DBMS Project Report

Submitted for

DATABASE MANAGEMENT SYSTEM (UCS310)

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Computer Science and Engineering Department

TIET, Patiala

JAN-JUN 2023

RAILWAY MANAGEMENT SYSTEM

Submitted by:

Radhika Goel (102117036)

Jasmine Kaur (102117041)

Jasleen Kataria (102117042)

Vansh Sandhir (102117060)

Submitted to:

Mrs. Archana Singh

Abstract

The Railway Management 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 in advance. 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 a 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 a design of the database of the railway reservation system based on relation model. Example of some SQL queries to retrieve data from rail management database.

INDEX

Introduction

Today, databases and their management are at the core of functioning in a world driven by data and connectivity. People encounter several activities every day that involve interaction with databases like banking, ticket booking, movie streaming, online shopping, and a lot more.

One such area that uses databases at its core is the railway industry. The main purpose of maintaining 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.

Problem Description:

Using the proposed system, 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 validation is done, 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 ticket ID is searched and the corresponding record is deleted. With this, the first ticket with waiting status also gets confirmed.

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.

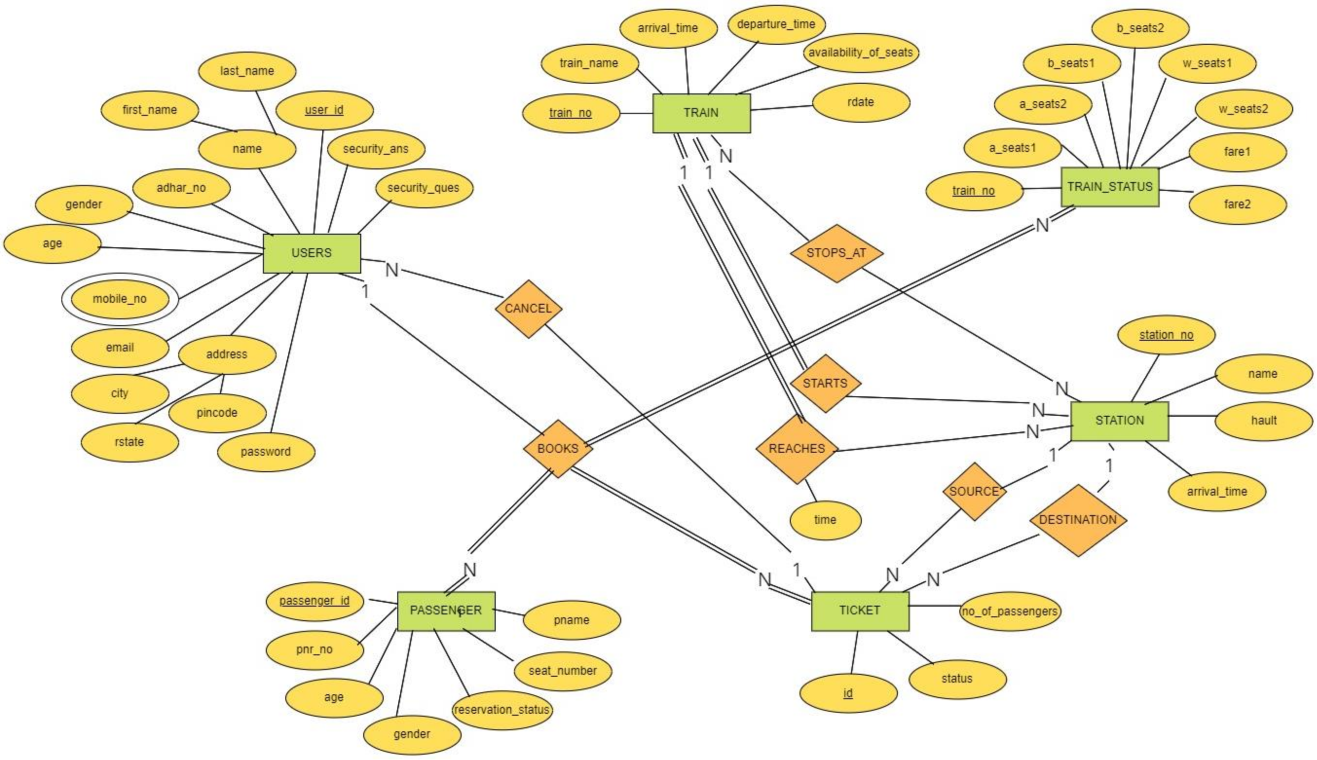
List of trains has to be maintained. Detailed passenger information is to be maintained in the booking procedure, the train number, train date, and category are read from the passenger. On the basis of the values provided by the passenger, corresponding record is retrieved from the TRAIN_STATUS. If the 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 and their Attributes

ENTITIES	**ATTRIBUTES**
USERS	**user_id** first_name last_name adhar_no gender age mobile_no email city rstate pincode password security_ques security_ans
TRAIN	**train_no** train_name arrival_time departure_time availability_of_seats rdate
STATION	**station_no** name hault arrival_time train_no
TRAIN_STATUS	**train_no** a_seats1 a_seats2 b_seats1 b_seats2 w_seats1 w_seats2 fare1 fare2
TICKET	**id** user_id status no_of_passengers train_no

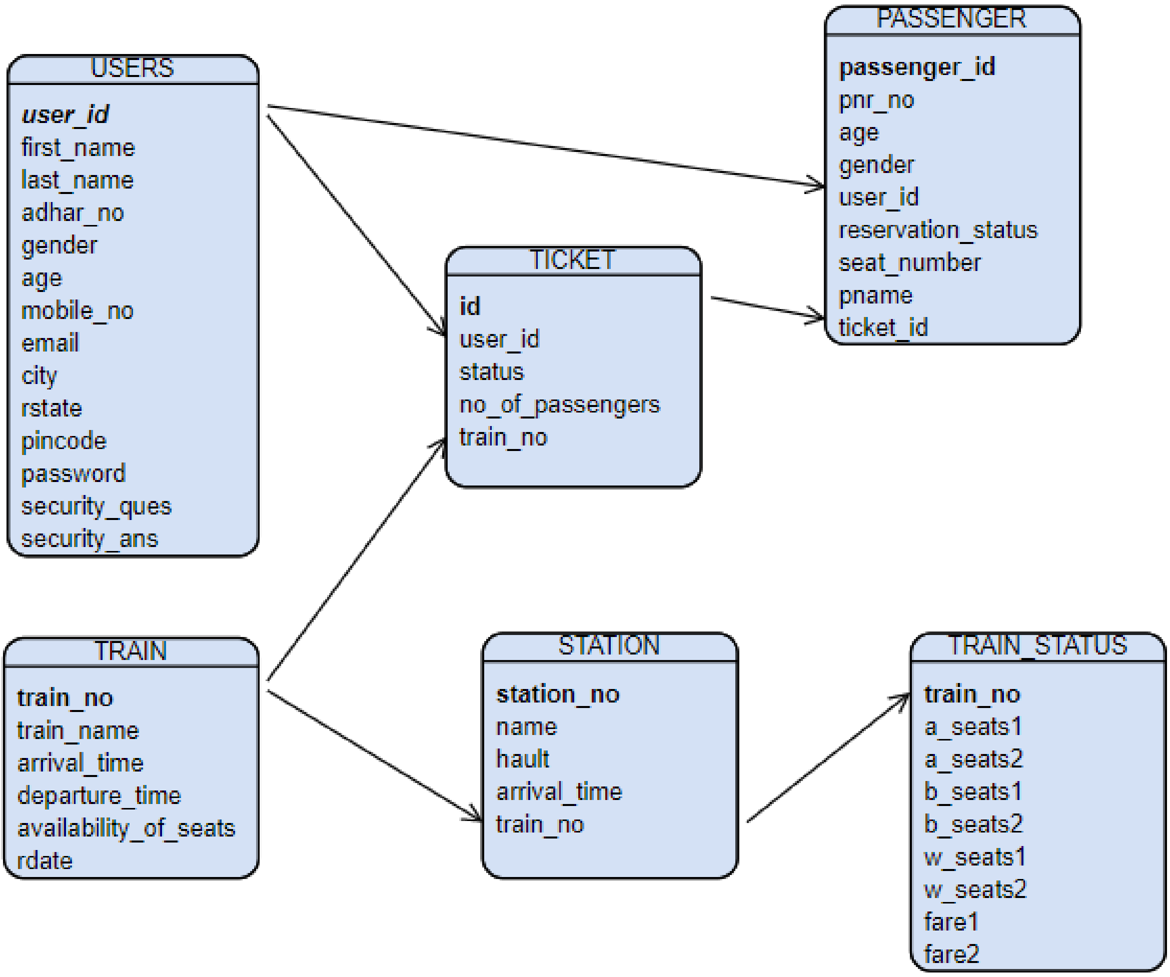
PASSENGER	**passenger_id** pnr_no age gender user_id reservation_status seat_number pname ticket_id

ER Diagram

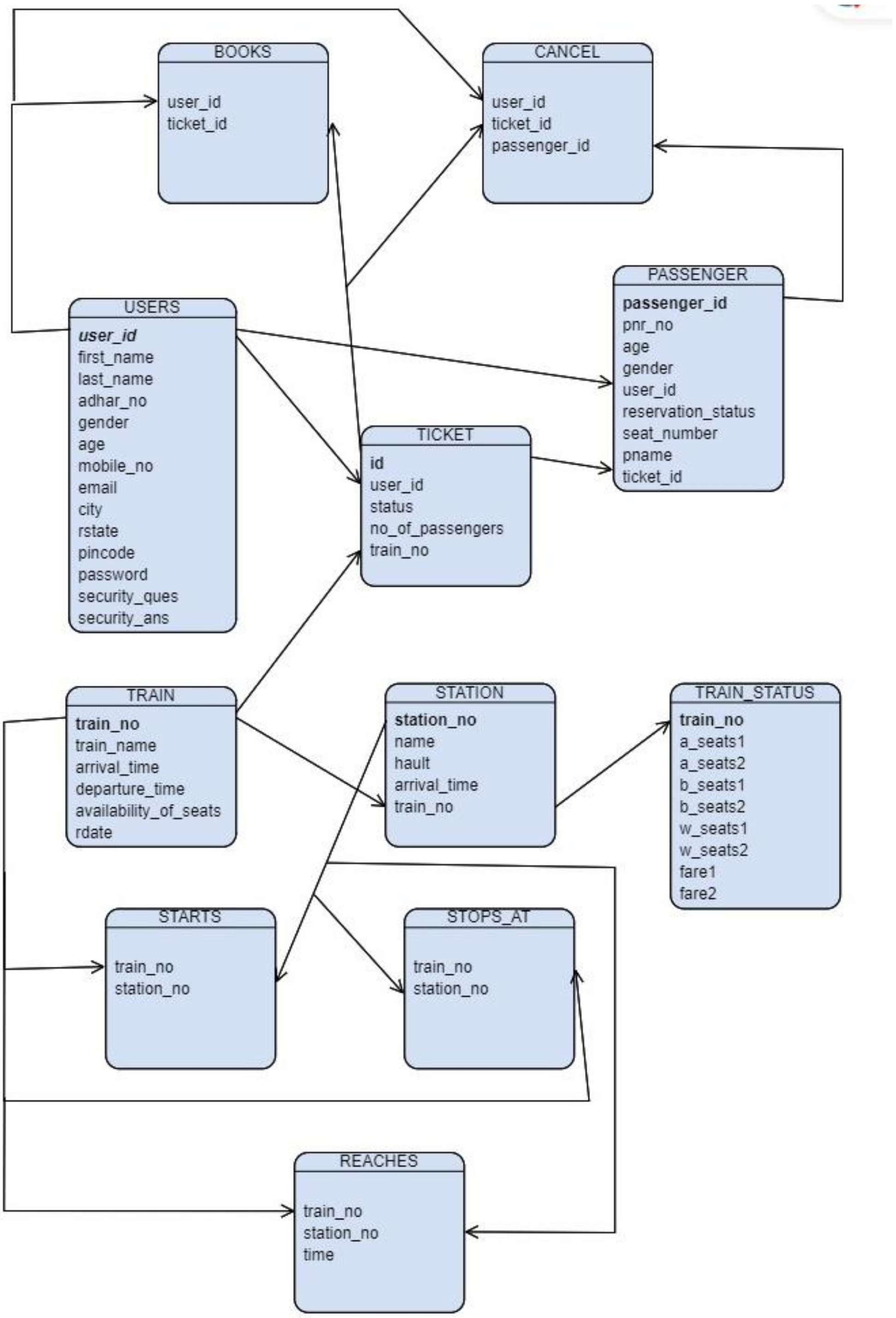


arrival_time
departure_time
availability_of_seats
train_name
rdate
train_no
TRAIN
b_seats2
b_seats1
w_seats1
a_seats2
w_seats2
a_seats1
fare1
train_no
TRAIN_STATUS
fare2
last_name
first_name
user_id
name
security_ans
adhar_no
security_ques
gender
age
USERS
mobile_no
email
address
city
pincode
rstate
password
CANCEL
BOOKS
STOPS_AT
STARTS
REACHES
time
station_no
name
STATION
hault
arrival_time
SOURCE
DESTINATION
passenger_id
PASSENGER
pname
pnr_no
seat_number
age
reservation_status
gender
TICKET
no_of_passengers
id
status
N
1
1
1
N
N
N
N
N
N
1
1
N
N
1
N
N

ER to Relational Table



Normalization



SQL Code

- **Creating Tables**

```
create table USERS
(
    user_id int primary key,
    first_name varchar(50),
    last_name varchar(50),
    adhar_no varchar(20),
    gender varchar(2),
    age int,
    mobile_no number(10),
    email varchar(50),
    city varchar(50),
    rstate varchar(50),
    pincode number(6),
    password varchar(50),
    security_ques varchar(50),
    security_ans varchar(50)
);
create table TRAIN
(
    train_no int primary key,
    train_name varchar(50),
    arrival_time varchar(50),
    departure_time varchar(50),
    availability_of_seats varchar(50),
    rdate date
);
```

```
create table STATION
(
    station_no int primary key,
    name varchar(50),
    hault int,
    arrival_time varchar(20),
    train_no int,
    foreign key(train_no) references TRAIN(train_no)
);
create table TRAIN_STATUS
(
    train_no int primary key,
    a_seats1 int,
    a_seats2 int,
    b_seats1 int,
    b_seats2 int,
    w_seats1 int,
    w_seats2 int,
    fare1 float,
    fare2 float
);
create table TICKET
(
    id int primary key,
    user_id int,
    status varchar(20),
    no_of_passengers int,
    train_no int,
    foreign key(user_id) references USERS(user_id),
    foreign key(train_no) references TRAIN(train_no)
);
create table PASSENGER
(
    passenger_id int primary key,
    pnr_no int,
    age int,
    gender varchar(2),
    user_id int,
    reservation_status varchar(20),
    seat_number varchar(5),
    pname varchar(50),
    ticket_id int,
    foreign key(user_id) references USERS(user_id),
    foreign key(ticket_id) references TICKET(id)
);
create table STARTS
(
    train_no int,
    station_no int,
    foreign key(train_no) references TRAIN(train_no),
    foreign key(station_no) references STATION(station_no)
);
create table STOPS_AT
(
    train_no int,
    station_no int,
    foreign key(train_no) references TRAIN(train_no),
    foreign key(station_no) references STATION(station_no)
);
create table BOOKS
```

```
(
    user_id int,
    ticket_id int,
    foreign key(user_id) references USERS(user_id),
    foreign key(ticket_id) references TICKET(id)
);
create table CANCEL
(
    user_id int,
    ticket_id int,
    passenger_id int,
    foreign key(user_id) references USERS(user_id),
    foreign key(ticket_id) references TICKET(id),
    foreign key(passenger_id) references PASSENGER(passenger_id)
);
create table REACHES
(
    train_no int,
    station_no int,
    time varchar(20),
    foreign key(train_no) references TRAIN(train_no),
    foreign key(station_no) references STATION(station_no)
);
```

- **Inserting Values and Displaying Tables**

insert into USERS values(1701, 'vijay', 'sharma', '309887340843','M',34,9887786655, 'vijay1@gmail.com','vijaywada', 'andhrapradesh', '520001','12345@#','favouritecolour','red');

insert into USERS values(1702,'rohit','kumar','456709871234','M',45,9809666555, 'rohithkumar@gmail.com','guntur','andhrapradesh','22004','12@#345','favouritebike','bmw');

insert into USERS values(1703, 'manasvi', 'sree','765843210987','F',20,9995550666,'manasvi57@gmail.com','guntur','andhrapradesh' ,'52200','40987hr', 'favouriteflower', 'rose');

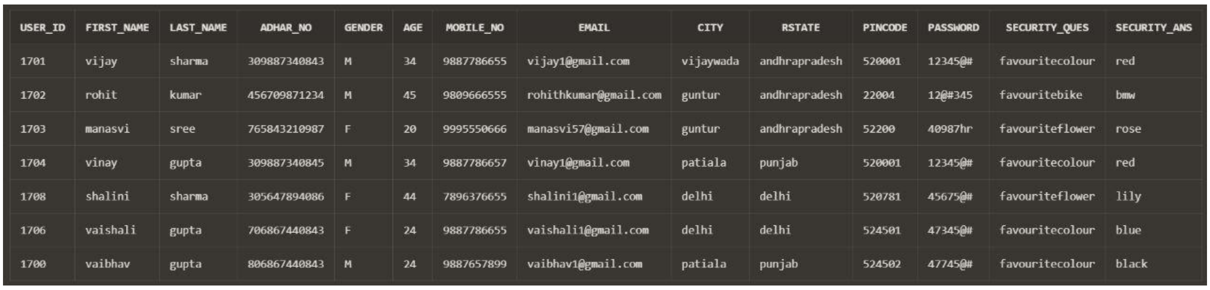
insert into USERS values(1704, 'vinay', 'gupta', '309887340845','M',34,9887786657, 'vinay1@gmail.com','patiala', 'punjab', '520001','12345@#','favouritecolour','red');

insert into USERS values(1708, 'shalini', 'sharma', '305647894086','F',44,7896376655, 'shalini1@gmail.com','delhi', 'delhi', '520781','45675@#','favouriteflower','lily');

insert into USERS values(1706, 'vaishali', 'gupta', '706867440843','F',24,9887786655, 'vaishali1@gmail.com','delhi', 'delhi', '524501','47345@#','favouritecolour','blue');

insert into USERS values(1700, 'vaibhav', 'gupta', '806867440843','M',24,9887657899, 'vaibhav1@gmail.com','patiala', 'punjab', '524502','47745@#','favouritecolour','black');

select * from USERS;



USER_ID	FIRST_NAME	LAST_NAME	ADHAR_NO	GENDER	AGE	MOBILE_NO	EMAIL	CITY	RSTATE	PINCODE	PASSWORD	SECURITY_QUES	SECURITY_ANS
1701	vijay	sharma	309887340843	M	34	9887786655	vijay1@gmail.com	vijaywada	andhrapradesh	520001	12345@#	favouritecolour	red
1702	rohit	kumar	456709871234	M	45	9809666555	rohithkumar@gmail.com	guntur	andhrapradesh	22004	12@#345	favouritebike	bmw
1703	manasvi	sree	765843210987	F	20	9995550666	manasvi57@gmail.com	guntur	andhrapradesh	52200	40987hr	favouriteflower	rose
1704	vinay	gupta	309887340845	M	34	9887786657	vinay1@gmail.com	patiala	punjab	520001	12345@#	favouritecolour	red
1708	shalini	sharma	305647894086	F	44	7896376655	shalini1@gmail.com	delhi	delhi	520781	45675@#	favouriteflower	lily
1706	vaishali	gupta	706867440843	F	24	9887786655	vaishali1@gmail.com	delhi	delhi	524501	47345@#	favouritecolour	blue
1700	vaibhav	gupta	806867440843	M	24	9887657899	vaibhav1@gmail.com	patiala	punjab	524502	47745@#	favouritecolour	black

insert into TRAIN values(12711,'pinakini_exp','11:30:00','11:40:00','A','6-june-2023');

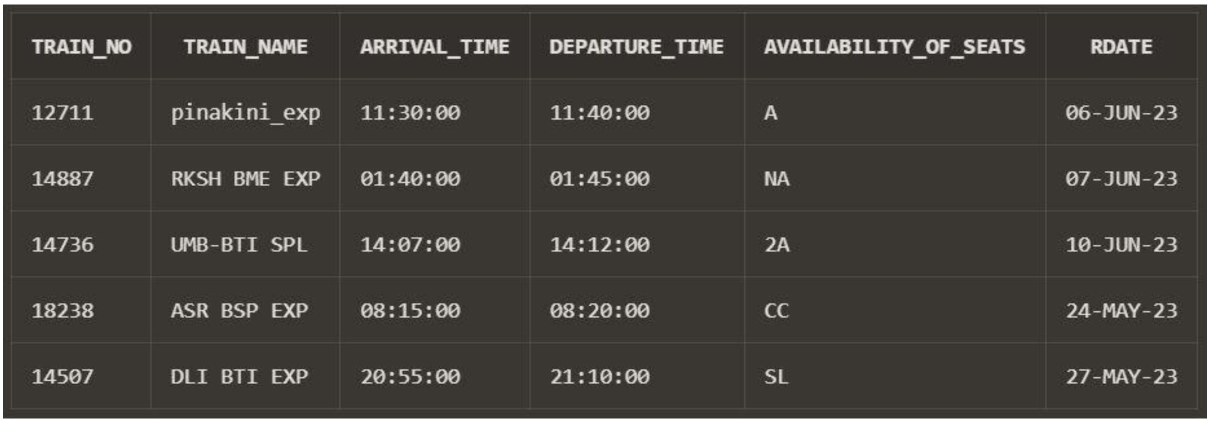
insert into TRAIN values(14887,'RKSH BME EXP','01:40:00','01:45:00','NA','7-june-2023');

insert into TRAIN values(14736,'UMB-BTI SPL','14:07:00','14:12:00','2A','10-june-2023');

insert into TRAIN values(18238,'ASR BSP EXP','08:15:00','08:20:00','CC','24-may-2023');

insert into TRAIN values(14507,'DLI BTI EXP','20:55:00','21:10:00','SL','27-may-2023');

select * from TRAIN;



TRAIN_NO	TRAIN_NAME	ARRIVAL_TIME	DEPARTURE_TIME	AVAILABILITY_OF_SEATS	RDATE
12711	pinakini_exp	11:30:00	11:40:00	A	06-JUN-23
14887	RKSH BME EXP	01:40:00	01:45:00	NA	07-JUN-23
14736	UMB-BTI SPL	14:07:00	14:12:00	2A	10-JUN-23
18238	ASR BSP EXP	08:15:00	08:20:00	CC	24-MAY-23
14507	DLI BTI EXP	20:55:00	21:10:00	SL	27-MAY-23

insert into STATION values(11,'vijayawada',10,'11:30:00', 12711);

insert into STATION values(12,'patiala',5,'01:40:00',14887);

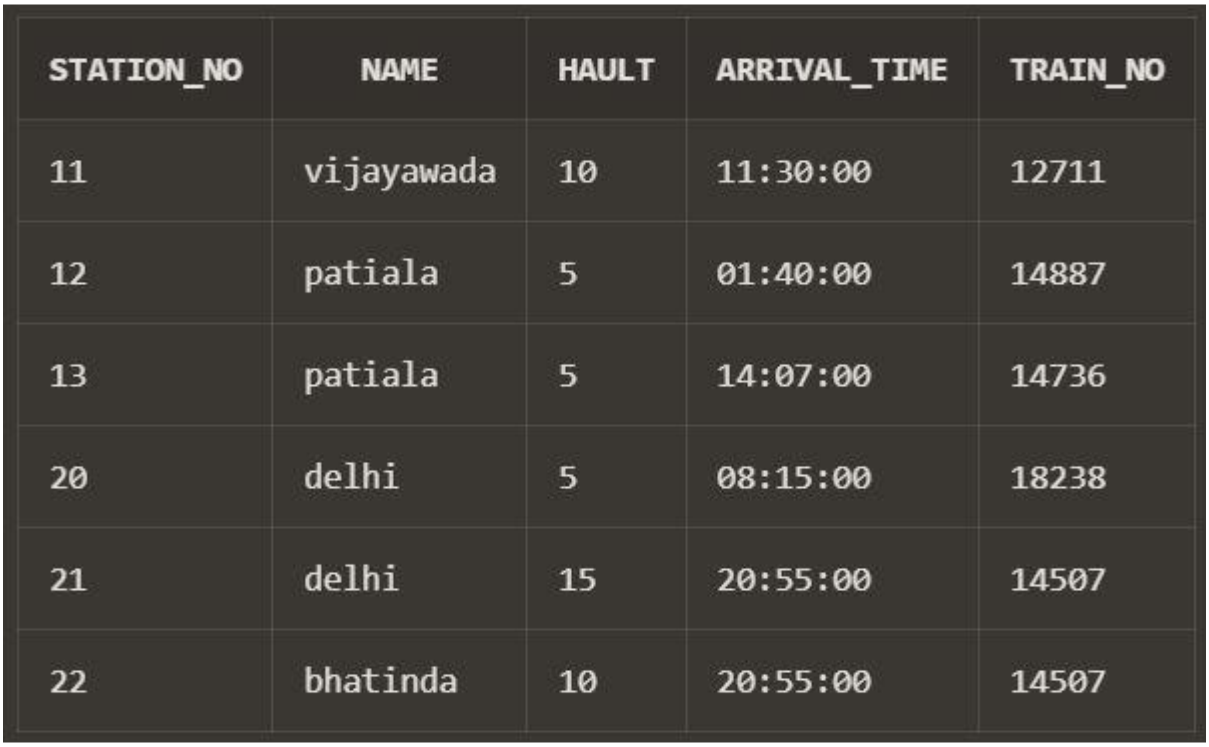
insert into STATION values(13,'patiala',5,'14:07:00',14736);

insert into STATION values(20,'delhi',5,'08:15:00',18238);

insert into STATION values(21,'delhi',15,'20:55:00',14507);

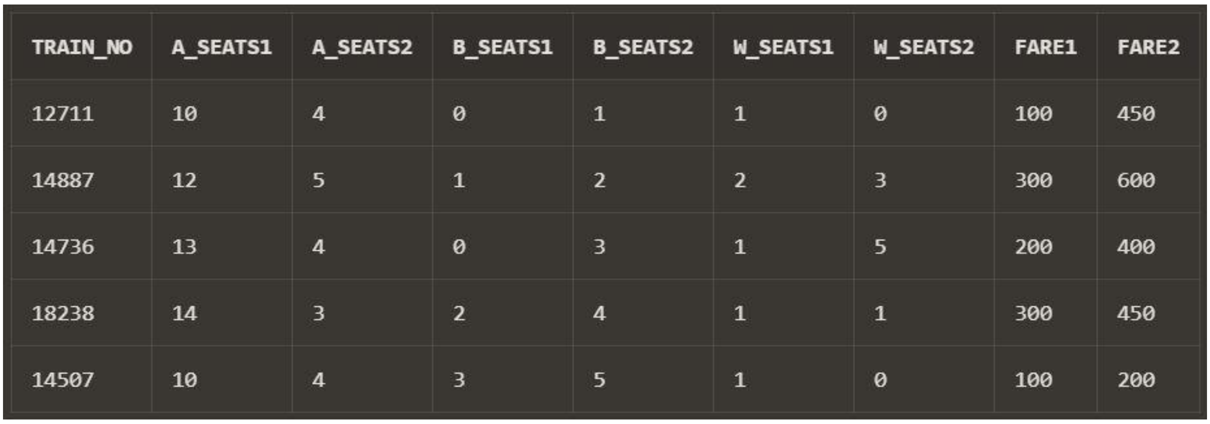
insert into STATION values(22,'bhatinda',10,'20:55:00',14507);

select * from STATION;



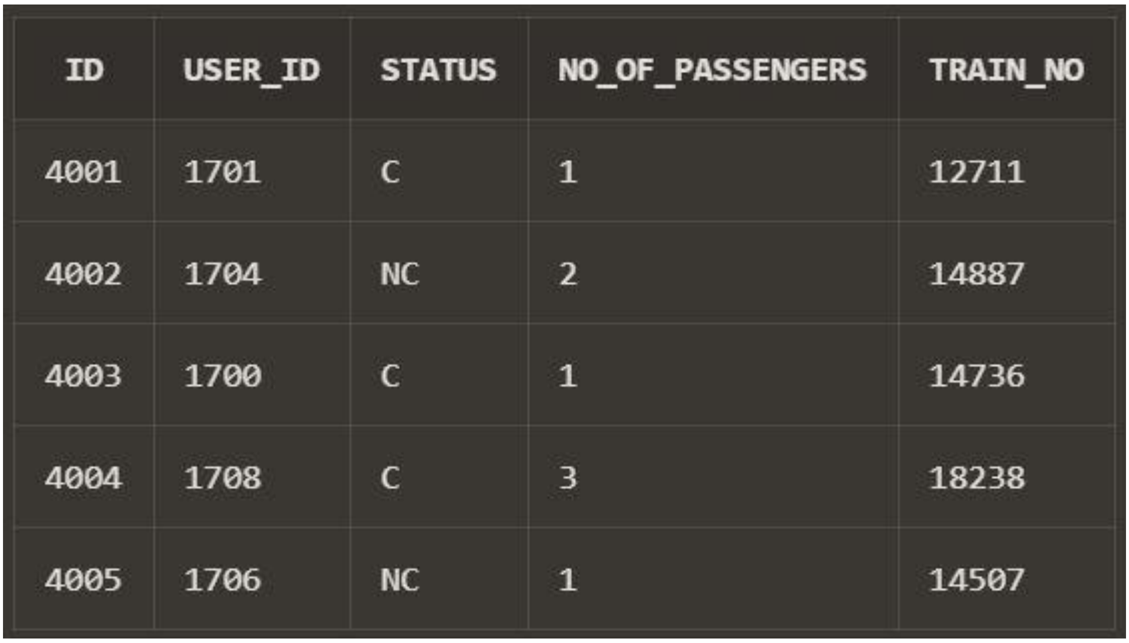
STATION_NO	NAME	HAULT	ARRIVAL_TIME	TRAIN_NO
11	vijayawada	10	11:30:00	12711
12	patiala	5	01:40:00	14887
13	patiala	5	14:07:00	14736
20	delhi	5	08:15:00	18238
21	delhi	15	20:55:00	14507
22	bhatinda	10	20:55:00	14507

```
insert into TRAIN_STATUS values(12711,10,4,0,1,1,0,100,450);
insert into TRAIN_STATUS values(14887,12,5,1,2,2,3,300,600);
insert into TRAIN_STATUS values(14736,13,4,0,3,1,5,200,400);
insert into TRAIN_STATUS values(18238,14,3,2,4,1,1,300,450);
insert into TRAIN_STATUS values(14507,10,4,3,5,1,0,100,200);
select * from TRAIN_STATUS;
```



TRAIN_NO	A_SEATS1	A_SEATS2	B_SEATS1	B_SEATS2	W_SEATS1	W_SEATS2	FARE1	FARE2
12711	10	4	0	1	1	0	100	450
14887	12	5	1	2	2	3	300	600
14736	13	4	0	3	1	5	200	400
18238	14	3	2	4	1	1	300	450
14507	10	4	3	5	1	0	100	200

```
insert into TICKET values(4001,1701,'C',1,12711);
insert into TICKET values(4002,1704, 'NC',2,14887);
insert into TICKET values(4003,1700,'C',1,14736);
insert into TICKET values(4004,1708,'C',3,18238);
insert into TICKET values(4005,1706,'NC',1,14507);
select * from TICKET;
```



ID	USER_ID	STATUS	NO_OF_PASSENGERS	TRAIN_NO
4001	1701	C	1	12711
4002	1704	NC	2	14887
4003	1700	C	1	14736
4004	1708	C	3	18238
4005	1706	NC	1	14507

insert into PASSENGER values(5001,78965,45,'M',1701 ,'C','B6-45',' ramesh',4001);

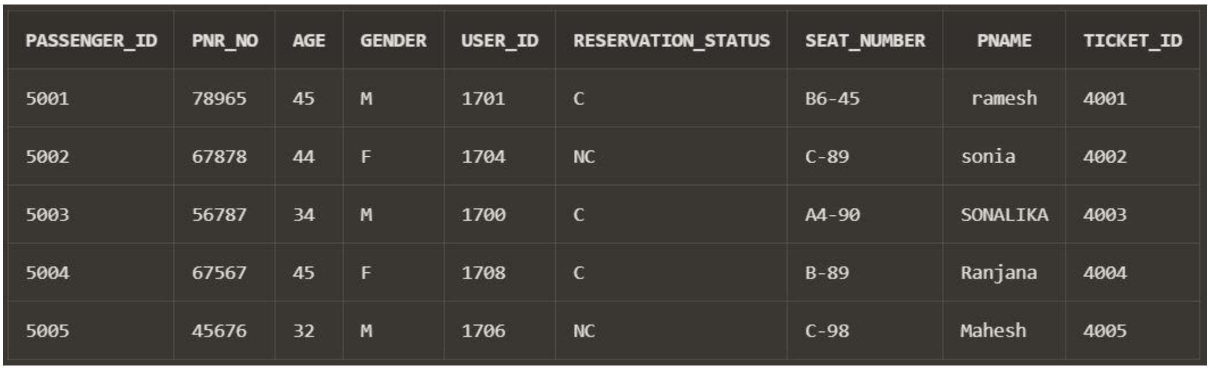
insert into PASSENGER values(5002,67878,44,'F',1704,'NC','C-89','sonia',4002);

insert into PASSENGER values(5003,56787,34,'M',1700,'C','A4-90','SONALIKA',4003);

insert into PASSENGER values(5004,67567,45,'F',1708,'C','B-89','Ranjana',4004);

insert into PASSENGER values(5005,45676,32,'M',1706,'NC','C-98','Mahesh',4005);

select * from PASSENGER;



PASSENGER_ID	PNR_NO	AGE	GENDER	USER_ID	RESERVATION_STATUS	SEAT_NUMBER	PNAME	TICKET_ID
5001	78965	45	M	1701	C	B6-45	ramesh	4001
5002	67878	44	F	1704	NC	C-89	sonia	4002
5003	56787	34	M	1700	C	A4-90	SONALIKA	4003
5004	67567	45	F	1708	C	B-89	Ranjana	4004
5005	45676	32	M	1706	NC	C-98	Mahesh	4005

insert into STARTS values(12711,11);

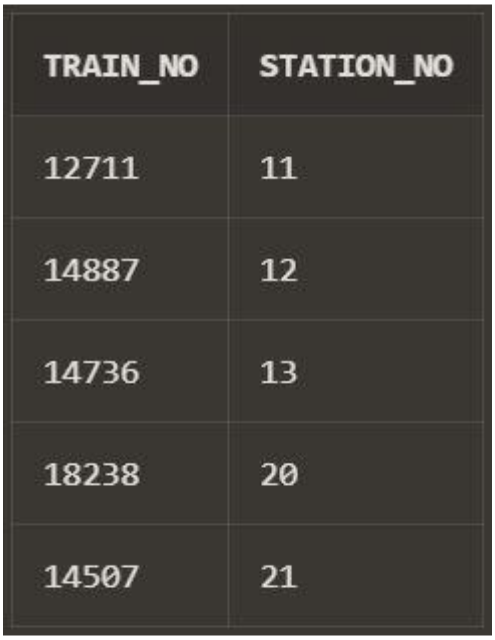
insert into STARTS values(14887,12);

insert into STARTS values(14736,13);

insert into STARTS values(18238,20);

insert into STARTS values(14507,21);

select * from STARTS;



TRAIN_NO	STATION_NO
12711	11
14887	12
14736	13
18238	20
14507	21

insert into STOPS_AT values(12711,20);

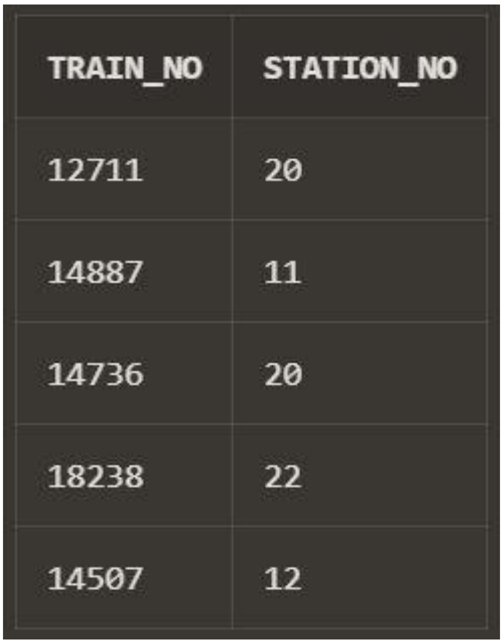
insert into STOPS_AT values(14887,11);

insert into STOPS_AT values(14736,20);

insert into STOPS_AT values(18238,22);

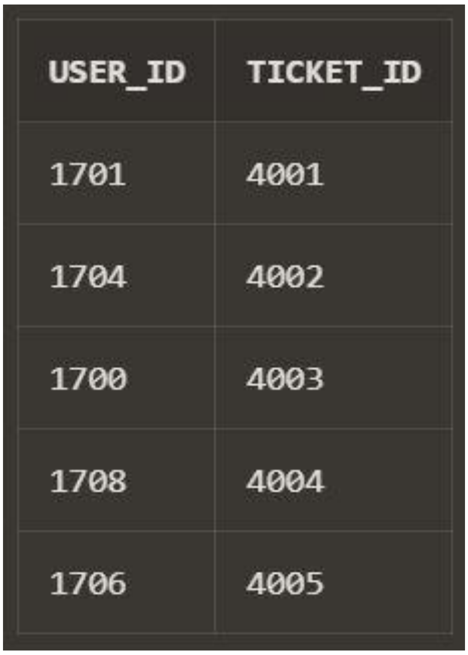
insert into STOPS_AT values(14507,12);

select * from STOPS_AT;



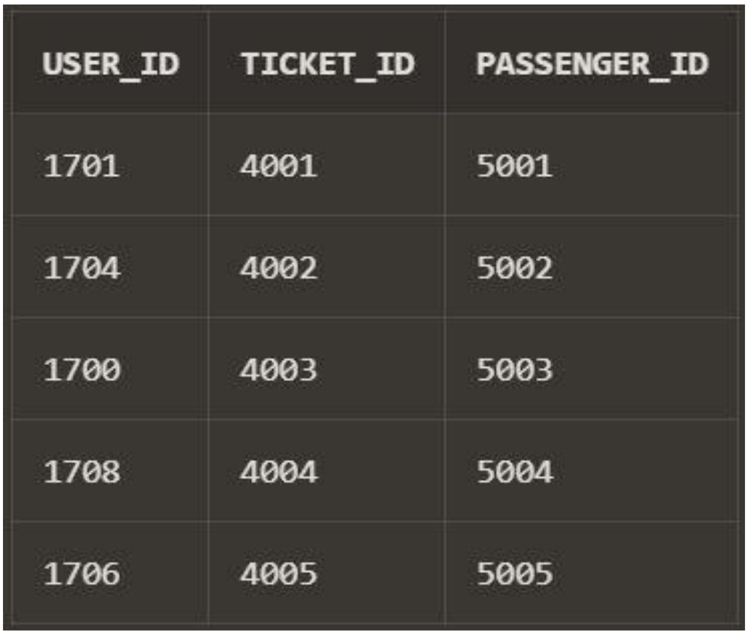
TRAIN_NO	STATION_NO
12711	20
14887	11
14736	20
18238	22
14507	12

```
insert into BOOKS values(1701,4001);
insert into BOOKS values(1704,4002);
insert into BOOKS values(1700,4003);
insert into BOOKS values(1708,4004);
insert into BOOKS values(1706,4005);
select * from BOOKS;
```



USER_ID	TICKET_ID
1701	4001
1704	4002
1700	4003
1708	4004
1706	4005

```
insert into CANCEL values(1701,4001,5001);
insert into CANCEL values(1704,4002,5002);
insert into CANCEL values(1700,4003,5003);
insert into CANCEL values(1708,4004,5004);
insert into CANCEL values(1706,4005,5005);
select * from CANCEL;
```



USER_ID	TICKET_ID	PASSENGER_ID
1701	4001	5001
1704	4002	5002
1700	4003	5003
1708	4004	5004
1706	4005	5005

insert into REACHES values(12711,20,'04:00:00');

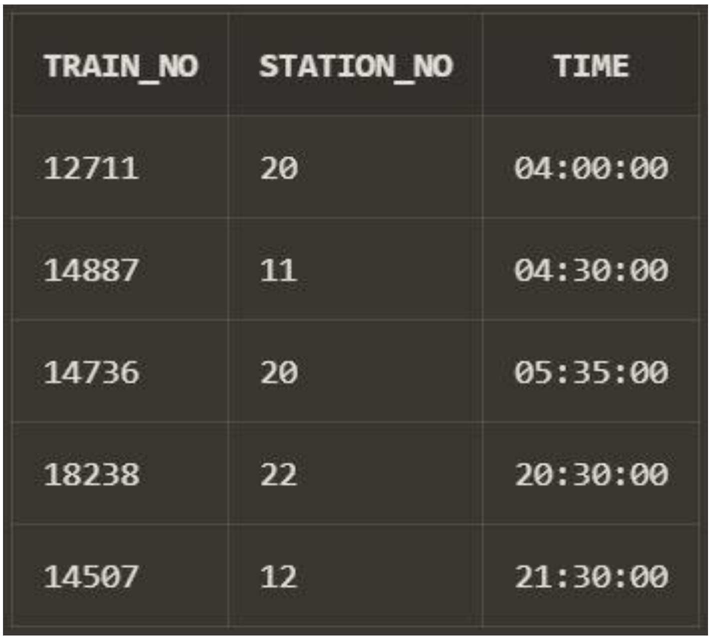
insert into REACHES values(14887,11,'04:30:00');

insert into REACHES values (14736,20,'05:35:00');

insert into REACHES values(18238,22,'20:30:00');

insert into REACHES values(14507,12,'21:30:00');

select * from REACHES;



TRAIN_NO	STATION_NO	TIME
12711	20	04:00:00
14887	11	04:30:00
14736	20	05:35:00
18238	22	20:30:00
14507	12	21:30:00

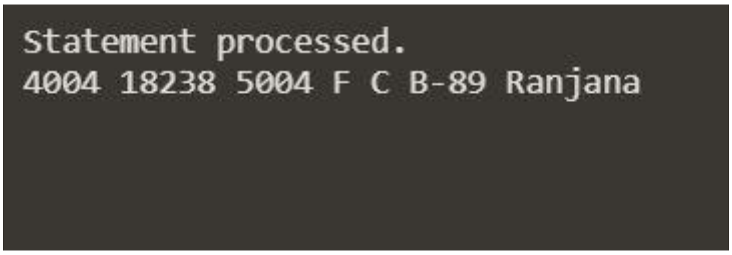
PL/SQL CODES

• Stored Procedures and Cursors

(i) Getting passengers info from train number

```
--getting passenger info from train no--

CREATE OR REPLACE PROCEDURE PASSENGER_INFO(dot date, tno int) AS

CURSOR C1 IS SELECT
t.id,tr.train_no,p.passenger_id,p.gender,p.reservation_status,p.seat_number,p.pname

FROM TRAIN tr, TICKET t, PASSENGER p

WHERE p.ticket_id=t.id AND

   t.train_no=tr.train_no AND

   dot=tr.rdate AND

   tno=tr.train_no;

rec C1%rowtype;

BEGIN

Open C1;

LOOP

fetch C1 into rec;

EXIT WHEN C1%NOTFOUND;

dbms_output.put_line(rec.id||' '||rec.train_no||' '||rec.passenger_id||' '||rec.gender||'
'||rec.reservation_status||' '||rec.seat_number||' '||rec.pname);

END LOOP;

close C1;

END;


Declare

dot date;

tno int;

Begin
```

```
dot:='24-may-2023';

tno:=18238;

PASSENGER_INFO(dot,tno);

End;
```



```
Statement processed.
4004 18238 5004 F C B-89 Ranjana
```

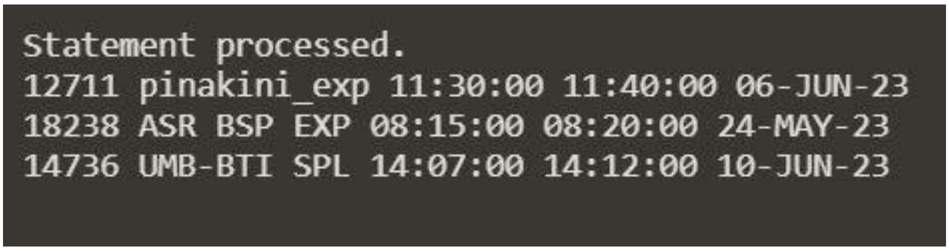
(ii) Getting train details by status

```
--getting train details by status

CREATE OR REPLACE PROCEDURE TRAINSBYSTATUS(IN_STATUS IN
VARCHAR) AS

CURSOR C2 is SELECT DISTINCT
t.train_no,t.train_name,t.arrival_time,t.departure_time,t.rdate

FROM TICKET P,TRAIN t

   WHERE P.train_no=t.train_no AND

   P.status=IN_STATUS;

rec C2%rowtype;

BEGIN

Open C2;

LOOP

fetch C2 into rec;

EXIT WHEN C2%NOTFOUND;

DBMS_OUTPUT.PUT_LINE(rec.train_no||' '||rec.train_name||' '||rec.arrival_time||'
'||rec.departure_time||' '||rec.rdate);

END LOOP;

CLOSE C2;

END;
```

BEGIN

TRAINSBYSTATUS('C');

END;

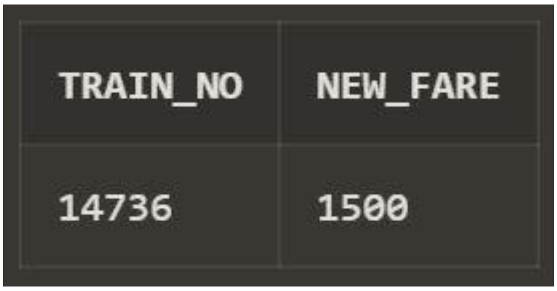


```
Statement processed.
12711 pinakini_exp 11:30:00 11:40:00 06-JUN-23
18238 ASR BSP EXP 08:15:00 08:20:00 24-MAY-23
14736 UMB-BTI SPL 14:07:00 14:12:00 10-JUN-23
```

• Triggers

Update TICKET_PRICE_HISTORY table when the price of the ticket is updated in TICKET

```
--CREATING TABLE TICKET_PRICE_HISTORY--
CREATE TABLE TICKET_PRICE_HISTORY
(
    train_no int,
    new_fare float
);

--CREATE A TRIGGER TRIGG_TICKET_PRICE_HISTORY--
CREATE OR REPLACE TRIGGER TRIGG_TICKET_PRICE_HISTORY
AFTER UPDATE OF fare1
ON TRAIN_STATUS
FOR EACH ROW
BEGIN
INSERT INTO TICKET_PRICE_HISTORY VALUES(:OLD.train_no,:OLD.fare1);
END;

--LET'S UPDATE A PRICE OR FARE OF A TICKET--
UPDATE
TRAIN_STATUS SET fare1=1500
WHERE train_no=14736;
select * from TICKET_PRICE_HISTORY;
```



TRAIN_NO	NEW_FARE
14736	1500

// price of train_no=14736 changed from 200 to 1500

Conclusion

In our project **Railway Management System** we have stored all the information about the Trains scheduled and the users booking tickets and even status of trains, seats, etc. This database is helpful for the applications which facilitate 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 the station for each and every query. We have considered the most important requirements only, many more features and details can be added to our project in order to obtain a more user friendly application. These applications are already in progress and in future they can be upgraded and may become part of amazing technology.

References

1. Sanjay Nishant, A. M. *"Railway Reservation System."* (2020).

2. MUSA, MUHAMMAD SANI. *“ONLINE TRAIN RESERVATION SYSTEM.”* (2015).

3. Jaekel, Birgit, and Thomas Albrecht. *"Operational railway management as part of an integrated railway management process." Aspekte der Verkehrstelematik–ausgewählte Veröffentlichungen 2014* (2014): 113.