Problem Statement

We have built a train reservation system. The detailed description of the problem is as:

1. Whenever a new user comes for reservation, he needs to register in the database by providing his basic details. If the user already has an existing account then he just needs to login using his email id and his chosen password.

2. A logged in user can book any number of tickets for any source and destination in the desired class. If some seats are available in the specified train for the specified class then a PNR is generated and the total cost is calculated and assigned to a ticket which will be handed to the user later on. For the remaining seats a waiting number is generated for each seat and is displayed to the user.

If a user is an Indian Railways employee, he is asked to provide his id which is verified in the database and if the id is valid the user is given concession according to his department.

If the user doesn’t want any of the seats in waiting list then he is allowed to cancel the booking for the waiting seats and a refund is made to the user.

The ticket provided to the user will have the source and destination name, the departure date and time, the total fare for journey, list of seat numbers and list of waiting numbers (if any).

3. Each train in database has its own available number of seats for different classes (ac1seats, ac2seats, ac3seats, sleeper1seats, and sleeper2seats) and each class has its own base price. Different trains have different base prices for each class. The total fare for a journey is calculated based upon the distance between specified source and destination stations.

4. If a user just wants to access the basic facilities such as checking availability of train between two stations,checking availability of seats or calculating fare for a particular journey then he doesn’t need to login.

5.The user can also enquire about the status of his PNR (without logging in) wherein he is given a list of confirmed seats and a list of waiting seats along with their waiting numbers. Supposing that another user with a waiting number higher up than that of the current user cancels his ticket, then the waiting number of the user is shifted up automatically.

6. An Admin mode is provided for the server side where only authenticated users can login. An authorized user is allowed to see all the database tables and modify these tables such as adding or deleting trains between stations.

Mapping of ER diagram to Relational Model

**User**(uid, fname, lname, phn, dob, password, hno, strtno, city, pin)

**Govtid**(eno, department)

**Train**(trainno, rdate, src, des, ac1seats, ac2seats, ac3seats, sleeper1seats, sleeper2seats, class, baseprice)

**Stations**(stno, stname)

**Intermediate\_stations**(trainno, stno)

**Schedule**(trainno, stnno, arrival\_time, departure\_time)

**Distance**(stno1,stno2, dist)

There may be NULL values present in ac1seats, ac2seats, ac3seats, sleeper1seats, sleeper2seats columns of Ticket relation. Thus we split the Ticket relation in four relations.

**Ticket**(uid, trainno, pnr, src\_u, des\_u, depart\_date, depart\_time, total\_fare)

uid, traino is also a candidate key.

**Seats**\_**requested**( pnr, class, count)

**Seats\_confirmed**(pnr, class, seatno)

**Seats\_waiting**(pnr, class, waitno)

Functional Dependencies

Other than the trivial FDs following fDs are also present:

**1. User**(uid, fname, lname, phn, dob, password, hno, strtno, city, pin)

uid 🡪fname, lname, phn, dob, password, hno, strtno,pin

pin 🡪 city

**2.** **Govtid**(eno, department)

eno 🡪 department

**3. Train**(trainno, rdate, src, des, ac1seats, ac2seats, ac3seats, sleeper1seats, sleeper2seats, class, baseprice)

trainno, rdate 🡪 , src, des, ac1seats, ac2seats, ac3seats, sleeper1seats, sleeper2seats, baseprice, class

trainno, class 🡪baseprice

**4. Stations**(stno, stname)

stno 🡪 stname

**5. Intermediate\_stations**(trainno, stno)

No FD present.

**6. Schedule**(trainno, stnno, arrival\_time, departure\_time)

trainno, stnnno 🡪arrival\_time, departure\_time

**7. Distance**(stno1,stno2, dist)

stno1, stno2 🡪 dist

**8. Ticket**(uid, trainno, pnr, src\_u, des\_u, depart\_date, depart\_time, total\_fare)

pnr 🡪 uid, trainno, src\_u, des\_u, depart\_date, depart\_time, total\_fare

uid, trainno 🡪 src\_u, des\_u, depart\_date, depart\_time, total\_fare

**9. Seats**\_**requested**(, pnr, class, count)

pnr 🡪 class, count

**10. Seats\_confirmed**(pnr, class, seatno)

pnr 🡪 class, seatno

**11. Seats\_waiting**(pnr, class, waitno)

pnr 🡪 class, waitno

Normalization

1NF:

The relation Ticket contained multivalues attributes in the form of seatno and waitno, we have split the Ticket relation in subrelations to resolve this problem.

2NF:

In train relation one of FD is

trainno, class 🡪 price

2NF says that every non-prime attribute should be fuly functional dependent on prime attribute.

But here trainno is prime and class is not.

Thus we need to split the train relation in two relations as:

**Train**(trainno, rdate, src, des, ac1seats, ac2seats, ac3seats, sleeper1seats, sleeper2seats, class, baseprice)

**Fares**(trainno, class, baseprice) **Train**(trainno, rdate, src, des, ac1seats, ac2seats, ac3seats, sleeper1seats, sleeper2seats)

In the user relation we have pin 🡪 city which violates the definition of 2NF. Therefore we split it as:

**User**(uid, fname, lname, phn, dob, password, hno, strtno, city, pin)

**User**(uid, fname, lname, phn, dob, Pins(pin, city)

password, hno, strtno, pin)

3NF:

3NF says that there shouldn’t be any transitive dependency through non-prime attributes.

In other words a relation R is in 3NF if one of following conditions hold:

For each FD of the form a🡪b

1. a🡪 b is trivial.

2. a is a superkey.

3. Each attribute a in (b-a) is a prime attribute.

In all the relations above either condition 1 or condition 2 holds except for train relation which was resolved by the decomposition in 2NF.

Therefore all relations are in 3NF.

Full description of each table:

create table user(

uid int not null auto\_increment primary key,

email varchar(20) unique not null,

dob date,

phn varchar(12),

fname varchar(20),

lname varchar(20),

password varchar(20)

);

CREATE TABLE `station` (

`stno` int(11) NOT NULL AUTO\_INCREMENT,

`stname` varchar(20) DEFAULT NULL,

PRIMARY KEY (`stno`)

) ;

CREATE TABLE `train` (

`trainno` varchar(10) NOT NULL,

`src` int(11) DEFAULT NULL,

`des` int(11) DEFAULT NULL,

`rdate` date,

`ac1seats` int(11) DEFAULT NULL,

`ac2seats` int(11) DEFAULT NULL,

`ac3seats` int(11) DEFAULT NULL,

`sleeper1seats` int(11) DEFAULT NULL,

`sleeper2seats` int(11) DEFAULT NULL,

PRIMARY KEY (`trainno`, `rdate`),

FOREIGN KEY (`src`) REFERENCES `station` (`stno`) ON DELETE CASCADE,

FOREIGN KEY (`des`) REFERENCES `station` (`stno`) ON DELETE CASCADE

) ;

CREATE TABLE `ticket` (

`pnr` varchar(11) NOT NULL,

`uid` int(11) DEFAULT NULL,

`trainno` varchar(10) DEFAULT NULL,

`src\_u` int(11) DEFAULT NULL,

`des\_u` int(11) DEFAULT NULL,

`totalfare` int(11) DEFAULT NULL,

`depart\_date` date DEFAULT NULL,

`depart\_time` time DEFAULT NULL,

PRIMARY KEY (`pnr`),

FOREIGN KEY (`uid`) REFERENCES `user` (`uid`) ON DELETE CASCADE,

FOREIGN KEY (`trainno`) REFERENCES `train` (`trainno`) ON DELETE CASCADE,

FOREIGN KEY (`src\_u`) REFERENCES `station` (`stno`) ON DELETE CASCADE,

FOREIGN KEY (`des\_u`) REFERENCES `station` (`stno`) ON DELETE CASCADE

);

CREATE TABLE `distance` (

`stno1` int(11) NOT NULL DEFAULT '0',

`stno2` int(11) NOT NULL DEFAULT '0',

`dist` float DEFAULT NULL,

PRIMARY KEY (`stno1`,`stno2`),

FOREIGN KEY (`stno1`) REFERENCES `station` (`stno`) ON DELETE CASCADE,

FOREIGN KEY (`stno2`) REFERENCES `station` (`stno`) ON DELETE CASCADE

) ;

CREATE TABLE `fares` (

`trainno` varchar(10) NOT NULL DEFAULT '',

`class` varchar(20) NOT NULL,

`price` int(11) DEFAULT NULL,

PRIMARY KEY (`trainno`,`class`),

FOREIGN KEY (`trainno`) REFERENCES `train` (`trainno`) ON DELETE CASCADE

) ;

CREATE TABLE `intermediate\_stations` (

`trainno` varchar(10) NOT NULL DEFAULT '',

`stno` int(11) NOT NULL DEFAULT '0',

PRIMARY KEY (`trainno`,`stno`),

FOREIGN KEY (`stno`) REFERENCES `station` (`stno`) ON DELETE CASCADE,

FOREIGN KEY (`trainno`) REFERENCES `train` (`trainno`) ON DELETE CASCADE

) ;

CREATE TABLE `schedule` (

`trainno` varchar(10) NOT NULL DEFAULT '',

`stno` int(11) NOT NULL DEFAULT '0',

`arrivaltime` time DEFAULT NULL,

`departuretime` time DEFAULT NULL,

PRIMARY KEY (`trainno`,`stno`),

FOREIGN KEY (`trainno`) REFERENCES `train` (`trainno`) ON DELETE CASCADE,

FOREIGN KEY (`stno`) REFERENCES `station` (`stno`) ON DELETE CASCADE

);

CREATE TABLE `seats\_confirmed` (

`class` varchar(20) NOT NULL DEFAULT '',

`pnr` varchar(11) NOT NULL DEFAULT '',

`trainno` varchar(10) DEFAULT NULL,

`seats\_no` varchar(5) NOT NULL DEFAULT '',

PRIMARY KEY (`class`,`pnr`,`seats\_no`),

FOREIGN KEY (`pnr`) REFERENCES `ticket` (`pnr`) ON DELETE CASCADE,

FOREIGN KEY (`trainno`) REFERENCES `train` (`trainno`) ON DELETE CASCADE

) ;

CREATE TABLE `seats\_required` (

`class` varchar(20) NOT NULL DEFAULT '',

`pnr` varchar(11) NOT NULL DEFAULT '',

`trainno` varchar(10) DEFAULT NULL,

`count` int(11) DEFAULT NULL,

PRIMARY KEY (`class`,`pnr`),

FOREIGN KEY (`pnr`) REFERENCES `ticket` (`pnr`) ON DELETE CASCADE,

FOREIGN KEY (`trainno`) REFERENCES `train` (`trainno`) ON DELETE CASCADE

) ;

CREATE TABLE `seats\_waiting` (

`class` varchar(20) NOT NULL DEFAULT '',

`pnr` varchar(11) NOT NULL DEFAULT '',

`trainno` varchar(10) DEFAULT NULL,

`waitno` int(11) NOT NULL DEFAULT '0',

PRIMARY KEY (`class`,`pnr`,`waitno`),

FOREIGN KEY (`pnr`) REFERENCES `ticket` (`pnr`) ON DELETE CASCADE,

FOREIGN KEY (`trainno`) REFERENCES `train` (`trainno`) ON DELETE CASCADE

) ;