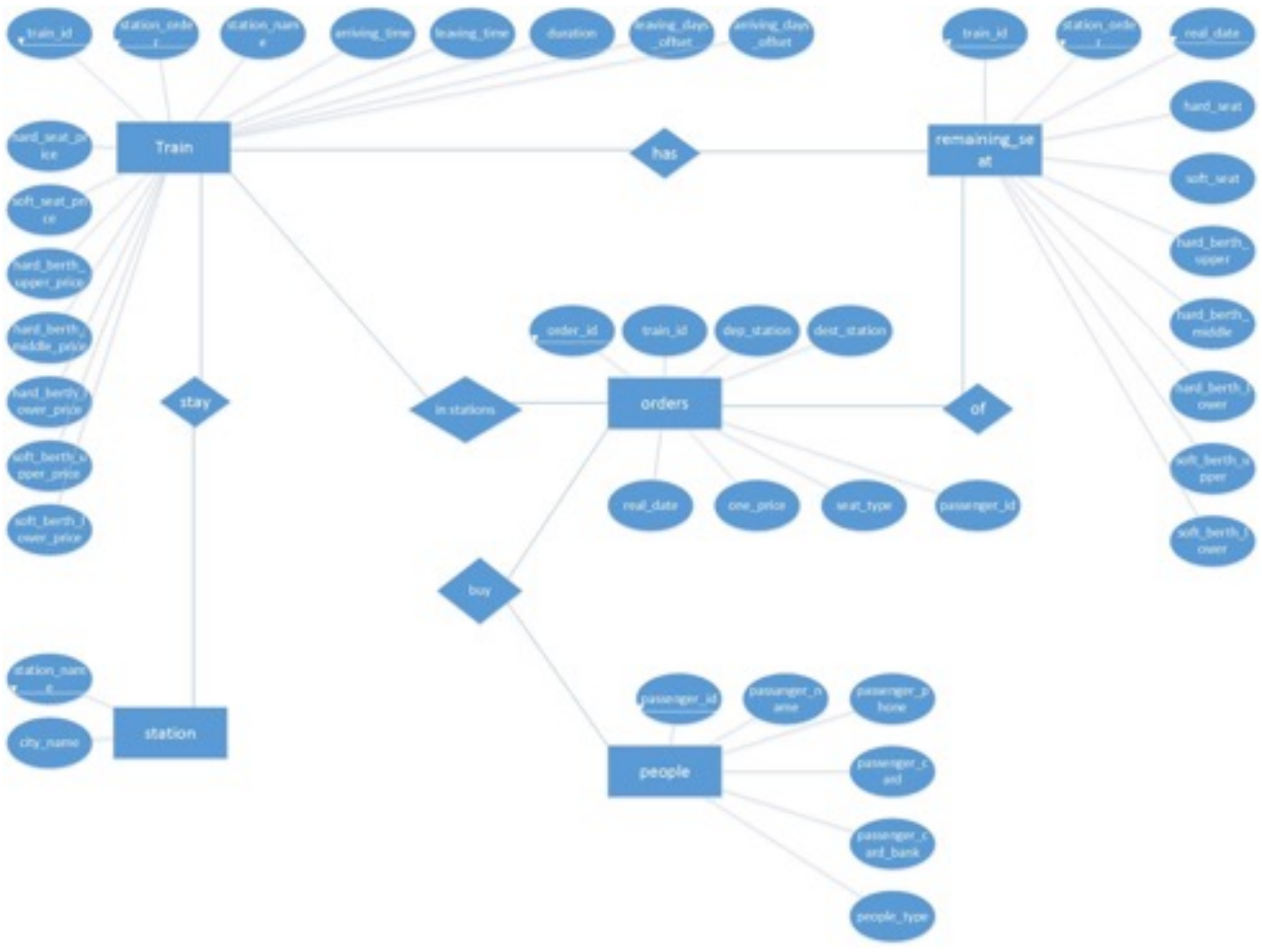
Lab2 设计分析

组名：Uncle Legend’s database
组员：常璐璠 2014K8009929001
王丽娜 2014K8009906002
张传奇 2014K8009929034

首先根据相应的应用需求，我们做出了ER图：



对此ER模型，我们开始建立出详细的关系模型，下面是我们用于建表时的sql语句：

```
--create database train12306;

create table station(
	station_name varchar(16) primary key,
	city_name varchar(16) not null
);


create table train(
	train_id char(5),
	station_order integer,
    station_name varchar(16),
    arriving_time time,
    leaving_time time, ,
    duration integer,
    hard_seat_price   decimal(5,1),
    soft_seat_price    decimal(5,1),
    hard_berth_upper_price decimal(5,1),
    hard_berth_middle_price decimal(5,1),
    hard_berth_lower_price decimal(5,1),
    soft_berth_upper_price decimal(5,1),
    soft_berth_lower_price decimal(5,1),
    leaving_days_offset    integer,
    arriving_days_offset integer,
    primary key(train_id,station_order),
    foreign key(station_name) references station(station_name)
);

create table remaining_seat(
    train_id char(5),
    station_order integer,
    real_date date,
    hard_seat integer default 5,
    soft_seat integer default 5,
    hard_berth_upper integer default 5,
    hard_berth_middle integer default 5,
    hard_berth_lower integer default 5,
    soft_berth_upper integer default 5,
    soft_berth_lower integer default 5,
    primary key(train_id,station_order,real_date),
    foreign key(train_id,station_order)references train(train_id,station_order)
);

create table people(
    passenger_id char(18) primary key,
    passenger_name varchar(16),
    passenger_phone char(10),
    passenger_card char(16),
    passenger_card_bank varchar(100),
    people_type integer not null
);

create table orders(
    order_id integer primary key,
    train_id char(5) not null,
    dep_station varchar(16),
    dest_station varchar(16),
    seat_type varchar(32) not null,
    one_price decimal(5,1) not null,
    dep_date date not null,
    passenger_id char(18) not null,
    foreign key(passenger_id)references people(passenger_id),
    foreign key(dep_station)references station(station_name),
    foreign key(dest_station)references station(station_name)
);
```

此外，我们还建立了Temp表用于减少与数据库的连接次数。每建立一个新的查询请求后，我们会将尽量多的信息传递到Temp表里，再根据用户需求从Temp表里抽取信息。待到新的查询时，drop掉当前

Temp表，建立新的Temp表，重复上述操作。

```
create table temp(
    train_id char(5) not null,
    dep_station varchar(16),
    dest_station varchar(16),
    seat_type varchar(32) not null,
    one_price decimal(5,1) not null,
    real_date date,
    dep_date date,
    dep_station_order integer,
    dest_station_order integer,
    leaving_time time,
    duration integer
);
```

用于查询的sql语句如下：

```
--REQUEST 3
insert into people
values($1, $2, $3, $4, $5, 0);


--REQUEST 4
select t.station_order, t.station_name, t.arriving_time, t.leaving_time, rs.hard_seat, t.hard_seat_price,
rs.soft_seat, t.soft_seat_price, rs.hard_berth_upper, t.hard_berth_upper_price, rs.hard_berth_middle,
t.hard_berth_middle_price, rs.hard_berth_lower, t.hard_berth_lower_price, rs.soft_berth_upper,
t.soft_berth_upper_price, rs.soft_berth_lower, t.soft_berth_lower_price
from train as t, remaining_seat as rs
where t.train_id = 'G111'--$1
and   t.train_id = rs.train_id
and   rs.real_date = '2016-11-01';--$2;


--REQUEST 5 因为涉及到不同的seat_type，故用c对sql进行了处理
--直达情形
main()
{
    char buffer[1024];

    char sty[7][30];
    sprintf（sty[0], "%s", "hard_seat"）;
    sprintf（sty[1], "%s", "soft_seat"）;
    sprintf（sty[2], "%s", "hard_berth_upper"）;
    sprintf（sty[0], "%s", "hard_berth_middle"）;
    sprintf（sty[0], "%s", "hard_berth_lower"）;
    sprintf（sty[0], "%s", "soft_berth_upper"）;
    sprintf（sty[0], "%s", "soft_berth_lower"）;

    int i;
    for(i=0;i<7,i++)
    {
        sprintf(buffer,"insert into
temp(train_id,seat_type,dep_station,leaving_time,duration,dest_station,one_price) \
                                select rs.train_id,’%s’,t1.station_name,t1.leaving_time,t2.duration-
t1.duration,t2.station_name,t2.%s_price-t1.%s_price
                                from remaining_seat as rs, train as t1, train as t2, station as s1, station as s2 \
                                where rs.real_date = date'2016-11-01'-t1.leaving_days_offset \
                                and s1.city_name = '北京' \
                                and s2.city_name = '沈阳' \
                                and rs.train_id = t1.train_id \
                                and t1.train_id = t2.train_id \
                                and rs.station_order between t1.station_order + 1 and t2.station_order \
                                and t1.station_order < t2.station_order \
                                and t1.station_name = s1.station_name \
                                and t2.station_name = s2.station_name \
                                and (t1.%s_price != 0 or t1.station_order=1) \
                                and t2.%s_price != 0 \
                                group by
rs.train_id,t1.station_name,t2.station_name,t1.leaving_time,t1.duration,t2.duration,t1.%s_price,t2.%s_pri
ce \
                                having min(rs.%s) > 0",sty[i]);
     }

}
```

一一次换乘且同站情形

```
main()
{
     char buffer[1024];

     char sty[7][30];
     sprintf（sty[0], "%s", "hard_seat"）；
     sprintf（sty[1], "%s", "soft_seat"）；
     sprintf（sty[2], "%s", "hard_berth_upper"）；
     sprintf（sty[0], "%s", "hard_berth_middle"）；
     sprintf（sty[0], "%s", "hard_berth_lower"）；
     sprintf（sty[0], "%s", "soft_berth_upper"）；
     sprintf（sty[0], "%s", "soft_berth_lower"）；

     int i;
     for(i=0;i<7,i++)
     {
          sprintf(buffer,"select rs1.train_id, rs2.train_id, '%s'
                                from remaining_seat as rs1, remaining_seat as rs2, train as t1, train as t2,
train as t3, train as t4, station as s1, station as s2, station as s3
                                where
                                   rs1.real_date = '2016-11-01'
                                and rs2.real_date = '2016-11-01'
                                and s1.city_name = '北京'
                                and s3.city_name = '上海'

                                and rs1.train_id = t1.train_id
                                and t1.train_id = t2.train_id

                                and rs2.train_id = t3.train_id
                                and t3.train_id = t4.train_id
```

```
                              and s1.city_name != s2.city_name
                              and t2.station_name = t3.station_name
                              and t2.arriving_time + interval '1 hour' < t3.leaving_time
                              and t2.arriving_time + interval '4 hours' > t3.leaving_time

                              and rs1.station_order between t1.station_order + 1 and t2.station_order
                              and t1.station_order < t2.station_order
                              and t1.station_name = s1.station_name
                              and t2.station_name = s2.station_name
                              and (t1.%s_price != 0 or t1.station_order=1)
                              and t2.%s_price != 0

                              and rs2.station_order between t3.station_order + 1 and t4.station_order
                              and t3.station_order < t4.station_order
                              and t3.station_name = s2.station_name
                              and t4.station_name = s3.station_name
                              and (t3.%s_price != 0 or t3.station_order=1)
                              and t3.%s_price != 0

                              group by rs1.train_id, rs2.train_id ,s2.station_name
                              having (min(rs1.%s) > 0 and min(rs2.%s) > 0);",sty[i]);
    }

}
```

——一次换乘、同城市不同站情形

```
main()
{
    char buffer[1024];

    char sty[7][30];
    sprintf（sty[0], "%s", "hard_seat"）；
    sprintf（sty[1], "%s", "soft_seat"）；
    sprintf（sty[2], "%s", "hard_berth_upper"）；
    sprintf（sty[0], "%s", "hard_berth_middle"）；
    sprintf（sty[0], "%s", "hard_berth_lower"）；
    sprintf（sty[0], "%s", "soft_berth_upper"）；
    sprintf（sty[0], "%s", "soft_berth_lower"）；

    int i;
    for(i=0;i<7,i++)
    {
        sprintf(buffer,"select rs1.train_id, rs2.train_id, '%s'
                              from remaining_seat as rs1, remaining_seat as rs2, train as t1, train as t2,
train as t3, train as t4, station as s1, station as s2, station as s3, station as s4
                              where
                                rs1.real_date = '2016-11-01'
                              and rs2.real_date = '2016-11-01'
                              and s1.city_name = '北京'
                              and s4.city_name = '上海'

                              and rs1.train_id = t1.train_id
                              and t1.train_id = t2.train_id

                              and rs2.train_id = t3.train_id
```

```
                                and t3.train_id = t4.train_id

                                and s1.city_name != s2.city_name
                                and s3.city_name != s4.city_name
                                and s2.city_name = s3.city_name
                                and t2.station_name != t3.station_name
                                and t2.arriving_time + interval '2 hours' < t3.leaving_time
                                and t2.arriving_time + interval '4 hours' > t3.leaving_time

                                and rs1.station_order between t1.station_order + 1 and t2.station_order
                                and t1.station_order < t2.station_order
                                and t1.station_name = s1.station_name
                                and t2.station_name = s2.station_name
                                and (t1.%s_price != 0 or t1.station_order=1)
                                and t2.%s_price != 0

                                and rs2.station_order between t3.station_order + 1 and t4.station_order
                                and t3.station_order < t4.station_order
                                and t3.station_name = s3.station_name
                                and t4.station_name = s4.station_name
                                and (t3.%s_price != 0 or t4.station_order=1)
                                and t4.%s_price != 0

                                group by rs1.train_id, rs2.train_id ,s2.station_name, s3.station_name
                                having (min(rs1.%s) > 0 and min(rs2.%s) > 0);",sty[i]);
    }

}
```

—两次换乘，同站＋不同站情形（同站＋同站，不同站＋同站，不同站＋不同站完全类似）

```
main()
{
    char buffer[1024];

    char sty[7][30];
    sprintf（sty[0], "%s", "hard_seat"）;
    sprintf（sty[1], "%s", "soft_seat"）;
    sprintf（sty[2], "%s", "hard_berth_upper"）;
    sprintf（sty[0], "%s", "hard_berth_middle"）;
    sprintf（sty[0], "%s", "hard_berth_lower"）;
    sprintf（sty[0], "%s", "soft_berth_upper"）;
    sprintf（sty[0], "%s", "soft_berth_lower"）;

    int i;
    for(i=0;i<7,i++)
    {
        sprintf(buffer,"select rs1.train_id, rs2.train_id, rs3.train_id, '%s'
                                from remaining_seat as rs1, remaining_seat as rs2, remaining_seat as r3,
train as t1, train as t2, train as t3, train as t4, train as t5, train as t6, station as s1, station as s2, station as
s3, station as s4, station as s5
                                where
                                  rs1.real_date = '2016-11-01'
                                and rs2.real_date = '2016-11-01'
                                and rs3.real_date = '2016-11-01'
and s1.city_name = '北京'
and s5.city_name = '上海'

and rs1.train_id = t1.train_id
and t1.train_id = t2.train_id

and rs2.train_id = t3.train_id
and t3.train_id = t4.train_id

and rs3.train_id = t5.train_id
and t5.train_id = t6.train_id

and s1.city_name != s2.city_name
and t2.station_name = t3.station_name
and t2.arriving_time + interval '1 hour' < t3.leaving_time
and t2.arriving_time + interval '4 hours' > t3.leaving_time

and rs1.station_order between t1.station_order + 1 and t2.station_order
and t1.station_order < t2.station_order
and t1.station_name = s1.station_name
and t2.station_name = s2.station_name
and (t1.%s_price != 0 or t1.station_order=1)
and t2.%s_price != 0

and rs2.station_order between t3.station_order + 1 and t4.station_order
and t3.station_order < t4.station_order
and t3.station_name = s2.station_name
and t4.station_name = s3.station_name
and (t3.%s_price != 0 or t3.station_order=1)
and t3.%s_price != 0

and s2.city_name != s3.city_name
and s4.city_name != s5.city_name
and s3.city_name = s4.city_name
and t4.station_name != t5.station_name
and t4.arriving_time + interval '2 hours' < t5.leaving_time
and t4.arriving_time + interval '4 hours' > t5.leaving_time

and rs2.station_order between t3.station_order + 1 and t4.station_order
and t3.station_order < t4.station_order
and t3.station_name = s2.station_name
and t4.station_name = s3.station_name
and (t3.%s_price != 0 or t3.station_order=1)
and t4.%s_price != 0

and rs3.station_order between t5.station_order + 1 and t6.station_order
and t5.station_order < t6.station_order
and t5.station_name = s4.station_name
and t6.station_name = s5.station_name
and (t5.%s_price != 0 or t5.station_order=1)
and t6.%s_price != 0


group by
rs1.train_id,rs2.train_id,rs3.train_id,t1.station_name,t2.station_name,t3.station_name,t4,station_name,t5.
station_name,t6.station_name,t1.leaving_time,t1.duration,t2.duration,t3.duration,t4.durtion,t5.duration,t6.
duration,t1.%s_price,t2.%s_price,t3.%s_price,t4.%sprice,t5.%s_price,t6.%s_price,
                                        having (min(rs1.%s) > 0 and min(rs2.%s) and min(rs3.%s) > 0);",sty[i]);
        }

}
```

ps. 由于查询结果不能统一，网页实现较为困难，故有换乘的情形仅给出了sql语句，尚未在网页上实现。

```
--REQUEST 8
select o.order_id, o.train_id, o.dep_station, o.dest_station, o.seat_type, o.one_peice, o.real_date,
t.leaving_time, t2.arrving_time
from train as t, train as t2, orders as o
where o.order_id= $1
and t.train_id = o.train_id
and t2.train_id = o.train_id
and t.station_name = o.dep_station
and t2.station_name = o.dest_station
and o.passenger_id = $2;

select o.order_id, o.train_id, o.dep_station, o.dest_station, o.real_date, t.leaving_time,
t.arrving_time,o.one_price
from train as t, orders as o
where t.train_id = o.train_id
and t.station_name = o.dep_station
and $1 <= o.real_date
and o.real_date<= $2
and o.passenger_id = $3;


--REQUEST 9
select count(*)
from orders;

select sum(one_price)
from orders;

select 5*count(train_id)
from orders;

select orders.train_id, count(*)as Number
from orders
group by train_id
order by Number desc
limit 10;
```

其余Request直接实现了一个form，沿用查询的cgi实现，不过输入参数的时候将出发城市和到达城市修改了顺序。

范式分析：

首先我们确认，这些属性都是不可再分割的，这样满足了 1NF。

对于那些联合主键的关系模式：我们需要检查 train表和remaining_seat表。train表中的每个属性都是由是何车次和过站顺序决定的，任取两者之一都不能决定；remaining_seat

表中的每个属性都是由是何车次，过站顺序和发车日期决定的，在其中任取都无法决定余票数据；这样我们确认了我们满足2NF。

另外我们还要检查是否满足3NF：大部分是满足的。

但是，在people表中，可以passenger_id→passenger_card→passenger_card_bank这样的依赖关系，但是这里面，我们对于passenger card到bank的映射具体函数尚不明确，所以只得进行记录。

在orders表中，可以有dep_station dest_station one_price → seat_type，这里为了能够让后面的订单显示的需求更方便，我们决定在这里加入seat_type这一项。

其余满足了3NF的表，一样也满足了BCNF