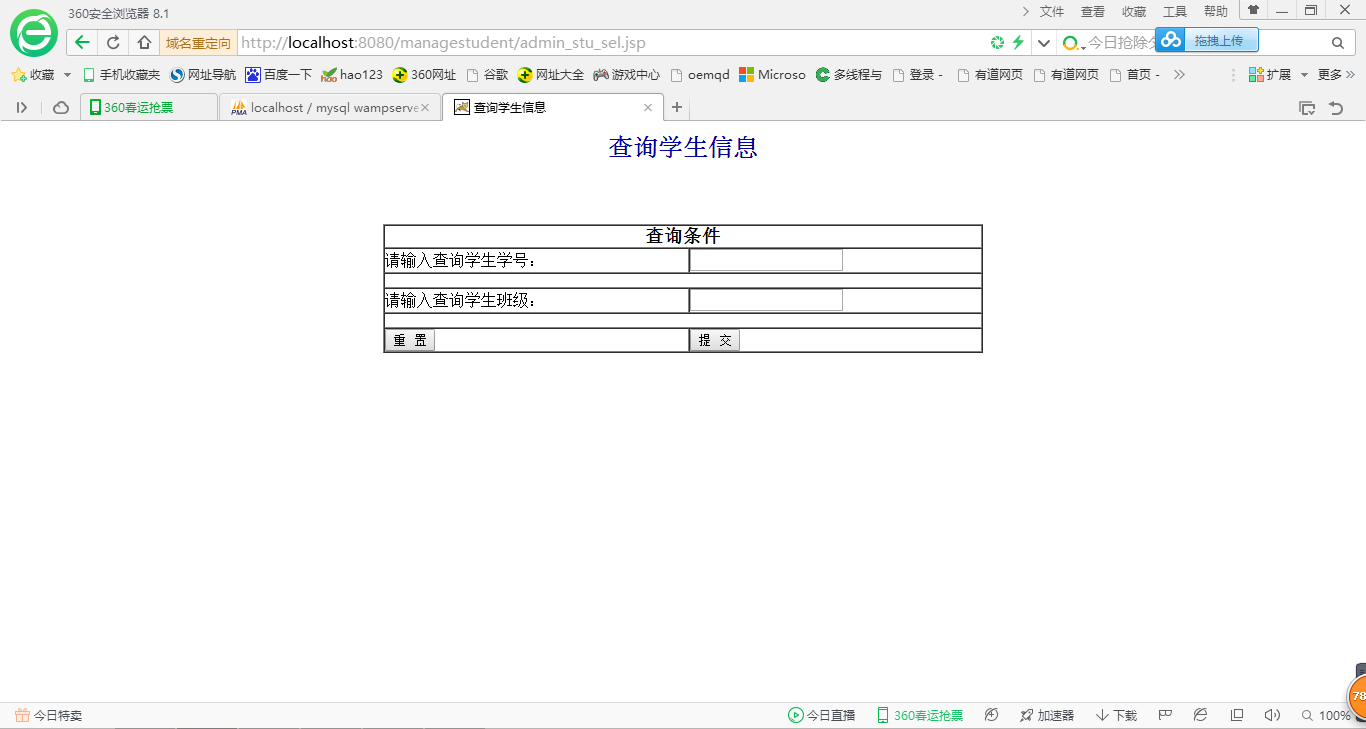
|  |
| --- |
| **Admin\_stu\_sel.jsp（部分代码）查询条件界面显示** |
| <form name=*"formselstu"* action=*"admin\_stu\_selrs.jsp?"* method=*"post"*><div align=*"center"*>  <font color=*"#0000a0"* face=*"华文宋体"*><strong><font size=*"5"*>查询学生信息<br><br><br></font></strong></font></div>  <table border=*"1"* width=*"600"* align=*"center"* cellspacing=*"0"* cellpadding=*"0"*>  <tr><th colspan=*"2"* height=*"15"*><font size=*"4"*>查询条件</font></th></tr>  <tr><td height=*"25"*>请输入查询学生学号：</td><td><input type=*"text"* name=*"id"*/></td></tr>  <tr><th colspan=*"2"* height=*"15"*></th></tr>  <tr><td height=*"25"*>请输入查询学生班级：</td><td><input type=*"text"* name=*"clas"*/></td></tr>  <tr><th colspan=*"2"* height=*"15"*></th></tr>  <tr><td height=*"20"*><input type=*"reset"* value=*"重 置"*/></td><td><input type=*"submit"* value=*"提 交"*/></td></tr>  </table>  </form> |

|  |
| --- |
| **Admin\_stu\_selrs.jsp（部分代码）查询结果界面显示** |
| <html>  <head>  <title>学生信息显示</title>  <base href=*"*<%=basePath%>*"*>  <jsp:useBean id=*"sdao"* scope=*"page"* class=*"DAO.StudentDAO"*></jsp:useBean>  <title></title>  </head>  <body bgcolor=*""*>  <%  request.setCharacterEncoding("utf-8");  String id=request.getParameter("id");  String clas=request.getParameter("clas");  Vector vector=**null**;  **if**(id.indexOf("003")>0||(clas.indexOf("计科1403")==0))  {  vector=sdao.Select(id,clas);  **int** i=vector.size();  student stu[]=**new** student[i];  %>  <div align=*"center"*><font size=*"5"* color=*"#000000"* face=*"华文宋体"*><b>学生信息如下：<br><br></b></font></div>  <table border=*"1"* cellspacing=*"0"* cellpadding=*"0"* align=*"center"*>  <tr align=*"center"*><td >学号</td><td>姓名</td><td >性别</td><td>民族</td><td>生日</td><td>班级</td><td>入学时间</td><td>爱好</td><td>选课信息</td></tr>  <%  **for**(**int** j=0;j<i;j++){  stu[j]=(student)vector.elementAt(j);  String rsid = stu[j].getId() ;  String rsname = stu[j].getName() ;  String rsex=stu[j].getSex();  String rnation=stu[j].getNation();  String rbirth=stu[j].getBirth();  String rsclass = stu[j].getClas() ;  String renrollment\_date=stu[j].getEnrollment\_date();  String rhobby=stu[j].getHobby();  %>  <tr align=*"center"*>  <td><%=rsid%></td>  <td><%=rsname%></td>  <td><%=rsex%></td>  <td><%=rnation%></td>  <td><%=rbirth%></td>  <td><%=rsclass%></td>  <td><%=renrollment\_date%></td>  <td><%=rhobby%></td>  <td><a href=*"student\_sel\_other\_cours.jsp?id=*<%=rsid %>*"*>查看</a></td>  </tr>  <%  }  }  **else**  {  ………………  }  **if**(i==0){  %>  <tr><th colspan=*"11"*>无该学生信息！</th></tr>  <%} }  %>  </table>  </body> |

|  |
| --- |
| **StudentDAO.java（部分代码）执行数据库查询** |
| **public** Vector Select(String id,String clas)**throws** Exception{  Vector all=**new** Vector();  System.*out*.println(clas);  System.*out*.println(id);  String sql1="select \* from `student` where ";  String sql=**null**;  **if**(id!="")  {  sql=sql1+"`stu-id` like '%"+id+"%'";  **if**(clas!="")  {  sql=sql+" and `clas` like '%"+clas+"%'";  }  }  **else**  {  **if**(clas!="")  {  sql=sql1+"`clas` like '%"+clas+"%'";  }  **else**  {  sql="select \* from `student`";  }  }  System.*out*.println(sql);  Statement stmt=**null**;  DatabaseConnect dbc=**null**;  dbc=**new** DatabaseConnect();  **try**{ stmt=dbc.*getConnection*('a').createStatement();  ResultSet rs=stmt.executeQuery(sql);  **while**(rs.next()) {  student stu=**new** student();  stu.setId(rs.getString("stu-id"));  stu.setName(rs.getString("name"));  stu.setNation(rs.getString("nation"));  stu.setSex(rs.getString("sex"));  stu.setBirth(rs.getString("birth"));  stu.setClas(rs.getString("clas")); stu.setEnrollment\_date(rs.getString("enrollment\_date"));  stu.setHobby(rs.getString("hobby"));    all.addElement(stu);  }  rs.close();  stmt.close();  } **catch** (Exception e) {  **throw** **new** Exception("查询中出现错误！！！");  } **finally** {  dbc.close();  }  **return** all;  }  **public** Vector SelectBySid(String id) **throws** Exception{  Vector all=**new** Vector();  **if**(id.indexOf("003")>0){  String sname=**null**;  String sql="select \* from `student` where `stu-id`='"+id+ "';";  Statement stmt=**null**;  DatabaseConnect dbc=**null**;  dbc=**new** DatabaseConnect();  **try**{  stmt=dbc.*getConnection*('a').createStatement();  ResultSet rs=stmt.executeQuery(sql);  **while**(rs.next()) {  student stu=**new** student();  stu.setId(rs.getString("stu-id"));  stu.setName(rs.getString("name"));  stu.setNation(rs.getString("nation"));  stu.setSex(rs.getString("sex"));  stu.setBirth(rs.getString("birth"));  stu.setClas(rs.getString("clas"));  stu.setEnrollment\_date(rs.getString("enrollment\_date"));  stu.setHobby(rs.getString("hobby")); stu.setPassword(rs.getString("password"));  all.addElement(stu);    }      rs.close();  stmt.close();  } **catch** (Exception e) {  **throw** **new** Exception("查询中出现错误！！！");  } **finally** {  dbc.close();  }  }  **if**(id.indexOf("001")>0){  String sname=**null**;  String sql="select \* from `student` where `sid`='"+id+ "';";  Statement stmt=**null**;  DatabaseConnect dbc=**null**;  dbc=**new** DatabaseConnect();  **try**{ stmt=dbc.*getConnection*('b').createStatement();  ResultSet rs=stmt.executeQuery(sql);  **while**(rs.next()) {  student stu=**new** student();  stu.setSid(rs.getString("sid"));  stu.setName(rs.getString("name"));  stu.setSex(rs.getString("sex"); stu.setPassword(rs.getString("password"));    stu.setQq(rs.getString("qq"));  stu.setTel(rs.getString("tel"));  stu.setProvince(rs.getString("province"));  all.addElement(stu);    }  rs.close();  stmt.close();  } **catch** (Exception e) {  **throw** **new** Exception("查询中出现错误！！！");  } **finally** {  dbc.close();  }  }  **return** all;  } |

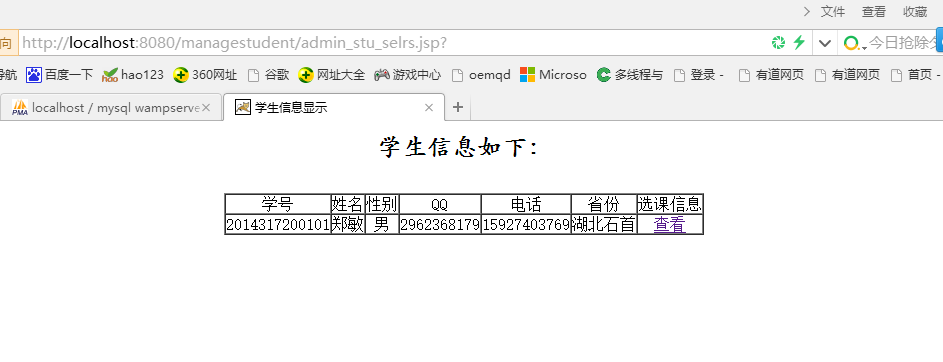
通过判断学生账号对不同数据库中的学生信息进行查询

界面展示

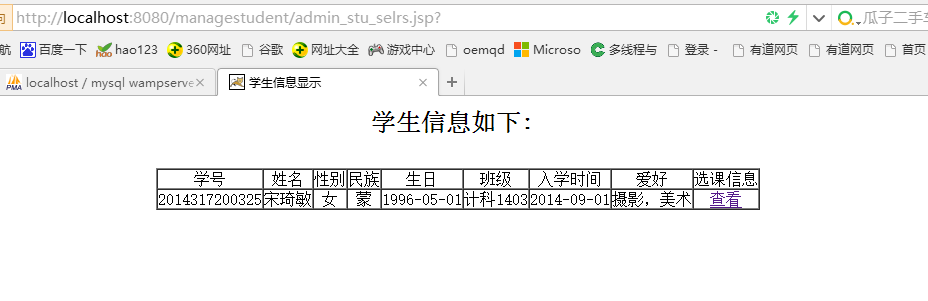


查询1班某一学生（输入学号、班级）

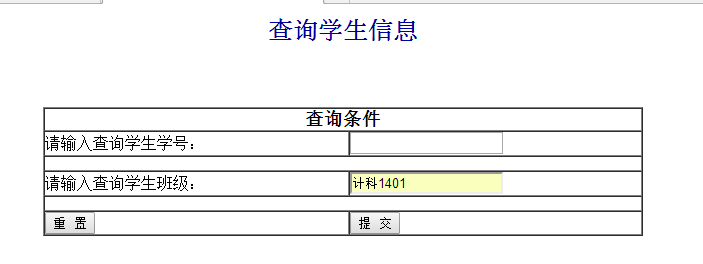




查询3班某一学生（输入学号、班级） 



查询1班所有同学（输入班级）





查询3班所有同学（输入班级）





1. 增加学生信息，删除学生信息，修改学生信息

学生信息操作的servlet程序

|  |
| --- |
| **Student\_slt.java（部分代码）接收jsp界面信息** |
| **public** **void** doPost(HttpServletRequest request, HttpServletResponse response)  **throws** ServletException, IOException {  response.setContentType("text/html");  request.setCharacterEncoding("utf-8");  response.setCharacterEncoding("utf-8");  String action = request.getParameter("action");  **if**("modify".equalsIgnoreCase(action)){  doUpdate(request,response);    }  **if** ("new".equalsIgnoreCase(action)) {  doAdd(request,response);    }  **if**("delete".equalsIgnoreCase(action)){  doDelete(request,response);    }  }  **public** **void** doAdd(HttpServletRequest request, HttpServletResponse response)  **throws** ServletException, IOException {  response.setContentType("text/html");  request.setCharacterEncoding("utf-8");  response.setCharacterEncoding("utf-8");  PrintWriter out = response.getWriter();  String sid=request.getParameter("sid");  String sname=request.getParameter("sname");  String ssex=request.getParameter("ssex");  String sclas=request.getParameter("sclas");  String spassword=request.getParameter("spassword");    String snation=request.getParameter("snation");  String sbirth=request.getParameter("sbirth");  String senrollment\_date=request.getParameter("senrollment\_date");  String shobby=request.getParameter("shobby");  String sqq=request.getParameter("sqq");  String stel=request.getParameter("stel");  String sprov=request.getParameter("sprov");  StudentDAO sdao=**new** StudentDAO();  **try** {  **if**(sclas.indexOf("1403")>0)  {    sdao.InsertStudent(sid, sname, ssex, snation, sbirth,sclas,senrollment\_date,shobby,spassword);  }  **else** **if**(sclas.indexOf("1401")>0)  {  sdao.InsertStudent1(sid, sname, spassword,ssex,sqq,stel,sprov);  }  out.println("<HTML>");  out.println(" <HEAD><TITLE>A Servlet</TITLE></HEAD>");  out.println(" <BODY bgcolor=>");  out.print(" <font size=6 color=black> 添加成功！</font> ");  out.println(" </BODY>");  out.println("</HTML>");  } **catch** (Exception e) {    out.println("<HTML>");  out.println(" <HEAD><TITLE>A Servlet</TITLE></HEAD>");  out.println(" <BODY bgcolor=>");  out.print(" <font size=6 color=black> 添加失败！</font> ");  out.println(" </BODY>");  out.println("</HTML>");    }  out.flush();  out.close();  }    **public** **void** doDelete(HttpServletRequest request, HttpServletResponse response)**throws** ServletException, IOException {  response.setContentType("text/html");  request.setCharacterEncoding("utf-8"); response.setCharacterEncoding("utf-8");  PrintWriter out = response.getWriter();  String sid=request.getParameter("sid");  StudentDAO sdao=**new** StudentDAO();  **try** {  **if**(sid.indexOf("003")>0)  {  sdao.DeletebyID(sid);  }  **else** **if**(sid.indexOf("001")>0)  {  sdao.DeletebyID1(sid);  }  out.println("<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.01 Transitional//EN\">");  out.println("<HTML>");  out.println(" <HEAD><TITLE>A Servlet</TITLE></HEAD>");  out.println(" <BODY bgcolor=>");  out.print(" <font size=6 color=black> 删除成功！</font> ");  out.println(" </BODY>");  out.println("</HTML>");    } **catch** (Exception e) {    out.println("<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.01 Transitional//EN\">");  out.println("<HTML>");  out.println(" <HEAD><TITLE>A Servlet</TITLE></HEAD>");  out.println(" <BODY bgcolor=>");  out.print(" <font size=6 color=black> 删除失败！</font> ");  out.println(" </BODY>");  out.println("</HTML>");  }  out.flush();  out.close();  } |

|  |
| --- |
| **StudentDAO.java（部分代码）数据库操作** |
| **public** **void** InsertStudent(String sid, String sname, String ssex, String snation, String sbirth,String sclas,String senrollment\_date,String shobby,String spassword) **throws** Exception{  String sql="INSERT INTO `student`(`stu-id`, `name`, `sex`, `nation`, `birth`, `clas`, `enrollment\_date`, `hobby`,`password`) VALUES ('"+sid+"','"+sname+"','"+ssex+"','"+snation+"','"+sbirth+"','"+sclas+"','"+senrollment\_date+"','"+shobby+"','"+spassword+"')";  Statement stmt=**null**;  DatabaseConnect dbc=**null**;  dbc=**new** DatabaseConnect();  **try**{  System.*out*.println(sql); stmt=dbc.*getConnection*('a').createStatement();  stmt.executeUpdate(sql);  stmt.close();  }**catch** (Exception e){  **throw** **new** Exception("插入中出现错误！！！");  }**finally** {  dbc.close();  }  }  **public** **void** DeletebyID(String sid) **throws** Exception{  String sql="DELETE FROM student WHERE `stu-id`='"+sid+"';";  Statement stmt=**null**;  DatabaseConnect dbc=**null**;  dbc=**new** DatabaseConnect();  **try**{  System.*out*.println(sql);  stmt=dbc.*getConnection*('a').createStatement();  stmt.executeUpdate(sql);  stmt.close();  }**catch** (Exception e){  **throw** **new** Exception("删除中出现错误！！！");  }**finally** {  dbc.close();  }  } |

* 增加学生界面显示

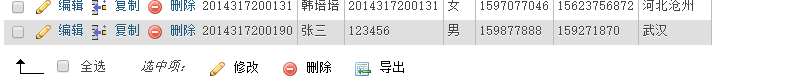
选择学生班级“计科1401”



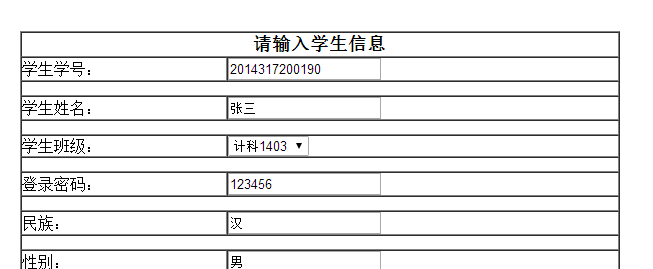
点击提交显示



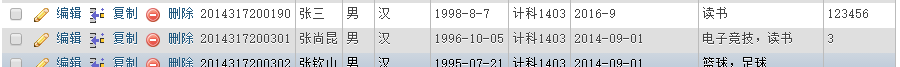
查看一班学生数据库，该学生成功加入



选择学生班级“计科1403”

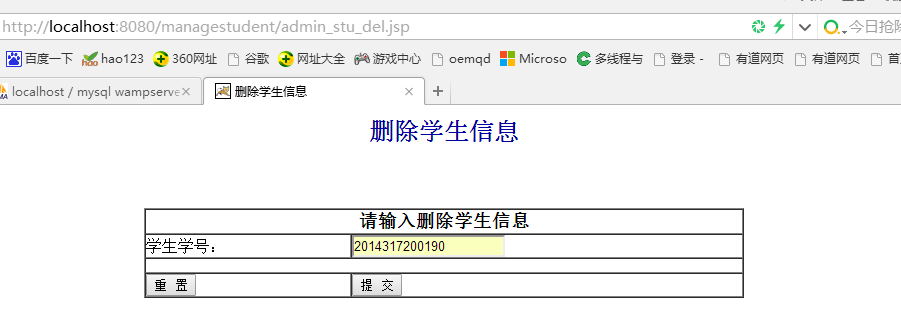


执行提交，成功添加。查看计科3班数据库

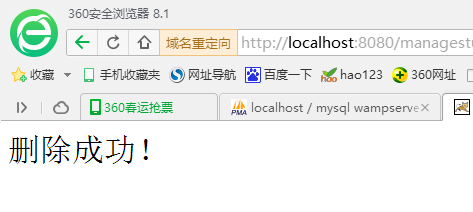


* 删除学生信息界面显示

输入刚添加到1班的数据库的学生学号，



执行提交，成功删除



查看数据库可以发现成功删除



删除3班数据库同理

1. 学生操作
2. 功能操作界面核心代码

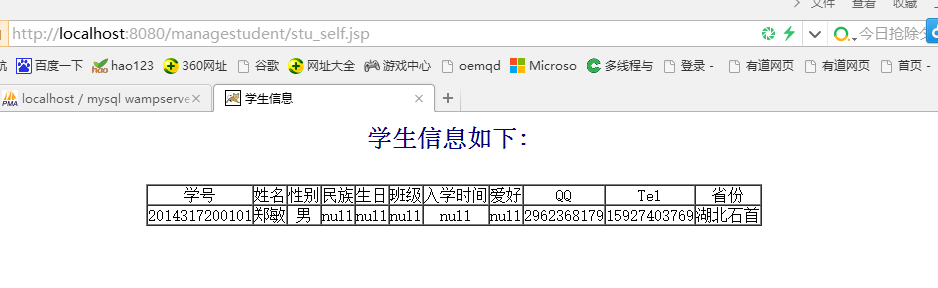
|  |
| --- |
| **Student\_index.jsp(部分代码)** |
| <tr>  <td width=*"25%"*><a href=*"stu\_self.jsp"* target=*"main"*>查看个人信息</td>  <td width=*"24%"*><a href=*"stu\_message\_modify.jsp"* target=*"main"*>修改个人信息</td>  <td width=*"23%"*><a href=*"student\_sel\_other\_cours.jsp?id=*<%=username %>*"*>查看成绩信息</td>  <td width=*"7%"*><a href=*"login.jsp"* target=*"main"*>退出</td>  </tr> |

1. 查看个人信息

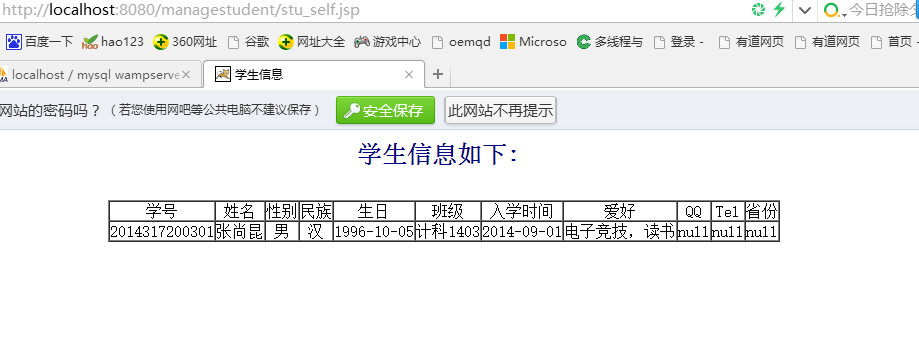
|  |
| --- |
| **Stu\_self.jsp(显示不同班级学生的个人信息)** |
| <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">  <html>  <head>  <title>学生信息</title>  <base href=*"*<%=basePath%>*"*>  <jsp:useBean id=*"sdao"* scope=*"page"* class=*"DAO.StudentDAO"*></jsp:useBean>  <title></title>  </head>  <body bgcolor=*""*>  <%  request.setCharacterEncoding("utf-8");  String name=(String)session.getAttribute("username");  String password=(String)session.getAttribute("password");  Vector vector=sdao.SelectBySid(name);  **int** i=vector.size();  student stu[]=**new** student[i];    %>  <div align=*"center"*><font size=*"5"* color=*"#000080"* face=*"华文宋体"*><b>学生信息如下：<br><br></b></font></div>  <table border=*"1"* cellspacing=*"0"* cellpadding=*"0"* align=*"center"*>    <tr align=*"center"*><td >学号</td><td>姓名</td><td >性别</td><td>民族</td><td>生日</td><td>班级</td><td>入学时间</td><td>爱好</td><td>QQ</td><td>Tel</td><td>省份</td></tr>  <%  **for**(**int** j=0;j<i;j++){  stu[j]=(student)vector.elementAt(j);  String rsid=**null**;  **if**(name.indexOf("003")>0)  {  rsid = stu[j].getId() ;  }  **else** **if**(name.indexOf("001")>0)  {  rsid= stu[j].getSid();  System.out.println(rsid);  }  String rsname = stu[j].getName();  String rsex=stu[j].getSex();  String rnation=stu[j].getNation();  String rbirth=stu[j].getBirth();  String rsclass = stu[j].getClas() ;  String renrollment\_date=stu[j].getEnrollment\_date();  String rhobby=stu[j].getHobby();  String rqq = stu[j].getQq() ;  String rtel=stu[j].getTel();  String rprov=stu[j].getProvince();  %>  <tr align=*"center"*>  <td><%=rsid%></td>  <td><%=rsname%></td>  <td><%=rsex%></td>  <td><%=rnation%></td>  <td><%=rbirth%></td>  <td><%=rsclass%></td>  <td><%=renrollment\_date%></td>  <td><%=rhobby%></td>  <td><%=rqq%></td>  <td><%=rtel%></td>  <td><%=rprov%></td>  </tr>  <%  }  **if**(i==0){  %>  <tr><th colspan=*"11"*>无该学生信息！</th></tr>  <%} %>  </table>    </body>  </html> |

界面显示

一班学生



三班学生



1. 修改个人信息

学生信息Servlet代码编写

|  |
| --- |
| **Student\_slt.java(部分代码)接收jsp界面信息** |
| **public** **class** student\_slt **extends** HttpServlet {  **public** **void** doPost(HttpServletRequest request, HttpServletResponse response)  **throws** ServletException, IOException {  response.setContentType("text/html");  request.setCharacterEncoding("utf-8");  response.setCharacterEncoding("utf-8");  String action = request.getParameter("action");  **if**("modify".equalsIgnoreCase(action)){  doUpdate(request,response);    }  …………  }  **public** **void** doUpdate(HttpServletRequest request, HttpServletResponse response)  **throws** ServletException, IOException {  response.setContentType("text/html");  request.setCharacterEncoding("utf-8");  response.setCharacterEncoding("utf-8");  PrintWriter out = response.getWriter();  String sid=request.getParameter("sid");  String sname=request.getParameter("sname");  String ssex=request.getParameter("ssex");  String snation=request.getParameter("snation");  String sbirth=request.getParameter("sbirth");  String sclas=request.getParameter("sclas");  String senrollment\_date=request.getParameter("senrollment\_date");  String shobby=request.getParameter("shobby");  String spassword=request.getParameter("spassword");    String sqq=request.getParameter("sqq");  String stel=request.getParameter("stel");  String sprov=request.getParameter("sprov");    StudentDAO sdao=**new** StudentDAO();  **try** {  **if**(sid.indexOf("003")>0)  sdao.UpdatebyID(sid,sname,ssex,snation,sbirth,sclas,senrollment\_date,shobby,spassword);  **else** **if**(sid.indexOf("001")>0){  sdao.UpdatebyID1(sid,sname,ssex,spassword,sqq,stel,sprov);    }  out.println("<HTML>");  out.println(" <HEAD><TITLE>A Servlet</TITLE></HEAD>");  out.println(" <BODY bgcolor=>");  out.print(" <font size=6 color=black> 修改成功！</font> ");  out.println(" </BODY>");  out.println("</HTML>");  } **catch** (Exception e) {  out.println("<HTML>");  out.println(" <HEAD><TITLE>A Servlet</TITLE></HEAD>");  out.println(" <BODY bgcolor=>");  out.print(" <font size=6 color=black>修改失败！</font> ");  out.println(" </BODY>");  out.println("</HTML>");  }  out.flush();  out.close();  } |

|  |
| --- |
| **StudentDAO.java(部分代码)数据库操作** |
| **public** **void** UpdatebyID(String sid,String sname,String ssex,String snation,String sbirth,String sclas,String senrollment\_date,String shobby,String spassword) **throws** Exception{    String sql="UPDATE `student` set `name`='"+sname+"',`sex`='"+ssex+"',`nation`='"+snation+"',`birth`='"+sbirth+"',`clas`='"+sclas+"',`enrollment\_date`='"+senrollment\_date+"',`hobby`='"+shobby+"',`password`='"+spassword+"' WHERE `stu-id`='"+sid+"';";  System.*out*.println("修改:"+sql);  Statement stmt=**null**;  DatabaseConnect dbc=**null**;  dbc=**new** DatabaseConnect();  **try**{  stmt=dbc.*getConnection*('a').createStatement();  stmt.executeUpdate(sql);  stmt.close();  }**catch** (Exception e){  **throw** **new** Exception("修改中出现错误！！！");  }**finally** {  dbc.close();    }    }  **public** **void** UpdatebyID1(String sid,String sname,String ssex,String spassword,String sqq,String stel,String sprov) **throws** Exception{    String sql="UPDATE `student` set `name`='"+sname+"',`sex`='"+ssex+"',`qq`='"+sqq+"',`tel`='"+stel+"',`province`='"+sprov+"',`password`='"+spassword+"' WHERE `sid`='"+sid+"';";  System.*out*.println("修改:"+sql);  Statement stmt=**null**;  DatabaseConnect dbc=**null**;  dbc=**new** DatabaseConnect();  System.*out*.println(sql);  **try**{  stmt=dbc.*getConnection*('b').createStatement();  stmt.executeUpdate(sql);  stmt.close();  }**catch** (Exception e){  **throw** **new** Exception("修改中出现错误！！！");  }**finally** {  dbc.close();    }  } |

界面显示（这里为简便，只修改密码）

* 先查看数据库待修改学生的基本信息

1班学生密码为“01”



3班学生密码为“3”

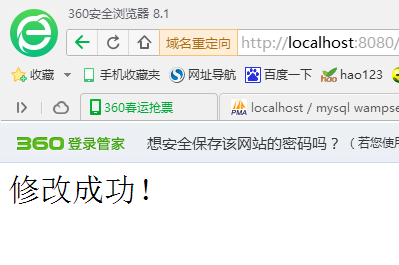


* 执行修改操作

1班学生：修改密码为“02”



点击提交，成功修改



查看数据库，验证密码修改成功



3班学生：修改密码为“4”



点击提交，成功修改



查看数据库，验证密码成功修改

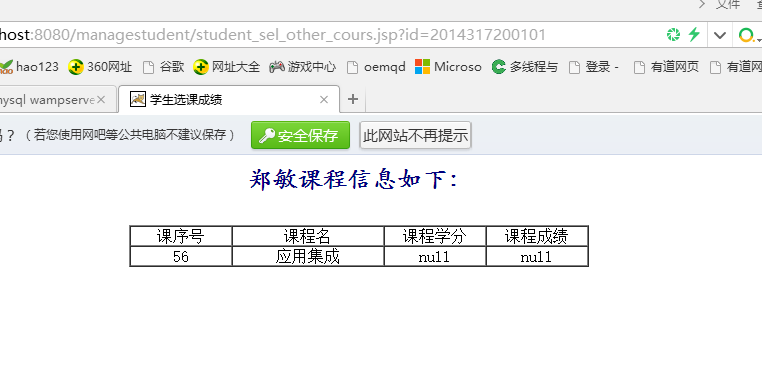


1. 查看成绩信息

|  |
| --- |
| **ElectDAO.java(部分代码)数据库操作** |
| **public** Vector ScSelect(String sid)**throws** Exception{  Vector all=**new** Vector();  String sql="SELECT course.`cour-id`,course.`cour-name`, elect.`cour-grade`, course.`credit`, elect.`stu-id` FROM elect,course where elect.`cour-id`=course.`cour-id` and elect.`stu-id`='"+sid+"';";  System.*out*.println(sql);  Statement stmt=**null**;  DatabaseConnect dbc=**null**;  dbc=**new** DatabaseConnect();  **try**{  stmt = dbc.*getConnection*('a').createStatement();  ResultSet rs = stmt.executeQuery(sql);  **while**(rs.next()) {  Elect cou=**new** Elect();  cou.setCourid(rs.getString("cour-id"));  cou.setCourgrade(rs.getString("cour-grade"));  all.addElement(cou);    }  System.*out*.println("11111111");  System.*out*.println(all.size());  rs.close();  stmt.close();  } **catch** (Exception e) {  **throw** **new** Exception("查询中出现错误！！！");  } **finally** {  dbc.close();  }  **return** all;    }    **public** Vector ScSelect1(String sid)**throws** Exception{  Vector all=**new** Vector();  String sql="SELECT elective\_course.`eid`,elective\_course.`cname`,stu\_ele\_course.`sid` FROM stu\_ele\_course,elective\_course where stu\_ele\_course.`eid`=elective\_course.`eid` and stu\_ele\_course.`sid`='"+sid+"';";  System.*out*.println(sql);  Statement stmt=**null**;  DatabaseConnect dbc=**null**;  dbc=**new** DatabaseConnect();  **try**{  stmt = dbc.*getConnection*('b').createStatement();  ResultSet rs = stmt.executeQuery(sql);  **while**(rs.next()) {  Elect cou=**new** Elect();  cou.setEid(rs.getString("eid"));    all.addElement(cou);    }  System.*out*.println("11111111");  System.*out*.println(all.size());  rs.close();  stmt.close();  } **catch** (Exception e) {  **throw** **new** Exception("查询中出现错误！！！");  } **finally** {  dbc.close();  }  **return** all;    }      **public** Vector SelectStudent(String sid)**throws** Exception{  Vector all=**new** Vector();  String sql="SELECT student.`stu-id`, student.`name`, course.`credit`, elect.`cour-id`, elect.`cour-grade` FROM elect,student,course where elect.`stu-id`=student.`stu-id` and elect.`cour-id`=course.`cour-id` ORDER BY elect.`cour-grade` DESC;";  Statement stmt=**null**;  DatabaseConnect dbc=**null**;  dbc=**new** DatabaseConnect();  **try**{  stmt = dbc.*getConnection*('a').createStatement();  ResultSet rs = stmt.executeQuery(sql);  **while**(rs.next()) {  student cou=**new** student();  cou.setId(rs.getString("stu-id"));  cou.setName(rs.getString("name"));  all.addElement(cou);    }  rs.close();  stmt.close();  } **catch** (Exception e) {  **throw** **new** Exception("查询中出现错误！！！");  } **finally** {  dbc.close();  }  **return** all;  }    **public** Vector SelectCourse(String sid)**throws** Exception{  Vector all=**new** Vector();  String sql="SELECT course.`cour-id`,course.`cour-name`, elect.`cour-grade`, course.`credit`, elect.`stu-id` FROM elect,course where elect.`cour-id`=course.`cour-id` and elect.`stu-id`='"+sid+"';";  Statement stmt=**null**;  DatabaseConnect dbc=**null**;  dbc=**new** DatabaseConnect();  System.*out*.println(sql);  **try**{  stmt = dbc.*getConnection*('a').createStatement();  ResultSet rs = stmt.executeQuery(sql);  **while**(rs.next()) {  Course cou=**new** Course();  cou.setCourid(rs.getString("cour-id"));  cou.setCourcredit(rs.getString("credit"));  cou.setCourname(rs.getString("cour-name"));  all.addElement(cou);  }  rs.close();  stmt.close();  } **catch** (Exception e) {  **throw** **new** Exception("查询中出现错误！！！");  } **finally** {  dbc.close();  }  **return** all;  }    **public** Vector SelectCourse1(String sid)**throws** Exception{  Vector all=**new** Vector();  String sql="SELECT elective\_course.`eid`,elective\_course.`cname`,stu\_ele\_course.`sid` FROM stu\_ele\_course,elective\_course where stu\_ele\_course.`eid`=elective\_course.`eid` and stu\_ele\_course.`sid`='"+sid+"';";  Statement stmt=**null**;  DatabaseConnect dbc=**null**;  dbc=**new** DatabaseConnect();  System.*out*.println(sql);  **try**{  stmt = dbc.*getConnection*('b').createStatement();  ResultSet rs = stmt.executeQuery(sql);  **while**(rs.next()) {  Course cou=**new** Course();  cou.setEid(rs.getString("eid"));    cou.setCname(rs.getString("cname"));  all.addElement(cou);  }  rs.close();  stmt.close();  } **catch** (Exception e) {  **throw** **new** Exception("查询中出现错误！！！");  } **finally** {  dbc.close();  }  **return** all;  } |

界面展示

* 1班学生



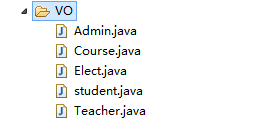
因为1班数据库中，没有课程学分，和课程成绩，所以显示为null

* 3班学生



1. 数据库中各表封装类实现（将两个数据库中相同功能表中的字段封装到一个类中）

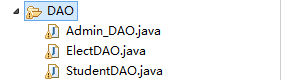
共需要编写如下类



例：Student类如下(其他同理)

|  |
| --- |
| **package** VO;  **public** **class** student {  **private** String id;  **private** String name;  **private** String sex;  **private** String nation;  **private** String birth;  **private** String clas;  **private** String enrollment\_date;  **private** String hobby;  **private** String password;  **private** String sid;  **private** String qq;  **private** String tel;  **private** String province;  **public** String getQq() {  **return** qq;  }  **public** String getTel() {  **return** tel;  }  **public** String getProvince() {  **return** province;  }  **public** String getSid() {  **return** sid;  }  **public** String getId() {  **return** id;  }  **public** **void** setQq(String qq) {  **this**.qq = qq;  }  **public** **void** setTel(String tel) {  **this**.tel = tel;  }  **public** **void** setProvince(String province) {  **this**.province = province;  }  **public** **void** setPassword(String password) {  **this**.password = password;  }  **public** String getPassword() {  **return** password;  }  **public** **void** setId(String id) {  **this**.id = id;  }  **public** **void** setSid(String sid) {  **this**.sid = sid;  }  **public** String getName() {  **return** name;  }  **public** **void** setName(String name) {  **this**.name = name;  }  **public** String getSex() {  **return** sex;  }  **public** **void** setSex(String sex) {  **this**.sex = sex;  }  **public** String getNation() {  **return** nation;  }  **public** **void** setNation(String nation) {  **this**.nation = nation;  }  **public** String getBirth() {  **return** birth;  }  **public** **void** setBirth(String birth) {  **this**.birth = birth;  }  **public** String getClas() {  **return** clas;  }  **public** **void** setClas(String clas) {  **this**.clas = clas;  }  **public** String getEnrollment\_date() {  **return** enrollment\_date;  }  **public** **void** setEnrollment\_date(String enrollment\_date) {  **this**.enrollment\_date = enrollment\_date;  }  **public** String getHobby() {  **return** hobby;  }  **public** **void** setHobby(String hobby) {  **this**.hobby = hobby;  }  } |

1. 用于数据库查询的java类共需编写如下程序



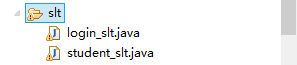
各部分代码已在上述各功能处展示

1. 为servlet配置xml文件

因为本工程共需servlet分别如下

Login\_slt.java 用于判断登录用户身份（管理员、计科1班学生、计科3班学生）

Student\_slt.java用于接收前台界面对学生进行”信息增加、修改、删除”操作提交到后台数据库中的数据



配置工程文件managestudent下的wel.xml文件

添加如下代码

|  |
| --- |
| <servlet>  <description>This is the description of my J2EE component</description>  <display-name>This is the display name of my J2EE component</display-name>  <servlet-name>login\_slt</servlet-name>  <servlet-class>slt.login\_slt</servlet-class>  </servlet>  <servlet>  <description>This is the description of my J2EE component</description>  <display-name>This is the display name of my J2EE component</display-name>  <servlet-name>student\_slt</servlet-name>  <servlet-class>slt.student\_slt</servlet-class>  </servlet>  <servlet-mapping>  <servlet-name>login\_slt</servlet-name>  <url-pattern>/login\_slt</url-pattern>  </servlet-mapping>  <servlet-mapping>  <servlet-name>student\_slt</servlet-name>  <url-pattern>/student\_slt</url-pattern>  </servlet-mapping> |

**第二部分：通过xml中间件的配置，集成四个班的学生信息系统（功能不完善，正在学习中）**

1. 核心代码

|  |
| --- |
| 数据库映射到xml文件 |
| hibernate-configuration>  <session-factory>  <property name="hibernate.connection.url">  jdbc:mysql://localhost/db\_stuman  </property>  <property name="hibernate.connection.driver\_class">  com.mysql.jdbc.Driver  </property>  <property name="hibernate.connection.username">root</property>  <property name="hibernate.connection.password">6347097</property>  <property name="hibernate.dialect">  org.hibernate.dialect.MySQLDialect  </property>  <property name="hibernate.show\_sql">true</property>  <property name="hibernate.hbm2ddl.auto">update</property>  <mapping resource="com/sanqing/course/model/Course.hbm.xml" />  <mapping resource="com/sanqing/course/model/CourseSchedule.hbm.xml" />  <mapping resource="com/sanqing/course/model/Mark.hbm.xml" />  <mapping resource="com/sanqing/course/model/Student.hbm.xml" />  <mapping resource="com/sanqing/course/model/Teacher.hbm.xml" />  <mapping resource="com/sanqing/course/model/Team.hbm.xml" />  <mapping resource="com/sanqing/course/model/User.hbm.xml" />  </session-factory>  </hibernate-configuration> |

|  |
| --- |
| **数据库连接实现.java** |
| package com.sanqing.course.util;  import org.hibernate.cfg.Configuration;  import org.hibernate.tool.hbm2ddl.SchemaExport;  public class ExportDB {  public static void main(String[] args) {    //读取hibernate.cfg.xml文件  Configuration cfg = new Configuration().configure();    SchemaExport export = new SchemaExport(cfg);    export.create(true, true);  }  } |

|  |
| --- |
| **数据中student表映射到xml文件** |
| <?xml version="1.0"?>  <!DOCTYPE hibernate-mapping PUBLIC  "-//Hibernate/Hibernate Mapping DTD 3.0//EN"  "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">  <hibernate-mapping package="com.sanqing.course.model">    <class name="Student" table="Student" lazy="false">    <id name="id" unsaved-value="null">  <generator class="uuid.hex"/>  </id>  <property name="code" not-null="false"/>  <property name="name" not-null="false"/>  <property name="enrollDate" not-null="false"/>  <property name="birthday" not-null="false"/>  <property name="sex" not-null="false"/>    <!-- 级联班级 -->  <many-to-one name="team" column="teamId" class="Team" />    </class>  </hibernate-mapping> |

|  |
| --- |
| **数据库中course表映射到xml文件** |
| <?xml version="1.0"?>  <!DOCTYPE hibernate-mapping PUBLIC  "-//Hibernate/Hibernate Mapping DTD 3.0//EN"  "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">  <hibernate-mapping package="com.sanqing.course.model">    <class name="Course" table="Course" lazy="false">    <id name="id" unsaved-value="null">  <generator class="uuid.hex"/>  </id>    <property name="name" not-null="false"/>    <!-- 级联Teacher -->  <set name="teachers" table="teacher\_course" inverse="true" lazy="true">  <key column="courseId"></key>  <many-to-many class="Teacher" column="teacherId"/>  </set>  </class>  </hibernate-mapping> |

用于数据库操作的java程序实现

|  |
| --- |
| **检索学生** |
| public ActionForward findAll(ActionMapping mapping, ActionForm form,HttpServletRequest request, HttpServletResponse response) throws Exception {  List<Student> list = studentDao.findAllStudents();  request.setAttribute("students", list);  return mapping.findForward("list");  } |
| **删除学生** |
| public ActionForward delete(ActionMapping mapping, ActionForm form,HttpServletRequest request, HttpServletResponse response) throws Exception {  //获取从页面表单中提交过来的值  StudentForm sf = (StudentForm)form;  try {  //调用业务逻辑操作  studentDao.deleteStudent(sf.getSelectFlag());  saveMessage(request, "studentForm.deleted");  } catch(Exception e) {  e.printStackTrace();  saveMessage(request, "studentForm.deleted.error");  }  ActionForward af = new ActionForward("student.do?p=list&pageNo=1", true);  return af;  } |
| **修改学生** |
| public ActionForward edit(ActionMapping mapping, ActionForm form,HttpServletRequest request, HttpServletResponse response) throws Exception {  String id = request.getParameter("id");  if(id != null && id.trim().length() > 0) {  Student student = studentDao.findStudentById(id);  //将Student的值复制给form  if(student != null) {  request.setAttribute("student", student);  }  }  List<Team> teams = teamDao.findAllTeams();  request.setAttribute("teams", teams);  return mapping.findForward("edit");  } |

学生信息操作封装类

|  |
| --- |
| **StudentDAO.java** |
| package com.sanqing.course.dao;  import java.util.List;  import com.sanqing.course.model.Course;  import com.sanqing.course.model.Student;  import com.sanqing.course.model.Teacher;  import com.sanqing.course.model.Team;  import com.sanqing.course.util.PageModel;  public interface StudentDAO {    //添加学生  public void addStudent(Student student) throws Exception;  //修改学生  public void modifyStudent(Student student) throws Exception;  //删除学生  public void deleteStudent(String[] studentIdList) throws Exception;    //根据Id查询学生  public Student findStudentById(String id) throws Exception;    //查询全部学生  public List<Student> findAllStudents() throws Exception;    } |

|  |
| --- |
| **StudentDAOImple.java** |
| public class StudentDAOImpl extends HibernateDaoSupport implements StudentDAO {  // 添加学生  public void addStudent(Student student) throws Exception{    this.getHibernateTemplate().save(student);  }  //修改学生  public void modifyStudent(Student student) throws Exception {  this.getHibernateTemplate().update(student);    }  // 删除学生  public void deleteStudent(String[] studentIdList) throws Exception {  for (int i=0; i<studentIdList.length; i++) {  Student student = (Student)this.getHibernateTemplate().load(Student.class, studentIdList[i]);  this.getHibernateTemplate().delete(student);  }  }  //查询全部学生  @SuppressWarnings("unchecked")  public List<Student> findAllStudents() {  List<Student> studentList = new ArrayList<Student>();  Session session = this.getSession();  Query query = session.createQuery("from Student s");  studentList = query.list();  return studentList;  }  //根据Id查询学生  public Student findStudentById(String id) {  Student student = (Student)this.getHibernateTemplate().load(Student.class, id);  return student;  }  } |

1. 实验总结

本项目工程第一部分实现两个数据库的连接操作，采用两个变量控制不同的用户登录，连接不同的数据库。该种方法实现的数据集成不具有灵活性，当需要再添加一个数据库时，需要修改多处代码。

本项目工程第二部分实现两个数据库的连接操作，是基于中间件的集成方法。每个数据源都用一个封装器进行封装（这里采用xml数据集成技术）。后台数据库用xml语言格式存储描述数据库的结构化信息。包括各表的结构化信息，表与表之间的级联信息。用户在全局模式的基础上向中间件发出数据库操作请求，中间件处理用户的请求，与各封装器交互，转化为各个数据源能够处理的子操作请求。

1. 实验感悟

通过本项目的设计开发，深入了解了数据集成的概念，现实价值。并学习了MVC架构基于中间件的数据集成方法以及xml可扩展标记语言用于描述数据库结构化信息。

通过本项目深入了解MVC【Model（模型），View( 视图 )，Control（控制）】设计模式,该模式将界面，业务逻辑，与数据三者分离开。本项目采用的典型模式为JSP+Servlet+Javabean，其中jsp为用户浏览器的界面显示，javabean为用户数据库的操作，servlet为用户接受在界面提交的对数据库的操作，并动态配置web.xml文件。在本项目中界面实现都是(.jsp)程序，servlet实现与javabean实现都是(.java)程序

通过java web学生信息管理系统的设计，学习到MVC的设计模式下的数据集成包括三个层次“用户层、中间件层、数据库层”其中中间件负责各异构数据源的数据访问、查询和协调数据源之间的信息。同时为用户提供统一的数据库访问接口。中间件的功能包括三部分“分解与合并查询用户请求、元数据的设计、查询数据库”。分解：用于将用户对数据可的请求操作分解为与各数据库对应的子请求；合并：用于将对各数据库的曹组结果合并后返回给用户；元数据：用于将异构数据源封装为统一的格式。查询：用于对不同的数据库进行增删改查等操作。