PRACTICAL – XII

OPEN ENDED PROBLEM
(Railway Reservation System)

Submitted By

Varun Gaur

SY CE-2B (2022-23 Batch)

210410107077/ 21BECEG102

INTRODUCTION:

The Railway Reservation System facilitates the passengers to enquire about the trains available on the basis of source and destination, Booking and Cancellation of tickets, enquire about the status of the booked ticket, etc. The aim of case study is to design and develop a database maintaining the records of different trains, train status, and passengers.

This project contains Introduction to the Railways reservation system .It is the computerized system of reserving the seats of train seats in advanced. It is mainly used for long route. On-line reservation has made the process for the reservation of seats very much easier than ever before.

In our country India, there are number of counters for the reservation of the seats and one can easily make reservations and get tickets. Then this project contains entity relationship model diagram based on railway reservation system and introduction to relation model .There is also design of the database of the railway reservation system based on relation model. Example of some SQL queries to retrieves data from rail management database.

Database is an organized collection of data. The data is typically organized to model aspects of reality in a way that supports processes requiring information. A DBMS makes it possible for end users to create, read, update and delete data in a database. The DBMS essentially serves as an interface between the database and end users or application programs, ensuring that data is consistently organized and remains easily accessible. The DBMS manages three important things: the data, the database engine that allows data to be accessed, locked and modified and the database schema, which defines the database's logical structure. These three foundational elements help provide concurrency, security, data integrity and uniform administration procedures. The DBMS can offer both logical and physical data independence. That means it can protect users and applications from needing to know where data is stored or having to be concerned about changes to the physical structure of data.

The main purpose of mamaliga database for Railway Reservation System is to reduce the manual errors involved in the booking and cancelling of tickets and make it convenient for the customers and providers to maintain the data about their customers and also about the seats available at them. Due to automation many loopholes that exist in the manual maintenance of the records can be removed. The speed of obtaining and processing the data will be fast. For future expansion the proposed system can be web enabled so that clients can make various enquiries about trains between stations. Due to this, sometimes a lot of problems occur and they are facing many disputes with customers. To solve the above problem, we design a data base which includes customer details, availability of seats in trains, no of trains and their details.

The railway reservation system facilitates the passengers to enquire about the trains available on the basis of source and destination, booking and cancellation of tickets, enquire about the status of the booked ticket, etc. The aim of case study is to design

and develop a database maintaining the records of different trains, train status, and passengers. The record of train includes its number, name, source, destination, and days on which it is available, whereas record of train status includes dates for which tickets can be booked, total number of seats available, and number of seats already booked.

Passengers can book their tickets for the train in which seats are available. For this, passenger has to provide the desired train number and the date for which ticket is to be booked. Before booking a ticket for a passenger, the validity of train number and booking date is checked. Once the train number and booking date are validated, it is checked whether the seat is available. If yes, the ticket is booked with confirm status and corresponding ticket ID is generated which is stored along with other details of the passenger. The ticket once booked can be cancelled at any time. For this, the passenger has to provide the ticket ID (the unique key). The ticket ID is searched and the corresponding record is deleted. With this, the first ticket with waiting status also gets confirmed.

List of Assumption Since the reservation system is very large in reality, it is not feasible to develop the case study to that extent and prepare documentation at that level. Therefore, a small sample case study has been created to demonstrate the working of the reservation system. To implement this sample case study, some assumptions have been made, which are as follows:

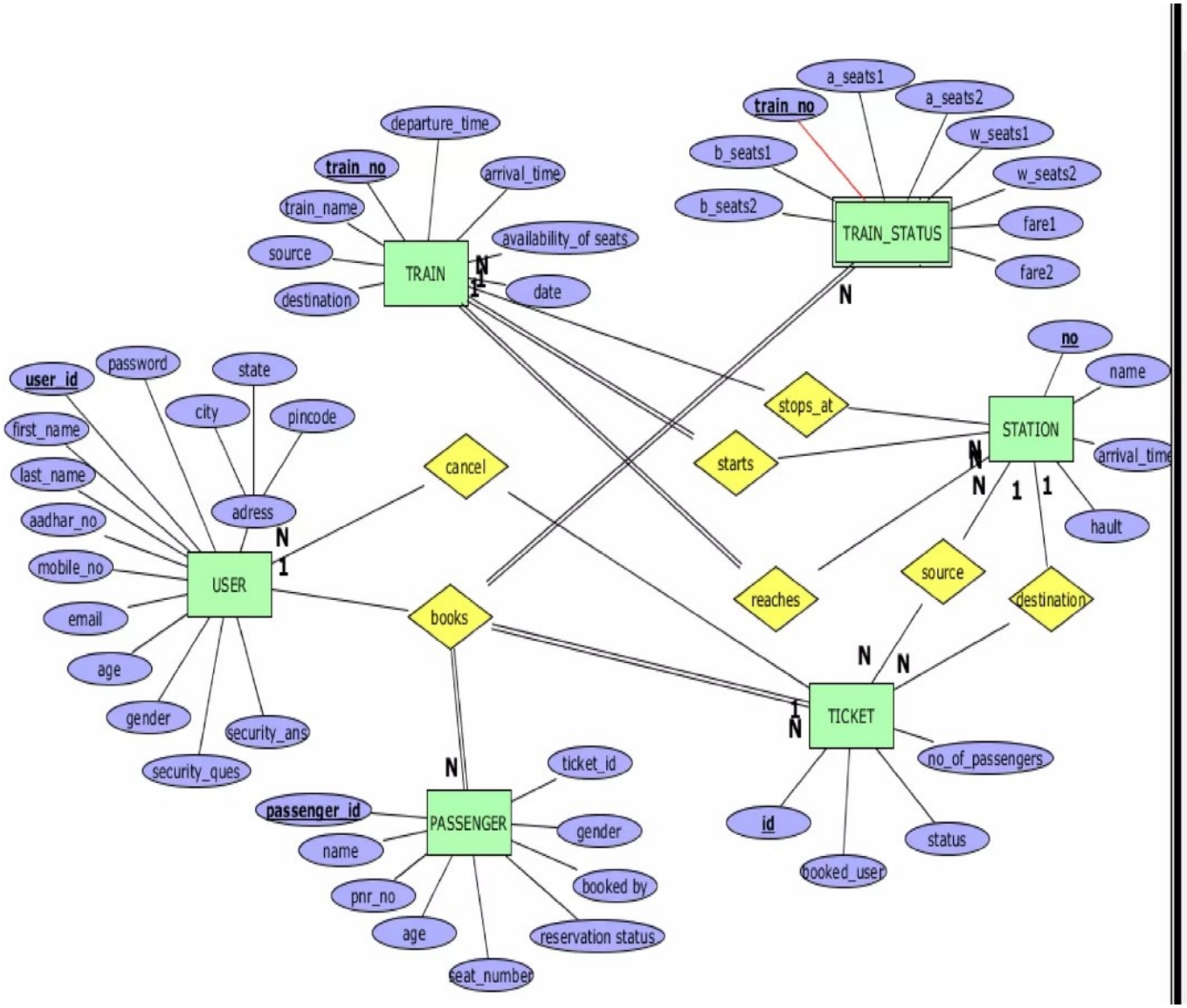
1. The number of trains has been restricted to 5.
2. The booking is open only for next seven days from the current date.
3. Only two categories of tickets can be booked, namely, AC and General.
4. The total number of tickets that can be booked in each category (AC and General) is 10.
5. The total number of tickets that can be given the status of waiting is 2.
6. The in- between stoppage stations and their bookings are not considered.

List of trams has to be maintained. Detailed Passenger information is to be maintained In the booking procedure, the train number, train date, and category are read from the passenger. On the basis of the values provided by the passenger, corresponding record is retrieved from the Train Status. If the desired category is AC, then total number of AC seats and number of booked AC seats are compared in order to find whether ticket can be booked or not. Similarly, it can be checked for the general category. If ticket can be booked, then passenger details are read and stored in the Passenger table. In the cancellation procedure, ticket ID is read from the passenger and corresponding record is searched in the Passenger. If the record exists, it is deleted. After deleting the record (if it is confirmed), first record with waiting status for the same train and same category are searched from the Passenger table and its status is changed to confirm.

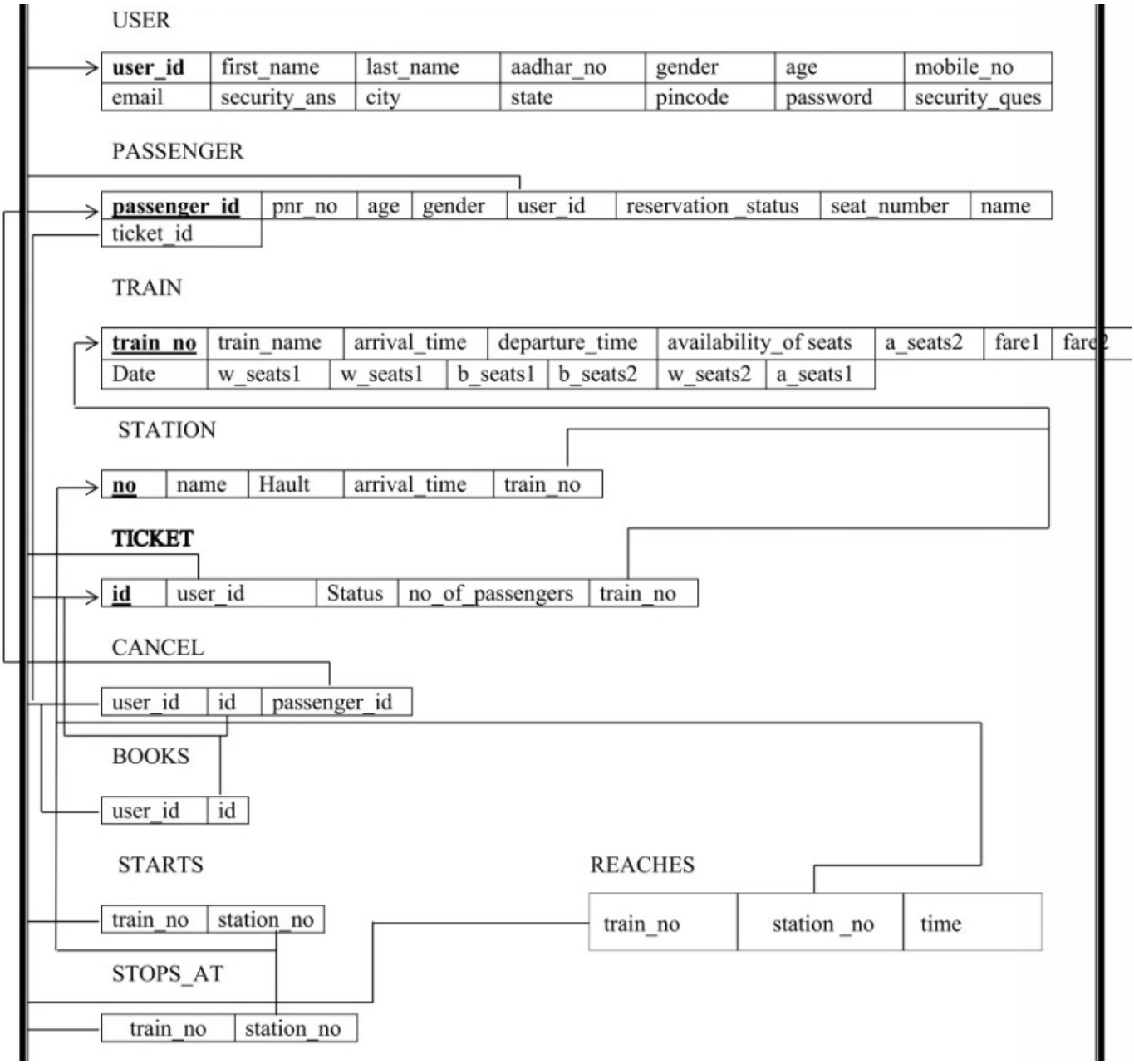
ENTITIES	ATTRIBUTES
User	**User_id**, Password, First_name, Last_name, Gender, Age, Email, Aadhar no, Mobile_no, City, State, Pincode, Security_ques, Security_ans
Passenger	**Passenger_id**, Name, Gender, Age, Pnr_no, Seat_no, Booked_by, Reservation_status

Train	Train no, Train_name, Source, Destination, Arrival_time, Departure_time, Avalibility of seats, A seats1, A seats2, A seats3, B seats1, B seats2, B seats3, W Seats1, W seats2, W seats3
Station	Name, **No,** Train_no, Arrival_time, Hault
Ticket	**Id** Train no Booked user Status No of passengers

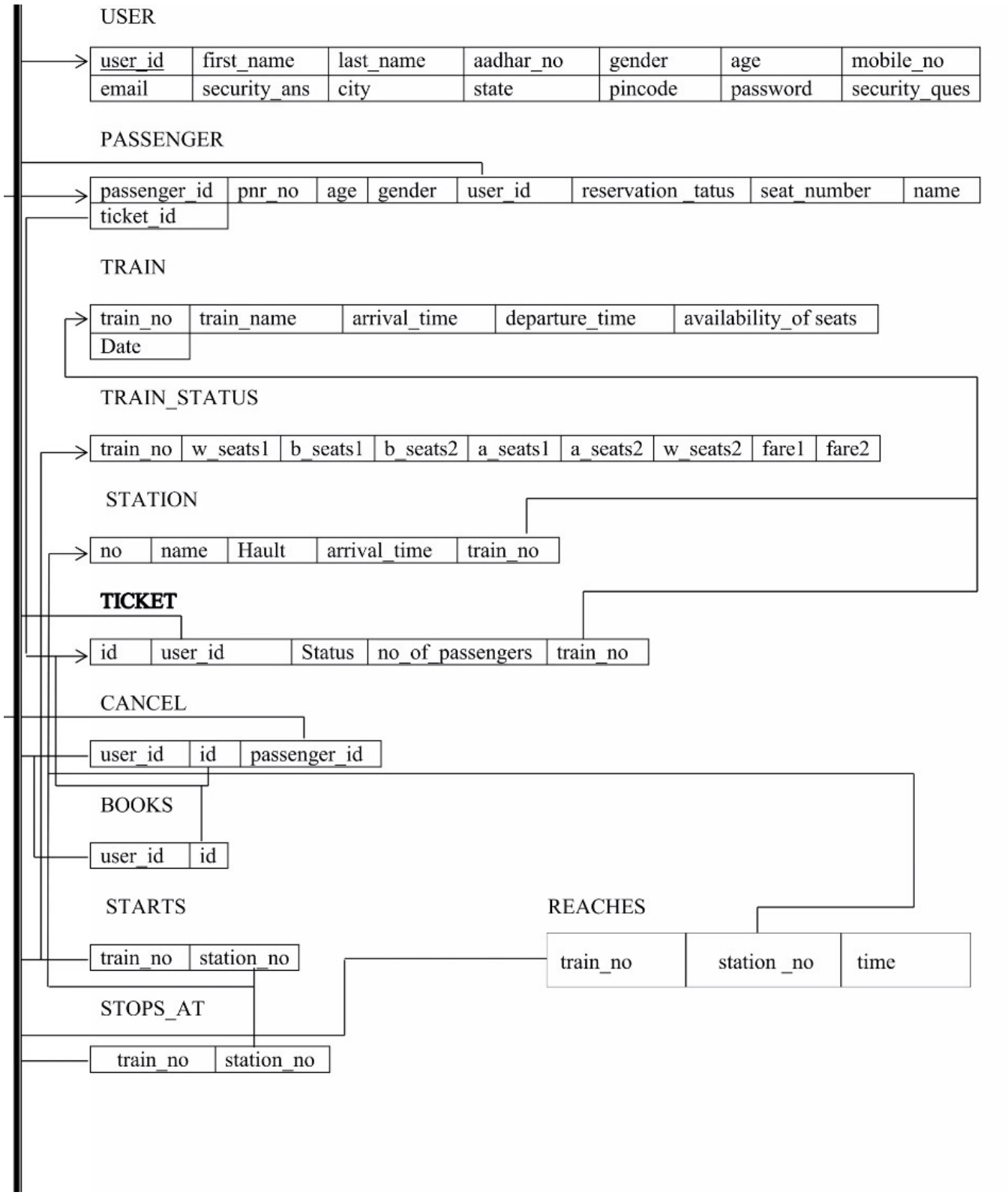
E-R MODEL DIAGRAM



SCHEMA DIAGRAM



NORMALIZATIONS AND FINAL LIST OF RELATIONSHIPS



- books -Temary relationship between USER, TRAIN,PASSENGER and TICKET.
- starts -Between TRAIN and STATION
- reaches -Between TRAIN and STATION
- cancel -Between USER and TICKET
- stops_at-Between TRAIN and STATION

CREATE SQL QUERIES

create table if not exists USER (user_id int primary key, first_name varchar(50), last_name varchar(50), adhar_no varchar(20), gender char, age int, mobile_no varchar(50), email varchar(50), city varchar(50), state varchar(50), pincode varchar(20),_password varchar(50), security_ques varchar(50), security_ans varchar(50));

create table if not exists TRAIN(train_no int primary key, train_name varchar(50), arrival_time time, departure_time time, availability of seats char, date date);

create table if not exists STATION(no int, name varchar(50), hault int, arrival_time time, train_noint, primary key(station_no, train_no), constarint foreign key(train_no) references TRAIN(train_no));

create table if not exists TRAIN STATUS(train no int primary key,b_seats 1 int, b_seats2 int, a_seats 1 int, a_seats2 int, w_seats 1 int, w_seats2 int, fare1 float, fare2 float);

create table if not exists TICKET(id int primary key, user_id int, status char, no_of_passengers int, train_no int, constraint foreign key(user_id) references USER(user_id), constraint foreign key(train_no) references TRAIN(train_no));

create table if not exists PASSENGER(passenger_id int primary key, pnr_no int, age int, gender char, user_id int, reservation_status char, seat number varchar(5), name varchar(50), ticket_id int, constraint foreign key(user_id) references USER (user_id), constraint foreign key(ticket_id) references TICKET(d));

create table if not exists STARTS(train_no int primary key, station_no int. constraint foreign key(train_no) references TRAIN(train_no),constraint foreign key(station_no) references STATION(no))

create table if not exists STOPS_AT(train_no int, station_no int, constraint foreign key(train_no) references TRAIN(train_no).constraint foreign key(station_no) references STATION(no));

create table if not exists REACHES(train_no int, station_no int, time time, constraint foreign key(train_no) references TRAIN(train_no). constraint foreign key(station_no) references STATION(no));

create table if not exists BOOKS(user_id int, id int, constraint foreign key(user_id) references USER(user_id). constraint foreign key(id) references TICKET(id));

create table if not exists CANCEL (user_id int, id int, passenger_id int, constraint foreign key(id) references TICKET(id), constraint foreign key(passenger_id) references PASSENGER(passenger_id), constraint foreign key(user_id) references USER(user id));

INSERT SQL QUERIES

insert into USER (user_id, first_name, last_name, aadhar_no, gender, age, mobile_no, email, city, state, pincode. password, security_ques,security_ans) values

(1701,'vijay', 'sharma', '309887340843', 'M', 34, '9887786655', 'vijay1@gmail.co m', 'vijayawada','andhrapradesh','520001','12345@#', 'favouritecolour','red'),

(1702,'rohith','kumar','456709871234','M',45,'9809666555','rohithkumar@gmail.co m', 'guntur', 'andhrapradesh','522004', '12@#345', 'favouritebike', 'bmw"),

(1703,'manasvi','sree','765843210987','F',20,'9995550666','manasvi57@gmail.com',' guntur', 'andhra pradesh','522004','0987hii','favourite flower','rose');

insert into TRAIN (train_no,train_name,arrival_time,departure_time,availability_of seats,date) values

(12711,'pinakini exp','113000','114000','A',20170410),

(12315,'cormandel exp','124500',125000', 'NA',20170410);

insert into STATION(no,name,hault, arrival_time,train_no) values

(111,'vijayawada', 10,'113000',12711),

(222,'tirupathi',5,'114500',12315);

insert into TRAIN STATUS (train_no,w seats1,b seats 1,b seats2,a seats),a seats2,w_seats 2, farel, fare2) values

(12711,10,4,0,1,1,0,100,450),

(12315,10,5,0,0,2,1,300,600);

insert into TICKET(id,user_id,status,no_of_passengers,train_no) values

(4001,1701,'C',1,,12711),

(4002,1702, NC',1,12315);

insert into PASSENGERS(passenger_id,pnr_no,age, gender,user_id,reservation _status,seat_number,name,ticket_id) values(5001,78965,45,'M',1701,'C','B6-45','ramesh',4001), (5002,54523,54,'F',1701, 'W','B3-21','surekha',4002);

insert into STARTS(train_no,station_no) values(12711,111).(12315,222);

insert into STOPS_AT(train_no,station_no) values(12711,222),(12315,111);

insert into REACHES(train_no,station_no,time) values(12711,222,'040000'), (12315,111,053500');

insert into BOOKS(user_id,id) values(1701,4001),(1702,4002);

insert into CANCEL(user_id,id.passenger_id) values(1701,4001,5001);

SQL QUERIES

1. Print user id and Name of all those user who booked ticket for pinakini express.

select u.user_id,concat (u.first_name, u.last_name)as name

from user u,train t,ticket tc

where u.user_id=tc.user_id and t.train_no-tc.train_no and t.train_name like 'pinakini exp';

```
mysql> select u.user_id,concat(u.first_name,u.last_name)as name
    -> from user u,train t,ticket tc
    -> where u.user_id=tc.user_id and t.train_no=tc.train_no and t.train_name
    -> like'pinakini exp';
+---------+--------------+
| user_id | name         |
+---------+--------------+
|    1701 | vijaysharma  |
|    1701 | vijaysharma  |
+---------+--------------+
2 rows in set (0.00 sec)
```

2. **Print details of passengers travelling under Ticket no 4001**

select*

from passenger

where ticket id like 4001:

```
ysql> select *
    -> from passenger
    -> where ticket_id like 4001;
+--------------+--------+------+--------+---------+--------------------+-------------+--------+-----------+
| passenger_id | pnr_no | age  | gender | user_id | reservation_status | seat_number | name   | ticket_id |
+--------------+--------+------+--------+---------+--------------------+-------------+--------+-----------+
|         5001 |  78965 |   45 | M      |    1701 | C                  | B6-45       | ramesh |      4001 |
+--------------+--------+------+--------+---------+--------------------+-------------+--------+-----------+
1 row in set (0.00 sec)
```

3. **Display all those train no's which reach station no**

select t.*

from train t,station s,reaches r

where t.train_no=r.train_no and r.station_no-s.no and s.name like 'vijayawada';

```
mysql> select t.*
    -> from train t,station s,reaches r
    -> where t.train_no=r.train_no and r.station_no=s.no and s.name like 'vijayawada';
+----------+--------------+--------------+----------------+-------------------------+------------+
| train_no | train_name   | arrival_time | departure_time | availability_of_seats   | date       |
+----------+--------------+--------------+----------------+-------------------------+------------+
|    12315 | cormandel exp | 12:45:00     | 12:50:00       | N                       | 2017-04-10 |
|    12255 | shatabdhi exp | 13:55:00     | 14:00:00       | N                       | 2017-04-11 |
+----------+--------------+--------------+----------------+-------------------------+------------+
2 rows in set (0.00 sec)
```

4. **Display time at which train no- reaches station no**

select r.*.s.name

from reaches r,station s

where r.station_no=s.no;

```
mysql> select r.*,s.name
    -> from reaches r,station s
    -> where r.station_no=s.no;
+----------+------------+----------+------------+
| train_no | station_no | time     | name       |
+----------+------------+----------+------------+
|    12711 |        222 | 04:00:00 | tirupathi  |
|    12315 |        111 | 05:35:00 | vijayawada |
|    12255 |        111 | 06:00:00 | vijayawada |
+----------+------------+----------+------------+
3 rows in set (0.00 sec)
```

5. **Display details of all those users who canceled tickets for train no-----**

select u.*

from user u,cancel c.ticket t

where c.user_id=u.user_id and c.id-t.id and t.train_no like 12711;

```
mysql> select u.*
    -> from user u,cancel c,ticket t
    -> where c.user_id=u.user_id and c.id=t.id and t.train_no like 12711;
+---------+------------+-----------+-------------+--------+------+-----------+---------------+-----------+---------------+---------+-----------+-----------------+
| user_id | first_name | last_name | adhar_no    | gender | age  | mobile_no | email         | city      | state         | pincode | _password | security_ques   |
+---------+------------+-----------+-------------+--------+------+-----------+---------------+-----------+---------------+---------+-----------+-----------------+
|    1701 | vijay      | sharma    | 389887340843 | M      |   34 | 9988776655 | vijay1@gmail.com | vijayawada | andhra pradesh | 520001  | 12345@#   | favourite colou |
+---------+------------+-----------+-------------+--------+------+-----------+---------------+-----------+---------------+---------+-----------+-----------------+
1 row in set (0.00 sec)
```

6. **Display the train no with increasing order of the fares of class 1**

select ts.train no,ts.farel,t.train_name

from train_status ts,train t

where t.train_no-ts.train_no order by farel asc;

```
mysql> select ts.train_no,ts.fare1,t.train_name
    -> from train_status ts,train t
    -> where t.train_no=ts.train_no
    -> order by fare1 asc;
+----------+-------+---------------+
| train_no | fare1 | train_name    |
+----------+-------+---------------+
|    12711 |   100 | pinakini exp  |
|    12315 |   300 | cormandel exp |
|    12255 |   400 | shatabdhi exp |
+----------+-------+---------------+
3 rows in set (0.00 sec)
```

7. **Display passenger details for train pinakini.**

select p."

from passenger p,train t,ticket te

where tc.train_no=t.train_no and tc.id=p.ticket_id and t.train_name like 'pinakini exp';

```
mysql> select p.*
    -> from passenger p,train t,ticket tc
    -> where tc.train_no=t.train_no and tc.id=p.ticket_id and t.train_name like
    -> 'pinakini exp'
    -> ;
+--------------+--------+------+--------+---------+--------------------+-------------+---------+-----------+
| passenger_id | pnr_no | age  | gender | user_id | reservation_status | seat_number | name    | ticket_id |
+--------------+--------+------+--------+---------+--------------------+-------------+---------+-----------+
|         5001 |  78965 |   45 | M      |    1701 | C                  | B6-45       | ramesh  |      4001 |
|         5003 |  55776 |   54 | M      |    1701 | C                  | B3-22       | mukhesh |      4003 |
+--------------+--------+------+--------+---------+--------------------+-------------+---------+-----------+
2 rows in set (0.00 sec)
```

8. **Display immediate train from tirupathi to Vijayawada**

select distinct t.*

from train t,station s,starts st.stops_at sa

where st.station_no=(select no from station where name like 'tirupathi') and sa.station_no-(select no from station where name like 'vijayawada')

order by date;

```
mysql> select distinct t.*
    -> from train t,station s,starts st,stops_at sa
    -> where st.station_no=(select no from station where name like 'tirupathi')
    ->  and sa.station_no=(select no from station where name like 'vijayawada')
    -> order by date;
+----------+---------------+--------------+----------------+-----------------------+------------+
| train_no | train_name    | arrival_time | departure_time | availability_of_seats | date       |
+----------+---------------+--------------+----------------+-----------------------+------------+
|    12315 | cormandel exp | 12:45:00     | 12:50:00       | N                     | 2017-04-10 |
|    12711 | pinakini exp  | 11:30:00     | 11:40:00       | A                     | 2017-04-10 |
|    12255 | shatabdhi exp | 13:55:00     | 14:00:00       | N                     | 2017-04-11 |
+----------+---------------+--------------+----------------+-----------------------+------------+
3 rows in set (0.01 sec)
```

9. **Display the train no which haults for more time in station no---------**

select train no

from station

having max(hault);

```
mysql>  select train_no
    -> from station
    -> having max(hault);
+----------+
| train_no |
+----------+
|    12711 |
+----------+
1 row in set (0.00 sec)
```

10. Display details of all those passengers whose status is confirmed for train no-

select t.*

from ticket t

where t.status like 'c' and t.train_no=12711;

```
mysql> select t.*
    -> from ticket t
    -> where t.status like 'c' and t.train_no=12711;
+------+---------+--------+-------------------+----------+
| id   | user_id | status | no_of_passengers  | train_no |
+------+---------+--------+-------------------+----------+
| 4001 |    1701 | C      |                 1 |    12711 |
| 4003 |    1701 | C      |                 1 |    12711 |
+------+---------+--------+-------------------+----------+
2 rows in set (0.00 sec)
```