Train Ticket Booking System

Description:

It’s a Booking System which provide Admin Easy of Maintaining data of Booked Tickets by Customer and also helps Customer to do their booking very Easily and Convenient way. With help of this System Admin can Generate various types of report which will help in day-to-day activities.

Requirements:

The requirement of this System is:

1. Admin:

Admin who wants to manage the data in Structured order and is able to generate Significant and beneficial report.

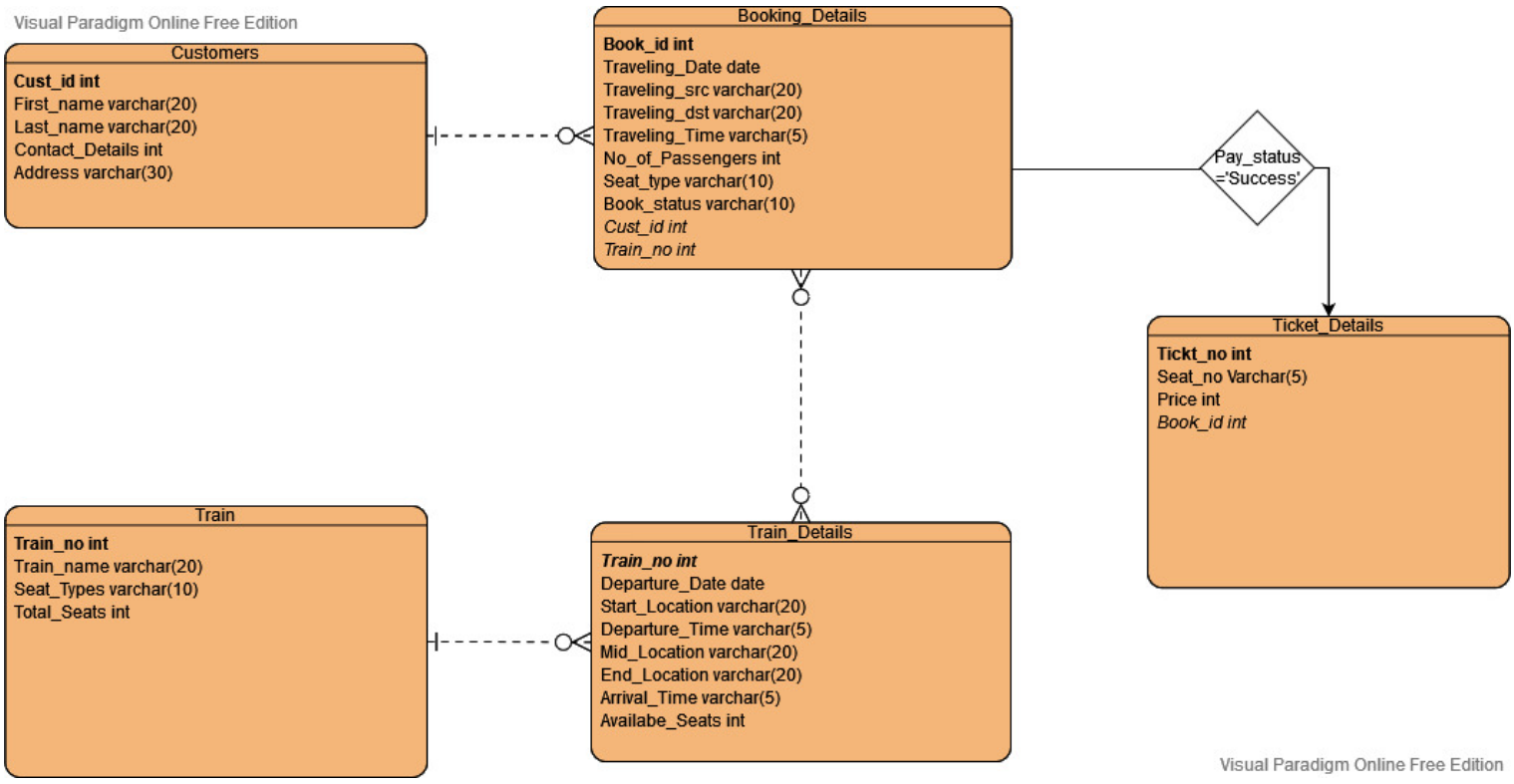
2. Customers:

Customers who will be able to book there tickets much easier and in Convenient way, as the customers will be able to see Train details and decide their Source and Destination of journey while proceeding their Bookings.

3. Interface:

Which will allow user to do their bookings through which data will be entered into the Database of Ticket Booking System.

ERD:


Visual Paradigm Online Free Edition
Customers
Cust_id int
First_name varchar(20)
Last_name varchar(20)
Contact_Details int
Address varchar(30)
Booking_Details
Book_id int
Traveling_Date date
Traveling_src varchar(20)
Traveling_dst varchar(20)
Traveling_Time varchar(5)
No_of_Passengers int
Seat_type varchar(10)
Book_status varchar(10)
Cust_id int
Train_no int
Pay_status ='Success'
Ticket_Details
Tickt_no int
Seat_no Varchar(5)
Price int
Book_id int
Train
Train_no int
Train_name varchar(20)
Seat_Types varchar(10)
Total_Seats int
Train_Details
Train_no int
Departure_Date date
Start_Location varchar(20)
Departure_Time varchar(5)
Mid_Location varchar(20)
End_Location varchar(20)
Arrival_Time varchar(5)
Availabe_Seats int
Visual Paradigm Online Free Edition

Tables:

1. Customer Details

-- Creating new Table Customers Details

CREATE TABLE customers_details

(cust_id int NOT NULL CONSTRAINT pk_cid PRIMARY KEY,first_name varchar(20), last_name varchar(20), contact_details int constraint conct_dtls check(contact_details>9), address varchar(30));

```
Name            Null?    Type
--------------- -------- ------------
CUST_ID         NOT NULL NUMBER(38)
FIRST_NAME               VARCHAR2(20)
LAST_NAME                VARCHAR2(20)
CONTACT_DETAILS          NUMBER(38)
ADDRESS                  VARCHAR2(30)
```

2. Train:

-- Creating new Table Train

CREATE TABLE train

(train_no int NOT NULL CONSTRAINT pk_tno PRIMARY Key, train_name varchar(20), seat_types VARCHAR(10), total_seats INT);

```
Name         Null?    Type
------------ -------- ------------
TRAIN_NO     NOT NULL NUMBER(38)
TRAIN_NAME            VARCHAR2(20)
SEAT_TYPES            VARCHAR2(10)
TOTAL_SEATS           NUMBER(38)
```

3. Train Details:

-- Creating new Table Train Details

CREATE TABLE train_details

(train_no int CONSTRAINT pk_stid PRIMARY KEY CONSTRAINT fk_trno REFERENCES train(train_no), departure_date date, start_location varchar(20), departure_time VARCHAR(10), mid_location VARCHAR(20), end_location VARCHAR(20), arrival_time VARCHAR(10), available_seats int);

```
Name            Null?    Type
--------------- -------- ------------
TRAIN_NO        NOT NULL NUMBER(38)
DEPARTURE_DATE           DATE
START_LOCATION           VARCHAR2(20)
DEPARTURE_TIME           VARCHAR2(10)
MID_LOCATION             VARCHAR2(20)
END_LOCATION             VARCHAR2(20)
ARRIVAL_TIME             VARCHAR2(10)
AVAILABLE_SEATS          NUMBER(38)
```

4. Booking Details:

-- Creating new Table Booking Details

CREATE table booking_details (book_id int NOT NULL Constraint pk_bkid primary key, traveling_date date, traveling_src varchar(20), traveling_dst varchar(20), traveling_time varchar(10), no_of_passengers int, seat_type varchar(10), book_status VARCHAR(10) DEFAULT 'Pending',cust_id int constraint fk_cid REFERENCES customers_details(cust_id), train_no int CONSTRAINT fk_train_no REFERENCES train_details(train_no));

```
Name             Null?    Type
---------------- -------- ------------
BOOK_ID          NOT NULL NUMBER(38)
TRAVELING_DATE            DATE
TRAVELING_SRC             VARCHAR2(20)
TRAVELING_DST             VARCHAR2(20)
TRAVELING_TIME            VARCHAR2(10)
NO_OF_PASSENGERS          NUMBER(38)
SEAT_TYPE                 VARCHAR2(10)
BOOK_STATUS               VARCHAR2(10)
CUST_ID                   NUMBER(38)
TRAIN_NO                  NUMBER(38)
```

5. Payment:

-- Creating new Table Payment Status

Create table payment

(pay_id int constraint pk_pid primary key, pay_mode varchar(10), pay_status varchar(10), book_id int constraint fk_bid REFERENCES booking_details(book_id));

```
Name        Null?    Type
----------- -------- ------------
PAY_ID      NOT NULL NUMBER(38)
PAY_MODE             VARCHAR2(10)
PAY_STATUS           VARCHAR2(10)
BOOK_ID              NUMBER(38)
```

6. Ticket Details:

-- Creating new Table Ticket Details

CREATE table ticket_details (ticket_no int NOT NULL constraint pk_tcktno Primary Key, seat_no varchar(20) CONSTRAINT uq_stno unique, price INT, book_id int CONSTRAINT fk_bkid references booking_details(book_id));

```
Name      Null?    Type
--------- -------- ------------
TICKET_NO NOT NULL NUMBER(38)
SEAT_NO            VARCHAR2(20)
PRICE              NUMBER(38)
BOOK_ID            NUMBER(38)
```

Reports:

1. Display Records of Customer who have done Booking:

--Display All the Customer Details who have done Bookings

select customers_details.cust_id, customers_details.first_name, customers_details.contact_details, customers_details.address, booking_details.book_id from booking_details left join customers_details on booking_details.cust_id = customers_details.cust_id;

Output:

	CUST_ID	FIRST_NAME	CONTACT_DETAILS	ADDRESS	BOOK_ID	TRAVELING_DATE	TRAVELING_SRC	TRAVELING_DST	NO_OF_PASSENGERS
1	1	Ajay	98765432	Mumbai	1	01-10-21	Margao	Manglore	1
2	2	Harshad	9115463	Gujrat	2	01-10-21	Margao	Manglore	1
3	3	Mannu	945624	Mumbai	3	03-10-21	VASCO DA GAMA	BRAHMAPUR	1
4	4	Bhushan	985647	Gujrat	4	04-10-21	Margao	C SHIVAJI MAH T	3
5	5	Rakesh	91678	Hyderabad	5	04-10-21	CASTLE ROCK	YESVANTPUR JN	2

2. Display Records of Specific Customer and their Booking Details:

select * from customers_details c left join booking_details b on b.cust_id = c.cust_id where b.book_id = 1;

Output:

	CUST_ID	FIRST_NAME	LAST_NAME	CONTACT_DETAILS	ADDRESS	BOOK_ID	TRAVELING_DATE	TRAVELING_SRC	TRAVELING_DST	TRAVELING_TIME	NO_OF_PASSENGERS	SEAT_TYPE	BOOK_STATUS	CUST_ID_1	TRAIN_NO
1	1	Ajay	Kedia	98765432	Mumbai	1	01-10-21	Margao	Manglore	2:30PM	1	GN	Confirm	1	6601

3. Display the Count of Customer whose address are same:

select count(cust_id), address from customers_details GROUP BY address;

Output:

	COUNT(CUST_ID)	ADDRESS
1	2	Mumbai
2	1	Hyderabad
3	2	Gujrat

4. Display the Count of Customers who are traveling from same Start location:

SELECT count(book_id), traveling_src from booking_details GROUP BY traveling_src;

Output:

	COUNT(CUST_ID)	ADDRESS
1	2	Mumbai
2	1	Hyderabad
3	2	Gujrat

5. Display records of Customers and their Booking who are traveling with more the one passenger:

select * from customers_details c,booking_details b where c.cust_id = b.cust_id and b.no_of_passengers > 1 order by b.cust_id asc;

Output:

	CUST_ID	FIRST_NAME	LAST_NAME	CONTACT_DETAILS	ADDRESS	BOOK_ID	TRAVELING_DATE	TRAVELING_SRC	TRAVELING_DST	TRAVELING_TIME	NO_OF_PASSENGERS	SEAT_TYPE	BOOK_STATUS	CUST_ID_1	TRAIN_NO
1	4	Bhushan	Bhatt	985647	Gujrat	4	04-10-21	Margao	C SHIVAJI MAH T	7:30AM	3	AC	Confirm	4	1152
2	5	Rakesh	Junjunwala	91678	Hyderabad	5	04-10-21	CASTLE ROCK	YESVANTPUR JN	11:05PM	2	GN	Confirm	5	7340

6. Display the Ticket Details of all the Customers who Booking Stats has been Confirmed.

select * from ticket_details t left join booking_details b on t.book_id = b.book_id where b.book_id = (SELECT book_id FROM payment where pay_status = 'Successful' and book_id = b.book_id);

Output:

	TICKET_NO	SEAT_NO	PRICE	BOOK_ID	BOOK_ID_1	TRAVELING_DATE	TRAVELING_SRC	TRAVELING_DST	TRAVELING_TIME	NO_OF_PASSENGERS	SEAT_TYPE	BOOK_STATUS	CUST_ID	TRAIN_NO
1	1001	G1	200	1	1	01-10-21	Margao	Manglore	2:30PM	1	GN	Confirm	1	6601
2	1002	SL1	400	3	3	03-10-21	VASCO DA GAMA	BRAHMAPUR	7:00AM	1	SL	Confirm	3	8048
3	1003	AC1 AC2 AC3	3600	4	4	04-10-21	Margao	C SHIVAJI MAH T	7:30AM	3	AC	Confirm	4	1152
4	1004	G2 G3	400	5	5	04-10-21	CASTLE ROCK	YESVANTPUR JN	11:05PM	2	GN	Confirm	5	7340

7. Display records of Booking on Specific Date.

select * from booking_details where traveling_date = '04-10-21';

Output:

	BOOK_ID	TRAVELING_DATE	TRAVELING_SRC	TRAVELING_DST	TRAVELING_TIME	NO_OF_PASSENGERS	SEAT_TYPE	BOOK_STATUS	CUST_ID	TRAIN_NO
1	4	04-10-21	Margao	C SHIVAJI MAH T	7:30AM	3	AC	Confirm	4	1152
2	5	04-10-21	CASTLE ROCK	YESVANTPUR JN	11:05PM	2	GN	Confirm	5	7340

8. Display all the Train and its Details.

select t.train_no, t.train_name, t.seat_types, td.departure_date, td.start_location, td.departure_time, td.mid_location, td.end_location, td.arrival_time, t.total_seats, td.available_seats from train t left join train_details td on t.train_no = td.train_no;

Output:

	TRAIN_NO	TRAIN_NAME	SEAT_TYPES	DEPARTURE_DATE	START_LOCATION	DEPARTURE_TIME	MID_LOCATION	END_LOCATION	ARRIVAL_TIME	TOTAL_SEATS	AVAILABLE_SEATS
1	1152	JANSHATABDI SPL	GN SL AC	04-10-21	Margao	7:30AM	RATNAGIRI	C SHIVAJI MAH T	2:00PM	15	15
2	6601	MAO MAQ SPL	GN SL AC	01-10-21	Margao	2:30PM	Karwar	Manglore	11:30PM	15	15
3	7340	VSG YPR EXP	GN SL AC	04-10-21	VASCO DA GAMA	11:05PM	CASTLE ROCK	YESVANTPUR JN	12:35PM	15	13
4	8048	VSG HWH SPL	GN SL AC	03-10-21	VASCO DA GAMA	7:00AM	BRAHMAPUR	HOWRAH JN	5:00AM	15	15

9. Display the Total amount of Fare price based on Train No.

select b.train_no,sum(price) from ticket_details t, booking_details b where t.book_id = b.book_id group by b.train_no;

Output:

	TRAIN_NO	SUM(PRICE)
1	6601	200
2	1152	3600
3	7340	400
4	8048	400

10. Display Updated Booking Status as Payment is done Successfully.

```
-- Triggers to Update Booking Status
CREATE OR REPLACE TRIGGER booking_status
BEFORE DELETE OR INSERT OR UPDATE
of book_status
ON booking_details
FOR EACH ROW
When(NEW.book_id > 0)
Begin
   dbms_output.put_line('Booking ID: '||:NEW.book_id);
   dbms_output.put_line('OLD Booking Status: '||:OLD.book_status);
   dbms_output.put_line('New Booking Status: '||:NEW.book_status);
   dbms_output.put_line('');
END;
/
```

Output:

```
Booking ID: 1
OLD Booking Status: Pending
New Booking Status: Confirm

Booking ID: 3
OLD Booking Status: Pending
New Booking Status: Confirm

Booking ID: 5
OLD Booking Status: Pending
New Booking Status: Confirm

Booking ID: 4
OLD Booking Status: Pending
New Booking Status: Confirm



4 rows updated.
```

11. Display updated Available Seats as Booking Status gets Confirmed.

```
--Trigger to update Available Seats
CREATE OR REPLACE TRIGGER available_seats
BEFORE DELETE OR INSERT OR UPDATE
of available_seats
ON train_details
FOR EACH ROW
WHEN(old.available_seats >0)
Declare
x number:=0;
Begin
    x := :old.train_no;
    dbms_output.put_line('Train.NO: '||x);
    dbms_output.put_line('OLD Available Seats: '||:OLD.available_seats);
    dbms_output.put_line('New Available Seats: '||:NEW.available_seats);
END;
/
```

```
Train.NO: 6601
OLD Available Seats: 15
New Available Seats: 14


1 row updated.
```

12. Updated Booking Status and Available Seats.

Booking status is updated to confirm when Payment is done successfully and Available seats are decremented by Total No. of passengers travelling in a particular Train.

```
--Procedure For Updating Booking Status and Available Seats When Payment Is Successful

Declare

   pid payment.pay_id%type:=1;

   pay_stat payment.pay_status%type;

   bkid payment.book_id%type;


   b_bkid BOOKING_DETAILS.book_id%type:=1;

   book_stat BOOKING_DETAILS.book_status%type;

   pssngr BOOKING_DETAILS.no_of_passengers%type;

   trno BOOKING_DETAILS.train_no%type;


   seats train_details.available_seats%type;

   td_trno train_details.train_no%type;

Begin


   -- Loop to go through all the values in the Tables

   LOOP

      select pay_id, pay_status, book_id into pid, pay_stat, bkid from payment where pay_id =
pid;

      select book_id, book_status, no_of_passengers, train_no into b_bkid, book_stat, pssngr,
trno from booking_details where book_id = bkid;

      select train_no, available_seats into td_trno, seats from train_details where train_no =
trno;


      -- Check if the payment is successful then Confoirm the Booking Status

      IF pay_stat = 'Paid' THEN
```

```
dbms_output.put_line('Pay ID: ' || pid || ' Pay Status:' || pay_stat);

dbms_output.put_line('');

update booking_details set book_status = 'Confirm' where book_id = (select book_id
from payment where pay_status= 'Paid' and book_id = bkid)

and traveling_date = any(select departure_date from train_details where
departure_date = booking_details.traveling_date) and train_no  = (select train_no from
train_details where available_seats > 0 and train_no = booking_details.train_no);


--Check the Available Seats

IF seats <= 0 and pay_stat = 'Paid' THEN


--If the available seats are Less then zero then Updating Available Seats to 0

update train_details set available_seats = 0 where available_seats <= 0;


dbms_output.put_line('Seats are Full for Train No: ' || td_trno);

dbms_output.put_line('');

update booking_details set book_status = 'Waiting' where train_no = (select
train_no from train_details where available_seats <= 0 and train_no =
booking_details.train_no)

and book_id = (select book_id from payment where pay_status  = 'Paid' and
book_id = booking_details.book_id);


ELSE

--If the available seats are Greater then zero then Deduction of Available Seats
With No_OF Passengers for a Particular Train

update train_details set available_seats = (Select t.available_seats - (Select
sum(no_of_passengers) From booking_details Where book_status = 'Confirm' and train_no =
t.train_no) as available_seats

From train_details t where train_no = trno) where train_no = trno;

dbms_output.put_line('');

END IF;


--If payment status is failed the Updating Booking Status to Failed

ELSIF pay_stat = 'Failed' THEN
```

```
        update booking_details set book_status = 'Failed' where book_id = (select pay_id
from payment where book_id = booking_details.book_id and pay_status = 'Failed');


      END IF;


      pid:= pid+1;
      IF pid > 10 THEN
      EXIT;


      END IF;
   END LOOP;


   Exception
   when no_data_found then
      dbms_output.put_line('Data Updated Successfuly');
   When others then
      dbms_output.put_line('Error!');
End;
/


--Procedure to View Ticket Details to Customer Based on Booking ID
Accept inpt Number PROMPT 'ENTER Book ID: ';
Declare
   user_input Number:= &inpt;
   tickt ticket_details%ROWTYPE;
Begin


   select * into tickt from ticket_details where book_id = user_input;
   dbms_output.put_line('Ticket_no: ' || tickt.ticket_no || ' Seat No: ' || tickt.seat_no || ' Price: '
|| tickt.price);
```

```
    Exception
    WHEN OTHERS THEN
    dbms_output.put_line('ERROR!');
End;
/

--Procedure to Calculate Ticket Price
Accept inpt Number Prompt 'Enter Book_Id';
Declare
    book booking_details%rowtype;
    ticket ticket_details%rowtype;
    price ticket_details.price%type;
    x number:= &inpt;
Begin
    select * into book from booking_details where book_status = 'Confirm' and book_id = x;
    select * into ticket from ticket_details where book_id = x;

    dbms_output.put_line('No of Passengers: ' || book.no_of_passengers );

    If book.seat_type = 'GN' then

        update ticket_details set price = 200 * book.no_of_passengers where book_id = x;

    ELSIF book.seat_type = 'SL' then

        update ticket_details set price = 400 * book.no_of_passengers where book_id = x;

    ELSIF book.seat_type = 'AC' then
```

```
    update ticket_details set price = 800 * book.no_of_passengers where book_id = x;

  End if;

  select price into price from ticket_details where book_id = x;
  dbms_output.put_line('Updated Price: ' || price);

  Exception
  when no_data_found then
    dbms_output.put_line('No such Book_ID');
  When others then
    dbms_output.put_line('Error!');
End;
/
```

Output:

TRAIN_NO	DEPARTURE_DATE	START_LOCATION	DEPARTURE_TIME	MID_LOCATION	END_LOCATION	ARRIVAL_TIME	AVAILABLE_SEATS
6601	01-10-21	Margao	2:30PM	Karwar	Manglore	11:30PM	1
8048	03-10-21	VASCO DA GAMA	7:00AM	BRAHMAPUR	HOWRAH JN	5:00AM	1
1152	04-10-21	Margao	7:30AM	RATNAGIRI	C SHIVAJI MAH T	2:00PM	0
7340	04-10-21	VASCO DA GAMA	11:05PM	CASTLE ROCK	YESVANTPUR JN	12:35PM	0

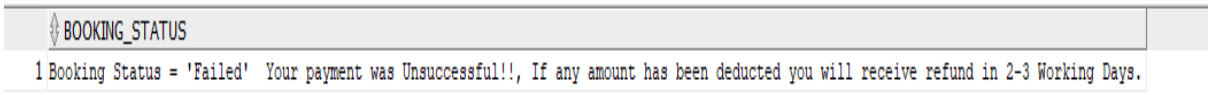
BOOK_ID	TRAVELING_DATE	TRAVELING_SRC	TRAVELING_DST	TRAVELING_TIME	NO_OF_PASSENGERS	SEAT_TYPE	BOOK_STATUS	CUST_ID	TRAIN_NO
1	01-10-21	Margao	Manglore	2:30PM	1	GN	Confirm	1	6601
2	01-10-21	Margao	Manglore	1:30PM	1	GN	Failed	2	6601
3	03-10-21	VASCO DA GAMA	BRAHMAPUR	7:00AM	1	SL	Confirm	3	8048
4	04-10-21	Margao	C SHIVAJI MAH T	7:30AM	3	AC	Waiting	4	1152
5	04-10-21	CASTLE ROCK	YESVANTPUR JN	11:05PM	2	GN	Waiting	5	7340

```
Pay ID: 1 Pay Status:Paid

Booking ID: 1
OLD Booking Status: Confirm
New Booking Status: Confirm

Train.NO: 6601
OLD Available Seats: 2
New Available Seats: 1

Booking ID: 2
OLD Booking Status: Failed
New Booking Status: Failed

Pay ID: 3 Pay Status:Paid

Booking ID: 3
OLD Booking Status: Confirm
New Booking Status: Confirm

Train.NO: 8048
OLD Available Seats: 2
New Available Seats: 1

Pay ID: 4 Pay Status:Paid

Seats are Full for Train No: 1152

Booking ID: 4
OLD Booking Status: Waiting
New Booking Status: Waiting

Booking ID: 5
OLD Booking Status: Confirm
New Booking Status: Waiting

Pay ID: 5 Pay Status:Paid

Seats are Full for Train No: 7340


Data Updated Successfuly


PL/SQL procedure successfully completed.
```

13. Display Message to user of Booking Status after payment.

```
--Case Statement for Checking Booking Status
Select
   Case
   When book_status = 'Failed' and pay_status = 'Failed' Then
      'Booking Status = "Failed"  Your payment was Unsuccessful!!, If any amount has been deducted you will receive refund in 2-3 Working Days.'
   Else
      'Your Ticket has been booked Sussessfully, Your will receive your Ticket Shortly..'
   End as Booking_Status
From booking_details b, payment p where b.book_id = p.book_id and b.book_id = 2;
```

Output:



	BOOKING_STATUS
1	Booking Status = 'Failed' Your payment was Unsuccessful!!, If any amount has been deducted you will receive refund in 2-3 Working Days.

14. Update the Fare Price based on Seat Type chosen by Customers based on No of passengers travelling.

```
--Procedure to Calculate Ticket Price
Accept inpt Number Prompt 'Enter Book_Id';
Declare
   book booking_details%rowtype;
   ticket ticket_details%rowtype;
   price ticket_details.price%type;
   x number:= &inpt;
Begin
   select * into book from booking_details where book_status = 'Confirm' and book_id = x;
   select * into ticket from ticket_details where book_id = x;

   dbms_output.put_line('No of Passengers: ' || book.no_of_passengers );

   If book.seat_type = 'GN' then

      update ticket_details set price = 200 * book.no_of_passengers where book_id = x;

   ELSIF book.seat_type = 'SL' then

      update ticket_details set price = 400 * book.no_of_passengers where book_id = x;

   ELSIF book.seat_type = 'AC' then

      update ticket_details set price = 800 * book.no_of_passengers where book_id = x;

   End if;

   select price into price from ticket_details where book_id = x;
```

```
    dbms_output.put_line('Updated Price: ' || price);


    Exception
    when no_data_found then
        dbms_output.put_line('No such Book_ID');
    When others then
        dbms_output.put_line('Error!');
End;
/
```

Output:

```
No of Passengers: 1
Updated Price: 200


PL/SQL procedure successfully completed.
```

15. Display Ticket Details of Customer.

```
--Procedure to View Ticket Details to Customer Based on Booking ID
Accept inpt Number PROMPT 'ENTER Book ID: ';
Declare
    user_input Number:= &inpt;
    tickt ticket_details%ROWTYPE;
Begin


    select * into tickt from ticket_details where book_id = user_input;
    dbms_output.put_line('Ticket_no: ' || tickt.ticket_no || ' Seat No: ' || tickt.seat_no || ' Price: '
|| tickt.price);
```

```
    Exception
    WHEN OTHERS THEN
    dbms_output.put_line('ERROR!');
End;
/
```

Output:

```
Ticket_no: 1001 Seat No: G1 Price: 200


PL/SQL procedure successfully completed.
```

SQL File

https://drive.google.com/file/d/1mrnLtJvBo15kUWQ3_DzJsmDqhhcVeY-R/view?usp=sharing