

**数据库系统原理实践报告**

|  |  |
| --- | --- |
| 姓 名： | 余文梦 |
| 学 院： | 计算机学院 |
| 专 业： | 计算机科学与技术 |
| 班 级： | CS1408班 |
| 学 号： | U201414775 |
| 指导教师： | 瞿彬彬 |

|  |  |
| --- | --- |
| 分数 |  |
| 教师签名 |  |

2017年 6 月 21 日

**目录**

[1 课程任务概述 1](#_Toc1619854695)

[2 实验一 基本SQL操作 2](#_Toc1376415581)

[2.1 任务要求 2](#_Toc1299832107)

[2.2 完成过程 3](#_Toc822881895)

[2.2.1 数据定义 3](#_Toc653751067)

[2.2.2数据更新 4](#_Toc112763809)

[2.2.3查询功能 8](#_Toc697326546)

[2.2.3选做部分 13](#_Toc501229540)

[2.3任务总结 17](#_Toc1568262488)

[3 DBMS综合运用 18](#_Toc24267247)

[3.1 任务要求 18](#_Toc959740824)

[3.2 完成过程 18](#_Toc1671546949)

[3.2.1 备份操作 18](#_Toc420517109)

[3.2.2 权限配置 18](#_Toc211347417)

[3.2.3 查询优化 19](#_Toc973226802)

[3.2.4 数据导入导出 20](#_Toc1925649857)

[3.2.5 储存过程与函数 20](#_Toc665692938)

[3.2.6 开发环境集成 20](#_Toc2107011803)

[3.3任务总结 21](#_Toc1874155896)

[4 综合实践任务 22](#_Toc506985117)

[4.1系统设计目标 22](#_Toc972425843)

[4.2 需求分析 22](#_Toc1042709713)

[4.2.1 功能需求 22](#_Toc260746464)

[4.2.2 性能需求 22](#_Toc1527824227)

[4.2.3 数据说明 23](#_Toc203276281)

[4.3 总体设计 26](#_Toc767516877)

[4.3.1 C/S架构图 26](#_Toc1954830676)

[4.3.2 系统模块划分方案 26](#_Toc460083854)

[4.3.3 业务流程图 26](#_Toc1169087733)

[4.4 数据库设计 27](#_Toc1344625244)

[4.4.1 概念模型设计 27](#_Toc1040975946)

[4.4.2 关系表设计 29](#_Toc641458780)

[4.4.3 数据库的代码实现 30](#_Toc573557178)

[4.5 详细设计与实现 32](#_Toc193324405)

[4.5.1 开发环境 32](#_Toc1464340675)

[4.5.2 功能设计与实现 32](#_Toc1227308245)

[4.5.3 触发器实现 41](#_Toc306088214)

[4.6 系统测试 42](#_Toc14183574)

[4.6.1 测试数据说明 42](#_Toc1728537786)

[4.6.2 登录测试 43](#_Toc1874350703)

[4.6.3 租车功能测试 44](#_Toc38450821)

[4.6.4 基本信息测试 48](#_Toc540794962)

[4.6.5 订单处理测试 48](#_Toc1398414004)

[4.6.6 车辆管理测试 50](#_Toc458967930)

[4.6.7 员工管理测试 52](#_Toc752142379)

[4.6.8 财务报表测试 53](#_Toc224157159)

[4.7 系统设计与实现总结 54](#_Toc237134139)

[5 课程总结 55](#_Toc1417835317)

[5.1 主要工作 55](#_Toc183685314)

[5.2 心得体会 55](#_Toc2111290035)

[附录 56](#_Toc1924820435)

# 1 课程任务概述

本次课程任务主要由两部分构成——课程实验和课程设计。课程实验部分主要完成基本SQL操作以及DBMS系统的综合应用。课程设计部分需要结合自己的兴趣进行选题，实现一个数据库的应用系统。具体包括需求分析、总体设计、数据库设计、详细设计与实现以及数据库应用系统的测试等步骤。

# 

# 2 实验一 基本SQL操作

## 2.1 任务要求

1）数据定义

参照下面的内容建立自己实验所需的关系数据

创建三个关系:

商品表【商品名称、商品类型】

GOODS【GNAME char(20),GTYPE char(10)】

主关键字为(商品名称)。商品类型为(电器、文具、服装等)

商场【商场名称,所在地区】

PLAZA【PNAME char(20)，PAREAchar(20)】

主关键字为商场名称。所在地区为(洪山、汉口、汉阳、武昌等)

销售价格表【商品名称、商场名称、当前销售价格、目前举办活动类型】

SALE【GNAME char(20),PNAME char(20),PRICE FLOAT,ATYPEchar(10)】

主关键字为(商品名称、商场名称)。举办活动类型为(送券、打折),也可为空值,表示当前未举办任何活动。表中记录如(‘哈森皮靴’,‘亚贸广场’,200,‘打折’),同一商场针对不同的商品可能采取不同的促销活动。

2）数据更新

（1）向上述表格中用 sql 语句完成增、删、个、改的操作;

（2）编写一个触发器,并测试该触发器;

（3）将 SALE 表中的打折记录插入到新表 SALE\_CHEAP 中,并基于 SALE\_CHEAP 表创建一个统计每个商场各自打折商品平均价格的视图。

3）用 SQL 语句完成下述查询需求:

（1）查询所有以“打折”方式销售的商品的当前销售情况,并按照价格的降序排列;

（2）查询所有没有任何活动的商品及其所在的商场,结果按照商品排序;

（3）查询价格在 200~500 元之间的商品名称、所在的商场名称、价格,结果按照商场名称排序;

（4）查询每种商品的最低价格、商品名称;

（5）查询以“送券”方式销售的商品总数超过 2 种的商场名称;

（6）查询以“送券”方式销售的商品总数超过 2 种的商场所在地区;

（7）查询价格为下列取值之一的商品名称、所在商场名称、目前举办活动的类型(88、188、288);

（8）查询以“老”字开头的所有商品的名称;

（9）查询同时销售“剃须刀”和“电池”的商场名称;

（10）查询不举办任何活动的商场;

（11）查询所销售的商品包含了“校园超市”所销售的所有商品的商场名称。

4）SQL 思考题部分(选做)

（1）SQLSERVER 的时间数据类型 datatime,可以记录形如“2010-5-1 13:30:15”的时刻,表示2010 年 5 月 1 日 13 点 30 分 15 秒时刻,通过查询 SQLSERVER 的帮助文件,解决下述 SQL语句的需求:

（1.1）在查询结果中将 datetime 属性的值转换成字符串类型;

（1.2）查询当前时刻之前 2 年、或者 2 个月、或者 2 天、或者 2 小时、或者 2 分钟的记录;

（1.3）查询某个时刻所在的年、月、日、时、分、秒;

（1.4）查询某个月的最后一天,返回 datetime 类型;

（1.5）假设在 SALE 表中有交易时间(DEELTM)属性,类型为 datetime,查出所有在整点时刻的交易,查出所有在整 15 分钟时刻(每个小时有 4 个整 15 分钟,例如 8:00、8:15、8:30、8:45)的交易。

（2）如何将查询结果的某些列的某些值转换成特殊的形式(例如成绩记录存储的都是百分制的成绩,对查询结果中 60 分以上的记录直接显示成绩,而对 60 分以下的记录则在成绩属性中显示“不及格”)

## 2.2 完成过程

### 2.2.1 数据定义

1. 创建一个MyGoods的数据库，然后使用create table命令创建三张表（商品表、商场表和销售价格表）。此部分代码如下。

/\*建表\*/

Create table GOODS(

Gname char(20) primary key,/\*商品名称\*/

Gtype char(10)/\*商品类别\*/

)COMMENT='商品表';

create table PLAZA(

Pname char(20) primary key,/\*商场名称\*/

Parea char(20)/\*商场所在区域\*/

)COMMENT='商场表';

create table SALE(

Gname char(20),

Pname char(20),

Price float,/\*价格\*/

Atype char(10),

primary key (Gname,Pname),/\*设定主码\*/

foreign key (Gname) references GOODS(Gname),

foreign key (Pname) references PLAZA(Pname)

)COMMENT='销售价格表';

### 2.2.2数据更新

1. 实验内容2.1部分。

功能要求：向上述表格中用 sql 语句完成增、删、个、改的操作。

解决需求：使用“insert into 表名 values(...);”语句完成表中数据的增加。导入完成后，GOODS表中的数据如图2. 1所示；PLAZA表中数据如图2. 2所示；SALE表中的数据如图2. 3所示。接下来，使用这些数据完成指定的需求。



图2. 1 GOODS表

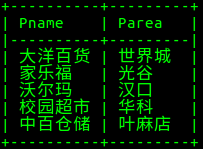


图2. 2 PLAZA表

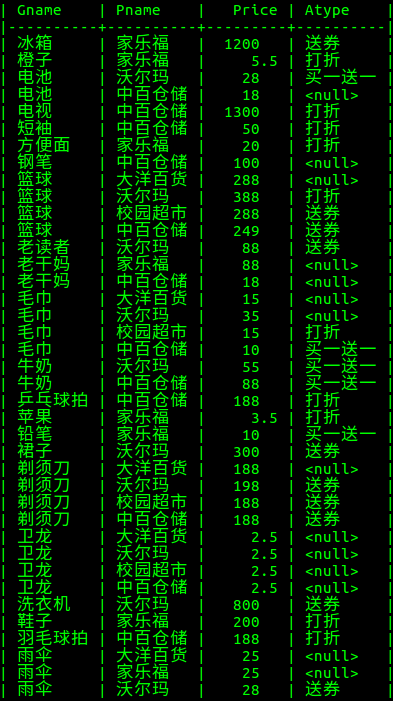


图2. 3 SALE表

使用delete语句完成表中数据的删除操作。特别地，为了不影响基本数据，先用“insert into SALE values('电视','家乐福',1200,'打折');”在库中插入了一条测试数据，紧接着用select语句查询此数据，查询结果如图2. 4所示。然后再用delete语句将此数据删除。删除数据后用select的查询结果如此部分代码如下所示，运行结果截图如图2. 5所示。

/\*删\*/

delete from SALE

where Pname = '家乐福' AND Gname = '电视';

select \* from SALE;

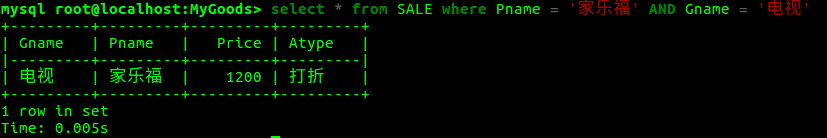


图2. 4 测试数据截图

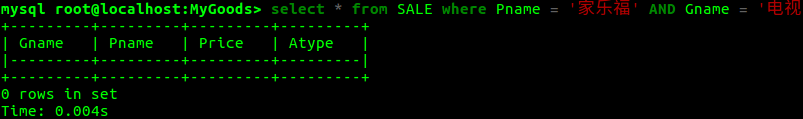


图2. 5 delete结果测试

接着，通过update语句完成数据的更改功能。功能代码如下所示。更改前的运行截图如图2. 6所示。更改后的运行截图如图2. 7所示。

/\*改\*/

update SALE

set Price = 300

where Pname = '沃尔玛' AND Gname = '裙子';

select \* from SALE;

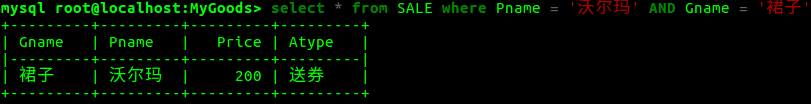


图2. 6 update前

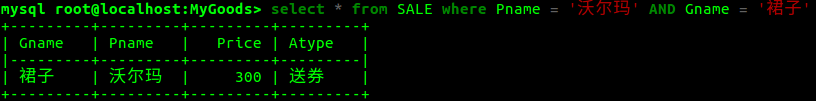


图2. 7 update后

1. 实验内容2.2部分。

功能要求：编写一个触发器,并测试该触发器。

解决需求：用create trigger创建一个触发器，当插入SALE表中商品的价格为负数时，将价格改为0。功能代码如下所示，程序运行结果截图如图2. 8所示。

create trigger fortest before insert ON SALE

for EACH row //以行为单位进行操作

begin

if new.Price < 0 then

set new.Price = 0;

end if;

end;

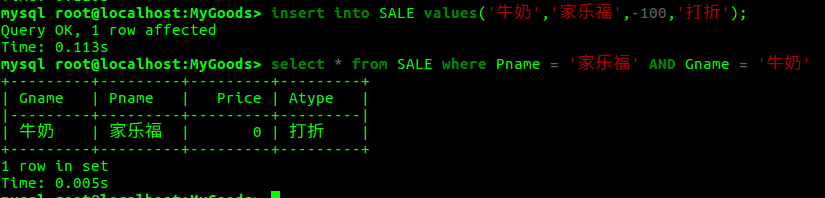


图2. 8 触发器运行结果

1. 实验内容2.3部分。

功能要求：将 SALE 表中的打折记录插入到新表 SALE\_CHEAP 中,并基于 SALE\_CHEAP 表创建一个统计每个商场各自打折商品平均价格的视图。

解决需求：首先，用create table语句创建一个SALE\_CHEAP表，该表的数据结构与表SALE一致。然后使用select语句将SALE表中打折的商品插入SALE\_CHEAP表中。实验代码如下所示。

/\*往打折表中插入数据\*/

INSERT INTO SALE\_CHEAP(GNAME,PNAME,PRICE,ATYPE)

SELECT GNAME,PNAME,PRICE,ATYPE

FROM SALE

WHERE ATYPE='打折';

执行结果生成的SALE\_CHEAP表如图2. 9所示。



图2. 9 SALE\_CHEAP表

接着，基于SALE\_CHEAP表，使用create view语句创建一个平均价格的视图，此部分实验代码如下，生成的视图结果见图2. 11。

create view AVE\_PRICE /\*创建视图AVE\_PRICE\*/

AS SELECT Pname,AVG(Price) /\*计算平均价格\*/

from SALE\_CHEAP

group by Pname /\*按商场名字分组\*/



图2. 10 视图AVE\_PRICE

### 2.2.3查询功能

（1）实验内容2.3部分。

功能需求：将 SALE 表中的打折记录插入到新表 SALE\_CHEAP 中,并基于 SALE\_CHEAP表创建一个统计每个商场各自打折商品平均价格的视图。

1. 实验内容3.1部分。

功能需求：查询所有以“打折”方式销售的商品的当前销售情况,并按照价格的降序排列。

解决需求：先从SALE表中查找出Atype为打折的商品，然后利用“order by”子句对查询结果进行降价排序。实验代码如下所示，执行结果见图2. 11。

select Gname,Price

from SALE

where Atype = '打折'

order by Price desc;/\*按价格降价排序\*/



图2. 11 查询结果1

1. 实验内容3.2部分。

功能需求：查询所有没有任何活动的商品及所在的商场,结果按照商品排序;

解决需求：此查询与上一个基本一致，新增了对IS NULL子句的使用。实验代码如下所示，执行结果见图2. 12。

select Gname,Pname

from SALE

where Atype IS NULL /\*判断Atype是否为空\*/

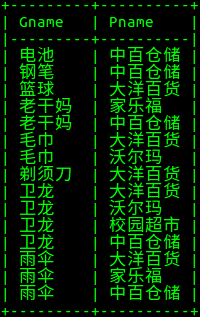


图2. 12 查询结果2

1. 实验内容3.3部分。

功能需求：查询价格在 200~500 元之间的商品名称、所在的商场名称、价格,结果按照商场名称排序;

解决需求：此查询新增了对比较运算符和逻辑与的使用。实验代码如下所示，执行结果见图2. 13。

select Gname,Pname,Price

from SALE where Price >= 200 and Price <= 500;



图2. 13 查询结果3

1. 实验内容3.4部分。

功能需求：查询每种商品的最低价格、商品名称;

解决需求：由于要求每种商品的最低价格，第一种思路是采用嵌套查询，第二种思路是采用group by子句先对表进行分组，然后调用MIN函数对每组进行求最小值。这里采用后一种思路，实验代码如下所示，执行结果见图2. 14。

select Gname,MIN(Price) /\*调用MIN子函数求最低价格\*/

from SALE

group by Gname;/\*按照名字分组\*/



图2. 14 查询结果4

1. 实验内容3.5部分。

功能需求：查询以“送券”方式销售的商品总数超过 2 种的商场名称;

解决需求：采用嵌套查询，先从PLAZA表中选出一个条目，然后在SALE表中找到对应商场的商品并且调用count子函数对活动类型为“送券”方式的商品进行计数，将计数结果大于2的输出。实验代码如下所示，执行结果见图2. 15。

select Pname

from PLAZA

where 2 < (

select count(\*) /\*调用count子函数进行计数\*/

from SALE

where SALE.Pname = PLAZA.Pname and Atype = '送券');



图2. 15 查询结果5

1. 实验内容3.6部分。

功能需求：查询以“送券”方式销售的商品总数超过 2 种的商场所在地区;

解决需求：基于上一个查询的实现，只需要将select所挑出的值改为Parea即可。实验代码基本一致，执行结果见图2. 16。



图2. 16 查询结果6



图2. 17 查询结果7

1. 实验内容3.7部分。

功能需求：查询价格为下列取值之一的商品名称、所在商场名称、目前举办活动的类型(88、188、288);

解决需求：此查询新增了对逻辑或（or）的使用。实验代码如下所示，执行结果见图2. 17。

select Gname,Pname,Atype

from SALE

where Price = '88' or Price = '188' or Price = '288';

1. 实验内容3.8部分。

功能需求：查询以“老”字开头的所有商品的名称;

解决需求：此查询新增了对正则规则的匹配，SQL语言中只支持任意字符“%”和任意单字符“\_”。实验代码如下所示，执行结果见图2. 18。

select Gname

from GOODS

where Gname like '老%';



图2. 18 查询结果8

1. 实验内容3.9部分。

功能需求：查询同时销售“剃须刀”和“电池”的商场名称;

解决需求：一开始，想到的是直接用INTERSECT子句对两个集合求交集找出同时满足两个条件的商品，却发现新版的MySql中不支持该子句，于是使用嵌套查询。实验代码如下所示，执行结果见图2. 19。

select Pname

from SALE

where Gname = '剃须刀' and

Pname IN(/\*在子查询中选出商品名称为电池\*/

select Pname

from SALE

where Gname = '电池');



图2. 19 查询结果9

1. 实验内容3.10部分。

功能需求：查询不举办任何活动的商场。

解决需求：先从SALE表中选出Pname和Atype两个属性列（用DISTINCT去重），并以此建立一个子表。接着利用嵌套查询找出有且仅有空的活动类型的数据项。实验代码如下所示，执行结果见图2. 20。

select Pname

from(/\*创建一个子表\*/

select DISTINCT Pname,Atype

from SALE

)as S1

where Atype NOT IN(

select Atype

from SALE

where SALE.Pname = S1.Pname and

SALE.Atype IS NOT NULL);

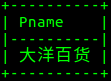


图2. 20 查询结果10

1. 实验内容3.11部分。

功能需求：查询所销售的商品包含了“校园超市”所销售的所有商品的商场。

解决需求：这是一个全称命题，而SQL语句中没有相应的语句，但是可以用存在量词对其进行表示。即改为“对于某商场，在校园超市里面，不存在某商品，该商场没有”。实验代码如下所示，执行结果见图2. 21。

select DISTINCT Pname

from PLAZA P1

where NOT EXISTS(

select \*

from SALE S1

where S1.Pname = '校园超市' and

NOT EXISTS(

select \*

from SALE S2

where P1.Pname = S2.Pname and

S2.Gname = S1.Gname)

);



图2. 21 查询结果11

经验证，上述查询结果全部与实验预期结果相符。

### 2.2.3选做部分

为了完成此部分，首先，在数据库中新建了一个SALE\_T表，其中在SALE表的基础上增加了时间属性。建表语句如下所示，插入一些基本数据后，得到的效果图如图2. 22所示。

create table SALE\_T(

Gname char(20),

Pname char(20),

Price float,

Atype char(10),

DEELTM DATETIME,

primary key (Gname,Pname)

)

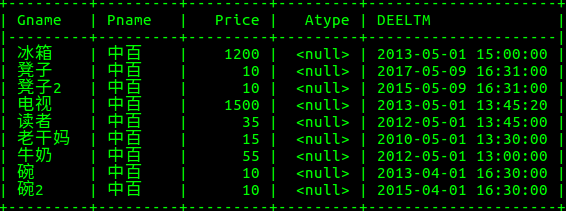


图2. 22 SALE\_T表内容

（1）实验内容3.1.1部分。

功能需求：在查询结果中将 datetime 属性的值转换成字符串类型。

解决需求：调用内置函数date\_format函数，将时间DEELTM属性转换为“年-月-日 时-分-秒”的格式输出。功能代码如下所示，运行结果如图2. 23所示。

/\*将当前时刻转换成字符串输出\*/

select date\_format(DEELTM,"%Y-%m-%d %H:%i:%s")

from SALE\_T;

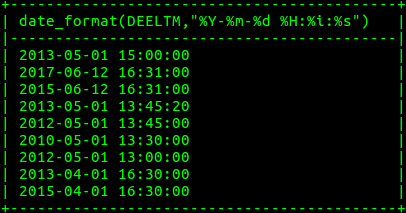


图2. 23 时间转换结果

（2）实验内容3.1.2部分。

功能需求：查询当前时刻之前 2 年、或者 2 个月、或者 2 天、或者 2 小时、或者 2 分钟的记录。

解决需求：以查询两天前记录为例，利用内部函数to\_days进行天数的转换，得到两天前的数据记录。功能代码如下所示，运行结果如图2. 24所示。

select \*

from SALE\_T

where to\_days(now()) - to\_days(DEELTM) = 2;

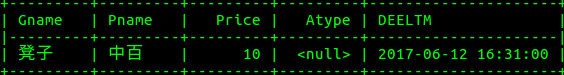


图2. 24 两天前记录查询结果

（3）实验内容3.1.3部分。

功能需求：查询某个时刻所在的年、月、日、时、分、秒;

解决需求：通过调用mysql内置的year()，month()，day()，hour()，minute()，second()等函数，完成时刻的查询。功能代码如下所示，运行结果如图2. 25所示。

/\*查询某个时刻的时分秒\*/

select year('2015-5-09 16:31:00');

select month('2015-5-09 16:31:00');

select day('2015-5-09 16:31:00');

select hour('2015-5-09 16:31:00');

select minute('2015-5-09 16:31:00');

select second('2015-5-09 16:31:00');

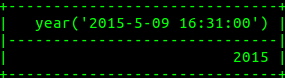


图2. 25 年的查询结果

（5）实验内容3.1.4部分。

功能需求：查询某个月的最后一天,返回 datetime 类型;

解决需求：通过调用mysql内置的last\_day()查询每个月的最后一天。功能代码如下所示，运行结果如图2. 26所示。

/\*查询某个月的最后一天\*/

select last\_day('2015-5-09 16:31:00');

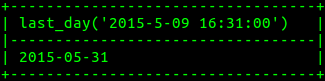


图2. 26 某月的最后一天

（6）实验内容3.1.5部分。

功能需求：假设在 SALE 表中有交易时间(DEELTM)属性,类型为 datetime,查出所有在整点时刻的交易,查出所有在整 15 分钟时刻(每个小时有 4 个整 15 分钟,例如 8:00、8:15、8:30、8:45)的交易。

解决需求：利用date\_format函数将datetime类型中的秒和分选出，然后利用where语句选出秒为0以及分为0、15、30和45的交易信息。功能代码如下所示，查询结果如图2. 27所示。

/\*查询所有在整15分钟时刻的交易\*/

select \*

from SALE\_T

where date\_format(DEELTM,"%s") = '00' and

(date\_format(DEELTM,"%i") = '00' or

date\_format(DEELTM,"%i") = '15' or

date\_format(DEELTM,"%i") = '30' or

date\_format(DEELTM,"%i") = '45');



图2. 27 查询整点时刻的交易信息

（7）实验内容3.2部分。

功能需求：如何将查询结果的某些列的某些值转换成特殊的形式。

解决需求：利用case语句对价格信息进行判断，如果价格大于1000,则显示结果输出“太高了”。功能代码如下所示，查询结果如图2. 28所示。

/\*修改查询结果中某些列为特殊的值,对于价格超过1000的显示太高了\*/

select Gname,Pname,(case when Price>1000 then '太高了' else Price end) as Price,Atype,DEELTM

from SALE\_T

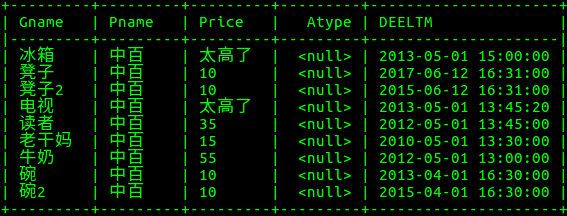


图2. 28 输出格式转换结果

## 2.3任务总结

在做本次实验过程中，碰到了不少的问题，可能是第一次接触数据库实验的缘故。

首先，因为自己的操作系统是linux，所以选择了mysql数据库环境，而mysql语言虽然大体上符合sql标准，但是有些关键字的支持还是不一样的，比如，mysql没有对集合的交并等操作。更重要的是，mysql自身的特性随着版本的不一致也有较大差异，这也造成自己在寻求帮助时，很多时候只能去查找mysql官方帮助文档，从那里得到的解决办法还是很有价值的。

另外，此次实验也考验了自己对数据库的基本操作能力，对其增删改查的特性也有了初步的体会。尤其是最后做时间相关的部分时，更是体会到它的便利性。这比自己原来直接用文件进行数据的存放简易多了。

最后，自己也深刻感受到目前掌握的只是其冰山一角，还有很多特性等待自己去了解和掌握。

# 

# 3 DBMS综合运用

## 3.1 任务要求

1. 学习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（选做）。
2. 学习系统的身份、权限配置操作
3. 了解系统的查询性能分析功能（选做）

通过对第一部分任务中的SQL语句的查询性能分析，学习SQL查询的关系代数处理及其优化原理。

（4）了解SQLSERVER的数据导入导出功能（选做）

操作SQLSERVER的企业管理器集成界面，通过交互式方式实现数据的导入和导出。

（5）了解SQLSERVER的存储过程、触发器、函数实现过程

通过查看SQLSERVER的联机帮助文档和示例，体会存储过程、函数和触发器的原理，尝试编制具备基本功能的存储过程、触发器和函数的实际例子，查看其执行效果。

（6）了解SQLSERVER与常用应用程序开发环境集成的实现过程（选作）。

通过开发简单的DBMS应用程序，了解基本的数据库应用编程方法与技术，包括：数据库连接、嵌入式的数据读取操作、数据更新操作、事务提交技术等等。

## 3.2 完成过程

### 3.2.1 备份操作

Mysql提供了一个很好用的备份工具——mysqldump，用起来也特别方便，比如：“mysqldump -u root -p 数据库名> a.sql”就可以把想要备份的数据库给备份成一个sql语言文件a.sql。本质上就是将数据库中的表格转换成“create table”和“insert value”等语句，通俗易懂。而进行数据库恢复时，只需要在mysql环境下执行该sql文件即可，其实就是把该文件当成自己写的文件一样处理。

### 3.2.2 权限配置

经过查询官方文档，了解到mysql数据库中有4个权限表：user、db、tables\_priv和columns\_priv，权限表的存取过程如下。

1. 先从user表中的host、user、password三个字段中判断连接的IP、用户名、密码是否存在表中，存在则通过身份验证。
2. 通过权限验证，进行权限分配时，按user>db>tables\_priv>columns\_priv的顺序进行分配。即先检查全局权限表的user，如果user中对应的权限为Y，则此用户对所有的数据库的权限都是Y，将不再检查其余表;否则，检查db表，依此类推，在mysql终端中进行权限管理的命令如下所示。

grant 权限 on databaseName.tableName to userName@localhost identified by “password”

Mysql共有27种权限，下面列举几种自己实践过的。

|  |  |
| --- | --- |
| Usage | 连接登录权限，默认授予 |
| Select | 查询权限 |
| Create | Create table权限 |
| Create view | Create view权限 |
| insert | Insert权限 |
| Alter | 修改表格权限 |
| Update | 更新表格权限 |
| delete | 删除表格权限 |
| drop | 删除表或数据库的权限 |
| Index | 创建索引的权限 |

下面，创建一个TEST用户，只赋予其查询SALE表的权限，之后利用select语句查询mysql库下的user表，得到的结果如图3. 1所示。最后用此用户登录mysql数据库，发现只能查询select表，如图3. 2所示。

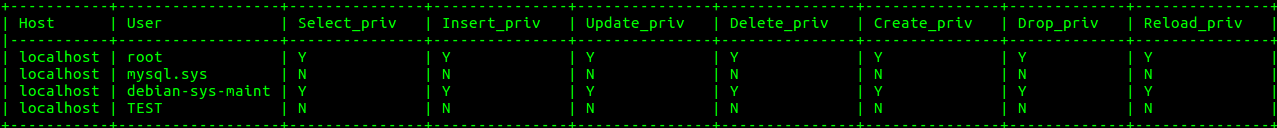


图3. 1 权限结果表



图3. 2 TEST用户登录测验

### 3.2.3 查询优化

结合老师理论课上所讲内容，可以得知数据库的查询优化动作主要为以下几步。

1. 将用户输入的查询语句构造成一颗查询优化树。
2. 利用查询优化树构造出关系代数语法树。
3. 最后，利用一些等价变换规则，对语法树进行等价变换，从而得到性能更优的查询树。

### 3.2.4 数据导入导出

（1）数据导出

Select ... Into outfile ‘文件名’

直接执行上面的语句会报错，因为需要指定输出文件的安全路径，可以先用语句‘show variables like “secure\_file\_priv”’找出安全路径，然后，在文件名中添加路径即可。当然，也可以通过修改配置文件更改路径。

（2）数据导入

Load data infile ‘文件名’into ...

也可以仿照前面的备份使用mysqldump命令进行数据的导入与导出。

### 3.2.5 储存过程与函数

经过查阅相关资料，了解到以下几点。

（1）存储过程

存储过程是预编译的，在首次运行一个存储过程时查询，优化器对其进行分析优化，并且给出最终被存储在系统表中的执行计划。而批处理的SQL语句在每次运行时都要进行编译和优化。所以存储过程的速度要快一些。

语法：create procedure 过程名（过程参数...）[特性...]过程体

（2）函数

存储函数需要返回一个类型，且函数体中必须要有一个有效的return语句，调用过程与调用内部函数MAX等类似。

语法：create function 函数名（参数...）

（3）触发器

触发器是一种特殊的存储过程，在特定表执行更新，插入，删除操作时触发，具有更好的数据控制能力。

### 3.2.6 开发环境集成

这里主要利用C++语言去连接并且操作数据库，编写的C++代码如下所示。

#include"mysql/mysql.h" /\*包含mysql头文件\*/

#include<iostream>

int main()

{

MYSQL DbObj;

mysql\_init(&DbObj);

/\*链接数据库\*/

if(!mysql\_real\_connect(&DbObj,"localhost","root","960606saandsb!","MyGoods",3360,NULL,0))

return 1;

std::cout << "OK" << endl;

/\*执行数据库查询语句\*/

if(!mysql\_query(&DbObj,"select \* from SALE\_T"))

std::cout << "query OK!" << endl;

mysql\_close(&DbObj);/\*关闭数据库\*/

return 0;

}

刚开始，使用“g++ test.cpp -o test”进行编译，一直显示变量未定义，如图3. 3所示。原因是未包含外部依赖库mysqlclient，于是使用“g++ -o test test.cpp -l mysqlclient”，得到正确的编译结果，运行编译后生成的可执行文件，运行结果见图3. 4。

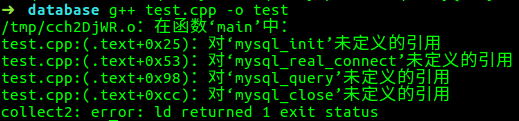


图3. 3 编译错误

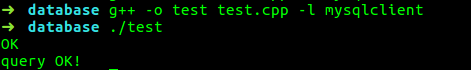


图3. 4 编译成功

## 3.3任务总结

通过本次实验，主要了解和掌握了mysql数据库的备份机制、权限管理和配置、数据导入导出、存储过程与函数和开发环境集成搭建等功能。其中，存储过程与函数部分碰到了不少的麻烦，因为很多看似可以的语句，实际运行起来却频频报错。这也使得自己觉得mysql的函数功能有点难以操作，不像C语言等简单明了。最后一个开发环境搭建那里，在查找依赖包的过程中，也碰到了不小的麻烦。特别是由于版本的问题，旧版本的依赖包变得不再适用，于是自己不得不在网上查找，这个步骤也耗费了不少的时间。后面想一想，每次实验又何尝不是这样的过程呢？

# 

# 4 综合实践任务

## 4.1系统设计目标

目前，虽然家庭拥有汽车量越来越多，但是当外出旅游或者办公时，租用汽车却是一个更好的选择。为了方便汽车租赁公司的数据管理，打算设计一个功能完善的汽车租借信息系统。完成用户、车辆、经手员工、租借情况、车辆损毁情况、交通违规罚款等信息的管理。

## 4.2 需求分析

### 4.2.1 功能需求

按照题目的要求，要实现的基本功能有以下几点：

1. 实现不同权限的浏览和更新。
2. 能够根据车辆使用情况计算押金退还金额。
3. 能查询客户的租借历史记录，并进行信誉度评价，进行会员制和非会员制的客户管理。
4. 能够管理车辆报修信息；
5. 能够生成租借公司的日、月、季度、年财务报表。

### 4.2.2 性能需求

为了达到上述要求，可以考虑此系统能有三种角色操作，即客户、普通员工和高级员工。他们拥有不同的权限，比如：客户可以查询自己的账户，租车记录等信息；普通员工可以进行用户信息和车辆信息中条目的增删改查等功能；高级员工可以进行所有信息的增删改查。在管理层面上，能够提供车辆信息界面，客户信息界面，租借服务界面以及员工信息界面等。

对于用户信息，能够查询用户编号、用户名、用户性别、用户年龄以及租车记录等信息。并且能够增对用户的历史记录进行评级。

对于员工信息，能够提供员工编号、员工姓名、性别、员工年龄以及员工处理的交易数和具体的交易编号等信息。并且能够对员工的工作历史进行等级评定。

对于车辆信息，能够查询到车辆的车辆编号、车型、颜色、损毁程度、目前状态（空闲或者在使用）和以往租借记录等信息。

对于租借信息，可以查询所有交易信息，包括交易编号、交易经手员工、用户编号、起止时间等信息。

另外，有一个对外的公共查询窗口，可以方便大众查看汽车的基本信息，而不用登录系统，使得该系统更加合理化、人性化。

### 4.2.3 数据说明

1. 数据项名：用户名

数据项含义说明：用字母和数字作为用户的唯一标识。

数据类型：CHAR

长度：20

1. 数据项名：用户密码

数据项含义说明：用字母和数字构成的用户登录密码。

数据类型：CHAR

长度：12

1. 数据项名：性别

数据项含义说明：用单字符（M/W）标识用户或者员工的性别。

数据类型：CHAR

长度：1

1. 数据项名：年龄

数据项含义说明：用户或者员工的年龄。

数据类型：int

1. 数据项名：用户信用评级

数据项含义说明：用单个字符(A/B/C/D)对用户进行评级。

数据类型：CHAR

长度：1

1. 数据项名：账户余额

数据项含义说明：用户的账户余额，当用户的账户余额大于10000时，自动给用户带上“黄金会员”的标识。

数据类型：float

1. 数据项名：员工编号

数据项含义说明：用字母和数字作为员工的唯一标识。

数据类型：CHAR

长度：20

1. 数据项名：员工密码

数据项含义说明：用字母和数字构成的员工登录密码。

数据类型：CHAR

长度：12

1. 数据项名：员工姓名

数据项含义说明：员工的姓名。

数据类型：CHAR

长度：20

1. 数据项名：交易数

数据项含义说明：员工入职以来的经手的交易总量。

数据类型：int

1. 数据项名：入职时间

数据项含义说明：员工的入职时间（格式：xxxx-xx-xx xx:xx:xx)。

数据类型：datetime

1. 数据项名：基本工资

数据项含义说明：员工的月基本工资。

数据类型：float

1. 数据项名：身份标识

数据项含义说明：员工的身份标识。

数据类型：CHAR

长度：1

1. 数据项名：车辆编号

数据项含义说明：代表车辆的唯一编号。

数据类型：CHAR

长度：12

1. 数据项名：车辆品牌

数据项含义说明：车辆的品牌标识。

数据类型：CHAR

长度：12

1. 数据项名：车辆颜色

数据项含义说明：车辆的颜色标识。

数据类型：CHAR

长度：12

1. 数据项名：车辆状态

数据项含义说明：用单字符(A/B/C)标识车辆的状态信息。

数据类型：CHAR

长度：20

1. 数据项名：购入费用

数据项含义说明：购入车辆时的开销。

数据类型：float

1. 数据项名：购入时间

数据项含义说明：车辆的买入时间（格式：xxxx-xx-xx xx:xx:xx)。

数据类型：datetime

1. 数据项名：押金

数据项含义说明：租用该车时，用户需要抵付的押金。

数据类型：float

1. 数据项名：租金

数据项含义说明：租用该车时，每小时的开销。

数据类型：float

1. 数据项名：vip

数据项含义说明：用单子符(Y/N)标识该车是否是vip用户专享车辆。

数据类型：CHAR

数据长度：1

1. 数据项名：交易编号

数据项含义说明：每笔租车交易的唯一标识。

数据类型：CHAR

长度：12

1. 数据项名：开始时间

数据项含义说明：交易的起始时间（格式：xxxx-xx-xx xx:xx:xx)。

数据类型：datetime

1. 数据项名：结束时间

数据项含义说明：交易的结束时间（格式：xxxx-xx-xx xx:xx:xx)。

数据类型：datetime

1. 数据项名：交易费用

数据项含义说明：交易的最终费用。

数据类型：float

1. 数据项名：罚款

数据项含义说明：结束交易时，所产生的罚款数。

数据类型：float

1. 数据项名：状态

数据项含义说明：交易结束时，员工对车辆的判定结果。

数据类型：CHAR

长度：1

1. 数据项名：奖金

数据项含义说明：员工的奖金。

数据类型：float

1. 数据项名：发奖时间

数据项含义说明：奖金的颁布时间（格式：xxxx-xx-xx xx:xx:xx)。

数据类型：datetime

## 4.3 总体设计

### 4.3.1 C/S架构图

该租车系统采用C/S架构模式，见。

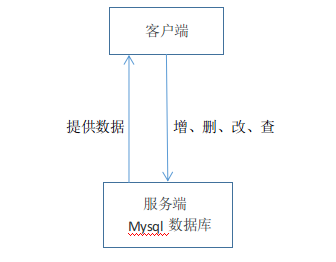


图4. 1 租车系统C/S架构图

### 4.3.2 系统模块划分方案

根据4.2小节的分析，可以得到租车系统功能模块图，见

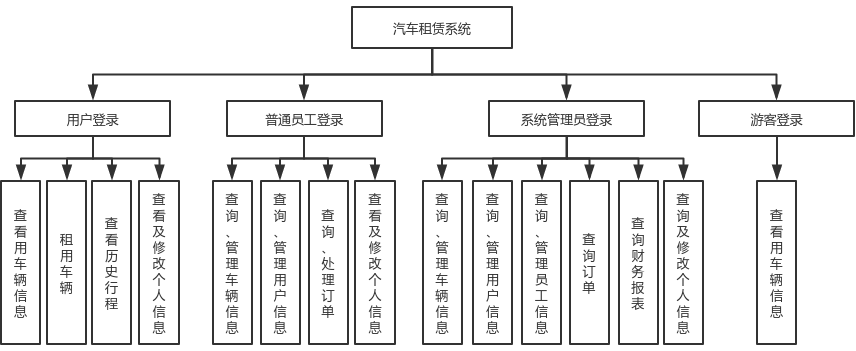


图4. 2 系统模块图

### 4.3.3 业务流程图

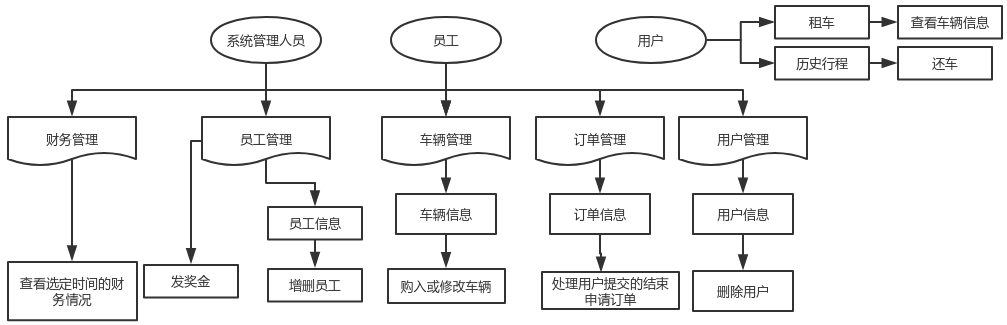


图4. 3 业务流程图

## 4.4 数据库设计

### 4.4.1 概念模型设计

首先设计各个分E-R图，其中，用户信息的E-R图如图4. 4所示；员工信息E-R图如图4. 5所示；车辆信息的E-R图如图4. 6所示；租借信息的E-R如图4. 7所示，奖金信息表如图4. 8所示。特别地，对于主码用了下划线进行标识。

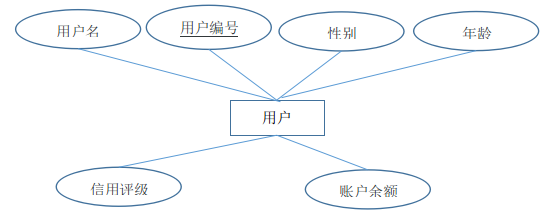


图4. 4 用户信息E-R图

备注：用户的账户余额达到10000时，可以享受黄金会员待遇。

普通客户——按照正常价格计费。

会员——享八折优惠，并且拥有会员专享特权，即可以使用会员专享车辆。

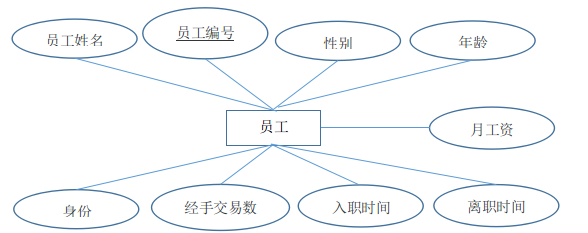


图4. 5 员工信息E-R图

备注：身份

0——普通员工

1——高级员工

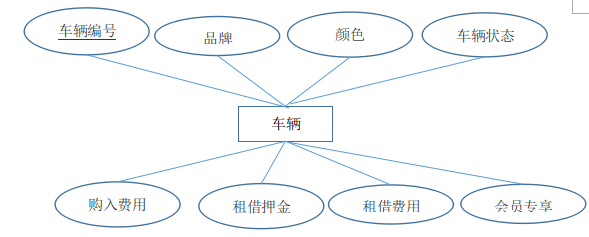


图4. 6 车辆信息E-R图

备注：车辆状态

A——空闲

B——正在使用

C——维修中

会员专享

Y——为会员专享车辆

N——非会员专享车辆

租借费用为每半个小时的价格，不足半个小时按照半个小时计算。

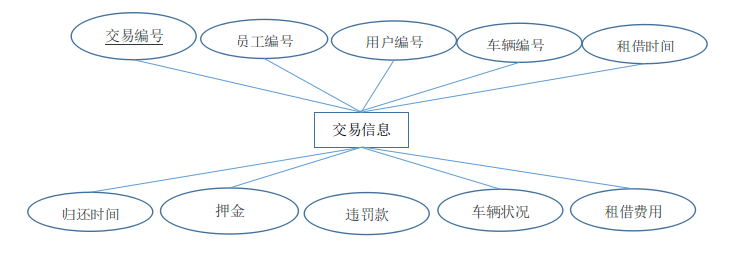


图4. 7 租借信息的E-R图

备注：车辆状况

A——优——正常缴费，并且提升用户信用评级

B——良——正常费用的1.2倍

C——一般——正常费用的1.5倍

D——差——正常费用的2倍，并且降低用户信用评级

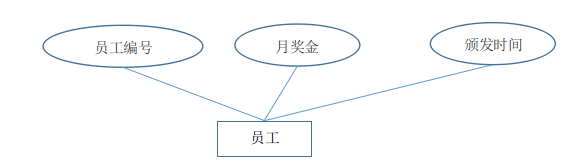


图4. 8 员工奖金的E-R图

然后，进行总体E-R图设计，如图4. 9所示。

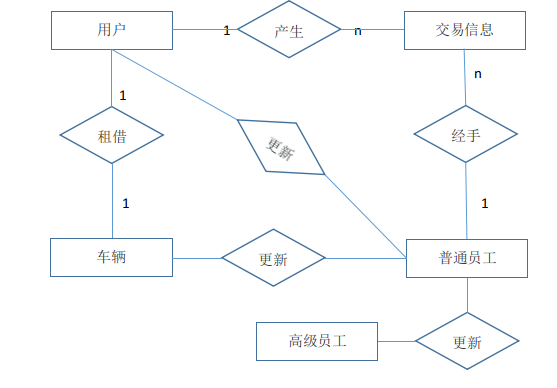


图4. 9 系统总体E-R图

### 4.4.2 关系表设计

完成了上述设计之后，便可以开始数据库的关系表设计。考虑到程序设计的简易性和通用性，本系统采用MySql提供后台数据库支持，在其中创建了下列五张表格。

表4. 1 用户信息表

|  |  |  |
| --- | --- | --- |
| 字段名 | 数据类型 | 长度与格式 |
| 用户名（Uname） | VarChar | 20 |
| 用户密码（Ucode） | VarChar | 12 |
| 性别（Sex） | Char | 1（M/W） |
| 年龄（Age） | Int | 短整数 |
| 信用评级（Cre） | Char | 1 |
| 账户余额（Acn） | Float | 浮点数 |
| 主码：用户编号（Unum） | | |
| 外码：无 | | |

表4. 2 员工信息表

|  |  |  |
| --- | --- | --- |
| 字段名 | 数据类型 | 长度与格式 |
| 员工姓名（Wname） | VarChar | 20 |
| 员工编号（Wnum） | VarChar | 12 |
| 员工密码（Ucode） | VarChar | 12 |
| 性别（Sex） | Char | 1（M/W） |
| 年龄（Age） | Int | 短整数 |
| 交易数（Tnum） | Int | 短整数 |
| 入职时间（Ctime) | DateTime | 时间 |
| 基本工资（Bsal） | Float | 浮点数 |
| 身份（Ide） | Char | 1（0/1） |
| 主码：员工编号（Wnum） | | |
| 外码：无 | | |

表4. 3 员工奖金表

|  |  |  |
| --- | --- | --- |
| 字段名 | 数据类型 | 长度与格式 |
| 员工编号（Wnum） | VarChar | 12 |
| 月奖金（Pride） | Float | 浮点数 |
| 发奖时间（Ctime) | DateTime | 时间 |
| 主码：员工编号（Wnum） | | |
| 外码：员工编号（Wnum） | | |

表4. 4 车辆信息表

|  |  |  |
| --- | --- | --- |
| 字段名 | 数据类型 | 长度与格式 |
| 车辆编号（Cnum） | VarChar | 12 |
| 品牌（Brand） | VarChar | 12 |
| 颜色（Color） | VarChar | 12 |
| 车辆状态（Sta） | Char | 1（A/B/C） |
| 购入费用（Fee） | Float | 浮点数 |
| 购买时间（Btime) | DateTime | 时间 |
| 押金（Cash） | Float | 浮点数 |
| 租借费用/h（Ree） | Float | 浮点数 |
| 会员专享（Vip） | Char | 1（Y/N） |
| 主码：车辆编号（Cnum） | | |
| 外码：无 | | |

表4. 5 租借信息表

|  |  |  |
| --- | --- | --- |
| 字段名 | 数据类型 | 长度与格式 |
| 订单编号（Tnum） | VarChar | 12 |
| 员工编号（Wnum） | VarChar | 12 |
| 用户名（Uname） | VarChar | 12 |
| 车辆编号（Cnum） | VarChar | 12 |
| 开始时间（Stm） | DateTime | 时间 |
| 结束时间（Ftm） | DateTime | 时间 |
| 租借费用（Cash） | Float | 浮点数 |
| 违罚款（Fine） | Float | 浮点数 |
| 车辆状况（Csta） | Char | 1（A/B/C/D） |
| 主码：交易编号（Tnum） | | |
| 外码：员工编号（Wnum），用户名（Uname），车辆编号（Cnum） | | |

### 4.4.3 数据库的代码实现

Create table User(

Uname Varchar(20) primary key,

Ucode Varchar(20),

Sex char(1),

Age int(4),

Cre Char(1),

Acn float(2)

)COMMENT='用户表';

Create table Worker(

Wname Varchar(20),

Wnum Varchar(12) primary key,

Wcode Varchar(20),

Sex char(1),

Age int(4),

Tnum int(4),

Ctime DATETIME,

Ide Char(1),

Bsal float(2)

)COMMENT='员工信息表';

Create table WorkerSal(

Wnum Varchar(12),

Prize float(2),

Gtime DATETIME,

foreign key (Wnum) references Worker(Wnum)

)COMMENT='员工奖金表';

Create table CarInfo(

Cnum Varchar(12) primary key,

Brand Varchar(12),

Color Varchar(12),

Sta char(1),

Fee float(2),

Btime DATETIME,

Cash float(2),

Ree float(2),

Vip Char(1)

)COMMENT='车辆信息表';

Create table RentInfo(

Tnum Varchar(12) primary key,

Wnum Varchar(12),

Uname Varchar(20),

Cnum Varchar(12),

Stm DATETIME,

Ftm DATETIME,

Cash float(2),

Fine float(2),

Csta char(1),

foreign key (Wnum) references Worker(Wnum),

foreign key (Uname) references User(Uname),

foreign key (Cnum) references CarInfo(Cnum)

)COMMENT='租借信息表';

## 4.5 详细设计与实现

### 4.5.1 开发环境

操作系统：Ubuntu 16.04LTS（Linux version 4.4.0-79-generic）

数据库后台系统：mysql 5.7.18

界面开发工具：Qt Create 4.2.1 Based on Qt 5.8.0

### 4.5.2 功能设计与实现

1. 数据库连接

程序运行的初始化阶段，完成后端数据库的连接操作。由于采用Qt作为开发工具，因此直接引用Qt的内部集成库QtSql/QSqlQuery，然后调用内部函数进行数据库的连接操作。具体实现可以参见下面代码。

QSqlDatabase dbconn;

dbconn = QSqlDatabase::addDatabase("QMYSQL");

dbconn.setHostName("localhost");//主机名字

dbconn.setDatabaseName("CarRentSys");//数据库名字

if(dbconn.open("root", "\*\*\*\*\*\*\*\*\*\*\*"))//用户名、密码，成功返回1

{

qDebug("成功链接数据库...");

query = (QSqlQuery)dbconn; //query绑定数据库操作引擎

w.show(); //显示登录界面窗口

}

1. 登录界面设计

系统共有四种登录选项，分别是“用户”、“普通员工”、“系统管理员”和“游客模式”。使用者在登录界面选中不同的登录选项，可以进入不同的操作界面，由此达到不同权限管理操作。其中，“游客模式”是免密码登录，其他三种会通过后台数据进行账户和密码匹配，只有匹配成功才能继续下一步。具体流程见图4. 10。

另外，还设置了用户注册接口，方便新用户的注册使用。最终界面见图4. 11。

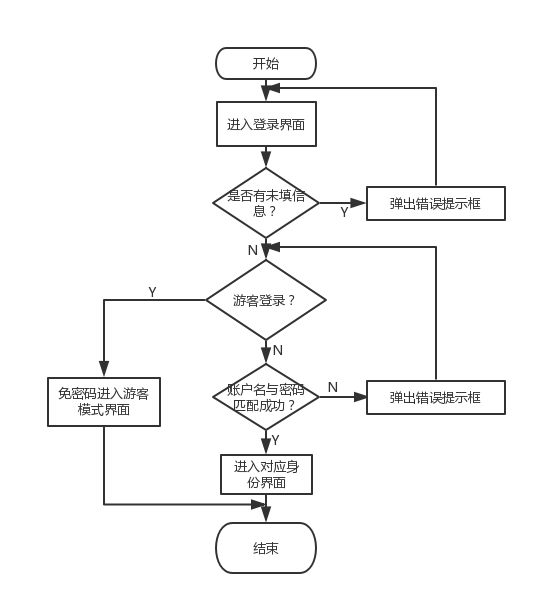


图4. 10 登录流程

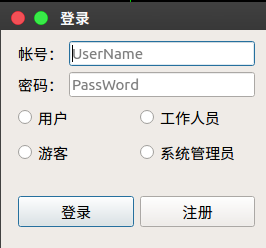


图4. 11 登录界面

1. 租车服务

租车服务是专为用户提供的一个操作接口，用户可以在这里浏览车辆基本信息以及进行租车操作。系统会进行车辆是否为VIP专享、用户余额是否充足以及用户是否有当前未完成订单等条件，对用户的租车行为进行判定，决定用户是否能够租用此车辆。具体流程见图4. 12。

其中，判断用户余额是否充足时，会根据用户的信用评级结合所选车辆的所需押金金额进行一个相应计算，最后得出相应结论。此部分界面见图4. 13。

另外，交易信息的订单编号是系统生成的一个随机数，由于进行了重复性检测，所以，内部保证了该编号的唯一性。

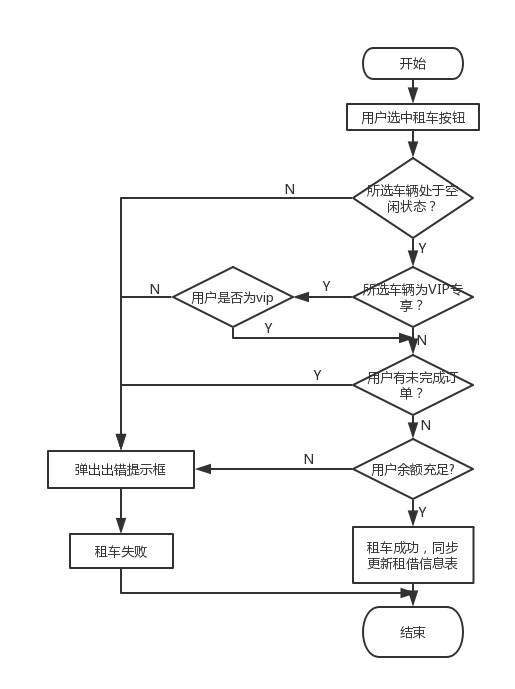


图4. 12 租车流程图

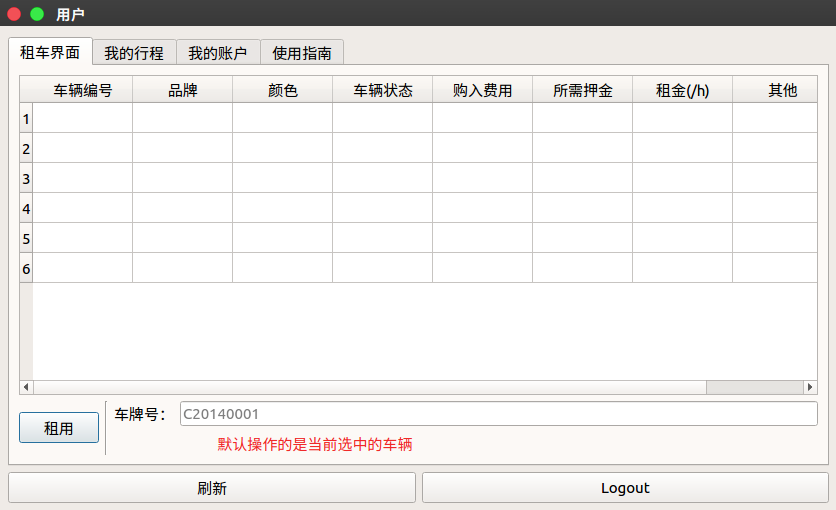


图4. 13 租车界面

1. 用户历史行程

由于数据库底层中并没有用户历史行程表，因为，此部分主要由订单信息表产生。通过订单信息中的外码依赖关系，可以查出当前用户的所有历史行程记录，包括当前的未完成订单等。

而对于当前行程，共有三种状态，分别为：正在使用中、结束提交申请中、已完成。由于在订单信息表中没有相应的状态标识，以及为了解决订单信息表对员工表的外码依赖关系，在员工信息表中特别设置了一个编号为W000000000的特殊员工，在生成订单时，经手员工编号全部被初始化为此编号。这样，可以根据经手员工编号以及订单的结束时间是否为空，从而判断但前行程的状态信息。具体判定流程见图4. 14。

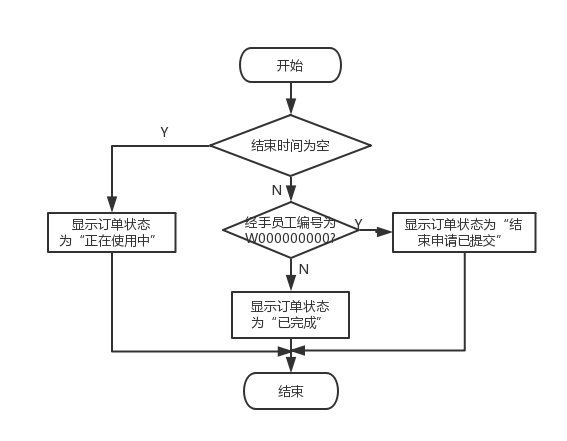


图4. 14 当前行程状态判断

1. 用户个人信息

在用户的个人信息界面，首先可以查看到当前用户的基本信息，包括用户名、等级、性别、年龄、账户余额和信用评级。然后，用户也可以根据自己的需求更改基本信息（只允许更改性别和年龄），修改密码以及进行账户充值相关操作。此部分界面见图4. 15。

其中，等级一栏是根据用户的账户余额进行判定的，当账户余额大于或者等于10000时，等级判定为“黄金会员”，否则，判定为“普通会员”。

信用评级一栏主要根据用户的历史行程记录进行评定，用户刚注册时，系统默认评级为A，当用户完成一个订单，然后员工对订单中车辆的使用状况判定为D时，用户的信用等级会下滑一级，直到为最后一级D为止。相应地，如果使用状况判定为A，用户的信用等级会上升一级，直到上升为最高一级A为止。

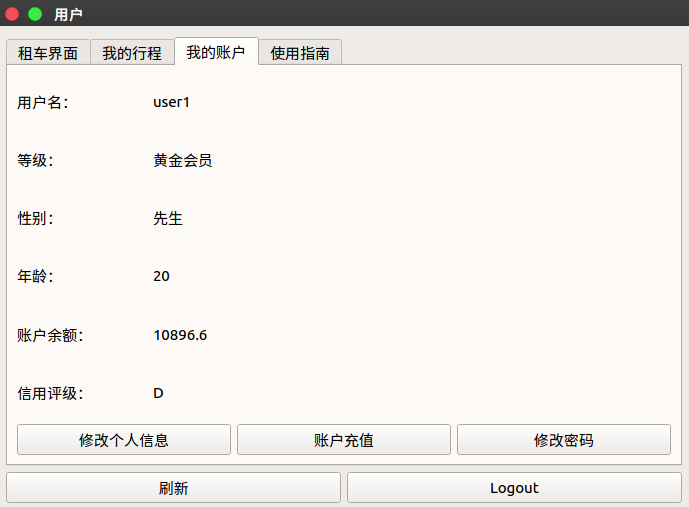


图4. 15 用户个人信息界面



图4. 16 车辆管理

1. 车辆管理

车辆管理界面是专为员工（包括系统管理员）提供的一个对车辆信息进行管理的操作界面，见图4. 16。

在此界面下，可以进行车辆的增加、删除以及车辆信息的修改等操作。增加车辆和修改车辆信息所用的都是同一个界面，如图4. 17所示。不同的是，增加车辆时，车辆的当前状态默认为“空闲”，不可改变；修改车辆信息时，当车辆为“使用中”时，不可以更改车辆的车牌号信息，其他两个状态时可以更改任意车辆信息。但是，当修改车牌号时，由于车牌号是主码，具有唯一性，所以不能将车牌号改为当前已经存在的。当然，首先会先判定车牌号是否发生改变。此部分可以参见下方核心代码。

if(CarNum != cnum)//判断车牌号是否被修改

{

const QString temp = "select Cnum from CarInfo where Cnum='" + cnum + "'";

query.exec(temp);

if(query.next())//查找成功，车牌编号已存在

{

QMessageBox::information(this, "Tips", "已存在！", QMessageBox::Ok);

return;

}

}

const QString temp2 = "update CarInfo set Cnum='" + cnum + "',Brand='" + brand + "',Color='" + color + "',Sta='" + sta + "',Fee=" + fee + ",Btime='" + time + "',Cash=" + cash + ",Ree=" + ree + ",Vip='" + vip + "' where Cnum='" + CarNum + "'";

if(query.exec(temp2))//更新车辆信息

QMessageBox::information(this, "Tips", "OK!", QMessageBox::Ok);



图4. 17 增加车辆界面

1. 客户管理

客户管理界面也是为员工和系统管理员提供的一个操作界面，在此界面下，只能删除客户，不能修改和增加客户，因为客户都是通过客户自行注册进行添加的。

删除客户时，由于客户原来可能存在历史行程订单，即其他表对其存在外码依赖关系，因此用户表中存在一个特殊用户，编号为“U000000”，每当删除用户时，会启动一个触发器，将租借信息表中的对应客户名改为“U000000”。这样操作，主要是为了不删除租车的历史记录，方便财务报表的生成。有关触发器部分可以见后面的介绍。

1. 订单管理

首先，普通员工和系统管理员都可以查看订单的所有信息，并且提供多种查询方式，包括按照租车时间段查询、按照车牌编号查询、只看待处理订单以及根据用户名查询，界面图见图4. 18。

特别地，对于普通员工，可以处理订单请求（即用户申请结束订单的请求）。主要考虑到系统管理人员的任务繁重，所以就只向普通员工开放了此操作接口。在处理请求过程中，员工可以对用户的车辆使用状况进行评定，该评定会直接影响用户的信用评级，也会影响订单的总费用。另外，员工也可以根据实际情况进行违罚款的处理。但是该违罚款并不是公司的收入，而是交通部分对用户车的一个罚款信息单。所以，后面并不会将违罚款计入公司的财务报表中。

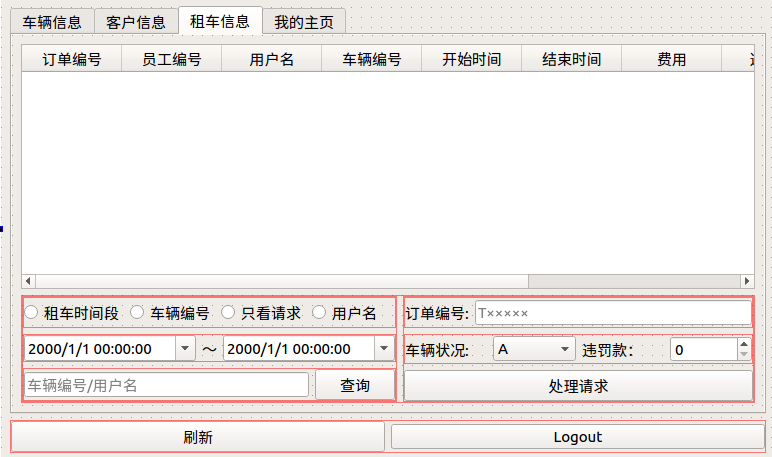


图4. 18 订单管理界面

1. 员工管理

员工管理界面是专为系统管理人员开放的一个操作界面，见图4. 19。

在此操作界面下，系统管理人员可以进行增加员工、删除员工以及更改员工的基本工资和发奖金等操作。与客户管理类似，当删除某员工时，为了保留内部数据的完整性，设置了一个编号为“W000000001”（“W000000000”已经被使用了）的员工，使得租车信息表和奖金表中的员工编号先被改为该特殊编号，再进行员工的删除操作。这些操作都通过触发器实现，后面会单独介绍。



图4. 19 员工管理界面



图4. 20 财务报表

1. 财务报表

财务报表也是只为系统管理人员开放的一个查询界面，见图4. 20。在这里，可以查看到选定时间段的“租车财务表”、“员工工资表”、“购车费用表”和“收入支出总和”。

此部分操作虽然只是一个简单的查询操作，但是在查询过程中几乎涉及了每一张表，比如为了生成员工工资表，需要查询员工表和奖金表。并且每次执行sql语句进行查询时，都需要对查询结果进行一次遍历操作。最终完成财务报表的展现。具体操作流程见图4. 21。

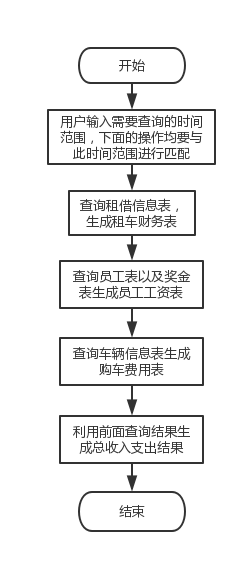


图4. 21 财务报表生成过程

1. 个人信息管理

员工与系统管理人员都可以拥有对自己个人信息进行修改的权限，以普通员工为例，见图4. 22。

在此界面下，展示了员工的所有基本信息，与用户的界面类似，也提供了修改密码和修改个人信息的操作接口。并且，主码字段（即员工编号）是不允许进行修改的。



图4. 22 员工个人信息管理

### 4.5.3 触发器实现

从4.5.2小节的介绍中，了解到在删除客户和员工时，为了满足外码的依赖关系，需要提前对租借信息表和奖金表中的外码进行修改。于是，考虑到操作的简易性，特别设计了两个触发器实现这些功能。这两个触发器的代码如下所示。

delimiter $$/\*定义结束符为$$\*/

create trigger DeleteUser before delete on User

for each row

begin

/\*在删除用户前将租借信息表中的用户名改为U000000\*/

update RentInfo set Uname='U000000' where RentInfo.Uname = old.Uname;

end$$

delimiter ;/\*还原结束符\*/

delimiter $$/\*定义结束符为$$\*/

create trigger DeleteWorker before delete on Worker

for each row

begin

/\*在删除员工前将租借信息表中经手员工编号改为W000000001\*/

update RentInfo set Wnum='W000000001' where Wnum=old.Wnum;

/\*同步删除该员工的奖金信息\*/

delete from WorkerSal where WorkerSal.Wnum=old.Wnum;

end$$

delimiter ;/\*还原结束符\*/

## 4.6 系统测试

### 4.6.1 测试数据说明

预先录入的员工信息见图4. 23，员工奖金信息见图4. 24，用户信息见图4. 25，车辆信息见图4. 26，租借订单信息见图4. 27。接下来以这些数据进行系统的测试工作。



图4. 23 员工信息

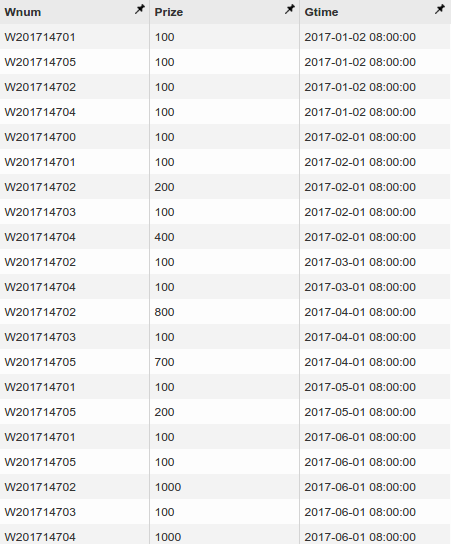


图4. 24 员工奖金

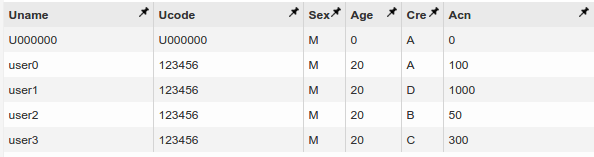
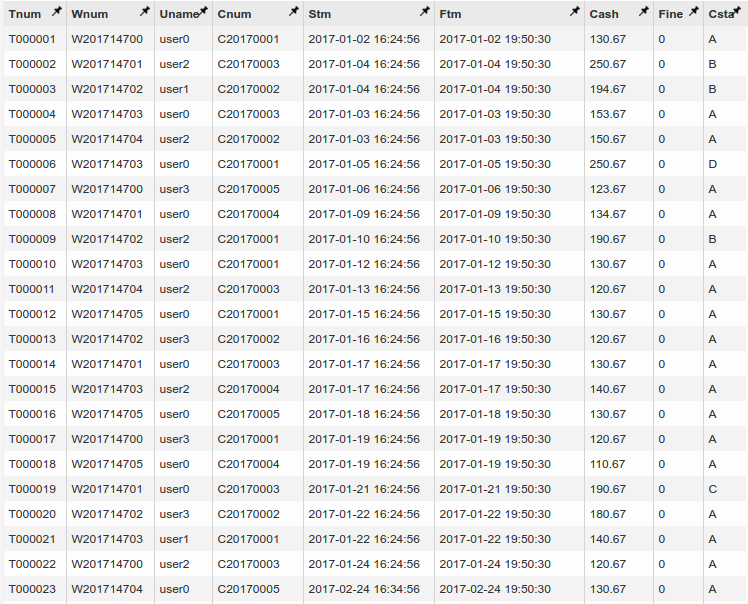


图4. 25 用户信息



图4. 26 车辆信息



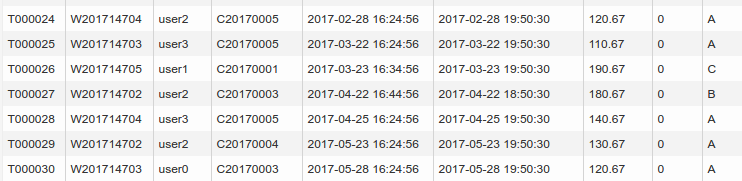


图4. 27 租借信息

### 4.6.2 登录测试

首先进行登录错误测试，以用户user0进行登录，从图4. 25可以查出该用户的密码为“123456”，此处输入密码“111111”，点击登录，系统弹出错误提示框。显示“用户名或密码不存在”，如图4. 28所示。接着，以正确的用户名和密码登录，进入用户操作界面，如图4. 29所示。该界面展示了车辆的基本信息，与图4. 26显示结果一致。

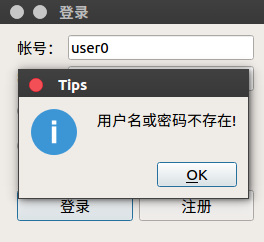


图4. 28 登录错误



图4. 29 用户登录界面

### 4.6.3 租车功能测试

首先，通过查看我的行程（见图4. 30），得知此时我的行程中没有未完成订单，于是先选中一辆比亚迪（该车正在维修中），点击租用，系统提示“该车还在修养之中”，如图4. 31。再点击一辆奔驰（仅限vip），系统提示“很抱歉，您不是vip用户，无法租用此车！”。前往“我的账户”，查看到当前账户余额为100，等级为“普通会员”，见图4. 33。通过账户充值操作（见图4. 34），充值金额10000,使得等级提升为“黄金会员”，充值后的结果见图4. 35。

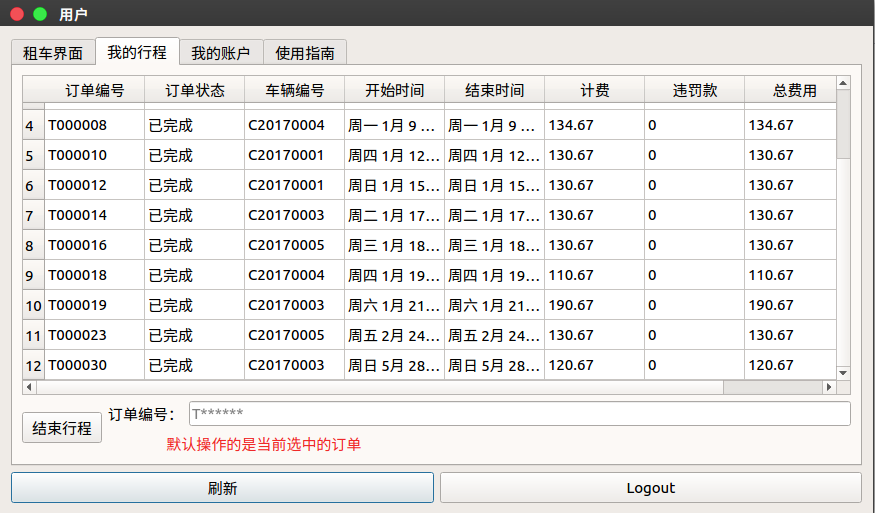


图4. 30 我的行程（1）

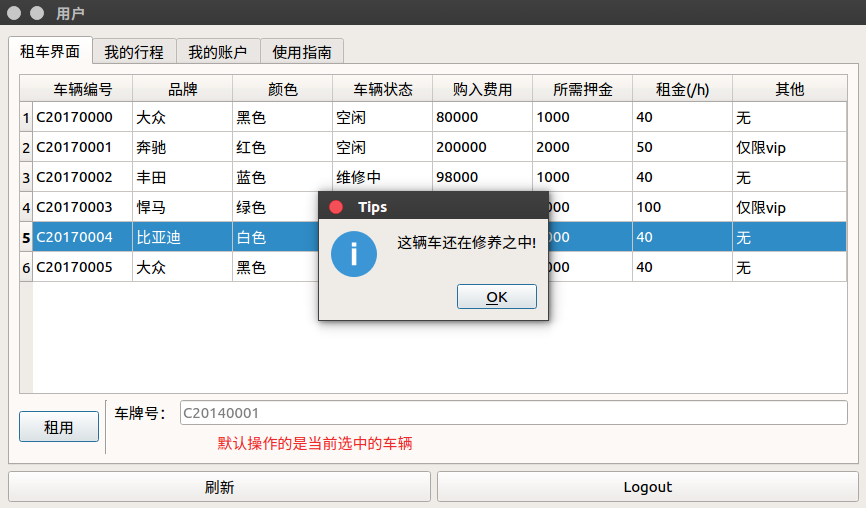


图4. 31 租车（1）

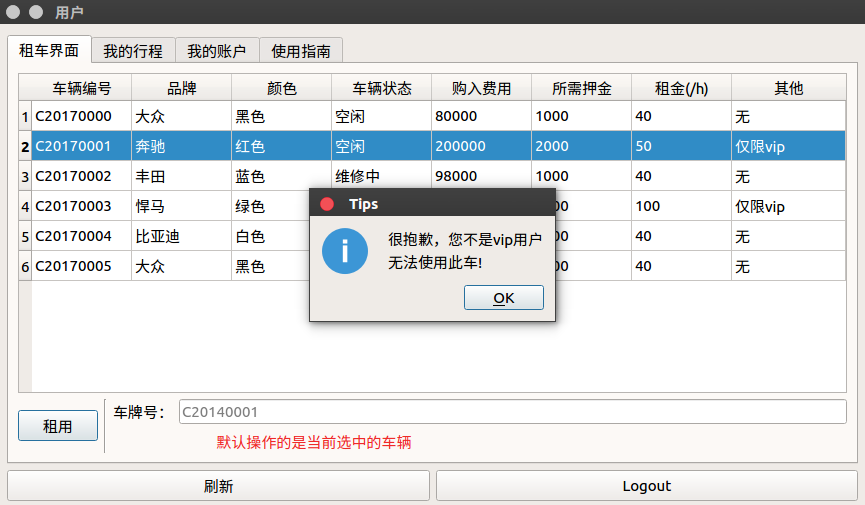


图4. 32 租车（2）



图4. 33 我的账户（1）

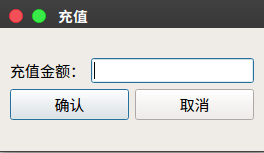


图4. 34 账户充值



图4. 35 我的账户（2）

接着，继续租用前面选中的那辆奔驰车，系统弹出“租车成功”的提示框，见图4. 36。然后，查看我的行程，点击下方的“刷新”按钮，发现多了一个正在使用的行程订单，生成时间为当前时间，在同步计费当中，见图4. 37。

最后，用户选中当前行程，点击“结束”按钮，系统弹出“订单结束申请已提交”的提示框，点击“刷新”后，原来行程状态由“正在行驶中...”变成了“结束申请处理中...”，见图图4. 38。

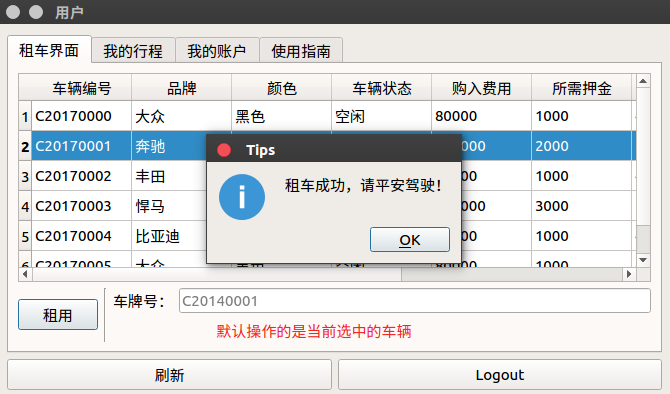


图4. 36 租车（3）

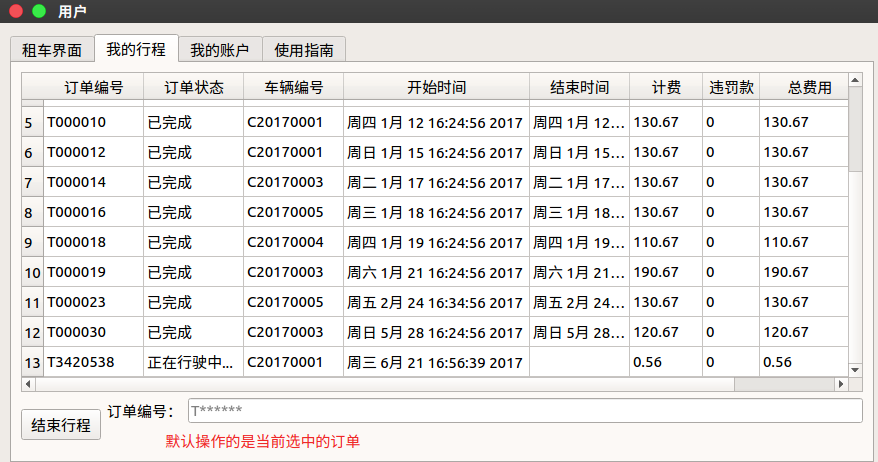


图4. 37 我的行程（2）



图4. 38 我的行程（3）

### 4.6.4 基本信息测试

切换到“我的账户”界面，点击“修改个人信息”，系统弹出一个“修改信息”界面，如图4. 39所示。由于用户名为唯一标识码，注册之后就不允许用户进行修改，所以显示为暗色。此处将用户性别更改为“女”，年龄更改为“30”。点击确认后，更新后的页面中，发现信息都进行了同步更改。由于密码等功能修改原理与基本信息修改原理一致，所以，后面也就不再单独测试。

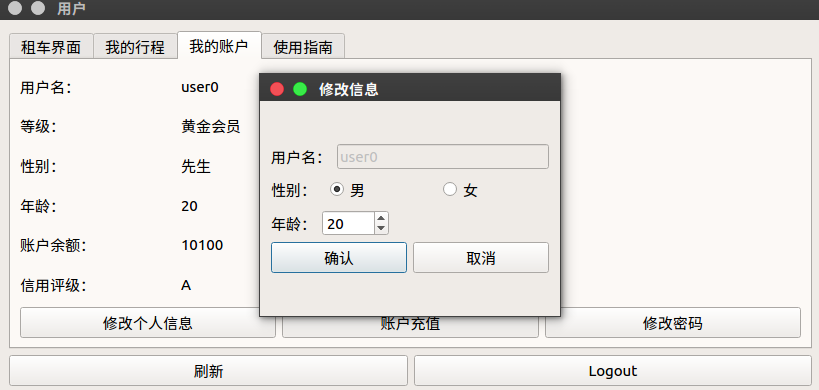


图4. 39 修改个人信息（1）

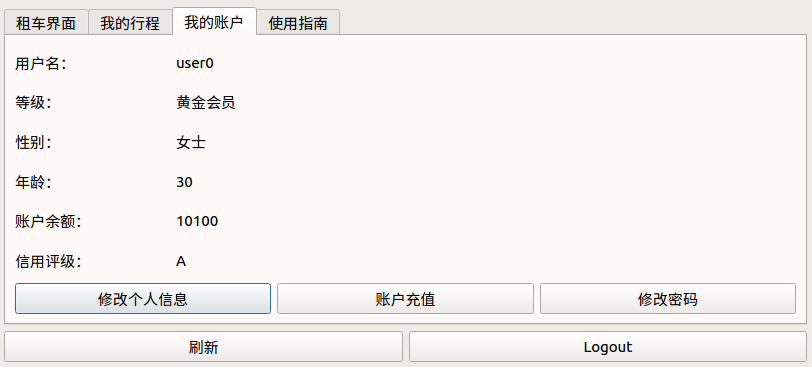


图4. 40 修改个人信息（2）

### 4.6.5 订单处理测试

以员工编号“W201714700”，密码“123456”登录之后，进入“租车信息”界面。该界面展示的是所有订单，如图4. 41所示。从图中红色圈住的部分可以看到上面测试中刚提交的一个订单申请，其员工编号为“W00000000”。也可以点击下方的“只看请求”选项，再点击“查询”，可以看到界面中只有一个请求订单了，见图4. 42。接着，选中该订单，在右下角有一些处理的操作选项，选中车辆状况判定为D（为了检测用户的信用评级是否由原来的“A”下降为“B”），违罚款设置为200。然后，点击“处理请求”。系统弹出提示框显示“处理成功”。然后，再通过下方按用户名查询，输入“user0”，点击“查询”，结果如图4. 43所示。

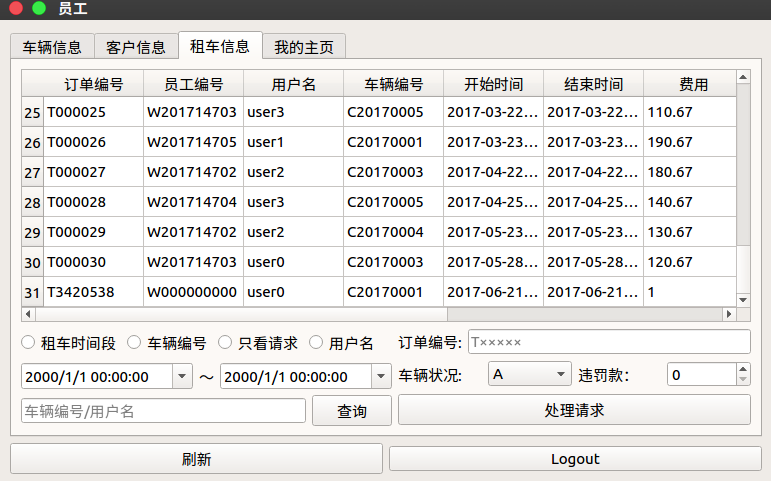


图4. 41 订单处理（1）

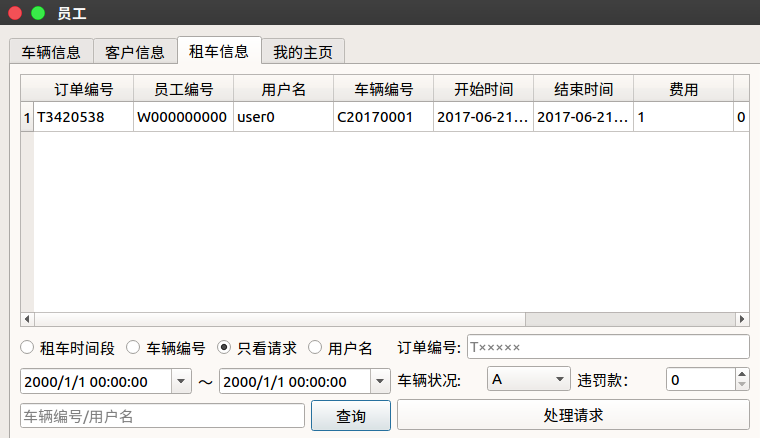


图4. 42 订单处理（2）



图4. 43 订单处理（3）

然后，再以user0登录，查看“我的行程”，看到订单状态有“结束申请提交中”变为了“已完成”，见图4. 44。然后，切换到“我的账户”，发现账户余额减了202,信用等级也由原先的“A”降为了“B”，由于余额小于10000,故身份降为“普通会员”，见图4. 45。



图4. 44 我的行程（4）

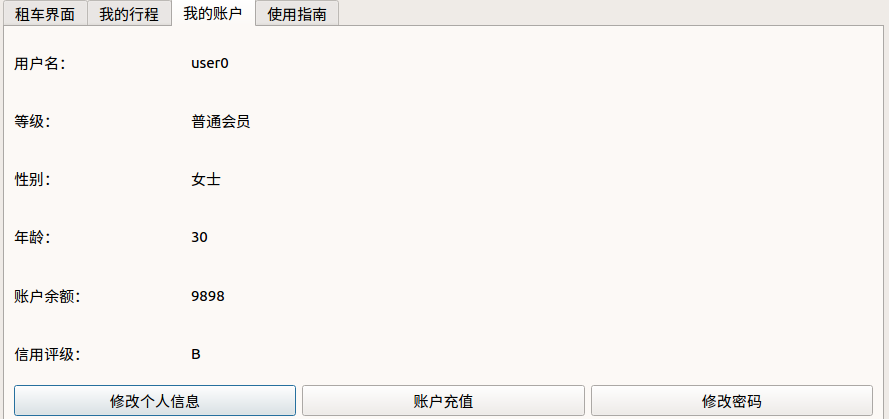


图4. 45 我的账户（3）

下面，以帐号“W201414775”，密码“960606”，选中“系统管理员”，登录系统，进行车辆管理、员工管理和财务报表这三项功能的测试。

### 4.6.6 车辆管理测试

车辆管理的总界面如图4. 46所示。选中左下方的“增加车辆”按钮，系统弹出“增加车辆”操作框，在操作框中录入需要增加的车辆信息（见图4. 47）后，点击“确认”按钮，系统提示“OK!”。然后，在车辆信息界面中最后一栏出现了刚添加的车辆信息，见图4. 48。



图4. 46 车辆管理（1）



图4. 47 增加车辆

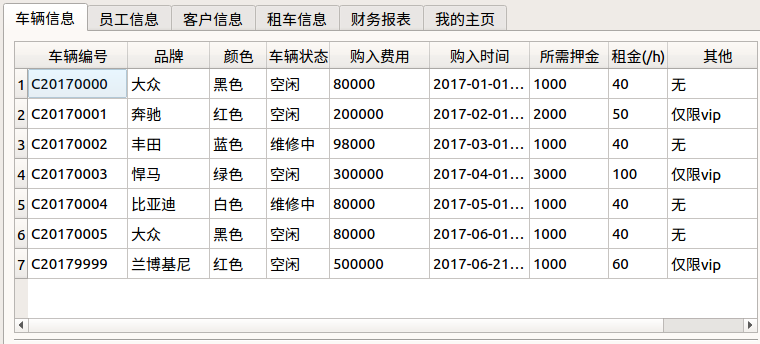


图4. 48 车辆管理（2）

接着，点击“修改信息”按钮，系统弹出一个与添加车辆类似的界面，此处，将编号“C20179999”车子的颜色修改为“蓝色”，状态改为“维修中”，结果如图4. 49所示。

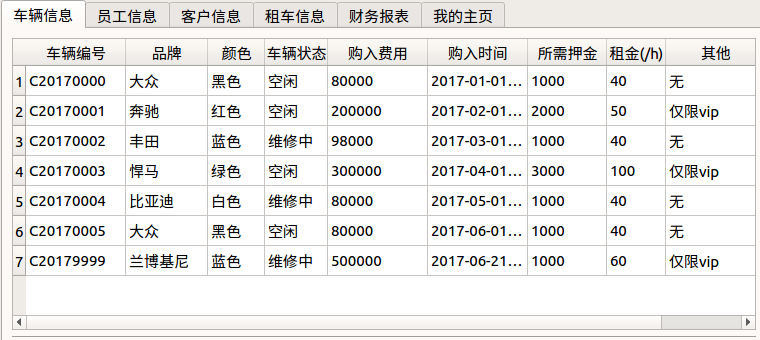


图4. 49 车辆管理（3）

然后，点击“删除车辆”按钮，将编号为“C20170000”的车辆删除，结果如图4. 50所示。



图4. 50 车辆管理（4）

### 4.6.7 员工管理测试

员工信息管理界面如图4. 51所示。由于增删员工和修改员工基本工资与4.6.5实现原理类似，因此，此处只测试“颁发奖金”的功能。对员工“W20171470”颁发1000元奖励，颁发之后，系统弹出提示框，显示“发奖金成功”，见图4. 52。

之后会在财务报表测试环节中检测是否实现成功。



图4. 51 员工信息管理



图4. 52 颁发奖金

### 4.6.8 财务报表测试

切换到“财务报表”界面，将查询时间设为“2017/5/1 12:00 ~ 2017/6/22 12:00”，查询到的结果如图4. 53所示。

图中，左上部财务报表中红色框住的部分为前面测试所生成的订单，订单金额为2元，是因为没有将违罚款计入，前面强调过此系统认为违罚款是交通部门的收入，不计入本公司的收入中。员工工资表中红色框住的部分可以看到编号“W201714700”的员工的奖金收入为1000,这也印证了4.6.5的测试结果。所有员工的基本工资根据“基本工资=月基本工资×月数”，在计算中，“月数=查询间隔的天数/30”。左下角的购车费用表，红色框住的部分显示的是前面测试中购入的车辆所花费用。右下角框住的是公司在这段时间内的收支总额统计结果，当然，结果为负数，因为用户量太少了！

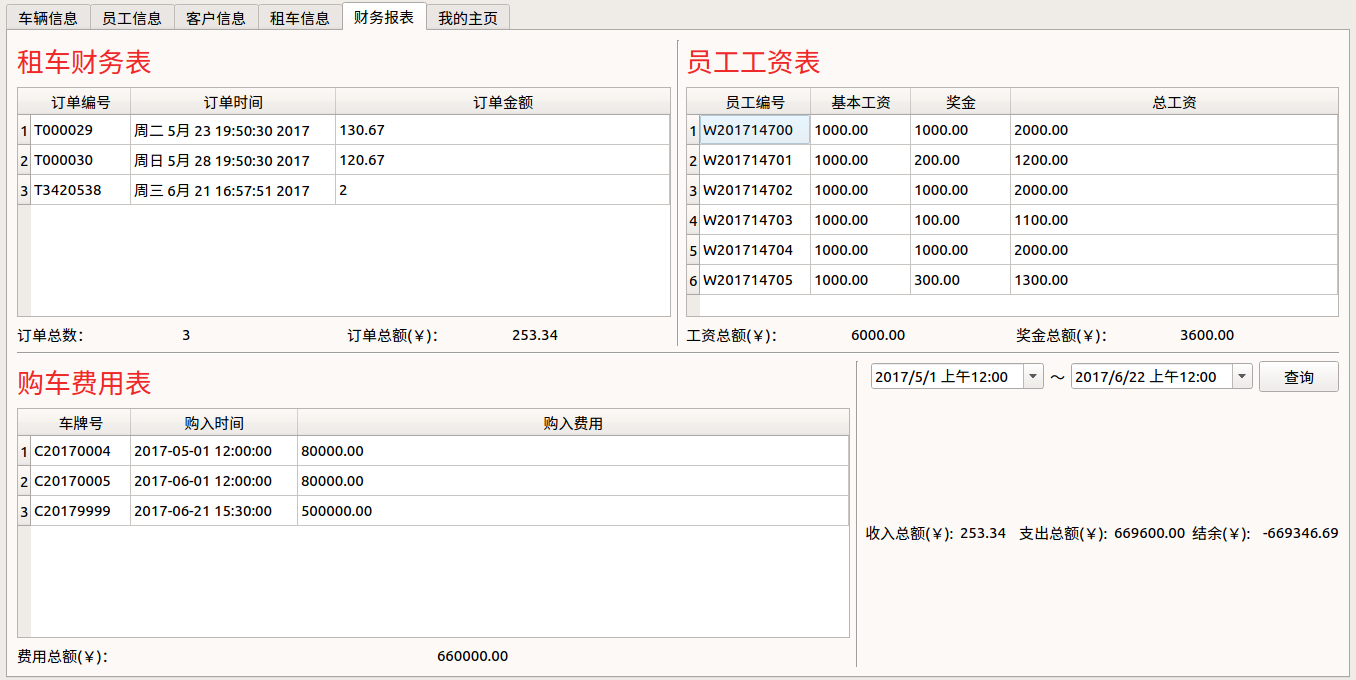


图4. 53 财务报表

到这里，测试工作就全部完成了，由于有些管理模块功能大都类似，所以，测试结果没有在上面展示出来。另外，测试样例也不齐全，还有很多改进的地方。

## 4.7 系统设计与实现总结

此次“汽车租赁系统”设计和实现中主要经历了以下几步。

（1）首先，根据模拟市场调研情况，进行了需求分析，主要包括系统的功能需求、性能需求以及数据项说明等。这部分为后续数据库的搭建奠定了基础。

（2）接着，在系统总体设计中，对系统进行了整体性分析，确定了系统为C/S架构模式，分析总结了系统的功能模块以及完成了系统业务流程图的设计。此部分进一步明确了系统的结构体系，为后面数据库开发以及应用设计作准备。

（3）然后，进入数据库的具体实现中。在数据库设计步骤，完成了数据库E-R图和数据库关系表的搭建工作。在底层使用mysql提供数据库支持，并且完成了数据库的后台实现。

（4）完成后台数据库后，开始开发前端应用系统，借助Qt开发出一个简易的数据库前端应用，共有四种登录角色，分别是“游客”、“用户”、“员工”和“系统管理员”。每种角色登录之后，进入不同的操作管理界面，实现了不同权限的管理。并且，通过触发器设计，丰富了系统的功能实现逻辑。

（5）最后，借由前端与后台的结合，实现了“汽车租赁系统”的设计。

# 5 课程总结

## 5.1 主要工作

1. 掌握了包括增删改查在内的基本SQL操作。
2. 学习了视图的创建，实现了触发器、函数和存储过程。
3. 掌握了对mysql中DATETIME时间格式的应用机制。
4. 在DBMS综合应用过程中，了解了mysql数据库的备份机制、身份权限配置、查询性能分析、数据的导入与导出功能以及集成开发环境的应用等。
5. 在课程设计过程中，了解并掌握了一个数据库应用系统的实现过程，包括对数据库的需求分析、综合设计以及前后端的整体开发等。

## 5.2 心得体会

在学习数据库之前，对于有数据储存需求时，自己第一下想到的就是创建文件，然后利用文件进行数据的读取和写入操作。然而，在此次数据库应用系统的设计与实现过程中，当尝试着使用数据库系统进行数据的增删改查等工作时，发现竟是如此的方便，比原先利用文件进行读写方便了简直不是一点点。

然而，在整个学习过程中，难免碰到一些困难与挑战。比如，在学习怎样正确地在Qt环境中使用后台数据库时，由于版本差异化影响使得自己在此处耗费了不少的时间；还有，由于自己前期数据库设计的并不合理，导致写到后面时，对前面的数据关系表进行了几次大的修改，而每一次大的修改就需要将前面写过的代码逻辑进行相应的修改，可见，在设计初期构建一份完善的数据库是多么的重要啊！

实质上，落笔之前，依然觉得自己的数据库设计还有很多可以改进的地方，比如设计的租赁系统其实并不符合真实环境需求，对于一些特殊情况也缺乏应有的考量，当然，最大的不足，还在于界面设计的真的有点丑，可能当初觉得数据库课程设计的重心应该放在数据库设计上，所以就没有耗费精力在界面的设计上了。

最后，不得不说，通过这次课程实践，自己收获还是蛮多的，也感谢老师和助教一路以来对自己的帮助，感恩！

# 附录

课程实践源代码见电子档部分。