PROJECT REPORT

On

Railway Station Database

COMPUTER SCIENCE AND ENGINEERING

B.E. Batch-2020

Submitted By:

Under the Guidance of:

Dr. Isha Kansal

(Mentor)

Jaskirat kaur

Roll No. 2010993547

DECLARATION

I hereby declare that the project work entitled **"Railway Station Database"** submitted to the Lovely Professional University, is a record of an original work done by me under the guidance of **Dr. Isha Kansal**, Assistant Professor Dept of Computer Science & Engineering, Lovely Professional University, and this project work is submitted in the partial fulfilment of the requirements for the award of the degree of Bachelor's of Technology in Computer Science & Engineering. The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Date: 13/11/2022

Name: Jaskirat kaur
Roll no: 2010993547

Acknowledgement

The successful presentation of the **RAILWAY STATION DATABASE** would be incomplete without the mention of the people who made it possible and whose constant guidance crowned my effort with success.

It is great pleasure for me to express my gratitude to our honorable Chancellor**,** for giving the opportunity and platform with facilities in accomplishing the project based laboratory report.

I express the sincere gratitude to our Pro Vice Chancellor & Head of Faculty**,** for his administration towards our academic growth.

I express sincere gratitude to our Head of School**,** for his leadership and constant motivation provided in successful completion of our academic semester.

I record it as my privilege to deeply thank our HOD CSE Dept. **,** for providing us the efficient faculty and facilities to make our ideas into reality.

I express my sincere thanks to our Project Guide, Assistant Professor, Department of Computer Science and Engineering, Lovely Professional University, Phagwara, for her novel association of ideas, encouragement, appreciation and intellectual zeal which motivated us to venture this project successfully.

Finally, it's pleased to acknowledge the indebtedness to all those who devoted themselves directly or indirectly to make this project report success.

Table of Contents

ABSTRACT

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 Railway Station Database. It is the computerized system of reserving the seats of train as well as checking the details of trains passing through that station. 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 system and introduction to relation model. There is also design of the database of the railway station database based on relation model. Example of some SQL queries to retrieve data from rail management database.

INTRODUCTION

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 maintain database for Railway Station is to reduce the manual errors involved in the booking and cancelling of tickets, searching for the trains and make it convenient for the customers and provides 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 database which includes customer details, availability of seats in trains, no. of trains and their details.

Technology Learnt

- **ORACLE** : Oracle database is a relational database management system. It is also called **OracleDB**, or simply **Oracle.** It is produced and marketed by **Oracle Corporation.** It was created in **1977** by **Lawrence Ellison** and other engineers. It is one of the most popular relational database engines in the IT market for storing, organizing, and retrieving data.

- **SQL** : SQL stands for Structured Query Language. SQL is used to communicate with a database. According to ANSI (American National Standards Institute), it is the standard language for relational database management systems.

  SQL statements are used to perform tasks such as update data on a database, or retrieve data from a database. Some common relational database management systems that use SQL are: Oracle, Sybase, Microsoft SQL Server, Microsoft Access, Ingres, etc.

  Although most database systems use SQL, most of them also have their own additional proprietary extensions that are usually only used on their system. However, the standard SQL commands such as "Select", "Insert", "Update", "Delete", "Create", and "Drop" can be used to accomplish almost everything that one needs to do with a database.

- **PL/SQL** : PL/SQL is a block structured language that enables developers to combine the power of SQL with procedural statements. All the statements of a block are passed to oracle engine all at once which increases processing speed and decreases the traffic.

  PL/SQL is basically a procedural language, which provides the functionality of decision making, iteration and many more features of procedural programming languages.

Problem Statement – Railway System

The railway network of our country is one of the most complex public establishments. You can design a database solution for this network and make the management of the same more natural. Your system should have the following pieces of information:

- **Station Names**

  Tracks that connect those stations (to keep things simple, you can assume that only one track runs between two stations) Train IDs with names.

- **Schedules of the trains**

  The train schedules should have information on the stations from where the train starts and by when it reaches the destination. It should also include information on which stations it passes through during its journey.

  To keep things simple, you can assume that every train completes its journey within a day, and they run daily. However, you'll also need to store information on the sequence of the stations a train passes through. For example, if a train starts from Delhi and goes to Kolkata through Lucknow, then you'll need to add the arrival and departure times of the train for all these stations. Keeping the stations in sequence will allow easy management of trains and their data.

  Till here, the project is rather easy. You can make it more challenging by adding the passenger's information of every train such as its coaches, seat numbers, types of coaches, passenger names, and so on. This project might take some time to complete, but it'll help you showcase your knowledge of database management solutions while solving a significant issue of a public authority.

Entity Relationship(ER) diagram

for

Railway Station Database

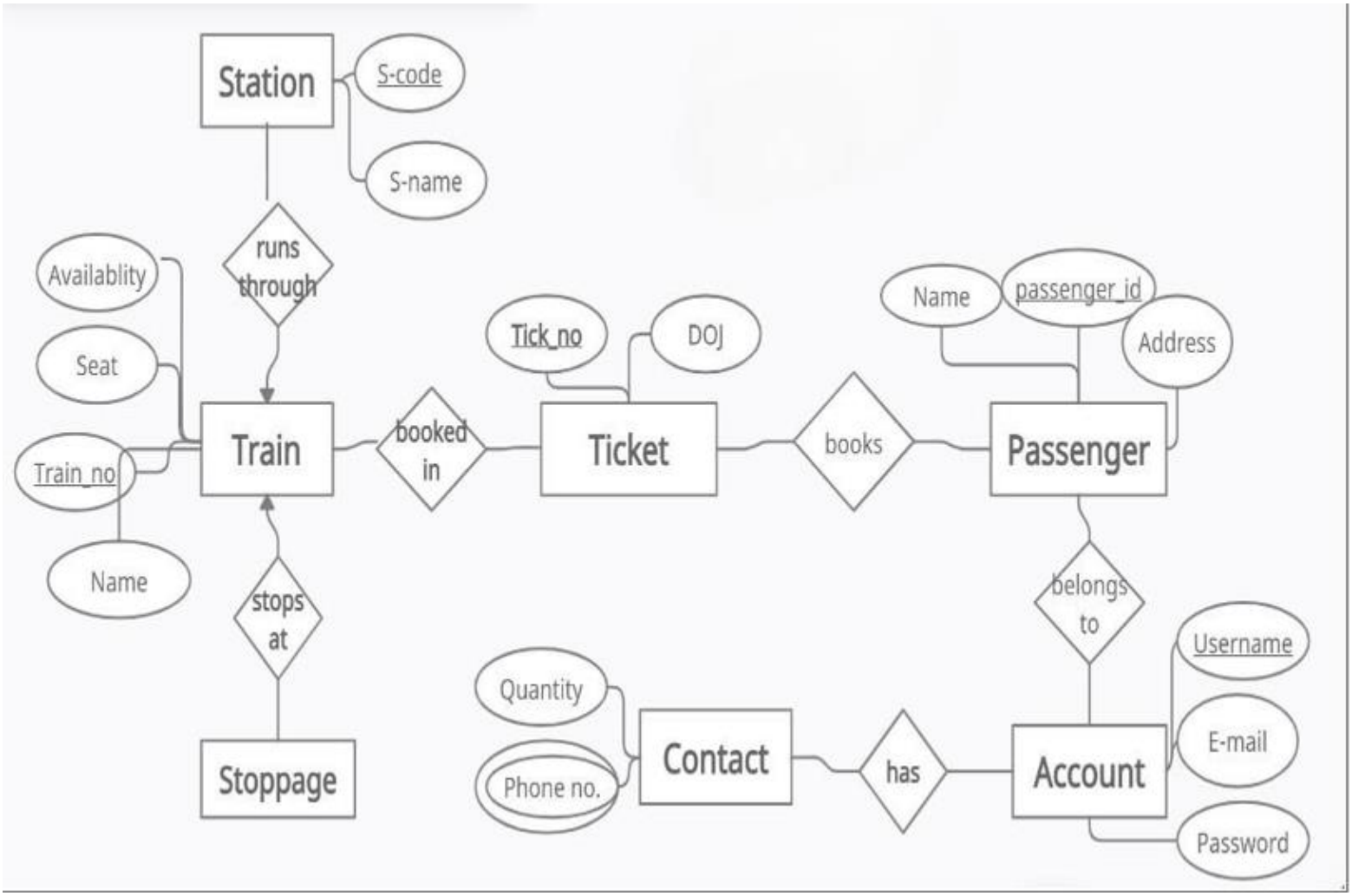

Table Description

- **Passenger :** This table stores all the information about the passengers who are booking their tickets and travel in trains. The details include Passenger_id, First_name, Last_name, Age, Gender, Ticket_no, Phone_no & class. ***Constraint:*** *Passenger_id will be unique for each passenger.*

- **Account :** This table consists of details about the user like Username, Password, E-mail and Address.
  Constraint : *Username will be unique for each user.*

- **Contact :** This table stores the contact details of the users who are booking the tickets. The details include Username and phone number.

- **Ticket :** This table will hold the data about the tickets which are being booked by the user. The details include Ticket number, Train number, Date of journey and Username.
  Constraint : Ticket number will be unique for each ticket.

- **Train :** This table consists of details about the trains in which the passenger is booking his/her ticket. The details which this table holds are Train number, Train name, Seats and Availability.
  Constraint : *Train number will be unique for each trains.*

- **Station :** This table consists of the data about the stations the train will cross during its journey. The details consists of Station Code and Name.
  Constraints : *Station code will be unique for each stations.*

Normalization

First normal Form

As per the rule of first normal form, an attribute (column) of a table cannot hold multiple values. It should hold only atomic values.

The above schema is in 1NF since all the attributes are atomic and not multivalued.

Since a passenger could have multiple phone numbers, it would violate the 1NF rules. Hence we have created a separate table called contact to handle this.

Second Normal Form

A table is said to be in 2NF if both the following conditions hold:

- Table is in 1NF (First normal form)

-No non-prime attribute is dependent on the proper subset of any candidate key of table.

If in Passenger table we consider ticket_no and first_name as the candidate key, then date_of_birth would depend only on the name and it would violate the 2NF.

Third Normal Form

A table design is said to be in 3NF if both the following conditions hold:

-Table must be in 2NF

-Transitive functional dependency of non-prime attribute on any super key should be removed.

Our schema follows the above rules and hence is in 3NF.

Creating 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)
  );

 CREATE TABLE Contact (
   Username varchar(15) NOT NULL,
   Phone_No char(10) NOT NULL,
   PRIMARY KEY (Username,Phone_No),
   FOREIGN KEY (Username) REFERENCES Account (Username)
 );

CREATE TABLE Passenger(
  Passenger_Id number(11) NOT NULL,
  First_Name varchar(20) NOT NULL,
  Last_Name varchar(20) NOT NULL,
  Gender char(1) NOT NULL,
  Phone_No char(10) DEFAULT NULL,
  Ticket_No number(10) NOT NULL,
  Age number(11) NOT NULL,
  Class varchar(20) NOT NULL,
  PRIMARY KEY (Passenger_Id),
  FOREIGN KEY (Ticket_No) REFERENCES Ticket(Ticket_No)
);

CREATE TABLE Ticket (
  Ticket_No number(10) NOT NULL,
  Train_No number(6) NOT NULL,
  Date_of_Journey varchar(20) NOT NULL,
  Username varchar(15) NOT NULL,
  PRIMARY KEY (Ticket_No),
```

```
  FOREIGN KEY (Username) REFERENCES Account (Username),
  FOREIGN KEY (Train_No) REFERENCES Train (Train_No)
 );

CREATE TABLE Train (
  Train_No number(6) NOT NULL,
  Name varchar(25) NOT NULL,
  Seat_Sleeper number(4) NOT NULL,
  Seat_First_Class_AC number(4) NOT NULL,
  Seat_Second_Class_AC number(4) NOT NULL,        Seat_Third_Class_AC
number(4) NOT NULL,
  Wifi char(1) NOT NULL,
  Food char(1) NOT NULL,
  Run_On_Sunday char(1) NOT NULL,
  Run_On_Monday char(1) NOT NULL,
  Run_On_Tuesday char(1) NOT NULL,
  Run_On_Wednesday char(1) NOT NULL,
  Run_On_Thursday char(1) NOT NULL,
  Run_On_Friday char(1) NOT NULL,
  Run_On_Saturday char(1) NOT NULL,
  PRIMARY KEY (Train_No)
);




CREATE TABLE Station(
  Station_Code char(5) NOT NULL,
  Station_Name varchar(25) NOT NULL,
  PRIMARY KEY (Station_Code)
 );

CREATE TABLE Stoppage(
```

Train_No number(6) NOT NULL,

Station_Code char(5) NOT NULL,

Arrival_Time varchar(20) DEFAULT NULL,

PASSENGER_ID	FIRST_NAME	LAST_NAME	GENDER	PHONE_NO	TICKET_NO	AGE	CLASS
101	Nishant	Rai	M	7521481862	1111	22	First Class AC
102	Ravi	Gupta	M	9754956677	1112	25	Third Class AC
103	Mukesh	Sharma	M	8765778592	1113	19	Sleepar
104	Vikky	Singh	M	9344758664	1114	21	Second Class AC

Departure_Time varchar(20) DEFAULT NULL,

PRIMARY KEY (Train_No,Station_Code),

FOREIGN KEY (Train_No) REFERENCES Train (Train_No),

FOREIGN KEY (Station_Code) REFERENCES Station (Station_Code)

);

Inserting the Values

insert into account values('rai123','007rai','rai007@gmail.com','Varanasi'); insert into contact values('rai123',7521481862); insert into Passenger values (101, 'Nishant', 'Rai', 'M',

7521481862 , 1111, 22, 'First Class AC'); insert into Ticket values(1111,12559,'15/11/22','rai123'); insert into train values(12559,'NDLS BSB SF EXP',479,47,96,192,'N','Y','Y','Y','Y','Y','Y','Y','Y'); insert into station values('ALD','ALLAHABAD JUNCTION'); insert into stoppage values(12559,'ALD','22:05:00','22:30:00');

USERNAME	PASSWORD	EMAIL_ID	ADDRESS
rai123	007rai	rai007@gmail.com	Varanasi
mukesh	mukhsb78	mukeshsfa4ehb@gmail.com	Mumbai
anshul	ansh711598	arajput1999@gmail.com	Agra
naveen	447nav	nav4457@gmail.com	Jalandhar
ravi	rav451992	ravi24144ehb@gmail.com	Banglore
vikky	vik1128	vikky9982b@gmail.com	Bhopal

TICKET_NO	TRAIN_NO	DATE_OF_JOURNEY	USERNAME
1112	12560	20/11/22	ravi
1114	12240	11/12/22	vikky
1111	12559	15/11/22	rai123
1113	12239	1/12/22	mukesh

Queries

- **Basic Queries**

If user wants to view the contents of the table.

- Select * from passenger;

If user wants to add some extra rows into the table.

- Insert into passenger values ('rai123','007rai','rai007@gmail.com','Varanasi');

If the user wants to discard the table.

- Drop table passenger;
- Truncate table passenger;

If the user wants to delete a particular row from the table.

- Delete from passenger where Username = ‘rai123’;

If the user wants to modify the columns.

- Alter table passenger modify Username char(30);

STATION_CODE	TRAIN_NO	ARRIVAL_TIME
LKO	12559	19:20:00
LKO	12240	11:20:00
LKO	12560	07:00:00
LKO	12239	22:20:00

If the user wants to view the distinct values from the table.

- select distinct username from contact;

If the user wants to update the rows of the table.

- Update passenger set address=’Lucknow’ where username=’anshul’;

If the user wants to find total number of first class seats available on any train that reaches Lucknow on Monday.

```
create view a(Station_code,Train_no,Arrival_Time)as SELECT
Stoppage.Station_code,Train_no,Arrival_Time  from Station
inner join Stoppage on
station.Station_code=Stoppage.Station_code where
station.Station_name='LUCKNOW';

select * from a;
```

```
create view c
(Station_code,Train_no,Arrival_Time,First_Class_seats,Run_on_monda y)as
SELECT
Station_code,train.Train_no,Arrival_Time,Seat_First_Class_AC,Run_on
_Monday from train inner join a on train.Train_No=a.Train_No where
train.Run_On_Monday='Y' AND train.Seat_First_Class_AC >0;

select * from c;
```

STATION_CODE	TRAIN_NO	ARRIVAL_TIME	FIRST_CLASS_SEATS	RUN_ON_MONDAY
LKO	12559	19:20:00	47	Y
LKO	12240	11:20:00	48	Y
LKO	12560	07:00:00	43	Y
LKO	12239	22:20:00	48	Y

select SUm(First_class_seats) from c;

SUM(FIRST_CLASS_SEATS)
186

If the user wants to search the phone number of any user using his email.

Select Phone_no from Contact where username IN (Select Username from account where Email_id='vikky9982b@gmail.com');

- **PL/SQL Queries**

If the user wants to check the details of the train using ticket number. (Switch case & Exception Handling is used in this)

```
DECLARE
  ticket_no number(20):= :ticket_no;

BEGIN
 case
   when ticket_no = 1111 then dbms_output.put_line('12559 NDLS BSB SF EXP');
when ticket_no = 1112 then dbms_output.put_line('12560 BSB NDLS SF EXP');     when
ticket_no = 1113 then dbms_output.put_line('12239 KOTA PATNA EXP');     when
ticket_no = 1114 then dbms_output.put_line('12240 PATNA KOTA EXP');     --else
dbms_output.put_line('PREMIUM MODEL');
 END CASE;

Exception
  when       case_not_found                    Then
dbms_output.put_line('Not a valid ticket number');
END;
```

If the user wants to search the details of a passenger using his ID. (*In this query the concept of User defined function and Conditional statements are used).*

```
DECLARE
  p_id passenger.passenger_id%type :=
:p_id;   p_name passenger.first_name%type;
p_lname passenger.last_name%type;   p_age
passenger.age%type;   --User defined
Exception    ex_invalid_id EXCEPTION;
```

```
BEGIN
  IF p_id <=101 THEN
   RAISE ex_invalid_id;
ELSE
    select first_name, last_name, age into p_name, p_lname, p_age
FROM passenger



WHERE passenger_id=p_id;
    dbms_output.put_line('Name : ' || p_name ||' ' || p_lname);
dbms_output.put_line('Age :' || p_age);    END IF;

EXCEPTION
  WHEN ex_invalid_id THEN
  dbms_output.put_line('Id should be greater than 100');
WHEN no_data_found THEN
  dbms_output.put_line('Passenger does not exist');
WHEN others THEN
  dbms_output.put_line('Error');
END;
```

```
Name : Ravi Gupta
Age :25

Statement processed.

0.00 seconds
```

If the user wants to fetch the details of all the passengers. *(Concept of Explicit cursor and loops will be used in this PL/SQL query)*

```
declare p_id
passenger.passenger_id%type; p_name
passenger.First_name%type;
p_age passenger.age%type; CURSOR p_passenger is
select passenger_id, First_name, age from passenger;
begin open p_passenger; loop
fetch p_passenger into p_id,p_name,p_age;  exit
when p_passenger%notfound;
dbms_output.put_line(p_id || ' ' || p_name ||' ' ||
p_age);
```

end loop; close p_passenger; end;

```
101 Nishant 22
102 Ravi 25
103 Mukesh 19
104 Vikky 21

Statement processed.
```

If the user wants to add or delete a passenger from a table.

(*In this query the concept of Package and Procedure has been used.)*

```
CREATE OR REPLACE PACKAGE BODY p_package AS
 --Adds a Passenger
 PROCEDURE addPassenger(p_id passenger.passenger_id%type, p_name
passenger.first_name%type, p_lname passenger.last_name, p_gender passenger.gender,
p_phone passenger.phone_no, p_tick passenger.ticket_no%type, p_age passenger.age%type,
p_class passenger.class%type);
 --Delete a Passenger
 PROCEDURE delPassenger(p_id passenger.passenger_id%type);
END p_package;

CREATE OR REPLACE PACKAGE BODY p_package AS
 PROCEDURE addPassenger(p_id passenger.passenger_id%type, p_name
passenger.first_name%type,p_lname passenger.last_name,p_gender passenger.gender,
p_phone passenger.phone_no, p_tick passenger.ticket_no%type, p_age passenger.age%type,
p_class passenger.class%type)
 IS
 BEGIN
  INSERT INTO passengers(passenger_id, first_name,last_name, gender, phone_no,
ticket_no, age, class)
  VALUES(p_id, p_name, p_lname, p_gender, p_phone, p_tick, p_age, p_class);
END addPassenger;



Procedure delPassenger(p_id passenger.passenger_id%type) IS
 BEGIN
```

```
  DELETE from passenger
  WHERE passenger_id = p_id;
 END delPassenger;
END p_package;

        DECLARE    pp_id
       passenger.passenger_id%type := :pp_id;
BEGIN
 --p_package.addcustomer(106,'Shivam','Sinha','M' ,7413547952, , 1115, 30, First Class AC);
p_package.delcustomer(:pp_id);
 END;
```

Conclusion

In our project Railway Station Database system we have stored all the information about the Trains scheduled and the users booking tickets and even status of trains, seats etc. This data is helpful for the application which facilitates passengers to book the train tickets and check the details of trains and their status from their place itself. It avoids inconveniences of going to railway station for each and every query they get. We had considered the most important requirements only, many more features and details can be added to our project in order to obtain even more user friendly applications. These applications are already in progress and in future they can be upgraded and may become part of amazing technology.

Complete Code

```
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)
);

insert into account values('rai123','007rai','rai007@gmail.com','Varanasi'); insert into account
values('naveen','447nav','nav4457@gmail.com','Jalandhar'); insert into account
values('ravi','rav451992','ravi24144ehb@gmail.com','Banglore'); insert into account
values('mukesh','mukhsb78','mukeshsfa4ehb@gmail.com','Mumbai'); insert into account
values('vikky','vik1128','vikky9982b@gmail.com','Bhopal'); insert into account
values('anshul','ansh711598','arajput1999@gmail.com','Agra');

select * from account;

CREATE TABLE Contact (
  Username varchar(15) NOT NULL,
  Phone_No char(10) NOT NULL,
  PRIMARY KEY (Username,Phone_No),
  FOREIGN KEY (Username) REFERENCES Account (Username)
);

insert into contact values('rai123',7521481862);
insert into contact values('rai123',7451541862);
insert into contact values('ravi',9754956677); insert
into contact values('ravi',8874125796); insert into
contact values('naveen',7785426981); insert into
contact values('mukesh',8765778592); insert into
contact values('vikky',9344758664); insert into
contact values('anshul',7144778554);

select * from contact; select distinct
username from contact;

CREATE TABLE Passenger(
  Passenger_Id number(11) NOT NULL,
  First_Name varchar(20) NOT NULL,
  Last_Name varchar(20) NOT NULL,
  Gender char(1) NOT NULL,
  Phone_No char(10) DEFAULT NULL,
```

```
  Ticket_No number(10) NOT NULL,
  Age number(11) NOT NULL,
  Class varchar(20) NOT NULL,
  PRIMARY KEY (Passenger_Id),
  FOREIGN KEY (Ticket_No) REFERENCES Ticket(Ticket_No)
);

insert into Passenger values (101, 'Nishant', 'Rai', 'M', 7521481862 , 1111, 22, 'First Class AC');
insert into Passenger values (102, 'Ravi', 'Gupta', 'M', 9754956677 , 1112, 25, 'Third Class AC');
insert into Passenger values (103, 'Mukesh', 'Sharma', 'M', 8765778592 , 1113, 19, 'Sleepar');  insert
into Passenger values (104, 'Vikky', 'Singh', 'M', 9344758664 , 1114, 21, 'Second Class AC');

select * from Passenger;

CREATE TABLE Ticket (
  Ticket_No number(10) NOT NULL,
  Train_No number(6) NOT NULL,
  Date_of_Journey varchar(20) NOT NULL,
  Username varchar(15) NOT NULL,
  PRIMARY KEY (Ticket_No),
  FOREIGN KEY (Username) REFERENCES Account (Username),
  FOREIGN KEY (Train_No) REFERENCES Train (Train_No)
);
alter table ticket add Price number(10);  update ticket
set Price=1500 where Ticket_No=1114;

insert  into  Ticket  values(1111,12559,'15/11/22','rai123');
insert into Ticket values(1112,12560,'20/11/22','ravi'); insert
into  Ticket  values(1113,12239,'1/12/22','mukesh');  insert
into Ticket values(1114,12240,'11/12/22','vikky');

select * from ticket;

CREATE TABLE Train (
  Train_No number(6) NOT NULL,
  Name varchar(25) NOT NULL,
```

```
  Seat_Sleeper number(4) NOT NULL,
  Seat_First_Class_AC number(4) NOT NULL,
  Seat_Second_Class_AC number(4) NOT NULL,
  Seat_Third_Class_AC number(4) NOT NULL,
  Wifi char(1) NOT NULL,
  Food char(1) NOT NULL,
  Run_On_Sunday char(1) NOT NULL,
  Run_On_Monday char(1) NOT NULL,
  Run_On_Tuesday char(1) NOT NULL,
  Run_On_Wednesday char(1) NOT NULL,
  Run_On_Thursday char(1) NOT NULL,
  Run_On_Friday char(1) NOT NULL,
  Run_On_Saturday char(1) NOT NULL,
  PRIMARY KEY (Train_No)
);

insert into train values(12559,'NDLS BSB SF EXP',479,47,96,192,'N','Y','Y','Y','Y','Y','Y','Y','Y');
insert into train values(12560,'BSB NDLS SF EXP',480,43,96,192,'N','Y','Y','Y','Y','Y','Y','Y','Y');
insert into train values(12239,'KOTA PATNA EX',432,48,80,144,'N','N','Y','Y','Y','Y','Y','Y','Y'); insert
into train values(12240,'PATNA KOTA EX',432,48,80,144,'N','N','Y','Y','Y','Y','Y','Y','Y');

select * from train;

CREATE TABLE Station(
  Station_Code char(5) NOT NULL,
  Station_Name varchar(25) NOT NULL,
  PRIMARY KEY (Station_Code)
);

insert into station values('ALD','ALLAHABAD JUNCTION');
insert into station values('CNB','KANPUR CENTRAL'); insert
into station values('GYN','GYANPUR ROAD'); insert into
station values('GZB','GHAZIABAD JUNCTION'); insert into
station values('LKO','LUCKNOW'); insert into station
values('NDLS','NEW DELHI');
```

```
SELECT * FROM STATION;

CREATE TABLE Stoppage(
  Train_No number(6) NOT NULL,
  Station_Code char(5) NOT NULL,
  Arrival_Time varchar(20) DEFAULT NULL,
  Departure_Time varchar(20) DEFAULT NULL,
  PRIMARY KEY (Train_No,Station_Code),
  FOREIGN KEY (Train_No) REFERENCES Train (Train_No),
  FOREIGN KEY (Station_Code) REFERENCES Station (Station_Code)
);

insert into stoppage values(12559,'ALD','22:05:00','22:30:00');
insert into stoppage values(12559,'CNB','01:30:00','01:38:00');
insert into stoppage values(12559,'LKO','19:20:00','19:30:00');
insert into stoppage values(12559,'NDLS','08:10:00',NULL);
insert into stoppage values(12560,'ALD','03:45:00','04:10:00');
insert into stoppage values(12560,'CNB','01:00:00','01:05:00');
insert into stoppage values(12560,'LKO','07:00:00',NULL); insert
into stoppage values(12560,'NDLS','18:35:00','18:55:00'); insert
into stoppage values(12239,'ALD','01:20:00','01:45:00'); insert
into stoppage values(12239,'CNB','04:15:00','04:20:00'); insert
into stoppage values(12239,'GYN','23:31:00','23:33:00'); insert
into stoppage values(12239,'GZB','11:30:00','11:32:00'); insert
into stoppage values(12239,'LKO','22:20:00','22:30:00'); insert
into stoppage values(12239,'NDLS','12:20:00',NULL); insert into
stoppage values(12240,'ALD','07:45:00','08:15:00'); insert into
stoppage values(12240,'CNB','04:55:00','05:00:00'); insert into
stoppage values(12240,'GYN','09:21:00','09:23:00'); insert into
stoppage values(12240,'GZB','23:03:00','23:05:00'); insert into
stoppage values(12240,'LKO','11:20:00',NULL); insert into
stoppage values(12240,'NDLS','22:15:00','22:25:00');

select * from stoppage;

create view a(Station_code,Train_no,Arrival_Time)as SELECT
```

```
Stoppage.Station_code,Train_no,Arrival_Time from Station inner join Stoppage on
station.Station_code=Stoppage.Station_code where
station.Station_name='LUCKNOW';

select * from a;

create view c(Station_code,Train_no,Arrival_Time,First_Class_seats,Run_on_monday)as SELECT
Station_code,train.Train_no,Arrival_Time,Seat_First_Class_AC,Run_on_monday from train inner
join a on train.Train_No=a.Train_No where train.Run_On_Monday='Y' AND
train.Seat_First_Class_AC >0;

select * from c;

select SUm(First_class_seats) from
c;

select * from train;

Select Phone_no from Contact where username IN (Select Username from account where
Email_id='vikky9982b@gmail.com');


select count(*) as total from
account;

create trigger cancellation10
before delete on ticket  for each
row   BEGIN
  set trainno=old.train_no;
set ticketno=old.ticket_no;
SET class = (SELECT p.class
            FROM PASSENGER p
            WHERE p.ticket_no = ticketno);   if
class='first class ac' then
   UPDATE Train set Seat_First_Class_AC = Seat_First_Class_AC+1 WHERE Train_No = trainno ;
elsif class='sleeper' then
```

```
   UPDATE Train set Seat_Sleeper = Seat_Sleeper+1 WHERE Train_No = trainno ;      elsif
class='second class ac' then
   UPDATE Train set Seat_Second_Class_AC = Seat_Second_Class_AC+1 WHERE Train_No = trainno ;
elsif class='third class ac' then
   UPDATE Train set Third_Class_AC = Seat_Third_Class_AC+1 WHERE Train_No = trainno ;
end if;
END;

DECLARE
  ticket_no number(20):= :ticket_no;

BEGIN
 case     when ticket_no = 1111 then dbms_output.put_line('12559 NDLS BSB SF
EXP');     when ticket_no = 1112 then dbms_output.put_line('12560 BSB NDLS SF
EXP');     when ticket_no = 1113 then dbms_output.put_line('12239 KOTA PATNA
EXP');     when ticket_no = 1114 then dbms_output.put_line('12240 PATNA KOTA
EXP');     --else dbms_output.put_line('PREMIUM MODEL');
 END CASE;

Exception          when  case_not_found          Then
dbms_output.put_line('Not a valid ticket number');
END;



select * from ticket;

DECLARE
 p_id passenger.passenger_id%type := :p_id;
p_name passenger.first_name%type;
p_lname passenger.last_name%type;   p_age
passenger.age%type;   --User defined
Exception   ex_invalid_id EXCEPTION;

BEGIN
 IF p_id <=101 THEN
  RAISE ex_invalid_id;   ELSE
```

```
    select first_name, last_name, age into p_name, p_lname, p_age
    FROM passenger
    WHERE passenger_id=p_id;
dbms_output.put_line('Name : ' || p_name ||' ' || p_lname);
dbms_output.put_line('Age :' || p_age);
 END IF;

EXCEPTION
 WHEN ex_invalid_id THEN
 dbms_output.put_line('Id should be greater than 100');   WHEN
no_data_found THEN
 dbms_output.put_line('Passenger does not exist');
WHEN others THEN
 dbms_output.put_line('Error');
END;

declare p_id
passenger.passenger_id%type;
p_name
passenger.First_name%type; p_age
passenger.age%type; CURSOR
p_passenger is select
passenger_id, First_name, age
from passenger; begin open
p_passenger; loop fetch
p_passenger into
p_id,p_name,p_age; exit when
p_passenger%notfound;
dbms_output.put_line(p_id || ' ' || p_name ||' ' || p_age);
end loop; close p_passenger; end;


CREATE OR REPLACE PACKAGE BODY p_package AS
 --Adds a Passenger
 PROCEDURE addPassenger(p_id passenger.passenger_id%type, p_name passenger.first_name%type,
p_lname passenger.last_name, p_gender passenger.gender, p_phone passenger.phone_no, p_tick
```

```
passenger.ticket_no%type, p_age passenger.age%type, p_class passenger.class%type);   --Delete a
Passenger
  PROCEDURE delPassenger(p_id passenger.passenger_id%type);
END p_package;



CREATE OR REPLACE PACKAGE BODY p_package AS
  PROCEDURE  addPassenger(p_id        passenger.passenger_id%type,          p_name
passenger.first_name%type,p_lname       passenger.last_name,p_gender          passenger.gender,
      p_phone passenger.phone_no,          p_tick  passenger.ticket_no%type,     p_age
      passenger.age%type,  p_class passenger.class%type)
  IS
  BEGIN
    INSERT INTO passengers(passenger_id, first_name,last_name, gender, phone_no, ticket_no, age, class)
    VALUES(p_id, p_name, p_lname, p_gender, p_phone, p_tick, p_age, p_class);
END addPassenger;

Procedure delPassenger(p_id passenger.passenger_id%type) IS
  BEGIN


    DELETE from passenger
    WHERE passenger_id = p_id;
  END delPassenger;
END p_package;



DECLARE
  pp_id passenger.passenger_id%type := :pp_id;
BEGIN
  --p_package.addcustomer(106,'Shivam','Sinha','M' ,7413547952, , 1115, 30, First Class AC);
p_package.delcustomer(:pp_id);
  END;

select * from passenger;
```