

**数据库系统原理实践报告**

|  |  |
| --- | --- |
| 项目名称： | 数据库系统原理实践 |
| 姓 名： | 饶鸿斌 |
| 专 业： | 计算机科学与技术 |
| 班 级： | CS1703 |
| 学 号： | U201714660 |
| 指导教师： | 丁晓峰 |

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

2020年 5 月 24 日

**教师评分页**

|  |  |
| --- | --- |
| 子目标 | 子目标评分 |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
|  |  |

目 录

[1 课程任务概述 1](#_Toc41258159)

[2 软件功能学习部分 3](#_Toc41258160)

[2.1 任务要求 3](#_Toc41258161)

[2.2 完成过程 3](#_Toc41258162)

[2.3任务总结 5](#_Toc41258163)

[3 Sql练习部分 5](#_Toc41258164)

[3.1 任务要求 5](#_Toc41258165)

[3.2 完成过程 5](#_Toc41258166)

[3.3任务总结 27](#_Toc41258167)

[4 综合实践任务 28](#_Toc41258168)

[4.1系统设计目标 28](#_Toc41258169)

[4.2 需求分析 28](#_Toc41258170)

[4.3 总体设计 28](#_Toc41258171)

[4.4 数据库设计 28](#_Toc41258172)

[4.5 详细设计与实现 28](#_Toc41258173)

[4.6 系统测试 28](#_Toc41258174)

[4.7 系统设计与实现总结 28](#_Toc41258175)

[4 课程总结 28](#_Toc41258176)

[附录 28](#_Toc41258177)

# 1 课程任务概述

实验一：软件功能学习部分

1.了解并练习数据库管理系统软件的备份功能。

2.了解并练习在数据库上配置用户权限的操作。

实验二：Sql练习部分

1.建表

2.数据更新

3.查询

4.了解系统的查询性能分析功能（选做）

5.DBMS函数及存储过程和事务（选座）

**关于正文章节标题的排版要求：**

报告章标题称为一级标题，章内小节标题依次分为二级标题、三级标题等。一级标题的编号用数字1，2，…编制；二级标题的编号用1.1，1.2，…编制；三级标题的编号用1.1.1，1.2.1，… 编制；四级及以后各级标题可依此类推。建议标题不超过3级（如1.1.1），超出部分可根据需要使用(1)，①，A，a)，…等形式描述。

标题编号与标题文字之间均用空格隔开，如：“1 引言”、“2.1 需求分析”。报告正文的一级标题（章）须另起一页居中排版。

1级标题宋体小三号字加黑，单倍行距，段前、段后均12磅间隔。

2级标题宋体四号字加黑，1.25倍行距，段前6磅，段后0磅间隔。

3级标题宋体小四号字加黑，单倍行距，段前3磅、段后8磅间隔。

**关于正文中的图的排版要求：**

正文中所有插图要求图面整洁，布局合理，线条粗细均匀，圆弧连接光滑，尺寸标注规范。所有曲线、图表、线路图、流程图、程序框图、示意图等不准徒手画，必须按国家规定标准或工程要求采用计算机或手工绘制。

所有插图均应有图号和图名。图号按章编，如第2章的图为图2.1、图2.2、…，第3章的图为图3.1、图3.2、…等。图名是插图的名称，扼要概括图的内容，字数不宜太多。图号和图名写在图下方，并相对于图居中排版。少数图有图注，图注写在图下面且字号应比图号、图名的字小一号，图名和图注后面均不加标点符号。

所有插图均应在正文中予以引用。引用某插图时，一般写为“…见图x.y”或“图x.y是…”。正文中的插图均须安排在文中第一次引用到该图的正文下面，要求先见文，后见插图，且图一般不跨页绘制。

图中文字、图号和图名，统一采用小五号宋体。

**关于正文中的表的排版要求：**

表格由表号、表名、表头、表身等组成。表号按章编，如第2章的表为表2.1、表2.2、…，第3章的表为表3.1、表3.2、…等。表名是表格的名称，扼要概括表的内容，字数不宜太多。表号、表名放在表的正上方，相对于表体居中排版。表号及表名后不加标点。表头包括栏头、行头，与表身一起构成表格的主体。表中的竖称为栏，横格称为行。表身的内容，一般包括：数据、文字、公式和表图等。表内的数据对应位要对齐。少数表有表注，表注写在表下面且字号应比表号、表名的字小一号。

所有表格均应在正文中予以引用。引用某表格时，一般写为“…见表x.y”或“表x.y是…”。表格应尽量靠近正文的叙述，一般应先见文，后见表，表不跨节。表格允许转页。表格转页部分可以不写表号和表名，但要重复书写表头，并在表头右上角写“（续）”字标注。

表中文字、图号和图名，统一采用小五号宋体。

**关于正文中公式的排版要求：**

公式一般另行居中写，公式末不加标点。若公式前有文字，如例、解、证、假定等，文字顶格写，公式仍居中写。一行写不下时，公式允许转行。公式转行需处理得当，做到既意义正确，又使版面美观匀称。

公式要有编号，公式编号按章编，如第2章的公式为(2.1)、(2.2)、…，第3章的公式为(3.1)、(3.2)、…等。公式编号写在公式右侧行末顶边线，并加圆括号。

公式一般应在正文中予以引用，引用时以公式编号指示公式。正文中常有公式中表示量的符号说明，采用“式中”二字作为标志。一般可写成接排形式，如“式中，A指……；B指……”。

**关于实践报告的目录格式要求：**

目录是课程实验报告的纲要。正文的各级标题（一般最多取三级）、附录应编入目录，但目录本身不出现在其中。

目录要求层次清晰，含标题及对应的起始页号。目录的最后 “附录”无章节号。

课程实验报告正文、参考文献和附录页面，使用“1，2，3，… ”编连续页码。页码应标在页面的右下角。

目录中正文的各级标题名称、参考文献和附录及其对应的起始页号，务必与报告中正文的各级标题名称、附录及其对应的起始页号保持一致。

# 软件功能学习部分

## 2.1 任务要求

1）练习SQL Server或其他某个主流关系数据库管理系统软件的备份方式，要求要有通过数据库的软件功能进行的备份和通过文件形式的脱机备份。

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 数据库备份

本次实验使用的是MySql数据库以及提供图形化界面的Navicat。

本次实验任务二部分的数据库名为lab1，我们使用该数据库来进行备份练习。在Navicat中点击备份，选中新建备份，依据提示进行备份。备份练习结果如图2.1所示：

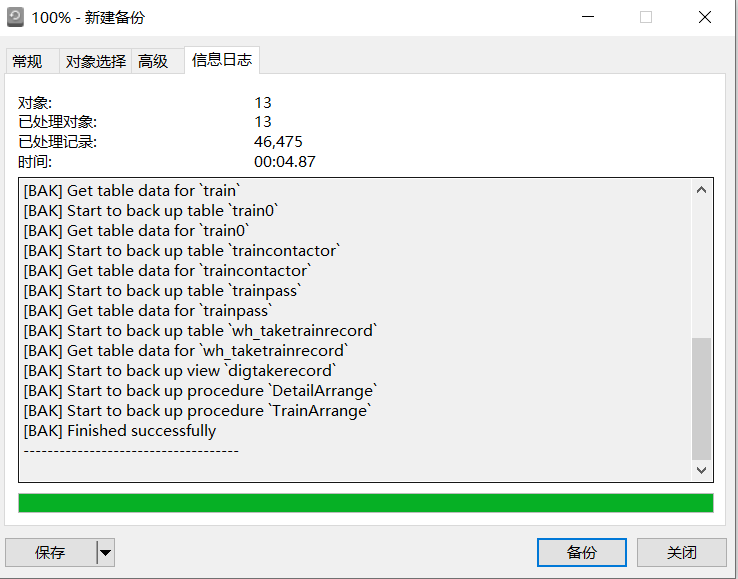


图2.1 新建备份结果

备份的信息将在Navicat中显示，显示的内容如图2.2所示：

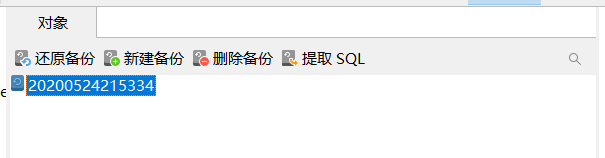


图2.2 备份结果显示

在新建备份后，右键点击还原备份即可还原备份数据，如图2.3所示：

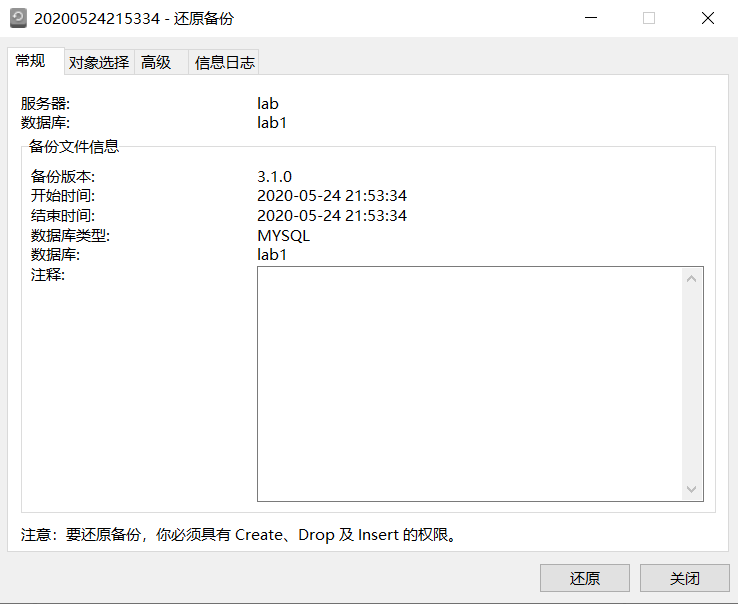


图2.3 还原备份

假设需要采用通过文件形式的脱机备份，则可以在对应目录下将整个数据库的文件拷贝下来。

### 2.2.2 用户权限配置

用户权限配置将在Navicat中完成，如图2.4所示：

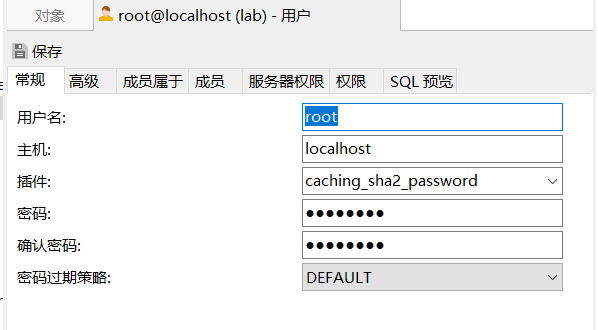


图2.4 用户权限配置

在图2.4对于的服务器权限和权限中可以针对用户进行相应的权限配置。

## 2.3任务总结

本次实验使用了图形化界面软件Navicat，所以难度不是很大。基本上主要集中对软件的使用上。

# 3Sql练习部分

## 3.1 任务要求

1）创建与“疫期乘坐列车”相关的关系，包括主码和外码的说明。同时向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。

2）数据更新。完成增删改操作、批处理操作、数据导入导出操作、创建视图操作、触发器实验等。

3）查询。用一条SQL语句完成16个小题中的查询需求。

4）了解系统的查询性能分析功能（选做）。

5）DBMS函数及存储过程和事务（选座）。

## 3.2 完成过程

### 3.2.1 建表

1）创建与“疫期乘坐列车”相关的关系。

创建表的sql语句如下所示：

-- 车站表【车站编号，车站名，所属城市】

-- Station (SID int, SName char(20), CityName char(20))

-- 其中，主码为车站编号。

create table station(

SID int,

sname char(20),

cityname char(22),

constraint PK\_station primary key(SID)

);

-- 车次表【列车流水号，发车日期，列车名称，起点站编号，终点站编号，开出时刻，终点时刻】

-- Train (TID int, SDate date, TName char(20), SStationID int, AStationID int, STime datetime, ATime datetime)

-- 其中，TID为主码，(列车名称，发车日期)为候选码；SStationID和AStationID都来源于车站表的SID。

create table train(

TID int,

SDate date,

TName char(20),

SStationID int,

AStationID int,

STime datetime,

ATime datetime,

constraint PK\_train primary key (TID),

constraint UK\_date\_tname unique(SDate,TName),

constraint FK\_train\_SStationId foreign key(SStationID) REFERENCES station(SID),

constraint FK\_train\_AStationId foreign key(AStationID) REFERENCES station(SID)

);

set FOREIGN\_KEY\_CHECKS =0;

drop table train;

set FOREIGN\_KEY\_CHECKS =1;

-- 建立部分车次表，以转存excel表数据

-- 增加的startID：起点站编号、destID：终点站编号;depart 出发时间；days:第几天到达

create table train0(

trainno char(11) primary key,

startID int,

start char(10),

destID int,

dest char(10),

depart time,

arrival time,

days int,

distance int

);

update train0 join station on train0.start = station.sname

set startid = station.SID;

update train0 join station on train0.dest = station.sname

set destid = station.SID;

create procedure TrainArrange()

begin

declare TID1 int;

declare SDate1 date;

declare TName1 char(20);

declare SStationID1 int;

declare AStationID1 int;

declare STime1 time;

declare ATime1 time;

declare d int;

declare arrivalDate Date;

declare done boolean default 0;

declare train\_cur cursor for select trainno,startID,destID,depart,arrival,days from train0 order by trainno;

declare continue handler for not FOUND set done =1;

open train\_cur;

fetch train\_cur into TName1,SStationID1,AStationID1,STime1,ATime1,d;

set TID1=1;

while not done do

set SDate1='2020-01-10';

while(SDate1<='2020-01-25') do

if(d<2) then set arrivalDate = SDate1;

else set arrivalDate = date\_add(SDate1,interval d-1 day);

end if;

-- 2020-01-23至2020-4-07武汉出发和武汉到达的车次停运

if not (

((arrivalDate>'2020-01-22' and arrivalDate<'2020-04-08') and

(AStationID1 in (select SID from station where cityname ='武汉'))) or

((SDate1>'2020-01-22' and SDate1 <'2020-04-08')and

(SStationID1 in (select SID from station where cityname ='武汉')))

)

then insert into train values(TID1,SDate1,TName1,SStationID1,AStationID1,

concat(SDate1, " ", STime1),

concat(arrivalDate, " ", ATime1));

set TID1 =TID1 +1;

end if;

set SDate1 =date\_add(SDate1,interval 1 day);

end while;

fetch train\_cur into TName1,SStationID1,AStationID1,STime1,ATime1,d;

end while;

close train\_cur;

end ;

drop procedure TrainArrange;

call TrainArrange();

-- 车程表【列车流水号，车站序号，车站编号，到达时刻，离开时刻】

-- TrainPass (TID int, SNo smallint, SID int, STime datetime, ATime datetime)

-- 其中，主码为(TID,SNo)。SID来源于车站表的SID。

create table trainpass(

TID int,

SNo smallint,

SID int,

STime datetime,

ATime datetime,

CONSTRAINT PK\_TrainPass primary key(TID,SNo),

CONSTRAINT FK\_TP\_TID FOREIGN key(TID) REFERENCES Train(TID),

CONSTRAINT FK\_TP\_SID FOREIGN key(SID) REFERENCES station(SID)

);

-- 中间表，导入部分车次的运行时刻表：车次，站次，···

create table timetable(

trainno char(11),

SNo smallint,

SID int,

sname char(20),

STime time,

ATime time,

D int,

PRIMARY key(trainno,SNo)

);

-- 填写SID车站的编号

update timetable join station on station.sname = timetable.sname

set timetable.SID = station.SID;

-- 遍历train表，每一行只取TID,SDtate和TName

-- 根据TNAME从timeable 中取出SNO,SID,ATIME,D,WTHER TRAIN0 = TNMAE对应的额遍历

-- 遍历每一个站点，从出发日期算出到站时间，离站时间然后插入

delimiter $$

create procedure DetailArrange()

begin

declare TID1,SNO1,SID1 INT;

declare SDate1 date;

declare TName1 char(20);

declare STime1 time;

declare ATime1 time;

declare d1 int;

declare arrivalDate Date;

declare departDate Date;

declare outer\_done boolean default 0;

declare inner\_done boolean default 0;

declare train\_cur cursor for select TID,SDate,TName from train order by TID;

declare tt\_cur cursor for

select SNO,SID,STime,ATime,D from timetable where trainno =TName1 order by SNO;

declare continue HANDLER for not found

begin

set outer\_done=1;

set inner\_done=1;

end;

open train\_cur;

fetch train\_cur into TID1,SDate1,TName1;

while not outer\_done do

set inner\_done =0;

open tt\_cur;

fetch tt\_cur into SNO1,SID1,STime1,ATime1,D1;

while not inner\_done do

if(D1<2) then set arrivalDate =SDate1;

else set arrivalDate = date\_add(SDate1,interval D1-1 day);

end if;

if STime1 <ATime1 then set departDate = arrivalDate;

else set departDate=date\_add(arrivalDate,interval 1 day);

end if;

if not ((arrivalDate>'2020-01-22' and arrivalDate<'2020-04-08')

and

(SID1 IN (SELECT SID FROM station where cityname='武汉'))) then

insert into trainpass values(TID1,SNO1,SID1,

concat(arrivalDate ," ", STime1),

concat(departDate ," ", ATime1));

end if;

fetch tt\_cur into SNO1,SID1,STime1,ATime1,D1;

END WHILE;

SET outer\_done=0;

close tt\_cur;

fetch train\_cur into TID1 ,SDate1,TName1;

end while;

close train\_cur;

end$$

drop procedure DetailArrange;

delimiter;

call DetailArrange();

-- 乘客表【乘客身份证号，姓名，性别，年龄】

-- Passenger (PCardID char(18), PName char(20), Sex bit, Age smallint)

-- 其中，主码为乘客身份证号；性别取值为0/1（“1”表示“男”，“0”表示“女”）。

create table Passenger (

PCardID char(18),

PName char(20),

Sex bit,

Age smallint,

CONSTRAINT PK\_passenger primary key(PCardID)

);

-- 键姓名表

-- sno为自增长列，

-- create table nmlist(

-- sno int primary key auto\_increment,

-- pid char(18),

-- nm char(10),

-- sex bit,

-- age smallint

-- );

-- 乘车记录表【记录编号，乘客身份证号，列车流水号，出发站编号，到达站编号，车厢号，席位排号，席位编号，席位状态】

-- TakeTrainRecord (RID int, PCardID char(18), TID int, SStationID int, AStationID int, CarrigeID smallint, SeatRow smallint，SeatNo char(1)，SStatus int)

-- 其中，主码、外码请依据应用背景合理定义。

-- CarrigeID若为空，则表示“无座”；

-- SeatNo只能取值为’A’、’B’、’C’、’E’、’F’，或为空值；

-- SStatus只能取值’0’（退票）、’1’（正常）、’2’（乘客没上车）。

create table TakeTrainRecord (

RID int,

PCardID char(18),

TID int,

SStationID int,

AStationID int,

CarrigeID smallint,

SeatRow smallint,

SeatNo char(1),

SStatus int,

CONSTRAINT PK\_TTR\_RID PRIMARY key(RID),

CONSTRAINT FK\_TTR\_PCardID foreign key (PCardID) REFERENCES passenger(PCardID),

CONSTRAINT FK\_TTR\_SStationID foreign key(SStationID) REFERENCES station(SID),

CONSTRAINT FK\_TTR\_AStationID foreign key(AStationID) REFERENCES station(SID),

CONSTRAINT CK\_TTR\_SeatNo check(SeatNo in('A','B','C','D','E','F')),

CONSTRAINT CK\_TTR\_SStatus check(SStatus in (0,1,2))

);

-- 诊断表【诊断编号，病人身份证号，诊断日期，诊断结果，发病日期】

-- DiagnoseRecord (DID int, PCardID char(18), DDay date, DStatus smallint, FDay date)

-- 其中，主码为DID；DStatus包括：1：新冠确诊；2：新冠疑似；3：排除新冠

create table DiagnoseRecord (

DID int,

PCardID char(18),

DDay date,

DStatus smallint,

FDay date,

CONSTRAINT PK\_DR\_DID PRIMARY key(DID),

CONSTRAINT CK\_DR\_DStatus check(DStatus in(1,2,3))

);

-- 乘客紧密接触者表【接触日期, 被接触者身份证号，状态，病患身份证号】

-- TrainContactor (CDate date, CCardID char(18), DStatus smallint, PCardID char(18))

-- 其中，主码为全码。DStatus包括：1：新冠确诊；2：新冠疑似；3：排除新冠

create table TrainContactor (

CDate date,

CCardID char(18),

DStatus smallint,

PCardID char(18),

CONSTRAINT PK\_TC\_all PRIMARY key (CDate,CCardID,DStatus,PCardID),

CONSTRAINT CK\_TC\_DStatus check(DStatus in(1,2,3))

);

在Navicat中运行对应的语句后，我们讲得到如图3.1所示的表：

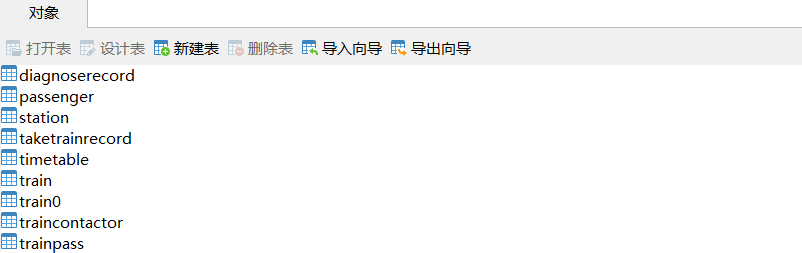


图3.1 数据库建表

其中train0、timeable为过渡表。

2）数据准备

利用Navicat的导入向导功能向各个表中导入老师所提供的数据，以passenger表为例，导入后passenger表的内容如图3.2所示：



图3.2 passenger表内容

其余表中的数据不再赘述。

### 3.2.2 数据更新

1）分别用一条sql语句完成对乘车记录表基本的增、删、改的操作

Sql语句如下所示：

INSERT INTO TakeTrainRecord VALUES(41,'65432319991017375X',1,1599,1621,7,2,'C',2);

UPDATE TakeTrainRecord SET SStatus = 1 where RID = 41;

DELETE FROM TakeTrainRecord WHERE RID = 41;

运行后结果如图3.3所示：



图3.3 对表的增、删、改结果

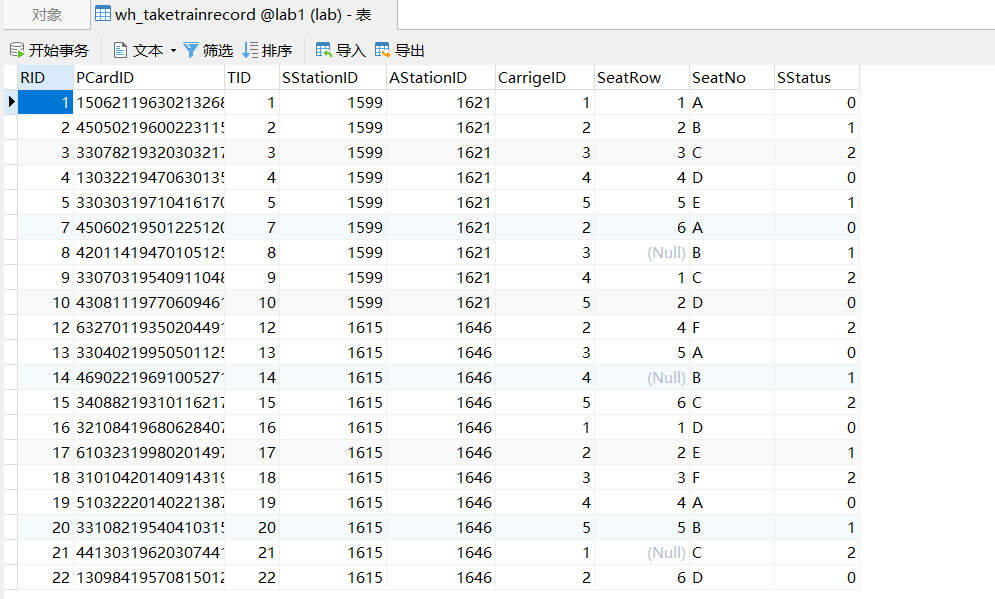
2）批处理操作

将乘车记录表中的从武汉出发的乘客的乘车记录插入到一个新表WH\_TakeTrainRecord中。

执行的sql语句如下所示：

CREATE TABLE WH\_TakeTrainRecord AS (SELECT \* FROM TakeTrainRecord where SStationID IN (SELECT SID FROM station WHERE cityname='武汉'));

执行后的结果如图3.4所示：

图3.4 wh\_taketrainrecord表内容

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。

运行的语句如下所示：

SELECT \* FROM WH\_TakeTrainRecord INTO OUTFILE "D:/MySQL/MySQL Server 8.0/Uploads/wh\_taketrainrecord.txt";

执行后的结果如图3.5和图3.6所示：

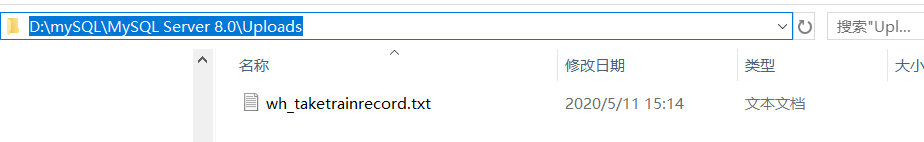


图3.5 数据导入导出结果

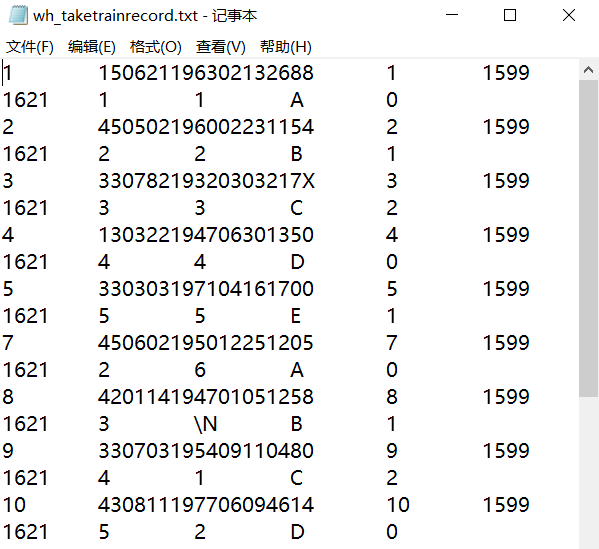


图3.6 数据导入导出结果

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。

执行的sql语句如下所示：

DROP TABLE IF EXISTS test;

CREATE TABLE test(

a int,

b int,

c int

);

INSERT INTO test VALUES (1,2,3);

INSERT INTO test VALUES (1,2,3);

INSERT INTO test VALUES (1,2,3);

#均发生了修改

SELECT \* FROM test;

DROP TABLE test;

运行结果如图3.7所示：



图3.7 对重复元组进行删除和修改结果

5）创建视图

创建一个新冠确诊病人的乘火车记录视图，其中的属性包括：身份证号、姓名、年龄、乘坐列车编号、发车日期、车厢号，席位排号，席位编号。按身份证号升序排序，如果身份证号一致，按发车日期降序排序（注意，如果病人买了票但是没坐车，不计入在内）。

执行的sql语句如下所示：

DROP VIEW IF EXISTS digTakeRecord;

CREATE VIEW digTakeRecord

AS SELECT passenger.PCardID,PName,Age,train.TID,SDate,CarrigeID,SeatRow,SeatNo

FROM passenger,train,taketrainrecord,diagnoserecord

WHERE passenger.PCardID=diagnoserecord.PCardID

AND passenger.PCardID=diagnoserecord.PCardID

AND train.TID=taketrainrecord.TID

AND diagnoserecord.DStatus=1

AND taketrainrecord.SStatus=1

ORDER BY passenger.PCardID,train.SDate DESC;

创建好的视图如图3.8所示：

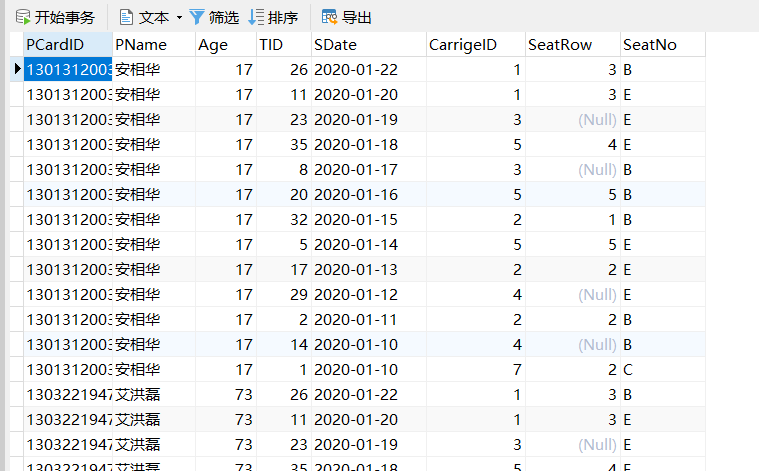


图3.8 视图digTakeRecord内容

6）触发器实验

编写一个触发器，用于实现以下完整性控制规则：

1) 当新增一个确诊患者时，若该患者在发病前14天内有乘车记录，则将其同排及前后排乘客自动加入“乘客紧密接触者表”，其中：接触日期为乘车日期。

2) 当一个紧密接触者被确诊为新冠时，从“乘客紧密接触者表”中修改他的状态为“1”。

执行的语句如下所示：

CREATE

TRIGGER trigger1 AFTER UPDATE ON DiagnoseRecord

FOR EACH ROW

BEGIN

declare Cdate date;

declare Cardid char(18);

declare Dstatus int;

declare Pcardid char(18);

set Dstatus =2;

set Pcardid = new.PCardID;

select TakeTrainRecord.PCardID into Cardid where TakeTrainRecord.TID in (select train.TID from TakeTrainRecord,train where TakeTrainRrcord.TID = train.ITD and

new.Pcardid = TakeTrainRecord.PCardID and

new.Dstatus =1 and

train.SDate>=date\_sub(new.FDay,interval 14 day)and

train.SDate<=new.FDay

);

if Cardid!=null then

insert into TrainContactor values(Cdate,Cardid,Dstatus,Pcardid);

END IF;

update TrainContactor set DStatus = new.DStatus where(TrainContactor.CCardID = new.PCardID);

END;

乘客记录表中插入一位新乘客

插入语句为：

insert into taketrainrecord values(42,110101201105182137,1,1599,1621,7,3,'A',1);

如图3.9所示：

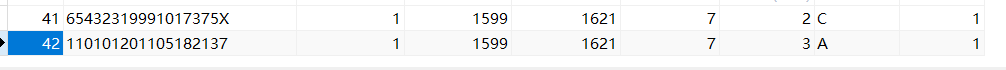


图3.9 新增乘客

让该乘客发病

插入语句为：

insert INTO diagnoserecord VALUES (43,110101201105182137,'2020-01-20',1,'2020-01-09');

原本的traincontactor表中没有数据。如图3.10所示：

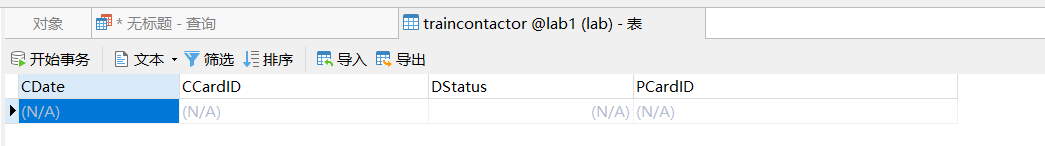


图3.10 traincontactor表中数据

依据上述的插入语句，traincontactor中将插入一条数据。如图3.11所示：

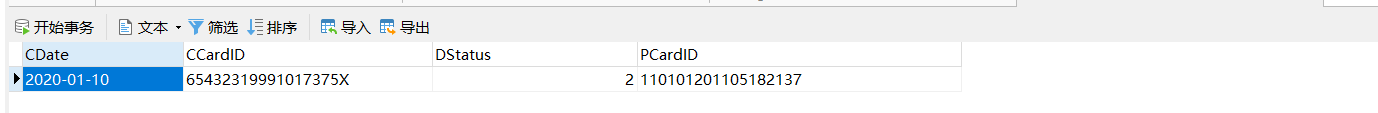


图3.11 traincontactor表中数据

### 3.2.3 查询

1）查询确诊者“张三”的在发病前14天内的乘车记录

查询语句如下所示：

select TakeTrainRecord.\*,passenger.PName from TakeTrainRecord,train,DiagnoseRecord,passenger where

TakeTrainRecord.TID = train.TID and

DiagnoseRecord.PCardID=TakeTrainRecord.PCardID and

passenger.PCardID =DiagnoseRecord.PCardID AND

passenger.PName= '张三' AND

DiagnoseRecord.DStatus = 1 AND

train.SDate <= DiagnoseRecord.FDay AND

train.SDate >= DATE\_SUB(DiagnoseRecord.FDay,INTERVAL 14 DAY);

查询结果如图3.12所示：

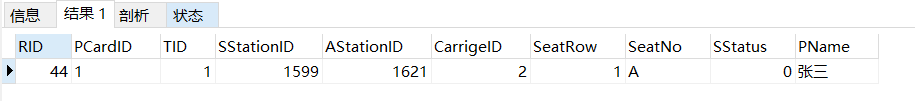


图3.12 查询结果

2）查询所有从城市“武汉”出发的乘客乘列车所到达的城市名

查询的sql语句如下：

select DISTINCT station.cityname from station,TakeTrainRecord where

station.SID = TakeTrainRecord.AStationID and

TakeTrainRecord.SStationID IN

(SELECT station.SID from station where station.cityname='武汉');

查询结果如图3.13所示：

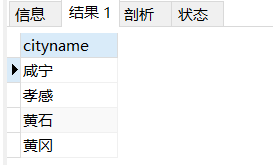


图3.13 查询结果

3）计算每位新冠患者从发病到确诊的时间间隔（天数）及患者身份信息，并将结果按照发病时间天数的降序排列

查询的sql语句如下：

select datediff (DiagnoseRecord.DDay,DiagnoseRecord.FDay)As DayInterval,passenger.\* from passenger,DiagnoseRecord where

passenger.PCardID =DiagnoseRecord.PCardID and

DiagnoseRecord.DStatus=1 ORDER BY DayInterval desc;

查询结果如图3.14所示：

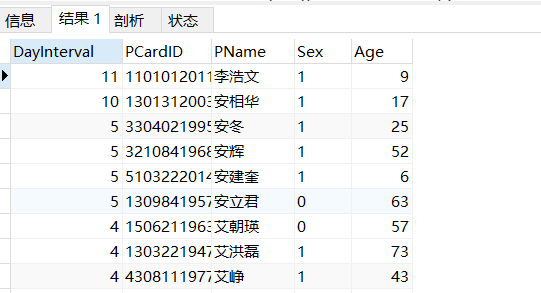


图3.14 查询结果

4）查询“2020-01-22”从“武汉”发出的所有列车

查询的sql语句如下：

select train.\* from train where train.SDate ='2020-01-22' and

train.SStationID in (select distinct SID FROM station where

station.cityname='武汉');

查询结果如图3.15所示：

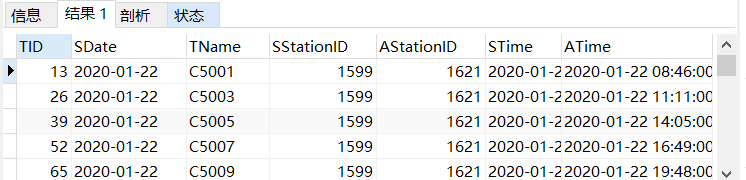


图3.15 查询结果

5）查询“2020-01-22”途经“武汉”的所有列车

查询的sql语句如下：

SELECT train.\*

FROM train,trainpass

WHERE train.TID=trainpass.TID AND train.SDate='2020-01-22'

AND trainpass.SID IN

(SELECT SID FROM station

WHERE CityName='武汉');

查询结果如图3.16所示：

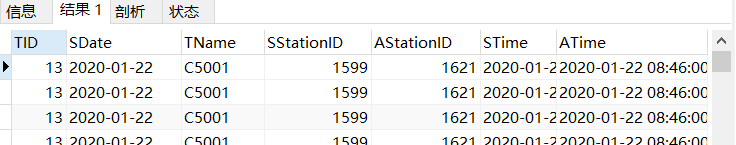


图3.16 查询结果

6）查询“2020-01-22”从武汉离开的所有乘客的身份证号、所到达的城市、到达日期

查询的sql语句如下：

select passenger.PCardID,station.cityname,train.ATime from passenger,TakeTrainRecord,station,train where passenger.PCardID=TakeTrainRecord.PCardID and

station.SID=TakeTrainRecord.AStationID and

train.TID=TakeTrainRecord.AStationID and

TakeTrainRecord.TID in

(select DISTINCT train.TID from train where train.SDate ='2020-01-22'and

train.SStationID in

(select distinct SID FROM station where

station.cityname='武汉'))

GROUP BY(cityname);

查询结果如图3.17所示：

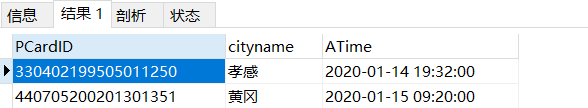


图3.17 查询结果

7）统计“2020-01-22” 从武汉离开的所有乘客所到达的城市及达到各个城市的武汉人员数

查询的sql语句如下：

select station.cityname,count(cityname) as number from passenger,taketrainrecord,station,train where

passenger.PCardID = taketrainrecord.PCardID and

station.SID = taketrainrecord.AStationID and

train.TID = taketrainrecord.TID and

taketrainrecord.TID in

(select DISTINCT train.TID from train where train.SDate ='2020-01-22'and

train.SStationID in

(select distinct SID FROM station where

station.cityname='武汉'))

GROUP BY(cityname);

查询结果如图3.18所示：



图3.18 查询结果

8）查询2020年1月到达武汉的所有人员

查询的sql语句如下：

select distinct passenger.\* from passenger,taketrainrecord,station,train where passenger.PCardID = taketrainrecord.PCardID and

taketrainrecord.AStationID in

(select distinct SID FROM station where

station.cityname='武汉')

and taketrainrecord.TID = train.TID and

train.ATime<'2020-02-01';

查询结果如图3.19所示：

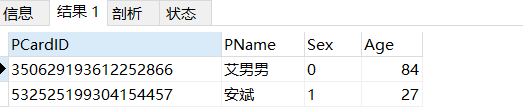


图3.19 查询结果

9）查询2020年1月乘车途径武汉的外地人员（身份证非“420”开头）

查询的sql语句如下：

select distinct passenger.\* from passenger,train,trainpass,taketrainrecord where

passenger.PCardID=taketrainrecord.PCardID and

trainpass.TID=train.TID and

left(passenger.PCardID,3)!='420' and

trainpass.SID in

(select distinct SID FROM station where

station.cityname='武汉') and

taketrainrecord.TID=train.TID and

train.STime>='2020-01-01' and

train.STime<'2020-02-01';

查询结果如图3.20所示：



图3.20 查询结果

10）统计“2020-01-22”乘坐过‘G107’号列车的新冠患者在火车上的密切接触乘客人数（每位新冠患者的同车厢人员都算同车密切接触）

查询的sql语句如下：

select count(\*) as counts from taketrainrecord where

(taketrainrecord.TID,taketrainrecord.CarrigeID)in

(select taketrainrecord.TID,taketrainrecord.CarrigeID from

passenger,taketrainrecord,train,diagnoserecord where

taketrainrecord.TID=train.TID and

train.ATime='2020-01-22' and

train.TName='G007' and

passenger.PCardID=taketrainrecord.PCardID and

diagnoserecord.PCardID=passenger.PCardID and

diagnoserecord.DStatus=1);

查询结果如图3.21所示：



图3.21 查询结果

11）查询一趟列车的一节车厢中有3人及以上乘客被确认患上新冠的列车名、出发日期，车厢号

查询的sql语句如下：

select train.TName,train.SDate,taketrainrecord.CarrigeID from

train,taketrainrecord,diagnoserecord where

train.TID=taketrainrecord.TID and

taketrainrecord.PCardID=diagnoserecord.PCardID and

diagnoserecord.DStatus=1

GROUP BY train.TName,train.SDate,taketrainrecord.CarrigeID having count(\*)>=3;

查询结果如图3.22所示：



图3.22 查询结果

12）查询没有感染任何周边乘客的新冠乘客的身份证号、姓名、乘车日期

查询的sql语句如下：

SELECT DISTINCT passenger.PCardID,PName,CDate

FROM passenger,traincontactor

WHERE passenger.PCardID=traincontactor.PCardID AND CCardID NOT IN

(SELECT DISTINCT CCardID FROM traincontactor WHERE DStatus=1);

查询结果如图3.23所示：

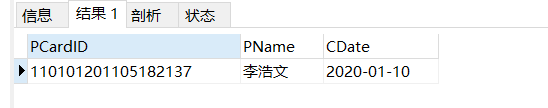


图3.23 查询结果

13）查询到达 “北京”、或“上海”，或“广州”（即终点站）的列车名，要求where子句中除了连接条件只能有一个条件表达式

查询的sql语句如下：

select DISTINCT train.TName from train where

train.AStationID in

(select station.SID from station where

station.cityname='北京' or station.cityname='上海' or

station.cityname='广州');

查询结果如图3.24所示：

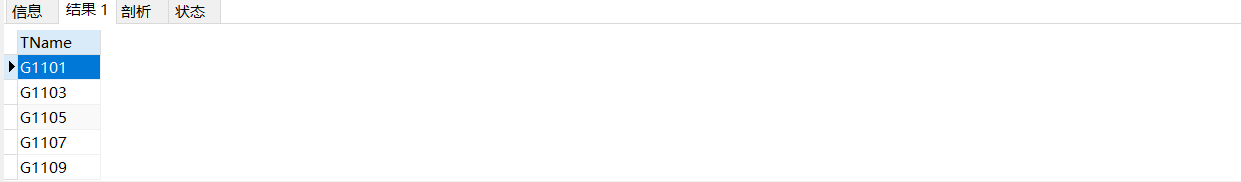


图3.24 查询结果

14）查询“2020-01-22”从“武汉站”出发，然后当天换乘另一趟车的乘客身份证号和首乘车次号，结果按照首乘车次号降序排列，同车次则按照乘客身份证号升序排列

查询的sql语句如下：

select train.TID,passenger.PCardID from train,passenger,taketrainrecord WHERE

passenger.PCardID = taketrainrecord.PCardID and

taketrainrecord.TID = train.TID and

train.TID in

(select train.TID from train where train.SDate='2020-01-22' and train.SStationID in

(select DISTINCT SID from station where

station.sname='武汉'))and

((taketrainrecord.AStationID,passenger.PCardID)in

(select taketrainrecord.SStationID,passenger.PCardID from train,passenger,taketrainrecord where

passenger.PCardID=taketrainrecord.PCardID and

taketrainrecord.TID = train.TID and train.TID in

(select train.TID from train where train.SDate='2020-01-22')))

ORDER BY train.TID desc,PCardID;

15）查询所有新冠患者的身份证号，姓名及其2020年以来所乘坐过的列车名、发车日期，要求即使该患者未乘坐过任何列车也要列出来

查询的sql语句如下：

select temp.PCardID,PName,Tname,SDate from

(select passenger.PCardID,PName from

passenger,diagnoserecord where

passenger.PCardID=diagnoserecord.PCardID and

diagnoserecord.DStatus=1)

temp left join

(select TName,SDate,PCardID from taketrainrecord,train where

taketrainrecord.TID=train.TID and

train.SDate>='2020-01-01')

temp1 on temp.PCardID = temp1.PCardID ORDER BY temp.PCardID,temp1.SDate desc;

查询结果如图3.25所示：

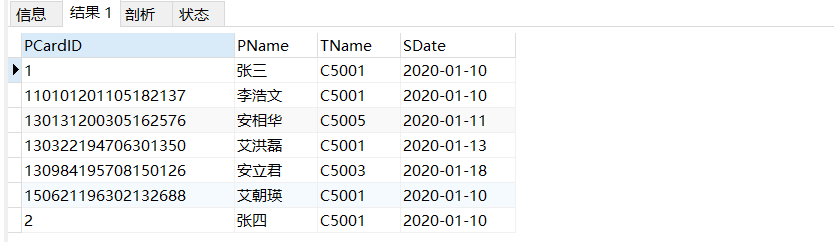


图3.25 查询结果

16）查询所有发病日期相同而且确诊日期相同的病患统计信息，包括：发病日期、确诊日期和患者人数，结果按照发病日期降序排列的前提下再按照确诊日期降序排列

查询的sql语句如下：

select FDay,DDay,count(\*) as total from

diagnoserecord where DStatus =1 group by

FDay,DDay ORDER BY FDay desc,DDay desc;

查询结果如图3.26所示：

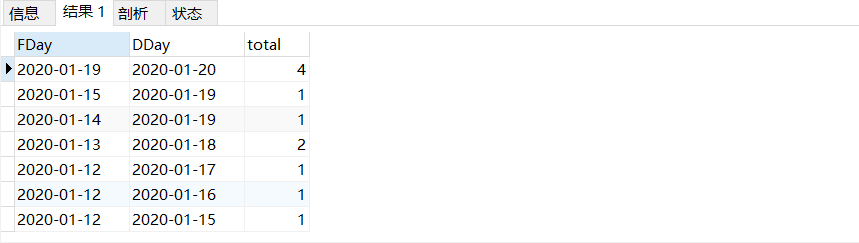


图3.26 查询结果

## 3.3任务总结

本次实验难度较大，任务量也较大。通过本次任务，我加深了对sql语句的运用。一开始只能做前面几个简单的查询语句，触发器也写的问题很大。然后通过查询之前的ppt和百度相关的知识点，一点点扣，最终把全部的任务都完成了。可以说，本次任务没有十分难的地方，都是平时学习到的知识点。但是，在写的时候需要我们小心，不要出错。例如空格、逗号、分号等遗漏错误，这是很不应该的。

# 4 综合实践任务

## 4.1系统设计目标

陈述系统的应用背景和总体目标（300字以内）。

## 4.2 需求分析

展开需求分析，给出需求分析的主要内容的阐述，包括具体的功能需求、性能需求、数据完整性需求、数据流图、数据字典等。

## 4.3 总体设计

阐述系统的总体设计方案，包括系统的C/S或者B/S架构图、功能模块组成及其说明、系统的总体业务流程图及其说明等等总体设计的主要内容。

## 4.4 数据库设计

阐述数据库设计的主要内容，依次是ER图设计及其说明、数据库逻辑结构设计（包括表、视图的逻辑结构说明，含主码、外码说明、用户定义完整性）、数据库物理设计（例如索引）。

## 4.5 详细设计与实现

阐述各主干功能的实现过程，包括主干功能的业务流程图、关键技术和算法说明、数据库事务的定义与实现、数据库函数和触发器的定义与实现等（不允许大段引用源码，如有必要引用必须加详细注释）。

## 4.6 系统测试

包括对测试数据的说明、测试过程阐述、测试结果分析。

## 4.7 系统设计与实现总结

分条目概括、总结应用系统设计的主要工作。

# 4 课程总结

逐条概括、总结此次课程实践的主要工作，阐述此次课程实践的心得体会，展望此次课程实践的有待改进和完善的工作。

# 附录