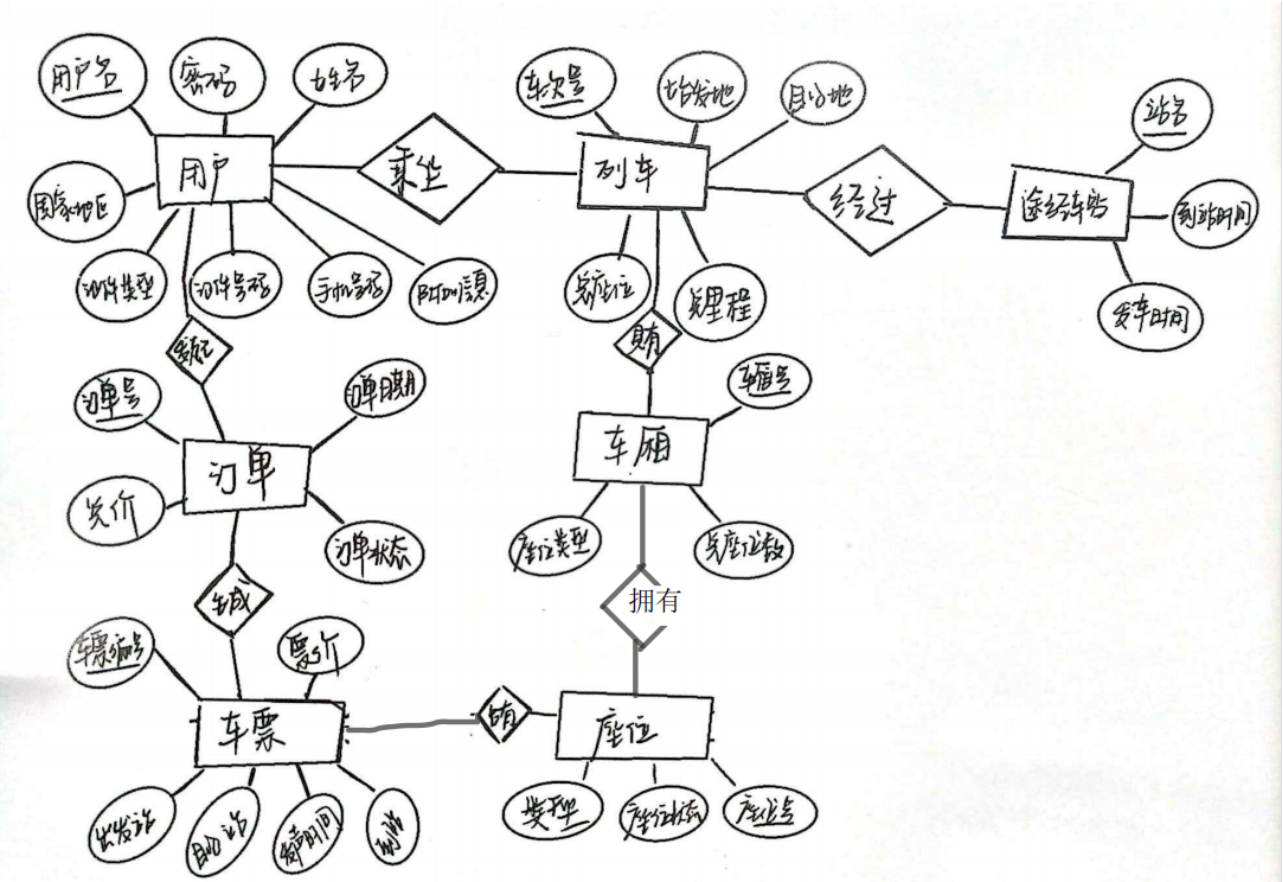
12306铁路购票系统的数据库设计

1.应用领域的详细需求描述

1. 用户注册：用户在进行网上购票的过程中，要先进行注册、填写有关信息。
2. 车票信息更新：系统随时跟据车辆的变化情况进行更新，增加车辆时可以在系统中插入该车的详细信息，同时也能根据需要修改某趟车经过的站点与发车时间。当某趟车停止、开行时可以删除该趟车的信息。同时包括对列车信息、站点信息、订票记录等进行更新。
3. 购买车票：用户发起订单，生成一张新的车票，并且改变座位状态
4. 改签、退票：和购买车票的操作类似，对座位状态进行更新。
5. 查询车票：可以通过起点站和终点站进行车票查询，也可以根据具体的车次输入进行查询，同时每次查询都会显示车次号，出发时间和到达时间。（点击列车可以详细查看经停站）

2.采用教材中介绍的方法实现如下设计



b) 将上述ER图转换成关系模式，并标明每个关系的主键属性和外键属性

用户（用户名，密码，姓名，国家地区，证件类型，证件号码，手机号码，附加信息）主键：用户名

列车（车次号，始发地，目的地，总座位，总里程，）主键：车次号

乘坐（用户名，车次号）主键：用户名，车次号；外键：用户名，车次号

途径车站（站名，到站时间，发车时间）主键：站名

经过（站名，车次号）主键：站名，车次号；外键：站名，车次号

订单（订单号，总价，订单日期，订单状态，用户名）主键：订单号；外键：用户名

车厢（车厢号，座位类型，总座位数，列车号）主键：车厢号；外键：列车号

座位（座位号，座位类型，座位状态，列车号，车厢号，车票编号）主键：座位号；外键：列车号，车厢号，车票编号

车票（车票编号，票价，座位号，车次号，车厢号，出发站，目的站，发车时间，到站时间，订单号）主键：车票编号；外键：座位号，订单号，车厢号，车次号

c) 用SQL语句创建上述关系模式。

```
create table 用户(
    用户名 char(18) primary key,
    密码 char(18) NOT NULL,
    姓名 char(18) NOT NULL,
    国家地区 char(18),
    证件类型 char(52) NOT NULL,
    证件号码 char(52) NOT NULL,
    手机号码 char(52),
    附加信息 char(256)
);

create table 列车(
    车次号 char(18) primary key,
    始发地 char(18),
    目的地 char(18),
    总座位 int,
    总里程 float
);
```

```
create table 乘坐（
     用户名 char(18),
     车次号 char(18),
     PRIMARY KEY(用户名,车次号),
     foreign key(用户名) references 用户(用户名) ,
     foreign key(车次号) references 列车(车次号)
);

create table 途径车站
(
   站名   char(50) primary key,
   到站时间  datetime,
   发车时间  datetime
);

create table 经过
(
   站名         char(50) not null,
   车次号       char(18) not null,
   primary key (站名, 车次号),
    foreign key(站名) references 途径车站(站名) ,
    foreign key(车次号) references 列车(车次号)
);

create table 订单
(
   订单号           char(18) primary key,
   订单日期         datetime,
   总价             float,
   订单状态         char(18),
   用户名          char(18) not null,
    foreign key(用户名) references 用户(用户名)
);
create table 车厢
(
   车次号                    char(18) not null,
   车厢号                    char(18) primary key,
   座位类型                   char(18),
   总座位数                   int,
    foreign key(车次号) references 列车(车次号)
);
create table 座位
(
```

```
    座位号                    char(18) primary key,
    车票编号                  char(18) not null,
    车厢号                    char(18) not null,
    车次号                    char(18) not null,
    座位类型                   char(18),
    状态（是否被占有）          bool,
     foreign key(车次号) references 车厢(车次号) ,
     foreign key(车厢号) references 车厢(车厢号)
);
create table 车票
(
    车票编号                   char(18) primary key,
    座位号                    char(18) not null,
    订单号                     char(18) not null,
    车次号                    char(18) not null,
    车厢号                    char(18) not null,
    出发站                    char(18),
    目的站                    char(18),
    发车时间                   datetime,
    到站时间                   datetime,
    车票票价                   float,
     foreign key(车次号) references 座位(车次号) ,
     foreign key(车厢号) references 座位(车厢号) ,
     foreign key(座位号) references 座位(座位号) ,
     foreign key(订单号) references 订单(订单号)
);
```

d) 给出该数据库模式上5个查询语句样例。

单表查询：

```
SELECT 始发地 FROM 列车 WHERE 车次号='GF200';
#在列车表里寻找车次号为GF200的列车的始发地
```

多表连接查询:

```
SELECT 列车.始发地
FROM 列车, 经过
WHERE 列车.车次号 = 经过.车次号 and 经过.站名='天津';
#寻找经过天津站的所有列车的始发地
```

多表嵌套查询：

```
SELECT 总里程
FROM 列车
WHERE 车次号
IN(SELECT 车次号 FROM 车票 WHERE 车票编号='666');
#寻找在车票表中，车票编号为666的车票的列车行进的总里程
```

EXISTS查询：

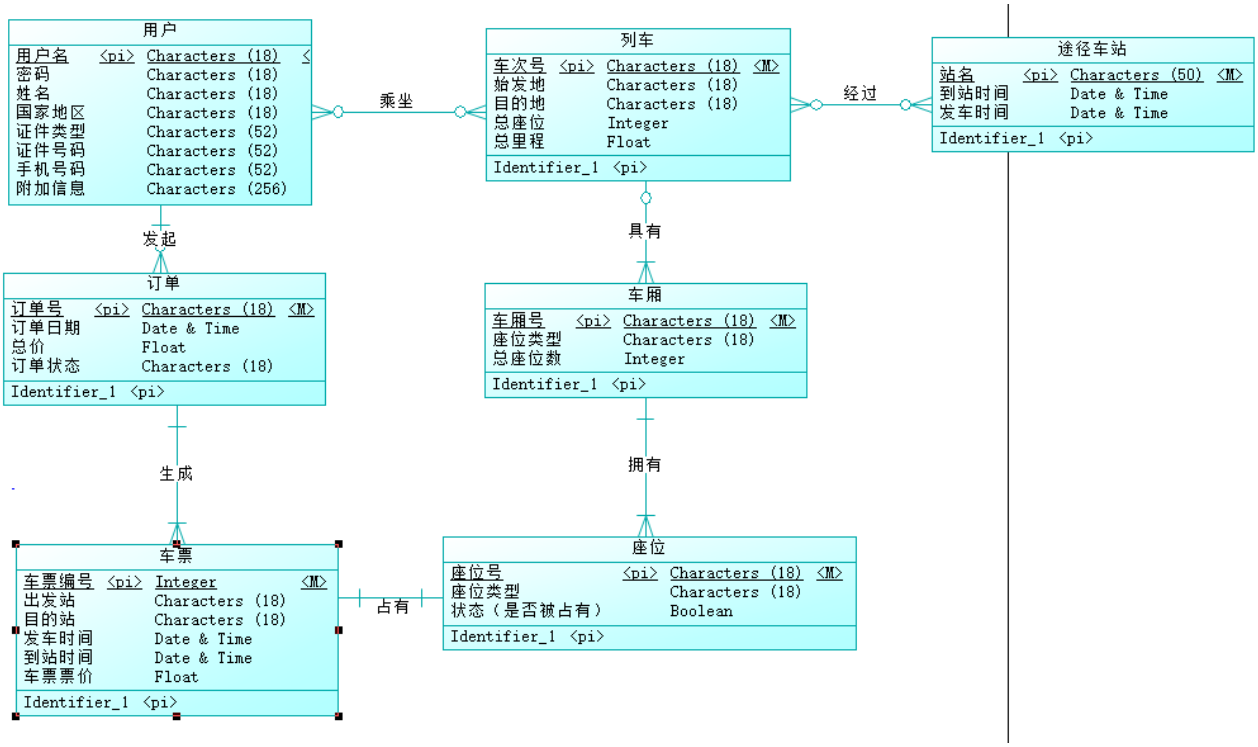
```
SELECT *
FROM 订单
WHERE
EXISTS(SELECT * FROM 用户 WHERE 用户名='Nicole') AND 订单.用户名
='Nicole';
#如果用户名为Nicole的用户存在，查找与她相关的所有订单
```

聚合操作查询：

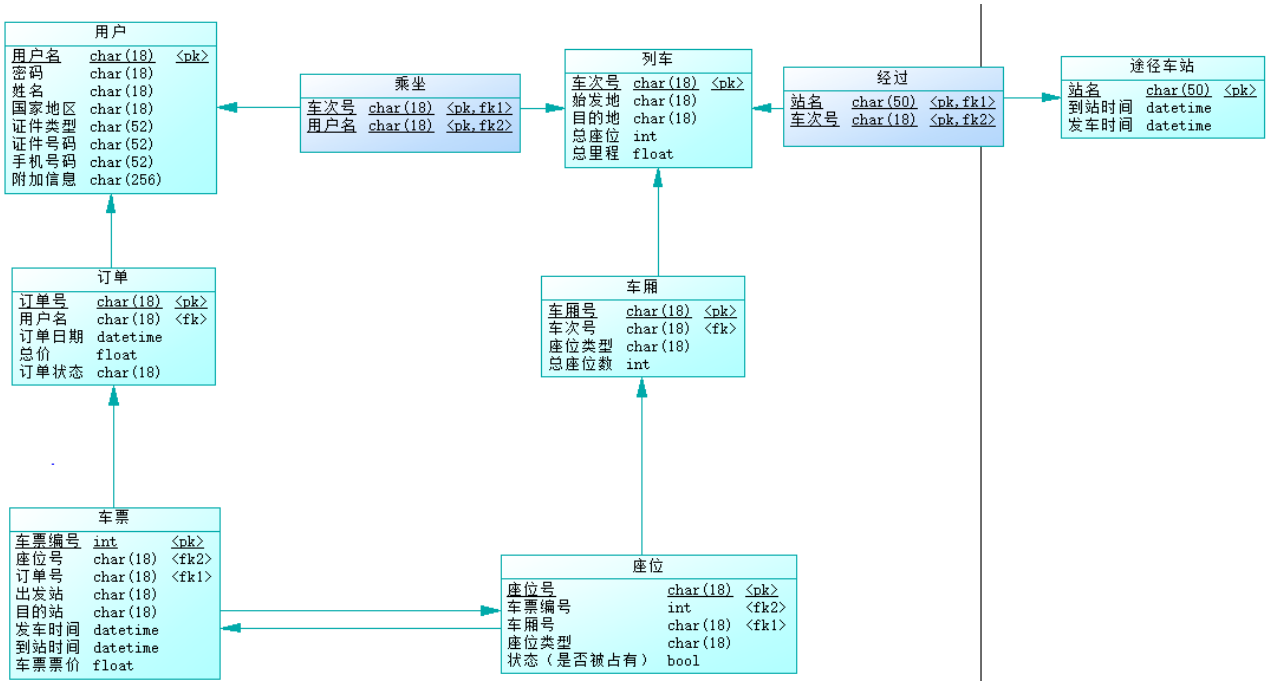
```
SELECT COUNT(*) FROM 用户 GROUP BY 国家地区;
#求用户表中不同国家的注册用户分别有多少个
```

3.使用**PowerDesigner**工具实现如下设计

a) 画出该领域的概念模型ER图，给出ER图截图；



b) 使用PowerDesigner工具，将上述ER图转为关系模型图，给出关系模型图截图；



c) 使用PowerDesigner工具，生成创建数据库的SQL语句

```
/*==============================================================*/
/* DBMS name:      MySQL 5.0                                    */
/* Created on:     2022/4/13 22:25:51                           */
/*==============================================================*/


drop table if exists 乘坐;
```

```
drop table if exists 列车;

drop table if exists 座位;

drop table if exists 用户;

drop table if exists 经过;

drop table if exists 订单;

drop table if exists 车厢;

drop table if exists 车票;

drop table if exists 途径车站;

/*==============================================================*/
/* Table: 乘坐                                                  */
/*==============================================================*/
create table 乘坐
(
   车次号                  char(18) not null,
   用户名                  char(18) not null,
   primary key (车次号, 用户名)
);

/*==============================================================*/
/* Table: 列车                                                  */
/*==============================================================*/
create table 列车
(
   车次号                  char(18) not null,
   始发地                  char(18),
   目的地                  char(18),
   总座位                  int,
   总里程                  float,
   primary key (车次号)
);

/*==============================================================*/
/* Table: 座位                                                  */
/*==============================================================*/
create table 座位
```

```
(
   座位号                    char(18) not null,
   车票编号                   int not null,
   车厢号                    char(18) not null,
   座位类型                   char(18),
   状态（是否被占有）              bool,
   primary key (座位号)
);

/*==============================================================*/
/* Table: 用户                                                  */
/*==============================================================*/
create table 用户
(
   用户名                    char(18) not null,
   密码                     char(18),
   姓名                     char(18),
   国家地区                   char(18),
   证件类型                   char(52),
   证件号码                   char(52),
   手机号码                   char(52),
   附加信息                   char(256),
   primary key (用户名)
);

/*==============================================================*/
/* Table: 经过                                                  */
/*==============================================================*/
create table 经过
(
   站名                     char(50) not null,
   车次号                    char(18) not null,
   primary key (站名, 车次号)
);

/*==============================================================*/
/* Table: 订单                                                  */
/*==============================================================*/
create table 订单
(
   订单号                    char(18) not null,
   用户名                    char(18) not null,
   订单日期                   datetime,
```

```
    总价                    float,
    订单状态                 char(18),
    primary key (订单号)
);

/*==============================================================*/
/* Table: 车厢                                                  */
/*==============================================================*/
create table 车厢
(
    车厢号                  char(18) not null,
    车次号                  char(18),
    座位类型                 char(18),
    总座位数                 int,
    primary key (车厢号)
);

/*==============================================================*/
/* Table: 车票                                                  */
/*==============================================================*/
create table 车票
(
    车票编号                 int not null,
    座位号                  char(18) not null,
    订单号                  char(18) not null,
    出发站                  char(18),
    目的站                  char(18),
    发车时间                 datetime,
    到站时间                 datetime,
    车票票价                 float,
    primary key (车票编号)
);

/*==============================================================*/
/* Table: 途径车站
*/
/*==============================================================*/
create table 途径车站
(
    站名                    char(50) not null,
    到站时间                 datetime,
    发车时间                 datetime,
    primary key (站名)
```

```
);

alter table 乘坐 add constraint FK_乘坐 foreign key (车次号)
      references 列车 (车次号);

alter table 乘坐 add constraint FK_乘坐2 foreign key (用户名)
      references 用户 (用户名);

alter table 座位 add constraint FK_占有2 foreign key (车票编号)
      references 车票 (车票编号);

alter table 座位 add constraint FK_拥有 foreign key (车厢号)
      references 车厢 (车厢号);

alter table 经过 add constraint FK_经过 foreign key (站名)
      references 途径车站 (站名);

alter table 经过 add constraint FK_经过2 foreign key (车次号)
      references 列车 (车次号);

alter table 订单 add constraint FK_发起 foreign key (用户名)
      references 用户 (用户名);

alter table 车厢 add constraint FK_具有 foreign key (车次号)
      references 列车 (车次号);

alter table 车票 add constraint FK_占有 foreign key (座位号)
      references 座位 (座位号);

alter table 车票 add constraint FK_生成 foreign key (订单号)
      references 订单 (订单号);
```

4.分析比较采用上述两种方法

a) 两种关系模式的设计是否存在差异？如有差异，这种差异是否对后期的实现带来不同的影响？

关系模式相同。但是PowerDesigner工具生成的SQL语句最后会添加外键约束和关系表与实体表合并声明，而自己写的SQL语言在创建时声明

b) PowerDesigner工具生成的SQL语句有什么样的特点？ 为什么会出现一些附加语句？它的作用是什么？

PowerDesigner工具生成的SQL语句更加的完整。

附加了drop table if exists 用户语句，即如果与创建的表的名字相同的表，就删除原来的表格，这样可以保证表格成功创建。

附加了外键约束语句，通过外键约束，确保表格之间数据的完整性和准确性。