上海电力学院

数据库设计课程设计



**题　　目: 车票实名销售系统**

**学生姓名： 黄海琨**

**学 号： 20131809 班 级： 2013054**

**院　　系：　　 计算机科学与技术学院**

**专业年级： 计算机科学与技术(卓越班)2013级**

**2015 年 6 月　16 日**

**目录**

[一、需求分析 1](#_Toc421746016)

[（1）问题描述 1](#_Toc421746017)

[（2）系统功能描述 1](#_Toc421746018)

[（3）安全性与完整性描述 1](#_Toc421746019)

[二、概念结构设计 1](#_Toc421746020)

[三、逻辑结构设计 3](#_Toc421746021)

[四、数据库的物理设计 4](#_Toc421746022)

[五、数据库设计实现及运行 4](#_Toc421746023)

[（1）数据库的创建 4](#_Toc421746024)

[（2）数据表的创建 4](#_Toc421746025)

[（3）视图的创建 6](#_Toc421746026)

[（4）存储过程的定义实现 6](#_Toc421746027)

[（5）触发器的定义实现 17](#_Toc421746028)

[（6）自行设计各模块中所涉及的操作语句 19](#_Toc421746029)

[1）插入数据操作（订票） 19](#_Toc421746030)

[2）删除数据操作（退票） 21](#_Toc421746031)

[3）按车次查询操作 22](#_Toc421746032)

[4）我的订单操作 23](#_Toc421746033)

[5）指定车次座位信息操作 24](#_Toc421746034)

[6）指定车次空余车票情况操作 25](#_Toc421746035)

[7）指定车次车票销售情况操作 26](#_Toc421746036)

[六、系统详细设计及实现 28](#_Toc421746037)

[（1）系统功能模块说明 28](#_Toc421746038)

[（2）模块的关键语句及关键技术说明 28](#_Toc421746039)

[七、附录：源程序代码 29](#_Toc421746040)

# 一、需求分析

（1）问题描述

在现代社会中，火车票的订票系统的智能型和数据库的数据海量性成为了关键。在本次车票实名制系统中，主要运用SQL Severs数据库和Java来实现的。使用数据库可以提高效率和降低成本

（2）系统功能描述

a.普通用户：

0.注册：若用户信息没有在人员信息表中，先在登录界面进行注册

1.订票：通过按出发地目的地和发车时间的查询来订票

2.退票：查找该用户的所有车票，选中相应的车票进行退票

3.按出发地目的地查询：输入出发地目的地的车站名称和发车时间查询列车

4.按车次查询：输入车次号进行列车详情的查询

5.按车站查询：输入站台名称查询经过该车站的列车信息

6.我的订单：指定乘客的客票购买信息

b.管理员用户：

1.指定车次的座位查询：按车次和座位等级统计座位总量

2.指定车次的车票剩余情况：指定日期指定车次的车票空余情况

3.指定车次的车票销售情况：统计指定日期指定车次的车票销售情况

（3）安全性与完整性描述

a.存储过程：按出发地目的地查询车次详情、是否拥有空余座位、订票、退票、增加用户

b.触发器：订票、退票

c.权限：普通用户和管理员用户拥有不同的权限

# 二、概念结构设计

列车信息（车次、类型、始发站、终点站、发车时间、到达时间、历时）

列车详情（车次、站次、车站名称、到达时间、停留时间、运行时间、里程、

一等座价格、二等座价格）

车站信息（车站编号、车站名称）

座位信息（车次、车厢号、座位号、座位类型）

购买信息（车次、发车日期、车厢号、座位号、购买日期、购买点、身份证号、

出发站、到达站、票价）

人员信息（姓名、身份证号、权限、密码）

剩余座位信息（车次、发车日期、站次、一等座空余、二等座空余）









# 三、逻辑结构设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 关系模式 | 属性 | 类型 | 是否空值 | 约束 | 索引 |
| 列车信息 | 车次 | 字符型 | NOT NULL | 主键 | 聚集索引 |
| 类型 | 字符型 | NOT NULL | 高铁或者动车 | 非聚集索引 |
| 始发站 | 字符型 | NOT NULL |  |  |
| 终点站 | 字符型 | NOT NULL |  |  |
| 发车时间 | 时间 | NOT NULL |  |  |
| 到达时间 | 时间 | NOT NULL |  |  |
| 历时 | 时间 | NOT NULL |  |  |
| 列车详情 | 车次 | 字符型 | NOT NULL | 主键，外键 | 聚集索引 |
| 站次 | 整型 | NOT NULL | 主键 | 聚集索引 |
| 车站名称 | 字符型 |  | 外键 | 非聚集索引 |
| 到达时间 | 时间 |  |  |  |
| 停留时间 | 时间 |  |  |  |
| 运行时间 | 时间 |  |  |  |
| 里程 | 整型 |  | 大于等于0 |  |
| 一等座价格 | money |  | 大于0 |  |
| 二等座价格 | money |  | 大于0 |  |
| 车站信息 | 车站编号 | 字符型 | NOT NULL |  |  |
| 车站名称 | 字符型 | NOT NULL | 主键 | 聚集索引 |
| 座位信息 | 车次 | 字符型 | NOT NULL | 主键，外键 | 聚集索引 |
| 车厢号 | 整型 | NOT NULL | 主键 | 聚集索引 |
| 座位号 | 字符型 | NOT NULL | 主键 | 聚集索引 |
| 座位类型 | 字符型 | NOT NULL | 一等座/二等座 |  |
| 购买信息 | 车次 | 字符型 | NOT NULL | 主键，外键 | 聚集索引 |
| 发车日期 | 日期 | NOT NULL |  | 非聚集索引 |
| 车厢号 | 整型 | NOT NULL | 主键 |  |
| 座位号 | 字符型 | NOT NULL | 主键 |  |
| 购买日期 | 日期 | NOT NULL |  |  |
| 购买点 | 字符型 |  |  |  |
| 身份证号 | 字符型 | NOT NULL | 主键，外键 |  |
| 出发站 | 字符型 | NOT NULL |  | 非聚集索引 |
| 到达站 | 字符型 | NOT NULL |  | 非聚集索引 |
| 票价 | money | NOT NULL |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 人员信息 | 姓名 | 字符型 | NOT NULL |  | 非聚集索引 |
| 身份证号 | 字符型 | NOT NULL | 主键 | 聚集索引 |
| 权限 | 整型 | NOT NULL | 0为旅客1为管理员 |  |
| 密码 | 字符型 | NOT NULL |  |  |
| 剩余座位信息 | 车次 | 字符型 | NOT NULL | 主键，外键 | 聚集索引 |
| 发车日期 | 日期 | NOT NULL | 主键，外键 | 聚集索引 |
| 站次 | 整型 | NOT NULL | 主键，外键 | 聚集索引 |
| 一等座空余 | 整型 | NOT NULL | 大于等于0 |  |
| 二等座空余 | 整型 | NOT NULL | 大于等于0 |  |

# 四、数据库的物理设计

**a.聚集索引**

所有表中的主键都为聚集索引

**b.非聚集索引**

create nonclustered index Index\_\_列车信息\_\_类型 on 列车信息(类型)

create nonclustered index Index\_\_列车详情\_\_车站名称 on 列车详情(车站名称)

create nonclustered index Index\_\_购买信息\_\_发车日期 on 购买信息(发车日期)

create nonclustered index Index\_\_购买信息\_\_出发站 on 购买信息(出发站)

create nonclustered index Index\_\_购买信息\_\_到达站 on 购买信息(到达站)

create nonclustered index Index\_\_人员信息\_\_姓名 on 人员信息(姓名)

# 五、数据库设计实现及运行

## （1）数据库的创建

create datebase ticket

## （2）数据表的创建

CREATE TABLE [dbo].[人员信息] (

[姓名] nvarchar(255) NOT NULL,

[身份证号] nvarchar(255) PRIMARY KEY,

[权限] int NOT NULL

)

CREATE TABLE [dbo].[列车信息] (

[车次] nvarchar(255) PRIMARY KEY,

[类型] nvarchar(255) CHECK(类型='动车'or 类型='高铁') NOT NULL,

[ 始发站] nvarchar(255) NOT NULL,

[ 终点站] nvarchar(255) NOT NULL,

[发车时间] time(0) NOT NULL,

[到达时间] time(0) NOT NULL,

[ 历时] time(0) NOT NULL

)

CREATE TABLE [dbo].[列车详情] (

[车次] nvarchar(255) NOT NULL,

[站次] int NOT NULL,

[车站名称] nvarchar(255),

[里程] int,

[到达时间] time(0),

[停留时间] time(0),

[运行时间] time(0),

[一等座价格] money,

[二等座价格] money,

PRIMARY KEY(车次,站次)

)

CREATE TABLE [dbo].[座位信息] (

[车次] nvarchar(255) NOT NULL,

[车厢号] int NOT NULL,

[座位号] nvarchar(255) NOT NULL,

[座位类型] nvarchar(255) NOT NULL,

PRIMARY KEY(车次,车厢号,座位号)

)

CREATE TABLE [dbo].[车站信息] (

[车站编号] nvarchar(255) NOT NULL,

[车站名称] nvarchar(255) PRIMARY KEY,

)

CREATE TABLE [dbo].[购买信息] (

[车次] nvarchar(255) NOT NULL,

[发车日期] smalldatetime NOT NULL,

[车厢号] int NOT NULL,

[座位号] nvarchar(255) NOT NULL,

[购买日期] smalldatetime NOT NULL,

[购买点] nvarchar(255),

[身份证号] nvarchar(255) NOT NULL,

[出发站] nvarchar(255) NOT NULL,

[到达站] nvarchar(255) NOT NULL,

PRIMARY KEY(车次,车厢号,座位号,身份证号)

)

CREATE TABLE [dbo].[剩余座位信息](

[车次] [nvarchar](55) NOT NULL,

[发车日期] [nvarchar](55) NOT NULL,

[站次] [int] NOT NULL,

[一等座空余] [int] NULL,

[二等座空余] [int] NULL,

PRIMARY KEY(车次,发车日期,站次)

)

alter table 列车详情

add constraint FK\_\_列车详情\_\_车站名称 foreign key (车站名称) references 车站信息 (车站名称),

constraint FK\_\_列车详情\_\_车次 foreign key (车次) references 列车信息 (车次),

constraint CK\_\_列车详情\_\_里程 check (里程>=0),

constraint CK\_\_列车详情\_\_价格 check (一等座价格>=0 or 二等座价格>=0)

alter table 座位信息

add constraint CK\_\_座位信息\_\_座位类型 check (座位类型='一等座' or 座位类型='二等座'),

constraint FK\_\_座位信息\_\_车次 foreign key (车次) references 列车信息 (车次)

alter table 人员信息

add constraint CK\_\_人员信息\_\_权限 check (权限=1 or 权限=0)

alter table 购买信息

add constraint FK\_\_购买信息\_\_身份证号 foreign key (身份证号) references 人员信息 (身份证号),

constraint FK\_\_购买信息\_\_车次 foreign key (车次) references 列车信息 (车次),

constraint CK\_\_购买信息\_\_购买日期 default getdate() for 购买日期

alter table 剩余座位信息

add constraint CK\_\_剩余座位信息\_\_空余座位 check (一等座空余>=0 or 二等座空余>=0)

## （3）视图的创建

--车票视图

create view s\_buyinfo

as

select 姓名,人员信息.身份证号,车次,发车日期,车厢号,座位号,出发站,到达站

from 购买信息 right join 人员信息 on 购买信息.身份证号=人员信息.身份证号

## （4）存储过程的定义实现

--查询列车信息(出发站目的地)

if exists (select name from sysobjects where name='getTrain\_dist' and type='P')

drop procedure getTrain\_dist

go

create procedure getTrain\_dist

@start nvarchar(255),@end nvarchar(255)

as

if OBJECT\_ID('tempdb..#temp') is not null

drop table #temp

set nocount on

create table #temp(车次 nvarchar(255),类型 nvarchar(255),发站 nvarchar(255),到站 nvarchar(255),发车时间 time(0),到达时间 time(0),运行时间 int,一等座价格 float,二等座价格 float)

declare num cursor for

(select distinct 车次

from 列车详情

where 车站名称 in (@start)

and 车次 in (select distinct 车次

from 列车详情

where 车站名称 in (@end))

)

open num

declare @t nvarchar(255)

fetch next from num into @t

while @@fetch\_status=0

begin

if((select distinct 站次 from 列车详情 where 车次 = @t and 车站名称 = @start)<

(select distinct 站次 from 列车详情 where 车次 = @t and 车站名称 = @end))

begin

insert #temp

select 列车详情.车次,类型,@start as 发站,@end as 到站,

(select 到达时间 from 列车详情 where 车次=@t and 车站名称=@start) as 发车时间,

(select 到达时间 from 列车详情 where 车次=@t and 车站名称=@end) as 到达时间,

datediff(minute,(select 到达时间 from 列车详情 where 车次=@t and 车站名称=@start),

(select 到达时间 from 列车详情 where 车次=@t and 车站名称=@end)) as 运行时间,

(select 一等座价格 from 列车详情 where 车次=@t and 车站名称=@end)-(select 一等座价格 from 列车详情 where 车次=@t and 车站名称=@start) as 一等座价格,

(select 二等座价格 from 列车详情 where 车次=@t and 车站名称=@end)-(select 二等座价格 from 列车详情 where 车次=@t and 车站名称=@start) as 二等座价格

from 列车详情 inner join 列车信息 on 列车详情.车次=列车信息.车次

group by 列车详情.车次,类型

having 列车详情.车次=@t

end

fetch next from num into @t

end

close num

deallocate num

select \* from #temp

truncate table #temp

drop table #temp

--查询空余座位（出发地目的地)

if exists (select name from sysobjects where name='getTrain\_freeseat' and type='P')

drop procedure getTrain\_freeseat

go

create procedure getTrain\_freeseat

@start nvarchar(255),@end nvarchar(255),@starttime nvarchar(255)

as

if OBJECT\_ID('tempdb..#temp') is not null

drop table #temp

set nocount on

create table #temp(车次 nvarchar(255),发车日期 date,一等座空余 int,二等座空余 int)

declare num1 cursor for

(select distinct 车次

from 列车详情

where 车站名称 in (@start)

and 车次 in (select distinct 车次

from 列车详情

where 车站名称 in (@end))

)

open num1

declare @t nvarchar(255)

fetch next from num1 into @t

while @@fetch\_status=0

begin

insert #temp

select distinct 车次,发车日期,min(一等座空余) as 一等座剩余,min(二等座空余) as 二等座剩余

from 剩余座位信息

group by 车次,发车日期,站次

having 站次>=(select 站次 from 列车详情 where 车站名称=@start and 车次=@t)

and 站次<(select 站次 from 列车详情 where 车站名称=@end and 车次=@t)

and 车次=@t

and 发车日期=@starttime

fetch next from num1 into @t

end

close num1

deallocate num1

select 车次,发车日期,min(一等座空余) as 一等座空余,min(二等座空余) as 二等座空余 from #temp

group by 车次,发车日期

truncate table #temp

drop table #temp

**宿主语句：**

**public** String[][] queryStartEnd(String start,String end,String starttime){

String[][] rows = **null**;

**int** rowcount=0;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url, userName, userPwd);

CallableStatement c=con.prepareCall("{call getTrain\_dist(?,?)}");

c.setString("start",start);

c.setString("end",end);

ResultSet rs=c.executeQuery();

**while**(rs.next())

rowcount++;

rows = **new** String[rowcount][12];

c=con.prepareCall("{call getTrain\_dist(?,?)}");

c.setString("start",start);

c.setString("end",end);

rs=c.executeQuery();

rowcount=0;

**while**(rs.next())

{

rows[rowcount][0] = rs.getString(1);

rows[rowcount][1] = rs.getString(2);

rows[rowcount][3] = rs.getString(3);

rows[rowcount][4] = rs.getString(4);

rows[rowcount][5] = rs.getString(5);

rows[rowcount][6] = rs.getString(6);

rows[rowcount][7] = rs.getString(7);

rows[rowcount][8] = rs.getString(8);

rows[rowcount][9] = rs.getString(9);

rowcount++;

}

rowcount=0;

c=con.prepareCall("{call getTrain\_freeseat(?,?,?)}");

c.setString("start",start);

c.setString("end",end);

c.setString("starttime",starttime);

rs=c.executeQuery();

**while**(rs.next())

{

rows[rowcount][2] = rs.getString(2);

rows[rowcount][10] = rs.getString(3);

rows[rowcount][11] = rs.getString(4);

rowcount++;

}

c.close();

con.close();

} **catch** (Exception e) {

e.printStackTrace();

}

**return** rows;

}

**测试结果：**



--订票

if exists (select name from sysobjects where name='setTrain\_ticket' and type='P')

drop procedure setTrain\_ticket

go

create procedure setTrain\_ticket

@num nvarchar(55),@starttime date,@buystation nvarchar(255),

@id nvarchar(55),@start nvarchar(255),@end nvarchar(255),@type nvarchar(255),@flag int output

as

if (select count(\*) from 购买信息 where 车次=@num and 发车日期=@starttime and 身份证号=@id)<1

begin

if OBJECT\_ID('tempdb..#temp') is not null

drop table #temp

set nocount on

create table #temp(数量 int)

declare @carriagenum int,@seatnum nvarchar(55),@price money,@price\_start money,@price\_end money,@t int

if @type='一等座'

begin

select @price\_start=一等座价格 from 列车详情 where 车次=@num and 车站名称=@start

select @price\_end=一等座价格 from 列车详情 where 车次=@num and 车站名称=@end

set @price = @price\_end - @price\_start

insert #temp

select 一等座空余

from 剩余座位信息

group by 车次,发车日期,站次,一等座空余

having 站次>=(select 站次 from 列车详情 where 车站名称=@start and 车次=@num)

and 站次<(select 站次 from 列车详情 where 车站名称=@end and 车次=@num)

and 车次=@num

and 发车日期=@starttime

end

else

begin

select @price\_start=二等座价格 from 列车详情 where 车次=@num and 车站名称=@start

select @price\_end=二等座价格 from 列车详情 where 车次=@num and 车站名称=@end

set @price = @price\_end - @price\_start

insert #temp

select 二等座空余

from 剩余座位信息

group by 车次,发车日期,站次,二等座空余

having 站次>=(select 站次 from 列车详情 where 车站名称=@start and 车次=@num)

and 站次<(select 站次 from 列车详情 where 车站名称=@end and 车次=@num)

and 车次=@num

and 发车日期=@starttime

end

select @carriagenum=车厢号,@seatnum=座位号

from 座位信息

where 车次=@num

and 座位类型=@type

and (车厢号 not in (select 车厢号 from 购买信息 where 车次=@num and 发车日期=@starttime)

or 座位号 not in (select 座位号 from 购买信息 where 车次=@num and 发车日期=@starttime))

select @t=min(数量) from #temp

truncate table #temp

drop table #temp

if(@t>0)

begin

insert

into 购买信息

values (@num,@starttime,@carriagenum,@seatnum,getdate(),@buystation,@id,@start,@end,@price)

set @flag=1

end

else

set @flag=2

end

else

set @flag=0

**宿主语句：**

**public** **int** addTicket(String num,String starttime,String buystation,String id,String start,String end,String type){

**int** flag=-1;

**try** {

Class.*forName*(driver);

Connection conn = DriverManager.*getConnection*(url, userName, userPwd);

CallableStatement c=conn.prepareCall("{call setTrain\_ticket(?,?,?,?,?,?,?,?)}");

c.setString("num",num);

c.setString("starttime",starttime);

c.setString("buystation",buystation);

c.setString("id",id);

c.setString("start",start);

c.setString("end",end);

c.setString("type",type);

c.registerOutParameter(8,java.sql.Types.*INTEGER*);

c.execute();

flag=c.getInt(8);

c.close();

conn.close();

} **catch** (Exception ex) {

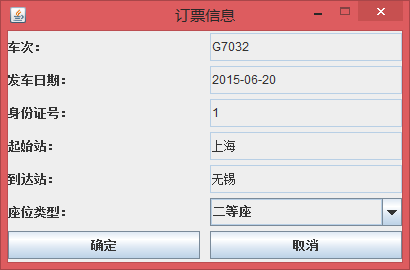
ex.printStackTrace();

}

**return** flag;

}

**测试结果：**



--退票

if exists (select name from sysobjects where name='setTrain\_ticket\_off' and type='P')

drop procedure setTrain\_ticket\_off

go

create procedure setTrain\_ticket\_off

@num nvarchar(55),@starttime date,@id nvarchar(55),@flag int output

as

set @flag=0

delete

from 购买信息

where 车次=@num and 发车日期=@starttime and 身份证号=@id

set @flag=1

**宿主语句：**

**public** **boolean** delTicket(String num,String starttime,String id){

**int** flag=-1;

**try** {

Class.*forName*(driver);

Connection conn = DriverManager.*getConnection*(url, userName, userPwd);

CallableStatement c=conn.prepareCall("{call setTrain\_ticket\_off(?,?,?,?)}");

c.setString("num",num);

c.setString("starttime",starttime);

c.setString("id",id);

c.registerOutParameter(4,java.sql.Types.*INTEGER*);

c.execute();

flag=c.getInt(4);

c.close();

conn.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

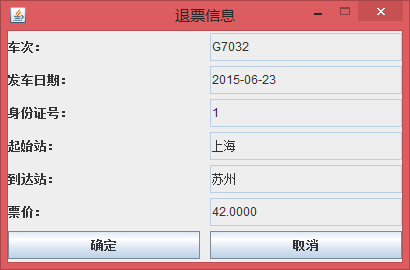
**if**(flag==1){

**return** **true** ;

}**else** **return** **false**;

}

**测试结果：**



--增加用户和管理员信息

if exists (select name from sysobjects where name='setPerson\_info' and type='P')

drop procedure setPerson\_infot

go

create procedure setPerson\_info

@id nvarchar(55),@name nvarchar(255),@level int

as

insert

into 人员信息

values (@name,@id,@level)

**宿主语句：**

**public** **boolean** isExistPerson(String id,String pwd,**int** level){

**int** flag=-1;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url, userName, userPwd);

Statement cmd = con.createStatement();

String sql = "select count(身份证号) from 人员信息 where 身份证号 = '"+id+"' and 权限='"+level+"' and 密码='"+pwd+"'";

ResultSet rs = cmd.executeQuery(sql);

rs.next();

flag=rs.getInt(1);

con.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

**if**(flag==1){

**return** **true** ;

}**else** **return** **false**;

}

**public** **boolean** addPerson(String name,String id,**int** level,String pwd){

**int** flag=-1;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url, userName, userPwd);

Statement cmd = con.createStatement();

String sql = "select count(身份证号) from 人员信息 where 身份证号 = '"+id+"' ";

ResultSet rs = cmd.executeQuery(sql);

rs.next();

flag=rs.getInt(1);

con.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

**if**(flag==0){

String sql = "insert into 人员信息(姓名,身份证号,权限,密码) values('"+name+"','"+id+"','"+level+"','"+pwd+"')";

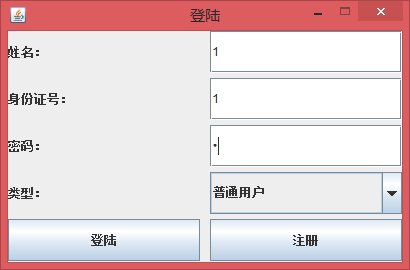
executeUpdate(sql);

**return** **true** ;

}**else** **return** **false**;

}

**测试结果：**



--按车站查询

if exists (select name from sysobjects where name='getTrain\_station' and type='P')

drop procedure getTrain\_station

go

create procedure getTrain\_station

@station nvarchar(255)

as

if OBJECT\_ID('tempdb..#temp') is not null

drop table #temp

set nocount on

create table #temp(车次 nvarchar(255),始发站 nvarchar(255),终点站 nvarchar(255),里程 int,到达时间 time(0),停留时间 time(0),运行时间 time(0),一等座价格 money,二等座价格 money)

declare num2 cursor for

(select distinct 车次 from 列车详情 where 车站名称=@station)

open num2

declare @t nvarchar(255)

fetch next from num2 into @t

while @@fetch\_status=0

begin

insert #temp

select 列车详情.车次,始发站,终点站,里程,列车详情.到达时间,停留时间,运行时间,一等座价格,二等座价格

from 列车详情 inner join 列车信息 on 列车详情.车次=列车信息.车次

group by 列车详情.车次,车站名称,始发站,终点站,里程,列车详情.到达时间,停留时间,运行时间,一等座价格,二等座价格

having 列车详情.车次=@t and 车站名称=@station

fetch next from num2 into @t

end

close num2

deallocate num2

select \* from #temp

truncate table #temp

drop table #temp

**宿主语句：**

**public** String[][] queryStation(String station){

String[][] rows = **null**;

**int** rowcount=0;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url, userName, userPwd);

CallableStatement c=con.prepareCall("{call getTrain\_station(?)}");

c.setString("station",station);

ResultSet rs=c.executeQuery();

**while**(rs.next())

rowcount++;

rows = **new** String[rowcount][9];

c=con.prepareCall("{call getTrain\_station(?)}");

c.setString("station",station);

rs=c.executeQuery();

rowcount=0;

**while**(rs.next())

{

**for** (**int** i = 0; i < 9; i++)

rows[rowcount][i] = rs.getString(i+1);

rowcount++;

}

c.close();

con.close();

} **catch** (Exception e) {

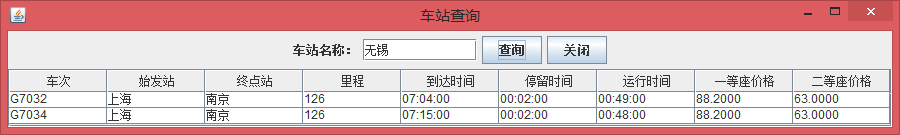
e.printStackTrace();

}

**return** rows;

}

**测试结果：**



## （5）触发器的定义实现

--订票触发器

if exists (select name from sysobjects where name='tr\_freeseat' and type='TR')

drop trigger tr\_freeseat

go

create trigger tr\_freeseat

on 购买信息

for insert

as

begin transaction

declare @num nvarchar(55),@carriagenum int,@seatnum nvarchar(55),@starttime date,

@start nvarchar(255),@end nvarchar(255),@level nvarchar(255)

select @start=出发站,@end=到达站,@num=车次,@carriagenum=车厢号,@seatnum=座位号,@starttime=发车日期

from inserted

select @level=座位类型

from 座位信息

where 车次=@num and 车厢号=@carriagenum and 座位号=@seatnum

if @level='一等座'

begin

update 剩余座位信息 set 一等座空余=一等座空余-1

where 站次>=(select 站次 from 列车详情 where 车站名称=@start and 车次=@num)

and 站次<(select 站次 from 列车详情 where 车站名称=@end and 车次=@num)

and 车次=@num

and 发车日期=@starttime

end

else

begin

update 剩余座位信息 set 二等座空余=二等座空余-1

where 站次>=(select 站次 from 列车详情 where 车站名称=@start and 车次=@num)

and 站次<(select 站次 from 列车详情 where 车站名称=@end and 车次=@num)

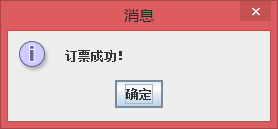
and 车次=@num

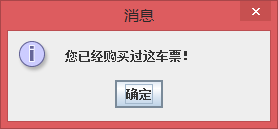
and 发车日期=@starttime

end

commit tran

**测试结果：**





--退票触发器

if exists (select name from sysobjects where name='tr\_freeseat\_off' and type='TR')

drop trigger tr\_freeseat\_off

go

create trigger tr\_freeseat\_off

on 购买信息

for delete

as

begin transaction

declare @num nvarchar(55),@carriagenum int,@seatnum nvarchar(55),@starttime date,

@start nvarchar(255),@end nvarchar(255),@level nvarchar(255)

select @start=出发站,@end=到达站,@num=车次,@carriagenum=车厢号,@seatnum=座位号,@starttime=发车日期

from deleted

select @level=座位类型

from 座位信息

where 车次=@num and 车厢号=@carriagenum and 座位号=@seatnum

if @level='一等座'

begin

update 剩余座位信息 set 一等座空余=一等座空余+1

where 站次>=(select 站次 from 列车详情 where 车站名称=@start and 车次=@num)

and 站次<(select 站次 from 列车详情 where 车站名称=@end and 车次=@num)

and 车次=@num

and 发车日期=@starttime

end

else

begin

update 剩余座位信息 set 二等座空余=二等座空余+1

where 站次>=(select 站次 from 列车详情 where 车站名称=@start and 车次=@num)

and 站次<(select 站次 from 列车详情 where 车站名称=@end and 车次=@num)

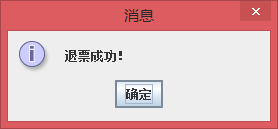
and 车次=@num

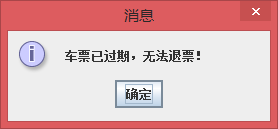
and 发车日期=@starttime

end

commit tran

**测试结果：**





## （6）自行设计各模块中所涉及的操作语句

### 1）插入数据操作（订票）

**a. 功能界面：**



**b. 功能界面简单描述：**

按照出发地和目的地及发车日期进行车票查询并且订票

**c. T-SQL语句与宿主语言嵌套使用代码段：**

**public** String[][] queryStartEnd(String start,String end,String starttime){

String[][] rows = **null**;

**int** rowcount=0;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url, userName, userPwd);

CallableStatement c=con.prepareCall("{call getTrain\_dist(?,?)}");

c.setString("start",start);

c.setString("end",end);

ResultSet rs=c.executeQuery();

**while**(rs.next())

rowcount++;

rows = **new** String[rowcount][12];

c=con.prepareCall("{call getTrain\_dist(?,?)}");

c.setString("start",start);

c.setString("end",end);

rs=c.executeQuery();

rowcount=0;

**while**(rs.next())

{

rows[rowcount][0] = rs.getString(1);

rows[rowcount][1] = rs.getString(2);

rows[rowcount][3] = rs.getString(3);

rows[rowcount][4] = rs.getString(4);

rows[rowcount][5] = rs.getString(5);

rows[rowcount][6] = rs.getString(6);

rows[rowcount][7] = rs.getString(7);

rows[rowcount][8] = rs.getString(8);

rows[rowcount][9] = rs.getString(9);

rowcount++;

}

rowcount=0;

c=con.prepareCall("{call getTrain\_freeseat(?,?,?)}");

c.setString("start",start);

c.setString("end",end);

c.setString("starttime",starttime);

rs=c.executeQuery();

**while**(rs.next())

{

rows[rowcount][2] = rs.getString(2);

rows[rowcount][10] = rs.getString(3);

rows[rowcount][11] = rs.getString(4);

rowcount++;

}

c.close();

con.close();

} **catch** (Exception e) {

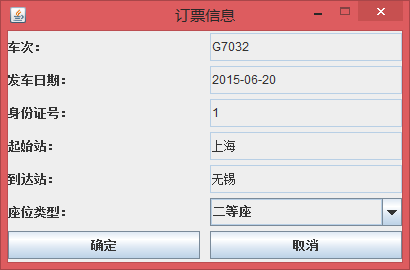
e.printStackTrace();

}

**return** rows;

}

**d. 测试结果：**



### 2）删除数据操作（退票）

**a. 功能界面：**



**b. 功能界面简单描述：**

查询改用户的所有购票记录并进行退票

**c. T-SQL语句与宿主语言嵌套使用代码段：**

**public** **boolean** delTicket(String num,String starttime,String id){

**int** flag=-1;

**try** {

Class.*forName*(driver);

Connection conn = DriverManager.*getConnection*(url, userName, userPwd);

CallableStatement c=conn.prepareCall("{call setTrain\_ticket\_off(?,?,?,?)}");

c.setString("num",num);

c.setString("starttime",starttime);

c.setString("id",id);

c.registerOutParameter(4,java.sql.Types.*INTEGER*);

c.execute();

flag=c.getInt(4);

c.close();

conn.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

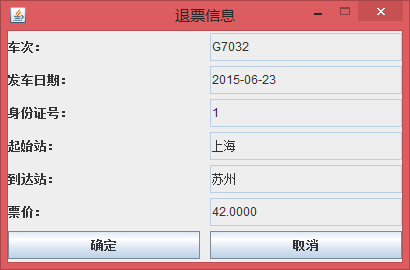
**if**(flag==1){

**return** **true** ;

}**else** **return** **false**;

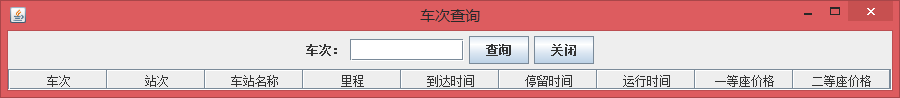
}

**d. 测试结果：**



### 3）按车次查询操作

**a. 功能界面：**



**b. 功能界面简单描述：**

按照指定车次查询车次详情

**c. T-SQL语句与宿主语言嵌套使用代码段：**

**public** String[][] queryNum(String num){

String[][] rows = **null**;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url,userName,userPwd);

Statement cmd = con.createStatement();

String sql = "select count(\*) from 列车详情 where 车次='"+num+"'";

ResultSet rs = cmd.executeQuery(sql);

rs.next();

**int** rowcount = rs.getInt(1);

rows = **new** String[rowcount][9];

rs = cmd.executeQuery("select \* from 列车详情 where 车次='"+num+"'");

rowcount = 0;

**while** (rs.next()) {

**for** (**int** i = 0; i < 9; i++)

rows[rowcount][i] = rs.getString(i+1);

rowcount++;

}

con.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

**return** rows;

}

**d. 测试结果：**



### 4）我的订单操作

**a. 功能界面：**



**b. 功能界面简单描述：**

指定用户的订单

**c. T-SQL语句与宿主语言嵌套使用代码段：**

**public** String[][] queryBuy(String id){

String[][] rows = **null**;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url,userName,userPwd);

Statement cmd = con.createStatement();

String sql = "select count(\*) from 购买信息 where 身份证号 ='"+id+"'";

ResultSet rs = cmd.executeQuery(sql);

rs.next();

**int** rowcount = rs.getInt(1);

rows = **new** String[rowcount][10];

rs = cmd.executeQuery("select \* from 购买信息 where 身份证号 ='"+id+"' order by 发车日期 desc");

rowcount = 0;

**while** (rs.next()) {

**for** (**int** i = 0; i < 10; i++)

rows[rowcount][i] = rs.getString(i+1);

rowcount++;

}

con.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

**return** rows;

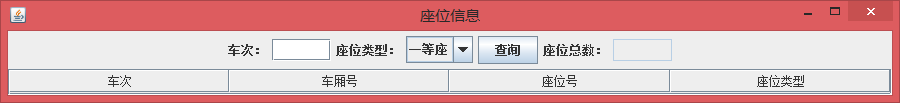
}

**d. 测试结果：**



### 5）指定车次座位信息操作

**a. 功能界面：**



**b. 功能界面简单描述：**

按指定车次查找座位信息

**c. T-SQL语句与宿主语言嵌套使用代码段：**

**public** String[][] querySeat(String num,String type){

String[][] rows = **null**;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url,userName,userPwd);

Statement cmd = con.createStatement();

String sql = "select count(\*) from 座位信息 where 车次='"+num+"' and 座位类型='"+type+"'";

ResultSet rs = cmd.executeQuery(sql);

rs.next();

**int** rowcount = rs.getInt(1);

rows = **new** String[rowcount][4];

rs = cmd.executeQuery("select \* from 座位信息 where 车次='"+num+"' and 座位类型='"+type+"'");

rowcount = 0;

**while** (rs.next()) {

**for** (**int** i = 0; i < 4; i++)

rows[rowcount][i] = rs.getString(i+1);

rowcount++;

}

con.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

**return** rows;

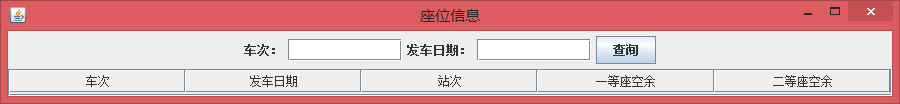
}

**d. 测试结果：**



### 6）指定车次空余车票情况操作

**a. 功能界面：**



**b. 功能界面简单描述：**

按指定车次查询空余车票情况

**c. T-SQL语句与宿主语言嵌套使用代码段：**

**public** String[][] queryFreeTicket(String num,String starttime){

String[][] rows = **null**;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url,userName,userPwd);

Statement cmd = con.createStatement();

String sql = "select count(\*) from 剩余座位信息 where 车次='"+num+"' and 发车日期='"+starttime+"'";

ResultSet rs = cmd.executeQuery(sql);

rs.next();

**int** rowcount = rs.getInt(1);

rows = **new** String[rowcount][5];

rs = cmd.executeQuery("select \* from 剩余座位信息 where 车次='"+num+"' and 发车日期='"+starttime+"'");

rowcount = 0;

**while** (rs.next()) {

**for** (**int** i = 0; i < 5; i++)

rows[rowcount][i] = rs.getString(i+1);

rowcount++;

}

con.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

**return** rows;

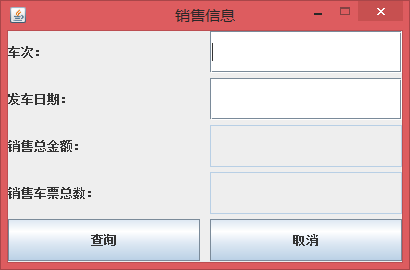
}

**d. 测试结果：**



### 7）指定车次车票销售情况操作

**a. 功能界面：**



**b. 功能界面简单描述：**

按指定车次查询车票销售情况

**c. T-SQL语句与宿主语言嵌套使用代码段：**

**public** **float**[] querySaleTicket(String num,String starttime){

**float**[] rows = **new** **float**[2];

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url,userName,userPwd);

Statement cmd = con.createStatement();

String sql = "select count(\*) from 购买信息 where 车次='"+num+"' and 发车日期='"+starttime+"'";

ResultSet rs = cmd.executeQuery(sql);

rs.next();

rows[0] = rs.getInt(1);

rs = cmd.executeQuery("select sum(票价) from 购买信息 where 车次='"+num+"' and 发车日期='"+starttime+"'");

rs.next();

String t=rs.getString(1);

**if**(t==**null**)

rows[1] = 0;

**else**

rows[1] = Float.*parseFloat*(t);

con.close();

} **catch** (Exception ex) {

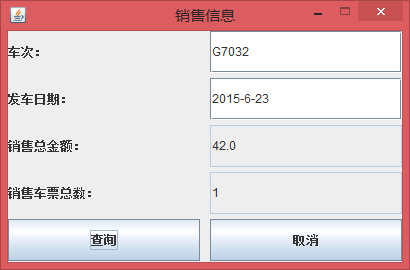
ex.printStackTrace();

}

**return** rows;

}

**d. 测试结果：**



# 六、系统详细设计及实现

## （1）系统功能模块说明



## （2）模块的关键语句及关键技术说明

在实现订票环节中，订票的存储过程的编写过程中使用了游标，打破了数据库原有的集合查询模式，实现单条数据行的查询和比较，并在输出的时候使用自建的临时表，把多条select语言的查询内容整合在一张表中并输出。

进行基本SQL语句的操作中使用

Statement cmd = con.createStatement();

ResultSet rs = cmd.executeQuery(sql);

进行存储过程的操作中使用

CallableStatement c=conn.prepareCall("{call xxx(?,?,?,?)}");

c.setString("xxx",num);

# 七、附录：源程序代码

**package** SQLSever;

**import** java.sql.\*;

**public** **class** TicketDao {

**private** String driver = "com.microsoft.sqlserver.jdbc.SQLServerDriver"; //加载JDBC驱动

**private** String url = "jdbc:sqlserver://localhost:1433; DatabaseName=ticket"; //连接服务器和数据库

**private** String userName = "sa"; //默认用户名

**private** String userPwd = "291910"; //密码

**private** **void** executeUpdate(String sql) {

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url, userName, userPwd);

Statement cmd = con.createStatement();

cmd.executeUpdate(sql);

con.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

}

**public** **boolean** isExistPerson(String id,String pwd,**int** level){

**int** flag=-1;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url, userName, userPwd);

Statement cmd = con.createStatement();

String sql = "select count(身份证号) from 人员信息 where 身份证号 = '"+id+"' and 权限='"+level+"' and 密码='"+pwd+"'";

ResultSet rs = cmd.executeQuery(sql);

rs.next();

flag=rs.getInt(1);

con.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

**if**(flag==1){

**return** **true** ;

}**else** **return** **false**;

}

**public** **boolean** addPerson(String name,String id,**int** level,String pwd){

**int** flag=-1;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url, userName, userPwd);

Statement cmd = con.createStatement();

String sql = "select count(身份证号) from 人员信息 where 身份证号 = '"+id+"' ";

ResultSet rs = cmd.executeQuery(sql);

rs.next();

flag=rs.getInt(1);

con.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

**if**(flag==0){

String sql = "insert into 人员信息(姓名,身份证号,权限,密码) values('"+name+"','"+id+"','"+level+"','"+pwd+"')";

executeUpdate(sql);

**return** **true** ;

}**else** **return** **false**;

}

**public** **int** addTicket(String num,String starttime,String buystation,String id,String start,String end,String type){

**int** flag=-1;

**try** {

Class.*forName*(driver);

Connection conn = DriverManager.*getConnection*(url, userName, userPwd);

CallableStatement c=conn.prepareCall("{call setTrain\_ticket(?,?,?,?,?,?,?,?)}");

c.setString("num",num);

c.setString("starttime",starttime);

c.setString("buystation",buystation);

c.setString("id",id);

c.setString("start",start);

c.setString("end",end);

c.setString("type",type);

c.registerOutParameter(8,java.sql.Types.*INTEGER*);

c.execute();

flag=c.getInt(8);

c.close();

conn.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

**return** flag;

}

**public** **boolean** delTicket(String num,String starttime,String id){

**int** flag=-1;

**try** {

Class.*forName*(driver);

Connection conn = DriverManager.*getConnection*(url, userName, userPwd);

CallableStatement c=conn.prepareCall("{call setTrain\_ticket\_off(?,?,?,?)}");

c.setString("num",num);

c.setString("starttime",starttime);

c.setString("id",id);

c.registerOutParameter(4,java.sql.Types.*INTEGER*);

c.execute();

flag=c.getInt(4);

c.close();

conn.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

**if**(flag==1){

**return** **true** ;

}**else** **return** **false**;

}

**public** String[][] queryStartEnd(String start,String end,String starttime){

String[][] rows = **null**;

**int** rowcount=0;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url, userName, userPwd);

CallableStatement c=con.prepareCall("{call getTrain\_dist(?,?)}");

c.setString("start",start);

c.setString("end",end);

ResultSet rs=c.executeQuery();

**while**(rs.next())

rowcount++;

rows = **new** String[rowcount][12];

c=con.prepareCall("{call getTrain\_dist(?,?)}");

c.setString("start",start);

c.setString("end",end);

rs=c.executeQuery();

rowcount=0;

**while**(rs.next())

{

rows[rowcount][0] = rs.getString(1);

rows[rowcount][1] = rs.getString(2);

rows[rowcount][3] = rs.getString(3);

rows[rowcount][4] = rs.getString(4);

rows[rowcount][5] = rs.getString(5);

rows[rowcount][6] = rs.getString(6);

rows[rowcount][7] = rs.getString(7);

rows[rowcount][8] = rs.getString(8);

rows[rowcount][9] = rs.getString(9);

rowcount++;

}

rowcount=0;

c=con.prepareCall("{call getTrain\_freeseat(?,?,?)}");

c.setString("start",start);

c.setString("end",end);

c.setString("starttime",starttime);

rs=c.executeQuery();

**while**(rs.next())

{

rows[rowcount][2] = rs.getString(2);

rows[rowcount][10] = rs.getString(3);

rows[rowcount][11] = rs.getString(4);

rowcount++;

}

c.close();

con.close();

} **catch** (Exception e) {

e.printStackTrace();

}

**return** rows;

}

**public** String[][] queryBuy(String id){

String[][] rows = **null**;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url,userName,userPwd);

Statement cmd = con.createStatement();

String sql = "select count(\*) from 购买信息 where 身份证号 ='"+id+"'";

ResultSet rs = cmd.executeQuery(sql);

rs.next();

**int** rowcount = rs.getInt(1);

rows = **new** String[rowcount][10];

rs = cmd.executeQuery("select \* from 购买信息 where 身份证号 ='"+id+"' order by 发车日期 desc");

rowcount = 0;

**while** (rs.next()) {

**for** (**int** i = 0; i < 10; i++)

rows[rowcount][i] = rs.getString(i+1);

rowcount++;

}

con.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

**return** rows;

}

**public** String[][] queryStation(String station){

String[][] rows = **null**;

**int** rowcount=0;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url, userName, userPwd);

CallableStatement c=con.prepareCall("{call getTrain\_station(?)}");

c.setString("station",station);

ResultSet rs=c.executeQuery();

**while**(rs.next())

rowcount++;

rows = **new** String[rowcount][9];

c=con.prepareCall("{call getTrain\_station(?)}");

c.setString("station",station);

rs=c.executeQuery();

rowcount=0;

**while**(rs.next())

{

**for** (**int** i = 0; i < 9; i++)

rows[rowcount][i] = rs.getString(i+1);

rowcount++;

}

c.close();

con.close();

} **catch** (Exception e) {

e.printStackTrace();

}

**return** rows;

}

**public** String[][] queryNum(String num){

String[][] rows = **null**;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url,userName,userPwd);

Statement cmd = con.createStatement();

String sql = "select count(\*) from 列车详情 where 车次='"+num+"'";

ResultSet rs = cmd.executeQuery(sql);

rs.next();

**int** rowcount = rs.getInt(1);

rows = **new** String[rowcount][9];

rs = cmd.executeQuery("select \* from 列车详情 where 车次='"+num+"'");

rowcount = 0;

**while** (rs.next()) {

**for** (**int** i = 0; i < 9; i++)

rows[rowcount][i] = rs.getString(i+1);

rowcount++;

}

con.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

**return** rows;

}

**public** String[][] querySeat(String num,String type){

String[][] rows = **null**;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url,userName,userPwd);

Statement cmd = con.createStatement();

String sql = "select count(\*) from 座位信息 where 车次='"+num+"' and 座位类型='"+type+"'";

ResultSet rs = cmd.executeQuery(sql);

rs.next();

**int** rowcount = rs.getInt(1);

rows = **new** String[rowcount][4];

rs = cmd.executeQuery("select \* from 座位信息 where 车次='"+num+"' and 座位类型='"+type+"'");

rowcount = 0;

**while** (rs.next()) {

**for** (**int** i = 0; i < 4; i++)

rows[rowcount][i] = rs.getString(i+1);

rowcount++;

}

con.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

**return** rows;

}

**public** String[][] queryFreeTicket(String num,String starttime){

String[][] rows = **null**;

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url,userName,userPwd);

Statement cmd = con.createStatement();

String sql = "select count(\*) from 剩余座位信息 where 车次='"+num+"' and 发车日期='"+starttime+"'";

ResultSet rs = cmd.executeQuery(sql);

rs.next();

**int** rowcount = rs.getInt(1);

rows = **new** String[rowcount][5];

rs = cmd.executeQuery("select \* from 剩余座位信息 where 车次='"+num+"' and 发车日期='"+starttime+"'");

rowcount = 0;

**while** (rs.next()) {

**for** (**int** i = 0; i < 5; i++)

rows[rowcount][i] = rs.getString(i+1);

rowcount++;

}

con.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

**return** rows;

}

**public** **float**[] querySaleTicket(String num,String starttime){

**float**[] rows = **new** **float**[2];

**try** {

Class.*forName*(driver);

Connection con = DriverManager.*getConnection*(url,userName,userPwd);

Statement cmd = con.createStatement();

String sql = "select count(\*) from 购买信息 where 车次='"+num+"' and 发车日期='"+starttime+"'";

ResultSet rs = cmd.executeQuery(sql);

rs.next();

rows[0] = rs.getInt(1);

rs = cmd.executeQuery("select sum(票价) from 购买信息 where 车次='"+num+"' and 发车日期='"+starttime+"'");

rs.next();

String t=rs.getString(1);

**if**(t==**null**)

rows[1] = 0;

**else**

rows[1] = Float.*parseFloat*(t);

con.close();

} **catch** (Exception ex) {

ex.printStackTrace();

}

**return** rows;

}

}

package SQLSever;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class LoginFrm extends JFrame implements ActionListener{

TicketDao ticketDao = new TicketDao();

JLabel lbl\_name =new JLabel("姓名：");

JLabel lbl\_id =new JLabel("身份证号：");

JLabel lbl\_pwd =new JLabel("密码：");

JLabel lb\_type=new JLabel("类型：");

JTextField txt\_name=new JTextField();

JTextField txt\_id=new JTextField();

JPasswordField txt\_pwd=new JPasswordField();

JComboBox cmb\_type=new JComboBox();

JButton btn\_OK=new JButton("登陆");

JButton btn\_new=new JButton("注册");

public LoginFrm(){

JPanel jp=(JPanel)this.getContentPane();

jp.setLayout(new GridLayout(5,2,10,5));

jp.add(lbl\_name);jp.add(txt\_name);

jp.add(lbl\_id);jp.add(txt\_id);

jp.add(lbl\_pwd);jp.add(txt\_pwd);

jp.add(lb\_type);jp.add(cmb\_type);

cmb\_type.addItem("普通用户");

cmb\_type.addItem("管理员");

jp.add(btn\_OK);jp.add(btn\_new);

this.setTitle("登陆");

this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

btn\_OK.addActionListener(this);

btn\_new.addActionListener(this);

this.setSize(400, 260);

this.setVisible(true);

this.setResizable(false);

}

public void actionPerformed(ActionEvent e){

String name=txt\_name.getText();

String id=txt\_id.getText();

String pwd=txt\_pwd.getText();

int level;

if (cmb\_type.getSelectedItem()=="普通用户")

level=0;

else level=1;

if (e.getSource() == btn\_OK) {

if(level==0){

if(ticketDao.isExistPerson(id,pwd,level)){

new MainFrm\_user(id);

dispose();

}

else JOptionPane.showMessageDialog(this, "error！");

}

else{

if(ticketDao.isExistPerson(id,pwd,level)){

new MainFrm\_admin(id);

dispose();

}

else JOptionPane.showMessageDialog(this, "error！");

}

}

if (e.getSource() == btn\_new) {

if(level==0){

if(ticketDao.addPerson(name,id,level,pwd)){

JOptionPane.showMessageDialog(this, "注册成功！");

dispose();

new LoginFrm();

}

else JOptionPane.showMessageDialog(this, "该用户已存在！");

}else JOptionPane.showMessageDialog(this, "您无法注册为管理员！");

}

}

public static void main(String[] args) {

new LoginFrm();

}

}

package SQLSever;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class MainFrm\_user extends JFrame {

String id;

private void initMenu() {

JMenuBar bar = new JMenuBar();

JMenu m1 = new JMenu("订票");

JMenu m2 = new JMenu("退票");

JMenu m3 = new JMenu("我的订单");

JMenuItem m11 = new JMenuItem("按出发地目的地");

JMenuItem m12 = new JMenuItem("按车次");

JMenuItem m13 = new JMenuItem("按车站");

JMenuItem m21 = new JMenuItem("退票");

JMenuItem m31 = new JMenuItem("我的订单");

m11.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

new StartEndinfoFrm(id);

}

});

m12.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

new NuminfoFrm();

}

});

m13.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

new StationinfoFrm();

}

});

m21.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

new BuyinfoFrm\_del(id);

}

});

m31.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

new BuyinfoFrm(id);

}

});

m1.add(m11);

m1.add(m12);

m1.add(m13);

m2.add(m21);

m3.add(m31);

bar.add(m1);

bar.add(m2);

bar.add(m3);

this.setJMenuBar(bar);

}

public MainFrm\_user(String id) {

this.id=id;

JPanel jp = (JPanel) this.getContentPane();

initMenu();

this.setTitle("火车票订票系统");

this.setSize(1000, 600);

this.setVisible(true);

this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

}

public static void main(String[] args) {

new MainFrm\_user("1");

}

}

package SQLSever;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class MainFrm\_admin extends JFrame {

String id;

private void initMenu() {

JMenuBar bar = new JMenuBar();

JMenu m1 = new JMenu("管理");

JMenuItem m11 = new JMenuItem("指定车次座位信息");

JMenuItem m12 = new JMenuItem("指定车次空余车票情况");

JMenuItem m13 = new JMenuItem("指定车次车票销售情况");

JMenuItem m14 = new JMenuItem("注销");

m11.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

new SeatinfoFrm();

}

});

m12.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

new FreeTicketinfoFrm();

}

});

m13.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

new SaleTicketinfoFrm();

}

});

m14.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

dispose();

new LoginFrm();

}

});

m1.add(m11);

m1.add(m12);

m1.add(m13);

m1.addSeparator();

m1.add(m14);

bar.add(m1);

this.setJMenuBar(bar);

}

public MainFrm\_admin(String id) {

this.id=id;

JPanel jp = (JPanel) this.getContentPane();

initMenu();

this.setTitle("火车票订票系统——管理");

this.setSize(1000, 600);

this.setVisible(true);

this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

}

public static void main(String[] args) {

new MainFrm\_admin("1");

}

}

package SQLSever;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class BookFrm extends JFrame implements ActionListener{

TicketDao ticketDao = new TicketDao();

JLabel lbl\_num =new JLabel("车次：");

JLabel lbl\_starttime =new JLabel("发车日期：");

JLabel lbl\_id =new JLabel("身份证号：");

JLabel lbl\_start =new JLabel("起始站：");

JLabel lbl\_end =new JLabel("到达站：");

JLabel lbl\_type =new JLabel("座位类型：");

JTextField txt\_num=new JTextField();

JTextField txt\_starttime=new JTextField();

JTextField txt\_id=new JTextField();

JTextField txt\_start=new JTextField();

JTextField txt\_end=new JTextField();

JComboBox cmb\_type=new JComboBox();

JButton btn\_OK=new JButton("确定");

JButton btn\_Cancel=new JButton("取消");

public BookFrm(String num,String starttime,String id,String start,String end){

txt\_num.setText(num);

txt\_starttime.setText(starttime);

txt\_id.setText(id);

txt\_start.setText(start);

txt\_end.setText(end);

JPanel jp=(JPanel)this.getContentPane();

jp.setLayout(new GridLayout(7,2,10,5));

txt\_num.setEditable(false);

txt\_starttime.setEditable(false);

txt\_id.setEditable(false);

txt\_start.setEditable(false);

txt\_end.setEditable(false);

jp.add(lbl\_num);jp.add(txt\_num);

jp.add(lbl\_starttime);jp.add(txt\_starttime);

jp.add(lbl\_id);jp.add(txt\_id);

jp.add(lbl\_start);jp.add(txt\_start);

jp.add(lbl\_end);jp.add(txt\_end);

jp.add(lbl\_type);jp.add(cmb\_type);

cmb\_type.addItem("二等座");

cmb\_type.addItem("一等座");

jp.add(btn\_OK);jp.add(btn\_Cancel);

this.setTitle("订票信息");

btn\_OK.addActionListener(this);

btn\_Cancel.addActionListener(this);

this.setSize(400, 260);

this.setVisible(true);

this.setResizable(false);

}

public void actionPerformed(ActionEvent e){

String num=txt\_num.getText();

String starttime=txt\_starttime.getText();

String start=txt\_start.getText();

String end=txt\_end.getText();

String id=txt\_id.getText();

String type=(String) cmb\_type.getSelectedItem();

if (e.getSource() == btn\_OK) {

int flag=ticketDao.addTicket(num,starttime,"网络",id,start,end,type);

switch(flag){

case 1:JOptionPane.showMessageDialog(this, "订票成功！");

dispose();break;

case 0:JOptionPane.showMessageDialog(this, "您已经购买过这车票！");break;

case 2:JOptionPane.showMessageDialog(this, "此车票已买完！");break;

}

}

if (e.getSource() == btn\_Cancel) {

dispose();

}

}

}

package SQLSever;

import java.awt.BorderLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

import javax.swing.JScrollPane;

import javax.swing.JTable;

import javax.swing.JTextField;

import javax.swing.table.DefaultTableModel;

public class BuyinfoFrm\_del extends JFrame implements ActionListener{

String id;

JButton btn\_del=new JButton("退票");

private JTable table = null;

private String[] cols = { "车次","发车日期","车厢号","座位号","购买日期 ","购买点","身份证号","出发站","到达站","票价"};

private TicketDao ticketDao = new TicketDao();

private void initTable\_station(String id) {

String[][] rows = ticketDao.queryBuy(id);

table = new JTable(rows, cols);

}

public BuyinfoFrm\_del(String id) {

this.id=id;

JPanel jp = (JPanel) this.getContentPane();

initTable\_station(id);

JScrollPane jsp\_table = new JScrollPane(table);

table.setAutoResizeMode(1);

jp.add(jsp\_table);

JPanel jp\_top = new JPanel();

jp\_top.add(btn\_del);

jp.add(jp\_top, BorderLayout.NORTH);

btn\_del.addActionListener(this);

this.setTitle("退票查询");

this.setSize(900, 400);

this.setVisible(true);

}

public void actionPerformed(ActionEvent e) {

if(e.getSource()==btn\_del){

if(table.getSelectedRow()!=-1)

if(table.getValueAt(table.getSelectedRow(), 0)!=""){

String num=(String) table.getValueAt(table.getSelectedRow(), 0);

String starttime=(String) table.getValueAt(table.getSelectedRow(), 1);

String start=(String) table.getValueAt(table.getSelectedRow(), 7);

String end=(String) table.getValueAt(table.getSelectedRow(), 8);

String price=(String) table.getValueAt(table.getSelectedRow(), 9);

new DeleteFrm(num,starttime,id,start,end,price);

dispose();

}

}

}

}

package SQLSever;

import java.awt.BorderLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JComboBox;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

import javax.swing.JScrollPane;

import javax.swing.JTable;

import javax.swing.JTextField;

import javax.swing.table.DefaultTableModel;

public class FreeTicketinfoFrm extends JFrame implements ActionListener{

int i = 0;

JButton btn\_OK=new JButton("查询");

JLabel lbl\_num =new JLabel("车次：");

JLabel lbl\_starttime =new JLabel("发车日期：");

JTextField txt\_num=new JTextField("",10);

JTextField txt\_starttime=new JTextField("",10);

private JTable table = null;

private String[] cols = { "车次","发车日期","站次","一等座空余","二等座空余"};

private TicketDao ticketDao = new TicketDao();

private void initTable\_station() {

String[][] rows = {{"","","","",""}};

table = new JTable(rows, cols);

}

private void updateTable(String num,String starttime) {

String[][] rows = ticketDao.queryFreeTicket(num,starttime);

table.setModel(new DefaultTableModel(rows, cols));

i=rows.length;

}

public FreeTicketinfoFrm() {

JPanel jp = (JPanel) this.getContentPane();

initTable\_station();

JScrollPane jsp\_table = new JScrollPane(table);

table.setAutoResizeMode(1);

updateTable("","");

jp.add(jsp\_table);

JPanel jp\_top = new JPanel();

jp\_top.add(lbl\_num);jp\_top.add(txt\_num);

jp\_top.add(lbl\_starttime);jp\_top.add(txt\_starttime);

jp\_top.add(btn\_OK);

jp.add(jp\_top, BorderLayout.NORTH);

btn\_OK.addActionListener(this);

this.setTitle("座位信息");

this.setSize(900, 400);

this.setVisible(true);

}

public void actionPerformed(ActionEvent e) {

String num = txt\_num.getText();

String starttime = txt\_starttime.getText();

if(e.getSource()==btn\_OK){

updateTable(num,starttime);

table.setAutoResizeMode(1);

}

}

public static void main(String[] args) {

new FreeTicketinfoFrm();

}

}

package SQLSever;

import java.awt.BorderLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

import javax.swing.JScrollPane;

import javax.swing.JTable;

import javax.swing.JTextField;

import javax.swing.table.DefaultTableModel;

public class BuyinfoFrm extends JFrame implements ActionListener{

String id;

JButton btn\_del=new JButton("关闭");

private JTable table = null;

private String[] cols = { "车次","发车日期","车厢号","座位号","购买日期 ","购买点","身份证号","出发站","到达站","票价"};

private TicketDao ticketDao = new TicketDao();

private void initTable\_station(String id) {

String[][] rows = ticketDao.queryBuy(id);

table = new JTable(rows, cols);

}

public BuyinfoFrm(String id) {

this.id=id;

JPanel jp = (JPanel) this.getContentPane();

initTable\_station(id);

JScrollPane jsp\_table = new JScrollPane(table);

table.setAutoResizeMode(1);

jp.add(jsp\_table);

JPanel jp\_top = new JPanel();

jp\_top.add(btn\_del);

jp.add(jp\_top, BorderLayout.NORTH);

btn\_del.addActionListener(this);

this.setTitle("我的订单");

this.setSize(900, 400);

this.setVisible(true);

}

public void actionPerformed(ActionEvent e) {

if(e.getSource()==btn\_del){

dispose();

}

}

}

package SQLSever;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Locale;

import java.util.Date;

import java.util.\*;

public class DeleteFrm extends JFrame implements ActionListener{

TicketDao ticketDao = new TicketDao();

JLabel lbl\_num =new JLabel("车次：");

JLabel lbl\_starttime =new JLabel("发车日期：");

JLabel lbl\_id =new JLabel("身份证号：");

JLabel lbl\_start =new JLabel("起始站：");

JLabel lbl\_end =new JLabel("到达站：");

JLabel lbl\_price =new JLabel("票价：");

JTextField txt\_num=new JTextField();

JTextField txt\_starttime=new JTextField();

JTextField txt\_id=new JTextField();

JTextField txt\_start=new JTextField();

JTextField txt\_end=new JTextField();

JTextField txt\_price=new JTextField();

JButton btn\_OK=new JButton("确定");

JButton btn\_Cancel=new JButton("取消");

public DeleteFrm(String num,String starttime,String id,String start,String end,String price){

txt\_num.setText(num);

txt\_starttime.setText(starttime);

txt\_id.setText(id);

txt\_start.setText(start);

txt\_end.setText(end);

txt\_price.setText(price);

JPanel jp=(JPanel)this.getContentPane();

jp.setLayout(new GridLayout(7,2,10,5));

txt\_num.setEditable(false);

txt\_starttime.setEditable(false);

txt\_id.setEditable(false);

txt\_start.setEditable(false);

txt\_end.setEditable(false);

txt\_price.setEditable(false);

jp.add(lbl\_num);jp.add(txt\_num);

jp.add(lbl\_starttime);jp.add(txt\_starttime);

jp.add(lbl\_id);jp.add(txt\_id);

jp.add(lbl\_start);jp.add(txt\_start);

jp.add(lbl\_end);jp.add(txt\_end);

jp.add(lbl\_price);jp.add(txt\_price);

jp.add(btn\_OK);jp.add(btn\_Cancel);

this.setTitle("退票信息");

btn\_OK.addActionListener(this);

btn\_Cancel.addActionListener(this);

this.setSize(400, 260);

this.setVisible(true);

this.setResizable(false);

}

public void actionPerformed(ActionEvent e){

String num=txt\_num.getText();

String starttime=txt\_starttime.getText();

String start=txt\_start.getText();

String end=txt\_end.getText();

String id=txt\_id.getText();

Date dt=new Date();

SimpleDateFormat matter1=new SimpleDateFormat("yyyy-MM-dd");

String ttime= matter1.format(dt);

long longstr1 = Long.valueOf(starttime.replaceAll("[-\\s:]",""));

long longstr2 = Long.valueOf(ttime.replaceAll("[-\\s:]",""));

if(e.getSource() == btn\_OK){

if (longstr1>longstr2) {

if(ticketDao.delTicket(num,starttime,id)){

JOptionPane.showMessageDialog(this, "退票成功！");

dispose();

new BuyinfoFrm\_del(id);

}

else JOptionPane.showMessageDialog(this, "失败！");

}

else

JOptionPane.showMessageDialog(this, "车票已过期，无法退票！");

}

if (e.getSource() == btn\_Cancel) {

dispose();

}

}

}

package SQLSever;

import java.awt.BorderLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

import javax.swing.JScrollPane;

import javax.swing.JTable;

import javax.swing.JTextField;

import javax.swing.table.DefaultTableModel;

public class NuminfoFrm extends JFrame implements ActionListener{

JButton btn\_del=new JButton("关闭");

JButton btn\_OK=new JButton("查询");

JLabel lbl\_num =new JLabel("车次：");

JTextField txt\_num=new JTextField("",10);

private JTable table = null;

private String[] cols = { "车次","站次","车站名称","里程","到达时间","停留时间","运行时间","一等座价格","二等座价格"};

private TicketDao ticketDao = new TicketDao();

private void initTable() {

String[][] rows = {{"","","","","","","","",""}};

table = new JTable(rows, cols);

}

private void updateTable(String num) {

String[][] rows = ticketDao.queryNum(num);

table.setModel(new DefaultTableModel(rows, cols));

}

public NuminfoFrm() {

JPanel jp = (JPanel) this.getContentPane();

initTable();

JScrollPane jsp\_table = new JScrollPane(table);

table.setAutoResizeMode(1);

updateTable("");

jp.add(jsp\_table);

JPanel jp\_top = new JPanel();

jp\_top.add(lbl\_num);jp\_top.add(txt\_num);

jp\_top.add(btn\_OK);jp\_top.add(btn\_del);

jp.add(jp\_top, BorderLayout.NORTH);

btn\_del.addActionListener(this);

btn\_OK.addActionListener(this);

this.setTitle("车次查询");

this.setSize(900, 400);

this.setVisible(true);

}

public void actionPerformed(ActionEvent e) {

String num;

if(e.getSource()==btn\_del){

dispose();

}

if(e.getSource()==btn\_OK){

num = txt\_num.getText();

updateTable(num);

table.setAutoResizeMode(1);

}

}

public static void main(String[] args) {

new NuminfoFrm();

}

}

package SQLSever;

import java.awt.BorderLayout;

import java.awt.GridLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JComboBox;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

import javax.swing.JScrollPane;

import javax.swing.JTable;

import javax.swing.JTextField;

import javax.swing.table.DefaultTableModel;

public class SaleTicketinfoFrm extends JFrame implements ActionListener{

int i = 0;

JLabel lbl\_num =new JLabel("车次：");

JLabel lbl\_money =new JLabel("销售总金额：");

JLabel lbl\_starttime =new JLabel("发车日期：");

JLabel lbl\_count =new JLabel("销售车票总数：");

JTextField txt\_num=new JTextField("",10);

JTextField txt\_starttime=new JTextField("",10);

JTextField txt\_money=new JTextField("",10);

JTextField txt\_count=new JTextField("",10);

JButton btn\_OK=new JButton("查询");

JButton btn\_Cancel=new JButton("取消");

public SaleTicketinfoFrm() {

JPanel jp=(JPanel)this.getContentPane();

jp.setLayout(new GridLayout(5,2,10,5));

txt\_count.setEditable(false);

txt\_money.setEditable(false);

jp.add(lbl\_num);jp.add(txt\_num);

jp.add(lbl\_starttime);jp.add(txt\_starttime);

jp.add(lbl\_money);jp.add(txt\_money);

jp.add(lbl\_count);jp.add(txt\_count);

jp.add(btn\_OK);jp.add(btn\_Cancel);

this.setTitle("销售信息");

btn\_OK.addActionListener(this);

btn\_Cancel.addActionListener(this);

this.setSize(400, 260);

this.setVisible(true);

this.setResizable(false);

}

public void actionPerformed(ActionEvent e) {

TicketDao ticketDao = new TicketDao();

float[] temp = new float[2];

String num = txt\_num.getText();

String starttime = txt\_starttime.getText();

if(e.getSource()==btn\_OK){

temp=ticketDao.querySaleTicket(num,starttime);

txt\_count.setText(String.valueOf((int)temp[0]));

txt\_money.setText(String.valueOf(temp[1]));

}

if(e.getSource()==btn\_Cancel){

dispose();

}

}

public static void main(String[] args) {

new SaleTicketinfoFrm();

}

}

package SQLSever;

import java.awt.BorderLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

import javax.swing.JScrollPane;

import javax.swing.JTable;

import javax.swing.JTextField;

import javax.swing.table.DefaultTableModel;

public class StationinfoFrm extends JFrame implements ActionListener{

JButton btn\_del=new JButton("关闭");

JButton btn\_OK=new JButton("查询");

JLabel lbl\_station =new JLabel("车站名称：");

JTextField txt\_station=new JTextField("",10);

private JTable table = null;

private String[] cols = { "车次","始发站","终点站","里程","到达时间","停留时间","运行时间","一等座价格","二等座价格"};

private TicketDao ticketDao = new TicketDao();

private void initTable() {

String[][] rows = {{"","","","","","","","",""}};

table = new JTable(rows, cols);

}

private void updateTable(String station) {

String[][] rows = ticketDao.queryStation(station);

table.setModel(new DefaultTableModel(rows, cols));

}

public StationinfoFrm() {

JPanel jp = (JPanel) this.getContentPane();

initTable();

JScrollPane jsp\_table = new JScrollPane(table);

table.setAutoResizeMode(1);

updateTable("");

jp.add(jsp\_table);

JPanel jp\_top = new JPanel();

jp\_top.add(lbl\_station);jp\_top.add(txt\_station);

jp\_top.add(btn\_OK);jp\_top.add(btn\_del);

jp.add(jp\_top, BorderLayout.NORTH);

btn\_del.addActionListener(this);

btn\_OK.addActionListener(this);

this.setTitle("车站查询");

this.setSize(900, 400);

this.setVisible(true);

}

public void actionPerformed(ActionEvent e) {

String station;

if(e.getSource()==btn\_del){

dispose();

}

if(e.getSource()==btn\_OK){

station = txt\_station.getText();

updateTable(station);

table.setAutoResizeMode(1);

}

}

public static void main(String[] args) {

new StationinfoFrm();

}

}

package SQLSever;

import java.awt.BorderLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JComboBox;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

import javax.swing.JScrollPane;

import javax.swing.JTable;

import javax.swing.JTextField;

import javax.swing.table.DefaultTableModel;

public class SeatinfoFrm extends JFrame implements ActionListener{

int i = 0;

JButton btn\_OK=new JButton("查询");

JLabel lbl\_num =new JLabel("车次：");

JLabel lbl\_type =new JLabel("座位类型：");

JLabel lbl\_all =new JLabel("座位总数：");

JComboBox cmb\_type=new JComboBox();

JTextField txt\_num=new JTextField("",5);

JTextField txt\_all=new JTextField("",5);

private JTable table = null;

private String[] cols = { "车次","车厢号","座位号","座位类型"};

private TicketDao ticketDao = new TicketDao();

private void initTable\_station() {

String[][] rows = {{"","","",""}};

table = new JTable(rows, cols);

}

private void updateTable(String num,String type) {

String[][] rows = ticketDao.querySeat(num,type);

table.setModel(new DefaultTableModel(rows, cols));

i=rows.length;

}

public SeatinfoFrm() {

JPanel jp = (JPanel) this.getContentPane();

initTable\_station();

JScrollPane jsp\_table = new JScrollPane(table);

table.setAutoResizeMode(1);

updateTable("","");

jp.add(jsp\_table);

JPanel jp\_top = new JPanel();

txt\_all.setEditable(false);

jp\_top.add(lbl\_num);jp\_top.add(txt\_num);

jp\_top.add(lbl\_type);jp\_top.add(cmb\_type);

cmb\_type.addItem("一等座");

cmb\_type.addItem("二等座");

jp\_top.add(btn\_OK);

jp\_top.add(lbl\_all);jp\_top.add(txt\_all);

jp.add(jp\_top, BorderLayout.NORTH);

btn\_OK.addActionListener(this);

this.setTitle("座位信息");

this.setSize(900, 400);

this.setVisible(true);

}

public void actionPerformed(ActionEvent e) {

String num = txt\_num.getText();

String type = (String) cmb\_type.getSelectedItem();

if(e.getSource()==btn\_OK){

updateTable(num,type);

table.setAutoResizeMode(1);

txt\_all.setText(String.valueOf(i));

}

}

public static void main(String[] args) {

new SeatinfoFrm();

}

}

package SQLSever;

import java.awt.BorderLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

import javax.swing.JScrollPane;

import javax.swing.JTable;

import javax.swing.JTextField;

import javax.swing.table.DefaultTableModel;

public class StartEndinfoFrm extends JFrame implements ActionListener{

String id;

JButton btn\_book=new JButton("订票");

JButton btn\_OK=new JButton("查询");

JLabel lbl\_start =new JLabel("出发站：");

JLabel lbl\_end =new JLabel("到达站：");

JLabel lbl\_time =new JLabel("日期：");

JTextField txt\_start=new JTextField("",10);

JTextField txt\_end=new JTextField("",10);

JTextField txt\_time=new JTextField("",10);

private JTable table = null;

private String[] cols\_station = { "车次","类型","发车日期","出发站","到达站","发车时间","到站时间","历时/分钟","一等座票价","二等座票价","一等座空余","二等座空余"};

private TicketDao ticketDao = new TicketDao();

private void initTable\_station() {

String[][] rows = {{"","","","","","","","","","","",""}};

table = new JTable(rows, cols\_station);

}

private void updateTable(String start,String end,String starttime) {

String[][] rows = ticketDao.queryStartEnd(start,end,starttime);

table.setModel(new DefaultTableModel(rows, cols\_station));

}

public StartEndinfoFrm(String id) {

this.id=id;

JPanel jp = (JPanel) this.getContentPane();

initTable\_station();

JScrollPane jsp\_table = new JScrollPane(table);

table.setAutoResizeMode(1);

updateTable("","","");

jp.add(jsp\_table);

JPanel jp\_top = new JPanel();

jp\_top.add(lbl\_start);jp\_top.add(txt\_start);

jp\_top.add(lbl\_end);jp\_top.add(txt\_end);

jp\_top.add(lbl\_time);jp\_top.add(txt\_time);

jp\_top.add(btn\_OK);jp\_top.add(btn\_book);

jp.add(jp\_top, BorderLayout.NORTH);

btn\_book.addActionListener(this);

btn\_OK.addActionListener(this);

this.setTitle("出发到达查询");

this.setSize(900, 400);

this.setVisible(true);

}

public void actionPerformed(ActionEvent e) {

String start0 = txt\_start.getText();

String end0 = txt\_end.getText();

String starttime0 = txt\_time.getText();

if(e.getSource()==btn\_book){

if(table.getSelectedRow()!=-1){

if(starttime0!=""){

if(table.getValueAt(table.getSelectedRow(), 2)!=""){

String num=(String) table.getValueAt(table.getSelectedRow(), 0);

String starttime=(String) table.getValueAt(table.getSelectedRow(), 2);

String start=(String) table.getValueAt(table.getSelectedRow(), 3);

String end=(String) table.getValueAt(table.getSelectedRow(), 4);

new BookFrm(num,starttime,id,start,end);

}

}

}

}

if(e.getSource()==btn\_OK){

start0 = txt\_start.getText();

end0 = txt\_end.getText();

starttime0 = txt\_time.getText();

updateTable(start0,end0,starttime0);

table.setAutoResizeMode(1);

}

}

public static void main(String[] args) {

new StartEndinfoFrm("1");

}

}