

Web应用开发实验报告

**(2018级)**



实验题目 实验7 DAO设计模式的应用

姓名

学号

专业 软件工程 班

学院 计算机科学与技术学院

提交日期 2020年5月31日

**目录**

**[第1题 3](#_Toc27990)**

**[1.1需求分析 3](#_Toc6996)**

[1、功能描述 3](#_Toc23371)

[2、输入信息 3](#_Toc18116)

[3、输出信息 3](#_Toc9902)

**[1.2开发步骤 4](#_Toc29344)**

**[1.3运行结果 5](#_Toc5030)**

**[1.4实验收获 8](#_Toc2607)**

**[1.5源代码 8](#_Toc8499)**

**[第2题 9](#_Toc5627)**

**[2.1需求分析 9](#_Toc19736)**

[1、功能描述 9](#_Toc24214)

[2、输入信息 9](#_Toc24592)

[3、输出信息 9](#_Toc27253)

**[2.2开发步骤 9](#_Toc19644)**

**[2.3运行结果 11](#_Toc22447)**

**[2.4问题及解决方案 16](#_Toc16109)**

**[2.5实验收获 18](#_Toc22689)**

**[2.6源代码 18](#_Toc9771)**

**实验7 DAO设计模式的应用**

1、测试基于DAO的MVC设计模式实现的添加客户信息功能，包括BaseDao.java、CustomerDao.java、addCustomer.jsp、AddCustomerServlet.java,使之能正常运行。要求dao类在com.dao包下，servlet类在com.controller包下，javabeans类在com.model包下，数据库访问采用数据源实现，然后增加查询、列出所有客户信息、修改和删除等功能。

* 查询：输入客户姓名查询，支持模糊查询。
* 列出所有可出信息：按客户号、姓名、Email、余额和操作，其中操作包括修改和删除链接；
* 修改：根据客户号修改该客户的信息；
* 删除：根据客户号删除该客户的信息；

**1.1需求分析**

**1、功能描述**

添加客户页面：输入客户号、姓名、Email、余额后，按确定按钮，若出现错误则在本页面第一行显示“添加失败”或“出现异常”，若输入正确则显示“添加成功！”。

查询客户页面：输入客户姓名，按确定按钮跳转至客户信息页面。

查询所有客户超链接：2个页面都有该超链接，点击跳转至显示所有客户信息页面。

显示所有客户信息页面：每行最后一列操作包括修改和删除链接。点击链接可实现修改客户信息和删除客户功能。

**2、输入信息**

添加客户页面：输入客户号、姓名、Email、余额。

查询客户页面：输入客户姓名。

**3、输出信息**

Input a Customer页面显示添加客户的结果，客户信息页面显示根据客户姓名查询到的客户信息，显示模糊查询的客户信息页面显示查到的一个或多个客户信息的表格，显示所有客户信息页面以表格形式输出，修改结果页面和删除结果页面输出操作结果。

**1.2开发步骤**

基于DAO的MVC设计模式：dao类在com.dao包下，servlet类在com.controller包下，javabeans类在com.model包下，数据库访问采用数据源实现。

**javabeans类**:Customer.java 包括客户号、客户名、Email、余额4个属性

**dao类**：封装数据访问实现。

1. BaseDao.java 采用数据源实现数据库访问

2、CustomerDao.java 实现所有的数据库操作

* 添加客户
* 根据姓名查找客户
* 根据姓名模糊查询
* 查找所有的客户
* 根据客户编号查找客户
* 修改客户信息
* 根据客户编号删除客户信息

**servlet类**：处理jsp内容。

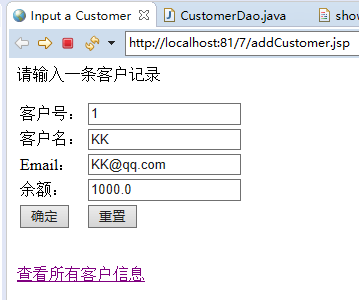
1. AddCustomerServlet.java 添加客户
2. QueryCustomer.java 查找客户
3. FuzzyQueryCustomer.java 模糊查询客户
4. ModifyCustomer.java 根据客户编号查找客户，添加到请求作用域，转发到修改页面modifyCustomer.jsp
5. ModifyCustomerInfo.java 修改客户信息
6. DeleteCustomer.java 删除客户
7. AllCustomer.java 列出所有客户信息

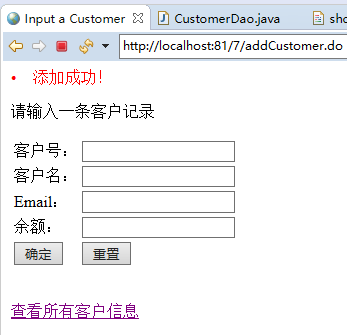
**jsp**:

1. addCustomer.jsp 第1行显示添加结果，还未添加则不显示此行。第2行显示“请输入一条客户记录”。<table>实现页面布局，表单中设置5行的表格，前4行为输入信息，第5行为“确定”“重置”按钮。点击确定跳转回本页面。设置一个“查看所有客户信息”的超链接，点击可跳转至showAllCustomer.jsp
2. queryCustomer.jsp 表单包括1行输入信息和“确定”按钮，点击确定跳转至showCustomer.jsp。第2行为“查询所有客户”的超链接，点击可跳转至showAllCustomer.jsp。
3. showCustomer.jsp 用<table>布局、EL表达式输出一个客户的信息。
4. fuzzyqueCustomer.jsp 页面布局和queryCustomer.jsp一样，输入框前的标签改为“请输入客户姓名或编号（支持模糊查询）：”。
5. showFuzzyQueryCustomer.jsp 输出一个或多个客户的信息，用到<table>布局、EL表达式和jsp标签。
6. modifyCustomer.jsp 一般不修改主键(cust\_id)的值。<table>布局，3行为要修改的属性，输入框起初显示属性原值，第4行为“确定”“重置”按钮。点击确定跳转至modifyCustomerResult.jsp。
7. modifyCustomerResult.jsp/deleteCustomerResult.jsp 显示修改/删除结果和返回链接，返回至showAllCustomer.jsp，信息会实时更新。
8. showAllCustomer.jsp 以表格形式列出所有可出信息：按客户号、姓名、Email、余额和操作，其中操作包括修改和删除链接，点击超链接跳转至相应操作的servlet并传递该列的客户号，用到<table>布局、EL表达式和jsp标签。

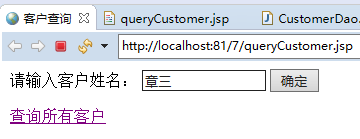
**1.3运行结果**

1. **添加客户页面**

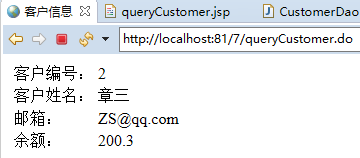




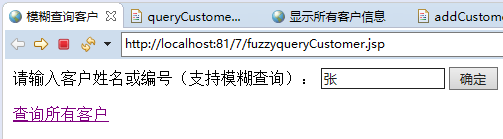
1. **客户查询页面**

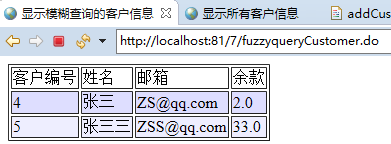


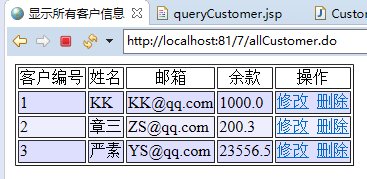
1. **查询结果**



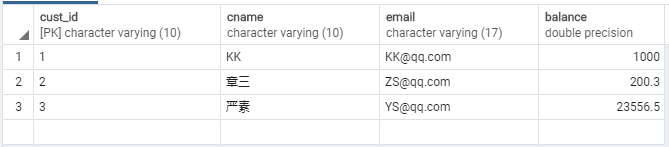
1. **模糊客户查询页面**



1. **模糊查询结果**
2. **显示所有客户信息页面**



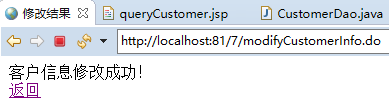
数据库中的数据：



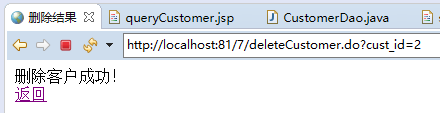
1. **修改**

修改余额为0

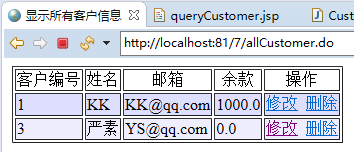
1. **修改结果**



1. **删除结果**



1. **返回**

返回到所有客户信息页面，实时更新

更新后的数据库信息：



**1.4实验收获**

1. 学会了用DAO封装数据访问实现，将业务逻辑和数据访问逻辑分离，从而使应用的维护变得简单，提高了应用程序的灵活性。
2. 学会了使用持久对象(可序列化的JavaBeans,要实现java.io.Serializable接口)保存应用数据，实现对象与关系数据的映射，更加方便地完成跨层传输数据。

**1.5源代码** 工程打包单独上传

2、实现某师生健康码管理系统的系统管理员管理功能，具体要求如下：

系统管理员登录后可设置学院、专业、班级等信息并进行查询、修改、删除等管理功能，可单独添加或批量导入教师数据和学生数据并进行查询、修改、删除等管理功能，教师数据包括姓名、身份证号、工号、学院、角色（系统管理员、校级管理员、院级管理员、普通教师），学生数据包括姓名、身份证号、学号、学院、专业、班级等信息。

**2.1需求分析**

**1、功能描述**

系统管理员登录后可设置学院、专业、班级等信息并进行查询、修改、删除等管理功能，可单独添加或批量导入教师数据和学生数据并进行查询、修改、删除等管理功能

**2、输入信息**

系统管理员登录页面：输入用户名和密码；

单独添加用户页面：输入教师或学生的数据；

筛选学生用户页面：输入学院、专业、班级；

查看所有用户页面底部：输入学号/工号。

**3、输出信息**

系统管理员登录成功页面：显示管理员信息和进入管理系统超链接；

管理系统：添加用户和批量导入显示操作结果，筛选显示查询到的学生，查看所有用户显示所有用户的信息，单独查询的用户显示一个用户的信息。

**2.2开发步骤**

基于DAO的MVC设计模式：dao类在com.dao包下，servlet类在com.controller包下，javabeans类在com.model包下，数据库访问采用数据源实现。

**javabeans类**:User.java、Student.java、Teacher.java、Sha256.java (把实验6做过的放到本实验中)

**dao类**：封装数据访问实现。

1、添加UserDao.java，实现数据库中对用户的更新操作

* 单独添加教师、学生
* 批量添加教师、学生
* 验证登录：根据工号、密码查找配对的系统管理员
* 根据工号(学号)查找教师(学生)
* 筛选学生：根据学院、专业、班级查找学生
* 查找所有教师、学生
* 修改教师、学生信息
* 根据工号(学号)删除教师(学生)信息

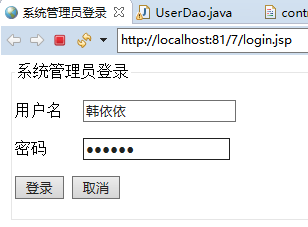
**servlet类**：处理jsp内容。

1. LoginServlet.java 管理员登录验证
2. AddUser.java 添加用户
3. ImportT.java、ImportS.java 批量导入教师、学生信息
4. ScreenS.java 筛选学生用户
5. AllT.java、AllS.java 列出所有教师、学生信息
6. ModifyT.java、ModifyS.java 根据工号(学号)查找教师(学生)，将查到的用户添加到请求作用域转发到修改页面
7. ModifyTInfo.java、ModifySInfo.java 得到修改页面输入的值，用新值修改用户信息，修改结果输出到修改页面
8. QueryT.java、QueryS.java 根据工号(学号)查找教师(学生)

**jsp**:

1. control.jsp 标题为“管理系统”，包含5个管理功能超链接，分别可以跳转到2-5
2. addUser.jsp 用实验6的注册页面即可，密码一行不用显示，表单标题为“添加用户(初始密码为1111)”
3. importUser.jsp 包含“批量导入学生数据”和“批量导入教师数据”2个超链接，点击进入importRes.jsp显示导入结果
4. screenS.jsp 包含学院、专业、班级3个输入框，查询按钮和返回超链接，点击查询跳转到showScreenS.jsp显示筛选结果
5. showAllS.jsp、showAllT.jsp 显示所有用户信息 底部可根据工号(学号)查找教师(学生)，点击查询跳转到showS.jsp(showT.jsp)显示查询结果，而后可点击“返回管理页面”超链接回到首页
6. modifyS.jsp、modifyT.jsp 第一行显示被修改者的姓名，后面输入修改信息
7. modifySRes.jsp、modifyTRes.jsp 显示修改结果和返回超链接
8. deleteSRes.jsp、deleteTRes.jsp 显示删除结果和返回超链接

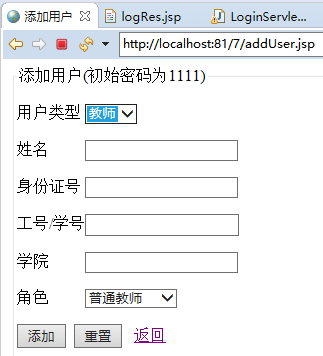
**2.3运行结果**

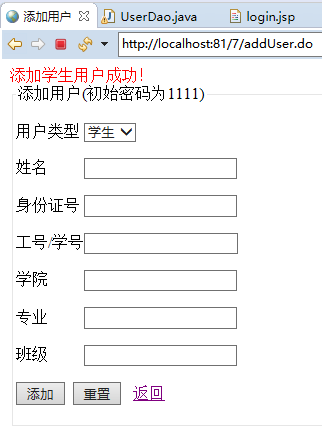
1. **系统管理员登录页面**

**2)管理系统页面**

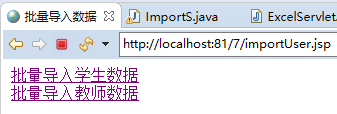


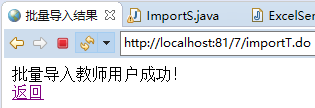
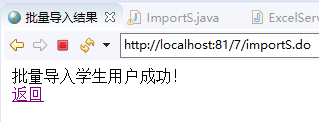
1. **添加用户页面**

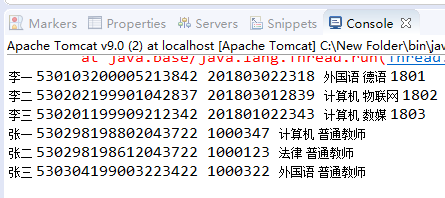


点击添加

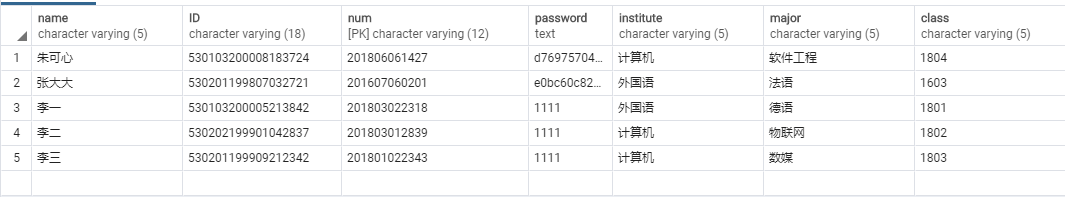
**4)批量导入**

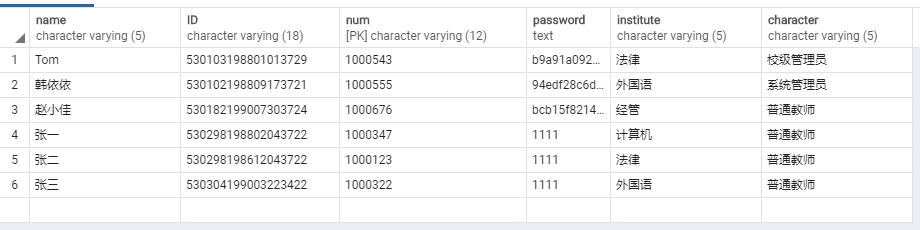




测试导入效果

数据库导入结果：

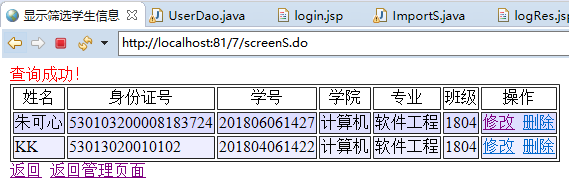




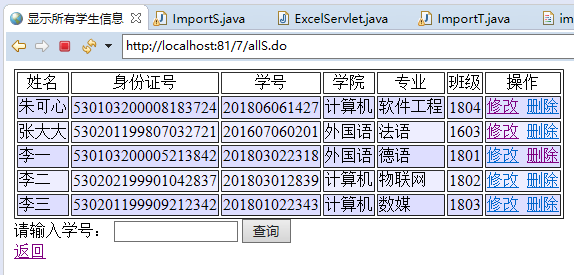
**5)根据学院、专业、班级筛选学生用户页面**



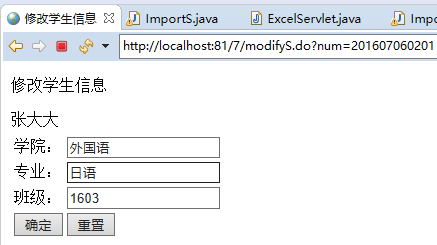
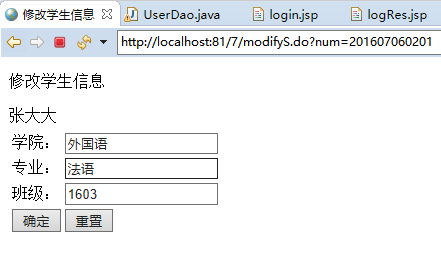
**6)显示筛选结果**



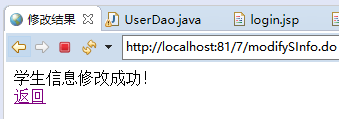
**7)显示所有学生信息页面**

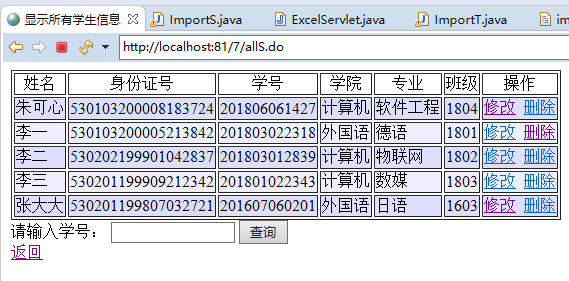


**8)修改操作**

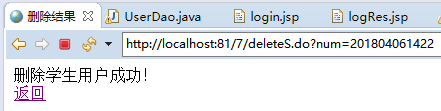


**9)修改结果**

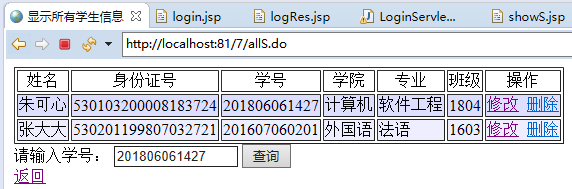


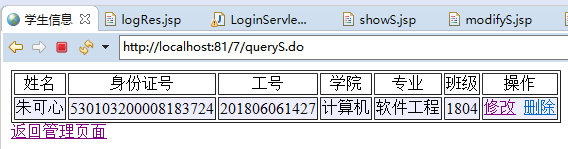


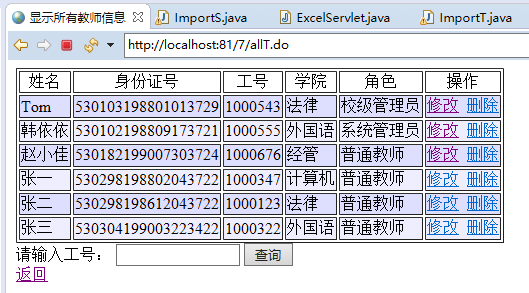
**10)删除结果**



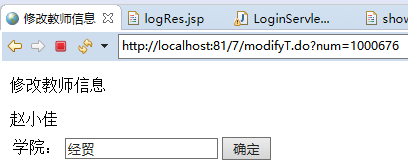
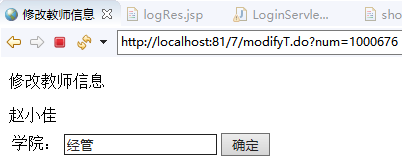
**11)查询学生**

点击查询

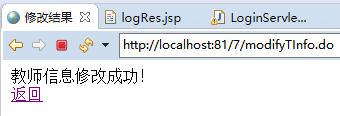


**12)显示所有教师信息页面**

**13)修改操作**

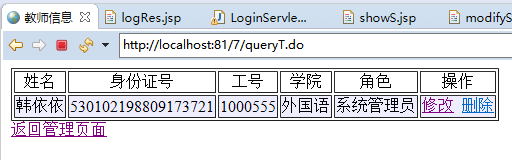
初始值修改后点击确定

**14)修改结果**



**15)查询教师**

IMG_256点击查询



**2.4问题及解决方案**

**问题**：筛选学生功能传递到筛选结果页面的学生变量为空；

**原因**：一定要注意pstmt.setString()和ResultSet rst = pstmt.executeQuery()的顺序，我原来把前者写到后者后面，固然得不到返回的结果；

**解决方案**：把这部分**try** (Connection conn = dataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql);

ResultSet rst = pstmt.executeQuery()) {

pstmt.setString(1, institute);

pstmt.setString(2, major);

pstmt.setString(3, clas); 写成这样：

**try** (Connection conn = dataSource.getConnection(); PreparedStatement pstmt = conn.prepareStatement(sql)) {

pstmt.setString(1, institute);

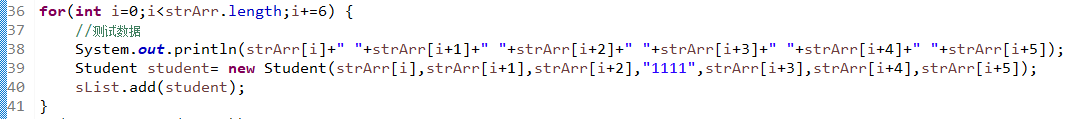
pstmt.setString(2, major);

pstmt.setString(3, clas);

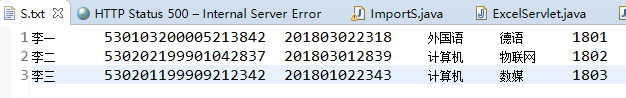
**try** (ResultSet rst = pstmt.executeQuery()) {

**问题**：批量导入时提示

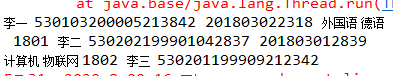


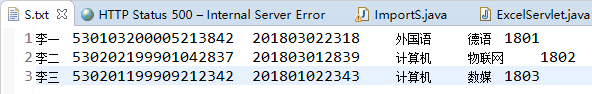
说明错误在这里面

**原因**：我输入数据时字符串间隔打了多个"\t"，且换行时没有"\t"隔开，导致读入时把一些"\t"当成元素读入，每行最后一个元素和第二行第一个元素被读成一个元素，超出范围的字符串没被读入



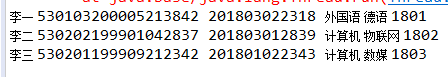
测试数据



**解决方案**：字符串间隔只用一个"\t"隔开

s1.append(s);写成s1.append(s+"\t");使每行最后一个元素和第二行第一个元素以"\t"隔开存入s1

测试数据



**2.5实验收获**

1、更加熟练掌握了DAO设计模式的应用，和MVC设计模式结合起来，使程序显得分类明显，井井有条。

**2.6源代码** 工程打包单独上传