**数据库系统课程设计**

**学生宿舍管理系统**

**实验报告**

**第十一组**

**2018年1月5日**

**组员分工情况**

|  |  |  |
| --- | --- | --- |
| **分工情况** | | |
| 1 | 刘艺语 | JDBC底层，8个界面实现，1个统计图实现，导入和导出报表功能，制作PPT，实验报告5、6部分 |
| 2 | 刘峰 | 界面设计，2个界面实现，修改了部分逻辑结构，修改了部分底层。实验报告3.5、3.6、4.3、5、6部分 |
| 3 | 刘渊晨 | 数据库管理，界面设计，2个界面实现，实验报告1-3.2、5、6部分 |
| 4 | 刘真 | JDBC底层，1个界面实现，1个统计图实现，制作PPT，实验报告5、6部分 |
| 5 | 张庆渝 | 2个界面实现，实验报告3.3、3.4、4.1、4.2、5、6部分 |

**目 录**

[1. 实验目的 5](#_Toc14267)

[2. 环境搭建 5](#_Toc7507)

[2.1. 选定语言或框架 5](#_Toc17714)

[2.2. 导入jar包 6](#_Toc16141)

[3. 系统实现 6](#_Toc19140)

[3.1. 需求分析 6](#_Toc4789)

[3.1.1. 信息要求 6](#_Toc13524)

[3.1.2. 功能模块 7](#_Toc28437)

[3.1.3. 数据项 8](#_Toc29482)

[3.2. 概念结构设计 10](#_Toc20564)

[3.2.1. 实体 10](#_Toc14026)

[3.2.2. 关系 11](#_Toc6902)

[3.2.3. E-R图 11](#_Toc13173)

[3.3. 逻辑结构设计 11](#_Toc14306)

[3.3.1. E-R图向关系模型的转换 11](#_Toc10768)

[3.3.2. 范式分析 13](#_Toc22930)

[3.3.3. 用户子模式 16](#_Toc27542)

[3.4. 物理结构设计 16](#_Toc31710)

[3.5. 数据库实施 17](#_Toc16441)

[3.5.1. 应用程序数据的载入 17](#_Toc31487)

[3.5.2. 程序的编码与调试 17](#_Toc15577)

[3.6. 数据库运行与维护 20](#_Toc17337)

[3.6.1. 运行 20](#_Toc8452)

[3.6.2. 维护 21](#_Toc28112)

[4. 系统运行结果 21](#_Toc27157)

[4.1. 登录界面 21](#_Toc23352)

[4.2. 学生界面 23](#_Toc27695)

[4.2.1. 学生主界面 23](#_Toc2455)

[4.2.2. 个人中心 23](#_Toc2085)

[4.2.3. 我的宿舍 24](#_Toc28953)

[4.2.4. 我的快递 25](#_Toc3472)

[4.2.5. 报修申请 25](#_Toc25256)

[4.2.6. 晚归申请 26](#_Toc4764)

[4.2.7. 离校信息 27](#_Toc10489)

[4.3. 教师界面 29](#_Toc5985)

[4.3.1. 教师主界面 29](#_Toc2847)

[4.3.2. 个人中心 29](#_Toc14517)

[4.3.3. 学生管理 30](#_Toc1141)

[4.3.4. 学生离/返校信息 34](#_Toc5852)

[4.3.5. 报修信息 35](#_Toc25892)

[4.3.6. 晚归信息 36](#_Toc30345)

[4.3.7. 财产管理 37](#_Toc26512)

[4.3.8. 学生快递管理 38](#_Toc5896)

[5. 实验遇到的问题与解决方法 40](#_Toc27245)

[6. 实验心得与体会 41](#_Toc20101)

# 实验目的

《数据库系统课程设计》是在学生系统地学习了《数据库系统》课程后，按照关系数据库的基本原理，综合运用所学的知识，特别是第七章的内容，设计开发一个小型的数据库管理信息系统。通过对一个实际问题的分析、设计与实现，将原理与应用相结合，学会如何把书本上学到的知识用于解决实际问题。

通过数据库课程设计，既培养学生的动手能力，又能使学生深入理解和灵活掌握教学内容。

# 环境搭建

## 选定语言或框架

本次实验选用JAVA语言和NetBeans IDE进行系统开发，使用SQL Server关系型数据库管理系统对数据进行管理。首先使用JAVA语言十分便捷，只需导入相应的jar包，使用JDBC驱动程序即可连接数据库；其次，NetBeans IDE 可进行界面可视化设计，功能强大，使用简单；最后，我们在数据库实验课上选用的是SQL Server DBMS，对其sql语句、使用方法已较为熟悉，所以在课程设计继续使用SQL Server进行数据管理。

## 导入jar包

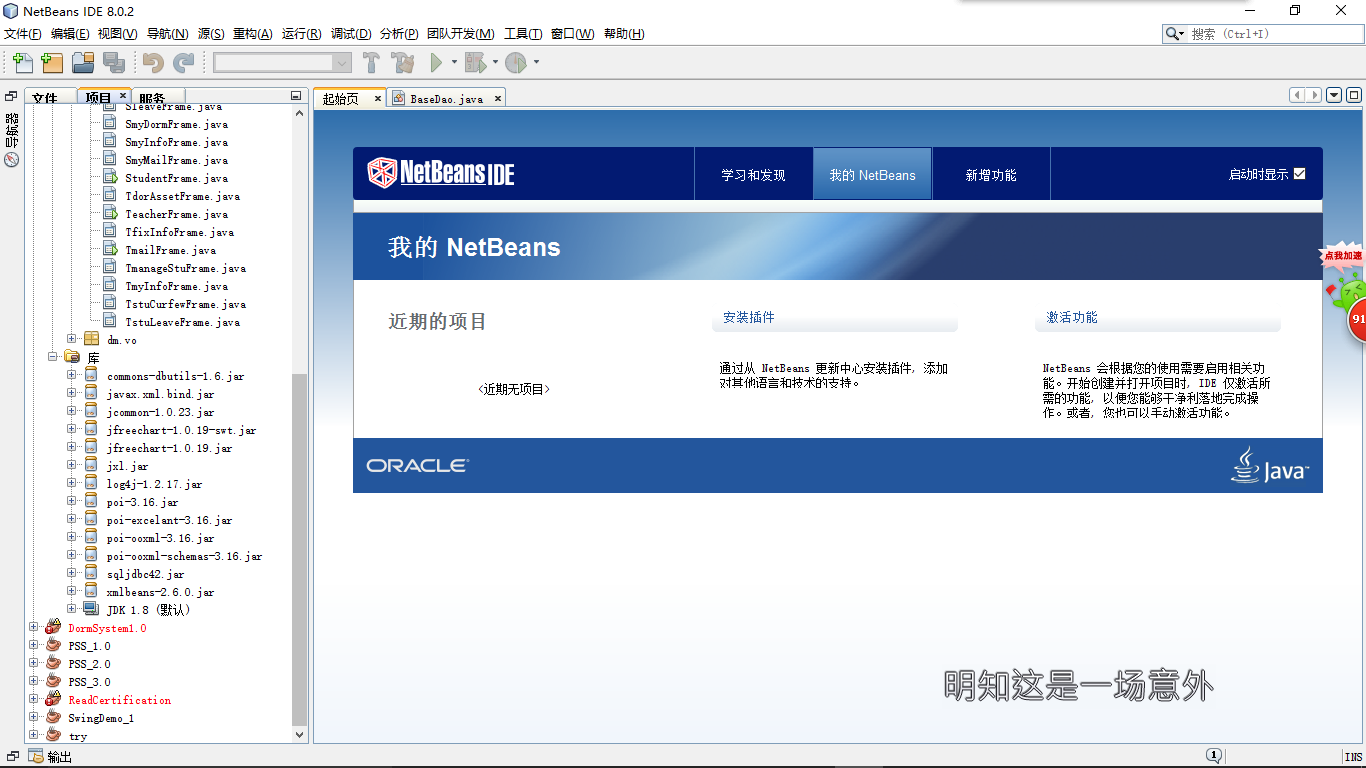


图 1 所有导入的jar包

# 系统实现

## 需求分析

### 信息要求

1. 学生的基本信息：入校时，每位同学都有唯一的学号，并被分配到指定的宿舍楼和指定的宿舍，也会有一个宿舍号，其入校时间就是他的入住时间。另外，为了管理方便，同一院系的学生的宿舍一般在一起，相应地会有其所在院系名称。
2. 宿舍的基本信息：每间宿舍都有唯一的宿舍号，入校时，宿舍会装公用电话机，相应地就有宿舍电话号码。
3. 宿舍财产的基本信息：每个宿舍的财产属于学校，比如电灯，床铺，柜子，桌椅等，为了对不同的财产进行区分，可以为每种财产分配不同的财产号。这样有利于财产的报修和管理。
4. 快件收发的基本信息：每栋宿舍楼都有邮件收发的管理，当本楼的同学有信件时，相应的会有信件的到达时间，接受信件的同学姓名和他所在的宿舍号。有时，同一个学生有多封信件需要接收，需要表示一个同学有多少封信件。当信件接收后，应有一个接收信件的时间，表示信件已成功到达指定的同学手中。
5. 报修的基本信息：宿舍楼中经常出现财产的损坏，比如灯泡坏了，厕所的马桶出故障了等，这时，同学们需要将财产损坏情况报告给宿舍楼管理员，以便学校派人进行维修。这时，需要记录报修的宿舍号和损坏的财产编号，同时记录报修的时间和损坏的原因。当损坏的财产维修完毕后，应记录解决时间，表示该报修成功解决。
6. 夜归的基本信息：宿舍楼在指定的时间关门（比如晚上12点），若有同学晚于关门时间回宿舍，需通知宿舍楼管理员，同时应登记晚归学生姓名，宿舍号，时间和晚归原因，以利于学校的管理和查证。
7. 离校的基本信息：每当放寒假或暑假时，同学们大部分都会回家；每当“五·一”或“十·一”放假时，同学们也有很多不会留在宿舍。这时，为加强学校对同学假期安全的管理，离校的同学应登记离校时间，待返校后记录返校时间，以便学校查证和管理。

### 功能模块

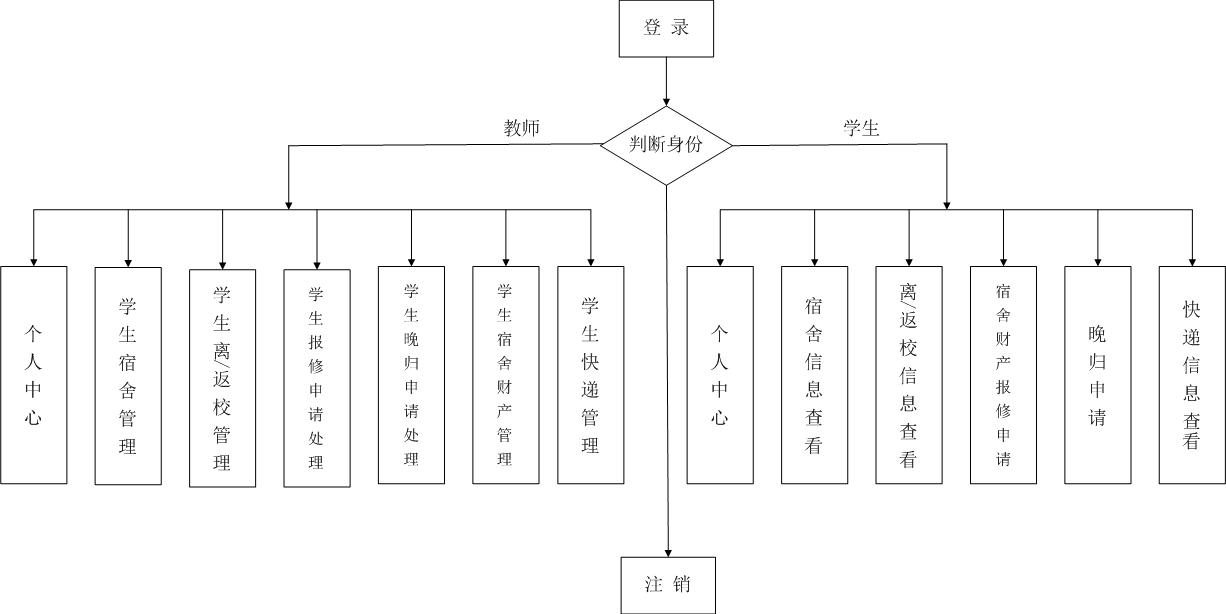


图 2 功能模块图

### 数据项

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据项** | **含义说明** | **符号表示** | **类型** | **长度** | **取值范围** | **取值含义** |
| 用户名 | 唯一标识每个用户 | Uno | char | 11 |  |  |
| 密码 | 用户登录系统的密码，可修改 | Pass | char | 20 |  |  |
| 权限 | 即用户身份，不同权限访问不同用户界面 | Pri | char | 1 | ‘0’或‘1’ | ‘0’代表学生，‘1’代表老师 |
| 学号 | 唯一标识每个学生 | Sno | char | 11 |  |  |
| 学生姓名 |  | Sname | char | 10 | 不能为空 |  |
| 学生性别 |  | Ssex | char | 2 | ‘男’或‘女’ |  |
| 年级 |  | Sgrade | char |  | 大于2000小于本年 |  |
| 院系 |  | Sdept | char | 20 |  |  |
| 入住时间 | 学生入校时间 | Scin | date | 无 |  |  |
| 教师号 | 唯一标识每个教师 | Tno | char | 11 |  |  |
| 教师姓名 |  | Tname | char | 10 |  |  |
| 教师性别 |  | Tsex | char | 2 | ‘男’或‘女’ |  |
| 宿舍号 | 唯一标识每个宿舍 | Dno | char | 6 | X#xxx | X表示宿舍楼，xxx表示宿舍号 |
| 宿舍规格 | 几人间 | Dsize | int |  | 大于0小于等于12 |  |
| 宿舍空床位 | 该宿舍可以添加的学生数量 | Dsurplus | int |  | 不能为空 |  |
| 宿舍电话 |  | Dtel | char | 20 | 不能为空 |  |
| 宿舍楼号 | 唯一标识每个宿舍楼 | Bno | char | 2 | 1-7 |  |
| 宿舍区 |  | Bblock | char | 6 | ‘北海苑’‘东海苑’‘南海苑’ |  |
| 门禁时间 |  | Btime | time |  |  |  |
| 财产号 | 唯一标识每种财产 | Ano | char | 3 |  |  |
| 财产名 |  | Aname | char | 20 |  |  |
| 宿舍财产数量 |  | Amount | int |  | 大于等于0 |  |
| 财产价格 |  | Aprice | int |  | 大于0 |  |
| 财产库存 |  | Astore | int |  | 大于等于0 |  |
| 报修时间 | 学生对损坏的财产申请报修日期 | ReportTime | date |  |  |  |
| 报修原因 | 描述财产损坏状况 | ReportReason | char | 50 |  |  |
| 解决时间 | 损坏财产修复日期 | FixTime | date |  |  |  |
| 快递号 | 唯一标识每个快递 | Mno | char | 20 |  |  |
| 快递到达时间 |  | Marrive | date |  |  |  |
| 快递接收时间 |  | Maccept | date |  |  |  |
| 学生离校时间 |  | Sltime | date |  |  |  |
| 学生返校时间 |  | Sreturn | date |  |  |  |
| 学生晚归时间 |  | NightTime | datetime |  |  |  |
| 学生晚归原因 |  | NightReason | char | 50 |  |  |

## 概念结构设计

### 实体

用户：{用户名，密码，权限}

学生：{学号，姓名，性别，年级，院系，宿舍号，入校时间}

教师：{教师号，姓名，性别}

宿舍：{宿舍号，宿舍规格，空床位数量，宿舍电话}

宿舍楼：{宿舍楼号，宿舍区，门禁时间}

财产：{财产号，财产名，财产价格，财产库存}

学生离校：{学号，离校时间，返校时间}

快递收发：{快递号，学号，到达时间，接收时间}

### 关系

学生晚归关系：{学号，晚归时间，晚归原因，教师号}

学生报修关系：{宿舍号，财产号，报修时间，报修原因，解决时间}

宿舍管理关系：{教师号，宿舍楼号}

宿舍财产关系：{宿舍号，财产号，财产数量}

### E-R图

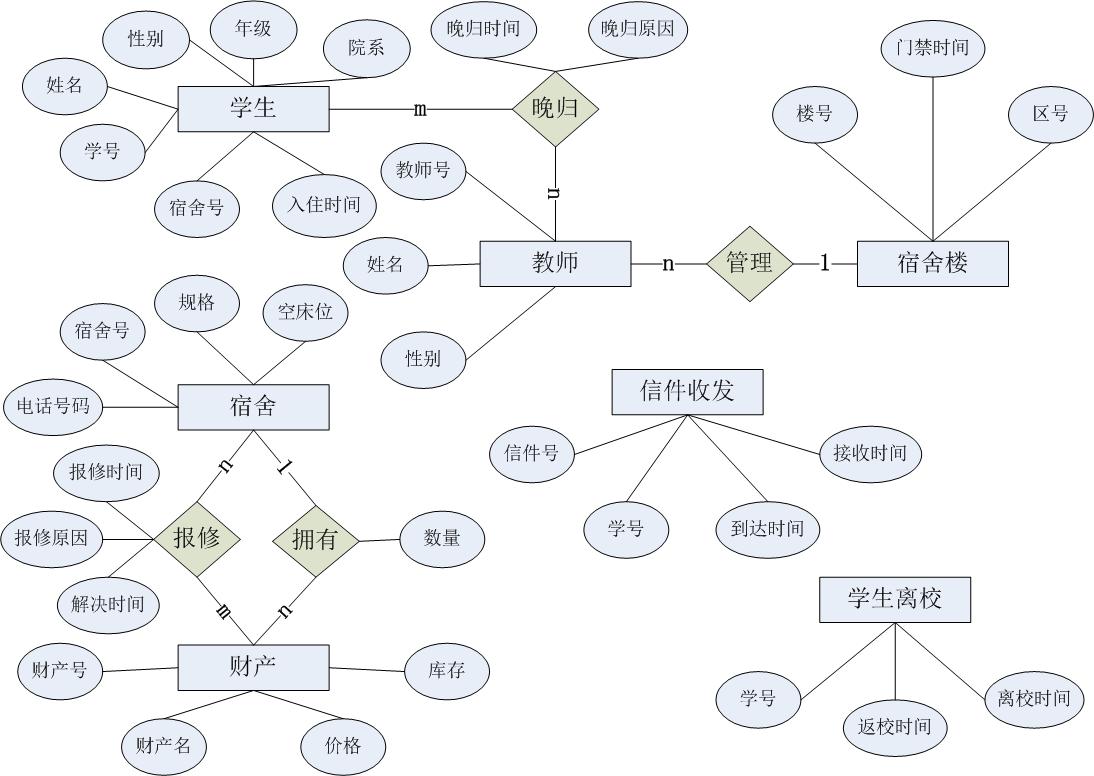


图 3 E-R图

## 逻辑结构设计

### E-R图向关系模型的转换

学生：{学号，姓名，性别，年级，院系，宿舍号，入校时间}

（符号表示：Student(Sno, Sname, Ssex, Sgrade, Sdept, Dno, Scin)）

此为学生实体对应的关系模式。

教师：{教师号，姓名，性别}

（符号表示：Teacher(Tno, Tname, Tsex)）

此为教师实体对应的关系模式。

宿舍：{宿舍号，宿舍规格，空床位数量，宿舍电话}

（符号表示：Dormitory(Dno, Bno, Dsize, Dsurplus, Dtel)）

此为宿舍实体对应的关系模式。

宿舍楼：{宿舍楼号，宿舍区，门禁时间}

（符号表示：Building(Bno, Bblock, Btime)）

此为宿舍楼实体对应的关系模式。

财产：{财产号，财产名，财产价格，财产库存}

（符号表示：Asset(Ano, Aname, Aprice, Astore)）

此为财产实体对应的关系模式。

学生离校：{学号，离校时间，返校时间}

（符号表示：LeaveSchool(Sno, Sltime, Sreturn)）

此为学生离校实体对应的关系模式。

快递收发：{快递号，学号，到达时间，接收时间}

（符号表示：Mail(Mno, Sno, Marrive, Maccept)）

此为快递收发实体对应的关系模式。

学生晚归关系：{学号，晚归时间，晚归原因，教师号}

（符号表示：Curfew(Sno, NightTime, NightReason, Tno)）

此为学生晚归关系对应的关系模式。

学生报修关系：{宿舍号，财产号，报修时间，报修原因，解决时间}

（符号表示：Repairs(Dno, Ano, ReportTime, ReportReason, FixTime)）

此为学生报修关系对应的关系模式。

宿舍管理关系：{教师号，宿舍楼号}

（符号表示：DT(Bno, Tno)）

此为宿舍管理关系对应的关系模式。

宿舍财产关系：{宿舍号，财产号，财产数量}

（符号表示：DA(Dno, Ano, Amount)）

此为宿舍财产关系对应的关系模式

用户：{用户名，密码，权限}

（符号表示：Users(Uno, Pass, Pri)）

此为用户实体对应的关系模式。

### 范式分析

针对关系模式

Student(Sno, Sname, Ssex, Sgrade, Sdept, Dno, Scin)

函数依赖：

SnoSname

SnoSsex

SnoSgrade

SnoSdept

SnoDno

SnoScin

结论：满足BCNF范式

针对关系模式

Teacher(Tno, Tname, Tsex)

函数依赖：

TnoTname

TnoTsex

结论：满足BCNF范式

针对关系模式

Dormitory(Dno, Bno, Dsize, Dsurplus, Dtel)

函数依赖：

DnoBno

DnoDsize

DnoDsurplus

DnoDtel

结论：满足BCNF范式

针对关系模式

Building(Bno, Bblock, Btime)

函数依赖：

BnoBblock

BnoBtime

结论：满足BCNF范式

针对关系模式

Asset(Ano, Aname, Aprice, Astore)

函数依赖：

AnoAname

AnoAprice

AnoAstore

结论：满足BCNF范式

针对关系模式

LeaveSchool(Sno, Sltime, Sreturn)

函数依赖：

(Sno, Sltime)Sreturn

结论：满足BCNF范式

针对关系模式

Mail(Mno, Sno, Marrive, Maccept)

函数依赖：

MnoSno

MnoMarrive

MnoMaccept

结论：满足BCNF范式

针对关系模式

Curfew(Sno, NightTime, NightReason, Tno)

函数依赖：

(Sno, NightTime)NightReason

(Sno, NightTime)Tno

结论：满足BCNF范式

针对关系模式

Repairs(Dno, Ano, ReportTime, ReportReason, FixTime)

函数依赖：

(Dno, Ano, ReportTime)ReportReason

(Dno, Ano, ReportTime)FixTime

结论：满足BCNF范式

针对关系模式

DT(Bno, Tno)

函数依赖：

BnoTno

结论：满足BCNF范式

针对关系模式

DA(Dno, Ano, Amount)

函数依赖：

(Dno, Ano)Amount

结论：满足BCNF范式

针对关系模式

Users(Uno, Pass, Pri)

函数依赖：

UnoPass

UnoPri

结论：满足BCNF范式

总结：由于在设计实体及其之间的联系的E-R图的时候，我们就尽量考虑了规范性的问题，又因为本数据库设计并不涉及多值依赖问题，所以本数据库是满足函数依赖范畴内规范化程度最高的BCNF范式的。数据冗余几乎没有。因此，也就不再需要优化。

### 用户子模式

创建视图：

1. 学生晚归信息视图：{学号，姓名，宿舍号，晚归时间，晚归原因，教师号}

此视图提供给老师和学生用户。学生申请晚归，教师决定是否批准。

1. 学生报修信息视图：{财产号，财产名，宿舍号，报修时间，报修原因，修理时间}

此视图提供给学生申请查看财产报修。

1. 学生查看宿舍信息视图：{宿舍区，宿舍楼号，宿舍号，门禁时间，宿舍电话，入住时间}

此视图提供给学生查看自己所在的宿舍楼、寝室的门禁时间、宿舍电话和入住时间信息。

1. 宿舍财产信息视图：{宿舍号，财产号，财产名，财产数量}

此视图提供给学生和教师查看某宿舍的财产信息。

1. 学生离校信息视图：{学生号，学生名，宿舍号，离校时间，返校时间}

此视图提供给教师管理查看学生的离校情况。

1. 报修信息管理视图：{财产号，财产名，财产价格，财产库存，宿舍号，报修时间，报修原因，解决时间}

此视图提供给教师管理查看报修的相关信息。

1. 学生快递信息视图：{快递号，学号，姓名，宿舍号，到达时间，接收时间}

此视图提供给教师管理查看学生的快递的相关信息。

## 物理结构设计

依赖于SQL Server数据库管理系统，我们不必自己设计。

## 数据库实施

### 应用程序数据的载入

1. 满足数据库的基本要求：实体完整性，参照完整性，用户定义的完整性。满足数据项的一些基本含义要求。
2. 还需要注意的是数据的逻辑规范限制，如：我们的宿舍楼表里的所在区限制在“北海苑”，“东海苑”，“南海苑”；用户的身份限制成老师和学生；性别限制在“男”和“女”；一个宿舍的人数不能超过该宿舍的容纳量；宿舍已有人数要和已住进该宿舍学生的数量相匹配；宿舍号和楼号要相匹配（宿舍号里面的信息含有楼号）；楼号和所在的区域名称相匹配（每个苑的楼号是固定的如：北海苑只有1、2、3号楼）。

### 程序的编码与调试

1. 编码

总体概览：



图 4 项目的所有包

具体介绍：

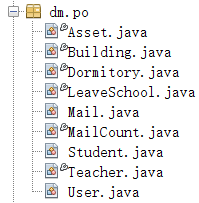


图 5 po包

PO:persistant object 持久对象,是与数据库中的表相映射的java对象。最简单的PO就是对应数据库中某个表中的一条记录，多个记录可以用PO的集合。PO中不包含任何对数据库的操作。（对每一个实体表封装成一个类，包含它们的setter和getter方法和构造函数）



图 6 vo包

VO:value object值对象。通常用于业务层之间的数据传递，和PO一样与数据库里的联系表对应，是抽象出的业务对象,我们为了在数据库设计满足范式的需求，还建立了视图，所以这里面的类不仅和联系表相对应还与视图对应。

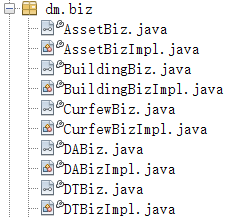


图 7 biz包

BIZ:包含业务层的的接口和接口方法的实现，对数据增、删、改、查的语句都包含在这里面。

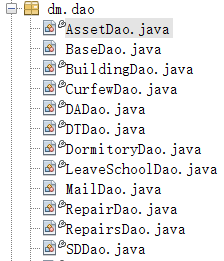


图 8 dao包

DAO:data access object 数据访问对象，此对象用于链接数据库，与PO、VO、BIZ结合使用对数据库中数据更新、删除、查询操作方法抽象的实现。

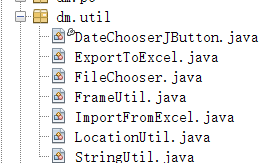


图 9 util包

UTIL:包含一些格式控制，时间按钮，窗体格式控制，导入/导出Excel表格等一些函数的封装。

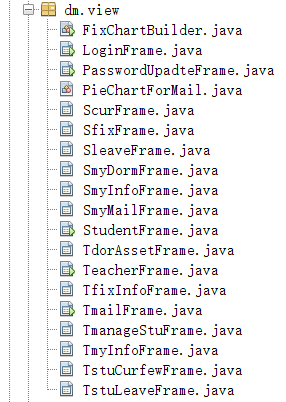


图 10 view包

VIEW:具体的窗体界面设计及实现。

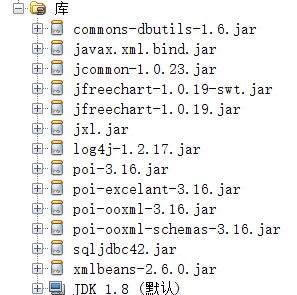


图 11 库

库：包含需要导入的jar包。

1. 调试

在调试期间我们遇到了语法错误、逻辑错误、SQL语句错误等问题，调试的方法采用断点调试，或者用简单的输出语句来判断程序出错的地方，从而完成对程序的正确性。

## 数据库运行与维护

### 运行

先采用少量的数据对程序进行运行，观察数据库的修改和程序执行想要达到的效果是否一致，若不一致则先对程序进行调试，可以多设置一些提示框，这样可以快速找到出错的地方，若无问题再使用SQL语句在数据库新建查询看是否能运行成功，若成功再与程序中的SQL语句做对比，直至找出错误为止。

### 维护

我们数据库的数据录入都是由SQL语句进行操作，可利用完整备份和差异备份对数据库进行还原操作，防止由于不当的操作对数据库中数据进行的破坏。

# 系统运行结果

## 登录界面



图 12 登录界面

**修改密码：**进入修改密码界面，见后图。

**身份：**根据不同的身份，进入不同的界面，获得不同的权限。



图 13 密码修改界面

**验证：**首先根据输入的用户名和密码以及选中的身份进行验证，验证成功后，才可以点击确认修改按钮。

## 学生界面

### 学生主界面



图 14 学生界面功能

点击注销可返回到登录界面。

### 个人中心

此模块学生用户只能查看自己的个人信息，不能做出修改。



图 15 个人中心界面

### 我的宿舍

此模块学生可以看到自己宿舍的详细情况，包括宿舍电话、宿舍财产等。



图 16 我的宿舍界面

### 我的快递

此模块学生用户只能查看自己的快递信息。

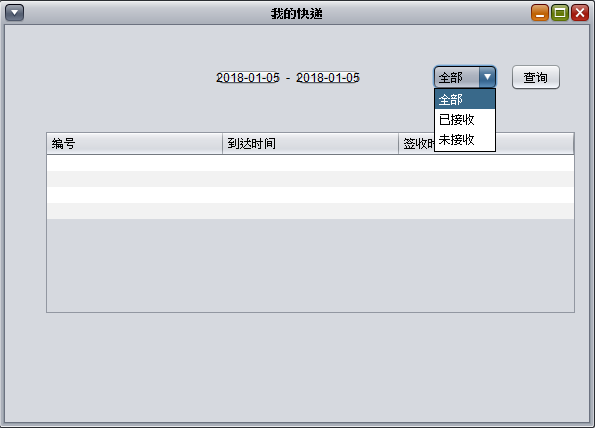


图 17 我的快递界面

**查询：**提供不同类型的查询，可以让同学查询全部、已签收的和未签收的快递信息。

### 报修申请

此模块提供查询和报修申请功能。

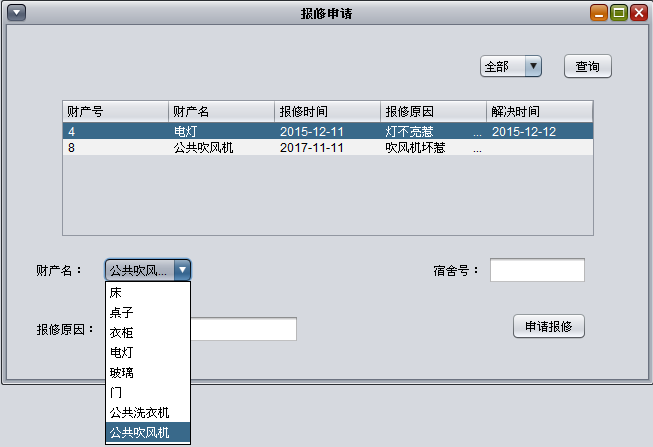


图 18 报修申请界面

**查询：**提供已解决和未解决两种分类的查询。

**申请报修：**学生用户可根据不同的财产损坏情况进行报修处理。

### 晚归申请

此模块提供查询和申请晚归的功能。



图 19 晚归申请界面

**查询：**提供已审批和未审批两种分类查询。

**申请：**需要学生输入晚归时间和晚归原因。输入的格式要求是，例如“2017-01-01”，“2017/01/01”。

### 离校信息

此模块提供查询和数据导入导出的功能。

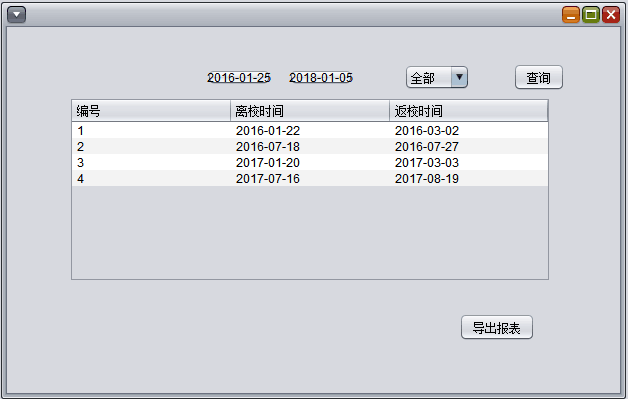


图 20 离校信息界面

**查询**：提供按时间查询。

**导出报表：**可将查询结果导出为.xls文件，结果如下图。

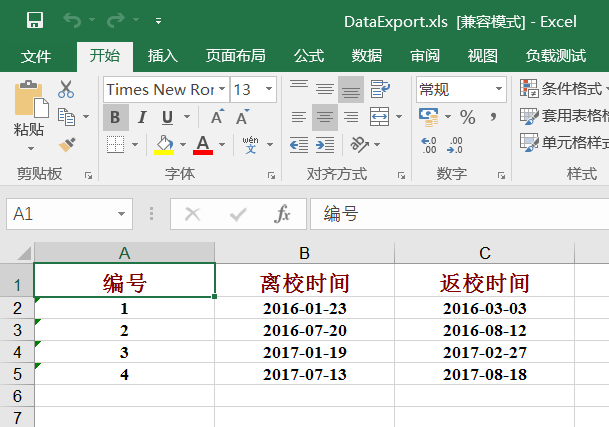


图 21 离校信息导出Excel表截图

## 教师界面

### 教师主界面

点击注销可返回到登录界面。

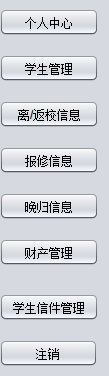


图 22 老师界面功能

### 个人中心

用于查看教师相关的基本信息。



图 23 个人中心界面

### 学生管理

完成对学生信息的增、删、改、查功能操作以及批量数据的导入/导出Excel表格。



图 24 学生管理界面

**学生查询：**可根据学生的每一个属性列进行前方一致查询，还可进行模糊查询（包含所有的属性列），当然在根据按照入住时间查询时，我们是将时间选择封装成一个按钮，当然结束时间一定要比开始时间大否则会报错。时间选择按钮在如下图：

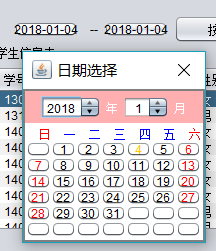


图 25 日期选择

**宿舍查询：**提供按宿舍号、宿舍楼、宿舍空位的下限进行前方一致查询。我们在更改或增加学生的宿舍时是在第二个可分配的宿舍表中选择，因此宿舍查询出来的结果都是满足在宿舍空位>=1的前提下。下拉框如下图：

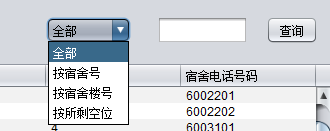


图 26 宿舍查询下拉框

**载入：**将已有的学生信息全部导入。

**增加：**增加一个学生需要输入他的学号和姓名，性别、年级、院系采用下拉框的形式，宿舍号是从表2中可供分配的宿舍进行选择。本着规范的原则，我们让用户在使用的时候尽可能减少手动输入的情况。

**删除：**需要先选择表1已有的学生，删除按钮才可用。删除我们采用的不是软删除相当于级联删除，若一个学生的信息被删除，他相应的离校、晚归、快递信息都会被删除。（会有确认删除的提示框）

**修改：**需要先选择表1已有的学生，修改按钮才可用。修改学生信息时需先从表1选择一个学生，学号是不能修改的，若学号输入错误只能将这个学生删除在重新增加，宿舍号的修改也需要在表2中选择和增加类似。

**重置：**将所有的可输入窗口清空。

**退出：**返回教师初始界面。

**批量导入：**先选择符合格式和要求的Excel文件，再点击打开

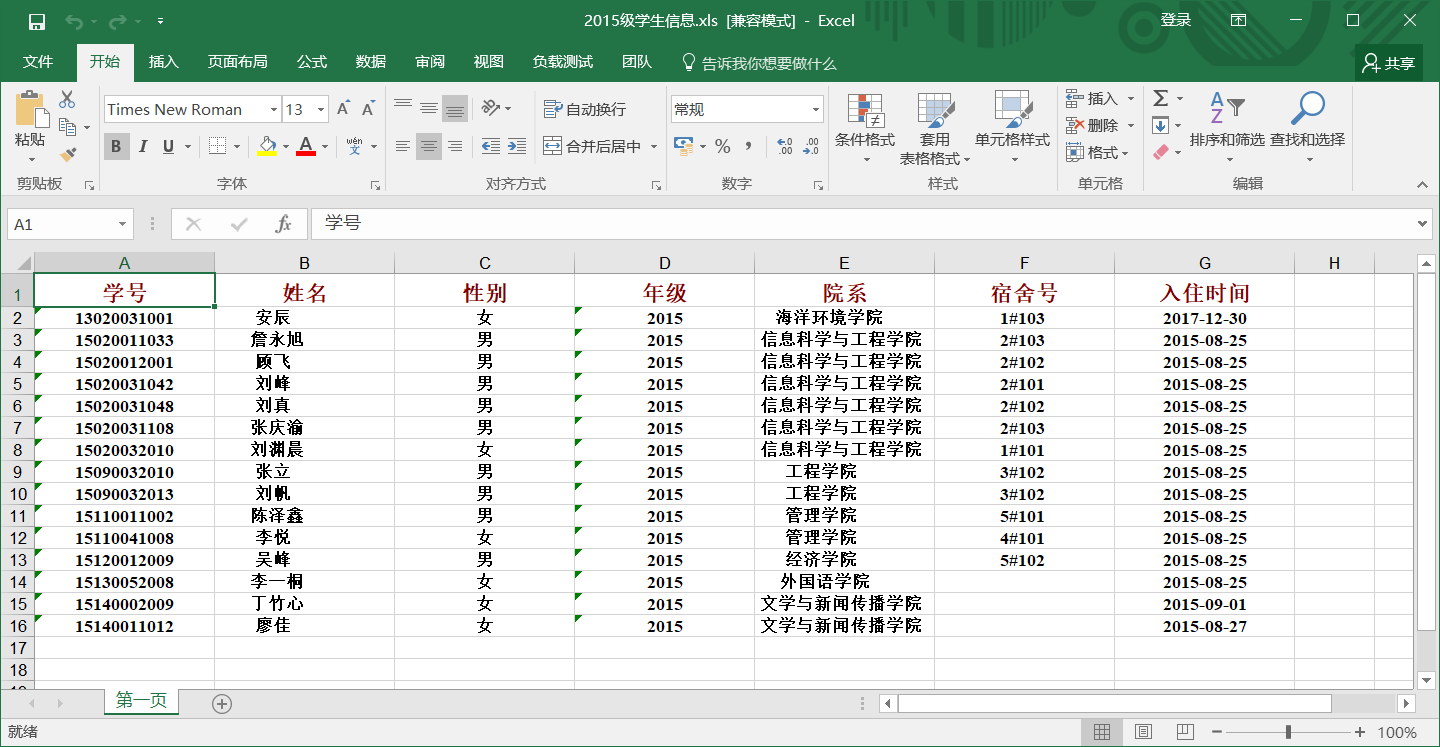


图 27 被导入的Excel表

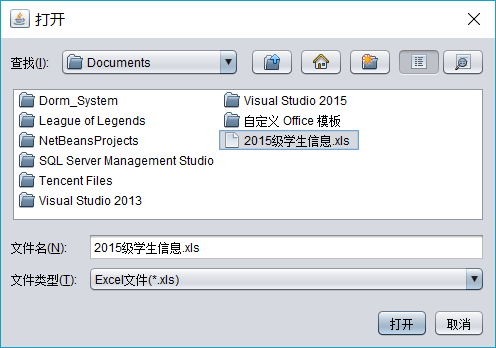


图 28 Excel文件选择

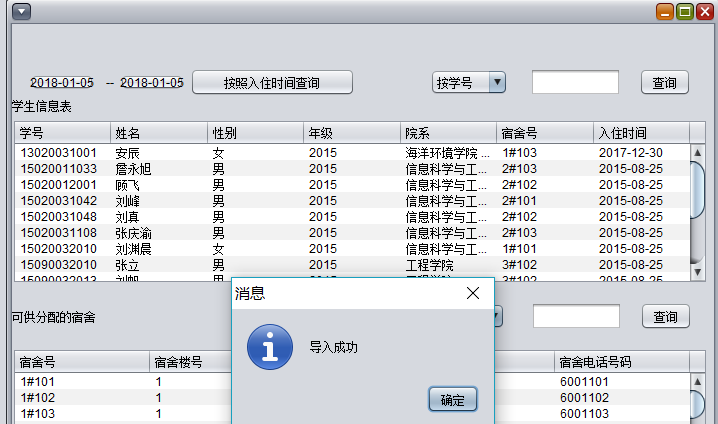


图 29 导入Excel表到学生信息表中

导入成功数据显示在第一个表中。

**导出报表**：先选择路径并编辑文件名

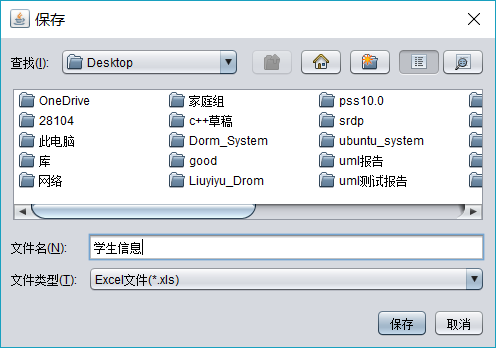


图 30 导出Excel表文件操作

点击保存：

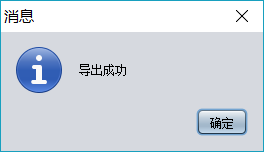


图 31 导出成功

打开Excel表格：

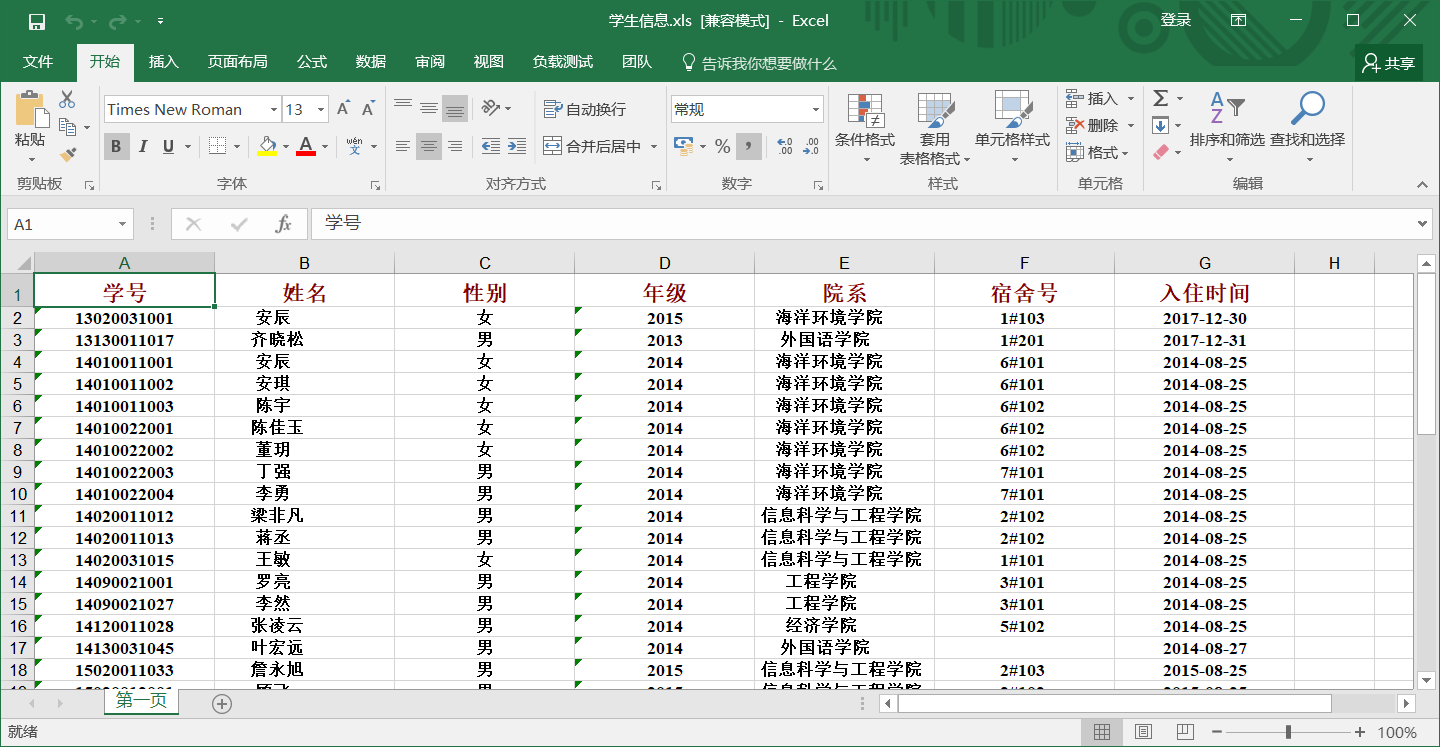


图 32 导出的Excel表

### 学生离/返校信息

对学生离校返校信息进行管理操作。



图 33 学生离/返校信息处理界面

**查询**：可全部导入，可按照学号、已返校、未返校、姓名、宿舍号进行前方一致查询学生离/返校信息。

**载入**：同查询条件未全部时一样。

**信息查询**：当输入学号或点击表中随意一个学生时姓名和宿舍号会自动导入。

**离校写入**：在信息查询后离校写入按钮会自动激活，时间会自动获取当前时间，再点击了离校写入的按钮后，还会有一个确认的消息框，点击“是”后数据库中该学生则会多出一条离校的信息。

**返校确认**：只有点击了那些离校未归的学生返校确认按钮才会被激活，返校时间也是自动获取当前时间，点击返校确认按钮还会有一条确认的消息框，点击“是”后返校信息也就写入数据库。

**删除**：删除表1中的一条记录，会有提示框进行删除确认。

**退出**：返回教师初始界面。

### 报修信息

对学生申请的报修信息进行管理操作。



图 34 报修信息管理界面

**查询**：可以按照财产号、财产名、宿舍号进行前方一致查询，还可以对表中所有的信息进行模糊查询。

**全部载入**：载入所有学生提交给教师的报修信息。

**生成图表**：对表中的报修财产进行汇总如下图所示：

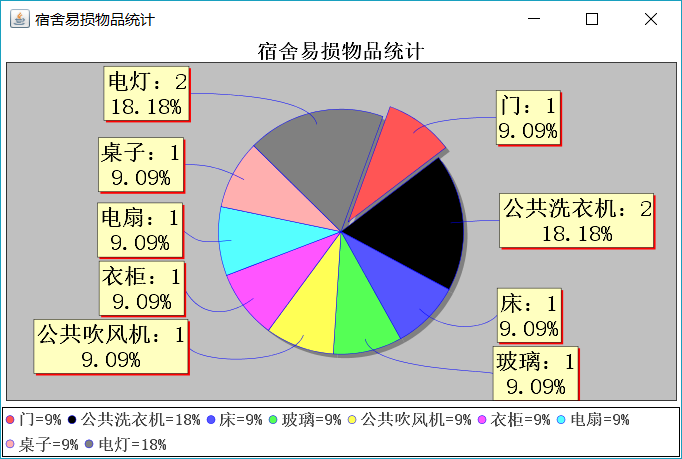


图 35 统计财产报修次数的饼状图

**已解决**：载入学生提交给老师的并且已经解决的报修信息。

**未解决**：载入学生提交给老师的并且还没解决的报修信息。

**修改/保存**：对还没解决的报修信息若解决了点击此按钮，则获取当前时间填入至解决时间。还可对已经解决的报修信息进行时间的修改。

**重置**：将所有可输入窗口清空。

**退出**：返回教师初始界面。

### 晚归信息

对学生晚归信息进行批准管理操作。



图 36 晚归信息管理界面

**查询**：可按照学号、姓名、宿舍号、教师号进行前方一致查询，还可对表中所有信息进行模糊查询。

**全部载入**：载入所有学生申请的晚归信息。

**已批准**：载入学生申请的且已经老师被批准的晚归信息。

**未批准**：载入学生申请的且还未被老师批准的晚归信息。

**批准**：只能对对未被批准的晚归信息进行处理。若已批准过的晚归信息点击批准按钮会予以“已批准过”的提示框。

**退出**：返回教师初始界面。

### 财产管理

仅可对学生宿舍的财产信息进行查看。

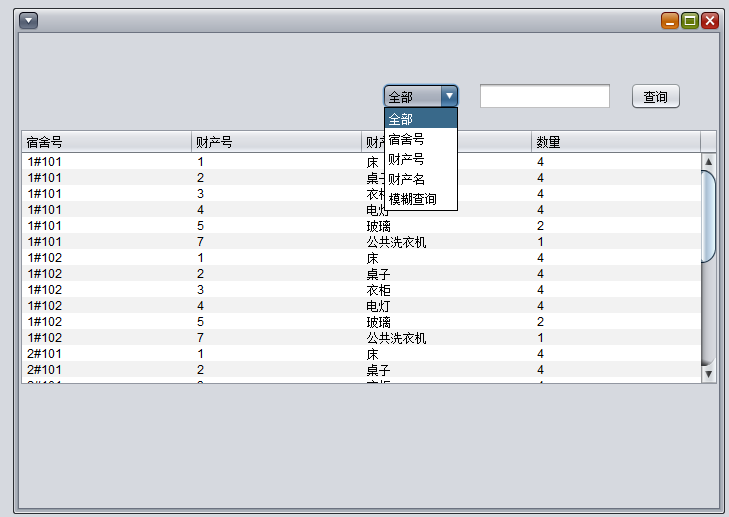


图 37 查看学生宿舍财产界面

**查询**：可全部查询，可对宿舍号、财产号、财产名进行前方一致查询，以及对表中信息进行模糊查询。

### 学生快递管理

对学生的快递进行签收确认。

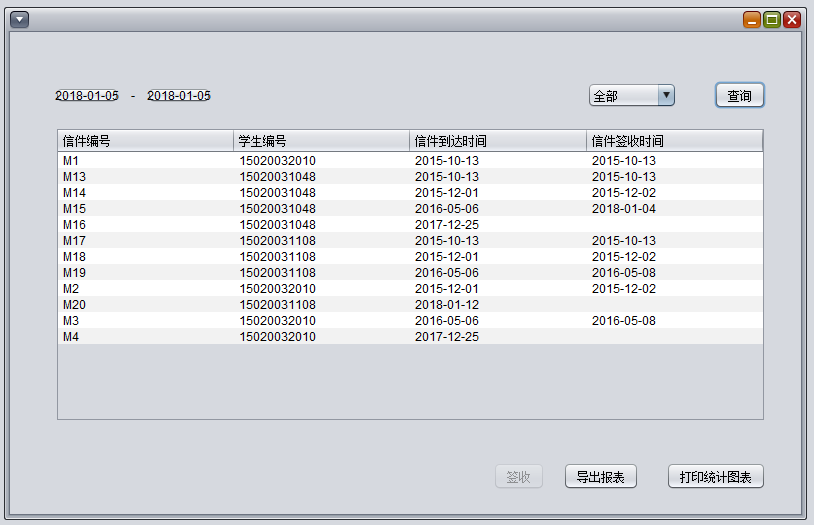


图 38 学生快递管理界面

**查询**：可按全部查询，可按到达时间和签收时间进行前方一致查询（时间按钮同学生管理界面的时间按钮一样）。

**签收**：仅可对未被签收的信件才可以点击签收按钮，签收时间为记录当前日期。

**导出报表**：选择文件夹输入文件名。

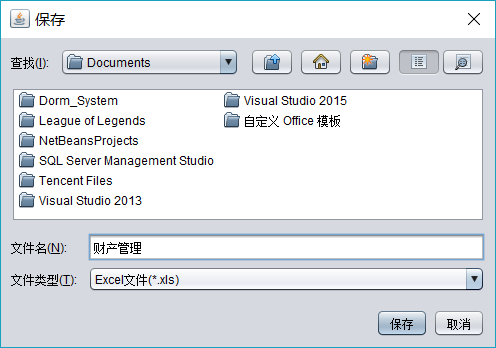


图 39 导出Excel表文件操作

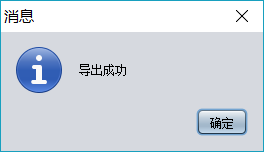


图 40 导出成功



图 41 导出的Excel表

**打印统计图表**：导出每个学生所收到的快递数量占比。

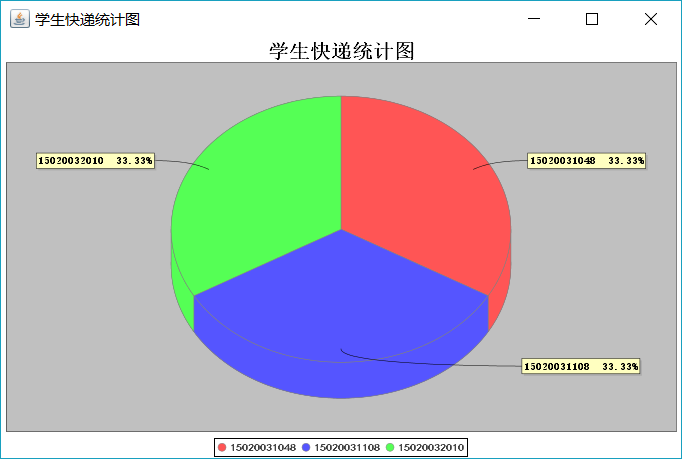


图 42 统计学生快递数量的饼状图

# 实验遇到的问题与解决方法

1. 我们对学生及宿舍的关系没有搞清楚，学生和宿舍是一个1对n的关系，即学生确定后其宿舍也是确定的，但是在最初设计时将学号和宿舍号单独提取了出来，弄了一个学生宿舍关系，其实是没必要的，因为既然学号就可以确定宿舍号直接将宿舍号作为学生的一个属性即可。
2. 当初在设计表的时候我们是朝着尽量满足范式，减少数据冗余，这样就会导致当反应在程序设计里面的时候查看一个信息只能看到他的学号却不知道他的姓名，这样就会信息缺失的情况，于是我们后面才加上的视图，这样就可以做到在数据库设计中满足范式需求又可以在程序设计里可以看到更多的信息。
3. 在宿舍这个关系表中，最初只有简单的宿舍号、楼号、宿舍电话几个属性列，由于没有容纳量的限制，一个宿舍可以不停的添加学生，显然这样是不合理的，于是我们又添加了宿舍容纳量和宿舍空位这样的属性列。
4. 在进行连接数据库操作时，对jdbc的不熟悉造成的设计困难。我们通过参照过往项目经验进行改写，使之适用于本次实验的设计。
5. 起初由于小组成员的编码格式未统一为UTF-8，造成后续开发过程中程序出现乱码等异常。后我们选取了小组成员当中正确的版本，统一了编码格式后，再继续在正确版本的基础上进行开发。
6. 导入外部jar包有几种不同的方式，我们起初使用的是最基本的方式，外部jar包独立于项目文件夹，每次都要重新配置Java Build Path，不够方便。之后采用了文件夹导入式，所需要的jar包放在项目文件夹中，在不同电脑上打开后，选中需要添加到项目中的jar包，右键选择"Add to Build Path"，这样jar包就成功导入到项目中了，十分快捷。
7. 信息系统的登录功能需要获取用户名和密码，如果用户名有效，我们就利用字符串匹配原理将获取到的密码与该用户在数据库中的密码进行比对。起初我们在测试该功能时发现总是因为密码不正确导致登录失败。一开始我以为是数据库和IDE的编码格式不匹配的问题，查阅各种资料才找到真正的症结所在。我们在数据库中规定用户密码格式为char，长度为20。那么我们使用JDBC获取到的密码就是一个长度为20的字符串。尽管真实密码长度可能仅有几位，但添加至数据库时会自动在后面补上空格。但用户在前端输入时并不知道数据库设计规则，只会输入真实长度的密码。将长度不一致的两个字符串比对，当然会匹配失败。知道了原理后，解决的方案就很简单了，只要用之前写好的字符串工具类（UTIL包）对密码字符串加上trim()操作，去除所有空格即可。
8. 对于删除这个操作，我们刚开始由于表的设计是没有级联删除的，于是在学生删除这个操作上我们得先考虑与学生相关的一些信息，如夜归信息，离校返校信息，学生快递信息，如果含有我们应该将这些信息先删除再删除这个学生。当然这些删除都是基于硬删除。
9. 在实际开发过程中，因为小组每个成员都需要对项目进行编程，采取互相拷贝的方式效率过低，所以我们使用了GitHub对项目进行托管，每个小组成员加入到项目中去，可随时上传修改以及下载更新。为了操作方便，我们选用了GitHub桌面版。但由于组员在不同步的情况下对程序源代码的修改，造成了文件内容混乱问题。我们通过以前的备份文件，排查出现冲突的原因，将出现冲突的文字段处理完毕。
10. 在项目开发过程中，我们一边学习GitHub版本控制原理一边开发，由于对其更新操作理解不透彻，我们遇到了小组成员上传自己代码时误将其它成员的文件删除的情况。尽管我们做好了多处备份的工作，但是由于时间紧张，还是有部分遗漏，短时间内不能恢复项目所有内容。在紧张地寻找解决方案的时候，我们了解到到GitHub有强制版本回退的操作，首先获取要回退到的版本的SHA1值，然后打开本地git bash将当前项目的head修改为要回退的版本，然后再重新提交，这种操作并不是撤销了之前的版本，而是在最后一个版本的基础上引出的。

# 实验心得与体会

1. 需求分析极其重要，需求分析结果是否准确反映用户的实际要求将直接影响到后面各阶段的设计，并影响到设计结果是否合理和实用。我们最初因为需求分析不够准确，后面的E-R图也因此做了多次修改，甚至在开发过程中，对一些界面需要实现的功能也进行了多次修改，足以证明需求分析是整个设计过程的基础，也是最困难、最耗时的一步。
2. 因为是小组完成项目，所以分工合作很重要，避免重复操作，提升效率。
3. 因为项目在不断进行更改，所以做好备份十分重要，否则很有可能因为有编码问题等比较严重的错误导致项目瘫痪。并且我们要定时做好最新的正确的项目备份。要善于运用我们课上所学的知识，例如：在数据库系统的实验课上，我们学习的对数据库进行备份与还原就派上了用场。
4. 在对某些功能的实现上运用当前所了解知识不足以实现时，我们应及时查阅相关资料，以最短时间内完成该方面的学习。
5. 当多人合作开发项目时，我们要学会使用版本控制工具，使开发更加方便高效。