NATIONAL INSTITUE OF TECHNOLOGY WARANGAL

DATABASE MANAGEMENT SYSTEM

PROJECT

ON

RAILWAY DATABASE MANAGEMENT

(Group Project)

Group Members:
Belide Sai Teja (201106)
M. Harshitha (205131)

Contents:

Abstract:

The Railway Reservation System which enables the passenger to enquire the details of availability of trains on the basis of starting station of the journey to the final destination, booking and cancellation of tickets, enquiring the status of booked tickets, etc. The aim of this case is to design and a database maintaining the records of different trains, train status and passengers.

This project depicts and contains an Introduction to Railway Reservation System. Now-a-days, Online reservation has made the Reservation process smoother and faster than ever before. It is mainly used for long routes. It is a computerized system of booking or reserving seats of a train well in advanced.

This project contains Entity Relationship Model Diagram on Railway Reservation System and Introduction to Relational Model. On further it is also a database design of Railway Reservation System based on Relational model. Some of the examples of retrieving data from database by the support of Structured Query Language (SQL).

Introduction :

Database is an organized collection of similar/related data. This data is organized in such a way that it supports passengers requiring information. Database Management System (DBMS) which is used to manage database which is in the form of tables, views, schemeas, etc. It makes possible the end users to create, read, update, and delete data in the database. It helps in registering and maintaining users, enforcing data security, monitoring performance, maintaining data integrity, dealing with concurrency control and recovering information corrupted by unexpected failure. It can alter 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 the data.

The main purpose of maintaining the database in Railway Reservation System is to reduce the human errors in handling the data while booking and cancelling the tickets and making convenient for the customers and providers to maintain the data about customers and availability of seats. The loopholes which arise due to manual working over the data can be removed. It can enable the fast accessing of the data from the database. For the future expansion this system can be web enabled so that the users can make various enquires about the availability of trains and seats availability. This database includes passenger details, availability of seats in trains, number of trains and their details.

Description of the project:

This project is about creating the database about Railway Reservation System.

The Railway Reservation System facilitates the passengers to enquire about the trains available on the basis of start and final destination of journey, booking and cancellation of tickets, etc. The aim of this project is to maintain the records of trains, train status, and passengers. The record of trains includes Train number, Train name, availability of seats, wi-fi and food facilities, and running status on different days of the week.

The passengers can book their tickets for the train in which seats are available based on the preferred journey. For this passenger needs to provide the desired train name, date of journey for which seat is to be booked. Before booking, train number and date of journey is validated, whether there is availability of seats or not. If so, available the tickets are booked with confirm status and corresponding ticket ID is generated and with all the details of the passenger. At any instant, if ticket is to be cancelled then Passenger ID (which is unique) has to be provided and deleted from the database. After this the ticket with waiting status is confirmed.

Since the Reservation System is huge it is not so easy and feasible to develop a case study. So that a list of assumptions is made and documented at that level. In order to make things simple, a sample data is preferred and used for working of the Reservation System. For the implementation of sample case study, some of the assumptions are made and listed below;

- The number of trains has been restricted to 4
- Category of tickets available are sleeper, First class, Second class, and Third class.
- In-between stoppage stations are not considered.
- List of trains and list of passengers are to be maintained.

In the booking procedure the train number, Date of journey and seat type is read from the passenger ticket. On the basis of the data provided by the passenger, the seats are checked and if yes, then the ticket is issued with a unique Ticket number. If no, then it will be pushed into waiting list. If the passenger cancels the ticket, the ticket number is checked and the respective details of the passenger are deleted and the first in the waiting list gets the ticket confirmed.

Entities and their Attributes:

The entities and Attributes of the database are as follows:

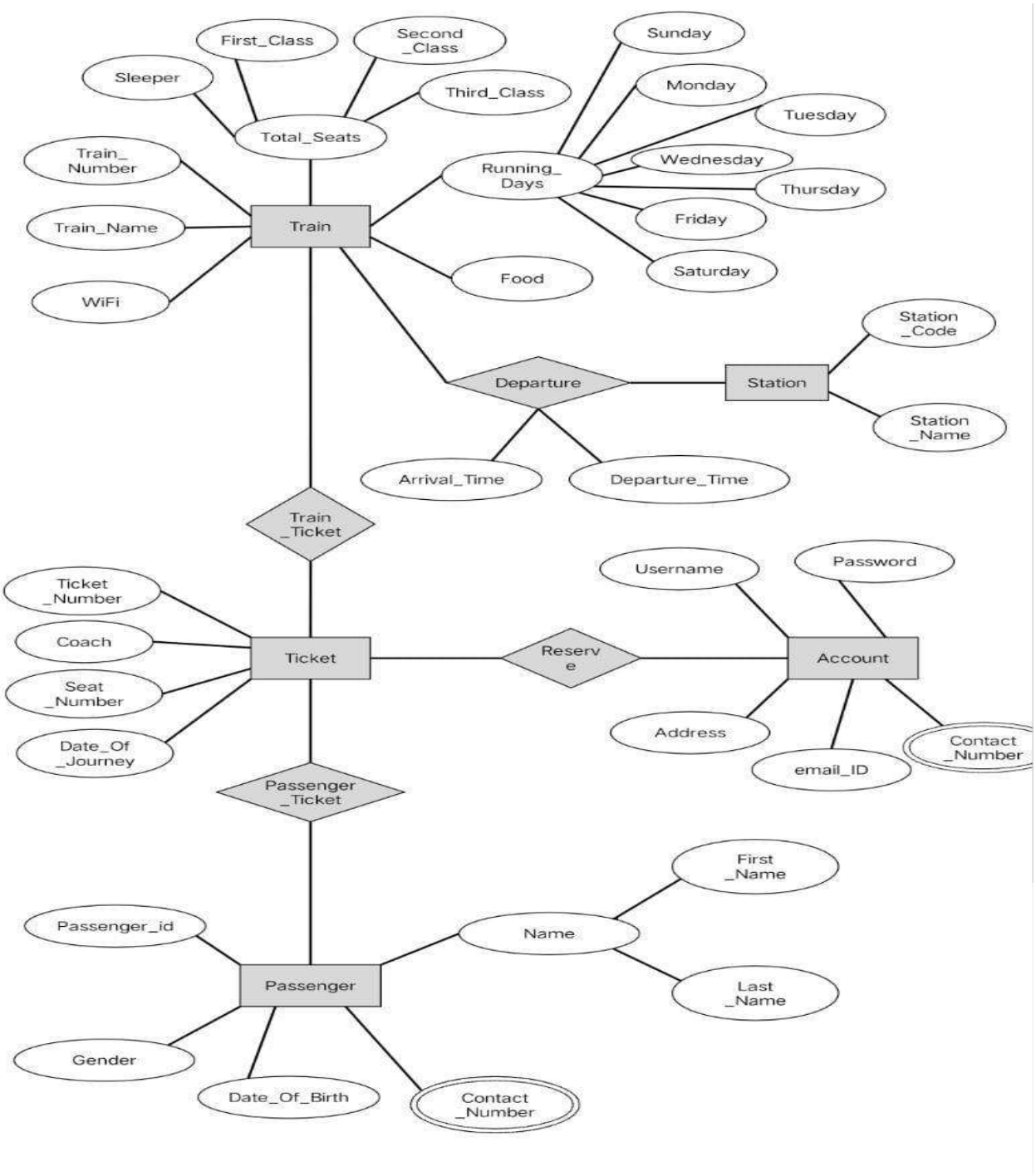
```
Account
(
Username varchar(15) NOT NULL,
        Password varchar(20) NOT NULL,
        email_ID varchar(35) NOT NULL,
        Address varchar(50) DEFAULT NULL,
        PRIMARY KEY(Username)
)
```

```
Contact
(
        Username varchar(15) NOT NULL DEFAULT ,
        Contact_Number char(10) NOT NULL DEFAULT ,
        PRIMARY KEY(Username,Contact_Number)
)
Train
(
        Train_Number char(6) NOT NULL DEFAULT '0',
        Train_Name varchar(25) NOT NULL,
        Seat_Sleeper char(4) NOT NULL,
        Seat_First_Class char(4) NOT NULL,
        Seat_Second_Class char(4) NOT NULL,
        Seat_Third_Class char(4) NOT NULL,
        WiFi char(3) NOT NULL,
        Food char(3) NOT NULL,
        Run_On_Sunday char(3) NOT NULL,
        Run_On_Monday char(3) NOT NULL,
        Run_On_Tuesday char(3) NOT NULL,
        Run_On_Wednesday char(3) NOT NULL,
        Run_On_Thursday char(3) NOT NULL,
        Run_On_Friday char(3) NOT NULL,
```

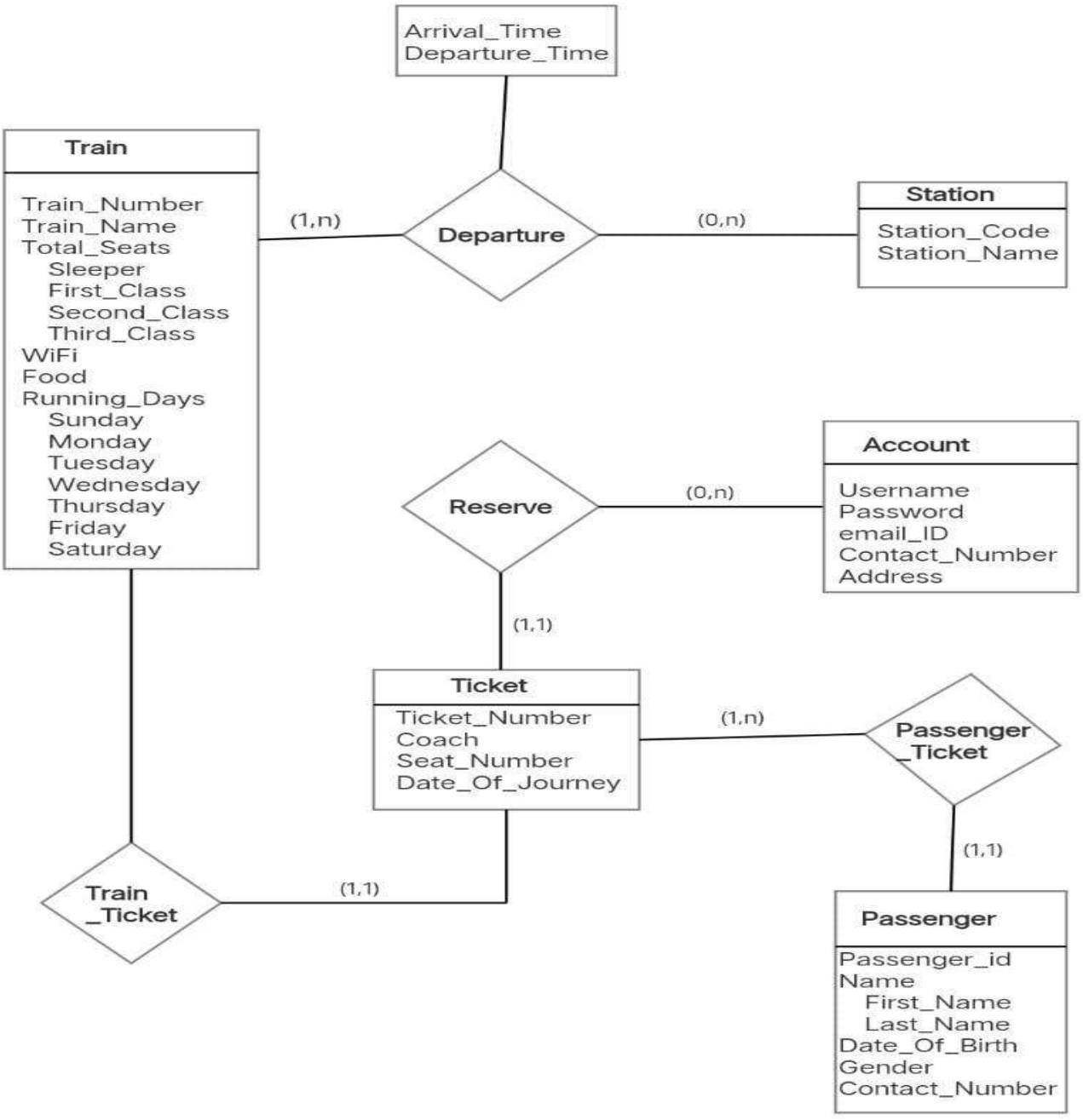
```
        Run_On_Saturday char(3) NOT NULL,
        PRIMARY KEY (Train_Number)
)
Ticket
(
        Ticket_Number char(10) NOT NULL,
        Train_Number char(6) NOT NULL,
        Date_Of_Journey DATE NOT NULL,
        Username varchar(15) NOT NULL,
        PRIMARY KEY (Ticket_Number),
        KEY Username(Username),
        KEY Train_Number(Train_Number)
)
Passenger
(
        Passenger_id varchar(12) NOT NULL,
        First_Name varchar(15) NOT NULL,
        Last_Name varchar(15) NOT NULL,
        Date_Of_Birth DATE NOT NULL,
        Gender varchar(1) NOT NULL,
        Contact_Number char(10) DEFAULT NULL,
        Ticket_Number char(10) NULL,
```

```
        Class varchar(10) NOT NULL,
        PRIMARY KEY (Passenger_id),
        KEY Ticket_Number (Ticket_Number)
)
Station
(
        Station_Code varchar(5) NOT NULL ,
        Station_Name varchar(25) NOT NULL,
        PRIMARY KEY (Station_Code)
)
Departure
(
        Station_Code varchar(5) NOT NULL,
        Train_Number char(6) NOT NULL,
        Arrival_Time TIME DEFAULT NULL,
        Departure_Time TIME DEFAULT NULL,
        PRIMARY KEY (Station_Code,Train_Number),
        KEY Station_Code (Station_Code)
)
```

ER DIAGRAM:


First_Class
Second _Class
Sleeper
Third_Class
Total_Seats
Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Running_ Days
Train_ Number
Train_Name
WiFi
Train
Food
Station _Code
Station _Name
Station
Departure
Arrival_Time
Departure_Time
Train _Ticket
Ticket _Number
Coach
Seat _Number
Date_Of _Journey
Ticket
Reserv e
Username
Password
Account
Address
email_ID
Contact _Number
Passenger _Ticket
First _Name
Last _Name
Name
Passenger_id
Passenger
Gender
Date_Of_Birth
Contact _Number

Relational Schema with Normalised Tables:



Functional Dependencies and Normalisation:

Train:

Train_Number -> (‘Train_Name’, ’Seat_Sleeper’, ‘Seat_First_Class’, ‘Seat_Second_Class’, ‘Seat_Third_Class’, ‘WiFi’, ‘Food’, ‘Run_On_Sunday’, ‘Run_On_Monday’, ‘Run_On_Tuesday’, ‘Run_On_Wednesday’, ‘Run_On_Thursday’, ‘Run_On_Friday’, ‘Run_On_Saturday’

Departure:

(‘Station_Code’, ‘Train_Number’) -> (‘Arrival_Time’, ‘Departure_Time’)

Station:

‘Station_Code’ -> (‘Station_Name’)

Account:

‘Username’ -> (‘Password’, ‘email_ID’, ‘Address’)

Ticket:

‘Ticket_Number’ -> (‘Coach’, ‘Seat_Number’, ‘Date_Of_Journey’, ‘Username’)

Passenger:

‘Passenger_id’ -> (‘Username’, ‘First_Name’, ‘Last_Name’, ‘Date_Of_Birth’, ‘Gender’, ‘Ticket_Number’, ‘Train_Number’)

Contact:

‘Username’ -> (‘Contact_Number’)

FIRST NORMAL FORM:

If every attribute in a relation is single-valued, then relation is said to be in First Normal Form.

The above schema is in First Normal Form as all the attributes are single valued.

Since a passenger can have multiple contact numbers, it violates the First Normal Form. To handle this a separate relation is made to store contact numbers of Passenger.

SECOND NORMAL FORM:

A relation to be in Second Normal Form,

- Relation must be in First Normal Form
- Relation has No Partial Dependency(i.e., no prime attribute is dependent on any proper subset of Candidate Key of the relation

In Passenger table, if we consider Ticket_Number and First_Name as Candidate Key, then the relation violates Second Normal Form as Date of Birth depend only on First_Name. Hence the relation is in Second Normal Form.

THIRD NORMAL FORM:

A relation to be in Third Normal Form,

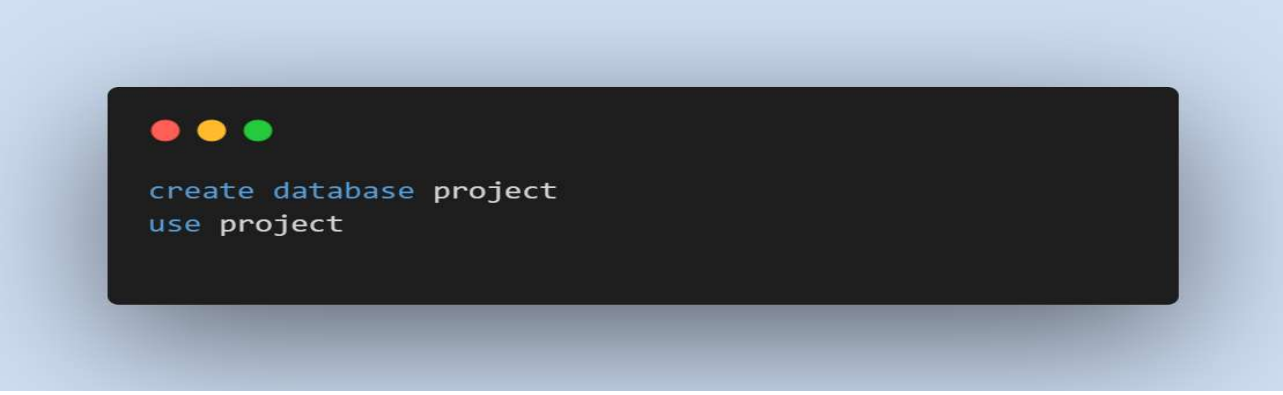
- Relation must be in Second Normal Form

- No non-prime attribute must define another non-prime attribute

The above schema satisfies all the conditions; hence it is in Third Normal Form.

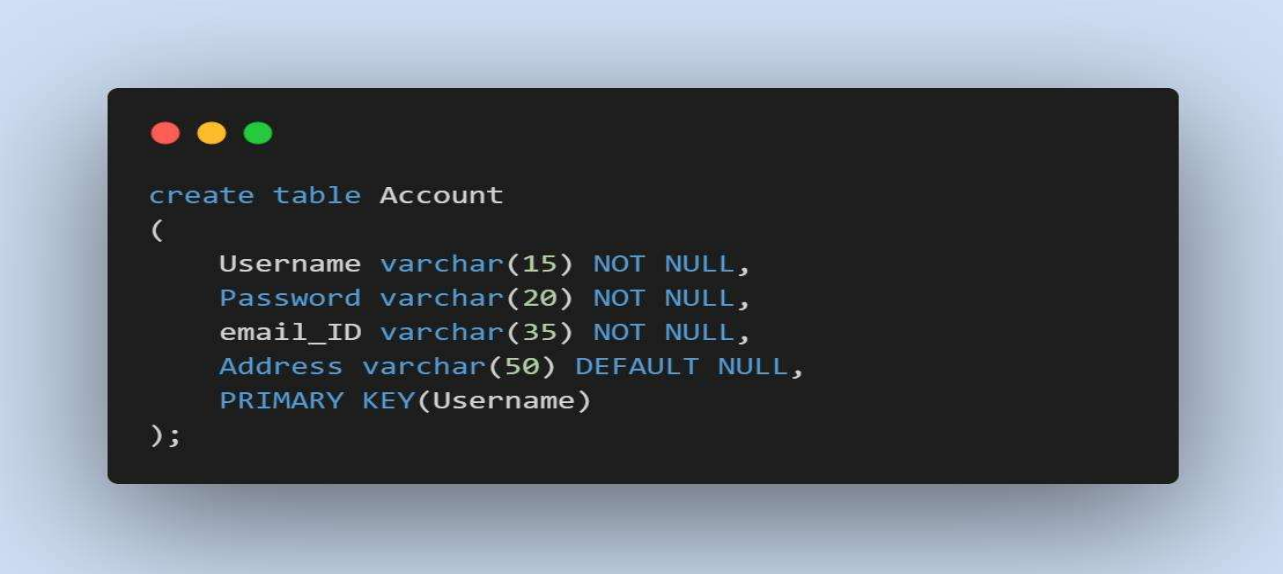
SQL CODE:

/* Create Database */



```
create database project
use project
```

/* Create all the Tables */



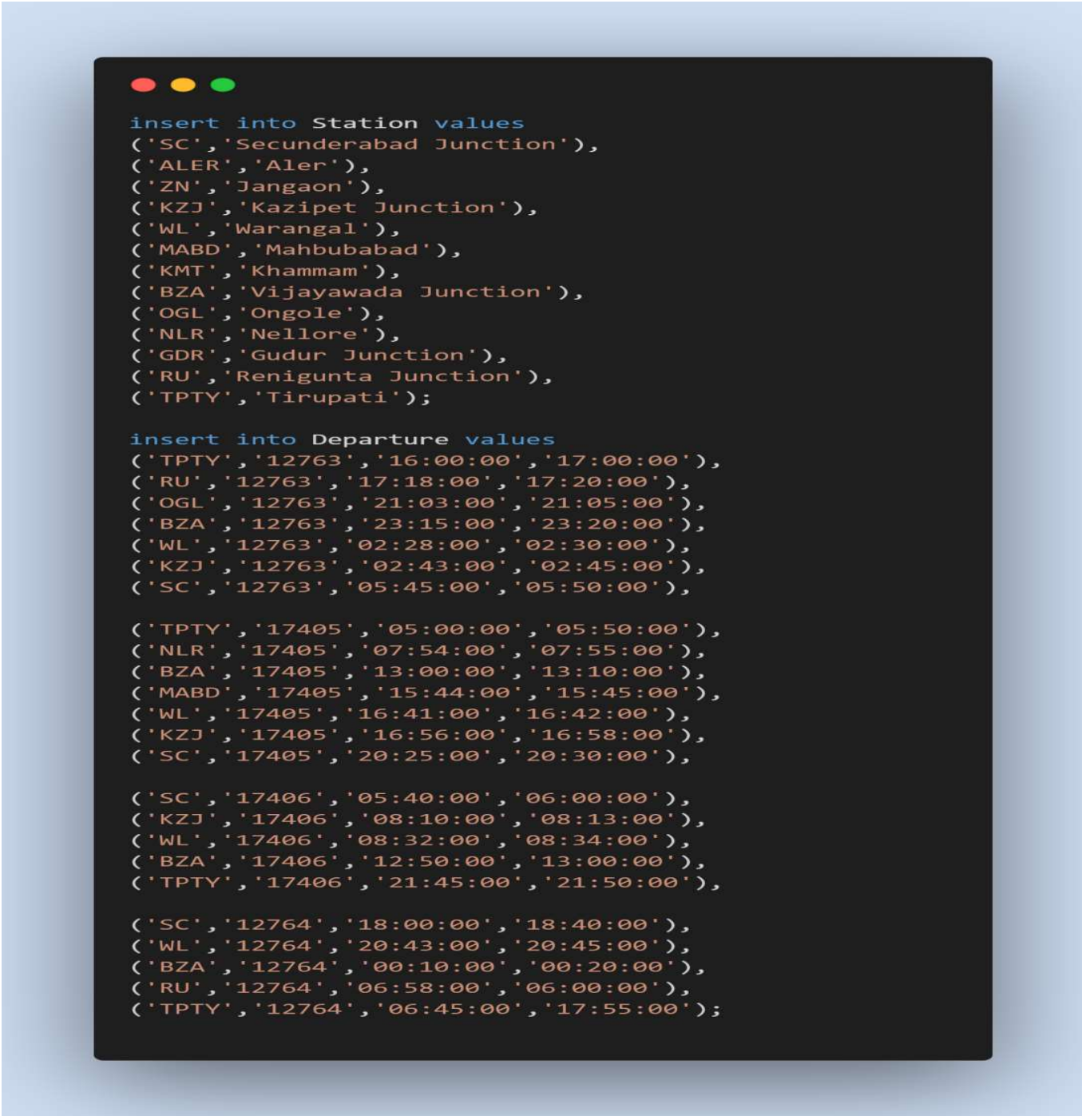
```
create table Account
(
    Username varchar(15) NOT NULL,
    Password varchar(20) NOT NULL,
    email_ID varchar(35) NOT NULL,
    Address varchar(50) DEFAULT NULL,
    PRIMARY KEY(Username)
);
```

```sql
create table Contact
(
    Username varchar(15) NOT NULL DEFAULT '',
    Contact_Number char(10) NOT NULL DEFAULT '',
    PRIMARY KEY(Username,Contact_Number),
    CONSTRAINT Contact_ibfk_1 FOREIGN KEY (Username)
REFERENCES Account(Username) ON DELETE CASCADE
);

create table Train
(
    Train_Number char(6) NOT NULL DEFAULT '0',
    Train_Name varchar(25) NOT NULL,
    Seat_Sleeper char(4) NOT NULL,
    Seat_First_Class char(4) NOT NULL,
    Seat_Second_Class char(4) NOT NULL,
    Seat_Third_Class char(4) NOT NULL,
    WiFi char(3) NOT NULL,
    Food char(3) NOT NULL,
    Run_On_Sunday char(3) NOT NULL,
    Run_On_Monday char(3) NOT NULL,
    Run_On_Tuesday char(3) NOT NULL,
    Run_On_Wednesday char(3) NOT NULL,
    Run_On_Thursday char(3) NOT NULL,
    Run_On_Friday char(3) NOT NULL,
    Run_On_Saturday char(3) NOT NULL,
    PRIMARY KEY (Train_Number)
);

create table Ticket
(
    Ticket_Number char(10) NOT NULL,
    Train_Number char(6) NOT NULL,
    Date_Of_Journey DATE NOT NULL,
    Username varchar(15) NOT NULL
    PRIMARY KEY (Ticket_Number),
    CONSTRAINT Ticket_ibfk_1 FOREIGN KEY (Username)
REFERENCES Account(Username) ON DELETE CASCADE,
    CONSTRAINT Ticket_ibfk_2 FOREIGN KEY
 (Train_Number) REFERENCES Train(Train_Number) ON
UPDATE CASCADE
);
```

```sql
create table Passenger
(
    Passenger_id varchar(12) NOT NULL,
    First_Name varchar(15) NOT NULL,
    Last_Name varchar(15) NOT NULL,
    Date_Of_Birth DATE NOT NULL,
    Gender varchar(1) NOT NULL,
    Contact_Number char(10) DEFAULT NULL,
    Ticket_Number char(10) NULL,
    Class varchar(10) NOT NULL,
    PRIMARY KEY (Passenger_id),
    CONSTRAINT Passenger_ibfk_1 FOREIGN KEY
 (Ticket_Number) REFERENCES Ticket(Ticket_Number)
ON DELETE CASCADE
);

create table Station
(
    Station_Code varchar(5) NOT NULL DEFAULT '',
    Station_Name varchar(25) NOT NULL,
    PRIMARY KEY (Station_Code)
);


create table Departure
(
    Station_Code varchar(5) NOT NULL DEFAULT '',
    Train_Number char(6) NOT NULL DEFAULT '0',
    Arrival_Time TIME DEFAULT NULL,
    Departure_Time TIME DEFAULT NULL,
    PRIMARY KEY (Station_Code,Train_Number),
    CONSTRAINT Departure_ibfk_1 FOREIGN KEY
 (Train_Number) REFERENCES Train (Train_Number)
ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT Departure_ibfk_2 FOREIGN KEY
 (Station_Code) REFERENCES Station (Station_Code)
ON DELETE CASCADE ON UPDATE CASCADE
);
```

/* Insert values into the Table */

```
insert into Account values
('akhil55','agfndjgkynsb','akhil@gmail.com',
'Airport Road,Shamshabad'),
('vikky24','ehgjdhnmjh','vikky@gmail.com',
'NGOs Colony,Warangal'),
('aasiya41','fdhsfnkgnk','aasiya@gmail.com',
'HBS Colony,Warangal'),
('prathima33','fghsgnnbg','prathima@gmail.com',
'VNR Colony,Hyderabad'),
('rajesh12','kghjgfbfh8nn','rajesh@gmail.com',
'SVS Mall Road,Chennai'),
('gokul','rebjbmmss','gokul@gmail.com',
'Uppal Hyderabad');

insert into Contact values
('aasiya41','1234567890'),
('akhil55','2345678901'),
('gokul','3456789012'),
('prathima33','4567890123'),
('rajesh12','5678901234'),
('vikky24','6789012345');


insert into Train values
('17406','Krishna Express','432','49','98','185','No',
'Yes',
'Yes','Yes','Yes','Yes','Yes','Yes','Yes'),
('17405','Krishna Express','432','49','98','185','No',
'Yes',
'Yes','Yes','Yes','Yes','Yes','Yes','Yes'),
('12764','Padmavathi SF Express','425','47','95','194'
,'No','No',
'Yes','Yes','Yes','No','Yes','Yes','No'),
('12763','Padmavathi SF Express','425','47','95','194'
,'No','No',
'Yes','Yes','Yes','No','Yes','Yes','No');
```



```
insert into Station values
('SC','Secunderabad Junction'),
('ALER','Aler'),
('ZN','Jangaon'),
('KZJ','Kazipet Junction'),
('WL','Warangal'),
('MABD','Mahbubabad'),
('KMT','Khammam'),
('BZA','Vijayawada Junction'),
('OGL','Ongole'),
('NLR','Nellore'),
('GDR','Gudur Junction'),
('RU','Renigunta Junction'),
('TPTY','Tirupati');

insert into Departure values
('TPTY','12763','16:00:00','17:00:00'),
('RU','12763','17:18:00','17:20:00'),
('OGL','12763','21:03:00','21:05:00'),
('BZA','12763','23:15:00','23:20:00'),
('WL','12763','02:28:00','02:30:00'),
('KZJ','12763','02:43:00','02:45:00'),
('SC','12763','05:45:00','05:50:00'),

('TPTY','17405','05:00:00','05:50:00'),
('NLR','17405','07:54:00','07:55:00'),
('BZA','17405','13:00:00','13:10:00'),
('MABD','17405','15:44:00','15:45:00'),
('WL','17405','16:41:00','16:42:00'),
('KZJ','17405','16:56:00','16:58:00'),
('SC','17405','20:25:00','20:30:00'),

('SC','17406','05:40:00','06:00:00'),
('KZJ','17406','08:10:00','08:13:00'),
('WL','17406','08:32:00','08:34:00'),
('BZA','17406','12:50:00','13:00:00'),
('TPTY','17406','21:45:00','21:50:00'),

('SC','12764','18:00:00','18:40:00'),
('WL','12764','20:43:00','20:45:00'),
('BZA','12764','00:10:00','00:20:00'),
('RU','12764','06:58:00','06:00:00'),
('TPTY','12764','06:45:00','17:55:00');
```

/* delimeter */

```
delimeter //
create trigger cancellation
  before delete on ticket
  for each row
BEGIN
  set @TrainNumber=old.Train_Number;
  set @TicketNumber=old.Ticket_Number;
  set @Class=(select p.class from Passenger p where
 p.Ticket_Number=@ticketNumber);
  if @Class='First' THEN
    UPDATE Train set Seat_First_Class=Seat_First_Class+1 where
 Train_Number=@TrainNumber;
  elseif @class='Sleeper' then
    UPDATE Train set Seat_Sleeper = Seat_Sleeper+1 WHERE
 Train_Number = @TrainNumber;
  elseif @class='Second' then
    UPDATE Train set Seat_Second_Class = Seat_Second_Class+1 WHERE
 Train_Number = @TrainNumber ;
  elseif @class='Third' then
    UPDATE Train set Third_Class = Seat_Third_Class+1 WHERE
 Train_Number = @TrainNumber ;
  end if;
END //
delimeter;
```

Conclusion and Future Scope:

This system can give information of any train, find trains between stations, book and cancel tickets. Some of the additional features like booking meals on train, et cetera can be included. Also payment gateways can be included for secure transactions.