train_12306实验设计及说明

目录文件列表

1.cgi_bin文件夹

cgi_bin文件夹下一共有20个.c文件和一个.h文件，

其中.h文件用于处理html传递参数时字符串由16进制表示转换为10进制表示并显示的问题，封装为一共urldecode()函数。

.c文件需要生成.cgi文件，最终的.cgi可执行文件，需要放在var/www/cgi-bin目录下，我们以need4.c举例，其编译代码为

```
root@219346cca3a7: gcc -g -Wall -o var/www/cgi-bin/need4.cgi need4.c -lpq
```

一般在docker容器中，我们统一把.c和.h文件放在 home/dbms/Lab2 目录下

2.html目录文件夹

html目录文件夹是前端页面，包含3个html文件

需要放在 var/www/html目录下，注意需要启动apache2服务器

启动命令是:

```
root@219346cca3a7: /etc/init.d/apache2 start
```

通常docker容器的ip地址并不是默认值，需要通过命令

在容器外部打开终端

```
ucas@ucas-cod-2022:~/Desktop/testpy$ sudo docker inspect 219346cca3a7
```

根据"IPAddress":"172.17.0.2"得到，因此在浏览器访问时，必须在 172.17.0.2/目录下

3.sql文件夹

包含7个sql文件，表示在未和前端用户交互时，可以看到的sql语句

4.数据处理文件夹

主要将.csv文件通过python的一些包进行转换，成为.tbl文件后，添加到创建的数据库表项中。

第一步:建表

数据项

1.车次表(train) tr

属性	存储代码	类型	长度
列车车次	tr_id	varchar	20
列车始发站	tr_departure	varchar	20
列车终点站	tr_destination	varchar	20
列车发车时间	tr_start	int	
列车到达时间	tr_arrive	int	

2. 中间经停站(pass) ps

属性	存储代码	类型	长度
列车车次	pass_id	varchar	20
列车站名	pass_station	varchar	20
列车到达时间	pass_arrive	int	
列车出发时间	pass_start	int	
票价(硬座)	pass_hardseat	double	
票价(软座)	pass_softseat	double	
票价(硬卧上)	pass_hardsleeperup	double	
票价(硬卧中)	pass_hardsleepermid	double	
票价(硬卧下)	pass_hardsleeperdown	double	
票价(软卧上)	pass_softsleeperup	double	
票价(软卧下)	pass_softsleeperdown	double	

3.座位(seat) se

属性	存储代码	类型	长度
列车车次	se_id	varchar	20
日期	se_date	int	
所在站点	se_station	varchar	20
硬座个数	se_hardseat	double	
软座个数	se_softseat	double	
硬卧上	se_hardsleeperup	double	
硬卧中	se_hardsleepermid	double	
硬卧下	se_hardsleeperdown	double	
软卧上	se_softsleeperup	double	
软卧下	se_softsleeperdown	double	

4.乘客(passenger)数据项

属性	存储代码	类型	长度
姓名	pa_name	varchar	20
手机号	pa_tele	varchar	11
用户名	pa_user	varchar	30
密码	pa_password	varchar	30

5.订单(order)数据项

属性	存储代码	类型	长度
订单号	ord_id	int	
车次1	ord_trid1	varchar	20
用户	ord_user	varchar	
出发日期	ord_day	integer	
出发时间1	ord_start1	int	
出发车站1	ord_departure1	varchar	20
到达时间1	ord_arrive1	int	

到达车站1	ord_destation1	varchar	
座位类型1	ord_type1	int	
票价1	ord_price1	float	
车次2	ord_trid2	varchar	20
出发时间2	ord_start2	int	
出发车站2	ord_departure2	varchar	20
到达时间2	ord_arrive2	int	
到达车站2	ord_destation2	varchar	
座位类型2	ord_type2	int	
票价2	ord_price2	float	
订票费	ord_bookprice	float	
总价格	ord_sum	float	
订单状态	ord_state	bool	

6.站点(station)

属性	存储代码	类型	长度
车站id	st_id	int	
车站名	st_station	varchar	20
城市名	st_city	varchar	20

数据库建表前操作

```
1.sudo docker ps -a
2.sudo docker start 219346cca3a7
3.sudo docker ps(显示正在运行的容器)
4.sudo docker exec -it 219346cca3a7 /bin/bash
==============接下来进入===========================================================
root@219346cca3a7 : sudo service postgresql start
root@219346cca3a7 : psql
root=# create database train_12306;
root=# \l(显示数据库)
root=# \q(退出，因为建表需要进入所在的数据库)
root@219346cca3a7 : psql -d train_12306(进入新的数据库)
train_12306=# create table station(); (开始建表)
root=# alter user root set client_encoding='utf8';(作用失效)
root@219346cca3a7 : psql -d train_12306 -f 1.sql > 1.txt
root@219346cca3a7 : /etc/init.d/apache2 start
root@219346cca3a7 : gcc -g -Wall -o /var/www/cgi-bin/need5_2.cgi need5_2.c -lpq
```

postgresql 查询操作

```
  (tips) PostgreSQL command:
      \l    show databases
      \c db use database db
      \dt   show tables
      \q    exit
    more command @ man psql
    sudo /etc/init.d/apache2 restart.
```

数据库建表操作

```sql
create table station(
    st_id integer,
    st_station varchar(20),
    st_city varchar(20),
    primary key(st_station)
);
/*数据库外键引用并不表示具体的逻辑关系，只是表明该表项是什么类型*/
create table train(
    tr_id varchar,
    tr_departure varchar(20),
    tr_destination varchar(20),
    tr_start int,
    tr_arrive int,
    primary key(tr_id),
    foreign key(tr_departure) references station(st_station),
    foreign key(tr_destination) references station(st_station)
);

create table pass(
    pass_id varchar(20),
    pass_station varchar(20),
    pass_arrive int DEFAULT NULL,
    pass_start int DEFAULT NULL,
    pass_hardseat float DEFAULT NULL,
    pass_softseat float DEFAULT NULL,
    pass_hardsleeperup float DEFAULT NULL,
    pass_hardsleepermid float DEFAULT NULL,
    pass_hardsleeperdown float DEFAULT NULL,
    pass_softsleeperup float DEFAULT NULL,
    pass_softsleeperdown float DEFAULT NULL,
    primary key(pass_id,pass_station),
    foreign key (pass_id) references train(tr_id),
    foreign key(pass_station) references station(st_station)
);


create table seat(
    se_id varchar(20),
    se_date int,
    se_station varchar(20),
    se_hardseat float DEFAULT NULL,
    se_softseat float DEFAULT NULL,
    se_hardsleeperup float DEFAULT NULL,
    se_hardsleepermid float DEFAULT NULL,
    se_hardsleeperdown float DEFAULT NULL,
    se_softsleeperup float DEFAULT NULL,
    se_softsleeperdown float DEFAULT NULL,
    primary key(se_id,se_date,se_station),
    foreign key(se_id,se_station) references pass(pass_id,pass_station)
);

/*需要更新*/
create table passenger(
    pa_name varchar(20),
    pa_tele varchar(11),
    pa_user varchar(30),
    pa_password varchar(30),
    primary key(pa_user)
);


create table orders(
    ord_user varchar(30),
    ord_id integer,
    ord_day integer,
    ord_trid1 varchar(20),
    ord_start1 integer,
    ord_departure1 varchar(20),
    ord_arrive1 integer,
    ord_destation1 varchar(20),
    ord_type1 varchar(20),
    ord_price1 float,

    ord_trid2 varchar(20),
    ord_start2 integer,
    ord_departure2 varchar(20),
    ord_arrive2 integer,
    ord_destation2 varchar(20),
    ord_type2 varchar(20),
    ord_price2 float,

    ord_bookprice float,
    ord_sum float,
    ord_state bool,
    primary key(ord_id),
    foreign key(ord_user) references passenger(pa_user)
);

create table count(
    count_num integer,
    primary key(count_num)
);
```

后续工作依赖于该表项完成

数据处理部分

1. python脚本处理数据
2. 数据库中，默认不存在的表项，不能够用 pass = 'NULL'来表示，而是用 pass='' 来表示为空
3. 由于编码的问题，采取的方案是，在数据库外面的 windows系统中，将 train-2016-10 的列车信息转换为 .tbl文件

   然后从虚拟机上，登陆邮箱下载.tbl文件到 Downloads文件夹

   接着拷贝 .tbl文件到 docker容器中：拷贝的命令是

   ```
   sudo docker cp seat.tbl/pass.tbl/train.tbl/station.tbl(或者直接写成 *tbl)
   219346cca3a7:/home/dbms/Lab2/train-2016-10
   ```

4.复制.tbl文件，复制命令如下

```
psql -d train_12306
copy pass from '/home/dbms/Lab2/train-2016-10/pass.tbl' with (format csv,
delimiter '|');
```

下面列出python数据处理脚本的具体代码，需要明确，我们一共建立了6个表项，但订单(orders)、乘客(passenger)都是需要根据后续动态输入建立，所以只需要对4个表项完成输入和处理操作。

首先看pass.py

```
import re
import os


def read_folder(folder):
    lst = os.listdir(folder)
    targets = [item for item in lst if item.endswith('.csv')]
    for item in targets:
        f = open(folder+'/'+item,'r')
        fd = open('train.tbl','a')
        f.readline()
        last_start = 0
        count = 0
        while True:
            line = f.readline()
            if not line:
                break

            line = line.replace(' ','')
            line = re.findall('(.*),(.*),(.*),(.*),(.*),(.*),(.*),(.*),(.*),
(.*)',line)
            #sn = int(line[0][0])
            if not line:
                break
            station = line[0][1]

            start = line[0][2]

            if start =='-':
                start = 'NULL'
            else:
                line_start = re.findall('(.*):(.*)',line[0][2])
                start = str(int(line_start[0][0])*60 + int(line_start[0][1]))

                start_int = int(start)
                while start_int < last_start:
                    start_int += 24*60

                start = str(start_int)


            arrive = line[0][3]

            if arrive =='-':
                arrive = 'NULL'
            else:
                line_arrive = re.findall('(.*):(.*)',line[0][3])
                arrive = str(int(line_arrive[0][0])*60 + int(line_arrive[0][1]))
                arrive_int = int(arrive)
                while arrive_int < last_start:
                    arrive_int += 24*60

                arrive = str(arrive_int)


            seat1 = line[0][7]

            if seat1 == '-':
                seat1_hard = 'NULL'
                seat1_soft = 'NULL'
            else:
                line_seat1 = re.findall('(.*)/(.*)',line[0][7])
                if line_seat1[0][0]=='-':
                    seat1_hard = 'NULL'
                else:
                    seat1_hard = line_seat1[0][0]

                if line_seat1[0][1]=='-':
                    seat1_soft = 'NULL'
                else:
                    seat1_soft = line_seat1[0][1]



            seat2 = line[0][8]


            if seat2 == '-':
                seat2_up = 'NULL'
                seat2_mid = 'NULL'
                seat2_down = 'NULL'
            else:
                line_seat2 = re.findall('(.*)/(.*)/(.*)',line[0][8])
                if line_seat2[0][0]=='-':
                    seat2_up = 'NULL'
                else:
                    seat2_up = line_seat2[0][0]

                if line_seat2[0][1]=='-':
                    seat2_mid = 'NULL'
                else:
                    seat2_mid = line_seat2[0][1]

                if line_seat2[0][2]=='-':
                    seat2_down = 'NULL'
                else:
                    seat2_down = line_seat2[0][2]

            seat3 = line[0][9]

            if seat3 == '-':
                seat3_up = 'NULL'
                seat3_down = 'NULL'
            else:
                line_seat3 = re.findall('(.*)/(.*)',line[0][9])
```

```
                if line_seat3[0][0]=='-':
                    seat3_up = 'NULL'
                else:
                    seat3_up = line_seat3[0][0]

                if line_seat3[0][1]=='-':
                    seat3_down = 'NULL'
                else:
                    seat3_down = line_seat3[0][1]
            if count == 0:
                fd.write( item[:-4]+'|' +  station + '|' + start +'|' + arrive +
'|'+ '0' + '|' + '0' + '|' + '0' + '|' + '0' + '|' + '0' + '|' + '0' + '|'
                + '0' + '\n')

            else:
                fd.write( item[:-4]+'|' +  station + '|' + start +'|' + arrive +
'|'+ seat1_hard + '|' + seat1_soft + '|' + seat2_up + '|' + seat2_mid + '|' +
seat2_down + '|' + seat3_up + '|'
                + seat3_down + '\n')
            count += 1

            if start == 'NULL':
                last_start = 0
            else:
                last_start = int(start)


read_folder('0')
read_folder('c')
read_folder('d')
read_folder('g')
read_folder('k')
read_folder('t')
read_folder('y')
read_folder('z')
```

　这是最复杂的一个部分，因为我们需要安装车次、站名、到达时间、出发时间、以及7种类型的票价依次录入。主要的数据处理在于调用re和os包，利用listdir来检索所有的文件夹（注：数据处理文件必须位于train-2016-10目录下）。

targets遍历所有的文件夹，找出其中的.csv文件。

对于每一个.csv文件，我们采用open函数打开，并处理后写入我们新创建的.tbl文件中。

readline读取每一行数据，直到结尾，根据findall进行正则表达式匹配，由于文件中都以逗号分隔，但对于票价的情况，如果某一站不存在硬卧的票价，即不出售硬卧，格式可能为-/-/-，或者-.所以需要分别讨论。

对于列车时间超越一天的情况，我们采取将其表示为整数的形式（x小时y分钟）转化为(x*60 + y),方便后续比较。

接下来是seat.py:

```
import re
import os
```

```python
def read_folder(folder):
    lst = os.listdir(folder)
    targets = [item for item in lst if item.endswith('.csv')]
    for item in targets:
        f = open(folder+'/'+item,'r')
        fd = open('seat.tbl','a')
        f.readline()
        while True:
            line = f.readline()
            if not line:
                break
            line = line.replace(' ','')
            line = re.findall('(.*),(.*),(.*),(.*),(.*),(.*),(.*),(.*),(.*),
(.*)',line)
            #sn = int(line[0][0])
            if not line:
                break
            station = line[0][1]

            seat1 = line[0][7]

            if seat1 == '-':
                seat1_hard = 'NULL'
                seat1_soft = 'NULL'
            else:
                line_seat1 = re.findall('(.*)/(.*)',line[0][7])
                if line_seat1[0][0]=='-':
                    seat1_hard = 'NULL'
                else:
                    seat1_hard = line_seat1[0][0]

                if line_seat1[0][1]=='-':
                    seat1_soft = 'NULL'
                else:
                    seat1_soft = line_seat1[0][1]

            if seat1_hard == 'NULL':
                seat1_hard = '0'
            else:
                seat1_hard = '5'

            if seat1_soft == 'NULL':
                seat1_soft ='0'
            else:
                seat1_soft = '5'


            seat2 = line[0][8]


            if seat2 == '-':
                seat2_up = 'NULL'
                seat2_mid = 'NULL'
                seat2_down = 'NULL'
            else:
                line_seat2 = re.findall('(.*)/(.*)/(.*)',line[0][8])
                if line_seat2[0][0]=='-':
                    seat2_up = 'NULL'
                else:
                    seat2_up = line_seat2[0][0]

                if line_seat2[0][1]=='-':
                    seat2_mid = 'NULL'
                else:
                    seat2_mid = line_seat2[0][1]

                if line_seat2[0][2]=='-':
                    seat2_down = 'NULL'
                else:
                    seat2_down = line_seat2[0][2]

            if seat2_up == 'NULL':
                seat2_up = '0'
            else:
                seat2_up = '5'

            if seat2_mid == 'NULL':
                seat2_mid = '0'
            else:
                seat2_mid = '5'

            if seat2_down == 'NULL':
                seat2_down = '0'
            else:
                seat2_down = '5'


            seat3 = line[0][9]

            if seat3 == '-':
                seat3_up = 'NULL'
                seat3_down = 'NULL'
            else:
                line_seat3 = re.findall('(.*)/(.*)',line[0][9])
                if line_seat3[0][0]=='-':
                    seat3_up = 'NULL'
                else:
                    seat3_up = line_seat3[0][0]

                if line_seat3[0][1]=='-':
                    seat3_down = 'NULL'
                else:
                    seat3_down = line_seat3[0][1]

            if seat3_up == 'NULL':
                seat3_up = '0'
            else:
                seat3_up = '5'

            if seat3_down == 'NULL':
                seat3_down = '0'
            else:
                seat3_down = '5'
            for i in range(0,31):
                fd.write( item[:-4]+'|' + str(i) +  '|'+ station + '|' + '5' +
'|' + '5' + '|' + '5' + '|' + '5' + '|' + '5' + '|' + '5' + '|'
                + '5' + '\n')


read_folder('0')
read_folder('c')
read_folder('d')
read_folder('g')
read_folder('k')
read_folder('t')
read_folder('y')
read_folder('z')
```

train.py

```
import re
import os

def read_folder(folder):
    lst = os.listdir(folder)
    targets = [item for item in lst if item.endswith('.csv')]
    for item in targets:
        f = open(folder + '/' + item,'r')
        fd = open('ttttrain.tbl','a')
        f.readline()

        last_start = 0
        destination = 'NULL'
        arrive = '0'

        line = f.readline()

        line = line.replace(' ','')
        line = re.findall('(.*),(.*),(.*),(.*),(.*),(.*),(.*),(.*),(.*),
(.*)',line)
        station = line[0][1]

        start = line[0][3]

        line_start = re.findall('(.*):(.*)',line[0][3])
        start = str(int(line_start[0][0])*60 + int(line_start[0][1]))

        last_start = int(start)

        while True:
            line = f.readline()
            if not line:
                break

            line = line.replace(' ','')
            line = re.findall('(.*),(.*),(.*),(.*),(.*),(.*),(.*),(.*),(.*),
(.*)',line)
            #sn = int(line[0][0])

            if not line:
                break

            destination = line[0][1]

            arrive = line[0][2]

            line_arrive = re.findall('(.*):(.*)',line[0][2])
            arrive = str(int(line_arrive[0][0])*60 + int(line_arrive[0][1]))

            arrive_int = int(arrive)
            while arrive_int < last_start:
                arrive_int += 24*60

            arrive = str(arrive_int)

        fd.write(item[:-4] + '|' + station + '|' + destination + '|' +
        start + '|' + arrive + '\n')

read_folder('0')
read_folder('c')
read_folder('d')
read_folder('g')
read_folder('k')
read_folder('t')
read_folder('y')
read_folder('z')
```

station.py

```
import re
import os

f = open('all-stations.txt','r')
fd = open('station.tbl','a')
while True:
    line = f.readline()
    if not line:
        break

    line = re.findall('(.*),(.*),(.*)',line)
    if not line:
        break
    fd.write(line[0][0] + '|' + line[0][1] + '|' + line[0][2] + '\n')
```

第二步，写sql语句

需求4

```
/*•网页输入
□车次序号，例如G101
□日期，例如：2022-5-1，默认为查询时间的第二天
•显示该车次所有信息
有静态信息
- 始发站，中间经停站，终点站
- 每站的发车时间和到达时间
- 票价
```

```
也有动态信息：每站余票（从始发站到当前站）
•每站余票上有超链接，点击跳转到需求7网页
预定始发站□被点击的当前站的票*/


select pass_station,pass_arrive,pass_start,
pass_hardseat,pass_softseat,pass_hardsleeperup,
pass_hardsleepermid,pass_hardsleeperdown,
pass_softsleeperup,pass_softsleeperdown,se_date,
se_hardseat,se_softseat,se_hardsleeperup,se_hardsleepermid,
se_hardsleeperdown,se_softsleeperup,se_softsleeperdown

from pass,seat
where pass_id ='1095' and se_date=2 and se_id = pass_id and se_station =
pass_station
order by pass_start;
```

需求4并不复杂，我们需要对pass进行限制，使得其id号为用户给定的id号，并且需求4需要给出座位余票信息，座位余票在seat表中记录。

第一步，实现两个表pass和seat的连接操作，并且根据相等条件，尽可能减少两个表直接笛卡尔积的表项个数。

第二步，条件判断，首先pass必须满足查询列车的条件，这里取列车号为'1095'。接着需要判断是哪一天的列车，由于我们假设30天内的列车全部行程相同，pass表不需要记录天数，seat表根据用户的买票个数，需要记录具体的天数。因此通过seat的se_date作限制。

第三步，连接的条件，是se_id = pass_id ，实现座位和列车站点之间的一一对应，然后打印所有需要的表项。

需求5

需求5是整个需求中最复杂的，并且需求5和需求6的sql语句，仅仅是参数不同，所以这里并不详细枚举需求6。

把需求5分成两个部分来写，第一个是直达列车，第二个是换乘列车。

直达列车

```
/*需求5：查询两地之间的车次
• 网页输入
□ 出发地城市名、到达地城市名
□ 出发日期，默认为查询时间的第二天
□ 出发时间，默认为00:00分 • 显示
□ 表格1：两地之间的直达列车和余票信息
□ 表格2：两地之间换乘一次的列车组合和余票信息
- 换乘地必须是同一城市
- 如果换乘地是同一车站，那么 1小时<= 换乘经停时间 <= 4小时
- 如果换乘地是同城的不同车站，那么 2小时<= 换乘经停时间 <= 4小时
- 显示两个列车的信息和总余票信息（两个列车余票的最小值）
□ 发车时间>=给定的出发时间
• 要求
□ 先显式直达表格，后显示换乘一次表格
□ 每个表格，按照先票价、再行程总时间、最后起始时间排序
□ 每个表格最多显示10行 • 余票上有链接，点击跳转到需求7网页
*/

create view emptyseat as
```

```sql
select p3.pass_id,
min(se_hardseat) as hardseat,
min(se_softseat) as softseat ,
min(se_hardsleeperup) as hardsleeperup,
min(se_hardsleepermid) as hardsleepermid,
min(se_hardsleeperdown) as hardsleeperdown,
min(se_softsleeperup) as softsleeperup,
min(se_softsleeperdown) as softsleeperdown

from pass as p1, pass as p2 ,pass as p3,seat,station as s1,station as s2
where s1.st_city = '泰安'
and s2.st_city = '苏州'
and p1.pass_station = s1.st_station
and p2.pass_station = s2.st_station
and p1.pass_id = p2.pass_id
and p1.pass_start < p2.pass_start
and p3.pass_id = p1.pass_id
and p3.pass_start > p1.pass_start
and p3.pass_arrive <= p2.pass_arrive
and se_id = p1.pass_id
and se_date = 2
group by p3.pass_id;


select p1.pass_id, p1.pass_start,
p1.pass_station,p2.pass_station,
p2.pass_hardseat - p1.pass_hardseat as pass_hardseat ,
p2.pass_softseat - p1.pass_softseat  as pass_softseat,
p2.pass_hardsleeperup - p1.pass_hardsleeperup as pass_hardsleeperup,
p2.pass_hardsleepermid - p1.pass_hardsleepermid as pass_hardsleepermid,
p2.pass_hardsleeperdown - p1.pass_hardsleeperdown as pass_hardsleeperdown,
p2.pass_softsleeperup - p1.pass_softsleeperup as pass_softsleeperup,
p2.pass_softsleeperdown - p1.pass_softsleeperdown as pass_softsleeperdown,
p2.pass_arrive - p1.pass_start as time_span,

(case
    when p2.pass_hardseat - p1.pass_hardseat  is NULL then 0
    when p2.pass_hardseat - p1.pass_hardseat  is not NULL then
emptyseat.hardseat
    end
) as hardseat,

(case
    when p2.pass_softseat - p1.pass_softseat is NULL then 0
    when p2.pass_softseat - p1.pass_softseat is not NULL then emptyseat.softseat
    end
) as softseat,

(case
    when p2.pass_hardsleeperup - p1.pass_hardsleeperup is NULL then 0
    when p2.pass_hardsleeperup - p1.pass_hardsleeperup is not NULL then
emptyseat.hardsleeperup
    end
) as hardsleeperup,

(case
    when p2.pass_hardsleepermid - p1.pass_hardsleepermid is NULL then 0
    when p2.pass_hardsleepermid - p1.pass_hardsleepermid is not NULL then
emptyseat.hardsleepermid
    end
) as hardsleepermid,

(case
    when p2.pass_hardsleeperdown - p1.pass_hardsleeperdown is NULL then 0
    when p2.pass_hardsleeperdown - p1.pass_hardsleeperdown is not NULL then
emptyseat.hardsleeperdown
    end
) as hardsleeperdown,

(case
    when p2.pass_softsleeperup - p1.pass_softsleeperup is NULL then 0
    when p2.pass_softsleeperup - p1.pass_softsleeperup is not NULL then
emptyseat.softsleeperup
    end
)as softsleeperup,

(case
    when p2.pass_softsleeperdown - p1.pass_softsleeperdown is NULL then 0
    when p2.pass_softsleeperdown - p1.pass_softsleeperdown is not NULL then
emptyseat.softsleeperdown
    end
)as softsleeperdown


from pass as p1, pass as p2,seat,emptyseat,station as s1,station as s2
where s1.st_city = '泰安'
and s2.st_city = '苏州'
and p1.pass_station = s1.st_station
and p2.pass_station = s2.st_station
and p1.pass_id = p2.pass_id
and se_date = 2
and se_id = p1.pass_id
and se_station = p1.pass_station
and p1.pass_start < p2.pass_start
and p1.pass_id = emptyseat.pass_id
order by
pass_hardseat,pass_softseat,pass_hardsleeperup,pass_hardsleepermid,pass_hardslee
perdown,
pass_softsleeperup,pass_softsleeperdown,time_span,p1.pass_start;

drop view emptyseat;
```

第一步，建立了一个名为emptyseat的新视图

emptyseat的作用在于，首先给定始发车站和终点站，我们想要买车票的限制条件是：在所有从始发站到终点站经过的站台中，列车剩余座位的最小值大于1.

emptyseat的作用是：首先从 pass as p1,pass as p2,pass as p3这3个pass表中，获取所有经停车站，并且限制始发站的发车时间必须小于终点站的到达时间，并且满足id相等，日期根据seat表得到，从而确定每一个站点，在给定id，给定日期时的余票情况。

第二步，确定各站的票价

需要由终点站票价减去始发站票价得到，但如果结果为空，则利用case when...语句赋0值。并打印相关信息

第三步，关闭视图

否则后续查询会显示视图已创建，影响查询速度。

换乘列车

```
create view handle as

select p1.pass_id as first_id,
p1.pass_start as first_start,
p1.pass_station as start_station,
p3.pass_station as transfer_station1,
p3.pass_arrive as first_arrive,
p4.pass_station as transfer_station2,
p4.pass_start as second_start,
p2.pass_id as second_id,
p2.pass_station as arrive_station,
p2.pass_arrive as second_arrive,

p3.pass_hardseat - p1.pass_hardseat as pass1_hardseat ,
p3.pass_softseat - p1.pass_softseat  as pass1_softseat,
p3.pass_hardsleeperup - p1.pass_hardsleeperup as pass1_hardsleeperup,
p3.pass_hardsleepermid - p1.pass_hardsleepermid as pass1_hardsleepermid,
p3.pass_hardsleeperdown - p1.pass_hardsleeperdown as pass1_hardsleeperdown,
p3.pass_softsleeperup - p1.pass_softsleeperup as pass1_softsleeperup,
p3.pass_softsleeperdown - p1.pass_softsleeperdown as pass1_softsleeperdown,


p2.pass_hardseat - p4.pass_hardseat as pass2_hardseat,
p2.pass_softseat - p4.pass_softseat  as pass2_softseat,
p2.pass_hardsleeperup - p4.pass_hardsleeperup as pass2_hardsleeperup,
p2.pass_hardsleepermid - p4.pass_hardsleepermid as pass2_hardsleepermid,
p2.pass_hardsleeperdown - p4.pass_hardsleeperdown as pass2_hardsleeperdown,
p2.pass_softsleeperup - p4.pass_softsleeperup as pass2_softsleeperup,
p2.pass_softsleeperdown - p4.pass_softsleeperdown as pass2_softsleeperdown,


p2.pass_arrive - p1.pass_start  as time_span



from pass as p1, pass as p2, pass as p3, pass as p4,
station as s1, station as s2,station as s3, station as s4
where s1.st_city = '南京'
and s2.st_city = '苏州'
and p1.pass_start >= 0
and p1.pass_station = s1.st_station
and p2.pass_station = s2.st_station
and p1.pass_id != p2.pass_id
and p3.pass_id = p1.pass_id
and p4.pass_id = p2.pass_id
and s3.st_station = p3.pass_station
and s4.st_city = s3.st_city
and p4.pass_station = s4.st_station

and p4.pass_station != p3.pass_station
and p1.pass_start < p2.pass_arrive
and p3.pass_station != p1.pass_station
and p3.pass_station != p2.pass_station
and p4.pass_station != p1.pass_station
and p4.pass_station != p2.pass_station


and p1.pass_start < p3.pass_arrive
and p4.pass_start < p2.pass_arrive

and p4.pass_start - p3.pass_arrive >= 120
and p4.pass_start - p3.pass_arrive <= 240

union all


select p1.pass_id as first_id,
p1.pass_start as first_start,
p1.pass_station as start_station,
p3.pass_station as transfer_station1,
p3.pass_arrive as first_arrive,
p4.pass_station as transfer_station2,
p4.pass_start as second_start,
p2.pass_id as second_id,
p2.pass_station as arrive_station,
p2.pass_arrive as second_arrive,

p3.pass_hardseat - p1.pass_hardseat as pass1_hardseat ,
p3.pass_softseat - p1.pass_softseat  as pass1_softseat,
p3.pass_hardsleeperup - p1.pass_hardsleeperup as pass1_hardsleeperup,
p3.pass_hardsleepermid - p1.pass_hardsleepermid as pass1_hardsleepermid,
p3.pass_hardsleeperdown - p1.pass_hardsleeperdown as pass1_hardsleeperdown,
p3.pass_softsleeperup - p1.pass_softsleeperup as pass1_softsleeperup,
p3.pass_softsleeperdown - p1.pass_softsleeperdown as pass1_softsleeperdown,


p2.pass_hardseat - p4.pass_hardseat as pass2_hardseat,
p2.pass_softseat - p4.pass_softseat  as pass2_softseat,
p2.pass_hardsleeperup - p4.pass_hardsleeperup as pass2_hardsleeperup,
p2.pass_hardsleepermid - p4.pass_hardsleepermid as pass2_hardsleepermid,
p2.pass_hardsleeperdown - p4.pass_hardsleeperdown as pass2_hardsleeperdown,
p2.pass_softsleeperup - p4.pass_softsleeperup as pass2_softsleeperup,
p2.pass_softsleeperdown - p4.pass_softsleeperdown as pass2_softsleeperdown,


p2.pass_arrive - p1.pass_start  as time_span


from pass as p1, pass as p2, pass as p3, pass as p4,
station as s1, station as s2
where s1.st_city = '南京'
and s2.st_city = '苏州'
and p1.pass_start >= 0
and p1.pass_station = s1.st_station
and p2.pass_station = s2.st_station
and p1.pass_id != p2.pass_id
and p3.pass_id = p1.pass_id
and p4.pass_id = p2.pass_id
```

```sql
and p4.pass_station = p3.pass_station
and p1.pass_start < p2.pass_arrive
and p3.pass_station != p1.pass_station
and p3.pass_station != p2.pass_station

and p1.pass_start < p3.pass_arrive
and p4.pass_start < p2.pass_arrive


and p4.pass_start - p3.pass_arrive >= 60
and p4.pass_start - p3.pass_arrive <= 240;


create view pass_union as

select handle.first_id as pass_union_first_id,
handle.start_station as pass_union_start_station,
handle.transfer_station1 as pass_union_transfer_station1,
handle.second_id as pass_union_second_id,
handle.transfer_station2 as pass_union_transfer_station2,
handle.arrive_station as pass_union_arrive_station,

seat1.se_hardseat as hardseat,
seat1.se_softseat as softseat,
seat1.se_hardsleeperup as hardsleeperup,
seat1.se_hardsleepermid as hardsleepermid,
seat1.se_hardsleeperdown as hardsleeperdown,
seat1.se_softsleeperup as softsleeperup,
seat1.se_softsleeperdown as softsleeperdown

from handle,pass as p5, seat as seat1

where p5.pass_id = handle.first_id
and p5.pass_start >= handle.first_start
and p5.pass_arrive <= handle.first_arrive
and seat1.se_id = p5.pass_id
and seat1.se_station = p5.pass_station
and seat1.se_date = 2

union all

select handle.first_id as pass_union_first_id,
handle.start_station as pass_union_start_station,
handle.transfer_station1 as pass_union_transfer_station1,
handle.second_id as pass_union_second_id,
handle.transfer_station2 as pass_union_transfer_station2,
handle.arrive_station as pass_union_arrive_station,

seat2.se_hardseat as hardseat,
seat2.se_softseat as softseat,
seat2.se_hardsleeperup as hardsleeperup,
seat2.se_hardsleepermid as hardsleepermid,
seat2.se_hardsleeperdown as hardsleeperdown,
seat2.se_softsleeperup as softsleeperup,
seat2.se_softsleeperdown as softsleeperdown

from handle,pass as p6, seat as seat2
```

```sql
where p6.pass_id = handle.second_id
and p6.pass_start >= handle.second_start
and p6.pass_arrive <= handle.second_arrive
and seat2.se_id = p6.pass_id
and seat2.se_station = p6.pass_station
and seat2.se_date = 2;


create view empty_union as

select u.pass_union_first_id as empty_union_first_id,
u.pass_union_start_station as empty_union_start_station,
u.pass_union_transfer_station1 as empty_union_transfer_station1,
u.pass_union_second_id as empty_union_second_id,
u.pass_union_transfer_station2 as empty_union_transfer_station2,
u.pass_union_arrive_station as empty_union_arrive_station,

min(u.hardseat) as hardseat,
min(u.softseat) as softseat ,
min(u.hardsleeperup) as hardsleeperup,
min(u.hardsleepermid) as hardsleepermid,
min(u.hardsleeperdown) as hardsleeperdown,
min(u.softsleeperup) as softsleeperup,
min(u.softsleeperdown) as softsleeperdown

from pass_union as u
group by u.pass_union_first_id,
u.pass_union_start_station,
u.pass_union_transfer_station1,
u.pass_union_second_id,
u.pass_union_transfer_station2,
u.pass_union_arrive_station;



select h.first_id as first_id,
h.first_start as first_start,
h.start_station as start_station,
h.transfer_station1 as transfer_station1,
h.first_arrive as first_arrive,
h.second_id as second_id,

h.second_start as second_start,
h.transfer_station2 as transfer_station2,

h.arrive_station as arrive_station,
h.second_arrive as second_arrive,
h.time_span as time_span,

least(h.pass1_hardseat,
h.pass1_softseat,
h.pass1_hardsleeperup,
h.pass1_hardsleepermid,
h.pass1_hardsleeperdown,
h.pass1_softsleeperup,
h.pass1_softsleeperdown) +

least(h.pass2_hardseat,
h.pass2_softseat,
h.pass2_hardsleeperup,
h.pass2_hardsleepermid,
h.pass2_hardsleeperdown,
h.pass2_softsleeperup,
h.pass2_softsleeperdown) as pass2_least,

least(u.hardseat,
u.softseat,
u.hardsleeperup,
u.hardsleepermid,
u.hardsleeperdown,
u.softsleeperup,
u.softsleeperdown) as seat_least


from handle as h ,empty_union as u


where  u.empty_union_first_id = h.first_id
and u.empty_union_start_station = h.start_station
and u.empty_union_transfer_station1 = h.transfer_station1
and u.empty_union_second_id = h.second_id
and u.empty_union_transfer_station2= h.transfer_station2
and u.empty_union_arrive_station = h.arrive_station

order by pass2_least,
time_span,
first_start;




drop view empty_union;
drop view pass_union;
drop view handle;
```

第一步，建立handle视图

首先有p3和p4两个pass表，作为起点p1 pass表和终点p2 pass表的中转车站，并打印14列票价。

需要满足列车的不同情况，采用union all 语句

分别处理中转站位于同一车站和同一城市不同车站的情况。

第二步，建立 pass_union视图

再创建两个表pass p5和pass p6，分别作为两辆列车的所有中间车站，并取出每一站的座位情况

第三步，建立empty_union视图

对上面两个列车中所有中间站的剩余座位数，取最小值。

这里我们只打印第一列车和第二列车中，7类票价中的最小值（非空），并且空余票价按第一列与第二列列车中的剩余空位最小者作为判断条件。

第四步，关闭视图。

需求7

```
/*• 显示
 每个车次显示（换乘需要同时显示2个车次）
- 车次
- 出发日期、出发时间、出发车站
- 到达日期、到达时间、到达车站
- 座位类型、本次车票价
 订票费：5元*车次数
 总票价
 注：通常有1组信息；对于换乘一次，有2组车次信息
• 用户点击确认，就生成订单
 记录到用户的历史订单中，修改车次对应的座位信息
 订单包含：订单号、上述车次、出发、到达、座位类型、票价、日期
和时间
• 用户点击取消，返回登录首页
*/
/*
插入完整记录
insert into 表名 values(值以逗号隔开，按create table 顺序 );
insert into Student
values (131234,'张飞',1995/1/1,M,'计算机',2013,85);

插入记录特定的列，其它列为空NULL 或默认值
insert into 表名(列名1，列名2,...) */

select
ord_id,
ord_day,
ord_trid1,
ord_start1,
ord_departure1,
ord_arrive1,
ord_destation1,
ord_type1,
ord_price1,

ord_trid2,
ord_start2,
ord_departure2,
ord_arrive2,
ord_destation2,
ord_type2,
ord_price2,
ord_sum,
ord_state

from orders
where  ord_user = '票'
order by ord_id;

update count
set count_num = count_num + 1;

insert into order
values (
    '',
```

```sql
select * from count,
    20,
    '',
    0,
    '',
    0,
    '',
    '',
    0.0,
    '',
    0,
    '',
    0,
    '',
    '',
    0.0,
    0.0,
    0.0,
    1
);

update seat
set se_hardseat = se_hardseat - 1
where se_id = '1'
and se_date = '2'
and se_station in (
    select pass_station
    from pass
    where pass_id = ''
    and pass_start > 0
    and pass_arrive <= 100
);


update seat
set se_softseat = se_softseat - 1
where se_id = '1'
and se_date = '2'
and se_station in (
    select pass_station
    from pass
    where pass_id = ''
    and pass_start > 0
    and pass_arrive <= 100
);

update seat
set se_hardsleeperup = se_hardsleeperup - 1
where se_id = '1'
and se_date = '2'
and se_station in (
    select pass_station
    from pass
    where pass_id = ''
    and pass_start > 0
    and pass_arrive <= 100
);
```

```sql
update seat
set se_hardsleepermid = se_hardsleepermid - 1
where se_id = '1'
and se_date = '2'
and se_station in (
    select pass_station
    from pass
    where pass_id = ''
    and pass_start > 0
    and pass_arrive <= 100
);

update seat
set se_hardsleeperdown = se_hardsleeperdown - 1
where se_id = '1'
and se_date = '2'
and se_station in (
    select pass_station
    from pass
    where pass_id = ''
    and pass_start > 0
    and pass_arrive <= 100
);

update seat
set se_softsleeperup = se_softsleeperup - 1
where se_id = '1'
and se_date = '2'
and se_station in (
    select pass_station
    from pass
    where pass_id = ''
    and pass_start > 0
    and pass_arrive <= 100
);

update seat
set se_softsleeperdown = se_softsleeperdown - 1
where se_id = '1'
and se_date = '2'
and se_station in (
    select pass_station
    from pass
    where pass_id = ''
    and pass_start > 0
    and pass_arrive <= 100
);

delete from order
where ord_id = 1;


update seat
set se_hardseat = se_hardseat + 1
where se_id = '1'
and se_date = '2'
and se_station in (
    select pass_station
    from pass
    where pass_id = ''
    and pass_start > 0
    and pass_arrive <= 100
);


update seat
set se_softseat = se_softseat + 1
where se_id = '1'
and se_date = '2'
and se_station in (
    select pass_station
    from pass
    where pass_id = ''
    and pass_start > 0
    and pass_arrive <= 100
);

update seat
set se_hardsleeperup = se_hardsleeperup + 1
where se_id = '1'
and se_date = '2'
and se_station in (
    select pass_station
    from pass
    where pass_id = ''
    and pass_start > 0
    and pass_arrive <= 100
);

update seat
set se_hardsleepermid = se_hardsleepermid + 1
where se_id = '1'
and se_date = '2'
and se_station in (
    select pass_station
    from pass
    where pass_id = ''
    and pass_start > 0
    and pass_arrive <= 100
);

update seat
set se_hardsleeperdown = se_hardsleeperdown + 1
where se_id = '1'
and se_date = '2'
and se_station in (
    select pass_station
    from pass
    where pass_id = ''
    and pass_start > 0
    and pass_arrive <= 100
);

update seat
set se_softsleeperup = se_softsleeperup + 1
where se_id = '1'
```

```
and se_date = '2'
and se_station in (
    select pass_station
    from pass
    where pass_id = ''
    and pass_start > 0
    and pass_arrive <= 100
);

update seat
set se_softsleeperdown = se_softsleeperdown + 1
where se_id = '1'
and se_date = '2'
and se_station in (
    select pass_station
    from pass
    where pass_id = ''
    and pass_start > 0
    and pass_arrive <= 100
);
```

根据c语言传递的参数进行座位增减，定义count表生成随机数。

需求8

```
/*
乘客可以查询历史订单
□给定出发日期范围，显示订单列表
□订单信息：订单号，日期、出发到达站、总票价、订单状
态（正常/取消）
□提供链接，点击显示订单具体信息，当订单包含2个车次
时，显示每个车次的信息
□提供链接，点击取消订单
- 取消的订单，在订单列表中仍将显示，但注明取消
*/

select ord_id,
ord_day,
ord_trid1,
ord_departure1,
ord_destation1,

ord_trid2,
ord_departure2,
ord_destation2,

ord_sum,
ord_state

from orders

where ord_day >= 2
and ord_day <= 3
and ord_user = '票';

update orders
set ord_state = 0
```

```
where ord_id = 123;
```

按筛选条件，对order表进行select

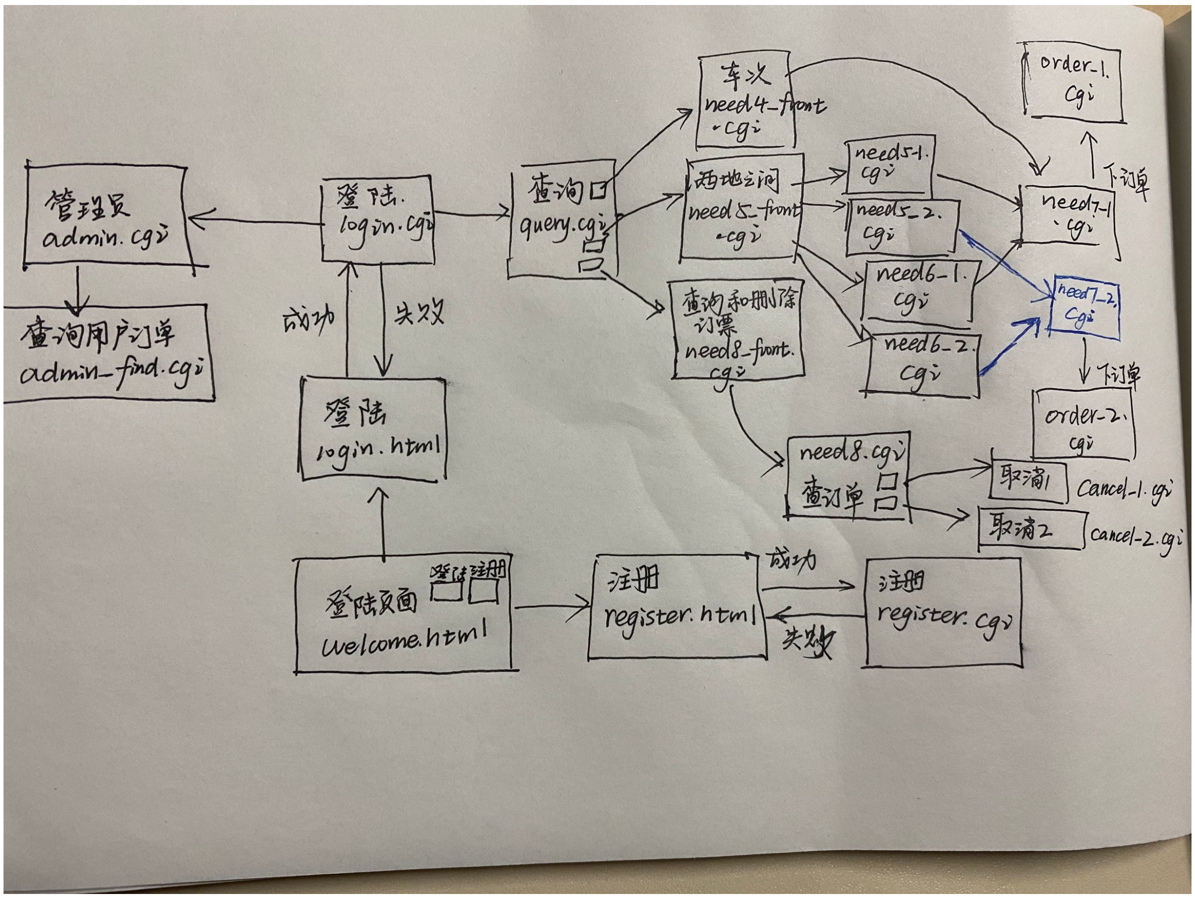
需求9

```
/*Admin登录后显示不同的登录首页
• Admin可以看到下述信息
□总订单数（不包括已经取消订单）
□总票价（不包括已经取消订单）
□最热点车次排序，排名前10的车次（不包括已经取消订单）
□当前注册用户列表
□查看每个用户的订单
*/

create  view as order_trainid
select ord_trid1 as order_trainid_id

from orders

where ord_state = 1

union

select ord_trid2 as order_trainid_id
from orders
where ord_state = 1

select order_trainid_id
from order_trainid
group by order_trainid_id  desc limit 10

drop view order_trainid;

/*创建最热点车次排序*/

select count(ord_id),
sum(ord_sum)
from orders
where ord_state = 1;

select *
from orders
where ord_user = 'defalut name';

select pa_user
from passenger;
```

第三步，完善前端信息

这里采用c语言中封装的PGconnect函数，在适当位置处建立超链接的部分，需要自己打印。仿照PQprint的打印格式，定位需要建立超链接的部分。

具体文件名称和顺序对应如下图所示：



前端部分处理的困难，主要在于确定哪些参数需要一直被传递，例如user的信息，从登陆之后，需要一直跟随查询、需求4、5、6、7等等。