

## 实验二：小型机票订票系统设计

学号：22920142203809

姓名：冯立刚

年级：2014 级

一、	实验目的：	2
二、	实验内容：	2
三、	实验步骤：	2
1、	建立关系模型，用 ER 图表示如下：	3
2、	定义数据库表	3
3、	软件功能实现	5
4、	相关实现代码说明：	6
四、	实验系统演示：	11
五、	实验总结：	18

## 一、 实验目的：

- 1、能够正确运用数据库基本理论和知识，复习、巩固、提高数据库方案设计、论证和分析方法。
- 2、熟悉关系数据库规范化设计理论，根据实验要求设计并建立科学合理的数据库，正确建立数据库中表与表之间的关系。
- 3、进一步正确理解数据库设计思路，培养分析问题、解决问题的能力，提高查询资料和撰写书面文件的能力。

## 二、 实验内容：

[系统描述]

小型机票订票系统应具备航班信息管理、旅客信息管理、机票信息查询及订购基本管理功能，具体要求如下：

- (1) 能对航班信息进行维护：输入、修改、删除航班信息。
  - (2) 能对旅客信息进行维护。
  - (3) 能进行航班信息、余票情况及旅客信息的各种查询。
  - (4) 能进行机票的订购及退订，必须动态刷新库存。
  - (5) 能对每个航班进行按月、季度、年份进行售票统计。
  - (6) 能对每个旅客进行按月、季度、年份进行出行统计。
1. 根据上述系统功能需求，使用 ER 图描述该管理信息系统的概念模型。
  2. 完成该管理信息系统的数据库总体设计方案，明确数据库中表的结构，各表中关键字的设置，表与表之间的关系。
  3. 根据系统功能需求，以 SQL 语句的形式分类列出系统应涉及的数据操作。
  4. 选用熟悉的数据库工具，根据设计方案正确建立数据库，并成功实现上述数据操作。
  5. 完成上述内容，并提交书面实验报告。

## 三、 实验步骤：

实验平台：windows10, IDEA2016, SQLServer2016

1、建立关系模型，用 ER 图表示如下：

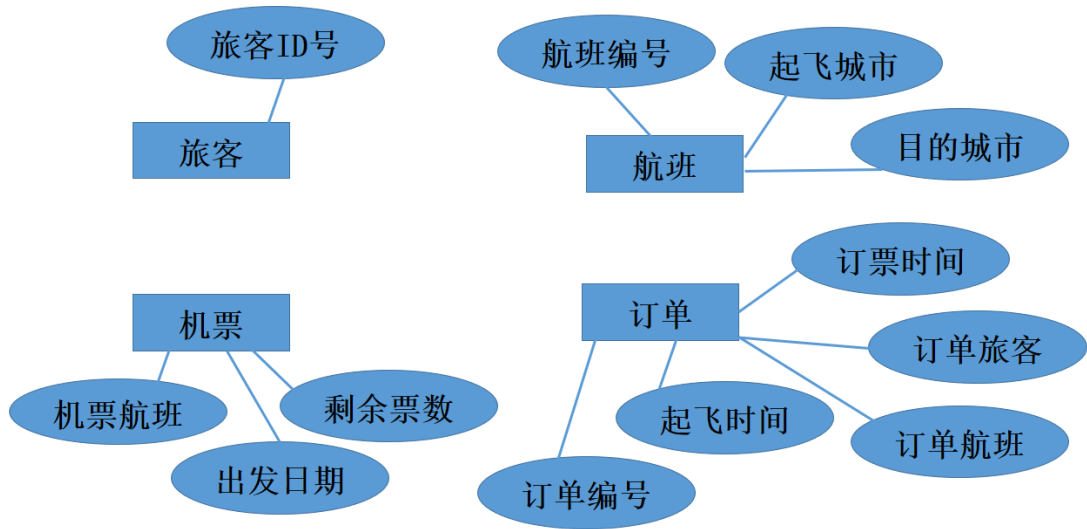


图 3-1-1：实体 ER 图

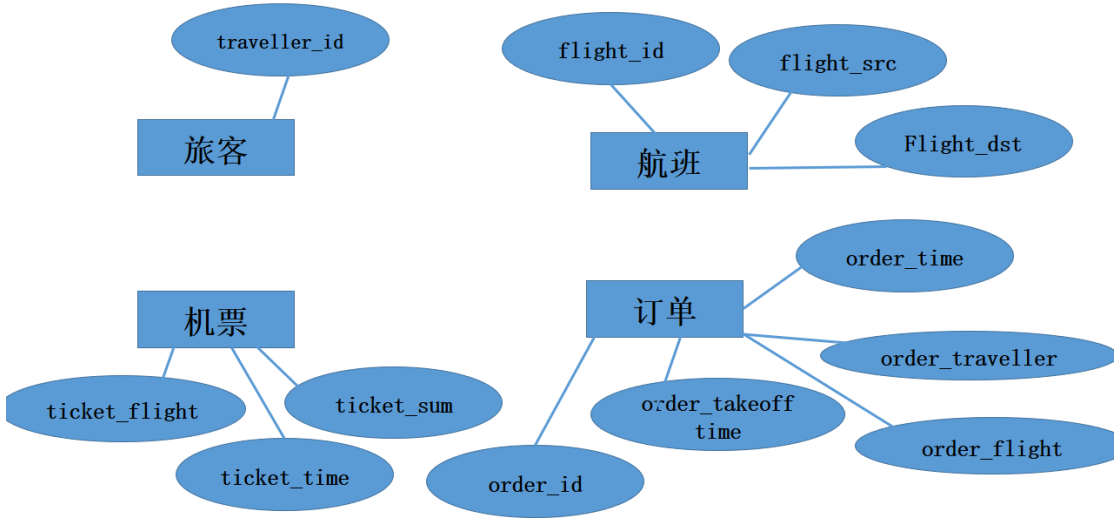


图 3-1-2：ER 图

2、定义数据库表

a. 航班表

DESKTOP-HI3K...订票系统 - dbo.航班			
列名	数据类型	允许 Null	值
flight_id	char(10)	<input type="checkbox"/>	
flight_src	varchar(50)	<input type="checkbox"/>	
flight_dst	varchar(50)	<input type="checkbox"/>	
		<input type="checkbox"/>	

b. 旅客表

DESKTOP-HI3K...订票系统 - dbo.旅客			
	列名	数据类型	允许 Null 值
PK	traveller_id	char(18)	<input type="checkbox"/>
			<input type="checkbox"/>

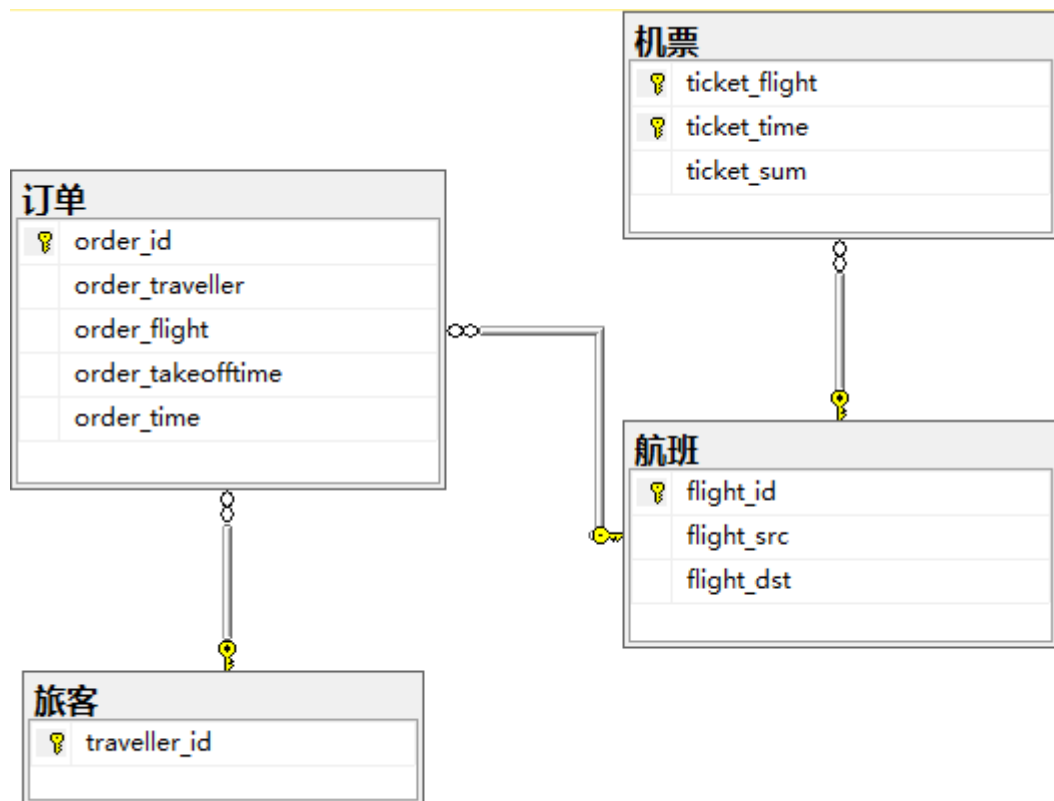
c. 机票表

DESKTOP-HI3K...订票系统 - dbo.机票			
	列名	数据类型	允许 Null 值
PK	ticket_flight	char(10)	<input type="checkbox"/>
	ticket_time	date	<input type="checkbox"/>
	ticket_sum	int	<input type="checkbox"/>
			<input type="checkbox"/>

d. 订单表

DESKTOP-HI3K...订票系统 - dbo.订单			
	列名	数据类型	允许 Null 值
PK	order_id	char(10)	<input type="checkbox"/>
	order_traveller	char(18)	<input type="checkbox"/>
	order_flight	char(10)	<input type="checkbox"/>
	order_takeofftime	date	<input type="checkbox"/>
	order_time	date	<input type="checkbox"/>
			<input type="checkbox"/>

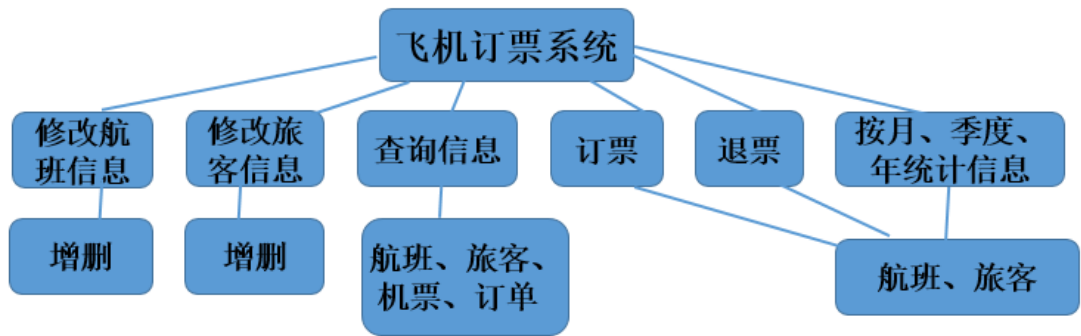
e. 机票订票系统的数据关系图:



在上述关系中，机票表中 `tiket_flight` 为航班表中 `flight_id` 的外键，删除航班表中特定航班时机票表中特定航班的所有机票被删除（级联删除）。订单表中 `order_traveller` 为旅客表中 `traveller_id` 的外键，当删除旅客信息时，订单表中所有该旅客的订单被删除。订单表中 `order_flight` 为航班表中 `flight_id` 的外键，当删除旅客信息时，订单表中所有该旅航班的订单被删除。

当旅客定票时，机票表中的剩余票数会减少，当旅客退票时，机票表中的剩余票数增加，从而实现动态刷新库存。订单表中的 `order_time` 和 `order_takeofftime` 是为了便于按月、按季度、按年份统计航班售票情况和旅客出行情况而设定的。

### 3、软件功能实现



#### 4、相关实现代码说明：

a.主函数里面使用 **switch** 结构，实现相关功能切换，不同命令执行不同的功能函数，当用户键入“quit”时，退出系统：

```

public static void main(String[] args) {
    connect_database();//连接数据库

    System.out.println("请输入对机票订票系统的操作：\n 1、
flight(航班修改)；\n 2、traveller(旅客修改)；" +
        "\n 3、query(查询航班，旅客等信息)；\n 4、
count(统计售票出行)；\n " +
        "5、order(预定机票)；\n 6、refund(退机票)；\n
7、quit(退出系统)；\n");

    Scanner sc=new Scanner(System.in);//System.in 代表标准输入，就是键盘输入

    sc.useDelimiter("\n");//Scanner 默认以空格作为分隔符，
    这里以回车作为分隔符
    while(sc.hasNext())
    {
        int flag=0;
        switch (sc.next())
        {
            case "flight":
                flight();

```

```

        break;
    case "traveller":
        traveller();
        break;
    case "query":
        query();
        break;
    case "order":
        order();
        break;
    case "refund":
        refund();
        break;
    case "count":
        count();
        break;
    case "quit":

        System.out.println("成功退出系统!");

        flag=1;
        break;
    default:

        System.out.println("提示：请重新输入正确的主
菜单命令!");
    }

    if (flag==1)//flag=1,则退出机票订票系统
        break;
}
}

```

## **b.连接数据库的操作:**

```

Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver"); //加载 JDBC 驱动

```

```

        con= DriverManager.getConnection(url); //解析
url,连接数据库

```

```

        if (con!=null)
        {

            System.out.println("连接成功.");
        }
    }
}

```

```

    }
    else
    {

        System.out.println("连接失败！");

    }

```

**c. 预定机票的操作，将旅客、航班、出发日期等信息输入到订单中：**

```

String order_id=getRandomString(10);

        String SQL_order="INSERT INTO 订单
(order_id,order_traveller,order_flight,order_takeofftime,
order_time)" +

"VALUES('"+order_id+"','"+arr[0]+"','"+arr[1]+"','"+arr[2]
]+"','"+arr[2]+"')";
        sql_exe_manipulate(SQL_order);

        String SQL_ticket_minus="UPDATE 机票
SET ticket_sum=ticket_sum-1" +
        "WHERE
ticket_flight='"+arr[1]+" AND ticket_time='"+arr[2]+"';
        sql_exe_manipulate(SQL_ticket_minus);

        System.out.println("预订机票:

"+order_id+"\n");

```

**d. sql\_exe\_manipulate() 和 getRandomString() 分别执行数据库 SQL 的更新，查询操作：**

```

public static void sql_exe_manipulate(String SQL) {
    try {
        Statement statement = con.createStatement();
        statement.executeUpdate(SQL);
    }

    catch (Exception e)
    {
        e.printStackTrace();
    }

}

//生成指定长度的随机数

```



```

        public static String getRandomString(int length) {
            String base =
"abcdefghijklmnopqrstuvwxyz0123456789";

            Random random = new Random();//构造函数

            StringBuffer sb = new StringBuffer();//StringBuffer
对象改变的是自身，在进行字符串处理时不生成新的对象，所以在内存使用上
要优先 String

            for (int i = 0; i < length; i++) {

                int number = random.nextInt(base.length());//生
成[0,base.length)区间的整数

                sb.append(base.charAt(number));
            }
            return sb.toString();
        }

```

#### e. 查询航班信息：

```

        public static void iflight() {
            try
            {

                String SQL="SELECT flight_id AS 航班编
号,flight_src AS 起飞城市,flight_dst AS 目标城市 FROM 航班";

                ResultSet resultSet=sql_exe_query(SQL);

                System.out.println(" 航班编号    起飞城市    目的城市
");

                while (resultSet.next()) {
                    System.out.println(resultSet.getString(1) +
"    " + resultSet.getString(2)+"    "
                    + resultSet.getString(3));
                }

            }
            catch (Exception e)
            {
                e.printStackTrace();
            }

        }

```

#### f. 按季度统计旅客出行情况

```
public static void count_traveller_season(String
traveller_id) {

    System.out.println("请输入你要查询的季度，例：2016 年春/
夏/秋/冬"2016 spring/summer/autumn/winter"\n");

    Scanner scanner = new Scanner(System.in);
    scanner.useDelimiter("\n");
    String season_s = scanner.next();
    String[] arr = season_s.split(" ");
    String time_start = arr[0] + "-01-01";
    String time_stop = arr[0] + "-03-31";
    switch (arr[1])
    {
        case "spring":
            time_start = arr[0] + "-01-01";
            time_stop = arr[0] + "-03-31";
            break;
        case "summer":
            time_start = arr[0] + "-04-01";
            time_stop = arr[0] + "-06-30";
            break;
        case "autumn":
            time_start = arr[0] + "-07-01";
            time_stop = arr[0] + "-09-30";
            break;
        case "winter":
            time_start = arr[0] + "-10-01";
            time_stop = arr[0] + "-12-31";
            break;
    }
    try {
        String SQL_season = "SELECT COUNT(*)" +
            "FROM 订单" +
            " WHERE order_traveller='" + traveller_id
+ "'" +
            " AND order_takeofftime >='" + time_start
+ "'" AND order_takeofftime <='" +time_stop + "'";
        ResultSet season_resultSet =
sql_exe_query(SQL_season);
        while (season_resultSet.next()) {
```

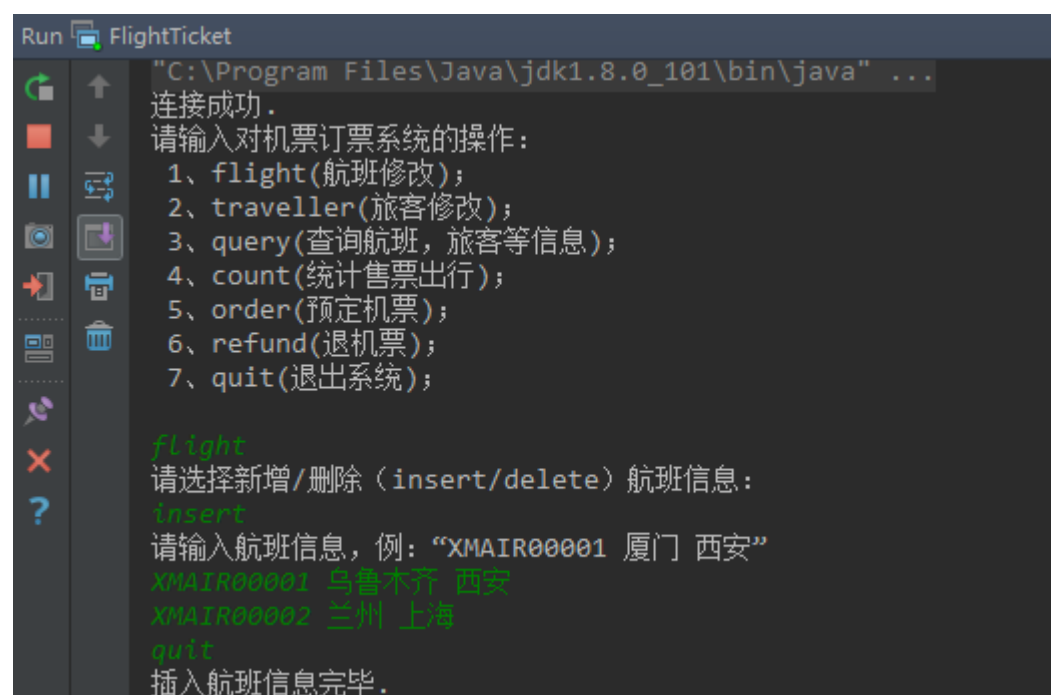
```

        System.out.println(traveller_id + "旅客" +
arr[0] + "年" + arr[1] + "季度共计出行： "
        + season_resultSet.getString(1) + " 次
");
    }
} catch (Exception e) {
    e.printStackTrace();
}
}
}

```

#### 四、 实验系统演示：

##### 1、 增加、删除航班：



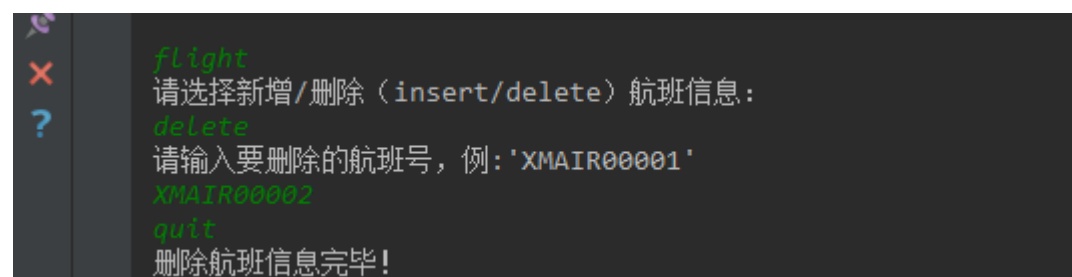
```

Run FlightTicket
"C:\Program Files\Java\jdk1.8.0_101\bin\java" ...
连接成功。
请输入对机票订票系统的操作：
1、flight(航班修改);
2、traveller(旅客修改);
3、query(查询航班，旅客等信息);
4、count(统计售票出行);
5、order(预定机票);
6、refund(退机票);
7、quit(退出系统);

flight
请选择新增/删除 (insert/delete) 航班信息：
insert
请输入航班信息，例："XMAIR00001 厦门 西安"
XMAIR00001 乌鲁木齐 西安
XMAIR00002 兰州 上海
quit
插入航班信息完毕。

```

图 4-1-1：插入航班的命令



```

flight
请选择新增/删除 (insert/delete) 航班信息：
delete
请输入要删除的航班号，例：'XMAIR00001'
XMAIR00002
quit
删除航班信息完毕！

```

图 4-1-2：删除航班的命令

	flight id	flight src	flight dst
▶	XMAIR00003	厦门	福州
	XMAIR00004	深圳	广州
	XMAIR00005	西安	天津
	XMAIR00006	广州	昆明
	XMAIR00007	海口	香港
	XMAIR00008	长沙	武汉
	XMAIR09033	郑州	西宁
	XMAIR32434	长沙	宁波
	XMAIR34234	郑州	南昌
	XMAIR38945	青岛	大连
*	NULL	NULL	NULL

4-1-3: 未插入航班前

	flight id	flight src	flight dst
▶	XMAIR00001	乌鲁木齐	西安
	XMAIR00002	兰州	上海
	XMAIR00003	厦门	福州
	XMAIR00004	深圳	广州
	XMAIR00005	西安	天津
	XMAIR00006	广州	昆明
	XMAIR00007	海口	香港
	XMAIR00008	长沙	武汉
	XMAIR09033	郑州	西宁
	XMAIR32434	长沙	宁波
	XMAIR34234	郑州	南昌
	XMAIR38945	青岛	大连
*	NULL	NULL	NULL

4-1-4: 新增航班后

## 2、 旅客注册:

```

traveller
请选择新增/删除 (insert/delete) 旅客信息:
insert
请输入旅客信息 (18位身份证号), 例: "62089819940312344X"
622826199505241535
622826199505241534
622826199505241533
quit
新增旅客信息完毕.

```

图 4-2-1: 增加旅客信息

```

traveller
请选择新增/删除 (insert/delete) 旅客信息:
delete
请输入要删除的旅客身份证号, 例: '620826199006211535'
622826199505241533
quit
删除旅客信息完毕!

```

图 4-2-2: 删除旅客信息

	traveller id
	622826199505241533
	622826199505241534
TOP-HI	6228261995052415350.1601.5 - sakaluwa)

图 4-2-3: 增加旅客后的旅客表

3、 查询信息：

```
query
请输入您要查询的信息：
1、 iflight(航班信息);
2、 itraveller(旅客信息);
3、 iticket(余票信息);
4、 iorder(订单信息);
5、 quit(退出查询);

iflight
  航班编号  起飞城市  目的城市
XMAIR00001  乌鲁木齐  西安
XMAIR00003  厦门      福州
XMAIR00004  深圳      广州
XMAIR00005  西安      天津
XMAIR00006  广州      昆明
XMAIR00007  海口      香港
XMAIR00008  长沙      武汉
XMAIR09033  郑州      西宁
XMAIR32434  长沙      宁波
XMAIR34234  郑州      南昌
XMAIR38945  青岛      大连
```

图 4-3-1： 查询航班信息

```
itraveller
  旅客 ID 号
622826199505241534
622826199505241535
```

图 4-3-2： 查询旅客信息

```
iticket
  航班机票  起飞时间  剩余机票
XMAIR00005  2016-12-01  200
XMAIR00006  2016-11-11  200
XMAIR00006  2016-11-25  200
XMAIR00006  2016-12-22  200
XMAIR00008  2016-01-22  200
XMAIR00008  2016-02-12  200
XMAIR00008  2016-04-09  200
XMAIR00008  2016-06-18  200
XMAIR00008  2016-08-15  200
XMAIR00008  2016-09-23  200
XMAIR00008  2016-10-22  199
XMAIR00008  2016-12-22  200
XMAIR09033  2017-01-15  200
```

图 4-3-3： 查询机票信息

order	订单编号	订单旅客	订单航班	起飞时间	订票时间
-------	------	------	------	------	------

图 4-3-4: 查询订单信息

4、订票:



图 4-4-1: 机票预订成功

	order id	order traveller	order flight	order takeof...	order time
▶	9ldy87ftvq	62282619950...	XMAIR00008	2016-09-23	2016-09-23
	ajbp3osggr	62282619950...	XMAIR00008	2016-12-22	2016-12-22
	bqctm75huw	62282619950...	XMAIR00005	2016-12-01	2016-12-01
	e38n36bsz0	62282619950...	XMAIR00008	2016-08-15	2016-08-15
	ncvxzbpps3	62282619950...	XMAIR00008	2016-04-09	2016-04-09
	pnz28msi28	62282619950...	XMAIR00008	2016-06-18	2016-06-18
	rlcs6dbmtj	62282619950...	XMAIR00008	2016-01-22	2016-01-22
	v0fpavh2md	62282619950...	XMAIR00008	2016-10-22	2016-10-22
	wepygjtj5ba	62282619950...	XMAIR00008	2016-02-12	2016-02-12
*	NULL	NULL	NULL	NULL	NULL

图 4-4-2: 预订机票生成的订单

	ticket flight	ticket time	ticket sum
	XMAIR00005	2016-12-01	199
	XMAIR00006	2016-11-11	200
	XMAIR00006	2016-11-25	200
	XMAIR00006	2016-12-22	200
	XMAIR00008	2016-01-22	199
	XMAIR00008	2016-02-12	199
	XMAIR00008	2016-04-09	199
	XMAIR00008	2016-06-18	199
	XMAIR00008	2016-08-15	199
	XMAIR00008	2016-09-23	199
	XMAIR00008	2016-10-22	199
	XMAIR00008	2016-12-22	199
	XMAIR09033	2017-01-15	200
*	NULL	NULL	NULL

图 4-4-3: 机票库存动态刷新

## 5、 退票:

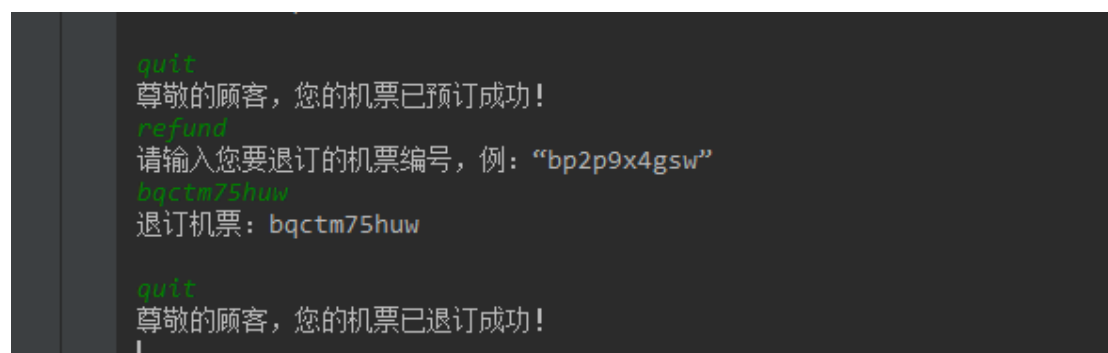


图 4-5-1: 退票

	order id	order traveller	order flight	order takeof...	order time
▶	9ldy87ftvq	62282619950...	XMAIR00008	2016-09-23	2016-09-23
	ajbp3osggr	62282619950...	XMAIR00008	2016-12-22	2016-12-22
	e38n36bsz0	62282619950...	XMAIR00008	2016-08-15	2016-08-15
	ncvxzbbps3	62282619950...	XMAIR00008	2016-04-09	2016-04-09
	pnz28msi28	62282619950...	XMAIR00008	2016-06-18	2016-06-18
	rlcs6dbmtty	62282619950...	XMAIR00008	2016-01-22	2016-01-22
	v0fpavh2md	62282619950...	XMAIR00008	2016-10-22	2016-10-22
	wepygjtj5ba	62282619950...	XMAIR00008	2016-02-12	2016-02-12
*	NULL	NULL	NULL	NULL	NULL

图 4-5-2: 相应订单被删除

	ticket flight	ticket time	ticket sum
▶	XMAIR00005	2016-12-01	200
	XMAIR00006	2016-11-11	200
	XMAIR00006	2016-11-25	200
	XMAIR00006	2016-12-22	200
	XMAIR00008	2016-01-22	199
	XMAIR00008	2016-02-12	199
	XMAIR00008	2016-04-09	199
	XMAIR00008	2016-06-18	199
	XMAIR00008	2016-08-15	199
	XMAIR00008	2016-09-23	199
	XMAIR00008	2016-10-22	199
	XMAIR00008	2016-12-22	199
	XMAIR09033	2017-01-15	200
*	NULL	NULL	NULL

图 4-5-3：机票库存被刷新

#### 6、统计航班售票情况：

```
count
请输入您要统计的信息：
    航班（月/季度/年）：flight month/season/year;
    旅客（月/季度/年）：traveller month/season/year;

flight month
请输入您要查询的航班编号：
XMAIR00008
请输入您要查询的月份，例：2016年11月份“2016 11”

2016 10
XMAIR00008航班2016年10月共计售票： 1 张
```

图 4-6-1：航班 XMAIR00008 在 2016 年 11 月售票 1 张

```
flight season
请输入您要查询的航班编号：
XMAIR00008
请输入您要查询的季度，例：2016年春/夏/秋/冬“2016 spring/summer/autumn/winter”

2016 winter
XMAIR00008航班2016年winter季度共计售票： 2 张
```

图 4-6-2：航班 XMAIR00008 在 2016 年冬季售票 2 张

```
flight year
请输入您要查询的航班编号：
XMAIR00008
请输入您要查询的年份，例：2016年“2016”

2016
XMAIR00008航班2016年共计售票： 8 张
```



图 4-6-3：航班 XMAIR00008 在 2016 年售票 8 张

7、 旅客出行情况统计：

```
count
请输入您要统计的信息：
    航班（月/季度/年）：flight month/season/year;
    旅客（月/季度/年）：traveller month/season/year;

traveller month
请输入您要查询的旅客ID号：
622826199505241535
请输入您要查询的月份，例：2016年11月份“2016 11”

2016 12
622826199505241535旅客2016年12月共计出行： 1 次
```

图 4-7-1：旅客 622826199505241535 在 2016 年 12 月出行 1 次

```
traveller season
请输入您要查询的旅客ID号：
622826199505241535
请输入您要查询的季度，例：2016年春/夏/秋/冬“2016 spring/summer/autumn/winter”

2016 autumn
622826199505241535旅客2016年autumn季度共计出行： 2 次
```

图 4-7-2：旅客 622826199505241535 在 2016 年秋季出行 2 次

```
traveller year
请输入您要查询的旅客ID号：
622826199505241535
请输入您要查询的年份，例：2016年“2016”

2016
622826199505241535旅客2016年共计出行： 8 次
```

图 4-7-3：旅客 622826199505241535 在 2016 年出行 8 次

8、 退出系统：

```
quit
成功退出系统！

Process finished with exit code 0
```

图 4-8-1：quit 命令退出机票订票系统

## 五、 实验总结：

1、以前的许多程序实验都是直接从程序中调用数据，用链表数组等将其存储。像这样连接数据库系统将数据和程序如此分离的程序实验，我还是第一次做，不过其基本功能类似。

2、本系统实现的功能有：航班信息的修改，旅客信息的修改，订票、退票功能，信息查询功能，售票出行情况统计功能。

3、该系统也存在很大的不足之处，由于我对于 GUI 编程并不是非常熟悉，所以采用了命令行的方式来实现交互，使得系统不好用。另一方面，该系统的容错能力也非常有限，很多情况下并没有对用户输入做检查，有时候可能导致秩序崩溃。

4、总之，这次数据库实验历时一周，数据库结构的设计、数据的填充，提高了我系统设计的能力，软件功能的实现也让我的编程能力得到了锻炼。对于开发一个系统也有了初步的认识，以后还要多写程序多实践来提高自己的编程和设计能力；