数据库设计报告

孔静 谈清扬 方言歌

2016/11/23

一、 需求分析

仿照 12306 设计和实现一个火车票订票系统，具有面向顾客的查询车次、订票、修改订单等基本功能，面向管理员的分析订单信息的功能。

查询

能够依据指定车次查询，指定出发地到达地查询等功能。给用户提供座位类型，是否转车，是否需要返程信息等功能。

预订

能够根据查询到的车次信息进行火车票预订。也可提供直接输入车次、日期、出发地、目的地预订。

分析订单

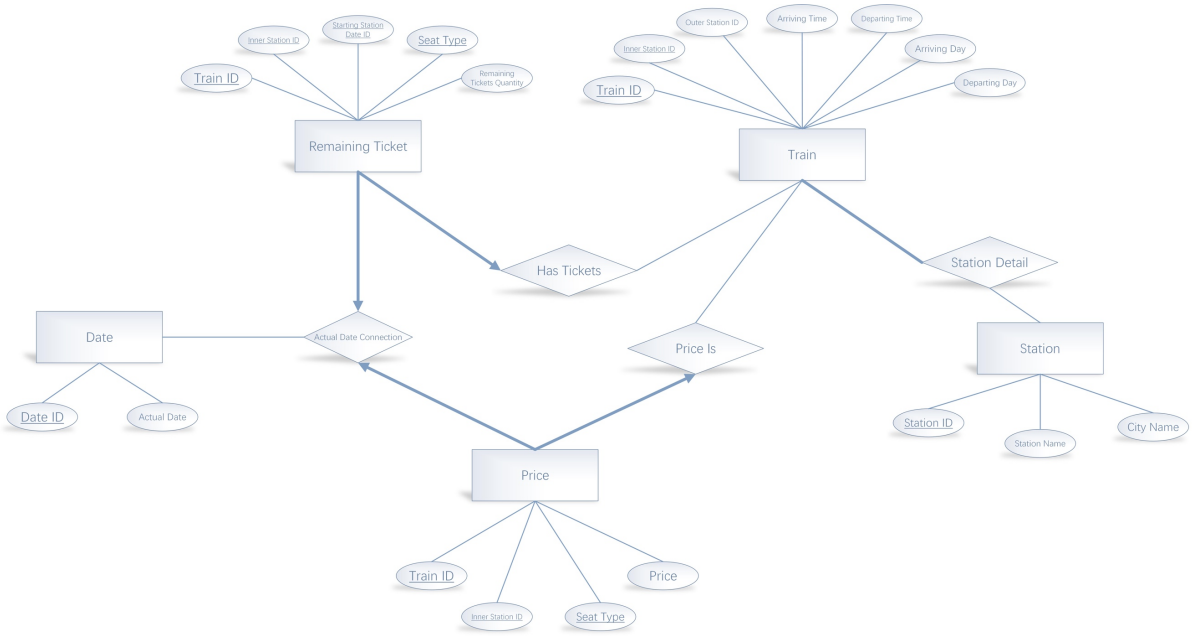
给管理员权限，能够分析最火订单等信息。

二、 ER 图设计

依据需求，设计关于火车信息的相关内容，乘客和订单相关的相关内容，管理员的相关内容。

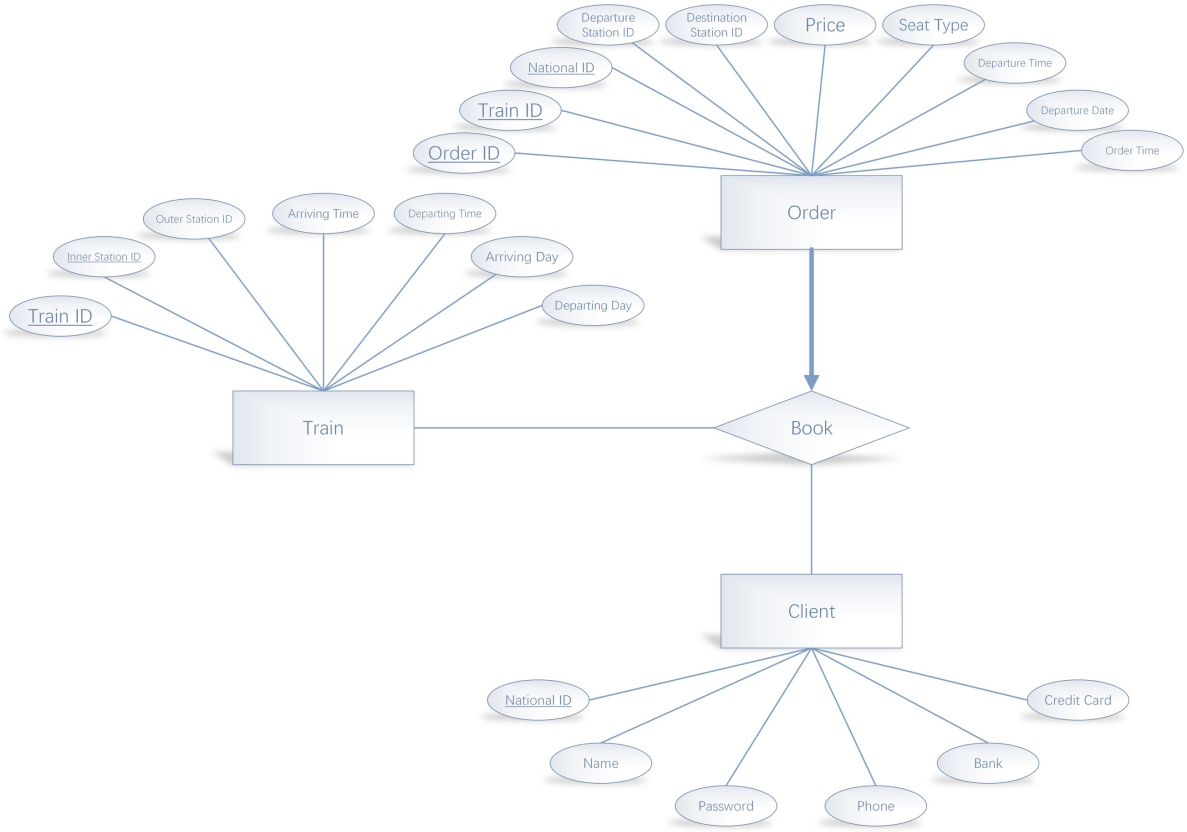
1. **火车信息相关内容**

包括实体集火车（时刻），价格，余票，日期信息等，包括联系余票关系，票价关系等。



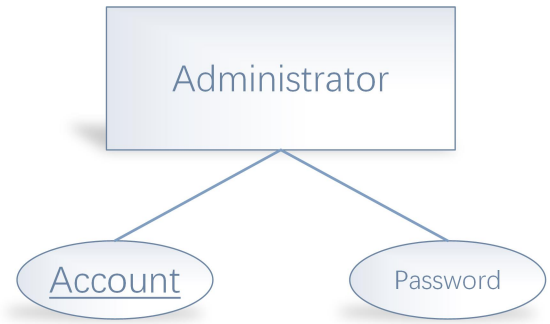
2. 乘客和订单相关内容

包括实体集乘客，订单等，包括联系预订等。



3. 管理员相关内容

包括实体集管理员，用于存储管理员的后台信息。



三、 关系模式

建立以下数据库表：

1. **Train（实体集 Train）**

（为方便时间计算，同时将出发时间、到达时间存储为分钟数，再存储一次）

```
create table Train
(
  Train_ID               varchar(30) not NULL,
  Inner_Station_ID       integer not NULL,
  Arriving_Time          time,
  Departing_Time         time,
  Arriving_Time_Number    integer,
  Departing_Time_Number   integer,
  Outer_Station_ID       integer,
  Arriving_Day              integer,
  Departing_Day          integer,
  primary key(Train_ID,Inner_Station_ID),
  foreign key(Outer_Station_ID) references Station(Station_ID)
);
```

2. **Remaining Ticket（实体集 Remaining Ticket）**

```
create table Remaining_Ticket
(
  Train_ID           varchar(30) not NULL,
  Inner_Station_ID    integer not NULL,
  Seat_Type          varchar(20),
  Starting_Station_Date_ID                integer,
```

```
  Remaining_Tickets_Quantity            integer,
  primary key(Train_ID,Inner_Station_ID,Seat_Type, Starting_Station_Date_ID),
  foreign key(Starting_Station_Date_ID) references The_Date(Date_ID),
  foreign key(Train_ID) references Train(Train_ID)
);
```

3. Price（实体集 Price）

```
create table Price
(
  Train_ID            varchar(30) not NULL,
  Inner_Station_ID     integer not NULL,
  Seat_Type           varchar(20),
  Price              decimal(5,1),
  primary key(Train_ID,Inner_Station_ID,Seat_Type)
);
```

4. Date（实体集 Date）

```
create table Date
(
  Date_ID             integer not NULL,
  Actual_Date         Date,
  primary key(Date_ID)
);
```

5. Client（实体集 Client）

```
create table Client
(
   National_ID char(18),
   Password   varchar(6),
   Name       varchar(6),
   Phone      char(10),
   Bank       varchar(10),
   Credit_Card char(16),
   primary key(National_ID)
);
```

6. Order（实体集 Order）

```
create table Order_Table
(
  Order_ID               varchar(100),
  Train_ID               varchar(30),
  National_ID            varchar(18),
  Departure_Station_ID    integer,
  Destination_Station_ID  integer,
  Price                  decimal(8,1),
  Seat_Type              varchar(20),
  Departure_Time         varchar(10),
  Departure_Date         date,
  Order_Time             timestamp,
  primary key(Order_ID,National_ID,Train_ID),
  foreign key(Train_ID) references Train(Train_ID),
  foreign key(Departure_Station_ID) references Station(Station_ID),
  foreign key(Destination_Station_ID) references Station(Station_ID),
  foreign key(National_ID) references Client(National_ID)
);
```

四、 范式细化与分析

1. 2NF

所有表都已经达到 2NF：

Train

只有同时确定列车车次即 Train_ID 和车次内的站点序号即 Inner_Station_ID 才能确定包括价格发车时间等的相关信息。

Remaining Ticket

只有同时确定列车车次，车次内站点序号，座位类型和日期才能确定余票数。

Price

没有主键中单独的某个键能够确定票价。

Date

主键中只有一个键，显然。

Client

主键中只有一个键，显然。

Order

由于生成的订单编号为时间，所以需要客户的信息联系确定，且一张订单可能涉及多个车次，所有主键中单独的元素均无法确认其他内容。

2. 3NF

1) 达到

Remaining_Ticket,Price,Client,Date。

2) 没有达到

Train：

由于为了计算方便，将出发到达时间转换为分钟数。如果为了消除此依赖，可以后续利用 postgresql 中的时间转化函数进行计算。

Order：

由于为了展示订单方便，存储了可以从 Train，Price 中等确定的时间和价格等。如果为了消除此依赖，可以在查询时多使用连接手段再次查询。

3. BCNF

达到 3NF 的几个表均能达到 BCNF。

五、 SQL 功能实现

1. 查票

以查询转乘一次为例（查询直达去掉关于转乘时间要求的限制，查询转乘两次扩展转乘要求到三次列车，同查询返程时，去程加上到达时间限制，返程则无需）：

3) 首先查询可能的车次和站编号信息

其中$1 为出发城市名，$2 为到达城市名，$3 为出发日期 id，$4 为返程日期 id

```
select d1.train_id as train_id_1, d3.train_id as train_id_2 , s1.station_name as start_name_1, d1.departing_time as start_time_1, d1.inner_station_id as start_id_1, s2.station_name as end_name_1, d2.arriving_time as end_time_1, d2.inner_station_id as end_id_1, s3.station_name as start_name_2, d3.departing_time as start_time_2,
```

```
d3.inner_station_id as start_id_2, s4.station_name as end_name_2, d4.arriving_time as end_time_2, d4.inner_station_id as end_id_2, t1.date_id as train_departing_day_1,$3 as departing_day_1, t1.date_id + d2.arriving_day as arriving_day_1, t2.date_id as train_departing_day_2, t2.date_id + d3.departing_day as departing_day_2, t2.date_id + d4.arriving_day as arriving_day_2, ((D3.departing_time_number+1440-D2.arriving_time_number)%1440 + (D2.arriving_day-D1.departing_day)*1440 + D2.arriving_time_number - D1.departing_time_number + (D4.arriving_day-D3.departing_day)*1440 + D4.arriving_time_number - D3.departing_time_number) as travel_time
into temp21
from train as d1, train as d2,train as d3,train as d4, station as s1, station as s2,station as s3, station as s4, date as t1, date as t2
where d1.train_id = d2.train_id and d1.inner_station_id < d2.inner_station_id
and d1.train_id not in(
    select train_id
    from temp13
)
and d3.train_id = d4.train_id and d3.inner_station_id < d4.inner_station_id--满足车站在列车行进中的先后顺序
and d3.train_id not in(
    select train_id
    from temp13
)--满足直达列车不再加入搜索（temp13 为直达列车搜索的中间结果）
and d1.outer_station_id = s1.station_id and s1.city_name = $1
and d2.outer_station_id = s2.station_id and s2.city_name = s3.city_name
and d3.outer_station_id = s3.station_id--满足在同一城市换乘
and d4.outer_station_id = s4.station_id and s4.city_name = $2
and (d3.departing_time_number + 1440 - d2.arriving_time_number)%1440 < 240--满足转乘时间不超过 4 小时的要求
and (((d3.departing_time_number+1440-d2.arriving_time_number)%1440>60 and s2.station_id = s3.station_id) or ((d3.departing_time_number+1440-d2.arriving_time_number)%1440>120 and s2.station_id != s3.station_id))--满足是否同一车站时不同的最小换成时间要求
and ((d3.departing_time_number>d2.arriving_time_number and t2.date_id = t1.date_id +
```

```
d2.arriving_day - d3.departing_day)
or (d3.departing_time_number<d2.arriving_time_number and t2.date_id = t1.date_id +
D2.arriving_day + 1 - d3.departing_day))
and t1.date_id = $3 - d1.departing_day--记录每辆车的初始站发车时间，方便过夜车余票统计
and t2.date_id + d4.arriving_day < $4;--如果同时查询返程车次时的限制，防止再返程出发日期前去程还未到
达
```

4) 计算票价

```
select t.train_id_1, t.train_id_2,p1.seat_type as seat_type_1,p3.seat_type as seat_type_2,
t.start_name_1, t.start_time_1, t.start_id_1, t.end_name_1, t.end_time_1, t.end_id_1,
t.train_departing_day_1, t.departing_day_1,t.arriving_day_1, t.start_name_2,
t.start_time_2, t.start_id_2, t.end_name_2, t.end_time_2, t.end_id_2,
t.train_departing_day_2, t.departing_day_2, t.travel_time,t.arriving_day_2, p2.price -
p1.price as price_1,p4.price - p3.price as price_2
into temp22
from temp21 as t, price as p1, price as p2,price as p3, price as p4
where p1.inner_station_id = t.start_id_1 and p2.inner_station_id = t.end_id_1
and p3.inner_station_id = t.start_id_2 and p4.inner_station_id = t.end_id_2
and p1.train_id = t.train_id_1 and p2.train_id = t.train_id_1
and p3.train_id = t.train_id_2 and p4.train_id = t.train_id_2
and p1.seat_type = p2.seat_type
and p3.seat_type = p4.seat_type
and p1.seat_type in
('Hard_Seat','Soft_Seat','Soft_Up','Soft_Down','Hard_Up','Hard_Middle',
'Hard_Down')
and p3.seat_type in
('Hard_Seat','Soft_Seat','Soft_Up','Soft_Down','Hard_Up','Hard_Middle',
'Hard_Down');--计算票价，同时也剔除出不售票的站
```

5) 计算余票

```
select
t.train_id_1,t.train_id_2,t.seat_type_1,t.seat_type_2,t.start_name_1,t.start_time_1,t.end
_name_1,t.end_time_1,t.departing_day_1,t.arriving_day_1,t.price_1,t.start_name_2,t.start_
```

```
time_2,t.end_name_2,t.end_time_2,t.departing_day_2,t.arriving_day_2,t.price_2,t.travel_ti
me, min(r1.tickets) as tickets_1, min(r2.tickets) as tickets_2
into temp23
from remaining_ticket as r1,remaining_ticket as r2 ,temp22 as t
where r1.inner_station_id between t.start_id_1 + 1 and t.end_id_1
and r1.date = t.train_departing_day_1
and r1.train_id = t.train_id_1
and r1.seat_type =  t.seat_type_1
and r2.inner_station_id between t.start_id_2 + 1 and t.end_id_2
and r2.date = t.train_departing_day_2
and r2.train_id = t.train_id_2
and r2.seat_type =  t.seat_type_2
group                                                                                by
t.train_id_1,t.train_id_2,t.seat_type_1,t.seat_type_2,t.start_name_1,t.start_time_1,t.end
_name_1,t.end_time_1,t.departing_day_1,t.arriving_day_1,t.price_1,t.start_name_2,t.start_
time_2,t.end_name_2,t.end_time_2,t.departing_day_2,t.arriving_day_2,t.price_2,t.travel_ti
me;
--计算余票数量，以区间内最少的余票量为准
```

6） 关联上实际日期

为查询快速和能够更改数据库以适应实际情况中不断更新的票价，利用日期 id 查询，在输出返回时，再关联上实际日期。

```
select
t.train_id_1,t.train_id_2,t.seat_type_1,t.seat_type_2,t.start_name_1,t.start_time_1,t.end
_name_1,t.end_time_1,t.price_1,t.start_name_2,t.start_time_2,t.end_name_2,t.end_time_2,t.
price_2,t.travel_time,         t.tickets_1,         t.tickets_2,d1.actual_date         as
departing_date_1,d2.actual_date        as        arriving_date_1,d3.actual_date        as
departing_date_2,d4.actual_date as arriving_date_2
into temp24
from temp23 as t, date as d1, date as d2,date as d3,date as d4
where d1.date_id = t.departing_day_1
and d2.date_id = t.arriving_day_1
and d3.date_id = t.departing_day_2
```

```
and d4.date_id = t.arriving_day_2;
```

7) 整理成输出格式的表

```
select distinct t.train_id_1,t.train_id_2,t.departing_date_1, t.departing_date_2,
t.start_name_1,t.start_name_2,t.start_time_1,t.start_time_2,t.arriving_date_1,t.arriving_
date_2,t.end_name_1,t.end_name_2,t.end_time_1,t.end_time_2,t.travel_time
into temp25
from temp24 as t;

alter table temp25 add Min_Price decimal(5,1) default 9999.9;
alter table temp25 add Min_Price_1 decimal(5,1) default 9999.9;
alter table temp25 add Min_Price_2 decimal(5,1) default 9999.9;

alter table temp25 add Hard_Seat_1 integer;
alter table temp25 add Hard_Seat_Price_1 decimal(5,1);

update temp25
set Hard_Seat_1 = t.tickets_1, Hard_Seat_Price_1 = t.price_1
from temp24 as t
where t.train_id_1 = temp25.train_id_1
and t.start_name_1 = temp25.start_name_1
and t.end_name_1 = temp25.end_name_1
and t.seat_type_1 = 'Hard_Seat';

update temp25
set Min_Price_1 = Hard_Seat_Price_1
where Min_Price_1 > Hard_Seat_Price_1;

alter table temp25 add Hard_Seat_2 integer;
alter table temp25 add Hard_Seat_Price_2 decimal(5,1);

update temp25
set Hard_Seat_2 = t.tickets_2, Hard_Seat_Price_2 = t.price_2
```

```
from temp24 as t

where t.train_id_2 = temp25.train_id_2

and t.start_name_2 = temp25.start_name_2

and t.end_name_2 = temp25.end_name_2

and t.seat_type_2 = 'Hard_Seat';


update temp25

set Min_Price_2 = Hard_Seat_Price_2

where Min_Price_2 > Hard_Seat_Price_2;



alter table temp25 add Soft_Seat_1 integer;

alter table temp25 add Soft_Seat_Price_1 decimal(5,1);


update temp25

set Soft_Seat_1 = t.tickets_1, Soft_Seat_Price_1 = t.price_1

from temp24 as t

where t.train_id_1 = temp25.train_id_1

and t.start_name_1 = temp25.start_name_1

and t.end_name_1 = temp25.end_name_1

and t.seat_type_1 = 'Soft_Seat';


update temp25

set Min_Price_1 = Soft_Seat_Price_1

where Min_Price_1 > Soft_Seat_Price_1;


alter table temp25 add Soft_Seat_2 integer;

alter table temp25 add Soft_Seat_Price_2 decimal(5,1);


update temp25

set Soft_Seat_2 = t.tickets_2, Soft_Seat_Price_2 = t.price_2

from temp24 as t

where t.train_id_2 = temp25.train_id_2
```

```
and t.start_name_2 = temp25.start_name_2
and t.end_name_2 = temp25.end_name_2
and t.seat_type_2 = 'Soft_Seat';

update temp25
set Min_Price_2 = Soft_Seat_Price_2
where Min_Price_2 > Soft_Seat_Price_2;



alter table temp25 add Hard_Up_1 integer;
alter table temp25 add Hard_Up_Price_1 decimal(5,1);

update temp25
set Hard_Up_1 = t.tickets_1, Hard_Up_Price_1 = t.price_1
from temp24 as t
where t.train_id_1 = temp25.train_id_1
and t.start_name_1 = temp25.start_name_1
and t.end_name_1 = temp25.end_name_1
and t.seat_type_1 = 'Hard_Up';

update temp25
set Min_Price_1 = Hard_Up_Price_1
where Min_Price_1 > Hard_Up_Price_1;

alter table temp25 add Hard_Up_2 integer;
alter table temp25 add Hard_Up_Price_2 decimal(5,1);

update temp25
set Hard_Up_2 = t.tickets_2, Hard_Up_Price_2 = t.price_2
from temp24 as t
where t.train_id_2 = temp25.train_id_2
and t.start_name_2 = temp25.start_name_2
and t.end_name_2 = temp25.end_name_2
```

```
and t.seat_type_2 = 'Hard_Up';

update temp25
set Min_Price_2 = Hard_Up_Price_2
where Min_Price_2 > Hard_Up_Price_2;

alter table temp25 add Hard_Middle_1 integer;
alter table temp25 add Hard_Middle_Price_1 decimal(5,1);

update temp25
set Hard_Middle_1 = t.tickets_1, Hard_Middle_Price_1 = t.price_1
from temp24 as t
where t.train_id_1 = temp25.train_id_1
and t.start_name_1 = temp25.start_name_1
and t.end_name_1 = temp25.end_name_1
and t.seat_type_1 = 'Hard_Middle';

update temp25
set Min_Price_1 = Hard_Middle_Price_1
where Min_Price_1 > Hard_Middle_Price_1;

alter table temp25 add Hard_Middle_2 integer;
alter table temp25 add Hard_Middle_Price_2 decimal(5,1);

update temp25
set Hard_Middle_2 = t.tickets_2, Hard_Middle_Price_2 = t.price_2
from temp24 as t
where t.train_id_2 = temp25.train_id_2
and t.start_name_2 = temp25.start_name_2
and t.end_name_2 = temp25.end_name_2
and t.seat_type_2 = 'Hard_Middle';

update temp25
```

```
set Min_Price_2 = Hard_Middle_Price_2
where Min_Price_2 > Hard_Middle_Price_2;

alter table temp25 add Hard_Down_1 integer;
alter table temp25 add Hard_Down_Price_1 decimal(5,1);

update temp25
set Hard_Down_1 = t.tickets_1, Hard_Down_Price_1 = t.price_1
from temp24 as t
where t.train_id_1 = temp25.train_id_1
and t.start_name_1 = temp25.start_name_1
and t.end_name_1 = temp25.end_name_1
and t.seat_type_1 = 'Hard_Down';

update temp25
set Min_Price_1 = Hard_Down_Price_1
where Min_Price_1 > Hard_Down_Price_1;

alter table temp25 add Hard_Down_2 integer;
alter table temp25 add Hard_Down_Price_2 decimal(5,1);

update temp25
set Hard_Down_2 = t.tickets_2, Hard_Down_Price_2 = t.price_2
from temp24 as t
where t.train_id_2 = temp25.train_id_2
and t.start_name_2 = temp25.start_name_2
and t.end_name_2 = temp25.end_name_2
and t.seat_type_2 = 'Hard_Down';

update temp25
set Min_Price_2 = Hard_Down_Price_2
where Min_Price_2 > Hard_Down_Price_2;
```

```
alter table temp25 add Soft_Up_1 integer;
alter table temp25 add Soft_Up_Price_1 decimal(5,1);

update temp25
set Soft_Up_1 = t.tickets_1, Soft_Up_Price_1 = t.price_1
from temp24 as t
where t.train_id_1 = temp25.train_id_1
and t.start_name_1 = temp25.start_name_1
and t.end_name_1 = temp25.end_name_1
and t.seat_type_1 = 'Soft_Up';

update temp25
set Min_Price_1 = Soft_Up_Price_1
where Min_Price_1 > Soft_Up_Price_1;

alter table temp25 add Soft_Up_2 integer;
alter table temp25 add Soft_Up_Price_2 decimal(5,1);

update temp25
set Soft_Up_2 = t.tickets_2, Soft_Up_Price_2 = t.price_2
from temp24 as t
where t.train_id_2 = temp25.train_id_2
and t.start_name_2 = temp25.start_name_2
and t.end_name_2 = temp25.end_name_2
and t.seat_type_2 = 'Soft_Up';

update temp25
set Min_Price_2 = Soft_Up_Price_2
where Min_Price_2 > Soft_Up_Price_2;

alter table temp25 add Soft_Down_1 integer;
alter table temp25 add Soft_Down_Price_1 decimal(5,1);
```

```
update temp25
set Soft_Down_1 = t.tickets_1, Soft_Down_Price_1 = t.price_1
from temp24 as t
where t.train_id_1 = temp25.train_id_1
and t.start_name_1 = temp25.start_name_1
and t.end_name_1 = temp25.end_name_1
and t.seat_type_1 = 'Soft_Down';

update temp25
set Min_Price_1 = Soft_Down_Price_1
where Min_Price_1 > Soft_Down_Price_1;

alter table temp25 add Soft_Down_2 integer;
alter table temp25 add Soft_Down_Price_2 decimal(5,1);

update temp25
set Soft_Down_2 = t.tickets_2, Soft_Down_Price_2 = t.price_2
from temp24 as t
where t.train_id_2 = temp25.train_id_2
and t.start_name_2 = temp25.start_name_2
and t.end_name_2 = temp25.end_name_2
and t.seat_type_2 = 'Soft_Down';

update temp25
set Min_Price_2 = Soft_Down_Price_2
where Min_Price_2 > Soft_Down_Price_2;

update temp25
set Min_Price = Min_Price_1+Min_Price_2;

alter table temp25 drop Min_Price_1;
alter table temp25 drop Min_Price_2;
```

--增加最低票价，统计同一种方案中，不同座位类型最低的票价，方便排序

8） 按要求输出

满足需求中按照票价，行程时间，出发时间计算的功能。

```
select * from temp25 order by Min_Price;

select * from temp25 order by travel_time;

select * from temp25 order by start_time_1;
```

9） 直达当天往返程

```
select t1.train_id as train_id1,
    t1.departing_date as departing_date1,
    t1.start_name as start_name1,
    t1.start_time as start_time1,
    t1.arriving_date as arriving_date1,
    t1.end_name as end_name1,
    t1.end_time as end_time1,
    t1.travel_time as travel_time1,
    t1.min_price as min_price1,
    t1.hard_seat as hard_seat1,
    t1.hard_seat_price as hard_seat_price1,
    t1.soft_seat as soft_seat1,
    t1.soft_seat_price as soft_seat_price1,
    t1.hard_up as hard_up1,
    t1.hard_up_price as hard_up_price1,
    t1.hard_middle as hard_middle1,
    t1.hard_middle_price as hard_middle_price1,
    t1.hard_down as hard_down1,
    t1.hard_down_price as hard_down_price1,
    t1.soft_up as soft_up1,
    t1.soft_up_price as soft_up_price1,
    t1.soft_down as soft_down1,
    t1.soft_down_price as soft_down_price1,
    t2.train_id as train_id2,
```

```
    t2.departing_date as departing_date2,
    t2.start_name as start_name2,
    t2.start_time as start_time2,
    t2.arriving_date as arriving_date2,
    t2.end_name as end_name2,
    t2.end_time as end_time2,
    t2.travel_time as travel_time2,
    t2.min_price as min_price2,
    t2.hard_seat as hard_seat2,
    t2.hard_seat_price as hard_seat_price2,
    t2.soft_seat as soft_seat2,
    t2.soft_seat_price as soft_seat_price2,
    t2.hard_up as hard_up2,
    t2.hard_up_price as hard_up_price2,
    t2.hard_middle as hard_middle2,
    t2.hard_middle_price as hard_middle_price2,
    t2.hard_down as hard_down2,
    t2.hard_down_price as hard_down_price2,
    t2.soft_up as soft_up2,
    t2.soft_up_price as soft_up_price2,
    t2.soft_down as soft_down2,
    t2.soft_down_price as soft_down_price2
into temp16 from temp15 as t1,
    temp151 as t2
where t2.departing_date > t1.arriving_date or (t2.departing_date = t1.arriving_date and
t2.start_time > t1.end_time);
--由于直达数据较少，加入功能能够允许当天往返，检查是否满足去程到达时间比回程发车时间早
```

2. 查询车次指定车次

1) 查询车次级内部站顺序

其中$1 为车次，$2 为日期。

```
select t.actual_date,
    $2 as date_id,
```

```
    d.train_id,
    d.inner_station_id,
    s.station_name,
    d.arriving_time,
    d.departing_time
into temp0
from date as t, station as s, database1 as d
where d.train_id = $1
and t.date_id + d.departing_day = $2
and s.station_id = d.outer_station_id;
--选出所需列车的站名次序
```

2) 查询票价

```
alter table temp0
add Hard_Seat_Price decimal(5,1);
update temp0 set Hard_Seat_Price = p.price
from possible_ticket as p
where temp0.train_id = p.train_id
and temp0.inner_station_id = p.inner_station_id
and p.seat_type = 'Hard_Seat';
alter table temp0
add Soft_Seat_Price decimal(5,1);
update temp0 set Soft_Seat_Price = p.price
from possible_ticket as p
where temp0.train_id = p.train_id
and temp0.inner_station_id = p.inner_station_id
and p.seat_type = 'Soft_Seat';
alter table temp0
add Hard_Sleeper_Up_Price decimal(5,1);
update temp0 set Hard_Sleeper_Up_Price = p.price
from possible_ticket as p
where temp0.train_id = p.train_id
    and temp0.inner_station_id = p.inner_station_id
```

```
    and p.seat_type = 'Hard_Up';
alter table temp0
add Hard_Sleeper_Middle_Price decimal(5,1);
update temp0 set Hard_Sleeper_Middle_Price = p.price
from possible_ticket as p
where temp0.train_id = p.train_id
    and temp0.inner_station_id = p.inner_station_id
    and p.seat_type = 'Hard_Middle';
alter table temp0
add Hard_Sleeper_Down_Price decimal(5,1);
update temp0 set Hard_Sleeper_Down_Price = p.price
from possible_ticket as p
where temp0.train_id = p.train_id
    and temp0.inner_station_id = p.inner_station_id
    and p.seat_type = 'Hard_Down';
alter table temp0
add Soft_Sleeper_Up_Price decimal(5,1);
update temp0 set Soft_Sleeper_Up_Price = p.price
from possible_ticket as p
where temp0.train_id = p.train_id
    and temp0.inner_station_id = p.inner_station_id
    and p.seat_type = 'Soft_Up';
alter table temp0
add Soft_Sleeper_Down_Price decimal(5,1);
update temp0 set Soft_Sleeper_Down_Price = p.price
from possible_ticket as p
where temp0.train_id = p.train_id
    and temp0.inner_station_id = p.inner_station_id
    and p.seat_type = 'Soft_Down';
```

3) 查询余票

```
update temp0 set actual_date = t.actual_date
from the_date as t
```

```
where t.date_id = temp0.date_id;
alter table temp0
add Hard_Seat integer;
update temp0 set Hard_Seat = r.tickets
from remaining_ticket as r
where r.train_id = temp0.train_id
    and r.inner_station_id = temp0.inner_station_id
    and r.inner_station_id > 1
    and r.the_day = temp0.date_id
    and r.seat_type = 'Hard_Seat';
alter table temp0
add Soft_Seat integer;
update temp0
set Soft_Seat = r.tickets
from remaining_ticket as r
where r.train_id = temp0.train_id
    and r.inner_station_id = temp0.inner_station_id
    and r.inner_station_id > 1
    and r.the_day = temp0.date_id
    and r.seat_type = 'Soft_Seat';
alter table temp0
add Hard_Up integer;
update temp0
set Hard_Up = r.tickets
from remaining_ticket as r
where r.train_id = temp0.train_id
    and r.inner_station_id = temp0.inner_station_id
    and r.inner_station_id > 1
    and r.the_day = temp0.date_id
    and r.seat_type = 'Hard_Up';
alter table temp0
add Hard_Middle integer;
update temp0
```

```
set Hard_Middle = r.tickets
from remaining_ticket as r
where r.train_id = temp0.train_id
    and r.inner_station_id = temp0.inner_station_id
    and r.inner_station_id > 1
    and r.the_day = temp0.date_id
    and r.seat_type = 'Hard_Middle';
alter table temp0
add Hard_Down integer;
update temp0
set Hard_Down = r.tickets
from remaining_ticket as r
where r.train_id = temp0.train_id
    and r.inner_station_id = temp0.inner_station_id
    and r.inner_station_id > 1
    and r.the_day = temp0.date_id
    and r.seat_type = 'Hard_Down';
alter table temp0
add Soft_Up integer;
update temp0 set Soft_Up = r.tickets
from remaining_ticket as r
where r.train_id = temp0.train_id
    and r.inner_station_id = temp0.inner_station_id
    and r.inner_station_id > 1
    and r.the_day = temp0.date_id
    and r.seat_type = 'Soft_Up';
alter table temp0
add Soft_Down integer;
update temp0
set Soft_Down = r.tickets
from remaining_ticket as r
where r.train_id = temp0.train_id
    and r.inner_station_id = temp0.inner_station_id
```

```
    and r.inner_station_id > 1
    and r.the_day = temp0.date_id
    and r.seat_type = 'Soft_Down';
update temp0
set Hard_Seat_Price = NULL,
    Soft_Seat_Price = NULL,
    Hard_Sleeper_Up_Price = NULL,
    Hard_Sleeper_Down_Price = NULL,
    Hard_Sleeper_Middle_Price = NULL,
    Soft_Sleeper_Down_Price = NULL,
    Soft_Sleeper_Up_Price = NULL
where Inner_Station_ID = 1;
```

4） 按照车次站内顺序排序

```
update temp0 set actual_date = t.actual_date
from the_date as t
where t.date_id = temp0.date_id;
alter table temp0
add Hard_Seat integer;
update temp0 set Hard_Seat = r.tickets
from remaining_ticket as r
where r.train_id = temp0.train_id
    and r.inner_station_id = temp0.inner_station_id
    and r.inner_station_id > 1
    and r.the_day = temp0.date_id
    and r.seat_type = 'Hard_Seat';
alter table temp0
add Soft_Seat integer;
update temp0
set Soft_Seat = r.tickets
from remaining_ticket as r
where r.train_id = temp0.train_id
    and r.inner_station_id = temp0.inner_station_id
```

```
    and r.inner_station_id > 1
    and r.the_day = temp0.date_id
    and r.seat_type = 'Soft_Seat';
alter table temp0
add Hard_Up integer;
update temp0
set Hard_Up = r.tickets
from remaining_ticket as r
where r.train_id = temp0.train_id
    and r.inner_station_id = temp0.inner_station_id
    and r.inner_station_id > 1
    and r.the_day = temp0.date_id
    and r.seat_type = 'Hard_Up';
alter table temp0
add Hard_Middle integer;
update temp0
set Hard_Middle = r.tickets
from remaining_ticket as r
where r.train_id = temp0.train_id
    and r.inner_station_id = temp0.inner_station_id
    and r.inner_station_id > 1
    and r.the_day = temp0.date_id
    and r.seat_type = 'Hard_Middle';
alter table temp0
add Hard_Down integer;
update temp0
set Hard_Down = r.tickets
from remaining_ticket as r
where r.train_id = temp0.train_id
    and r.inner_station_id = temp0.inner_station_id
    and r.inner_station_id > 1
    and r.the_day = temp0.date_id
    and r.seat_type = 'Hard_Down';
```

```
alter table temp0
add Soft_Up integer;
update temp0 set Soft_Up = r.tickets
from remaining_ticket as r
where r.train_id = temp0.train_id
    and r.inner_station_id = temp0.inner_station_id
    and r.inner_station_id > 1
    and r.the_day = temp0.date_id
    and r.seat_type = 'Soft_Up';
alter table temp0
add Soft_Down integer;
update temp0
set Soft_Down = r.tickets
from remaining_ticket as r
where r.train_id = temp0.train_id
    and r.inner_station_id = temp0.inner_station_id
    and r.inner_station_id > 1
    and r.the_day = temp0.date_id
    and r.seat_type = 'Soft_Down';
update temp0
set Hard_Seat_Price = NULL,
    Soft_Seat_Price = NULL,
    Hard_Sleeper_Up_Price = NULL,
    Hard_Sleeper_Down_Price = NULL,
    Hard_Sleeper_Middle_Price = NULL,
    Soft_Sleeper_Down_Price = NULL,
    Soft_Sleeper_Up_Price = NULL
where Inner_Station_ID = 1;
```

3. 乘客注册与用户信息查询

1) 用户注册表

分别输入用户的一些个人信息，然后插入用户表中。

```
insert into client(National_ID, Passward, Name, Phone, Bank, Credit_Card)
```

```
values($1,$2,$3,$4,$5,$6);
--注册
```

2) 用户信息查询

其中$1 为用户身份证号，$2 为密码。

```
select *
from client
where national_id = $1 and password = $2;
--验证登录使用，是否存在这个账户，并且密码正确。
```

4. 买票

1) 依据指定车次，日期，出发站，到达站给出座位类型

```
select r.seat_type
from remaining_ticket as r,
    train as d1,
    train as d2,
    the_date as t
where d1.train_id = $1
    and d2.train_id = $1
    and r.train_id = $1
    and d1.station_name = $2
    and d2.station_name = $3
    and r.inner_station_id between d1.inner_station_id + 1
    and d2.inner_station_id
    and d1.inner_station_id < d2.inner_station_id
    and r.date + d1.departing_day = t.date_id
    and t.actual_date = cast($4 as date)
    and (t.actual_date > current_date or ( t.actual_date = current_date and
d1.departing_time > current_time))
group by r.seat_type having min(r.tickets) > 0;
```

2) 再次查询确认是否有余票

```
select d1.inner_station_id as inner1,
```

```
    d2.inner_station_id as inner2,
    d1.train_id,
    min(r.tickets),
    r.date,
    r.seat_type
into temp41
from station as s1,
    station as s2,
    train as d1,
    train as d2,
    remaining_ticket as r,
    the_date as td
where s1.station_name = $1
    and s2.station_name = $2
    and s1.station_id = d1.outer_station_id
    and s2.station_id = d2.outer_station_id
    and d1.train_id = d2.train_id
    and d1.train_id = $3
    and r.inner_station_id between d1.inner_station_id + 1 and d2.inner_station_id
    and td.actual_date = cast($4 as date)
    and r.train_id = d1.train_id
    and r.date + d1.departing_day = td.date_id
    and (td.actual_date > current_date or ( td.actual_date = current_date and
d1.departing_time > current_time))
    and r.seat_type = $5
group by d1.inner_station_id, d2.inner_station_id, d1.train_id, r.tickets, r.date,
r.seat_type;
--验证所预订票信息，可以买票，并且还未发车。
----------------------------------------
select *
from temp41
where min > 0;
--是否还有票，1 行，代表可以买票，0 行不能买票。
```

3) 购票

加入订单表，并将对应票数减一。

```
insert into order_table(order_id,
    train_id,
    national_id,
    departure_station_id,
    destination_station_id,
    price,
    seat_type,
    departure_time,
    departure_date,
    order_time)
select $1, t.train_id, $2, t.inner1, t.inner2, t.price, t.seat_type, d.departing_time,
td.actual_date, current_timestamp
from temp41 as t, train as d, the_date as td, client as c
where t.train_id = d.train_id and t.inner1 = d.inner_station_id and td.date_id = t.date +
d.departing_day and c.national_id = $2;

update remaining_ticket
set tickets = tickets - 1
from temp41 as t
where remaining_ticket.inner_station_id between t.inner1 + 1 and t.inner2
    and remaining_ticket.train_id = t.train_id
    and remaining_ticket.date = t.date
    and remaining_ticket.seat_type = t.seat_type
    and t.min > 0;
```

5. 订单查询

1) 指定订单号查询

$1 为订单号，$2 为客户身份证号（由于订单号为下单时间，需要利用客户信息与下单时间联合确定）。

```
select o.order_id,
```

```
    o.train_id,
    o.departure_date,
    o.departure_time,
    s1.station_name,
    s2.station_name,
    o.price,
    o.seat_type,
    o.order_time
from order_table as o,
    station as s1,
    station as s2,
    train as d1,
    train as d2
where o.order_id = $1
    and d1.train_id = o.train_id
    and d2.train_id = o.train_id
    and o.national_id = $2
    and s1.station_id = d1.outer_station_id
    and d1.inner_station_id = o.departure_station_id
    and s2.station_id = d2.outer_station_id
    and d2.inner_station_id = o.destination_station_id
order by o.departure_date,
    o.departure_time;
--给定订单号和用户，查找订单
```

2)　指定日期查询

$1 为初始日期，$2 为结束日期，$3 为发起查询的客户的身份证。

```
select o.order_id,
    o.train_id,
    o.departure_date,
    o.departure_time,
    s1.station_name,
    s2.station_name,
```

```
    o.price
from order_table as o,
    station as s1,
    station as s2,
    train as d1,
    train as d2,
    the_date as td1,
    the_date as td2,
    the_date as td
where o.departure_date = td.actual_date
    and td1.actual_date = cast($1 as date)
    and td2.actual_date = cast($2 as date)
    and td.date_id between td1.date_id and td2.date_id
    and o.national_id = $3
    and d1.train_id = o.train_id
    and d2.train_id = o.train_id
    and s1.station_id = d1.outer_station_id
    and d1.inner_station_id = o.departure_station_id
    and s2.station_id = d2.outer_station_id
    and d2.inner_station_id = o.destination_station_id;
--按照给定日期区间查找订单
```

3) 依据客户信息查询

给出查询客户所有的订单。

```
select o.order_id,
    o.train_id,
    o.departure_date,
    o.departure_time,
    s1.station_name,
    s2.station_name,
    o.price
from order_table as o,
    station as s1,
```

```
    station as s2,
    train as d1,
    train as d2
where o.national_id = $1
    and d1.train_id = o.train_id
    and d2.train_id = o.train_id
    and s1.station_id = d1.outer_station_id
    and d1.inner_station_id = o.departure_station_id
    and s2.station_id = d2.outer_station_id
    and d2.inner_station_id = o.destination_station_id;
--查找某位客户的所有订单
```

4) 删除订单

从订单表中删除订单

```
delete from order_table where order_id = $1 and national_id = $2
```

删除订单后需要恢复余票

```
update remaining_ticket
set tickets = tickets + 1
from order_table as o, the_date as td, database1 as d
where remaining_ticket.inner_station_id between o.departure_station_id + 1
    and o.destination_station_id
    and remaining_ticket.train_id = o.train_id
    and remaining_ticket.the_day + d.departing_day = td.date_id
    and td.actual_date = o.departure_date
    and d.train_id = o.train_id
    and d.inner_station_id = o.departure_station_id
    and o.order_id = $1
    and o.national_id = $2
    and remaining_ticket.seat_type = o.seat_type;
--恢复余票
```

6. 管理员分析订单

1) 验证管理员身份

```
select *
from administrator
where account = $1 and password = $2;
--验证管理员信息
```

2) 总订单数

```
select distinct order_id, national_id
from order_table;
--计算订单数量，行数就是总订单数。
--在 order_table 里面主键是 order_id,train_id,national_id
--订单号是定单时间转换单，要所以加用户信息以确认。
```

3) 总票价 总订票费

```
select sum(price)
from order_table;
--计算总价格
----------------------------------------
select *
from order_table;
--计算订单费用，一张票在 order_table 里是一条记录
--总行数×5，就是总订单费
```

4) 热点车次分析

```
select train_id, countt
from
(   select train_id, row_number() over(order by countt desc) as rownum, countt
    from
    (   select distinct train_id, COUNT(*) over(PARTITION by train_id) countt
        from order_table
    )   as t1
```

```
)    as t2
order by rownum;
--按照列车订单数量排序
```