

成绩：_____

南京信息工程大学

Java 程序设计 课程设计

题 目： 学生信息管理系统
及车站信息管理系统的设计

学 院： 计算机与软件学院

专 业： 物联网工程

学生姓名： 张逸飞

学 号： 20171375029

指导教师： 陈金辉

年 月 日

目 录

1. 前言	3
2. 需求分析	2
3. 概要设计	3
4. 详细设计	5
5. 系统实现（包括运行界面）	11
6. 测试	153
7. 总结	153

1. 前言

为了充分应用 Java 程序设计语言和数据库系统的相关知识，将其应用于项目实践中，因而产生了两个项目：车站票据管理系统及学生成绩管理系统。

对于车票信息管理系统，联系生活实践，应具有如下功能：首先，应该能够查询车站所有车票或指定车票信息的查询功能；其次，应该具有提供给旅客买票的功能，即购票功能；在此基础上，为了更具人性化，应具备退票功能。具体实现可以使用数据库维护两张表，一是乘客购票信息表，二是车站车票信息表。

对于学生成绩管理系统，应具有以下功能：学生成绩查询功能，包括特定学生成绩查询；学生成绩排序功能；求各科成绩平均值的功能；学生选课情况查询功能；增删学生信息的功能；修改学生信息的功能等。使用数据库维护两张表，一是学生成绩表，二是学生选课情况表。

环境：系统环境为 Linux，Java 环境为 JDK 1.8，编程环境为 Eclipse。

前端实现可以使用 Java Swing 提供的 GUI 编程功能，主要涉及 JButton、JLabel、JPanel、JTable、JText、JComboBox、JMenu、JMenuBar 等类。

数据库使用 MySQL。主要在 Bash 中执行命令，辅助以可视化数据库管理软件 MySQL Work Bench。

2. 需求分析

车票信息管理系统：

车票信息管理系统需要解决的问题如下：方便用户选购所需的车票。联系生活实践的经验，我们可以对该问题做出更加详尽的分析。用户所需要的购票体验应该是清楚、快捷、方便。要做到这些要求，首先应当分析用户的查询需求，提供相关查询功能，使用户能更快的找到需要购买的车票。然后车票的相关信息应该尽可能翔实，但又不能过多而淹没了一些关键的信息，比如出发时间等。因此显示的车票信息应该尽量满足所有人的需求的同时做到足够清楚。在用户查询到所需车票时，购买车票应该足够方便快捷。

由上述需求分析可以得知该项目应该具有的功能：按照出发地和目的地查找的区间查询功能；按车次查找的查询功能。车票应该包含下列属性：车次、始发站、终点站、发车时间、预计到达时间、票价和剩余票数。乘客购票信息表应包含以下属性：车次、始发站、终点站、发车时间、预计到达时间和票价。

由此可以看出整个系统涉及的两个对象，一是车站信息，二是乘客信息。两者除了剩余车票属性外，其它几乎一致。对于车站信息，访问的人需要关心是否有车票剩余，而对于已经买到的车票，剩余票数的信息是多余的。两者的关联在于用户的买票行为。

学生信息管理系统：

学生信息管理系统需要解决的问题如下：方便管理者进行数据的录入、修改、删除、查找。录入的信息应该包括学生的学号、各科成绩、班级信息等，修改应该能够修改学生的全部属性；删除特定学生的所有信息；查找要能够按指定条件查找。查找的实现比较关键，很容易可以想到，删除和修改的操作都是在查找到学生的前提下进行修改。

由上述需求分析可以得知该项目应该具有的功能：按照学号、班级、姓名查找的查询功能，删除信息功能，修改学生信息的功

能，增添学生信息的功能。学生信息应该包含以下属性：学号、班级、姓名、各科成绩、总分等。除此以外应该实现按各科排序，求平均值等常用功能。

3. 概要设计

关于整个系统框架的初步设想：我可以把整个系统分为三个部分，一是内核（Kernel）部分，负责控制与数据库管理系统的交互，对外提供一些接口。二是图形用户界面（GUI）部分，负责控制界面的样式，提供人机交互的控件。三是主函数（Main）调用部分，负责运行整个系统。

车票信息管理系统：

内核部分（Kernel）：

提供的接口：

- getDatabase() 获取所有数据库的名称
- getTable() 获取所有表的名称
- ConnectToMysql() 连接到数据库
- ChooseDatabase() 选择数据库
- GetAllColumnValue() 获取所有列的值
- GetAllColumnName() 获取所有列的列名
- GetAllColumnNameAndType() 获取列名和属性
- GetColumnNum() 获取列的数量
- Search() 查找
- Delete() 删除
- SelectAll() 选择所有符合条件的值
- Decrease() 某行值减一
- Increase() 某行值加一
- GetAllStartPlaceName() 获取所有始发地名称
- GetAllEndPlaceName() 获取所有终点名称
- GetAllUserTableColumnName() 获取用户表列名

GetAllUserTableValue() 获取用户表值
InsertUserdata() 插入用户数据
DeleteUserdata() 删除用户数据

图形用户界面（GUI）部分：

ErrorPopup 类：主要用于弹出错误信息
Login 类：登陆界面
OperateScreen 类：操作主界面
RightOrderMenu 类：右键订票菜单
RightRefundMenu 类：右键退票菜单
SearchButton 类：查找界面
SelectWindow 类：选择数据库和表单
ShowButton 类：弹出查找到的表单信息
Test 类：用于测试
Universe 类：通用类
UserButton 类：弹出用户购票表单

主函数调用（Main）部分：

Main() 使用登陆界面类创建对象

:

学生信息管理系统：

内核（Kernel）部分：

提供的接口：

getDatabase() 获取所有数据库的名称
getTable() 获取所有表的名称
ConnectToMysql() 连接到数据库
ChooseDatabase() 选择数据库
Insert() 插入学生数据
GetAllColumnValue() 获取所有列的值
GetAllColumnName() 获取所有列的列名
GetAllColumnNameAndType() 获取列名和属性
GetColumnNum() 获取列的数量

Delete() 删除学生数据
SelectAll() 选择所有列
Average() 计算当前各科平均值
Search() 查找学生数据
ASCsort() 按总分升序排序
DESCsort() 按总分降序排序

图形用户界面（GUI）部分：

DeleteButton 类：删除按钮事件
ErrorPopup 类： 错误弹出界面
InsertButton 类：插入按钮事件
Login 类：登陆界面
OperateScreen 类：操作界面
SearchButton 类：查找按钮事件
SelectWindow 类：选择按钮事件
ShowButton 类：展示按钮事件
Test 类：用于测试
Universe 类：通用类

主函数调用（Main）部分：

Main() 使用登陆界面类创建对象

4. 详细设计

关于各部分的设计思路：

内核：

根据 kernel 的作用：控制和数据库管理系统的交互，很容易

可以想到，kernel 应该具有控制连接、传递命令、查看表单内容等方法。由于考虑到代码的复用和封装，还可以再将其细分为：生成命令的方法、创建新会话的方法（保证各个命令的执行结果相互独立）、传递命令的方法、获取表单信息的方法（这里另外构建了一个 showTable 类封装了获取表单信息方法的代码并给 kernel 提供接口）等。根据项目的不同，生成命令的方法和传递命令的方法有些许区别。如，对于车票信息管理系统需要的区间查询功能，应该有特殊的 search() 方法；而对于学生信息管理系统，需要具有执行排序命令的方法。

图形用户界面：

Login 类：

根据数据库管理系统的权限特点，应该设计一个登陆界面验证用户的身份信息，并且输入的账号、密码等信息应传递给数据库管理系统验证并授权。Login 是初始的界面，在登陆成功后应该关闭当前界面，进入下个界面以推进事务的进行。

OperateScreen 类：

该类是操作主界面，应该具有该项目的主要功能控件。这些主要控件需要自行设计。

DeleteButton 类：

单击删除按钮应该弹出一个新的删除界面用于数据的删除。在删除按钮类设计的过程中，需要额外设计一个删除界面，并设置鼠标单击的事件响应，打开删除界面。

SearchButton 类：

单击查找按钮应该出现查找界面。在车票信息管理系统的查找界面中可以提供两种选择，一是按车次查找，二是按地点区间查找。对于车次查找只要有文本框输入车次，对于按地点区间查找可以采用下拉列表的形式，在选出出发地后，自动筛选出可以到达的目的地供用户选取。对于学生信息管理系统的查找界面提供任意信息查找的功能，具体实现方式是提供行表单，每列分别对应信息的属性，对输入的每个条件进行并运算，然后查找。

InsertButton 类：

插入按钮事件应该弹出插入界面。插入界面提供一张空白表格供用户输入信息，单击确认按钮即可插入所输信息。

车票信息管理系统和学生信息管理系统的共用代码：

内核部分：

连接到数据库的方法：

```
public static boolean ConnectToMysql(String username,String password){
    userName = username;
    passWord = password;
    try {
        //装载jdbc的mysql驱动
        Class.forName("com.mysql.jdbc.Driver");
        //根据用户名和密码创建连接
        conn =
        DriverManager.getConnection(LocalHostURL,userName,password);
        stat = conn.createStatement();
        return true;
    }catch(SQLException se) {
        //异常处理
        new ErrorPopup("连接失败");
        return false;
    }catch(ClassNotFoundException ce) {
        new ErrorPopup("载入驱动失败");
        return false;
    }
}
```

返回命令结果的方法：（这里使用特定的 statement 对象来传递命令）

```
private static ResultSet newStatementReturnRes(String order,String
                                                FunctionName,Statement newstate) {
    ResultSet newResultSet;
    try {
        newResultSet = newstate.executeQuery(order);
        return newResultSet;
    }catch(SQLException se) {
        new ErrorPopup("查询失败");
        return null;
    }
}
```

获取所有列名的方法

这里的 showTable 是 Kernel 包中的一个类，封装了关于获取表单信息方法的实现

```
public static String[] GetAllColumnName() {
    return showTable.GetColumnName(res);
}
```

获取表单的所有值的方法

```
public static String[][] GetAllColumnValue(){  
    return showTable.GetColumnValue(res);  
}
```

获取列名和列的属性值的方法

```
public static String[] GetAllColumnNameAndType() {  
    return showTable.GetColumnNameAndType(res);  
}
```

获取列的数量的方法

```
public static int GetColumnNum() {  
    return showTable.GetColumnNum(res);  
}
```

学生信息管理系统的部分代码：

查找满足 condition 条件的所有行，并以二维数组的形式返回

```
public static String[][] Search(String tableName, String[][] condition) {  
    //FunctionReturn类是用于生成有返回结果的MySQL命令  
    String order = FunctionReturn.Search(tableName,condition);  
    Statement newstat = createNewStatement();  
    //newStatementReturnRes是Kernel的私有方法，用于执行有返回结果的MySQL命令  
    ResultSet newrs = newStatementReturnRes(order,"Search", newstat);  
    String[][] res = showTable.GetColumnValue(newrs);  
    return res;  
}
```

删除方法，可以按照学生的学号删除某特定学生的信息

```
public static void Delete(String TableName,String ID) {  
    //FunctionReturn类是用于生成有返回结果的MySQL命令  
    String order = FunctionVoid.Delete(TableName, ID);  
    Statement newstat = createNewStatement();  
    //newStatementVoid()是kernel类的私有方法，用于执行没有返回结果的sql命令  
    newStatementVoid(order, "Delete",newstat);  
    SelectAll(curTable);  
}
```

插入方法，插入一行学生的信息

```
public static void Insert(String tablename,String[] value) {  
    String[] columnName = GetAllColumnName();  
    String order = FunctionVoid.Insert(tablename,columnName,value);  
    Statement newstat = createNewStatement();  
    newStatementVoid(order,"Insert", newstat);  
}
```

```

        SelectAll(curTable);
    }
    选择所有的行
    public static void SelectAll(String TableName) {
        //FunctionReturn类封装了所有有返回值的命令的生成方法
        String order = FunctionReturn.SelectAll(TableName);
        curTable = TableName;
        executeReturn(order, "SelectAll");
    }

```

车票信息管理系统的部分代码：

获取表单中所有的始发地信息

```

    public static String[] GetAllStartPlaceName() {
        Set<String> recSet = new HashSet<>();
        String[][] allvalues = GetAllColumnValue();
        int row = allvalues.length;
        for(int i = 0; i < row; i++) {
            recSet.add(allvalues[i][2]);
        }
        String[] ret = new String[recSet.size()];
        Iterator<String> it = recSet.iterator();
        int cnt = 0;
        while(it.hasNext()) {
            ret[cnt++] = it.next();
        }
        return ret;
    }

```

获取所有终点信息类似

获取特定地点区间的所有车次信息

```

    public static String[][] GetAllAccordingToStAndEd(String st,String ed){
        String order = "SELECT * FROM " + curTable + " WHERE 始发站='"+st+"'
        AND 终点站='"+ed+"'";
        Statement newstat = createNewStatement();
        ResultSet newres = newStatementReturnRes(order,
        "GetAllAccordingToStAndEd", newstat);
        String[][] ret = showTable.GetColumnValue(newres);
        return ret;
    }

```

在用户买票后插入到用户购票表单中，并且剩余票数减一

```

    public static void InsertUserdata(String[] val) {

```

```

String[] colname = GetAllUserTableColumnName();
String tablename = userTable;
String order = FunctionVoid.InsertUserdata(tablename, colname,
val);

Statement newstat = createNewStatement();
newStatementVoid(order, "InsertUserdata", newstat);
}

```

用户退票后从购票表单中删除票信息，并且剩余票数加一，代码和插入类似

数据库设计：

车站信息管理系统：

车票表单：

Field	Type	Null	Key	Default	Extra
车次	varchar(20)	NO	PRI	NULL	
类型	enum('汽车','动车','高铁')	NO		NULL	
始发站	varchar(20)	YES		NULL	
终点站	varchar(20)	YES		NULL	
发车时间	time	YES		NULL	
预计抵达时间	time	YES		NULL	
票价	int(11)	YES		NULL	
剩余票	int(11)	YES		NULL	

用户数据表单：

Field	Type	Null	Key	Default	Extra
车次	varchar(20)	NO	PRI	NULL	
类型	enum('汽车','动车','高铁')	NO		NULL	
始发站	varchar(20)	YES		NULL	
终点站	varchar(20)	YES		NULL	
发车时间	time	YES		NULL	
预计抵达时间	time	YES		NULL	
票价	int(11)	YES		NULL	

学生信息管理系统：

Field	Type	Null	Key	Default	Extra
ID	varchar(10)	NO	PRI	NULL	
StudentName	varchar(20)	NO		NULL	
Class	int(11)	YES		NULL	
Chinese	int(11)	YES		NULL	
Maths	int(11)	YES		NULL	
English	int(11)	YES		NULL	
Summary	int(11)	YES		NULL	

系统设计的特点：

整个系统采用内核功能和用户图形界面单独设计的方式，使内核具有较为良好的通用性。因此用于学生信息管理系统的内核只要另外实现较少的一些函数接口，就可以移植到车票信息管理系统中，提高了代码的复用性。另外此设计封装了与数据库管理系统交互的方法，对于功能实现的修改，在不改变接口的情况下，只需要在内核的方法中修改特定方法，就可以在整个系统中生效。

对于用户图形界面的设计，采用单独设计各个功能类，然后集中到一个操作界面类上。这样设计的目的是减少各个类之间的牵连，因为不同的功能类之间的属性方法相互调用很少甚至几乎没有，这样就提高了各个类的封装性，有利于数据的保护。

5. 系统实现（包括系统运行界面等）

车站信息管理系统：

功能选择

查询

用户数据

查找

退出

车次	类型	始发站	终点站	发车时间	预计到达时间	票价	剩余票
01100	动车	广州	深圳	18:55:00	19:55:00	142	25
01125	动车	广州	深圳	21:25:00	22:25:00	453	20
01419	动车	惠州	深圳	07:22:00	11:22:00	180	47
01442	动车	惠州	深圳	16:38:00	20:38:00	453	37
01744	动车	广州	深圳	18:58:00	22:58:00	332	48
01761	动车	深圳	惠州	15:02:00	19:02:00	177	38
01789	动车	惠州	深圳	19:51:00	23:51:00	352	35
02214	动车	惠州	深圳	22:51:00	22:51:00	144	19
02289	动车	深圳	广州	22:02:00	00:02:00	376	14
03063	动车	广州	深圳	08:10:00	14:10:00	598	27
03286	动车	广州	深圳	15:55:00	21:55:00	299	24
03419	动车	深圳	惠州	16:56:00	22:56:00	429	7
03486	动车	惠州	深圳	06:29:00	07:29:00	12	36
03545	动车	深圳	惠州	18:49:00	19:49:00	327	9
03728	动车	广州	深圳	13:42:00	19:42:00	311	23
03889	动车	深圳	广州	16:51:00	17:51:00	162	35
03330	动车	广州	深圳	00:34:00	16:34:00	388	38
03556	动车	惠州	深圳	11:39:00	12:39:00	187	36
03708	动车	深圳	惠州	13:54:00	22:54:00	352	5
03789	动车	深圳	惠州	18:29:00	18:29:00	152	36
03820	动车	惠州	深圳	06:23:00	22:23:00	230	24
03833	动车	惠州	深圳	22:22:00	23:22:00	462	7
04020	动车	广州	深圳	17:40:00	23:40:00	329	25
0408	动车	深圳	惠州	06:25:00	16:25:00	475	34
04213	动车	广州	深圳	15:37:00	18:37:00	47	26
04257	动车	广州	深圳	14:55:00	15:55:00	384	49
04806	动车	深圳	广州	18:34:00	20:34:00	486	6
05086	动车	深圳	广州	19:04:00	19:04:00	376	18
05095	动车	深圳	广州	20:02:00	20:02:00	562	21
05225	动车	广州	深圳	01:30:00	15:30:00	27	21
05980	动车	深圳	广州	14:03:00	23:03:00	18	46
05952	动车	深圳	广州	14:20:00	14:20:00	31	13
06061	动车	深圳	广州	22:12:00	22:12:00	136	14
06070	动车	深圳	广州	01:50:00	23:50:00	394	45
06278	动车	深圳	广州	16:03:00	19:03:00	414	25
06325	动车	深圳	广州	14:20:00	15:20:00	183	49

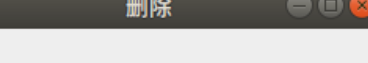
车次	类型	始发站	终点站	发车时间	预计到达时间	票价	剩余票
29	汽车	漳州	哈尔滨	10:52:00	15:52:00	11	40

确认购票？

确认

取消

学生信息管理系统



The screenshot shows a 'Delete' dialog box on the left and a 'Find' button on the right. The dialog box has a title bar with the text '删除' (Delete) and standard window controls. It contains a label '学号' (Student ID) next to a text input field, and a '确认' (Confirm) button below it. The 'Find' button is a blue rectangular button with the text '查找' (Find) on the right side.

ID	StudentName	Class	Chinese	Maths	English	Summary
10003	randomname	3	100	100	100	300

插入数据

6. 测试

测试数据：

关于这两个系统的测试数据，考虑到可行性和便利性，我采用的方案是随机生成。

简要的说明生成方案：

使用Python语言编写的随机数据生成器。

对于车票信息管理系统，车次信息格式为 [A-Z][1-9][0-9][0-9][0-9]；始发地和目的地在各省的地级市中随机抽取两个；发出时间是在00:00到23:59之间随机产生，预计抵达时间为出发时间加上随机生成的路程时间（12小时内）；车票价格随机生成区间 [10, 500]；剩余票数的随机生成区间 [0, 50]。

对于学生信息管理系统类似，但学生姓名采用了全英文的形式。

测试结果：

在最初的测试中发现数据库中的中文数据产生了乱码的现象，原因在于数据库采用的编码方案为ASCII和Latin，在修改为UTF-8后恢复正常。对于系统的测试，暂未发现异常。

7. 总结

总体来说，比较完整地实现了系统所需要的功能，但也暴露出来很多问题。

最重要的一个对于数据表单的设计。在学生信息管理系统中，把所有的列属性全都加入到了一张表内，导致表属性过于庞杂并且容易在后续操作中产生异常。通过对数据库理论的更深入的学习后，了解到了能避免该问题的方案：BF范式。经过范式的划分后，每个表单的属性更加清晰，而且在插入和删除的过程中能避免很多异常的发生。这为我往后的系统设计提供了新的思路，也避免了很多问题。其他问题还有关于系统的构架方案。

由于是第一次设计具有用户图形界面的具有复数功能的系统，在如

何构架系统方面出了一些问题，尤其是用户图形界面各个控件之间的关系没有搞清楚。在最初设计用户图形界面的时候，由于各个类之间的区别并不是很明显，导致某个控件类需要用到很多其他控件类的属性和方法，而这些属性和方法本来应该是私有的，却因为某个类的一次需要改为公有。这就导致了各个类的封装性被严重破坏，产生了很多问题，对今后的代码维护和修改也带来了麻烦。导致首次设计的图形用户界面被弃置。仔细考虑各类的关系后，设计出当前仍然比较粗糙图形用户界面。在这之间浪费了很多时间。总之，虽然成果并没有达到特别理想的状态，但仍然不失为一次宝贵的经验教训，为以后的项目设计铺平了道路。