



华中科技大学

数据库系统原理实践报告

综合设计题目： 航班信息管理系统

姓 名： 潘翔

学 院： 计算机科学与技术学院

专 业： 物联网工程

班 级： IOT1601

学 号： U20161489

指导教师： 袁平鹏

分数	
教师签名	

2018 年 5 月 12 日

目 录

1 课程任务概述.....	1
2 软件功能学习部分.....	2
2.1 任务要求.....	2
2.2 完成过程.....	2
2.2.1 环境配置.....	2
2.2.2 添加用户并授权.....	2
2.2.3 数据备份与恢复.....	3
2.3 任务总结.....	4
3 Sql 练习部分.....	5
3.1 任务要求.....	5
3.2 完成过程.....	6
3.2.1 创建表格.....	6
3.2.2 插入数据.....	8
3.2.3 数据更新.....	9
3.2.4 数据查询.....	12
3.2.5 了解系统的查询性能分析功能（选做）.....	20
3.2.6 DBMS 函数及存储过程和事务（选做）.....	21
3.3 任务总结.....	22
4 数据库应用系统设计.....	24
4.1 系统设计目标.....	24
4.1.1 系统功能的基本要求：.....	24
4.1.2 数据库要求：.....	24
4.2 需求分析.....	25
4.3 总体设计.....	25
4.3.1 系统功能模块划分.....	25
4.3.2 系统架构方案.....	25
4.4 数据库设计.....	26
4.4.1 系统操作流程圖.....	26
4.4.2 系统操作表格圖.....	27

4.4.3 E-R 图.....	27
4.4.4 数据流图.....	28
4.4.5 数据库逻辑结构设计.....	28
4.4.6 数据表格设计.....	30
4.5 详细设计与实现.....	34
4.5.1 开发环境与运行环境.....	34
4.5.2 航班事务管理.....	34
4.5.3 订单事务管理.....	36
4.5.4 权限系统.....	37
4.5.5 报表系统.....	37
4.5.6 通用数据显示模块.....	38
4.6 系统测试.....	41
4.6.1 航班事务测试.....	41
4.6.2 订单事务测试.....	44
4.6.3 报表测试.....	48
4.7 系统设计与实现总结.....	49
5 课程总结.....	50
附录.....	51
文件说明.....	51
Main.....	52
Mysqlquerymodel.....	54
Addflight_dialog.....	61
Chooseseat_dialog.....	73
Login_dialog.....	84
Mainwindow.....	90
Print_dialog.....	124
Report_dialog.....	133

1 课程任务概述

软件功能学习部分：练习 DBMS 软件的使用

SQL 联系部分：练习 SQL 语句的使用

数据库系应用系统设计：自行选择所擅长的 DBMS 软件以及数据库应用系统（客户端程序或者网站）的程序开发工具，参考后面的题目例子，拟定一个自己感兴趣的数据库应用系统题目，完成该小型数据库应用系统的设计与实现工作。主要包括：需求调研与分析、总体设计、数据库设计、详细设计与实现、测试等环节的工作。

2 软件功能学习部分

2.1 任务要求

完成下列 1~2 题，并在实践报告中叙述过程，可适当辅以插图（控制在 A4 三页篇幅以内）

- 1) 练习 sqlserver 的两种完全备份方式：数据和日志文件的脱机备份、系统的备份功能。
- 2) 练习在新增的数据库上增加用户并配置权限的操作。

2.2 完成过程

2.2.1 环境配置

- 1)操作系统：Arch Linux x64
- 2)数据库：MariaDB
- 3)可视化工具：MySQL-workbench
- 4)安装过程：

```
yaourt -S mariadb  
systemctl start mariadb.service
```

2.2.2 添加用户并授权

- 1)创建用户
`CREATE USER 'hover'@'localhost' IDENTIFIED BY '123456';`
- 2)创建数据库
`CREATE DATABASE DBlabs;`
- 3)用户授权
`GRANT ALL ON DBlabs.* TO 'hover'@'localhost';`
- 4)查看当前所有的用户
`SELECT User, Host, Password FROM mysql.user;`

```
wings@hover ~/MY/myBlog master mycli -u root
Password:
Version: 1.16.0
Chat: https://gitter.im/dbcli/mycli
Mail: https://groups.google.com/forum/#!forum/mycli-users
Home: http://mycli.net
Thanks to the contributor - Steve Robbins
mariadb root@localhost:(none)> SELECT User, Host, Password FROM mysql.user;
+-----+-----+-----+
| User | Host | Password |
+-----+-----+-----+
| root | localhost | *30F8DB72BF9C809EFCDBCF6C46733B5390942250 |
| root | 127.0.0.1 | |
| root | ::1 | |
| hover | localhost | *6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9 |
+-----+-----+-----+
5 rows in set
Time: 0.017s
mariadb root@localhost:(none)>
```

图 2-1 添加用户情况

Tables_in_DBlabs	
80S	Table - Picture -
ACTIN	
ACTOR	10T1601_98
Danda	
FILM	
FORTEST	
SHOW	
THEATER	
import_table	

图 2-2 显示表格

2.2.3 数据备份与恢复

1) 逻辑备份 mysqldump

使用逻辑备份，备份恢复所需的 SQL

`mysqldump -u hover -p --databases DBlabs > DBlabsBack.sql`

```

DROP TABLE IF EXISTS `80S`;
/*!50001 DROP VIEW IF EXISTS `80S`*/;
SET @saved_cs_client      = @@character_set_client;
SET character_set_client = utf8;
/*!50001 CREATE TABLE `80S` (
  `ACTID` tinyint NOT NULL,
  `ISLEADING` tinyint NOT NULL,
  `FID` tinyint NOT NULL,
  `LEADING NUM` tinyint NOT NULL,
  `MAXGRADE` tinyint NOT NULL
) ENGINE=MyISAM */;
SET character_set_client = @saved_cs_client;

--
-- Table structure for table `ACTIN`
--

DROP TABLE IF EXISTS `ACTIN`;
/*!40101 SET @saved_cs_client      = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `ACTIN` (
  `ACTID` int(11) NOT NULL,
  `FID` int(11) DEFAULT NULL,
  `ISLEADING` char(1) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `GRADE` int(11) DEFAULT NULL,
  KEY `FID` (`FID`),
  KEY `ACTID` (`ACTID`),
  CONSTRAINT `ACTIN_ibfk_2` FOREIGN KEY (`FID`) REFERENCES `FILM` (`FID`),
  CONSTRAINT `ACTIN_ibfk_3` FOREIGN KEY (`ACTID`) REFERENCES `ACTOR` (`ACTID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

```

图 2-3 操作文件结果

2)物理备份 mysqlhotcopy

`yaourt -S perl-dbi`

`yaourt -S perl-dbd-mysql`

```

wingsheover@usr/lib/mysql: sudo mysqlhotcopy -u root -p 990123 DBlabs --addtodest /opt/sql_back
Flushed 8 tables with read lock ('DBlabs`.`ACTIN`, 'DBlabs`.`ACTOR`, 'DBlabs`.`FILM`, 'DBlabs`.`FORTEST`, 'DBlabs`.`SHOW`, 'DBlabs`.`THEATER`, 'DBlabs`.`co
untFilm`, 'DBlabs`.`import_table`) in 0 seconds.
Locked 2 views ('DBlabs`.`80S` READ, 'DBlabs`.`Danda`) in 0 seconds.
Copying 21 files...
Copying indices for 0 files...
Unlocked tables.
mysqlhotcopy copied 10 tables (21 files) in 0 seconds (0 seconds overall).

```

图 2-4 mysqlhotcopy 操作结果

2.3 任务总结

1. hotcopy 过程需要 reload 权限，故使用 root 账户进行操作
2. 环境过程中使用 mysql 的开源实现 mariadb 和 mysqlworkbench 实现，使用 mycli 进行命令行操作
3. 学习了不同的数据库保存策略，在数据库的维护过程中，可以结合定时计划和 trigger 进行数据库的定时备份

3 Sql 练习部分

3.1 任务要求

1) 创建下列跟电影相关的关系，包括主码和外码的说明

电影表【电影编号，电影名称，电影类型，导演姓名，电影时长（以分钟计），是否 3D，用户评分】

FILM(FID int, FNAME char(30), FTYPE char(10), DNAME char(30), length int, IS3D char(1), GRADE int)。

主码为电影编号，IS3D 取值为'Y'表示是 3D 电影，'N'表示不是，用户评分规定为 0~100 分之间或者为空值。

演员表【演员编号，演员姓名，性别，出生年份】

ACTOR(ACTID int, ANAME char(30), SEX char(2), BYEAR int)

主码为演员编号

参演表【演员编号，电影编号，是否主角，用户对该演员在该电影中的评分】

ACTIN(ACTID int, FID int, ISLEADING char(1), GRADE int)

主码、外码请依据应用背景合理定义。ISLEADING 取值为'Y'表示是，'N'表示不是主角，也可能取空值，表示不太确定该演员在该电影中是否主角。GRADE 规定为 0~100 分之间或者为空值。

电影院表【电影院编号，电影院名字，影院所在行政区，影院地址】

THEATER (TID int, TNAME char(20), TAREA char(20), ADDRESS char(30))

主码为电影院编号，影院所在行政区取值如“洪山区”、“武昌区”等等。

上映表【电影编号，影院编号，上映年份，上映月份】

SHOW(FID int, TID int, PRICE int, YEAR int, MONTH int)

假定一部电影在一家影院只上映一次，主码、外码请依据应用背景合理定义。

2) 观察性实验

验证在建立外码时是否一定要参考被参照关系的主码，并在实验报告中简述过程和结果。

3) 数据准备

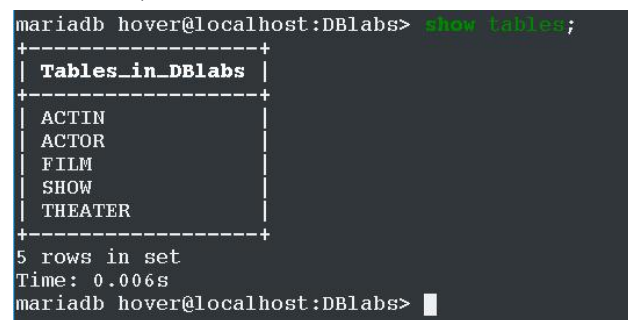
依据后续实验的要求，向上述表格中录入适当数量的实验数据，从而对相关的实验任务能够起到验证的作用。

3.2 完成过程

3.2.1 创建表格

1)创建表格

```
CREATE TABLE FILM(FID int, FNAME char(30), FTYPE char(10), DNAME
char(30), length int, IS3D char(1), GRADE int);
CREATE TABLE ACTOR(ACTID int, ANAME char(30), SEX char(2), BYEAR
int);
CREATE TABLE ACTIN(ACTID int, FID int, ISLEADING char(1), GRADE int);
CREATE TABLE THEATER(TID int, TNAME char(20), TAREA char(20),
ADDRESS char(30));
CREATE TABLE `SHOW`(FID int, TID int , PRICE int, YEAR int , MONTH int);
```



```
mariadb hover@localhost:DBlabs> show tables;
+-----+
| Tables_in_DBlabs |
+-----+
| ACTIN             |
| ACTOR             |
| FILM              |
| SHOW              |
| THEATER           |
+-----+
5 rows in set
Time: 0.006s
mariadb hover@localhost:DBlabs> █
```

图 3.1 显示表格创建结果


2)定义主键

```
ALTER TABLE `FILM` ADD PRIMARY KEY (FID);
ALTER TABLE `ACTOR` ADD PRIMARY KEY (ACTID);
ALTER TABLE `ACTIN` ADD PRIMARY KEY (ACTID);
ALTER TABLE `THEATER` ADD PRIMARY KEY (TID);
ALTER TABLE `SHOW` ADD PRIMARY KEY (FID);
```

使用

```
SELECT
t.TABLE_NAME,
c.COLUMN_NAME
FROM
INFORMATION_SCHEMA.TABLE_CONSTRAINTS AS t,
information_schema.TABLES AS ts,
```

```
information_schema.KEY_COLUMN_USAGE As
c
WHERE
t.TABLE_NAME = ts.TABLE_NAME
AND ts.TABLE_NAME = c.TABLE_NAME
AND t.CONSTRAINT_TYPE = 'PRIMARY KEY'
```



```
mariadb hover@localhost:DBlabs>
->
-> SELECT
-> t.TABLE_NAME,
-> c.COLUMN_NAME
-> FROM
-> information_schema.TABLE_CONSTRAINTS AS t,
-> information_schema.TABLES AS ts,
-> information_schema.KEY_COLUMN_USAGE AS c
-> WHERE
-> t.TABLE_NAME = ts.TABLE_NAME
-> AND ts.TABLE_NAME = c.TABLE_NAME
-> AND t.CONSTRAINT_TYPE = 'PRIMARY KEY'
```

TABLE_NAME	COLUMN_NAME
ACTIN	ACTID
ACTOR	ACTID
FILM	FID
SHOW	FID
THEATER	TID

图 3.2 显示主键结果

3)定义外键

ACTIN 外键:

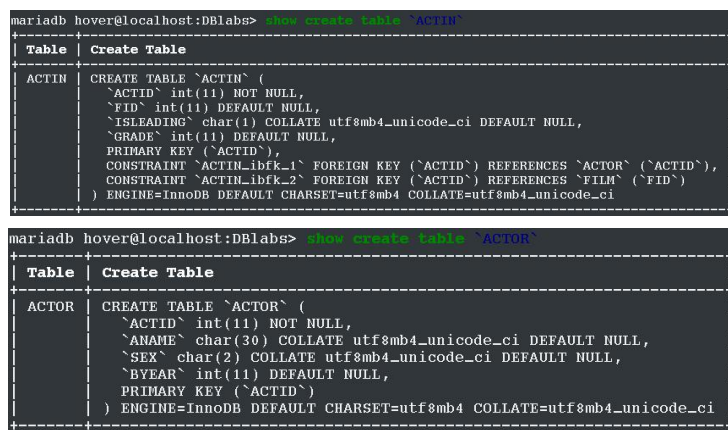
```
ALTER TABLE ACTIN ADD FOREIGN KEY (ACTID) REFERENCES ACTOR
(ACTID);
ALTER TABLE ACTIN ADD FOREIGN KEY (FID) REFERENCES FILM(FID);
```

SHOW 外键:

```
ALTER TABLE `SHOW` ADD FOREIGN KEY (FID) REFERENCES FILM(FID);
ALTER TABLE `SHOW` ADD FOREIGN KEY (TID) REFERENCES THEATER
(TID);
```

使用

show create table (表名)查看外键



```
mariadb hover@localhost:DBlabs> show create table `ACTIN`
```

Table	Create Table
ACTIN	CREATE TABLE `ACTIN` (`ACTID` int(11) NOT NULL, `FID` int(11) DEFAULT NULL, `ISLEADING` char(1) COLLATE utf8mb4_unicode_ci DEFAULT NULL, `GRADE` int(11) DEFAULT NULL, PRIMARY KEY (`ACTID`), CONSTRAINT `ACTIN_ibfk_1` FOREIGN KEY (`ACTID`) REFERENCES `ACTOR` (`ACTID`), CONSTRAINT `ACTIN_ibfk_2` FOREIGN KEY (`ACTID`) REFERENCES `FILM` (`FID`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci

```
mariadb hover@localhost:DBlabs> show create table `ACTOR`
```

Table	Create Table
ACTOR	CREATE TABLE `ACTOR` (`ACTID` int(11) NOT NULL, `ANAME` char(30) COLLATE utf8mb4_unicode_ci DEFAULT NULL, `SEX` char(2) COLLATE utf8mb4_unicode_ci DEFAULT NULL, `BYEAR` int(11) DEFAULT NULL, PRIMARY KEY (`ACTID`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci

mariadb	hover@localhost:DBlabs>	show create table `ACTOR`
Table	Create Table	
ACTOR	CREATE TABLE `ACTOR` (`ACTID` int(11) NOT NULL, `ANAME` char(30) COLLATE utf8mb4_unicode_ci DEFAULT NULL, `SEX` char(2) COLLATE utf8mb4_unicode_ci DEFAULT NULL, `BYEAR` int(11) DEFAULT NULL, PRIMARY KEY (`ACTID`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci	

mariadb	hover@localhost:DBlabs>	show create table `THEATER`
Table	Create Table	
THEATER	CREATE TABLE `THEATER` (`TID` int(11) NOT NULL, `TNAME` char(20) COLLATE utf8mb4_unicode_ci DEFAULT NULL, `TAREA` char(20) COLLATE utf8mb4_unicode_ci DEFAULT NULL, `ADDRESS` char(30) COLLATE utf8mb4_unicode_ci DEFAULT NULL, PRIMARY KEY (`TID`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci	

Table	Create Table	
SHOW	CREATE TABLE `SHOW` (`FID` int(11) NOT NULL, `TID` int(11) DEFAULT NULL, `PRICE` int(11) DEFAULT NULL, `YEAR` int(11) DEFAULT NULL, `MONTH` int(11) DEFAULT NULL, KEY `TID` (`TID`), KEY `FID` (`FID`), CONSTRAINT `SHOW_ibfk_2` FOREIGN KEY (`TID`) REFERENCES `THEATER` (`TID`), CONSTRAINT `SHOW_ibfk_3` FOREIGN KEY (`FID`) REFERENCES `FILM` (`FID`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci	

图 3.3 定义外键结果

3.2.2 插入数据

INSERT INTO FILM(FID , FNAME , FTYPE , DNAME , length , IS3D , GRADE)
VALUES(4, 'film4', 'Adventure' , 'NH3' ,122, 'Y', 93)

```
mariadb hover@localhost:DBlabs> select * from `FILM`;
```

FID	FNAME	FTYPE	DNAME	length	IS3D	GRADE
1	film1	Action	Amazon	120	Y	90
2	film2	Love	Gray	119	N	91
3	film3	Suspence	Hover	121	Y	92

图 3.4 FILM 插入结果

INSERT INTO ACTOR(ACTID, ANAME, SEX, BYEAR)
VALUES(4, 'Perch', 'M', 2005)

ACTID	ANAME	SEX	BYEAR
1	Petter	M	1999
2	Bob	M	1980
3	Angla	F	1970
4	Perch	M	2005
5	孙维正	M	1996
6	徐光磊	M	1998
7	倪鸿	M	1997
8	沁苑翔	M	1999
9	梦桐	F	1997

图 3.5 Actor 插入结果

TID	TNAME	TAREA	ADDRESS
1	洪山区 1	洪山区	洪山区 1号
2	洪山区 2	洪山区	洪山区 2号
3	武昌区 1	武昌区	武昌区 1号
4	武昌区 2	武昌区	武昌区 2号

图 3.6 Theater 插入结果

FID	TID	PRICE	YEAR	MONTH
4	1	35	2017	3
4	2	35	2017	1
4	3	35	2017	2
4	4	35	2017	1
4	5	10	2018	1

图 3.7 Show 插入结果

3.2.3 数据更新

1)分别用一条 sql 语句完成对电影表基本的增、删、改的操作;

INSERT INTO FILM(FID , FNAME , FTYPE , DNAME , length , IS3D , GRADE)
VALUES(4, 'film4', 'Adventure' , 'NH3' ,122, 'Y', 93)

```
mariadb hover@localhost:DBlabs> select * from `FILM`
```

FID	FNAME	FTYPE	DNAME	length	IS3D	GRADE
1	film1	Action	Amazon	120	Y	90
2	film2	Love	Gray	119	N	91
3	film3	Suspence	Hover	121	Y	92
4	film4	Adventure	NH3	122	Y	93

图 3.8 sql 插入

update `FILM` set `IS3D`='N' where `FID` = 4;

FID	FNAME	FTYPE	DNAME	length	IS3D	GRADE
1	film1	Action	Amazon	120	Y	90
2	film2	Love	Gray	119	N	91
3	film3	Suspence	Hover	121	Y	92
4	film4	Adventure	NH3	122	N	93

图 3.9 sql 修改

DELETE FROM `FILM` WHERE FID = 4;

```
mariadb hover@localhost:DBlabs> select * from `FILM`
```

FID	FNAME	FTYPE	DNAME	length	IS3D	GRADE
1	film1	Action	Amazon	120	Y	90
2	film2	Love	Gray	119	N	91
3	film3	Suspence	Hover	121	Y	92

图 3.10 sql 删除

2)批处理操作:将演员表中的 90 后演员记录插入到一个新表 YOUNG_ACTOR 中)

3)数据导入导出:通过查阅 DBMS 资料学习数据导入导出功能，并将任务 2.1 所建表格的数据导出到操作系统文件，然后再将这些文件的数据导入到相应空表。

#	FID	FNAME	FTYPE	DNAME	length	IS3D	GRADE
1	1	film1	Action	Amazon	120	Y	90
2	2	film2	Love	Gray	119	N	91
3	3	film3	Suspence	Hover	121	Y	92
4	4	film4	Adventure	NH3	122	N	93

图 3.11 操作界面 import/export

	A	B	C	D	E	F	G
1	FID	FNAME	FTYPE	DNAME	length	IS3D	GRADE
2	1	film1	Action	Amazon	120	Y	90
3	2	film2	Love	Gray	119	N	91
4	3	film3	Suspence	Hover	121	Y	92
5	4	film4	Adventure	NH3	122	N	93

图 3.12 导出 csv 文件

create table import_table like `FILM`

```
1 • SELECT * FROM DBlabs.import_table;
```

#	FID	FNAME	FTYPE	DNAME	length	IS3D	GRADE
1	0	FNAME	FTYPE	DNAME	0	I	0
2	1	film1	Action	Amazon	120	Y	90
3	2	film2	Love	Gray	119	N	91
4	3	film3	Susp...	Hover	121	Y	92
5	4	film4	Adve...	NH3	122	N	93

图 3.13 导入至新表

4)观察性实验: 建立一个关系，但是不设置主码，然后向该关系中插入重复元组，然后观察在图形化交互界面中对已有数据进行删除和修改时所发生的现象。

ID	DATA
1	1
1	1
1	1
2	2

图 3.14 重复元素插入结果

update `FORTEST` set id=3 where id=1;

ID	DATA
3	1
3	1
3	1
2	2

图 3.15 重复元组修改

`delete from 'FORTEST' where id=3;`

ID	DATA
2	2

图 3.16 重复元组删除

5)创建视图：创建一个有 80 后演员作主角的参演记录视图，其中的属性包括：演员编号、演员姓名、出生年份、作为主角参演的电影数量、这些电影的用户评分的最高分。

```
create view '80S' as
select 'ACTID','ISLEADING','FID',count(*) as 'LEADING NUM',max('GRADE')
as 'MAXGRADE'
from 'ACTIN'
WHERE 'FID' in
(select 'FID' from 'ACTOR'
join 'ACTIN'
on 'ACTIN'.'ACTID'='ACTOR'.'ACTID'
where 1980<='BYEAR' and 'BYEAR'<=1989 and 'ISLEADING'='Y'
group by 'FID')
group by 'ACTID';
```

ACTID	ISLEADING	FID	LEADING NUM	MAXGRADE
1	Y	4	1	90
2	N	4	0	90
3	Y	4	1	90
4	Y	4	1	90
5	Y	5	1	96
6	Y	5	1	97
7	Y	5	1	98
8	Y	5	1	96
13	Y	4	1	90
14	Y	4	2	90

图 3.17 视图创建结果

6)触发器实验：编写一个触发器，用于实现对电影表的完整性控制规则：当增加一部电影时，若导演的姓名为周星驰，则电影类型自动设置为“喜剧”。

```
create trigger StephenChowComedy
after insert on `FILM`
for each row
begin
update `FILM` SET `FTYPE`='Comdedy' where new.`DNAME`='周星驰';
end;
```

```
insert into `FILM`(`FID`,`FNAME`,`DNAME`) VALUES(13,'喜剧之王','周星驰');
```

FID	FNAME	FTYPE	DNAME	length	IS3D	GRADE
1	film1	Action	Amazon	120	Y	90
2	film2	Love	Gray	119	N	91
3	film3	Suspence	Hover	121	Y	92
4	film4	Adventure	NH3	122	N	93
5	战狼	War	吴京	90	Y	93
6	战狼2	War	吴京	90	Y	94
7	film7	Fantasy	director1	120	Y	70
8	film8	Documentar	director2	120	Y	100
9	film9	Music	NH3	70	N	75
10	英雄本色	Action	吴宇森	90	N	95
11	喋血双雄	Action	吴宇森	90	N	94
12	小众电影	Minor	沁苑翔	30	N	<null>
13	喜剧之王	Comdedy	周星驰	<null>	<null>	<null>

图 3.18 触发器结果

3.2.4 数据查询

1)查询“战狼”这部电影在洪山区各家影院的2017年的上映情况，并按照上映的月份的降序排列；

```
select TID, PRICE, MONTH from `SHOW` where `YEAR` =2017 and `TID` in
(select `TID` from `THEATER` where `TAREA` = '洪山区' and `FNAME` = '战狼')
order by `MONTH` desc;
```

TID	MONTH
1	35
2	35

2 rows in set
Time: 0.013s

图 3.19 查询 1 结果

2)查询所有无参演演员信息的电影的基本信息，并且将结果按照电影类型的升序排列，相同类型的电影则按照用户评分的降序排列；

```
select * from `FILM` where `FID` not in (select `FID` from `ACTIN`) order by `FTYPE` ASC, `GRADE` DESC;
```

FID	FNAME	FTYPE	DNAME	length	IS3D	GRADE
4	film4	Adventure	NH3	122	N	93
2	film2	Love	Gray	119	N	91
3	film3	Suspence	Hover	121	Y	92
5	战狼	War	吴京	90	Y	93

图 3.20 查询 2 结果

3)查询所有直到 2017 年仍未上映的电影编号、电影名称、导演姓名；

```
select * from `SHOW` group by `FID` having `YEAR` < 2018;
```

FID	TID	PRICE	YEAR	MONTH
4	1	35	2017	3

图 3.21 查询 3 结果

4)查询在每家电影院均上映过的电影编号；

```
select `FID` FROM `SHOW` group by `FID` having COUNT(*)=(SELECT COUNT(*) FROM `THEATER` );
```

FID
4

图 3.22 查询 4 结果

5)查询所有用户评分低于 80 分或者高于 89 分的电影编号、电影名称、导演姓名及其用户评分，要求 where 子句中只能有一个条件表达式；

```
select `FID`,`FNAME`,`DNAME`,`GRADE` from `FILM`
where `GRADE` not between 80 and 89;
```

FID	FNAME	DNAME	GRADE
1	film1	Amazon	90
2	film2	Gray	91
3	film3	Hover	92
4	film4	NH3	93
5	战狼	吴京	93
6	战狼2	吴京	94
7	film7	director1	70
8	film8	director2	100
9	film9	NH3	75
10	英雄本色	吴宇森	95
11	喋血双雄	吴宇森	94

图 3.23 查询 5 结果

6)查询每个导演所执导的全部影片的最低和最高用户评分；

```
select `DNAME`,MAX(`GRADE`) AS MAXGRADE,MIN(`GRADE`) AS
MINGRADE FROM `FILM` GROUP BY `DNAME`;
```

DNAME	MAXGRADE	MINGRADE
Amazon	90	90
director1	70	70
director2	100	100
Gray	91	91
Hover	92	92
NH3	93	93
吴京	94	93

图 3.24 查询 6 结果

7)查询至少执导过 2 部电影的导演姓名、执导电影数量；

```
select `DNAME`,count(*) as movieNum FROM `FILM` GROUP BY `DNAME`
HAVING count(*)>=2;
```

DNAME	movieNum
NH3	2
吴京	2

图 3.25 查询 7 结果

8)查询至少 2 部电影的用户评分超过 80 分的导演及其执导过的影片数量、平均用户评分;

```
select 'DNAME',count(*) as greatMovieNum FROM 'FILM' where
'GRADE'>80 GROUP BY 'DNAME' having count(*)>=2;
```

DNAME	greatMovieNum
吴京	2

图 3.26 查询 8 结果

9)查询至少执导过2部电影的导演姓名以及跟这些导演合作过的演员编号、姓名;

```
select
ACTIN.ACTID,ANAME
from
(select 'DNAME','FID' from 'FILM' where 'GRADE'>80 group by 'DNAME'
having count(*)>=2) as a,
ACTIN,
ACTOR
WHERE a.'FID'=ACTIN.'FID'
AND ACTIN.'ACTID'=ACTOR.'ACTID';
```

ACTID	ANAME
5	孙维正
6	徐光磊
7	倪鸿
8	沁苑翔

图 3.27 查询 9 结果

10)查询每个演员担任主角的电影中的平均用户评分;

用户评分：用户对于演员的评分

```
(select a.`ACTID`,AVG(`GRADE`) from `ACTIN` as a,`ACTOR` as b where
a.`ACTID`=b.`ACTID` and a.`ISLEADING`='Y' group by
`ACTID`);
```

ACTID	AVG(`GRADE`)
1	78.3333
3	90.0000
4	90.0000
5	96.0000
6	97.0000
7	98.0000
8	96.0000

图 3.28 查询 10 结果

11)查询用户评分超过 90 分的电影的最早上映年月；

```
select `SHOW`.`FID`,min(str_to_date(concat(CAST(`YEAR` AS
CHAR),CAST(`MONTH` AS CHAR)),`%Y%m`)) as 'earliestTime' from
`FILM`
join
`SHOW`
on `FILM`.`FID`=`SHOW`.`FID`
where `GRADE`>90
group by `SHOW`.`FID`;
```

FID	earliestTime
4	2017-01-00
5	2016-01-00
6	2017-12-00

图 3.29 查询 11 结果

12)查询用户评分超过 90 分的电影的最早上映年月及其相应的上映影院编号；

```
select `SHOW`.`FID`,`SHOW`.`TID`,min(str_to_date(concat(CAST(`YEAR` AS
CHAR),CAST(`MONTH` AS CHAR)),`%Y%m`)) as 'earliestTime' from `FILM`
join
```

```
`SHOW`
on `FILM`.`FID`=`SHOW`.`FID`
where `GRADE`>90
group by `SHOW`.`FID`;
```

FID	FNAME	earliestShowTime
4	film4	2017
6	战狼 2	2017

图 3.30 查询 12 结果

13)查询每个电影的上映总次数;

```
select `FID`,COUNT(`FID`) AS 'totalShowTimes' from `SHOW` group by `FID`
```

FID	totalShowTimes
4	5
6	3

图 3.31 查询 13 结果

14)查询执导过动作片，或者警匪片，或者枪战片的导演的姓名，要求 where 子句中只能有一个条件表达式;

```
select `DNAME` from `FILM` where `FTYPE` in ('Action','Shooting','Police')
group by `DNAME`;
```

DNAME
Amazon
Gray
吴宇森

图 3.32 查询 14 结果

15)查询所有“战狼”系列的电影的编号、电影名称、上映电影院名称及其上映年月，结果按照电影名称的升序排列;

```
select a.`FID`,a.`FNAME`,b.`TID`,b.`YEAR`,b.`MONTH`,c.`TNAME` from `FILM`
as a,`SHOW` as b,`THEATER` as c
```

```
where a.`FNAME` LIKE '%战狼%'
and a.`FID`=b.`FID`
and b.`TID`=c.`TID`
order by `FNAME` asc;
```

FID	FNAME	TID	YEAR	MONTH	TNAME
5	战狼	3	2016	2	武昌区 1
5	战狼	2	2016	1	洪山区 2
5	战狼	1	2016	1	洪山区 1
6	战狼 2	1	2018	1	洪山区 1
6	战狼 2	3	2017	12	武昌区 1
6	战狼 2	2	2018	2	洪山区 2

图 3.33 查询 15 结果

16)查询在同一个年月上映 1 号和 2 号电影的影院编号；

```
select a.`TID`,a.`FID`,b.`FID`,a.`YEAR`,a.`MONTH` from `SHOW` as a,`SHOW`
as b where a.`MONTH`=b.`MONTH` and a.`YEAR`=b.`YEAR`
and a.`FID`=1 and b.`FID`=2 and a.`TID`=b.`TID` group by a.`TID`;
```

TID	FID	FID	YEAR	MONTH
2	1	2	2017	1

图 3.34 查询 16 结果

17)查询所有没参演过用户评分 85 分以下电影的演员的编号、姓名；

```
select `ACTID`,`ANAME` from `ACTOR` where `ACTID` not in (select
b.`ACTID` from `FILM` as a,`ACTIN` as b where a.`FID`=b.`
FID` and a.`GRADE`<85);
```

ACTID	ANAME
2	Bob
5	孙维正
6	徐光磊
7	倪鸿
8	沁苑翔
9	梦桐
10	张国荣
11	周润发

图 3.35 查询 17 结果

18)查询参演过“吴宇森”执导过的所有电影的演员姓名；

```
select `ANAME` from `ACTOR`
where `ACTID` in
(select `ACTID` from `ACTIN`
where `FID` in (select `FID` from `FILM` where `DNAME`='吴宇森'))
group by `ACTID`
having count(*)=(select count(*) from `FILM` where `DNAME`='吴宇森'));
```

ANAME
周润发

图 3.36 查询 18 结果

19)查询所有的演员的编号、姓名及其参演过的电影名称，要求即使该演员未参演过任何电影也要能够输出其编号、姓名；

```
select `ACTID`,`ANAME`,`FNAME` from
(select a.`ACTID`,`ANAME`,b.`FID` from `ACTOR` as a
left join
`ACTIN` as b
on a.`ACTID`=b.`ACTID`) as c
left join
`FILM` as d
on c.`FID`=d.`FID`;
```

ACTID	ANAME	FNAME
1	Petter	film4
1	Petter	film1
1	Petter	film2
1	Petter	film3
1	Petter	film7
2	Bob	film4
3	Angla	film4
3	Angla	film9
4	Perch	film4
4	Perch	film7
5	孙维正	战狼
6	徐光磊	战狼
7	倪鸿	战狼
8	沁苑翔	战狼
9	梦桐	<null>
10	张国荣	英雄本色
11	周润发	英雄本色
11	周润发	喋血双雄
12	不演戏	<null>

图 3.37 查询 19 结果

20)查询所有上映超过 3 次但没有用户评分的电影编号、名称。

```
select a.`FID`,b.`FNAME` from `SHOW` as a
left join `FILM` as b
on a.`FID`=b.`FID`
where b.`GRADE` is NULL
group by a.`FID`
having count(*)>=3;
```

FID	FNAME
12	小众电影

图 3.38 查询 20 结果

3.2.5 了解系统的查询性能分析功能（选做）

目的: 选择上述 3.2.4 任务中某些较为复杂的 SQL 语句，查看其执行之前系统给出的分析计划和实际的执行计划，记录观察的结果，并对其进行简单的分析。

使用 explain 查询 3.2.4(20)SQL 语句执行计划:

```
explain select a.`FID`, b.`FNAME` from `SHOW` as a
left join `FILM` as b
on a.`FID`=b.`FID`
where b.`GRADE` is NULL
group by a.`FID`
having count(*)>=3;
```

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	a	index	<null>	FID	4	<null>	15	Using index
1	SIMPLE	b	eq_ref	PRIMARY	PRIMARY	4	DBlabs.a.FID	1	Using where

图 3.39 查询查询性能分析功能结果

其中:

table:显示关于哪张表

type:连接使用的类型 此处为 index 连接(group 操作) 和 等值连接(left join 操作)

possible_keys :显示可能应用在这张表中的索引

key :实际使用的索引。如果为 NULL，则没有使用索引

key_len :使用的索引的长度

ref :显示索引的哪一列被使用了,如果可能,为常数

rows :MYSQL 认为必须检查的用来返回请求数据的行数

Extra :关于 MYSQL 如何解析查询的额外信息

3.2.6 DBMS 函数及存储过程和事务（选做）

1) 通过系统帮助文档学习系统关于时间、日期、字符串类型的函数，为电影表增加首映时间属性，然后查询下个月首映的电影信息。

```
alter table `FILM` add premiere date;
```

```
update `FILM` set `premiere`=date_format("2018-1-1","%y-%m-%d") where `FID`=1;
```

```
update `FILM` set `premiere`=date_format("2018-6-1","%y-%m-%d") where `FID`=6;
```

```
select * from `FILM` where month(`premiere`)-month(now())=1;
```

FID	FNAME	FTYPE	DNAME	length	IS3D	GRADE	premiere
6	战狼 2	War	吴京	90	Y	94	2018-06-01

图 3.40 下月首映电影结果

2) 编写一个依据演员编号计算在其指定年份参演的电影数量的自定义的函数，并利用其查询 2017 年至少参演过 5 部电影的演员编号。

自定义函数:

```
create function countActorFilmNumByYear(actid int,yearnum int)
```

```
returns int
```

```
begin
```

```
return (select count(*) from `ACTOR`
```

```
join `ACTIN`
```

```
on `ACTOR`.`ACTID`=`ACTIN`.`ACTID`
```

```
join `FILM`
```

```
on `ACTIN`.`FID`=`FILM`.`FID` and year(`premiere`)=yearnum
```

```
where `ACTOR`.`ACTID`=actid);
```

```
end;
```

countActorFilmNumByYear(1,2018)
1

图 3.41 自定义函数测试

```
select `ACTID`,countActorFilmNumByYear(`ACTID`,2017) from `ACTOR` where countActorFilmNumByYear(`ACTID`,2017)>=5;
```

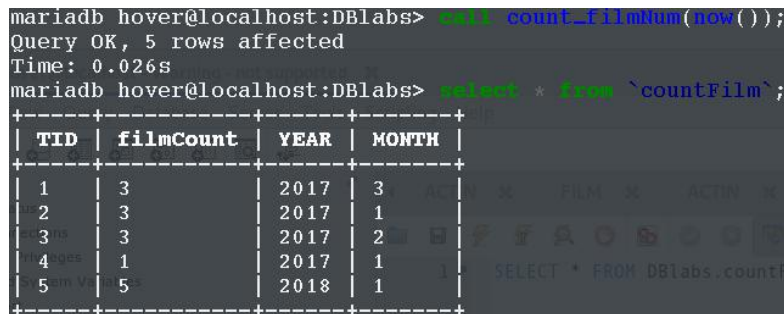
ACTID	countActorFilmNumByYear(`ACTID`,2017)
1	6
2	5

图 3.42 2017 年上映电影中出演 5 场以上演员及电影数

3) 尝试编写 DBMS 的存储过程，建立每家影院的上映电影总数的统计表，并通过存储过程更新该表。

```
create or replace procedure count_filmNum(stime date)
begin
create or replace table `countFilm` select `TID`,count(*) as
`filmCount`,`YEAR`,`MONTH` from `SHOW`
where str_to_date(concat(CAST(`YEAR` AS CHAR),CAST(`MONTH` AS
CHAR)),`%Y%m`)< stime group by `TID`;
end;

call count_filmNum(now());
```



```
mariadb hover@localhost:DBlabs> call count_filmNum(now());
Query OK, 5 rows affected
Time: 0.026s
mariadb hover@localhost:DBlabs> select * from `countFilm`;
```

TID	filmCount	YEAR	MONTH
1	3	2017	3
2	3	2017	1
3	3	2017	2
4	1	2017	1
5	5	2018	1

图 3.43 截止当前时间之前(now())的各电影院数目统计

4) 尝试在 DBMS 的交互式界面中验证事务机制的执行效果。

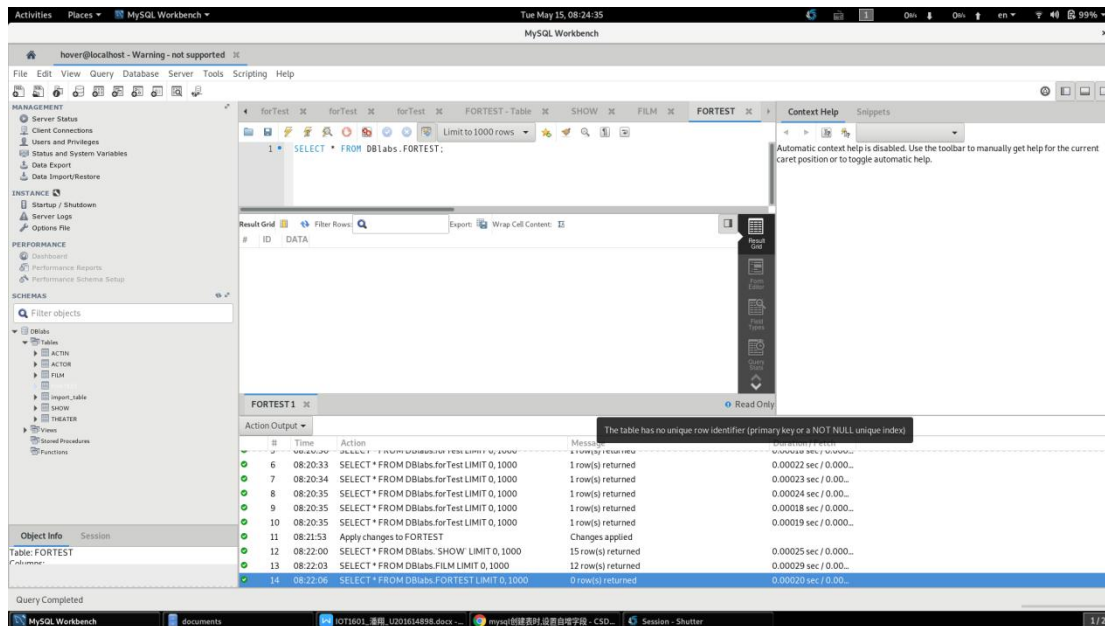
3.3 任务总结

1. 当存在外键时，删除主键出错：

```
mariadb hover@localhost:DBlabs> drop table `SHOW`;
ERROR 1025 (Error on rename of './DBlabs/#sql-231_1h') to './DBlabs/SHOW': (errno: 150 "Foreign key constraint is incorrectly formed")
```

删除外键，然后删除主键

2. 当无 unique 和无主键时，无法使用可视化终端插入数值



此处疑似为可视化终端的不兼容，采用命令行进行插入

3. 时间函数的使用：mysql 提供了较为友好的时间函数，内部采用时间戳，故进行排序的时候效率高于自己使用年份和月份二次排序，故采用转换为标准时间格式进行排序

4. 创建触发器时需要 root 权限

5. 自定义函数权限

This function has none of DETERMINISTIC, NO SQL, or READS SQL DATA in its declaration and binary logging is enabled

Warning: This function has none of DETERMINISTIC, NO SQL, or READS SQL DATA in its declaration and binary logging is enabled (you might want to use the log_bin_trust_function_creators variable).

使用 set global log_bin_trust_function_creators=TRUE;

6. 整个实验的过程中熟练了 SQL 语句的使用，同时了解到了 MySQL 的内部一些实现细节，如压缩表格，数据存储格式，便于更好的进行数据库开发

7. 了解到 MySQL 的执行计划，能够更好的优化 SQL 语句

4 数据库应用系统设计

4.1 系统设计目标

自行选择所擅长的 DBMS 软件以及数据库应用系统（客户端程序或者网站）的程序开发工具，参考后面的题目例子，拟定一个自己感兴趣的数据库应用系统题目，完成该小型数据库应用系统的设计与实现工作。主要包括：需求调研与分析、总体设计、数据库设计、详细设计与实现、测试等环节的工作。

选择题目:机票预订系统

4.1.1 系统功能的基本要求：

- 每个航班信息的输入。
- 每个航班的坐位信息的输入；
- 当旅客进行机票预定时，输入旅客基本信息，系统为旅客安排航班，打印取票通知和帐单；
- 旅客在飞机起飞前一天凭取票通知交款取票；
- 旅客能够退订机票；
- 能够查询每个航班的预定情况、计算航班的满座率。

4.1.2 数据库要求：

在数据库中至少应该包含下列数据表：

- 航班信息表；
- 航班坐位情况表；
- 旅客订票信息表；
- 取票通知表；
- 帐单。

使用 C/S 架构

4.2 需求分析

1. 进行用户行为约束，对于普通用户，某些操作一旦进行无法更改：
 - a) 航班选座
 - b) 订单确认
2. 用户行为的时间约束：
 - a) 旅客在飞机起飞前一天凭取票通知交款取票
 - b) 用户检票时间约束
3. 系统查询设计，完成满足指定要求的查询

4.3 总体设计

4.3.1 系统功能模块划分

系统总体分为：航班查询，航班预订，订单管理，账单管理，通知系统

1. 航班查询：进行航班的查询操作
 - a) 模糊查询：利用时间，地点，价格等条件查询航班
 - b) 精确查询：利用航班号查询
2. 航班预订：进行航班预订的相关操作
 - a) 座位选取
 - b) 订单生成和确认
 - c) 于订单管理模块，账单管理模块交互
3. 订单管理：进行订单的相关操作
 - a) 管理订单状态，已经完成的订单和约束订单无法操作
 - b) 更新订单状态
4. 账单管理：生成系统的账单

4.3.2 系统架构方案

系统采用 B/S 架构，各模块之间进行较为清晰的接口定义，使模块之间尽量解耦合和独立，采用 QT 框架，C++语言开发，使用 QSqlQuery 模块与 MySQL(Mariadb) 进行交互。

1. 用户权限：使用 DB 中的 tabel 来进行管理
2. 用户操作约束：在前端界面进行判断，SQL 语句仅处理正常逻辑，便于系统实现和鲁棒性

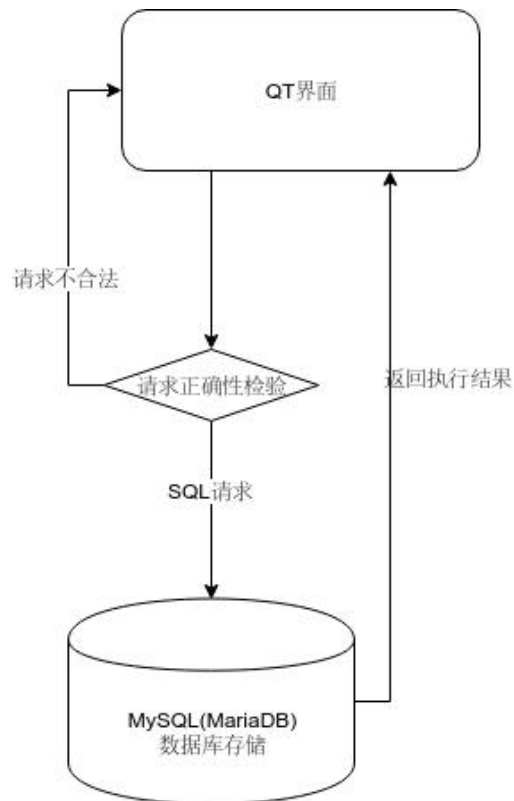


图 4-1 系统 B/S 架构图

4.4 数据库设计

4.4.1 系统操作流程

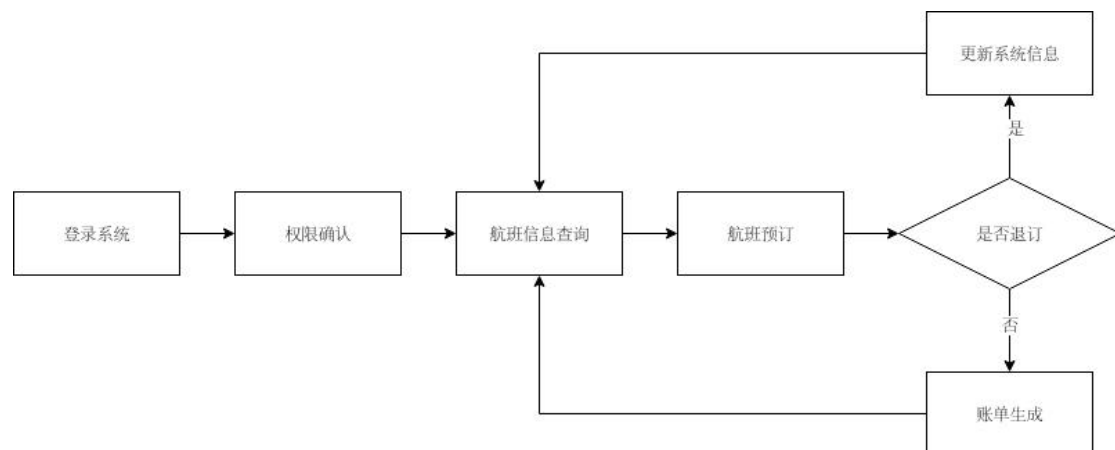


图 4-2 系统操作流程

4.4.2 系统操作表格图

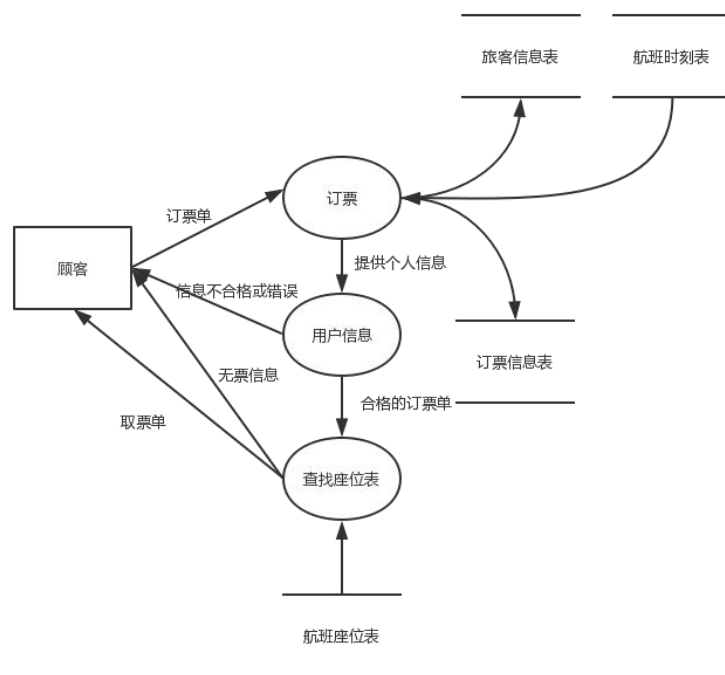


图 4-3 操作表格图

4.4.3 E-R 图

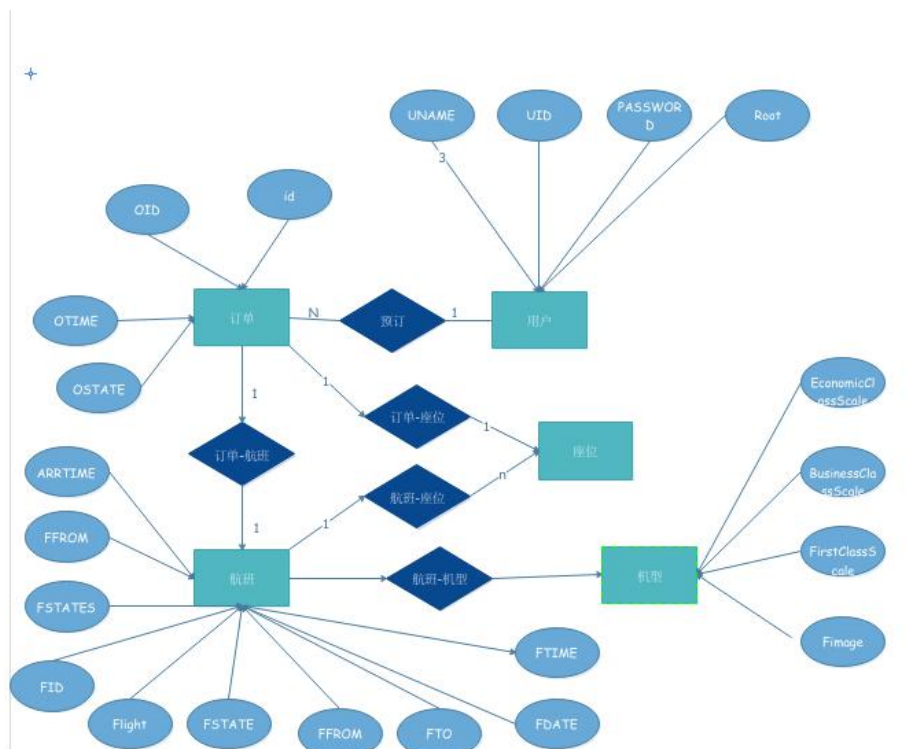


图 4-4 系统设计 E-R 图

4.4.4 数据流图

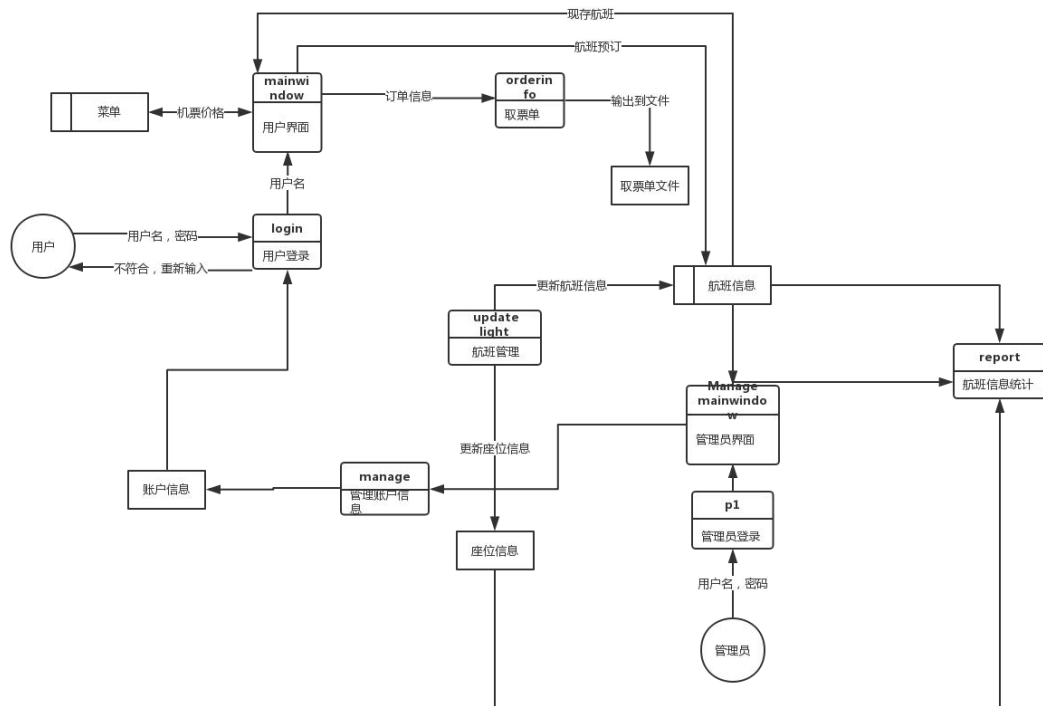


图 4-5 系统设计数据流图

4.4.5 数据库逻辑结构设计

1. 用户权限表：对于用户的行为和权限进行约束管理，同时便于统计用户信息
 - a) 用户名 UID
 - b) 用户姓名 UNAME
 - c) 密码 PASSWORD
 - d) 用户权限
2. 航班信息表：用于航班信息的管理
 - a) 航班号 FID
 - b) 航空公司 Flight
 - c) 飞机机型 FMODEL
 - d) 航班状态：FSTATE
 - i. 已降落 landed
 - ii. 延误 delayed
 - iii. 登机口 gate
 - iv. 登机 boarding

- v. 登机手续办理 check-in
 - e) 起飞城市: FFROM
 - f) 飞往城市: FTO
 - g) 起飞日期: DATA
 - h) 起飞时间: TIME
 - i) 座位状态: FSTATUS
- 3. 用户订单/账单表: 用于用户订单的管理, 账单的生成
 - a) 订单号 OID
 - b) 航班号 FID
 - c) 用户号 UID
 - d) 座位号 SID
 - e) 订单时间 OTIME
 - f) 订单金额 OAMOUNT
 - g) 订单状态 OSTATE
- 4. 航班座位表
 - a) 航班号 FID
 - b) 座位号 SID
- 5. 航班模型表
 - a) 飞机型号 FMODEL
 - b) 舱位设置 矩阵形式

4.4.6 数据表格设计

(1) 用户权限表

用户权限表用作用户的登录验证，进入系统后的权限验证。

表 4-1 用户权限表

属性	约束	数据类型	说明
UID	Primary key	int	用户编号
USERNAME	unique	varchar	用户名
PASSWORD		varchar	密码
Root		tinyint(bool)	是否是管理员

(2) 航班信息表

航班信息表用作航班信息管理，和航班状态的维护，便于用户查询预订航班。

表 4-2 航班信息表

属性	约束	数据类型	说明
FID	Primary key	int	航班编号
Flight	unique	varchar	用户名
FMODEL	外码： Reference 飞机型号 FSTATUSinfo(FMODEL)	varchar	飞机机型
FSTATE		varchar	航班状态
FFROM		varchar	
FTO		varchar	
FDATE		Date	
FTIME		Time	
ARRDATE		Date	
ARRTIME		Time	
FSTATUS		varchar	航班座位情况 (Not Full,FULL)

(3) 飞机型号表:

用于记录飞机型号，存储飞机座位排布图和座位排布信息。

表 4-3 飞机型号表

属性	约束	数据类型	说明
FMODEL	Primary key	varchar	飞机机型
Fimage	unique	LONGBLOB	机型图片
FirstClassScale		varchar	头等舱规模 (如: 2 排*每排 4 个人 2*4)
BussinessClass Scale		varchar	商务舱规模 同上
EconomyClass Scale		varchar	经济舱规模 同上

(4) 航班座位表

用于存储航班的座位信息，判断当前航班时候还有位置，并存储座位的等级信息，用于报表输出

表 4-4 航班座位表

属性	约束	数据类型	说明
FID	外码：参照航班信息表 FLIGHTInfo(FID)	int	航班编号
SID	由 FID 和 SID 复合主码	varchar	座位号
USABLE		Tinyint(bool)	座位是否可用
SeatRank		varchar	座位等级 (头等舱 F 商务舱 C 经济舱 Y)

4.5 详细设计与实现

4.5.1 开发环境与运行环境

操作系统：ArchLinux x64

IDE:Qt Creator

编译器：8.1.1 20180531 (GCC)

数据库:10.1.33-MariaDB MariaDB Server

因 Qt 框架的跨平台特性，可以发行各平台的发行版本

4.5.2 航班事务管理

(1) 航班添加

从飞机型号表(FMODELinfo)中利用 SQL 语句查询出飞机的型号，并设置型号选择款，在界面中进行输入值的约束，将输入值整理为 SQL 语句插入数据库，同时根据飞机型号查询座位排列，根据座位排列自动编排序号将座位插入座位信息表。

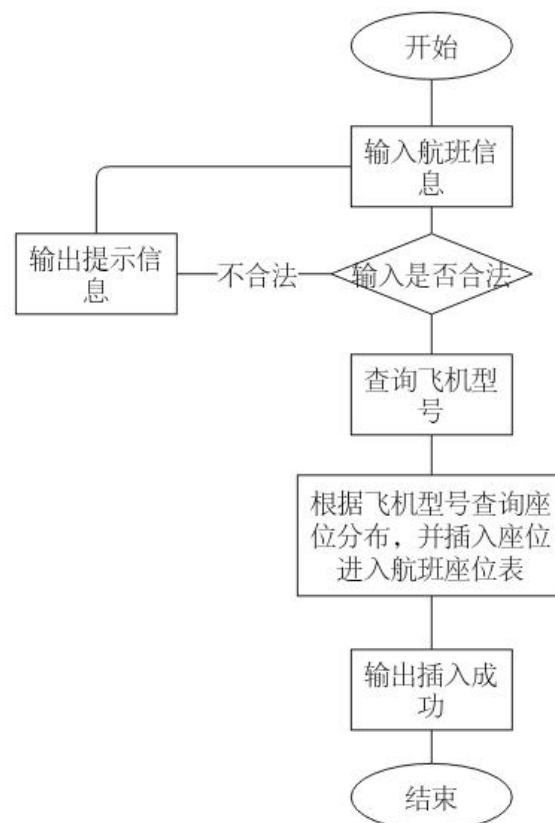


图 4-4 航班添加流程图

(2) 航班删除

根据删除航班的航班号 FID 查询是否存在用户订单为此航班，如果存在用户订单为此航班，则拒绝操作，并给出提示信息。否则，正常删除航班信息并删除座位信息。

(3) 航班修改

当用户的权限为 ROOT 管理员权限时，将航班信息总表设置为可修改，但航班号 FID 为主码，设置锁定。

(4) 航班查询

航班总表输出所有的航班，航班查询框根据条件筛选查询。

用户必须输入航班的始飞地和到达地，如果二者为空，给出提示信息。

用户必须输入正确的时间，如未修改时间，给出提示信息：利用初始设置时间，进行修改时间判断，是否修改。

根据输入的条件进行筛选，如果未输入，则默认条件为空。

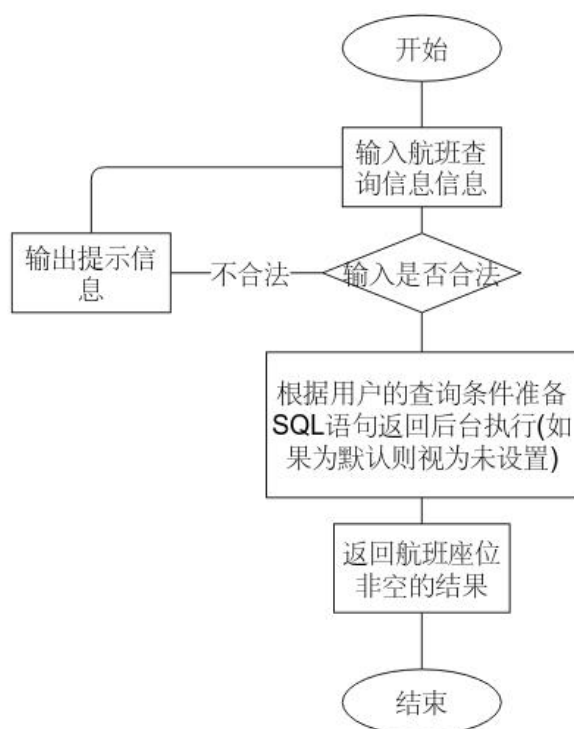


图 4-5 航班查询流程图

4.5.3 订单事务管理

(1) 预订

用户查询航班,选中想要预订的航班,根据选中的航班号 FID 进行座位查询,如无座位则返回。

进入座位选择界面:

根据飞机机型的座位排布图自动生成座位的选择按钮阵列,查询飞机机型表获取机型座位分布图并显示,用户每次只能选择一个座位,且已经被预订的座位置为红色,同时用户可以根据上方列表进行选取。

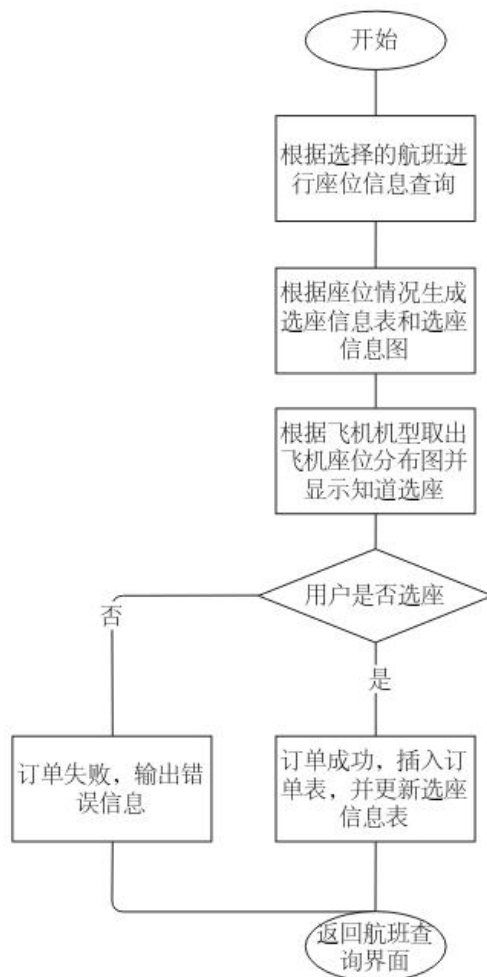


图 4-6 选座生成订单流程图

(2) 退订

判断订单状态，如订单状态为 `unfinish`，则可以进行退订，同时将座位的状态设置为未占用。

如果订单状态为其他，则输出订单状态错误，无法退订错误信息。

(3) 订单取票 check-in

校验订单状态：如果订单状态为 `unfinish` 则继续

校验当前时间，如当前时间为订单的前一天或者订单的当天，则进行订单的 `checki-in`，调用打印接口进行订单信息的打印输出。

4.5.4 权限系统

进入系统时，查询用户表 `user` 进行用户名和密码的确认，并查询是否是管理员信息进行相应设置。

如果是管理员：开放航班总表修改权限，开放航班添加删除权限。

4.5.5 报表系统

在进行座位选取时，进行座位信息的统计输出。

管理员可以查看所有航空公司最近三个月的统计图：

获取所有的航空公司名称并统计数目。

获取当前月份，并查询各公司在上上个月和上个月的统计情况，调用 `QT charts` 进行统计图绘制，并设置表头。

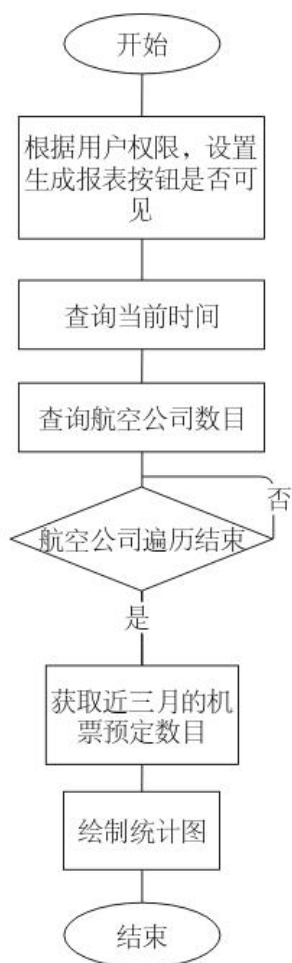


图 4-7 统计图生成模块流程图

4.5.6 通用数据显示模块

因为 QT 中提供两种 SQL 接口，sqlmodel 提供利用 sql 语句显示表格的少量功能，无法进行修改和选中设定一些操作，而 sqltable 采用对象映射框架，完全避免的 sql 语句，为满足实验要求，减少模块耦合性和后期扩展性，继承 sqlmodel 模块进行重载，完成模块的自动化显示设置功能。

(1) 中英文映射

利用 SQL 语句查询表头，获取表头进行自动设置，因本系统采用英文表头，且较为明晰，故未启用映射为中文。

映射模块：使用 map 对于表头映射为中文之后进行设置

(2) 表头查询：

```

bool MySqlQueryModel::set_opTitle()
{
    QSqlQuery query;

```

```

query.prepare("SHOW COLUMNS FROM "+opName);
bool isOk = query.exec();
//qDebug() <<"title"<<isOk;
if(!isOk)
{
    return false;
}
opTitle.clear();
while (query.next())
{
    opTitle.append(query.value(0).toString());
}
QSqlQuery priquery;
priquery.prepare("SHOW COLUMNS FROM FLIGHTinfo where
`Key`='PRI'");
isOk = priquery.exec();
priquery.next();
opPRI=priquery.value(0).toString();
for (int i = 0; i < opTitle.size(); ++i)
{
    if (opTitle.at(i) ==opPRI)
    {
        opPRI_col=i;
        break;
    }
}
return true;
}

```

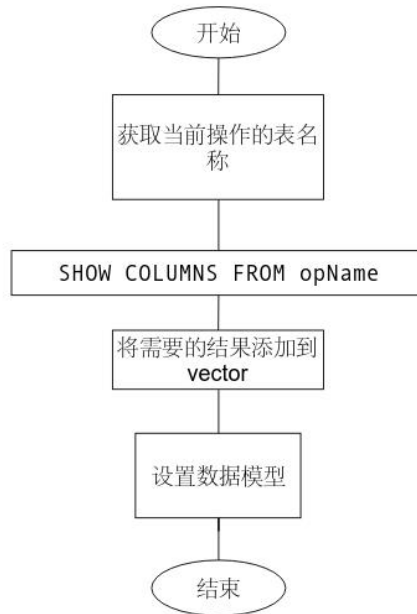


图 4-8 表头查询模块流程图

(3) 修改限制

利用 `SHOW COLUMNS FROM FLIGHTinfo where `Key`='PRI'` 获取当前的主码，并设置为禁止修改

(4) 用户切换窗口

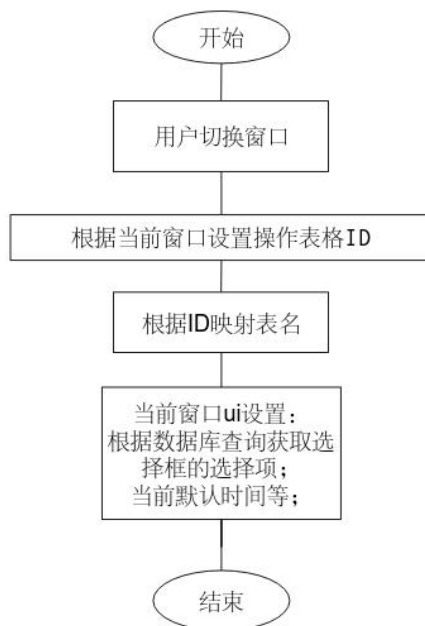


图 4-9 用户切换窗口后台操作流程

4.6 系统测试

4.6.1 航班事务测试

(1) 添加航班

如过未修改时间，则进行提示：

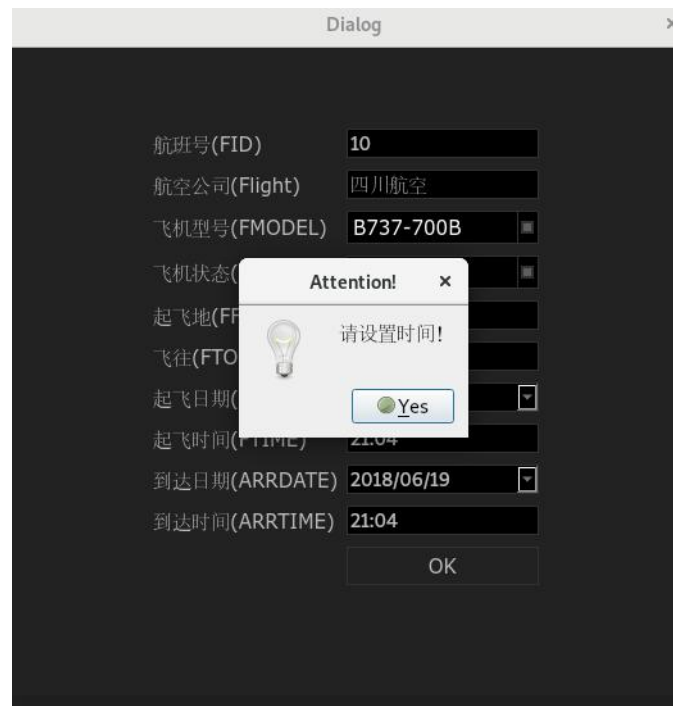


图 4-10 添加航班时间错误提示图

如过数据错误(主码冲突)，则进行提示：



图 4-11 添加航班主码错误提示图

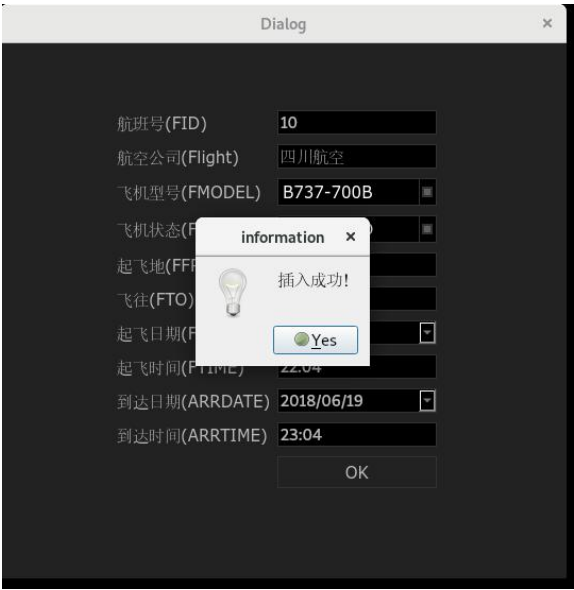


图 4-12 添加航班成功提示图

(2) 修改航班

管理员可以直接进行修改，如果修改不满足数据要求则不会改变，数据约束在数据库中进行的。

航班

航班查询

订单管理

账单查询

报表

FID	Flight	FMODEL	FSTATE	FFROM	FTO	FDATE	FTIME	ARRDATE	ARRTIME	FST
2	东方航空	B737-700B	SCHEDULED	武汉	上海	6/19/18	9:43 AM	6/19/18	10:43 AM	NOT F
3	南方航空	B737-700B	SCHEDULED	上海	北京	6/19/18	11:23 AM	6/19/18	1:23 PM	NOT F
4	南方航空	B737-700B	SCHEDULED	武汉	成都	5/20/18	4:03 PM	5/20/18	6:05 PM	NOT F
6	南方航空	B737-700B	SCHEDULED	北京	哈尔滨	5/19/18	4:53 PM	5/19/18	7:54 PM	NOT F
7	海南航空	B737-700B	SCHEDULED	海口	北京	4/19/18	4:59 PM	4/19/18	6:59 PM	NOT F
8	东方航空	B737-700B	SCHEDULED	南宁	重庆	5/19/18	6:36 PM	5/19/18	9:36 PM	NOT F
9	东方航空	B737-700B	SCHEDULED	北京	拉萨	6/20/18	6:45 PM	6/20/18	8:44 PM	NOT F
10	四川航空	B737-700B	SCHEDULED	成都	重庆	6/19/18	10:04 PM	6/19/18	11:04 PM	NOT F

add

Delete

show Report

图 4-13 表格修改航班图

(3) 删除航班

航班信息(F)

订单管理(O)

账单查询(T)

航班

航班查询

订单管理

账单查询

报表

FID	Flight	FMODEL	FSTATE	FFROM	FTO	FDATE	FTIME	ARRDATE	ARRTIME	FST
2	东方航空	B737-700B	SCHEDULED	武汉	上海	6/19/18	9:43 AM	6/19/18	10:43 AM	NOT F
3	南方航空	B737-700B	SCHEDULED	上海	北京	6/19/18	11:23 AM	6/19/18	1:23 PM	NOT F
4	南方航空	B737-700B	SCHEDULED	武汉	成都	5/20/18	4:03 PM	5/20/18	6:05 PM	NOT F
6	南方航空	B737-700B	SCHEDULED	北京	哈尔滨	5/19/18	4:53 PM	5/19/18	7:54 PM	NOT F
7	海南航空	B737-700B	SCHEDULED	海口	北京	4/19/18	4:59 PM	4/19/18	6:59 PM	NOT F
8	东方航空	B737-700B	SCHEDULED	南宁	重庆	5/19/18	6:36 PM	5/19/18	9:36 PM	NOT F
9	东方航空	B737-700B	SCHEDULED	北京	拉萨	6/20/18	6:45 PM	6/20/18	8:44 PM	NOT F
10	四川航空	B737-700B	SCHEDULED	成都	重庆	6/19/18	10:04 PM	6/19/18	11:04 PM	NOT F

add

Delete

show Report

图 4-14 删除航班错误图

如果正常删除，同时删除座位

图 4-15 删除航班图

(4) 航班查询

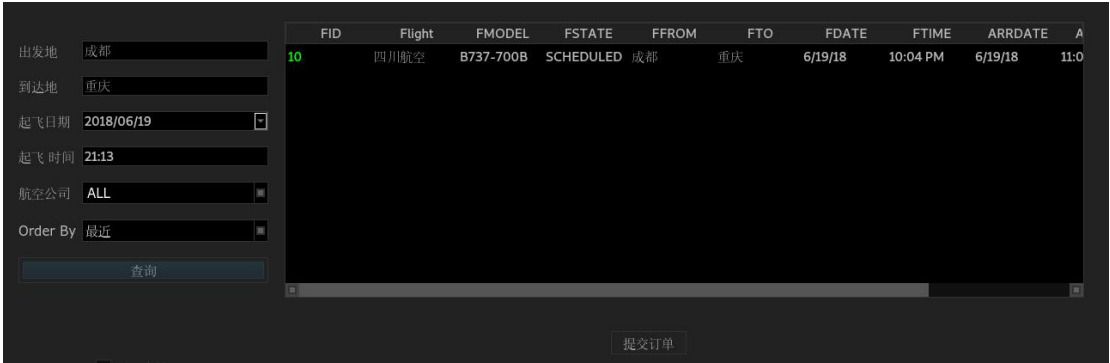


图 4-15 航班查询图

4.6.2 订单事务测试

(1) 预订

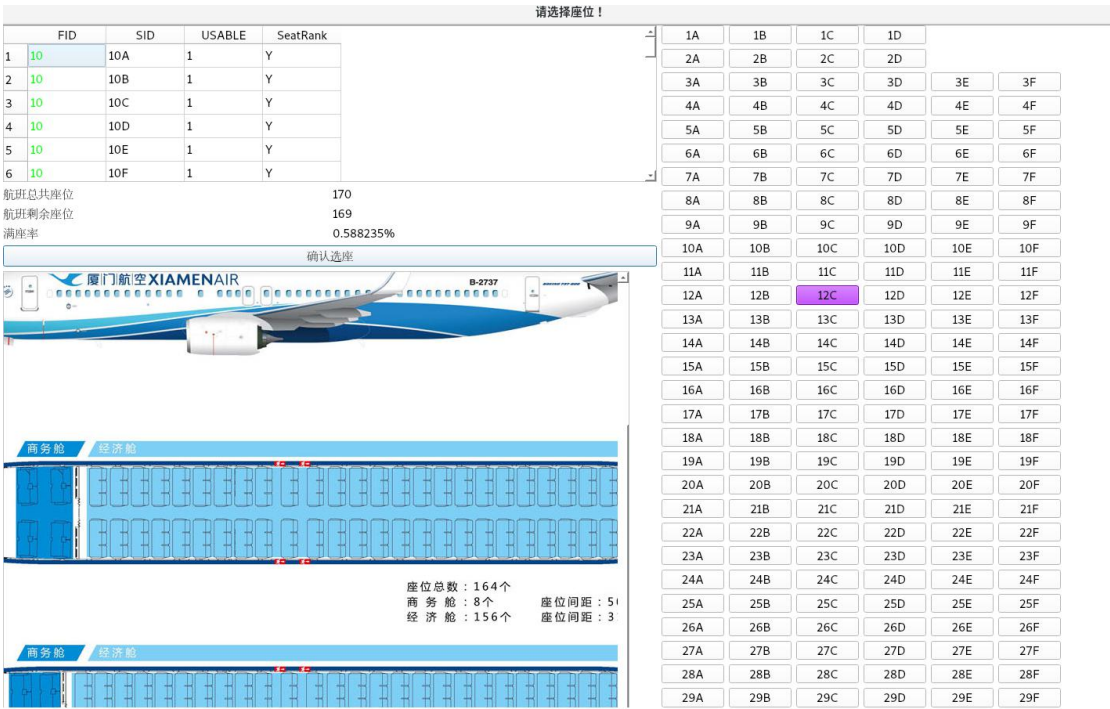


图 4-16 选座系统图

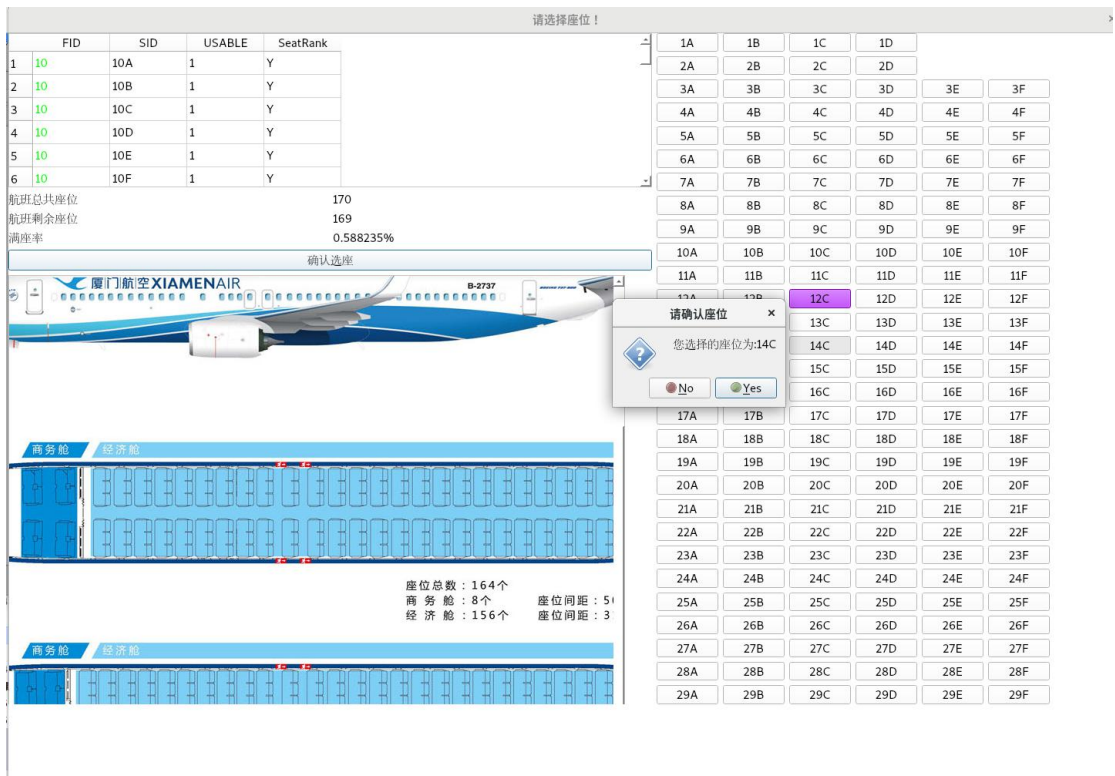


图 4-16 确认选座图

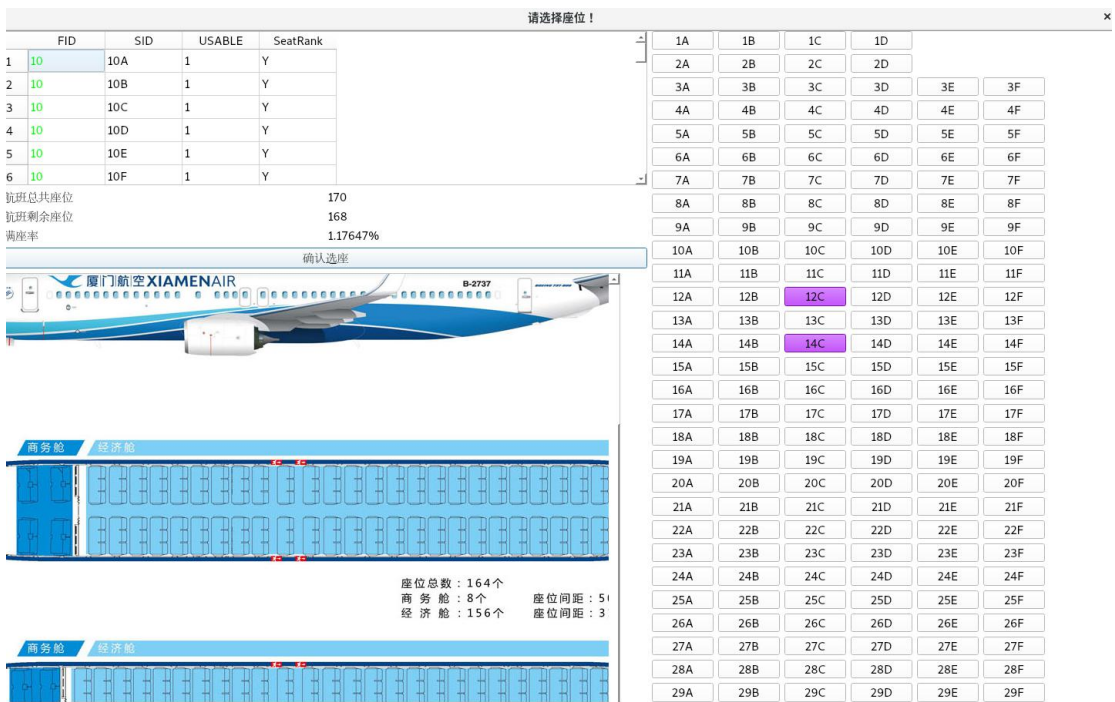


图 4-16 再次查询图

(2) 退订

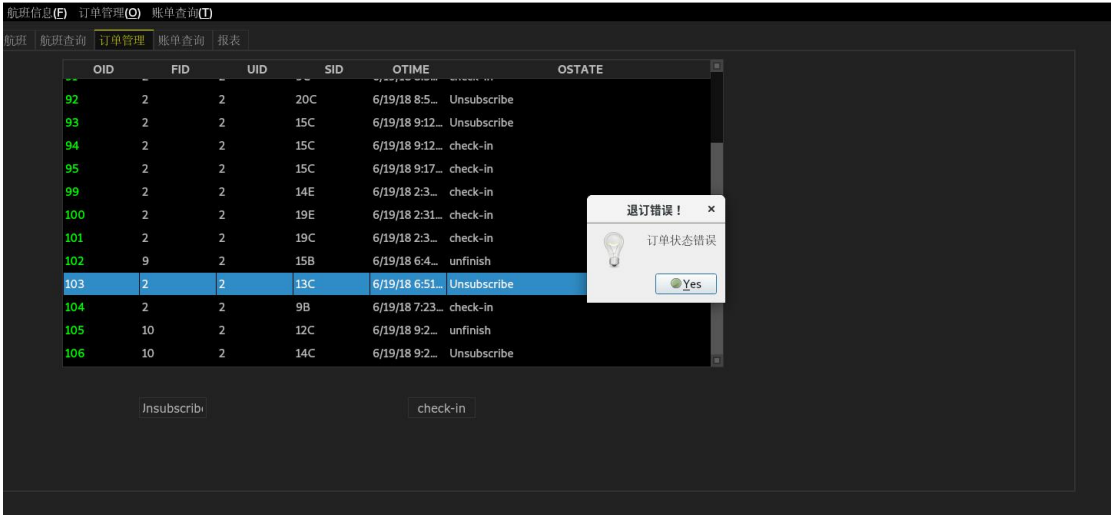


图 4-16 退订错误图

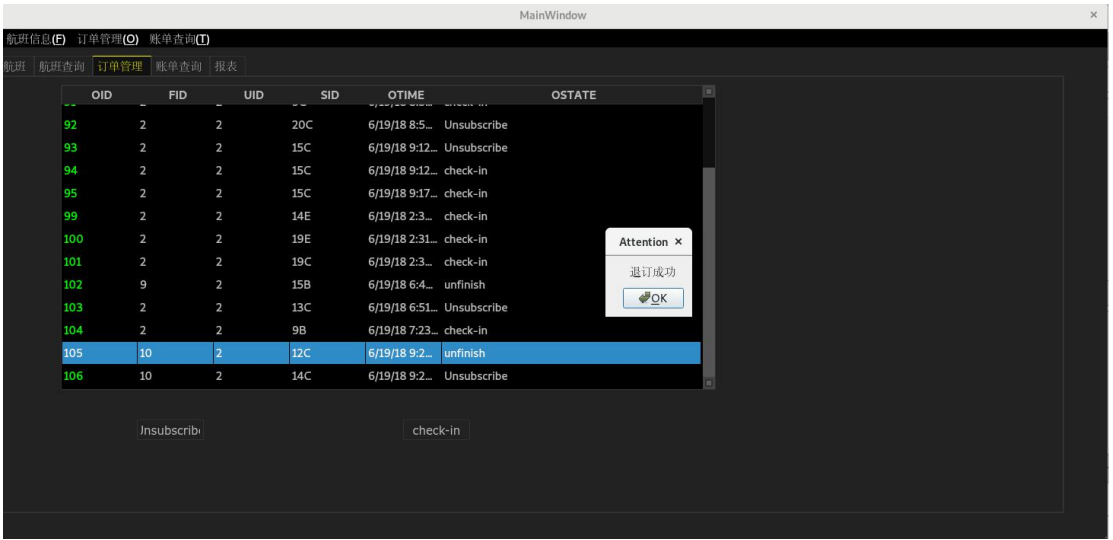


图 4-16 退订成功图



图 4-16 退订再次预订图

(3) 取票并输出机票

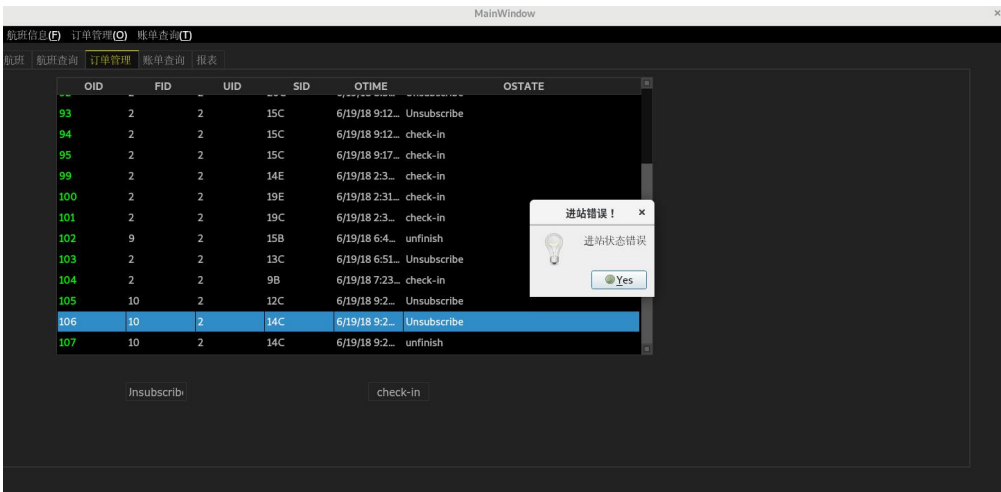


图 4-16 检票错误图



图 4-16 机票确认图



图 4-16 机票模拟打印图（输出到文件）

4.6.3 报表测试

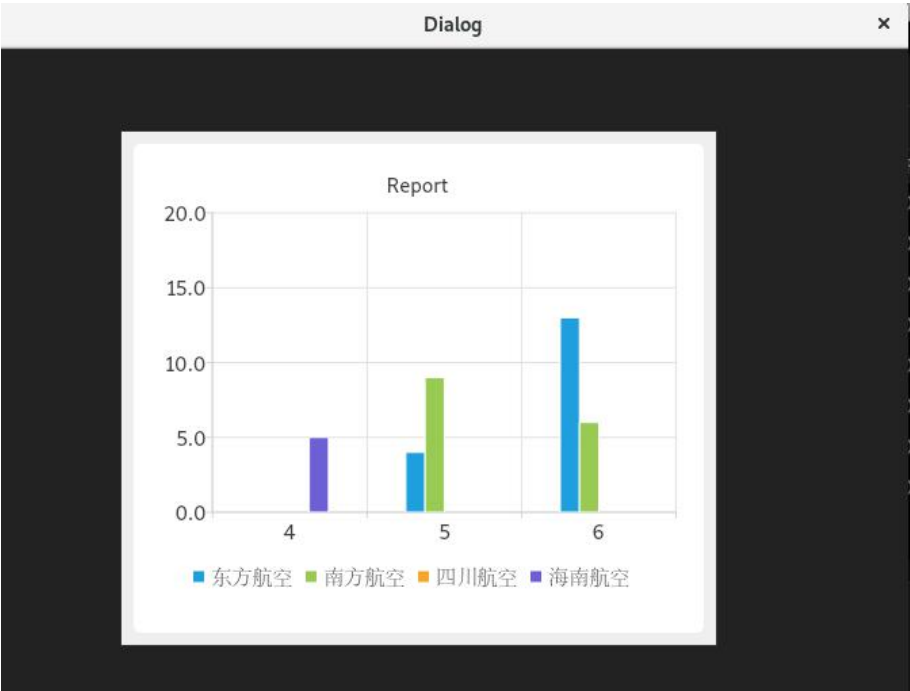


图 4-15 报表测试图

4.7 系统设计与实现总结

在设计中，进一步熟练了 SQL 语句的使用，同时对于 QT 框架的数据模型有了更好的理解，自己能够在框架上继承修改形成自己的框架，而不是仅仅依赖于原有的框架。在界面的实现上，尽可能的采用了自动化布局而不采用手工布局，提高了代码的可拓展性，降低了耦合性，同事在整个设计的过程中，对代码不断的重构，使代码能够精炼的实现功能。

同时，在程序嵌入 SQL 上，可以利用 SQL 语句获得执行结果，也可以利用 model 设置 SQL 语句获取执行结果，在不同的情境下使用不同的语句。

遇到如下问题：

1. SQL 语句中的一些语法如 `exit` 在进行复杂嵌套的时候，QT 中经常执行不成功，经过反复调试，确认是框架问题，只能手写。
2. SQL 语句中的参数绑定，不能绑定表名，在进行自动化设置的时候，只能手动拼接 STR 进行查询。
3. 由于 SQL 返回的错误类型模糊，很多情况下只能在前端认为判断错误原因，进行错误提示，此处为较为麻烦之处，可考虑读取 log 进行错误输出，而不用人工判断，使模型进一步自动化。

数据库设计阶段，对数据库功能、数据字典的设计，加深了对于关系模式、E-R 图、范式、数据流图等概念的理解。

数据库操纵类编写，基于 QT 信号槽的设置，设计过程中，将主要的数据库操作继承子 `QSqlQuery` 基类进行重载，初期考虑使用 DAO 模型进行对象映射，但此时对于底层的操作可以不提供接口进行权限控制，实现更为简单，故使用 `mysqlmodel` 进行基本的数据操作类封装形成基本的 DAO 模型，并提供清晰的接口，上层只需要更新操纵表，即可以对当前表的相关属性进行自动设置。

后期测试，使用了不同的数据进行测试，且在模块编写中，对模块进行了单元测试，增强了鲁棒性。

整个过程中，对于代码进行了一定程度的重构，以便于对模块的复用。

5 课程总结

因为之前有利用过数据库进行过相关软件的编写，故在整个过程中有更多的机会去尝试一些之前没有尝试过的部分，如触发器，事务处理，备份等部分，这些部分能够帮助更好的理解数据库，同时应用好数据库，对于错误的处理，比如进行测试的过程中，可能会造成数据库数据错误，故测试前需要进行备份。

在进行存储过程中，了解到并尝试了其中的一些方法，比如进行二进制文件存储，可以采用存储二进制流的方式，或者进行文件相对地址的存储，应用再进行文件读取。对于航班座位表类似于矩阵的存储，对于现代针对机器学习存在专用的矩阵数据库，而传统数据库中，可以另开一表，但存在一定程度的冗余，可以采用存储矩阵形式和矩阵 01 表示压缩，但此时如果每次写回数据库，压缩解压操作开销较大，如果存在内存中，一段时间后写回，若出现故障，数据一致性错误，体会到不同的实现方式的不同优缺点，需要根据情形选择。

对于错误的处理，学会了阅读 SQL 返回的错误代码，但错误提示往往语焉不详，仍然需要人工判断，对于较为复杂的错误处理，仍然有待学习，如系统的崩溃处理。

整个课程过程中，应用了已有的理论知识，并学习到了新的知识，同时发现了自己现有的不足，需要进一步学习。

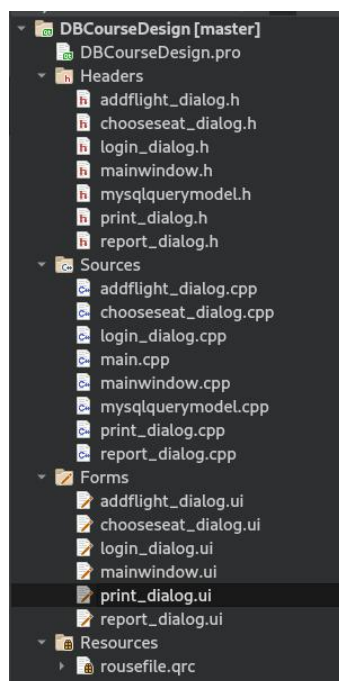
附录

文件说明

采用驼峰命名法，所有的类名为当前类的功能。

- .h 文件为头文件用于函数和信号槽的声明
- .cpp 文件用于函数的实现和相关信号槽的绑定
- .ui 文件为界面文件

整体架构



整体文件架构图

- addflight_dialog 添加航班相关事务的处理和界面绘制
- chooseseat_dialog 添加航班相关事务的处理和界面绘制
- login_dialog 登录处理和相关权限管理及系统初始化
- Mainwindow 主体窗口绘制和相关响应
- Mysqlquerymodel mysql 相关操作的封装和继承重载
- print_dialog 机票打印和行程单确认相关确认
- report_dialog 统计报表相关事务的处理和界面绘制
- rousefile.qrc 界面的样式文件及图标文件

Main

main.cpp

```

/* FileName:main.cpp
 * Author:Hover
 * E-Mail:hover@hust.edu.cn
 * GitHub:HoverWings
 * Description: The main loop of the total Program and link the DB
 */

#include "mainwindow.h"
#include "login_dialog.h"
#include <QApplication>
#include <QSqlQuery>
#include <QMessageBox>
#include <QBuffer>
#include <QDir>
int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    QSqlDatabase db = QSqlDatabase::addDatabase("QMYSQL");
    db.setHostName("hover");
    db.setDatabaseName("DBCOURSEDESIGN");
    db.setUserName("root");
    db.setPassword("990123");
    bool ok = db.open();
    if(ok)
    {
        qDebug()<<"成功连接数据库";
    }
    else
    {
        QMessageBox::warning(NULL,"警告","无法连接数据库");
    }
    a.setWindowIcon(QIcon(":/main.ico"));

    QString currentPath;
    QDir dir;
    currentPath=dir.currentPath();
    qDebug()<<"path"<<currentPath;

```

```

    QSqlQuery query;
    query.prepare("select * from `FLIGHTinfo` as f where exists(select * from
`FSTATUSinfo` as s where s.`FID`=f.`FID` and s.`USABLE`='1') and f.`FID`=3;");
    bool isOK=query.exec();
    while(query.next())
    {
        qDebug()<<query.value(0).toString();
    }
    login_dialog l;
    l.db=&db;
    l.show();

    return a.exec();
}

```


Mysqlquerymodel

mysqlquerymodel.h

```

/* FileName:mysqlquerymodel.h
 * Author:Hover
 * E-Mail:hover@hust.edu.cn
 * GitHub:HoverWings
 * Description:This class inherits from the basical QSqlQueryModel to make the
module can edit and autofit the UI
 * Main Funciton:
 * 1. Judge the role to control the edit Authority
 * 2. Auto fit the process table and get the tabel title from the DB
 * 3. Implement the title to the DB dictionary mapping, which is choosable
 * 4. Set the table properity automaticly
 */

#ifndef MYSQLQUERYMODEL_H
#define MYSQLQUERYMODEL_H
#include "mainwindow.h"
#include <QSqlQueryModel>
#include <QVector>
#include <QMap>
/*
 * op tabel
 * 1:user
 * 2:FLIGHTinfo
 * 3:
 */

class MySqlQueryModel : public QSqlQueryModel
{
    Q_OBJECT
public:

    //public var
    int opView;
    int opTable;
    QString opName;
    QString opPRI;
    int opPRI_col;
    QVector<QString> opTitle;    //table title

```

```

    QVector<QString> addVec;    //the vec
    class MainWindow* mw;
    QMap<QString,QString> map;

    // public function
    QSqlQueryModel();
    explicit QSqlQueryModel(QObject *parent = 0);
    Qt::ItemFlags flags(const QModelIndex &index) const;
    QVariant data(const QModelIndex &item, int role=Qt::DisplayRole) const;
    bool addData();
    bool deleteData(QModelIndex &index);
    bool setData(const QModelIndex &index, const QVariant &value, int role);

    bool set_op();
    bool set_opTitle();
    bool set_opName();
    bool set_opIndex(QModelIndex &index);
    void refresh();

private:
    //private var
    int opRow;
    int opCol;

    //private function
};

#endif // MYSQLQUERYMODEL_H

```

mysqlquerymodel.cpp

```

/* FileName:mysqlquerymodel.cpp
 * Author:Hover
 * E-Mail:hover@hust.edu.cn
 * GitHub:HoverWings
 * Description:This class inherits from the basical QSqlQueryModel to make the
module can edit and autofit the UI
 */
#include "mysqlquerymodel.h"

```

```
#include <QSqlQuery>
#include <QColor>

MySQLQueryModel::MySQLQueryModel(QObject *parent) :
    QSqlQueryModel(parent)
{
    // set map
    //    map["FLIGHT"]

}

Qt::ItemFlags MySQLQueryModel::flags
(
    const QModelIndex &index) const //返回表格是否可更改的标志
{
    Qt::ItemFlags flags = QSqlQueryModel::flags(index);
    if(opView==0)
    {
        if (index.column()!=0) // value can be changed except first
            flags |= Qt::ItemIsEditable;
    }
    return flags;
}

// add data
bool MySQLQueryModel::addData()
{
    return true;
}

// update
bool MySQLQueryModel::setData(const QModelIndex &index, const QVariant
&value, int /* role */)
{
    set_op();
    QModelIndex temp=index;
    set_opIndex(temp);

    qDebug()<<"FIGHTinfo changed";
}
```

```

        if (index.column() < 1 || index.column() > opTitle.size())
            return false;
//          QModelIndex primaryKeyIndex =
 QSqlQueryModel::index(index.row(), 0);
//          int id = data(primaryKeyIndex).toInt(); //获取 id 号
//          qDebug()<<id;
//          QSqlQuery query;
//          query.prepare("update "+opName+" SET "+opTitle[opCol]+"
= :value WHERE "+opPRI+"= :pri");
//          query.bindValue(":FID", id);
        QSqlQuery query;
        QModelIndex data_index = this->index(index.row(),opPRI_col);
        QVariant data = this->data(data_index);
        qDebug()<<data;
        query.prepare("update "+opName+" SET "+opTitle[opCol]+" = :value WHERE
"+opPRI+"=:pri");
        query.bindValue(":pri", data);
        query.bindValue(":value",value);
        bool isOk = query.exec();
        qDebug() <<isOk;

        refresh();
        return true;
    }

// delete data
bool QSqlQueryModel::deleteData(QModelIndex &index)
{
    set_op();
    set_opIndex(index);
    //DELETE FROM 表名称 WHERE 列名称 = 值
    qDebug()<<opPRI;
    QSqlQuery query;
    // get pri
    QModelIndex data_index = this->index(index.row(),opPRI_col);
    QVariant data = this->data(data_index);
    qDebug()<<data;
    query.prepare("DELETE FROM "+opName+" WHERE "+opPRI+"= :pri");
    query.bindValue(":pri", data);
    bool isOk = query.exec();

```

```

        qDebug() <<"delete"<<isOk;
        refresh();
        return true;
    }

bool MySqlQueryModel::set_opTitle()
{
    QSqlQuery query;
    query.prepare("SHOW COLUMNS FROM "+opName);
    bool isOk = query.exec();
    //qDebug() <<"title"<<isOk;
    if(!isOk)
    {
        return false;
    }
    opTitle.clear();
    while (query.next())
    {
        opTitle.append(query.value(0).toString());
        // Title mapping
    }
    QSqlQuery priquery;
    priquery.prepare("SHOW COLUMNS FROM FLIGHTinfo where
`Key`='PRI'");
    isOk = priquery.exec();
    priquery.next();
    opPRI=priquery.value(0).toString();
    for (int i = 0; i < opTitle.size(); ++i)
    {
        if (opTitle.at(i) ==opPRI)
        {
            opPRI_col=i;
            break;
        }
    }
    return true;
}

void MySqlQueryModel::refresh() //更新显示
{

```

```

//qDebug() << "refresh";
switch (this->opTable)
{
case 0:
    mw->on_action_F_triggered();
    //qDebug() << "refresh";
    break;
default:
    break;
}
//    setQuery("select * from user");
//
//    setHeaderData(1, Qt::Horizontal, QObject::tr("name"));
}

//更改数据显示样式
QVariant MySqlQueryModel::data(const QModelIndex &index, int role) const
{
    QVariant value = QSqlQueryModel::data(index, role);
    //第一个字段的字体颜色为红色
    if (role == Qt::TextColorRole && index.column() == 0)
        return qVariantFromValue(QColor(Qt::green));
    return value;
}

// set op property
bool MySqlQueryModel::set_op()
{
    set_opName();
    qDebug()<<set_opTitle();
    return true;
}

bool MySqlQueryModel::set_opIndex(QModelIndex &index)
{
    opRow=index.row();

```

```
        opCol=index.column();
        return true;
    }

    bool MySqlQueryModel::set_opName()
    {
        switch(opTable)
        {
            case (0):opName="FLIGHTinfo";break;
            case (1):opName="ORDERinfo";break;
            case (2):opName="FSTATUSinfo";break;
        }
        return true;
    }
```

Addflight_dialog

addflight_dialog.h

```

/* FileName:addflight_dialog.h
 * Author:Hover
 * E-Mail:hover@hust.edu.cn
 * GitHub:HoverWings
 * Description:
 * The definition of addflight module
 */

#ifndef ADDFLIGHT_DIALOG_H
#define ADDFLIGHT_DIALOG_H
#include <QDialog>
#include <QString>
#include < QSqlQuery>
#include <QLineEdit>
#include <QComboBox>
#include <QTextEdit>
#include <QVector>
#include <QMessageBox>
#include <QDebug>
#include <QChart>

namespace Ui
{
    class addFlight_Dialog;
}

class addFlight_Dialog : public QDialog
{
    Q_OBJECT

public:
    explicit addFlight_Dialog(QWidget *parent = 0);
    ~addFlight_Dialog();

private:
    Ui::addFlight_Dialog *ui;
    bool isDateChanged=false;
    bool isTimeChanged=false;

```



```
void setFMODEL_Combox();

private slots:
    void dateChanged();
    void timeChanged();
    void on_post_Button_clicked();
};

#endif // ADDFLIGHT_DIALOG_H
```

addflight_dialog.cpp

```
/* FileName:addflight_dialog.cpp
 * Author:Hover
 * E-Mail:hover@hust.edu.cn
 * GitHub:HoverWings
 * Description:
 * The implementation of addflight module
 */
#include "addflight_dialog.h"
#include "ui_addflight_dialog.h"

addFlight_Dialog::addFlight_Dialog(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::addFlight_Dialog)
{
    ui->setupUi(this);

    // set time edit
    ui->dateEdit->setDisplayFormat("yyyy/MM/dd");
    ui->dateEdit->setCalendarPopup(true);
    ui->timeEdit->setDisplayFormat("HH:mm");

    ui->dateEdit->setDateTime(QDateTime::currentDateTime());
    ui->timeEdit->setDateTime(QDateTime::currentDateTime());

    ui->dateEdit_2->setDisplayFormat("yyyy/MM/dd");
    ui->dateEdit_2->setCalendarPopup(true);
    ui->timeEdit_2->setDisplayFormat("HH:mm");
```

```

ui->dateEdit_2->setDateTime(QDateTime::currentDateTime());
ui->timeEdit_2->setDateTime(QDateTime::currentDateTime());

isDateChanged=false;
isTimeChanged=false;

connect(ui->dateEdit,SIGNAL(dateChanged(QDate)),this,SLOT(dateChanged()));

connect(ui->timeEdit,SIGNAL(timeChanged(QTime)),this,SLOT(timeChanged()));
    setFMODEL_Combox();

    //set FSTATE_combox
    ui->FSTATE_comboBox->addItem("SCHEDULED");
    ui->FSTATE_comboBox->addItem("CANCELLED");

    //set FID
    ui->FID_lineEdit->setValidator(new QIntValidator(0, 10000, this));

    // set qss
    QFile qssfile(":/test.qss");
    qssfile.open(QFile::ReadOnly);
    QString qss;
    qss = qssfile.readAll();
    setStyleSheet(qss);

}

void addFlight_Dialog::dateChanged()
{
    isDateChanged=true;
}

void addFlight_Dialog::timeChanged()
{
    isTimeChanged=true;
}

```

```

void addFlight_Dialog::setFMODEL_Combox()
{
    // set flight combox
    QComboBox* com=ui->FMODEL_comboBox;
    com->clear();
    QSqlQuery query;
    QString opName="FMODELinfo";
    QString showItem="FMODEL";
    query.prepare("select "+showItem+" FROM "+opName+" group by
"+showItem);
    bool isOk = query.exec();
    if(!isOk)
    {
        return;
    }
    while (query.next())
    {
        com->addItem(query.value(0).toString());
    }
    //com->addItem("ALL");
    com->setCurrentIndex(-1);
}

addFlight_Dialog::~addFlight_Dialog()
{
    delete ui;
}

void addFlight_Dialog::on_post_Button_clicked()
{
    if(isDateChanged==false || isTimeChanged==false)
    {
        QMessageBox::information(NULL, "Attention!", "请设置时间！ ",
QMessageBox::Yes);
        return;
    }

    QString FDATE=ui->dateEdit->text();
    QString FTIME=ui->timeEdit->text();

```

```

QString ARDATE=ui->dateEdit_2->text();
QString ARRTIME=ui->timeEdit_2->text();
QVector<QString> vec;
QString FID=ui->FID_lineEdit->text();
QString Flight=ui->Flight_lineEdit->text();
QString FMODEL=ui->FMODEL_comboBox->currentText();
QString FSTATE=ui->FSTATE_comboBox->currentText();
QString FFROM=ui->FFROM_lineEdit->text();
QString FTO=ui->FTO_lineEdit->text();
QString FSTATUS="NOT FULL";
vec.append(FID);
vec.append(Flight);
vec.append(FMODEL);
vec.append(FSTATE);
vec.append(FFROM);
vec.append(FTO);
QVectorIterator<QString> i(vec);
while (i.hasNext())
{
    if(i.next().isEmpty()==true)
    {
        QMessageBox::information(NULL, "Please Fit the form", "请填写完整表单!", QMessageBox::Yes);
        return;
    }
}
QString query;
QString str="";
bool isOK;
str+="insert into FLIGHTinfo
values(:FID,:Flight,:FMODEL,:FSTATE,:FFROM,:FTO,:FDATE,:FTIME,:ARRDATE,:ARRTIME,:FSTATUS)";
query.prepare(str);
query.bindValue(":FID",FID);
query.bindValue(":Flight",Flight);
query.bindValue(":FMODEL",FMODEL);
query.bindValue(":FSTATE",FSTATE);
query.bindValue(":FFROM",FFROM);
query.bindValue(":FTO",FTO);
query.bindValue(":FDATE",FDATE);

```

```

query.bindValue(":FTIME",FTIME);
query.bindValue(":ARRDATE",ARRDATE);
query.bindValue(":ARRTIME",ARRTIME);
query.bindValue(":FSTATUS",FSTATUS);
isOK=query.exec();
if(!isOK)
{
    QMessageBox::information(NULL, "information", "插入失败！ 数据错误！
", QMessageBox::Yes);
    qDebug()<<"插入失败!";
    return;
}
QMessageBox::information(NULL, "information", "插入成功！ ",
QMessageBox::Yes);
//process FSTATUS info
query.clear();
str="select * from FMODELinfo where FMODEL = :FMODEL";
query.prepare(str);
query.bindValue(":FMODEL",FMODEL);
isOK=query.exec();
QString F;
QString C;
QString Y;
if(query.next())
{
    qDebug()<<query.value(0).toString();
    F=query.value(2).toString();
    qDebug()<<query.value(2).toString();
    C=query.value(3).toString();
    qDebug()<<query.value(3).toString();
    Y=query.value(4).toString();
    qDebug()<<query.value(4).toString();
}
int Frow=F.split("*")[0].toInt();
int Fcol=F.split("*")[1].toInt();
int Crow=C.split("*")[0].toInt();
int Ccol=C.split("*")[1].toInt();
int Yrow=Y.split("*")[0].toInt();
int Ycol=Y.split("*")[1].toInt();

```

```

QString SID;
query.clear();
str="insert into FSTATUSinfo values(:FID,:SID,:USABLE,:SeatRank)";
query.prepare(str);
query.bindValue(":FID",FID);
query.bindValue(":USABLE","1");
int begin;
int end;
begin=1;
end=Frow+1;
query.bindValue(":SeatRank","F");
for(int i=begin;i<end;i++)
{
    for(int j=0;j<Fcol;j++)
    {
        SID=QString::number(i,10)+QString('A'+j);
        query.bindValue(":SID",SID);
        isOK=query.exec();
        if(!isOK)
        {
            qDebug()<<SID<<"插入失败";
        }
    }
}
begin=Frow+1;
end=Frow+Crow+1;
query.bindValue(":SeatRank","C");
for(int i=begin;i<end;i++)
{
    for(int j=0;j<Ccol;j++)
    {
        SID=QString::number(i,10)+QString('A'+j);
        query.bindValue(":SID",SID);
        isOK=query.exec();
        if(!isOK)
        {
            qDebug()<<SID<<"插入失败";
        }
    }
}
}

```

```

begin=Frow+Crow+1;
end=Frow+Crow+Yrow+1;
query.bindValue(":SeatRank","Y");
for(int i=begin;i<end;i++)
{
    for(int j=0;j<Ycol;j++)
    {
        SID=QString::number(i,10)+QString('A'+j);
        query.bindValue(":SID",SID);
        isOK=query.exec();
        if(!isOK)
        {
            qDebug()<<SID<<"插入失败";
        }
    }
}
this->close();
}

```

addflight_dialog.ui

```

<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
    <class>addFlight_Dialog</class>
    <widget class="QDialog" name="addFlight_Dialog">
        <property name="geometry">
            <rect>
                <x>0</x>
                <y>0</y>
                <width>533</width>
                <height>504</height>
            </rect>
        </property>
        <property name="windowTitle">
            <string>Dialog</string>
        </property>
        <widget class="QWidget" name="gridLayoutWidget">
            <property name="geometry">
                <rect>
                    <x>110</x>
                    <y>60</y>

```

```

        <width>301</width>
        <height>363</height>
    </rect>
</property>
<layout class="QGridLayout" name="gridLayout">
    <item row="1" column="0">
        <widget class="QLabel" name="label_2">
            <property name="text">
                <string>航空公司(Flight)</string>
            </property>
        </widget>
    </item>
    <item row="4" column="1">
        <widget class="QLineEdit" name="FFROM_lineEdit"/>
    </item>
    <item row="3" column="1">
        <widget class="QComboBox" name="FSTATE_comboBox">
            <property name="editable">
                <bool>true</bool>
            </property>
        </widget>
    </item>
    <item row="6" column="1">
        <widget class="QLineEdit" name="FTO_lineEdit"/>
    </item>
    <item row="8" column="0">
        <widget class="QLabel" name="label_8">
            <property name="text">
                <string>起飞时间(FTIME)</string>
            </property>
        </widget>
    </item>
    <item row="2" column="0">
        <widget class="QLabel" name="label_3">
            <property name="text">
                <string>飞机型号(FMODEL)</string>
            </property>
        </widget>
    </item>

```



```

<item row="7" column="0">
  <widget class="QLabel" name="label_7">
    <property name="text">
      <string>起飞日期(FDATE)</string>
    </property>
  </widget>
</item>
<item row="10" column="0">
  <widget class="QLabel" name="label_10">
    <property name="text">
      <string>到达时间(ARRTIME)</string>
    </property>
  </widget>
</item>
<item row="2" column="1">
  <widget class="QComboBox" name="FMODEL_comboBox">
    <property name="editable">
      <bool>true</bool>
    </property>
  </widget>
</item>
<item row="7" column="1">
  <widget class="QDateEdit" name="dateEdit">
    <property name="displayFormat">
      <string>yyyy/mm/dd</string>
    </property>
  </widget>
</item>
<item row="1" column="1">
  <widget class="QLineEdit" name="Flight_lineEdit"/>
</item>
<item row="9" column="0">
  <widget class="QLabel" name="label_9">
    <property name="text">
      <string>到达日期(ARRDATE)</string>
    </property>
  </widget>
</item>
<item row="0" column="0">

```

```

<widget class="QLabel" name="label">
  <property name="text">
    <string>航班号(FID)</string>
  </property>
</widget>
</item>
<item row="8" column="1">
  <widget class="QTimeEdit" name="timeEdit">
    <property name="displayFormat">
      <string>hh:mm</string>
    </property>
  </widget>
</item>
<item row="6" column="0">
  <widget class="QLabel" name="label_6">
    <property name="text">
      <string>飞往(FTO)</string>
    </property>
  </widget>
</item>
<item row="3" column="0">
  <widget class="QLabel" name="label_4">
    <property name="text">
      <string>飞机状态(FSTATE)</string>
    </property>
  </widget>
</item>
<item row="4" column="0">
  <widget class="QLabel" name="label_5">
    <property name="text">
      <string>起飞地(FFROM)</string>
    </property>
  </widget>
</item>
<item row="0" column="1">
  <widget class="QLineEdit" name="FID_lineEdit"/>
</item>
<item row="9" column="1">
  <widget class="QDateEdit" name="dateEdit_2">

```

```

        <property name="displayFormat">
            <string>yyyy/mm/dd</string>
        </property>
    </widget>
</item>
<item row="10" column="1">
    <widget class="QTimeEdit" name="timeEdit_2">
        <property name="displayFormat">
            <string>hh:mm</string>
        </property>
    </widget>
</item>
<item row="11" column="1">
    <widget class="QPushButton" name="post_Button">
        <property name="text">
            <string>OK</string>
        </property>
    </widget>
</item>
</layout>
</widget>
</widget>
<tabstops>
    <tabstop>FID_lineEdit</tabstop>
    <tabstop>Flight_lineEdit</tabstop>
    <tabstop>FMODEL_comboBox</tabstop>
    <tabstop>FSTATE_comboBox</tabstop>
    <tabstop>FFROM_lineEdit</tabstop>
    <tabstop>FTO_lineEdit</tabstop>
    <tabstop>dateEdit</tabstop>
    <tabstop>timeEdit</tabstop>
    <tabstop>dateEdit_2</tabstop>
    <tabstop>timeEdit_2</tabstop>
    <tabstop>post_Button</tabstop>
</tabstops>
<resources/>
<connections/>
</ui>

```

Chooseseat_dialog

chooseseat_dialog.h

```

/* FileName:chooseseat_dialog.h
 * Author:Hover
 * E-Mail:hover@hust.edu.cn
 * GitHub:HoverWings
 * Description:The definition of chooseseat module
 */

#ifndef CHOOSESEAT_DIALOG_H
#define CHOOSESEAT_DIALOG_H

#include <QDialog>
#include <QLabel>
#include <QPushButton>
#include <QGridLayout>
#include <QDebug>
#include <QSqlDatabase>
#include <QSqlQuery>
#include <QSqlQueryModel>
#include <QList>
#include <QString>
#include "mysqlquerymodel.h"
#include <QButtonGroup>
#include <QMessageBox>
#include <QGraphicsView>

namespace Ui
{
    class chooseSeat_Dialog;
}

class chooseSeat_Dialog : public QDialog
{
    Q_OBJECT

public:
    int FID;
    void queryFSTATUS(int FID);
    class MainWindow* mw;
    explicit chooseSeat_Dialog(class MainWindow *parent = 0,int FID=-1);

```

```

~chooseSeat_Dialog();
QString seatName;

private:
    QButtonGroup *buttonGroup;
    QMap<QString, QPushButton*> pushButtonMap;
    Ui::chooseSeat_Dialog *ui;
    void setButtonGroup(int FID);
    void setImage();
    void initButton();
    void buttonJudge(int buttonId);
    class MySQLQueryModel* StatusModel=NULL;

private slots:
    void on_chooseSeat_pushButton_clicked();
};

#endif // CHOOSESEAT_DIALOG_H

```

chooseseat_dialog.cpp

```

/* FileName:chooseseat_dialog.cpp
 * Author:Hover
 * E-Mail:hover@hust.edu.cn
 * GitHub:HoverWings
 * Description:The implementation of chooseseat module
 */
#include "chooseseat_dialog.h"
#include "ui_chooseseat_dialog.h"

chooseSeat_Dialog::chooseSeat_Dialog(class MainWindow *parent,int FID) :
    QDialog(parent),
    ui(new Ui::chooseSeat_Dialog)
{
    ui->setupUi(this);
    this->FID=FID;
    qDebug()<<this->FID;
    setWindowTitle(tr("请选择座位！"));
}

```

```

setStyleSheet("background-color:white;"
              "QPushButton{"
              "background-color:white;"
              "color:black;"
              "text-align:center;"
              "border-radius: 8px;"
              "border: 2px groove gray;"
              "border-style: outset;"
              "}");
setButtonGroup(FID);
setImage();
queryFSTATUS(FID);
mw=parent;

//    //set qss
//    QFile qssfile(":/test.qss");
//    qssfile.open(QFile::ReadOnly);
//    QString qss;
//    qss = qssfile.readAll();
//    this->setStyleSheet(qss);

//setAttribute(Qt::WA_DeleteOnClose);
// set focus

}

//RESIZE!
void chooseSeat_Dialog::setImage()
{
    QString str = "select * from FMODELinfo where FMODEL= (select FMODEL
FROM FLIGHTinfo where FID= "+QString::number(FID,10)+" )";
    QSqlQuery query;
    query.exec(str);
    QPixmap photo;
    if(query.next())
    {
        QLabel *PicLabel = new QLabel();
        photo.loadFromData(query.value(1).toByteArray(), "JPG"); //从数据库中
        读出图片为二进制数据，图片格式为 JPG，然后显示到 QLabel 里
        //photo.scaledToHeight(ui->graphicsView->maximumHeight());
    }
}

```

```

        //photo.scaledToWidth(ui->graphicsView->maximumWidth());
        photo.scaled(ui->graphicsView->maximumSize());
        PicLabel->setPixmap(photo);
        PicLabel->setScaledContents(true);
    }

    QGraphicsScene *scene = new QGraphicsScene;
    scene->addPixmap(photo);
    ui->graphicsView->setScene(scene);
    ui->graphicsView->resize(photo.width()*0.5 + 10, photo.height()*0.5 + 10);
    ui->graphicsView->show();
}

void chooseSeat_Dialog::setButtonGroup(int FID)
{
    // set button group
    int totalNum=0;
    int usableNum=0;
    QGridLayout *layout = ui->gridLayout_0;
    QSqlQuery query;
    QString opName="FLIGHTinfo";
    QString showItem="FLIGHT";
    QString str="select * from FSTATUSinfo where FID =" +QString::number(FID,
10);
    query.prepare(str);
    bool isOk = query.exec();
    if(!isOk)
    {
        qDebug()<<"set seat button fail!";
        return;
    }
    ;
    QPushButton* pushbutton=NULL;
    int rowBefore=1;
    int row=1;
    int col=-1;
    int len=0;
    int usable=-1;
    while (query.next()) // when .next() then go to the next

```

```

{
    totalNum+=1;
    //qDebug()<<query.value(2).toString();
    usable=query.value(2).toString().toInt();
    len=query.value(1).toString().length();
    //qDebug()<<query.value(1).toString();
    row=query.value(1).toString().mid(0,len-1).toInt();
    if(row!=rowBefore)
    {
        col=0;
        rowBefore=row;
    }
    else
    {
        col+=1;
    }
    pushbutton = new QPushButton();
    pushbutton->setText(query.value(1).toString());
    pushButtonMap[query.value(1).toString()]=pushbutton;
    layout->addWidget(pushbutton,row,col,1,1);
    if(usable)
    {
        usableNum+=1;
        pushbutton->setCheckable(true);
        pushbutton->setAutoExclusive(true);
    }
    else
    {
        pushbutton->setCheckable(false);
        pushbutton->setStyleSheet("background-color: rgb(170, 0, 255);");
//set the background color
    }
}
qDebug()<<"totalNum"<<totalNum;
qDebug()<<"usableNum"<<usableNum;
ui->label_totalSeatNum->setText(QString::number(totalNum,10));
ui->label_usableSeatNum->setText(QString::number(usableNum,10));
float fnum=((float)totalNum-usableNum)/totalNum*100;
ui->label_fullRate->setText(QString("%1").arg(fnum)+"%");
}

```



```

chooseSeat_Dialog::~chooseSeat_Dialog()
{
    delete ui;
}

//query the choosable seat
void chooseSeat_Dialog::queryFSTATUS(int FID)
{
    StatusModel=new MySqlQueryModel(this);
    StatusModel->opTable=2;
    StatusModel->set_op();
    QString str="select * from "+StatusModel->opName+" where FID
="+QString::number(FID, 10)+" AND USABLE = 1";
    //qDebug()<<str;
    StatusModel->setQuery(str);
    StatusModel->query().bindValue(":FID",FID);
    StatusModel->query().exec();
    if(StatusModel->query().next())
    {
        qDebug()<<StatusModel->query().value(0).toString();
    }
    else
    {
        qDebug()<<"No Search Result!";
    }
    for(int i=0;i<StatusModel->opTitle.size();i++)
    {
        StatusModel->setHeaderData(i, Qt::Horizontal, StatusModel->opTitle[i]);
    }
    ui->FSTATUS_tableView->setModel(StatusModel);

    ui->FSTATUS_tableView->setSelectionBehavior(QAbstractItemView::SelectRows);
}

void chooseSeat_Dialog::on_chooseSteat_pushButton_clicked()
{
    QMapIterator<QString,QPushButton*> i(pushButtonMap);

```

```

seatName="";
while (i.hasNext())
{
//      qDebug() <<i.next().key()<<i.next().value();
      if(i.next().value()->isChecked()==true)
      {
          seatName=i.key();
          break;
      }
}
if(seatName=="")
{
    QModelIndex selIndex=ui->FSTATUS_tableView->currentIndex();
    int selRow=selIndex.row();
    //int selCol=selIndex.column();
    qDebug()<<selRow;
    QModelIndex index= StatusModel->index(selRow,1);
    QVariant data = StatusModel->data(index);
    qDebug()<<data.toString();
    seatName=data.toString();
}

QMessageBox::StandardButton rb = QMessageBox::question(NULL, "请确认座位", "您选择的座位为:"+seatName, QMessageBox::Yes | QMessageBox::No,
QMessageBox::Yes);
if(rb == QMessageBox::Yes)
{
    mw->seatName=seatName;
    this->close();
}
if(rb == QMessageBox::No)
{
    return;
}

//rb = information(NULL, "Show Qt", "Do you want to show Qt dialog?",
QMessageBox::Yes QMessageBox::No, QMessageBox::Yes);
}

```

chooseseat_dialog.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
  <class>chooseSeat_Dialog</class>
  <widget class="QDialog" name="chooseSeat_Dialog">
    <property name="geometry">
      <rect>
        <x>0</x>
        <y>0</y>
        <width>1450</width>
        <height>1000</height>
      </rect>
    </property>
    <property name="minimumSize">
      <size>
        <width>50</width>
        <height>1000</height>
      </size>
    </property>
    <property name="windowTitle">
      <string>Dialog</string>
    </property>
    <widget class="QWidget" name="layoutWidget">
      <property name="geometry">
        <rect>
          <x>0</x>
          <y>0</y>
          <width>1351</width>
          <height>1120</height>
        </rect>
      </property>
      <layout class="QHBoxLayout" name="horizontalLayout">
        <item>
          <layout class="QVBoxLayout" name="verticalLayout">
            <item>
              <widget class="QTableView" name="FSTATUS_tableView">
                <property name="minimumSize">
                  <size>
                    <width>50</width>
                    <height>200</height>
                  </size>
                </property>
              </widget>
            </item>
          </layout>
        </item>
      </layout>
    </widget>
  </widget>
</ui>
```

```

        </size>
    </property>
</widget>
</item>
<item>
    <layout class="QGridLayout" name="gridLayout_1">
        <item row="3" column="1">
            <widget class="QLabel" name="label_fullRate">
                <property name="text">
                    <string>TextLabel</string>
                </property>
            </widget>
        </item>
        <item row="3" column="0">
            <widget class="QLabel" name="label_3">
                <property name="text">
                    <string>满座率</string>
                </property>
            </widget>
        </item>
        <item row="2" column="0">
            <widget class="QLabel" name="label">
                <property name="text">
                    <string>航班剩余座位</string>
                </property>
            </widget>
        </item>
        <item row="1" column="0">
            <widget class="QLabel" name="label_2">
                <property name="text">
                    <string>航班总共座位</string>
                </property>
            </widget>
        </item>
        <item row="2" column="1">
            <widget class="QLabel" name="label_usableSeatNum">
                <property name="text">
                    <string>TextLabel</string>
                </property>
            </widget>
        </item>
    </layout>
</item>

```

```
</widget>
</item>
<item row="1" column="1">
  <widget class="QLabel" name="label_totalSeatNum">
    <property name="text">
      <string>TextLabel</string>
    </property>
  </widget>
</item>
</layout>
</item>
<item>
  <widget class="QPushButton" name="chooseSteat_pushButton">
    <property name="text">
      <string>确认选座</string>
    </property>
  </widget>
</item>
<item>
  <widget class="QGraphicsView" name="graphicsView">
    <property name="sizePolicy">
      <sizepolicy hsizeType="Preferred" vsizeType="Preferred">
        <horstretch>0</horstretch>
        <verstretch>0</verstretch>
      </sizepolicy>
    </property>
    <property name="minimumSize">
      <size>
        <width>0</width>
        <height>800</height>
      </size>
    </property>
    <property name="maximumSize">
      <size>
        <width>800</width>
        <height>600</height>
      </size>
    </property>
    <property name="autoFillBackground">
      <bool>true</bool>
    </property>
  </widget>
</item>
</layout>
</item>
</form>
</form>
```

```
        </property>
    </widget>
</item>
</layout>
</item>
<item>
    <layout class="QGridLayout" name="gridLayout_0"/>
</item>
</layout>
</widget>
</widget>
<resources/>
<connections/>
</ui>
```

Login_dialog

login_dialog.h

```

/* FileName:login_dialog.h
 * Author:Hover
 * E-Mail:hover@hust.edu.cn
 * GitHub:HoverWings
 * Description:The definition of login module
 */

#ifndef LOGIN_DIALOG_H
#define LOGIN_DIALOG_H

#include <QDialog>
#include <QDebug>
#include <QString>
#include < QSqlQuery>
#include <QVector>
#include <QMessageBox>
#include "mainwindow.h"
#include "string"

namespace Ui
{
    class login_dialog;
}

class login_dialog : public QDialog
{
    Q_OBJECT

public:
    explicit login_dialog(QWidget *parent = 0);
    QSqlDatabase* db;
    ~login_dialog();

private slots:
    void on_login_pushButton_clicked();

    void on_password_lineEdit_editingFinished();

private:

```

```

        Ui::login_dialog *ui;

};

#endif // LOGIN_DIALOG_H

login_dialog.cpp

/* FileName:login_dialog.cpp
 * Author:Hover
 * E-Mail:hover@hust.edu.cn
 * GitHub:HoverWings
 * Description:The implementation of login module
 */
#include "login_dialog.h"
#include "ui_login_dialog.h"

login_dialog::login_dialog(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::login_dialog)
{
    ui->setupUi(this);
    ui->username_lineEdit->setText("hover");
    ui->password_lineEdit->setText("123456");
    ui->password_lineEdit->setEchoMode(QLineEdit::PasswordEchoOnEdit);
    ui->login_pushButton->setFocus();
    // set qss
    QFile qssfile(":/test.qss");
    qssfile.open(QFile::ReadOnly);
    QString qss;
    qss = qssfile.readAll();
    setStyleSheet(qss);
}

login_dialog::~login_dialog()
{
    delete ui;
}

void login_dialog::on_login_pushButton_clicked()
{

```



```

QString username=ui->username_lineEdit->text();
QString password=ui->password_lineEdit->text();
QString query;
query.exec("select DBCourseDesign;");
query.prepare("select * from user where `username`=:username and
password=:password;");
query.bindValue(0, username);
query.bindValue(1, password);
query.exec();
bool isRoot=false;
if (query.next())
{
    qDebug() << query.value(3).toString();
    if(query.value(3).toInt()==1)
    {
        isRoot=true;
    }
}
else
{
    QMessageBox::warning(NULL,"警告","用户名或者密码错误! ");
    return;
}
MainWindow* w=new MainWindow(NULL,isRoot);
// set user information
w->UID=query.value(0).toInt();
w->userName=query.value(1).toString();
w->isRoot=isRoot;
w->db=this->db;
this->close();
w->show();
}

void login_dialog::on_password_lineEdit_editingFinished()
{
    ui->login_pushButton->setFocus();
}

```

login_dialog.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
  <class>login_dialog</class>
  <widget class="QDialog" name="login_dialog">
    <property name="geometry">
      <rect>
        <x>0</x>
        <y>0</y>
        <width>776</width>
        <height>435</height>
      </rect>
    </property>
    <property name="windowTitle">
      <string>Dialog</string>
    </property>
    <widget class="QLineEdit" name="username_lineEdit">
      <property name="geometry">
        <rect>
          <x>320</x>
          <y>110</y>
          <width>211</width>
          <height>27</height>
        </rect>
      </property>
    </widget>
    <widget class="QLineEdit" name="password_lineEdit">
      <property name="geometry">
        <rect>
          <x>320</x>
          <y>160</y>
          <width>211</width>
          <height>27</height>
        </rect>
      </property>
    </widget>
    <widget class="QLabel" name="label">
      <property name="geometry">
        <rect>
          <x>210</x>
```

```

        <y>110</y>
        <width>68</width>
        <height>19</height>
    </rect>
</property>
<property name="text">
    <string>userName</string>
</property>
</widget>
<widget class="QLabel" name="label_2">
    <property name="geometry">
        <rect>
            <x>210</x>
            <y>160</y>
            <width>68</width>
            <height>19</height>
        </rect>
    </property>
    <property name="text">
        <string>password</string>
    </property>
</widget>
<widget class="QLabel" name="label_3">
    <property name="geometry">
        <rect>
            <x>300</x>
            <y>40</y>
            <width>171</width>
            <height>19</height>
        </rect>
    </property>
    <property name="text">
        <string>机票预订系统</string>
    </property>
</widget>
<widget class="QPushButton" name="pushButton">
    <property name="geometry">
        <rect>
            <x>420</x>
            <y>290</y>

```

```

        <width>88</width>
        <height>27</height>
    </rect>
</property>
<property name="text">
    <string>注册</string>
</property>
</widget>
<widget class="QPushButton" name="login_pushButton">
    <property name="geometry">
        <rect>
            <x>540</x>
            <y>290</y>
            <width>88</width>
            <height>27</height>
        </rect>
    </property>
    <property name="text">
        <string>登录</string>
    </property>
</widget>
</widget>
<resources/>
<connections/>
</ui>

```

Mainwindow

mainwindow.h

```

/* FileName:mainwindow.h
 * Author:Hover
 * E-Mail:hover@hust.edu.cn
 * GitHub:HoverWings
 * Description:The definition of mainwindow slot,function and the interface with
other module
 */
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include <QSqlDatabase>
#include <QSqlQueryModel>
#include <QMessageBox>
#include <QSqlQuery>
#include <QSqlError>
#include <QDebug>
#include <QDate>
#include <QTime>
#include <QUrl>
#include <QComboBox>
#include <QWebChannel>
#include <QPrinter>
#include <QPalette>
#include <QTableWidget>
#include "mysqlquerymodel.h"
#include "chooseseat_dialog.h"
#include "addflight_dialog.h"
#include "print_dialog.h"
#include "report_dialog.h"

namespace Ui
{
    class MainWindow;
}

class MainWindow : public QMainWindow
{

```

Q_OBJECT

public:

```
int UID;
QString userName;
bool isRoot=false;
QString seatName;

QSqlDatabase* db;
explicit MainWindow(QWidget *parent = 0, bool isRoot = false);
class MySqlQueryModel* myModel=NULL;
QTableView * pOpView=NULL;
~MainWindow();
```

private:

```
Ui::MainWindow *ui;

int selRow;
int selCol;
bool isDataChanged=false;
bool isTimeChanged=false;
void setFlight_Combox();
void setTab();
void setTab2();
void transQuery();
```

```
//debug funciton
void printSQLException();
```

public slots:

```
void on_action_F_triggered();
```

private slots:

```
//void on_deleteButton_clicked();
//void on_addButton_clicked();
```

```

void on_queryButton_clicked();
void on_tabWidget_currentChanged(int index);
void on_postButton_clicked();
void on_Unsubscribe_pushButton_clicked();


void dateChanged();
void timeChanged();


void on_addFlightButton_clicked();
void on_deleteFlightButton_clicked();
void on_checkin_pushButton_clicked();
void on_pushButton_clicked();
};

#endif // MAINWINDOW_H


mainwindow.cpp
/* FileName:login_dialog.cpp
 * Author:Hover
 * E-Mail:hover@hust.edu.cn
 * GitHub:HoverWings
 * Description:The implementation of mainwindow slot,function and the interface
with other module
 */
#include "mainwindow.h"
#include "ui_mainwindow.h"

MainWindow::MainWindow(QWidget *parent,bool isRoot) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{

    ui->setupUi(this);
    //set model pointer
    MySqlQueryModel* myModel = new MySqlQueryModel(this);
    this->myModel=myModel;
    myModel->mw=this;

```

```

// set table vie selection
ui->tableView_1->setSelectionBehavior(QAbstractItemView::SelectRows);
ui->tableView_2->setSelectionBehavior(QAbstractItemView::SelectRows);

// set
ui->tableView_0->horizontalHeader()->setStretchLastSection(true);           //
最后一列补全所有空白位置
ui->tableView_0->verticalHeader()->hide();
ui->tableView_1->horizontalHeader()->setStretchLastSection(true);           //
最后一列补全所有空白位置
ui->tableView_1->verticalHeader()->hide();
ui->tableView_2->horizontalHeader()->setStretchLastSection(true);           //
最后一列补全所有空白位置
ui->tableView_2->verticalHeader()->hide();
//ui->tableView_2->horizontalHeader()->set                               //最后一列补全所有空白
位置
// set default datetime
ui->dateEdit->setDisplayFormat("yyyy/MM/dd");
ui->dateEdit->setCalendarPopup(true);
ui->timeEdit->setDisplayFormat("HH:mm");
//this->setStyleSheet("background-color:black;");

// set qss
QFile qssfile(":/test.qss");
qssfile.open(QFile::ReadOnly);
QString qss;
qss = qssfile.readAll();
setStyleSheet(qss);

//set root
this->isRoot=isRoot;
if(!isRoot)
{
    ui->addFlightButton->hide();
    ui->deleteFlightButton->hide();
}

//set map

```



```
//    QWebEnginePage *page=new QWebEnginePage(this);
//    QWebChannel *channel=new QWebChannel(this);
//    page->load(QUrl("FILE:./baidumap/demo/1_0.html"));
//    page->setWebChannel(channel);
//    ui->webEngineView->load(page);
//    ui->webEngineView->show();

//    QWebEngineView *view=new QWebEngineView(this);
//    view->load(QUrl("FILE:./baidumap/demo/1_0.html"));
//    setCentralWidget(view);
//    resize(1024,680);
//    ui->map_widget->setUrl(QUrl("https://map.baidu.com/"));
```

```
}
```

```
MainWindow::~MainWindow()
```

```
{
    delete ui;
}
```

```
void MainWindow::on_action_F_triggered() // print FLIGHTinfor 1 to view
```

```
{
    // show flight info
    myModel->opTable=0;
    myModel->set_op();
    myModel->setQuery("select * from "+myModel->opName);
    for(int i=0;i<myModel->opTitle.size();i++)
    {
        myModel->setHeaderData(i, Qt::Horizontal, myModel->opTitle[i]);
    }
}
```

```

    }
    pOpView->setModel(myModel);
    return;
}

//void MainWindow::on_deleteButton_clicked()
//{
//    qDebug()<<"delete!!!";
//    //myModel->set_op();
//    //myModel->set_opIndex(index);
//    QModelIndex selIndex=pOpView->currentIndex();
//    selRow=selIndex.row();
//    selCol=selIndex.column();
//    myModel->deleteData(selIndex);
//}

//void MainWindow::on_addButton_clicked()
//{
//    myModel->set_op();
//    QSqlQuery query;
//    query.prepare("insert into "+myModel->opName+" values()");
//    bool isOk = query.exec();
//    if(!isOk)
//    {
//        return;
//    }
//    myModel->refresh();
//}

//Query by the condition
void MainWindow::on_queryButton_clicked()
{
    if(ui->checkBox_2->isChecked())
    {
        transQuery();
    }
    QString str;
    str="select * from "+myModel->opName+" as f where ";
    if(ui->FFROM_lineEdit->text().isEmpty())
    {

```

```

        QMessageBox::about(NULL, "Attention!", "请输入起始地! ");
        return;
    }
    else
    {
        // the first condition and don't need 'and'
        qDebug()<<"起始地"<<ui->FFROM_lineEdit->text();
        str+="FFROM =";
        str+=" ";
        str+=ui->FFROM_lineEdit->text();
        str+=" ";
    }
    if(ui->FTO_lineEdit->text().isEmpty())
    {
        QMessageBox::about(NULL, "Attention!", "请输入到达地! ");
        return;
    }
    else
    {
        qDebug()<<"到达地"<<ui->FTO_lineEdit->text();
        str+=" and ";
        str+="FTO =";
        str+=" ";
        str+=ui->FTO_lineEdit->text();
        str+=" ";
    }
    if(isDataChanged)
    {
        QDate date = ui->dateEdit->date();
        QString dateStr=date.toString("yyyy-MM-dd");
        qDebug()<<dateStr;
        str+=" and ";
        str+="FDATE =";
        str+=" ";
        str+=dateStr;
        str+=" ";
    }
    if(isTimeChanged) // the flight time should before the time

```

```

{
    QTime time = ui->timeEdit->time();
    QString timeStr=time.toString("HH:mm:ss");
    qDebug()<<timeStr;
    str+=" and ";
    str+="FTIME <=";
    str+=" ";
    str+=timeStr;
    str+=" ";
}
if(ui->FLIGHT_comboBox->currentText()!="ALL")
{
    QString Flight=ui->FLIGHT_comboBox->currentText();
    qDebug()<<"FLIGHT:"<<Flight;
    str+=" and ";
    str+="Flight =";
    str+=" ";
    str+=Flight;
    str+=" ";
}
if(ui->order_comboBox->currentText()=="最近")
{
    //str+=" and ";
    str+="order by ";
    //str+=" ";
    str+="FDATE";
    //str+=" ";
}
//query.prepare("select * from `FLIGHTinfo` as f where exists(select * from
`FSTATUSinfo` as s where s.`FID`=f.`FID` and s.`USABLE`=1) and f.`FID`=3;");
//str+=" and ";
//str+=" exists( select * from `FSTATUSinfo` where
`FSTATUSinfo`.`FID`=`FLIGHTinfo`.`FID`)"
//and FSTATUSinfo.USABLE=1
myModel->setQuery(str);
for(int i=0;i<myModel->opTitle.size();i++)
{
    myModel->setHeaderData(i, Qt::Horizontal, myModel->opTitle[i]);
}
pOpView->setModel(myModel);

```

```

        return;
    }

void MainWindow::on_tabWidget_currentChanged(int index)
{
    myModel->clear();
    switch(index)
    {
        case 0:
        {
            myModel->opTable=0; //FLIGHTinfo
            myModel->opView=0;  //add and delete
            pOpView=ui->tableView_0;
            setTab();
            break;
        }
        case 1:
        {
            myModel->opTable=0;
            myModel->opView=1;
            myModel->set_op();
            pOpView=ui->tableView_1;
            setFlight_Combox();
            break;
        }
        case 2:
        {
            //order table
            myModel->opTable=1;
            myModel->opView=2;
            myModel->set_op();
            pOpView=ui->tableView_2;
            setTab2();
            break;
        }
        case 3:
        {
            //bill table
            QSqlQuery query;
            QString str="";

```

```

        QString viewName=userName+"_ORDER";
        qDebug()<<viewName;
        qDebug()<<UID;
        str="create or replace VIEW "+ viewName +" AS select OID,
FLIGHTinfo.FID , SID , FFROM, FTO, FDATE-DAY(1) as 'pick up time', FDATE,
FTIME , OSTATE FROM FLIGHTinfo , ORDERinfo where
FLIGHTinfo.FID=ORDERinfo.FID and OSTATE!='Unsubscribe' AND UID =
"+QString::number(UID,10);
        query.prepare(str);
        bool isOK=query.exec();
        if(!isOK)
        {
            qDebug()<<"create user-order fail!";
            return;
        }
        myModel->opName=viewName;
        myModel->set_opTitle();
        pOpView=ui->tableView_3;
        setTab();
        break;
    }
}
}

```

```

void MainWindow::transQuery()
{
    QSqlQuery query;

}

```

```

void MainWindow::setTab()
{
    //set title
    //    QPalette palette;
    //    palette.setBrush(pOpView->backgroundRole(), Qt::black);
    //    pOpView->setPalette(palette);
    myModel->clear();
    myModel->setQuery("select * from "+myModel->opName);
}

```

```

for(int i=0;i<myModel->opTitle.size();i++)
{
    myModel->setHeaderData(i, Qt::Horizontal, myModel->opTitle[i]);
}
pOpView->setModel(myModel);
}

void MainWindow::setTab2()
{
    myModel->clear();
    myModel->setQuery("select * from "+myModel->opName+" where UID
="+QString::number(UID,10));
    for(int i=0;i<myModel->opTitle.size();i++)
    {
        myModel->setHeaderData(i, Qt::Horizontal, myModel->opTitle[i]);
    }
    pOpView->setModel(myModel);
}

void MainWindow::dateChanged()
{
    isDataChanged=true;
}

void MainWindow::timeChanged()
{
    isTimeChanged=true;
}

void MainWindow::setFlight_Combox()
{
    // set time edit
    ui->dateEdit->setDateTime(QDateTime::currentDateTime());
    ui->timeEdit->setDateTime(QDateTime::currentDateTime());
    isDataChanged=false;
    isTimeChanged=false;

connect(ui->dateEdit,SIGNAL(dateChanged(QDate)),this,SLOT(dateChanged()));

```

```
connect(ui->timeEdit,SIGNAL(timeChanged(QTime)),this,SLOT(timeChanged()));
```

```

// set flight combox
QComboBox* com=ui->FLIGHT_comboBox;
com->clear();
QString query;
QString opName="FLIGHTinfo";
QString showItem="FLIGHT";
query.prepare("select "+showItem+" FROM "+opName+" group by
"+showItem);
bool isOk = query.exec();
if(!isOk)
{
    return;
}
while (query.next())
{
    com->addItem(query.value(0).toString());
}
com->addItem("ALL");
com->setCurrentIndex(-1);
}

```

```

//QString MainWindow::getFlight()
//{
//    return ui->FFROM_lineEdit->text();
//}

```

```

void MainWindow::on_postButton_clicked()
{
    QString opName="ORDERinfo";
    seatName="";
    //    if (res == QDialog::Accepted)
    //    {
    //        QMessageBox::information(this, "INFORMATION", "You clicked OK

```



```

button!");
//    }
//    if (res == QDialog::Rejected)
//    {
//        QMessageBox::information(this, "INFORMATION", "You clicked
CANCEL button!");

//    }
    qDebug()<<seatName;
    qDebug()<<myModel->opView;
    int row= pOpView->currentIndex().row();
    if(row==-1)
    {
        QMessageBox::information(NULL, "订单错误", "请先查询",
QMessageBox::Yes, QMessageBox::Yes);
        return;
    }
    QAbstractItemModel *model =pOpView->model();
    QVector<QString> chsVecs;
    //int row=0;
    for(int i=0;i<myModel->opTitle.size();i++)
    {
        QModelIndex index= model->index(row,i);//选中行第一列的内容
        QVariant data = model->data(index);
        qDebug()<<data.toString();
        chsVecs.append(data.toString());
    }
    int FID=chsVecs[0].toInt();
    int UID=this->UID;

    // process choose seat;
    chooseSeat_Dialog* chooseSeatD=new chooseSeat_Dialog(this,FID);
    chooseSeatD->exec(); // to get the seat id

    // close the window without choose the seat
    if(seatName=="")
    {
        QMessageBox::information(NULL, "订单错误！ ", "未选择座位",
QMessageBox::Yes, QMessageBox::Yes);

```

```

        return;
    }
    QString OTIME =QDateTime::currentDateTime().toString("yyyy-MM-dd
hh:mm:ss");
    qDebug()<<OTIME;
    QString OSTATE="unfinish";
    QSqlQuery query;
    query.prepare("insert into "+opName+" (FID, UID,SID, OTIME, OSTATE)
VALUES(:FID,:UID,:SID,:OTIME,:OSTATE)");
    query.bindValue(":FID", FID);
    query.bindValue(":UID", UID);
    query.bindValue(":SID", seatName);
    query.bindValue(":OTIME", OTIME);
    query.bindValue(":OSTATE", OSTATE);
    bool isOK1 = query.exec();
    // decrease seat
    query.clear();
    opName="FSTATUSinfo";
    query.prepare("update "+opName+" set USABLE = 0 where SID = :SID ");
    query.bindValue(":SID",seatName);
    bool isOk2 = query.exec();
    if(isOK1&&isOk2)
    {
        setTab();
        qDebug()<<"预订成功";
    }
    else
    {
        // roll backs
        printSQLError();
        qDebug()<<"预订失败";
        return;
    }
}

// debug function
void MainWindow::printSQLError()
{

    QSqlDatabase* db=(this->db);

```

```

QSqlError error;
error=db->lastError();
qDebug()<<error;
if(error.isValid())//发生错误时 isValid()返回 true
{
    switch (error.type())
    {
        case QSqlError::NoError:
            qDebug()<<"无错误";
            break;
        case QSqlError::ConnectionError://连接错误
            qDebug()<<error.text();
            break;
        case QSqlError::StatementError://语句错误
            qDebug()<<error.text();
            break;
        case QSqlError::TransactionError://事务错误
            qDebug()<<error.text();
            break;
        default://未知错误
            qDebug()<<error.text();
            break;
    }
}

}

void MainWindow::on_Unsubscribe_pushButton_clicked()
{
    QString orderstr="";
    int row= pOpView->currentIndex().row();
    QAbstractItemModel *model =pOpView->model ();
    QVector<QString> chsVecs;
    for(int i=0;i<myModel->opTitle.size();i++)
    {
        QModelIndex index= model->index(row,i);//选中行第一列的内容
        QVariant data = model->data(index);
        qDebug()<<data.toString();
    }
}

```

```

        chsVecs.append(data.toString());
    }
    int OID=chsVecs[0].toInt();
    QString OSTATE=chsVecs.last();
    if(OSTATE=="Unsubscribe")
    {
        //QMessageBox::StandardButton rb = QMessageBox::information(NULL, "
退订错误! ", "订单状态错误", QMessageBox::Yes, QMessageBox::Yes);

        QMessageBox::information(NULL, "退订错误! ", "订单状态错误",
QMessageBox::Yes, QMessageBox::Yes);
    }
    //qDebug()<<chsVecs;
    //QMessageBox::StandardButton rb = QMessageBox::question(NULL, "订单退
订! ", "你确定要退订订单:"+QString::number(OID,10)+"从"+FFROM+"飞往
"+FTO+" 于 "+FTIME, QMessageBox::Yes | QMessageBox::No,
QMessageBox::Yes);
    QMessageBox::StandardButton rb = QMessageBox::question(NULL, "订单退
订! ", "你确定要退订订单:"+QString::number(OID,10), QMessageBox::Yes |
QMessageBox::No, QMessageBox::Yes);
    if(rb == QMessageBox::Yes)
    {
        //process ORDERinfo
        QSqlQuery query;
        //qDebug()<<"opName:"<<myModel->opName;
        query.prepare("update "+myModel->opName+" set OSTATE = :OSTATE
where OID = :OID ");
        query.bindValue(":OSTATE","Unsubscribe");
        query.bindValue(":OID",OID);
        bool isOk1 = query.exec();

        // process the FSTATUS
        query.clear();
        QString opName="FSTATUSinfo";
        query.prepare("update "+opName+" set USABLE = 1 where SID = (select
SID from ORDERinfo where OID= :OID) ");
        query.bindValue(":OID",OID);
        bool isOk2 = query.exec();
        if(isOk1&&isOk2)
    
```

```

    {
        QMessageBox::about(NULL, "Attention", "退订成功");
        setTab();
        qDebug()<<"退订成功";
        return;
    }
    else
    {
        // roll back
        QMessageBox::about(NULL, "Attention", "退订失败");
        qDebug()<<"删除失败";
        return;
    }
}
}

```

```

void MainWindow::on_addFlightButton_clicked()
{
    addFlight_Dialog* addflightdialog=new addFlight_Dialog(NULL);
    addflightdialog->exec();
}

```

```

void MainWindow::on_deleteFlightButton_clicked()
{
    int row= pOpView->currentIndex().row();
    QAbstractItemModel *model =pOpView->model();
    QVector<QString> chsVecs;
    //int row=0;
    for(int i=0;i<myModel->opTitle.size();i++)
    {
        QModelIndex index= model->index(row,i);//选中行第一列的内容
        QVariant data = model->data(index);
        qDebug()<<data.toString();
        chsVecs.append(data.toString());
    }
}

```

```

int FID=chsVecs[0].toInt();
int UID=this->UID;
QMessageBox::StandardButton rb = QMessageBox::question(NULL, "航班删除!", "你确定要删除航班:"+QString::number(FID,10), QMessageBox::Yes |
QMessageBox::No, QMessageBox::Yes);
if(rb == QMessageBox::Yes)
{
    QSqlQuery query;
    query.prepare("delete from FLIGHTinfo where FID=:FID");
    query.bindValue(":FID",FID);
    bool isOK=query.exec();
    if(isOK)
    {
        QMessageBox::information(this, "INFORMATION", "删除成功!");

        query.clear();
        QString str="delete from FSTATUSinfo where FID = :FID";
        query.prepare(str);
        query.bindValue(":FID",FID);
        isOK=query.exec();
        qDebug()<<"删除座位"<<isOK;
        setTab();
    }
    else
    {
        QMessageBox::information(this, "INFORMATION", "删除失败!存在
顾客订单，无法删除");
    }
}
else
{
    return;
}
}

void MainWindow::on_checkin_pushButton_clicked()

```

```

{
    QString orderstr="";

    int row= pOpView->currentIndex().row();
    QAbstractItemModel *model =pOpView->model ();
    QVector<QString> chsVecs;
    for(int i=0;i<myModel->opTitle.size();i++)
    {
        QModelIndex index= model->index(row,i);//选中行第一列的内容
        QVariant data = model->data(index);
        qDebug()<<data.toString();
        chsVecs.append(data.toString());
    }
    int OID=chsVecs[0].toInt();
    int FID=chsVecs[1].toInt();

    QDateTime now= QDateTime::currentDateTime();
    QDateTime before=now.addDays(-1);
    //bill table
    QSqlQuery query;
    query.prepare("select FDATE, FTIME from FLIGHTinfo where FID =:FID");
    query.bindValue(":FID",FID);
    query.exec();
    QDateTime FDT;
    if(query.next())
    {
        QDate FDATE=query.value(0).toDate();
        QTime FTIME=query.value(1).toTime();
        //FDT=QDateTime(FDATE)+QDate(FTIME.QTime);
        FDT=QDateTime(FDATE);
    }
    if(!(FDT>before&&FDT<now))
    {
        QMessageBox::information(NULL, "进站错误! ", "时间错误无法进站! ",
QMessageBox::Yes, QMessageBox::Yes);
        return;
    }

    QString OSTATE=chsVecs.last();
    if(OSTATE=="Unsubscribe" or OSTATE=="check-in" )

```

```

{
    //QMessageBox::StandardButton rb = QMessageBox::information(NULL, "
退订错误! ", "订单状态错误", QMessageBox::Yes, QMessageBox::Yes);

    QMessageBox::information(NULL, "进站错误! ", "进站状态错误",
QMessageBox::Yes, QMessageBox::Yes);
    return;
}
//qDebug()<<chsVecs;
//QMessageBox::StandardButton rb = QMessageBox::question(NULL, "订单退
订! ", "你确定要退订订单:"+QString::number(OID,10)+"从"+FFROM+"飞往
"+FTO+" 于 "+FTIME, QMessageBox::Yes | QMessageBox::No,
QMessageBox::Yes);
    QMessageBox::StandardButton rb = QMessageBox::question(NULL, "进站! ", "
你进站:"+QString::number(OID,10), QMessageBox::Yes | QMessageBox::No,
QMessageBox::Yes);
    if(rb == QMessageBox::Yes)
    {
        //process ORDERinfo
        QSqlQuery query;
        //qDebug()<<"opName:"<<myModel->opName;
        query.prepare("update "+myModel->opName+" set OSTATE = :OSTATE
where OID = :OID ");
        query.bindValue(":OSTATE","check-in");
        query.bindValue(":OID",OID);
        bool isOk1 = query.exec();
        // process the FSTATUS
        if(!isOk1)
        {
            // roll back
            QMessageBox::about(NULL, "Attention", "进站失败");
            qDebug()<<"进站失败";
            return;
        }

        // print pdf
        bool isOK;
        QString SID;
    }
}

```



```

QString SeatRank;
QVector<QString> pV;
pV.append(userName);
//SID
query.clear();
query.prepare("select * from ORDERinfo where OID =:OID ");
query.bindValue(":OID",OID);
isOK=query.exec();
if(!isOK)
{
    qDebug()<<"SID 查询错误! ";
}
if(query.next())
{
    SID=query.value(3).toString();
    pV.append(SID);
}

//seat rank
query.clear();
query.prepare("select SeatRank from FSTATUSinfo where SID = :SID and
FID=:FID ");
query.bindValue(":SID",SID);
query.bindValue(":FID",FID);
isOK=query.exec();
if(!isOK)
{
    qDebug()<<"seat rank 查询错误! ";
}
if(query.next())
{
    SeatRank=query.value(0).toString();
    pV.append(SeatRank);
}

//Flightinfo
query.clear();
query.prepare("select * from FLIGHTinfo where FID =:FID ");

```

```

        query.bindValue(":FID",FID);
        query.exec();
        if(query.next())
        {
            for(int i=0;i<10;i++)
            {
                pV.append(query.value(i).toString());
            }
        }

        // TICKET time
        pV.append(QDateTime::currentDateTime().toString("yyyy-MM-dd
hh:mm:ss"));

        print_Dialog* printdialog=new print_Dialog(pV,this);
        //set printdialog
        printdialog->exec();

        QMessageBox::about(NULL, "Attention", "进站成功");
        setTab();
        qDebug()<<"进站成功";
        return;
    }
}

void MainWindow::on_pushButton_clicked()
{
    report_Dialog* reportdialog=new report_Dialog(this);
    reportdialog->exec();
}

```

mainwindow.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
  <class>MainWindow</class>
  <widget class="QMainWindow" name="MainWindow">
    <property name="geometry">
      <rect>
        <x>0</x>
        <y>0</y>
        <width>1458</width>
        <height>679</height>
      </rect>
    </property>
    <property name="windowTitle">
      <string>MainWindow</string>
    </property>
    <widget class="QWidget" name="centralWidget">
      <widget class="QTabWidget" name="tabWidget">
        <property name="geometry">
          <rect>
            <x>0</x>
            <y>0</y>
            <width>1401</width>
            <height>601</height>
          </rect>
        </property>
        <property name="currentIndex">
          <number>1</number>
        </property>
        <widget class="QWidget" name="tab_0">
          <attribute name="title">
            <string>航班</string>
          </attribute>
          <widget class="QPushButton" name="addFlightButton">
            <property name="geometry">
              <rect>
                <x>70</x>
                <y>450</y>
                <width>88</width>
                <height>27</height>
              </rect>
            </property>
          </widget>
        </widget>
      </widget>
    </widget>
  </widget>
</ui>
```

```

    </rect>
</property>
<property name="text">
    <string>add</string>
</property>
</widget>
<widget class="QPushButton" name="deleteFlightButton">
    <property name="geometry">
        <rect>
            <x>190</x>
            <y>450</y>
            <width>88</width>
            <height>27</height>
        </rect>
    </property>
    <property name="text">
        <string>Delete</string>
    </property>
</widget>
<widget class="QTableView" name="tableView_0">
    <property name="geometry">
        <rect>
            <x>40</x>
            <y>30</y>
            <width>1051</width>
            <height>321</height>
        </rect>
    </property>
</widget>
<widget class="QPushButton" name="pushButton">
    <property name="geometry">
        <rect>
            <x>360</x>
            <y>450</y>
            <width>151</width>
            <height>27</height>
        </rect>
    </property>
    <property name="text">
        <string>show Report</string>
    </property>

```

```

        </property>
    </widget>
</widget>
<widget class="QWidget" name="tab_1">
    <attribute name="title">
        <string>航班查询</string>
    </attribute>
    <widget class="QTableView" name="tableView_1">
        <property name="geometry">
            <rect>
                <x>360</x>
                <y>60</y>
                <width>931</width>
                <height>321</height>
            </rect>
        </property>
    </widget>
    <widget class="QPushButton" name="postButton">
        <property name="geometry">
            <rect>
                <x>740</x>
                <y>420</y>
                <width>88</width>
                <height>27</height>
            </rect>
        </property>
        <property name="text">
            <string>提交订单</string>
        </property>
    </widget>
<widget class="QWidget" name="gridLayoutWidget">
    <property name="geometry">
        <rect>
            <x>50</x>
            <y>70</y>
            <width>291</width>
            <height>311</height>
        </rect>
    </property>
    <layout class="QGridLayout" name="gridLayout">

```

```

<item row="3" column="1">
  <widget class="QTimeEdit" name="timeEdit"/>
</item>
<item row="0" column="1">
  <widget class="QLineEdit" name="FFROM_lineEdit"/>
</item>
<item row="2" column="0">
  <widget class="QLabel" name="label_3">
    <property name="text">
      <string>起飞日期</string>
    </property>
  </widget>
</item>
<item row="1" column="1">
  <widget class="QLineEdit" name="FTO_lineEdit">
    <property name="text">
      <string/>
    </property>
  </widget>
</item>
<item row="3" column="0">
  <widget class="QLabel" name="label_4">
    <property name="text">
      <string>起飞 时间</string>
    </property>
  </widget>
</item>
<item row="4" column="0">
  <widget class="QLabel" name="label_5">
    <property name="text">
      <string>航空公司</string>
    </property>
  </widget>
</item>
<item row="6" column="0" colspan="2">
  <widget class="QPushButton" name="queryButton">
    <property name="text">
      <string>查询</string>
    </property>
  </widget>
</item>

```

```

</widget>
</item>
<item row="4" column="1">
  <widget class="QComboBox" name="FLIGHT_comboBox">
    <property name="editable">
      <bool>true</bool>
    </property>
  </widget>
</item>
<item row="1" column="0">
  <widget class="QLabel" name="label_2">
    <property name="text">
      <string>到达地</string>
    </property>
  </widget>
</item>
<item row="0" column="0">
  <widget class="QLabel" name="label">
    <property name="text">
      <string>出发地</string>
    </property>
  </widget>
</item>
<item row="5" column="0">
  <widget class="QLabel" name="label_6">
    <property name="text">
      <string>Order By</string>
    </property>
  </widget>
</item>
<item row="5" column="1">
  <widget class="QComboBox" name="order_comboBox">
    <item>
      <property name="text">
        <string>最近</string>
      </property>
    </item>
    <item>
      <property name="text">

```

```

        <string>价格</string>
    </property>
</item>
</widget>
</item>
<item row="2" column="1">
    <widget class="QDateEdit" name="dateEdit"/>
</item>
</layout>
</widget>
<widget class="QCheckBox" name="checkBox_2">
    <property name="geometry">
        <rect>
            <x>140</x>
            <y>450</y>
            <width>93</width>
            <height>25</height>
        </rect>
    </property>
    <property name="text">
        <string>是否中转</string>
    </property>
</widget>
<widget class="QWidget" name="layoutWidget">
    <property name="geometry">
        <rect>
            <x>60</x>
            <y>480</y>
            <width>201</width>
            <height>29</height>
        </rect>
    </property>
    <layout class="QHBoxLayout" name="horizontalLayout">
        <item>
            <widget class="QLabel" name="label_8">
                <property name="text">
                    <string> 中转时间</string>
                </property>
            </widget>

```



```

</item>
<item>
  <widget class="QLineEdit" name="transtime_lineEdit"/>
</item>
</layout>
</widget>
</widget>
<widget class="QWidget" name="tab_2">
  <attribute name="title">
    <string>订单管理</string>
  </attribute>
  <widget class="QTableView" name="tableView_2">
    <property name="geometry">
      <rect>
        <x>80</x>
        <y>10</y>
        <width>861</width>
        <height>401</height>
      </rect>
    </property>
  </widget>
  <widget class="QPushButton" name="checkin_pushButton">
    <property name="geometry">
      <rect>
        <x>530</x>
        <y>450</y>
        <width>88</width>
        <height>27</height>
      </rect>
    </property>
    <property name="text">
      <string>check-in</string>
    </property>
  </widget>
  <widget class="QPushButton" name="Unsubscribe_pushButton">
    <property name="geometry">
      <rect>
        <x>180</x>
        <y>450</y>
        <width>88</width>

```

```

        <height>27</height>
    </rect>
</property>
<property name="text">
    <string>Unsubscribe</string>
</property>
</widget>
</widget>
<widget class="QWidget" name="tab_3">
    <attribute name="title">
        <string>账单查询</string>
    </attribute>
    <widget class="QTableView" name="tableView_3">
        <property name="geometry">
            <rect>
                <x>10</x>
                <y>60</y>
                <width>771</width>
                <height>371</height>
            </rect>
        </property>
    </widget>
    <widget class="QLabel" name="label_7">
        <property name="geometry">
            <rect>
                <x>20</x>
                <y>30</y>
                <width>241</width>
                <height>19</height>
            </rect>
        </property>
        <property name="text">
            <string>The Recent Flight</string>
        </property>
    </widget>
    <widget class="QWebEngineView" name="webEngineView">
        <property name="geometry">
            <rect>
                <x>810</x>
                <y>160</y>
            </rect>
        </property>
    </widget>
</widget>

```

```

        <width>300</width>
        <height>200</height>
    </rect>
</property>
<property name="url">
    <url>
        <string>about:blank</string>
    </url>
</property>
</widget>
</widget>
<widget class="QWidget" name="tab">
    <attribute name="title">
        <string>报表</string>
    </attribute>
    <widget class="QComboBox" name="FLIGHT_comboBox_4">
        <property name="geometry">
            <rect>
                <x>80</x>
                <y>450</y>
                <width>219</width>
                <height>27</height>
            </rect>
        </property>
        <property name="editable">
            <bool>true</bool>
        </property>
    </widget>
    <widget class="QGraphicsView" name="graphicsView">
        <property name="geometry">
            <rect>
                <x>80</x>
                <y>50</y>
                <width>731</width>
                <height>321</height>
            </rect>
        </property>
    </widget>
</widget>
</widget>

```

```

</widget>
<widget class="QMenuBar" name="menuBar">
  <property name="geometry">
    <rect>
      <x>0</x>
      <y>0</y>
      <width>1458</width>
      <height>24</height>
    </rect>
  </property>
  <widget class="QMenu" name="menu_F">
    <property name="title">
      <string>航班信息(&F)</string>
    </property>
    <addaction name="separator"/>
    <addaction name="action_F"/>
  </widget>
  <widget class="QMenu" name="menu_O">
    <property name="title">
      <string>订单管理(&O)</string>
    </property>
    <addaction name="action_Q"/>
    <addaction name="action_L"/>
    <addaction name="action_G"/>
    <addaction name="action_B"/>
  </widget>
  <widget class="QMenu" name="menu_T">
    <property name="title">
      <string>账单查询(&T)</string>
    </property>
  </widget>
  <addaction name="menu_F"/>
  <addaction name="menu_O"/>
  <addaction name="menu_T"/>
</widget>
<widget class="QToolBar" name="mainToolBar">
  <attribute name="toolBarArea">
    <enum>TopToolBarArea</enum>
  </attribute>

```

```

    <attribute name="toolBarBreak">
        <bool>false</bool>
    </attribute>
</widget>
<widget class="QStatusBar" name="statusBar"/>
<action name="action_F">
    <property name="text">
        <string>航班信息(&F)</string>
    </property>
</action>
<action name="action_Q">
    <property name="text">
        <string>航班查询(&Q)</string>
    </property>
</action>
<action name="action_G">
    <property name="text">
        <string>取票(&G)</string>
    </property>
</action>
<action name="action_B">
    <property name="text">
        <string>退订机票(&B)</string>
    </property>
</action>
<action name="action_L">
    <property name="text">
        <string>订单查询(&L)</string>
    </property>
</action>
</widget>
<layoutdefault spacing="6" margin="11"/>
<customwidgets>
    <customwidget>
        <class>QWebEngineView</class>
        <extends>QWidget</extends>
        <header location="global">QtWebEngineWidgets/QWebEngineView</header>
    </customwidget>
</customwidgets>

```

```

<tabstops>
  <tabstop>tabWidget</tabstop>
  <tabstop>FFROM_lineEdit</tabstop>
  <tabstop>FTO_lineEdit</tabstop>
  <tabstop>dateEdit</tabstop>
  <tabstop>timeEdit</tabstop>
  <tabstop>FLIGHT_comboBox</tabstop>
  <tabstop>order_comboBox</tabstop>
  <tabstop>queryButton</tabstop>
  <tabstop>postButton</tabstop>
  <tabstop>tableView_0</tabstop>
  <tabstop>addFlightButton</tabstop>
  <tabstop>deleteFlightButton</tabstop>
  <tabstop>tableView_2</tabstop>
  <tabstop>tableView_1</tabstop>
  <tabstop>checkin_pushButton</tabstop>
  <tabstop>Unsubscribe_pushButton</tabstop>
  <tabstop>tableView_3</tabstop>
</tabstops>
<resources/>
<connections/>
</ui>

```

Print_dialog

print_dialog.h

```

/* FileName:print_dialog.h
 * Author:Hover
 * E-Mail:hover@hust.edu.cn
 * GitHub:HoverWings
 * Description:The declaration of print module
 */
#ifndef PRINT_DIALOG_H
#define PRINT_DIALOG_H

#include <QDialog>
#include <QString>
#include <QPrinter>
#include <QPixmap>
#include <QPainter>

namespace Ui {
class print_Dialog;
}

class print_Dialog : public QDialog
{
    Q_OBJECT

public:
    explicit print_Dialog(QVector<QString> pV,QWidget *parent = 0);
    QVector<QString> apV;
    ~print_Dialog();

private slots:
    void on_pushButton_clicked();

private:
    Ui::print_Dialog *ui;
};

#endif // PRINT_DIALOG_H

```

print_dialog.cpp

```

/* FileName:print_dialog.cpp
 * Author:Hover
 * E-Mail:hover@hust.edu.cn
 * GitHub:HoverWings
 * Description:The implementation of print module and call the system API to print
the Ticket to PDF file simulatedly
 */
#include "print_dialog.h"
#include "ui_print_dialog.h"

print_Dialog::print_Dialog(QVector<QString> pV,QWidget *parent) :
    QDialog(parent),
    ui(new Ui::print_Dialog)
{

    ui->setupUi(this);
    apV=pV;
    ui->user_label->setText(pV[0]);
    ui->SID_label->setText(pV[1]);
    ui->SeatRank_label->setText(pV[2]);
    ui->FID_label->setText(pV[3]);
    ui->Flight_label->setText(pV[4]);
    ui->FMODEL_label->setText(pV[5]);
    ui->FSTATE_label->setText(pV[6]);
    ui->FFROM_label->setText(pV[7]);
    ui->FTO_label->setText(pV[8]);
    ui->FDATE_label->setText(pV[9]);
    ui->FTIME_label->setText(pV[10]);
    ui->ARRDATE_label->setText(pV[11]);
    ui->ARRTIME_label->setText(pV[12]);
    ui->TIAKCTIME_label->setText(pV[13]);
}

print_Dialog::~~print_Dialog()
{
    delete ui;
}

void print_Dialog::on_pushButton_clicked()

```



```
{
    ui->pushButton->hide();
    QPainter printer_pixmap(QPrinter::HighResolution);
    printer_pixmap.setPageSize(QPrinter::A4); //设置纸张大小为 A4
    printer_pixmap.setOutputFormat(QPrinter::PdfFormat); //设置输出格式为 pdf
    printer_pixmap.setOutputFileName("./tickets/"+apV[0]+"_"+apV[3]+".pdf");
    //设置输出路径

    QPixmap pixmap = QPixmap::grabWidget(this, this->rect()); //获取界面的图
    片

    QPainter painter_pixmap;
    painter_pixmap.begin(&printer_pixmap);
    QRect rect = painter_pixmap.viewport();
    int multiple = rect.width()/pixmap.width();
    painter_pixmap.scale(multiple, multiple); //将图像(所有要画的东西)在 pdf 上
    放大 multiple-1 倍

    painter_pixmap.drawPixmap(0, 0, pixmap); //画图
    painter_pixmap.end();
    this->close();
}
```

print_dialog.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
    <class>print_Dialog</class>
    <widget class="QDialog" name="print_Dialog">
        <property name="geometry">
            <rect>
                <x>0</x>
                <y>0</y>
                <width>615</width>
                <height>625</height>
            </rect>
        </property>
        <property name="windowTitle">
            <string>Dialog</string>
        </property>
```

```

<widget class="QWidget" name="gridLayoutWidget">
  <property name="geometry">
    <rect>
      <x>150</x>
      <y>100</y>
      <width>301</width>
      <height>363</height>
    </rect>
  </property>
  <layout class="QGridLayout" name="gridLayout">
    <item row="14" column="0">
      <widget class="QLabel" name="label_13">
        <property name="text">
          <string>出票时间</string>
        </property>
      </widget>
    </item>
    <item row="6" column="1">
      <widget class="QLabel" name="FSTATE_label">
        <property name="text">
          <string>TextLabel</string>
        </property>
      </widget>
    </item>
    <item row="7" column="1">
      <widget class="QLabel" name="FFROM_label">
        <property name="text">
          <string>TextLabel</string>
        </property>
      </widget>
    </item>
    <item row="13" column="1">
      <widget class="QLabel" name="ARRTIME_label">
        <property name="text">
          <string>TextLabel</string>
        </property>
      </widget>
    </item>
    <item row="1" column="1">
      <widget class="QLabel" name="SID_label">

```

```

    <property name="text">
      <string>TextLabel</string>
    </property>
  </widget>
</item>
<item row="14" column="1">
  <widget class="QLabel" name="TIAKCTIME_label">
    <property name="text">
      <string>TextLabel</string>
    </property>
  </widget>
</item>
<item row="0" column="1">
  <widget class="QLabel" name="user_label">
    <property name="text">
      <string>TextLabel</string>
    </property>
  </widget>
</item>
<item row="0" column="0">
  <widget class="QLabel" name="label_11">
    <property name="text">
      <string>用户姓名(user)</string>
    </property>
  </widget>
</item>
<item row="3" column="1">
  <widget class="QLabel" name="FID_label">
    <property name="text">
      <string>TextLabel</string>
    </property>
  </widget>
</item>
<item row="4" column="0">
  <widget class="QLabel" name="label_2">
    <property name="text">
      <string>航空公司(Flight)</string>
    </property>
  </widget>
</item>

```

```

<item row="4" column="1">
  <widget class="QLabel" name="Flight_label">
    <property name="text">
      <string>TextLabel</string>
    </property>
  </widget>
</item>
<item row="12" column="1">
  <widget class="QLabel" name="ARRDATE_label">
    <property name="text">
      <string>TextLabel</string>
    </property>
  </widget>
</item>
<item row="9" column="1">
  <widget class="QLabel" name="FTO_label">
    <property name="text">
      <string>TextLabel</string>
    </property>
  </widget>
</item>
<item row="11" column="1">
  <widget class="QLabel" name="FTIME_label">
    <property name="text">
      <string>TextLabel</string>
    </property>
  </widget>
</item>
<item row="11" column="0">
  <widget class="QLabel" name="label_8">
    <property name="text">
      <string>起飞时间(FTIME)</string>
    </property>
  </widget>
</item>
<item row="5" column="0">
  <widget class="QLabel" name="label_3">
    <property name="text">
      <string>飞机型号(FMODEL)</string>
    </property>
  </widget>
</item>

```

```

</widget>
</item>
<item row="10" column="0">
  <widget class="QLabel" name="label_7">
    <property name="text">
      <string>起飞日期(FDATE)</string>
    </property>
  </widget>
</item>
<item row="13" column="0">
  <widget class="QLabel" name="label_10">
    <property name="text">
      <string>到达时间(ARRTIME)</string>
    </property>
  </widget>
</item>
<item row="12" column="0">
  <widget class="QLabel" name="label_9">
    <property name="text">
      <string>到达日期(ARRDATE)</string>
    </property>
  </widget>
</item>
<item row="9" column="0">
  <widget class="QLabel" name="label_6">
    <property name="text">
      <string>飞往(FTO)</string>
    </property>
  </widget>
</item>
<item row="6" column="0">
  <widget class="QLabel" name="label_4">
    <property name="text">
      <string>飞机状态(FSTATE)</string>
    </property>
  </widget>
</item>
<item row="7" column="0">
  <widget class="QLabel" name="label_5">

```

```

    <property name="text">
        <string>起飞地(FFROM)</string>
    </property>
</widget>
</item>
<item row="1" column="0">
    <widget class="QLabel" name="label_15">
        <property name="text">
            <string>座位号码(SID)</string>
        </property>
    </widget>
</item>
<item row="3" column="0">
    <widget class="QLabel" name="label">
        <property name="text">
            <string>航班号(FID)</string>
        </property>
    </widget>
</item>
<item row="5" column="1">
    <widget class="QLabel" name="FMODEL_label">
        <property name="text">
            <string>TextLabel</string>
        </property>
    </widget>
</item>
<item row="10" column="1">
    <widget class="QLabel" name="FDATE_label">
        <property name="text">
            <string>TextLabel</string>
        </property>
    </widget>
</item>
<item row="2" column="0">
    <widget class="QLabel" name="label_16">
        <property name="text">
            <string>座位等级(SeatRank)</string>
        </property>
    </widget>

```

```

</item>
<item row="2" column="1">
  <widget class="QLabel" name="SeatRank_label">
    <property name="text">
      <string>TextLabel</string>
    </property>
  </widget>
</item>
</layout>
</widget>
<widget class="QPushButton" name="pushButton">
  <property name="geometry">
    <rect>
      <x>250</x>
      <y>520</y>
      <width>88</width>
      <height>27</height>
    </rect>
  </property>
  <property name="text">
    <string>机票打印</string>
  </property>
</widget>
<widget class="QLabel" name="label_12">
  <property name="geometry">
    <rect>
      <x>250</x>
      <y>50</y>
      <width>111</width>
      <height>41</height>
    </rect>
  </property>
  <property name="text">
    <string> 电子机票</string>
  </property>
</widget>
</widget>
<resources/>
<connections/>
</ui>

```

Report_dialog

report_dialog.h

```

/* FileName:report_dialog.h
 * Author:Hover
 * E-Mail:hover@hust.edu.cn
 * GitHub:HoverWings
 * Description:The declaration of report module to draw the report of DB data
 */

#ifndef REPORT_DIALOG_H
#define REPORT_DIALOG_H

#include <QDialog>
#include <QString>
#include <QSql>
#include <QSqlQuery>
#include <QComboBox>
#include <QDebug>
#include <QDate>
#include <QDateTime>
#include <QtWidgets>
#include <QtCharts>
#include <QWidget>
#include <QGraphicsView>

namespace Ui
{
    class report_Dialog;
}

class report_Dialog : public QDialog
{
    Q_OBJECT

public:
    explicit report_Dialog(QWidget *parent = 0);
    ~report_Dialog();

private slots:
    void on_FLIGHT_comboBox_currentTextChanged(const QString &arg1);

```



```

        void on_show_pushButton_clicked();

private:
    Ui::report_Dialog *ui;
    int flightNum;
    void setFlight_Combox();
    void fun();
};

#endif // REPORT_DIALOG_H

report_dialog.cpp

/* FileName:report_dialog.cpp
 * Author:Hover
 * E-Mail:hover@hust.edu.cn
 * GitHub:HoverWings
 * Description:The implementation of report module to draw the report of DB data
 */
#include "report_dialog.h"
#include "ui_report_dialog.h"

report_Dialog::report_Dialog(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::report_Dialog)
{
    ui->setupUi(this);
    setFlight_Combox();
}

report_Dialog::~report_Dialog()
{
    delete ui;
}

void report_Dialog::setFlight_Combox()
{
    // set flight combox

```

```

QComboBox* com=ui->FLIGHT_comboBox;
com->clear();
QString query;
QString opName="FLIGHTinfo";
QString showItem="FLIGHT";
query.prepare("select "+showItem+" FROM "+opName+" group by
"+showItem);
bool isOk = query.exec();
flightNum=0;
if(!isOk)
{
    return;
}
while (query.next())
{
    com->addItem(query.value(0).toString());
    flightNum+=1;
}
com->addItem("ALL");
com->setCurrentIndex(-1);
}

void report_Dialog::on_FLIGHT_comboBox_currentTextChanged(const QString
&arg1)
{
    //this->resize(440, 300);
}

void report_Dialog::fun()
{
    QSqlQuery query;
    QString opName="FLIGHTinfo";
    QString showItem="FLIGHT";
    query.prepare("select "+showItem+" FROM "+opName+" group by
"+showItem);
    bool isOk = query.exec();
    flightNum=0;
    if(!isOk)

```

```

    {
        return;
    }
    while (query.next())
    {
        flightNum+=1;
    }

    QComboBox * com=ui->FLIGHT_comboBox;
    QString text=com->currentText();
    if(text=="ALL") // show all report
    {
        qDebug()<<"ALL";
//        return;
    }
    QDate now= QDate::currentDate();
    QDate M1month=now.addMonths(-1);
    QDate M2month=now.addMonths(-2);
    QVector<QBarSet*> BarSetV;
//    QBarSet *set0 = new QBarSet("TOTAL");
//    BarSetV.append(set0);
//2
    for(int i=0;i<flightNum;i++)
    {

        QString Flight=com->itemText(i);
        QBarSet *set = new QBarSet(Flight);
        BarSetV.append(set);
        QSqlQuery query;
        query.prepare("select * from FLIGHTinfo where Flight= '"+Flight+"' and
date_format(FDATE, '%Y %m') = date_format DATE_SUB(curdate(), INTERVAL 2
MONTH), '%Y %m')");
        query.bindValue(":FLIGHT",Flight);
        bool isOK=query.exec();
        if(!isOK)
        {
            qDebug()<<Flight<<"错误! ";
        }
        int flightSell=0;

```

```

while (query.next())
{
    int FID=query.value(0).toInt();
    QSqlQuery query1;
    query1.prepare("select *,count(*) from FSTATUSinfo where
FID= :FID group by USABLE order by USABLE");
    query1.bindValue(":FID",FID);
    query1.exec();
    query1.next();
    int sellNum=query1.value(4).toInt();
    if(sellNum>150)
    {
        query1.next();
        sellNum=query1.value(4).toInt();
    }
    flightSell+=sellNum;
}
qDebug()<<Flight<<flightSell<<"上上个月";
set->append(flightSell);
}
//1
for(int i=0;i<flightNum;i++)
{

    QString Flight=com->itemText(i);
    QBarSet *set =BarSetV[i];
    BarSetV.append(set);
    QSqlQuery query;
    query.prepare("select * from FLIGHTinfo where Flight= '"+Flight+"' and
date_format(FDATE, '%Y %m') = date_format(DATE_SUB(curdate(), INTERVAL 1
MONTH),'%Y %m')");
    query.bindValue(":FLIGHT",Flight);
    bool isOK=query.exec();
    if(!isOK)
    {
        qDebug()<<Flight<<"错误! ";
    }
    int flightSell=0;
    while (query.next())
    {

```

```

        int FID=query.value(0).toInt();
        QSqlQuery query1;
        query1.prepare("select *,count(*) from FSTATUSinfo where
FID= :FID group by USABLE order by USABLE");
        query1.bindValue(":FID",FID);
        query1.exec();
        query1.next();
        int sellNum=query1.value(4).toInt();
        if(sellNum>150)
        {
            query1.next();
            sellNum=query1.value(4).toInt();
        }
        flightSell+=sellNum;
    }
    qDebug()<<Flight<<flightSell<<"上个月";
    set->append(flightSell);
}
//0
for(int i=0;i<flightNum;i++)
{

    QString Flight=com->itemText(i);
    QBarSet *set =BarSetV[i];
    BarSetV.append(set);
    QSqlQuery query;
    query.prepare("select * from FLIGHTinfo where Flight= '"+Flight+"' and
date_format(FDATE, '%Y %m') = date_format(DATE_SUB(curdate(), INTERVAL 0
MONTH),'%Y %m')");
    query.bindValue(":FLIGHT",Flight);
    bool isOK=query.exec();
    if(!isOK)
    {
        qDebug()<<Flight<<"错误! ";
    }
    int flightSell=0;
    while (query.next())
    {
        int FID=query.value(0).toInt();
        QSqlQuery query1;

```

```

        query1.prepare("select *,count(*) from FSTATUSinfo where
FID= :FID group by USABLE order by USABLE");
        query1.bindValue(":FID",FID);
        query1.exec();
        query1.next();
        int sellNum=query1.value(4).toInt();
        if(sellNum>150)
        {
            query1.next();
            sellNum=query1.value(4).toInt();
        }
        flightSell+=sellNum;
    }
    qDebug()<<Flight<<flightSell<<"这个月";
    set->append(flightSell);
}

QBarSeries *barseries = new QBarSeries();
for(int i=0;i<flightNum;i++)
{
    barseries->append(BarSetV[i]);
}

QChart *chart = new QChart();
chart->addSeries(barseries);
//chart->addSeries(lineseries);
chart->setTitle("Report");

QStringList categories;
categories << QString::number(M2month.month(),10) <<
QString::number(M1month.month(),10) << QString::number(now.month(),10);
//categories << "Jan" << "Feb" << "Mar" << "Apr" << "May" << "Jun";
QBarCategoryAxis *axisX = new QBarCategoryAxis();
axisX->append(categories);
//chart->setAxisX(axisX, lineseries);
chart->setAxisX(axisX, barseries);
//axisX->setRange(QString("Jan"), QString("Jun"));

QValueAxis *axisY = new QValueAxis();
//chart->setAxisY(axisY, lineseries);

```

```

chart->setAxisY(axisY, barseries);
axisY->setRange(0, 20);

chart->legend()->setVisible(true);
chart->legend()->setAlignment(Qt::AlignBottom);

QChartView *chartView = new QChartView(chart);
chartView->setRenderHint(QPainter::Antialiasing);
ui->graphicsView->setViewport(chartView);
}

void report_Dialog::on_show_pushButton_clicked()
{
    fun();
}

```

report_dialog.ui

```

<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
    <class>report_Dialog</class>
    <widget class="QDialog" name="report_Dialog">
        <property name="geometry">
            <rect>
                <x>0</x>
                <y>0</y>
                <width>672</width>
                <height>640</height>
            </rect>
        </property>
        <property name="windowTitle">
            <string>Dialog</string>
        </property>
        <widget class="QGraphicsView" name="graphicsView">
            <property name="geometry">
                <rect>
                    <x>90</x>
                    <y>60</y>
                    <width>441</width>
                    <height>381</height>
                </rect>
            </property>
        </widget>
    </widget>
</ui>

```

```

    </property>
</widget>
<widget class="QWidget" name="layoutWidget">
    <property name="geometry">
        <rect>
            <x>190</x>
            <y>530</y>
            <width>175</width>
            <height>29</height>
        </rect>
    </property>
    <layout class="QHBoxLayout" name="horizontalLayout">
        <item>
            <widget class="QComboBox" name="FLIGHT_comboBox"/>
        </item>
        <item>
            <widget class="QPushButton" name="show_pushButton">
                <property name="text">
                    <string>PushButton</string>
                </property>
            </widget>
        </item>
    </layout>
</widget>
</widget>
<resources/>
<connections/>
</ui>

```