A Project Activity Report

Submitted for Database Management System

TOPIC- FLIGHT MANAGEMENT SYSTEM

THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY,
(A DEEMED TO BE UNIVERSITY), PATIALA, PUNJAB INDIA
Jan - May 2024

INDEX

WHY NEED IT?

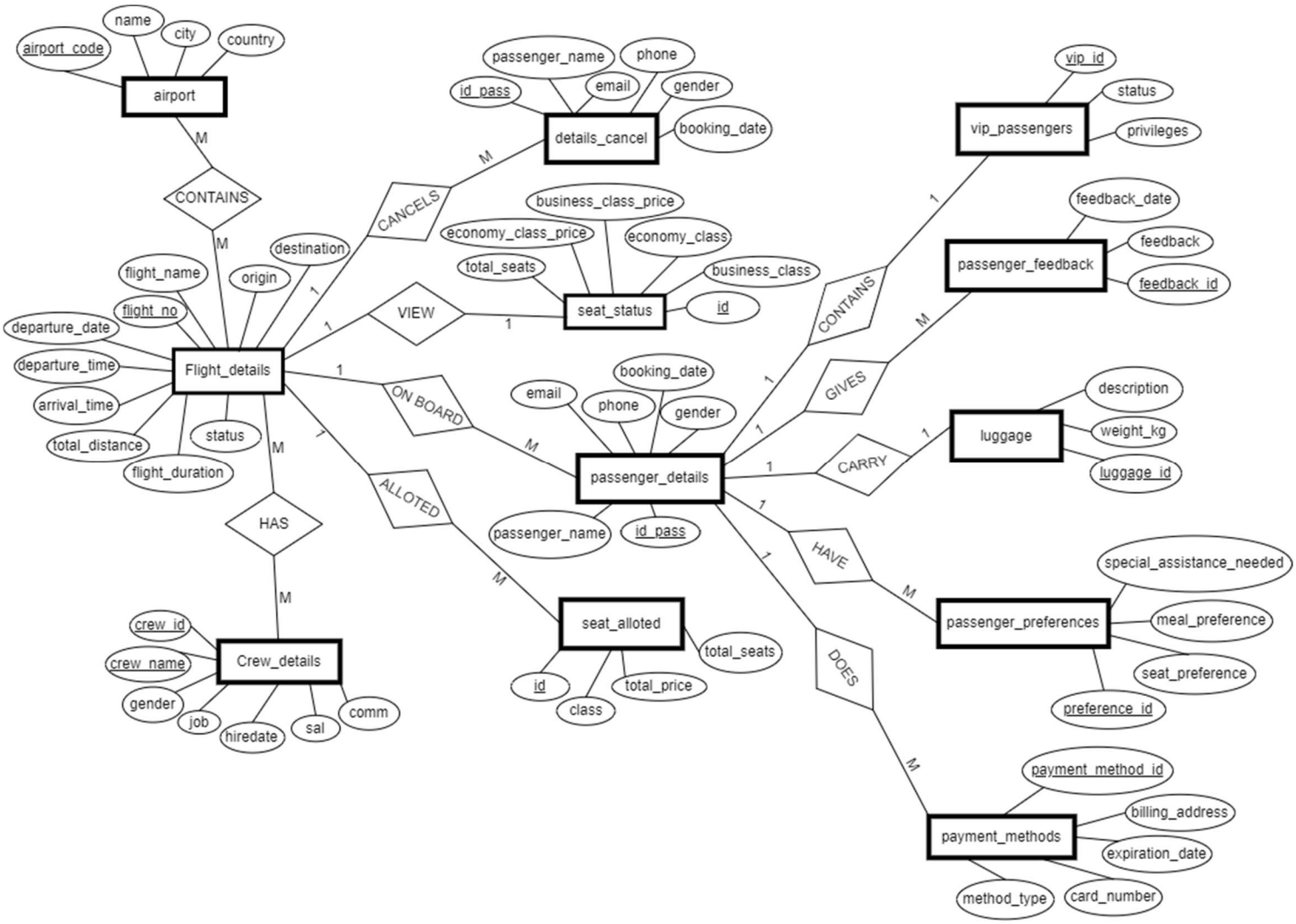
Flight management systems grapple with the complexity of handling diverse flight-related data such as schedules, aircraft specifications, crew assignments, and passenger details. Manual oversight of these resources poses challenges and inefficiencies.

To overcome these hurdles, the project aims to craft a robust flight management system adept at managing this array of data seamlessly. It will encompass comprehensive storage of flight-related information, including aircraft particulars, routes, crew rosters, and passenger manifests.

This system must possess the capability to efficiently process large volumes of data while remaining adaptable to future growth in air traffic and fleet expansion. It's imperative to prioritize the system's design to ensure data integrity, coherence, and dependability.

Ultimately, the flight management system endeavour to elevate operational effectiveness, bolster safety measures, and elevate the overall experience for airline personnel and passengers alike.

ER DIAGRAM


airport
airport_code
name
city
country
CONTAINS
M
M
Flight_details
flight_name
flight_no
departure_date
departure_time
arrival_time
total_distance
flight_duration
status
origin
destination
1
CANCELS
M
details_cancel
passenger_name
phone
id_pass
email
gender
booking_date
VIEW
1
1
seat_status
business_class_price
economy_class_price
economy_class
total_seats
business_class
id
ON BOARD
1
M
passenger_details
email
phone
booking_date
gender
passenger_name
id_pass
ALLOTED
1
M
seat_alloted
total_seats
id
class
total_price
HAS
M
1
Crew_details
crew_id
crew_name
gender
job
hiredate
sal
comm
CONTAINS
1
1
vip_passengers
vip_id
status
privileges
GIVES
M
1
passenger_feedback
feedback_date
feedback
feedback_id
CARRY
1
1
luggage
description
weight_kg
luggage_id
HAVE
1
M
passenger_preferences
special_assistance_needed
meal_preference
seat_preference
preference_id
DOES
1
M
payment_methods
payment_method_id
billing_address
expiration_date
method_type
card_number

ER TO TABLE

- Flight_details (flight_no, flight_name, origin, destination, departure_date, departure_name, arrival_date, arrival_time, total_distance, flight_duration, status)

- Crew_details (crew_id, crew_name, gender, job, hiredate, sal, comm)

- Passenger_details (passenger_name, email, phone_number, gender, booking_date)

- Seat_status (id, business_class, business_class_price, economy_class, economy_class_price, total_seats)

- Seat_alloted (flight_no, class, total_seats, total_price)

- Details_cancel (passenger_name, email, phone_number, gender, booking_date)

- Airport(airport_code,name,city,country)

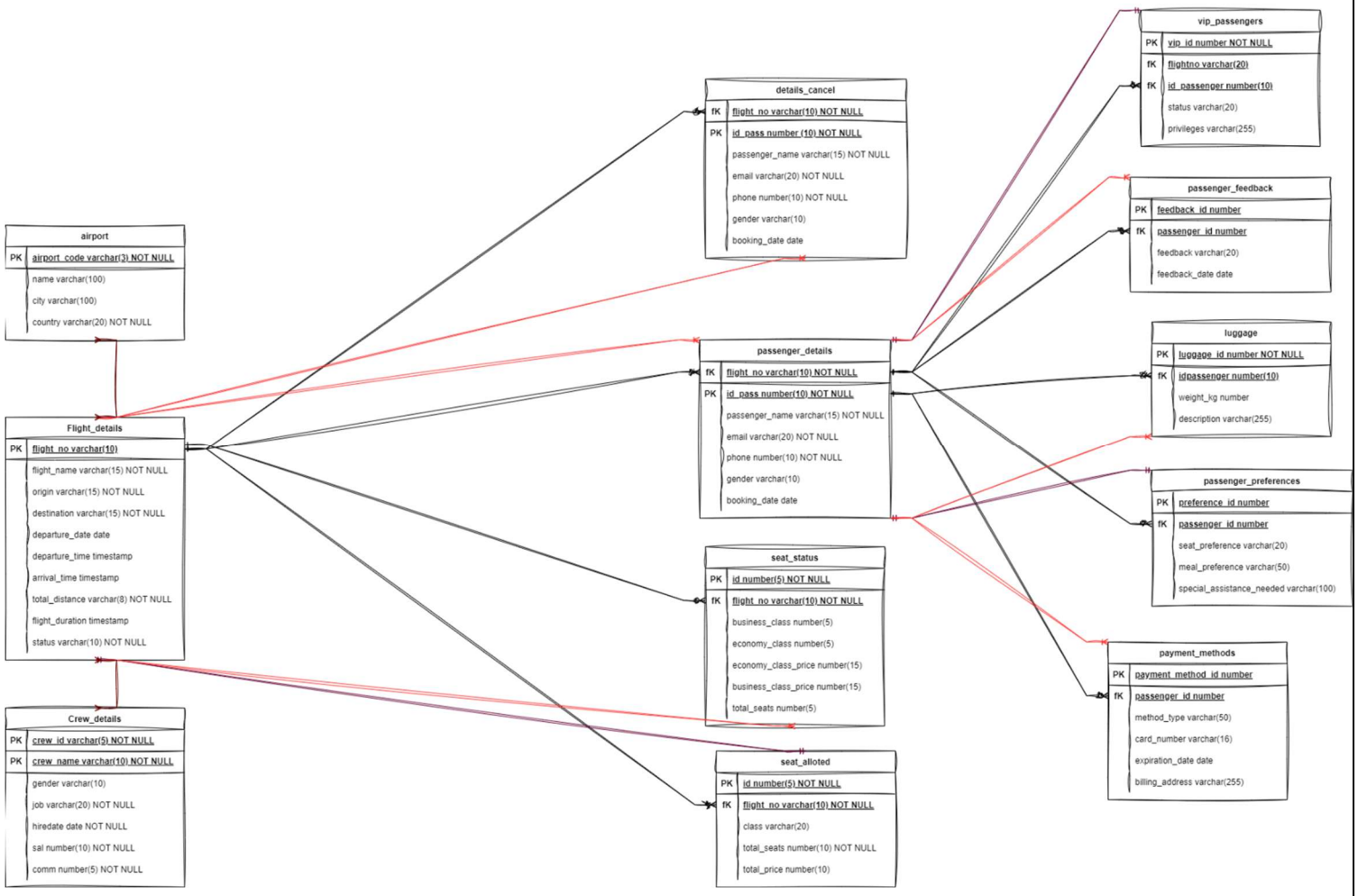
- Luggage(luggage_id,idpassenger,weight_kg,description)

- Passenger_preferences(preference_id,passenger_id,seat_preference,meal_preference,special_assistance_needed)

- Payment_methods(payment_methods_id,passenger_id,method_type,card_number,expiration_data,billing_address)

- Passenger_feedback(feedback_id,passenger_id,feedback,feedback_date)

- Vip_passengers(vip_id,flightno,id_passenger,status,privileges)



RELATIONSHIP

- Flight_details has one to many relationship with Passenger_details.
- Flight_details has one to many relationship with Seat_status.
- Flight_details has one to one relationship with Seat_alloted.
- Flight_details has one to many relationship with Details_cancel.
- Flight_details has many to many relationship with crew_details.
- Passenger_details has one to one relationship with vip_passengers.
- Passenger_details has one to one relationship with passengers_preference.
- Passenger_details has one to many relationship with passengers_feedback.
- Passenger_details has one to many relationship with luggage.
- Passenger_details has one to many relationship with payment_method.
- Airport has many to many relationship with Flight_details.

NORMALIZATION

FIRST NORMAL FORM:

The first normal form rule is that there should be no nesting or repeating groups in a table. Now an entity type that contains only one value for an attribute in an entity instance ensures the application of first normal form for the entity type. So, in a way any entity type with an entity identifier is by default in first normal form.
Hence, all tables in our relational schema are in 1st normal form as none of the tables contain any composite attribute or multi-valued attribute or their combinations. As an example, let's look at the passenger_details table below:

Name	Null?	Type
FLIGHT_NO		VARCHAR2(10)
PASSENGER_NAME	NOT NULL	VARCHAR2(15)
ID_PASS	NOT NULL	NUMBER(10)
EMAIL	NOT NULL	VARCHAR2(20)
PHONE	NOT NULL	NUMBER(10)
GENDER		VARCHAR2(10)
BOOKING_DATE		DATE

In the Table passenger_details there is a multiple attribute named as Phone due to which this table is not in 1NF. To convert this table to 1NF Table is broken into 2 atomic tables which are shown below.

Table passenger containing all the information except passenger phone number:

Name	Null?	Type
FLIGHT_NO		VARCHAR2(20)
PASSENGER_NAME		VARCHAR2(20)
ID_PASS		NUMBER(10)
EMAIL		VARCHAR2(20)
GENDER		VARCHAR2(10)
BOOKING_DATE		DATE

Table passengerphone containing phone number of passengers:

```
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
IDPASS                                             NUMBER(10)
PHONE                                              NUMBER(10)
```

SECOND NORMAL FORM:

The second normal form rule is that the key attributes determine all non-key attributes. A violation of second normal form occurs when there is a composite key, and part of the key determines some non-key attributes. The second normal form deals with the situation when the entity identifier contains two or more attributes, and the non-key attribute depends on part of the entity identifier. Let us look at table crew_details where:

(crew_id,crew_name)->{gender,job,hiredate,sal,comm}
{crew_id}->{job}

Original table:

crew_id	crew_name	gender	job	hiredate	sal	comm
C001	John	Male	Pilot	2023-01-15	80000	5000
C002	Mary	Female	Flight Attendant	2023-03-20	35000	2000
C003	Alex	Male	Mechanic	2023-05-10	45000	3000

The table already appears to be in 1NF since each cell contains a single value, and there are no repeating groups.In 2NF, we need to ensure that each non-prime attribute is fully functionally dependent on the entire primary key. In this case, the job attribute is dependent only on part of the primary key (crew_id).We need to split the table to remove this partial dependency.

Now, the crew_details table contains information only about the crew members themselves, while the job_details table contains information about their respective roles. This separation ensures that each table contains data without any partial dependencies on the primary key.

Normalized Tables:

Table 1: crew_details

crew_id	crew_name	gender	hiredate	sal	comm
C001	John	Male	2023-01-15	80000	5000
C002	Mary	Female	2023-03-20	35000	2000
C003	Alex	Male	2023-05-10	45000	3000

Table 2: job_details

crew_id	job
C001	Pilot
C002	Flight Attendant
C003	Mechanic

THIRD NORMAL FORM:

The third normal form rule is that the non-key attributes should be independent. This normal form is violated when there exists a dependency among non-key attributes in the form of a transitive dependency. Since a transitive dependency can be resolved by moving the dependency attributes to a new entity type with one-to-many relationship, we have used this property to solve the dependencies in our tables, making them in accordance with the 3rd normal form.

For 3NF following are the necessary conditions:

- If table is in 2NF.
- No non-key attribute is transitively dependent on the primary key.

Let us look at table flight_details:

```
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
FLIGHT_NO                                 NOT NULL VARCHAR2(10)
FLIGHT_NAME                               NOT NULL VARCHAR2(15)
ORIGIN                                    NOT NULL VARCHAR2(15)
DESTINATION                               NOT NULL VARCHAR2(15)
DEPARTURE_DATE                                     DATE
DEPARTURE_TIME                                     TIMESTAMP(6)
ARRIVAL_TIME                                       TIMESTAMP(6)
TOTAL_DISTANCE                            NOT NULL VARCHAR2(8)
FLIGHT_DURATION                                    TIMESTAMP(6)
STATUS                                    NOT NULL VARCHAR2(10)
```

1. The table already appears to be in 2NF since it has a composite primary key (flight_no, departure_date), and each non-prime attribute is fully functionally dependent on the entire primary key.
2. Now, let's check for any transitive dependencies:

- total_distance and flight_duration both seem to be functionally dependent on flight_no alone, not on each other or any other non-prime attribute.
- flight_name, origin, destination, departure_time, arrival_time, and status also appear to be functionally dependent only on flight_no.

Since there are no transitive dependencies, the table is already in 3NF.

Similarly, all other tables are also in 3NF.

PL/SQL CODES

CREATING TABLES:

1. FLIGHT DETAILS:

```
SQL> create table flight_details
  2      (
  3          flight_no varchar(10) primary key,
  4          flight_name varchar(15) not null,
  5          origin varchar(15)  not null,
  6          destination varchar(15) not null,
  7          departure_date date,
  8          departure_time timestamp,
  9          arrival_time timestamp,
 10          total_distance varchar(8) not null,
 11          flight_duration timestamp,
 12          status varchar(10) not null
 13          );

Table created.
```

```
SQL> insert into flight_details
  2      values('AI7012','Air Indigo','Amritsar','Delhi','2-apr-24','2-apr-24 09:05:00','2-apr-24 10:50:00','200km','2-a
pr-24 01:45:00','Ontime');

1 row created.

SQL>
SQL> insert into flight_details(flight_no,flight_name,origin,destination,departure_date,departure_time,arrival_time,tota
l_distance,flight_duration,status)
  2      values('AII7123','Air India','Delhi','Banglore','13-may-24','13-may-24 6:30:00','13-may-24 8:30:00','500km','13
-may-24 02:00:00','Ontime');

1 row created.

SQL>
SQL> insert into flight_details(flight_no,flight_name,origin,destination,departure_date,departure_time,arrival_time,tota
l_distance,flight_duration,status)
  2      values('JT4567','Jet Airways','Delhi','Mumbai','13-mar-24','13-mar-24 9:00:00','13-mar-2024 11:00:00','500km','
13-mar-24 02:00:00','Ontime');

1 row created.

SQL>
SQL> insert into flight_details(flight_no,flight_name,origin,destination,departure_date,departure_time,arrival_time,tota
l_distance,flight_duration,status)
  2      values('NA7890','Nepal Airlines','Kathmandu','Delhi','23-mar-24','23-mar-24 3:00:00','23-mar-24 4:30:00','400km
','13-mar-24 01:30:00','Ontime');

1 row created.

SQL>
SQL> insert into flight_details(flight_no,flight_name,origin,destination,departure_date,departure_time,arrival_time,tota
l_distance,flight_duration,status)
  2      values('SPJ1234','SpiceJet','Banglore','Calcutta','4-apr-24','4-apr-24 2:15:00','4-apr-24 4:30:00','250km','4-a
pr-24 02:15:00','Ontime');

1 row created.
```

2. CREW DETAILS:

```
SQL> create table crew_details(
  2       crew_id varchar(5) ,
  3       crew_name varchar(10),
  4       gender varchar(10),
  5       job varchar(20) not null,
  6       hiredate date not null,
  7       sal number(10) not null,
  8       comm number(5) not null,
  9       primary key(crew_id,crew_name)
 10  );

Table created.
```

```
SQL> insert into crew_details (crew_id,crew_name,gender,job,hiredate,sal,comm) values ('s789','Saramsh','m','Pilot','11-
May-20',50000,10000);

1 row created.

SQL>
SQL> insert into crew_details (crew_id,crew_name,gender,job,hiredate,sal,comm) values ('m169','Member','m','Co-Pilot','1
0-Dec-21',30000,5000);

1 row created.

SQL>
SQL> insert into crew_details (crew_id,crew_name,gender,job,hiredate,sal,comm) values ('g123','Gaurav','m','Ticket Agent
','01-Jan-22',20000,5000);

1 row created.

SQL>
SQL> insert into crew_details (crew_id,crew_name,gender,job,hiredate,sal,comm) values ('sj45','Sanjay','m','Flight Instr
uctor','01-Apr-23',22000,3000);

1 row created.

SQL>
SQL> insert into crew_details (crew_id,crew_name,gender,job,hiredate,sal,comm) values ('gk010','Gurkirat','m','Flight Ma
rketing','20-Feb-24',20000,3000);

1 row created.
```

3. SEAT STATUS:

```
SQL> create table seat_status(
  2     id number(5) primary key,
  3     flight_no references flight_details(flight_no),
  4     business_class number(5),
  5     economy_class number(5),
  6     economy_class_price number(15) ,
  7     business_class_price number(15),
  8     total_seats number(5)
  9  );

Table created.
```

```
SQL> insert into seat_status
  2     values (1037,'AII7123',20,60,10000,4000,80);

1 row created.

SQL>
SQL> insert into seat_status
  2     values (1567,'SPJ1234',30,60,6000,2500,90);

1 row created.

SQL>
SQL> insert into seat_status
  2     values (1003,'AI7012',40,80,5000,2000,120);

1 row created.

SQL>
SQL> insert into seat_status
  2     values (1045,'JT4567',30,70,8000,4000,100);

1 row created.

SQL>
SQL>
SQL> insert into seat_status
  2     values (1722,'NA7890',40,80,20000,10000,120);
```

4. PASSENGERS DETAILS:

```
SQL> create table passenger_details(
  2  flight_no varchar(10) references flight_details(flight_no),passenger_name varchar(15) not null,
  3  id_pass number(10) primary key,
  4  email varchar(20) not null,
  5  phone number(10) not null, gender varchar(10),
  6  booking_date date
  7  );

Table created.
```

```
SQL> insert into passenger_details values('AI7012', 'Priya', 2341,'priya@123', 7845987411, 'female', '2-apr-24');

1 row created.

SQL> insert into passenger_details values('AII7123', 'Rahul', 24351,'ra123hul', 9874561115, 'male', '13-may-2024');

1 row created.

SQL> insert into passenger_details values('JT4567', 'Naina',25718, 'naiiina5', 7458964787, 'female', '13-mar-24');

1 row created.

SQL> insert into passenger_details values('NA7890', 'Sagar',2527, 'sagar45', 9878794560, 'male', '23-mar-24');

1 row created.

SQL> insert into passenger_details values('SPJ1234','Vandana', 62547,'123vanii', 7845874591, 'female', '4-apr-24');

1 row created.
```

5. DETAILS_CANCEL:

(This table is a copy of table passenger_details. All the data which will be deleted from passenger_details table will be stored in this table so that other passengers can check the availability of cancelled seats also.)

```
 1  create table details_cancel(flight_no varchar(10) references flight_details(flight_no),
 2  passenger_name varchar(15) not null,
 3  email varchar(20) not null,
 4  phone number(10) not null,
 5  gender varchar(10),
 6* booking_date date)
QL> /

able created.
```

6. SEAT_ALLOTED:

```
  1  create table seat_alloted(id number(10)primary key,
  2  flight_no references flight_details(flight_no),
  3  class varchar(20),
  4  total_seats number(20),
  5* total_price number(20))
SQL> /

Table created.

SQL>
```

```
SQL> insert into seat_alloted values(233,'AII7123','business',4,null);

1 row created.

SQL> insert into seat_alloted values(986,'JT4567','business',1,null);

1 row created.

SQL> insert into seat_alloted values(125,'JT4567','economy',6,null);

1 row created.

SQL>
```

7. AIRPORT:

```
SQL> CREATE TABLE airport (
  2      airport_code VARCHAR(3) PRIMARY KEY,
  3      name VARCHAR(100),
  4      city VARCHAR(100),
  5      country VARCHAR(100)
  6  )
  7  /

Table created.
```

```
SQL> INSERT INTO airport VALUES('DEL', 'Indira Gandhi International Airport', 'New Delhi', 'India');

1 row created.

SQL> INSERT INTO airport VALUES('BOM', 'Chhatrapati Shivaji Maharaj International Airport', 'Mumbai', 'India');

1 row created.

SQL> INSERT INTO airport VALUES('MAA', 'Chennai International Airport', 'Chennai', 'India');

1 row created.

SQL> INSERT INTO airport VALUES('BLR', 'Kempegowda International Airport', 'Bengaluru', 'India');

1 row created.

SQL> INSERT INTO airport VALUES('HYD', 'Rajiv Gandhi International Airport', 'Hyderabad', 'India');

1 row created.
```

8. VIP_PASSENGERS:

```
  7* );
SQL> CREATE TABLE vip_passengers (
  2      vip_id NUMBER PRIMARY KEY,
  3      flightno varchar(20) references flight_details(flight_no),
  4      id_passenger number(10) references passenger_details(id_pass),
  5      status VARCHAR(20),
  6      privileges VARCHAR(255)
  7  );

Table created.

SQL>
```

```
SQL> INSERT INTO vip_passengers VALUES(2, 'AII7123', 24351, 'Platinum', 'Priority check-in, Upgrade vouchers');

1 row created.

SQL> INSERT INTO vip_passengers VALUES(3, 'JT4567', 25718, 'Silver', 'Extra baggage allowance');

1 row created.

SQL> INSERT INTO vip_passengers VALUES(4, 'NA7890', 2527, 'Gold', 'Complimentary meals, Dedicated hotline');

1 row created.

SQL> INSERT INTO vip_passengers VALUES(5, 'SPJ1234', 62547, 'Platinum', 'Chauffeur service, Premium lounge access');

1 row created.
```

9. LUGGAGE:

```
SQL> CREATE TABLE luggage (
  2      luggage_id NUMBER PRIMARY KEY,
  3      idpassenger number(10) references passenger_details(id_pass),
  4      weight_kg NUMBER,
  5      description VARCHAR(255)
  6  );

Table created.
```

```
SQL> INSERT INTO luggage VALUES(2, 24351, 10, 'Small backpack with laptop and documents');

1 row created.

SQL> INSERT INTO luggage VALUES(3, 25718, 20, 'Medium-sized bag with sports equipment');

1 row created.

SQL> INSERT INTO luggage VALUES(4, 2527, 18, 'Rolling suitcase with toiletries and clothes');

1 row created.

SQL> INSERT INTO luggage VALUES(5, 62547, 12, 'Gym bag with workout gear and shoes');

1 row created.
```

10. PAYMENTS_METHODS:

```
SQL> CREATE TABLE payment_methods (
  2      payment_method_id NUMBER PRIMARY KEY,
  3      passenger_id NUMBER,
  4      method_type VARCHAR(50),
  5      card_number VARCHAR(16),
  6      expiration_date DATE,
  7      billing_address VARCHAR(255),
  8      FOREIGN KEY (passenger_id) REFERENCES passenger_details(id_pass)
  9  );

Table created.
```

```
SQL> INSERT INTO payment_methods VALUES(1, 24351, 'Credit Card', '1234567890123456', TO_DATE('2024-12-01', 'YYYY-MM-DD')
, '123 Main St, City, Country');

1 row created.

SQL> INSERT INTO payment_methods VALUES(2, 25718, 'Debit Card', '9876543210987654', TO_DATE('2025-06-01', 'YYYY-MM-DD'),
 '456 Elm St, City, Country');

1 row created.

SQL> INSERT INTO payment_methods VALUES(3, 2527, 'Credit Card', '5678901234567890', TO_DATE('2023-10-01', 'YYYY-MM-DD'),
 '789 Oak St, City, Country');
```

11.PASSENGER_PREFERENCES:

```
SQL> CREATE TABLE passenger_preferences (
  2      preference_id NUMBER PRIMARY KEY,
  3      passenger_id NUMBER,
  4      seat_preference VARCHAR(20),
  5      meal_preference VARCHAR(50),
  6      special_assistance_needed VARCHAR(100),
  7      FOREIGN KEY (passenger_id) REFERENCES passenger_details(id_pass)
  8  );

Table created.
```

```
SQL> INSERT INTO passenger_preferences VALUES(1, 24351, 'Window', 'Vegetarian', 'Wheelchair assistance');

1 row created.

SQL> INSERT INTO passenger_preferences VALUES(2, 25718, 'Aisle', 'Kosher', 'None');

1 row created.

SQL> INSERT INTO passenger_preferences VALUES(3, 2527, 'Middle', 'Regular', 'Extra legroom');

1 row created.

SQL> INSERT INTO passenger preferences VALUES(4, 62547, 'Window', 'Gluten-free', 'Hearing impaired assistance');

1 row created.
```

12. PASSENGER_FEEDBACK:

```
SQL> CREATE TABLE passenger_feedback (
  2      feedback_id NUMBER PRIMARY KEY,
  3      passenger_id NUMBER,
  4      feedback varchar(100),
  5      feedback_date DATE,
  6      FOREIGN KEY (passenger_id) REFERENCES passenger_details(id_pass)
  7  );

Table created.
```

```
SQL> INSERT INTO passenger_feedback values(2,25718, 'The flight was delayed.', TO_DATE('2024-05-15', 'YYYY-MM-DD'));

1 row created.

SQL> INSERT INTO passenger_feedback values (3, 2527, 'The food was excellent.', TO_DATE('2024-06-10', 'YYYY-MM-DD'));

1 row created.

SQL> INSERT INTO passenger_feedback values(4, 62547, 'The staff was very helpful.', TO_DATE('2024-07-20', 'YYYY-MM-DD'))
;

1 row created.
```

1. **Flight_details:**

```
create table flight_details(
    flight_no varchar(10) primary key,
    flight_name varchar(15) not null,
    origin varchar(15) not null,
    destination varchar(15) not null,
    departure_date date,
    departure_time timestamp,
    arrival_time timestamp,
    total_distance varchar(8) not null,
    flight_duration timestamp,
    status varchar(10) not null
);

insert into flight_details values('A17012', 'Air Indigo', 'Amritsar', 'Delhi', '2-apr-24', '2-apr-24
09:05:00', '2-apr-24 10:50:00', '200km', '2-apr-24 01:45:00', 'Ontime');
insert into flight_details values('AII7123', 'Air India', 'Delhi', 'Banglore', '13-may-24', '13-may-24
6:30:00', '13-may-24 8:30:00', '500', '13 -may-24 02:00:00', 'Ontime');
insert into flight_details values('JT4567', 'Jet Airways', 'Delhi', 'Mumbai', '13-mar-24', '13-mar-24
9:00:00', '13-mar-2024 11:00:00', '500km','13-mar-24 02:00:00', 'Ontime');
insert into flight_details values('NA7890', 'Nepal Airlines', 'Kathmandu', 'Delhi', '23-mar-24', '23-
mar-24 3:00:00', '23-mar-24 4:30:00','400km',' 13-mar-24 01:30:00', 'Ontime');
insert into flight_details values('SPJ1234', 'SpiceJet', 'Banglore', 'Calcutta', '4-apr-24', '4-apr-24
2:15:00', '4-apr-24 4:30:00', '250km', '4-apr-24 02:15:00', 'Ontime');
```

2. **Crew_details:**

```
create table crew_details(
    crew_id varchar(5) ,
    crew_name varchar(10) not null,
    gender varchar(10),
    job varchar(20) not null,
    hiredate date not null,
    sal number(10) not null,
    comm number(5) not null,
    primary key(crew_id,crew_name)
);


insert into crew_details (crew_id, crew_name, gender, job, hiredate, sal, comm) values ('s789',
```

'Saramsh', 'm', 'Pilot', '11- May-20', 50000,10000);
insert into crew_details (crew_id,crew_name, gender, job, hiredate, sal, comm) values ('m169', 'Member', 'm', 'Co-Pilot', '10-Dec-21',30000, 5000);
insert into crew_details (crew_id, crew_name, gender, job, hiredate, sal, comm) values ('g123', 'Gaurav', 'm', 'Ticket Agent', '01-Jan-22', 20000, 5000);
insert into crew_details (crew_id, crew_name, gender, job, hiredate, sal, comm) values ('sj45', 'Sanjay', 'm', 'Flight Instructor', '01-Apr-23', 22000, 3000);
insert into crew_details (crew_id, crew_name, gender, job, hiredate, sal, comm) values ('gk010', 'Gurkirat','m', 'Flight Marketing', '20-Feb-24', 20000, 3000);

3. Seat_status:

```
create table seat_status(
   id number(5) primary key,
   flight_no references flight_details(flight_no),
   business_class number(5),
   economy_class number(5),
   economy_class_price number(15),
   business_class_price number(15),
   total_seats number(5)
);

insert into seat_status values(1037, 'AII7123', 20, 60, 10000, 4000, 80);
insert into seat_status values(1567, 'SPJ1234', 30, 60, 6000, 2500, 90);
insert into seat_status values(1003, 'A17012', 40, 80, 5000, 2000, 120);
insert into seat_status values(1045, 'JT4567', 30, 70, 8000, 4000, 100);
insert into seat_status values(1722, 'NA7890', 40, 80, 20000, 10000, 120);
```

4. Passenger_details:

```
create table passenger_details(
flight_no varchar(10) references flight_details(flight_no),passenger_name varchar(15) not null,
id_pass number(10) primary key,
email varchar(20) not null,
phone number(10) not null, gender varchar(10),
booking_date date
);

insert into passenger_details values('AI7012', 'Priya',2341 'priya@123', 7845987411, 'female', '2-apr-24');
insert into passenger_details values('AII7123', 'Rahul', 24351,'ra123hul', 9874561115, 'male', '13-
```

```
may-2024');
insert into passenger_details values('JT4567', 'Naina', 25718,'naiiina5', 7458964787, 'female', '13-
mar-24');
insert into passenger_details values('NA7890', 'Sagar', 2527,'sagar45', 9878794560, 'male', '23-
mar-24');
insert into passenger_details values('SPJ1234', 'Vandana',62547, '123vanii', 7845874591, 'female',
'4-apr-24');
```

5. Details_cancel:

```
create table details_cancel(
    flight_no varchar(10) references flight_details(flight_no),
    passenger_name varchar(15) not null,
    email varchar(20) not null,
    phone number(10) not null,
    gender varchar(10),
    booking_date date
);
```

6. Seat_alloted:

```
create table seat_alloted(
    id number(10) primary key,
    flight_no references flight_details(flight_no),
    class varchar(20),
    total_seats number(20),
    total_price number(20)
);
```

7. Airport:

```
CREATE TABLE airport (
    airport_code VARCHAR(3) PRIMARY KEY,
    name VARCHAR(100),
    city VARCHAR(100),
    country VARCHAR(100)
)
/
INSERT INTO airport VALUES('DEL', 'Indira Gandhi International Airport', 'New Delhi',
'India');
```

```
INSERT INTO airport VALUES('BOM', 'Chhatrapati Shivaji Maharaj International Airport',
'Mumbai', 'India');
INSERT INTO airport VALUES('MAA', 'Chennai International Airport', 'Chennai', 'India');
INSERT INTO airport VALUES('BLR', 'Kempegowda International Airport', 'Bengaluru',
'India');
INSERT INTO airport VALUES('HYD', 'Rajiv Gandhi International Airport', 'Hyderabad',
'India');
/
```

8. Vip_Passengers:

```
CREATE TABLE vip_passengers (
    vip_id NUMBER PRIMARY KEY,
    flightno varchar(20) references flight_details(flight_no),
    id_passenger number(10) references passenger_details(id_pass),
    status VARCHAR(20),
    privileges VARCHAR(255)
)
/
INSERT INTO vip_passengers VALUES(1, 'A17012', 2341, 'Gold', 'Priority boarding, Lounge
access');
INSERT INTO vip_passengers VALUES(2, 'AII7123', 24351, 'Platinum', 'Priority check-in,
Upgrade vouchers');
INSERT INTO vip_passengers VALUES(3, 'JT4567', 25718, 'Silver', 'Extra baggage allowance');
INSERT INTO vip_passengers VALUES(4, 'NA7890', 2527, 'Gold', 'Complimentary meals,
Dedicated hotline');
INSERT INTO vip_passengers VALUES(5, 'SPJ1234', 62547, 'Platinum', 'Chauffeur service,
Premium lounge access');
```

9. Luggage:

```
CREATE TABLE luggage (
    luggage_id NUMBER PRIMARY KEY,
    idpassenger number(10) references passenger_details(id_pass),
    weight_kg NUMBER,
    description VARCHAR(255)
);
/

SQL> INSERT INTO luggage VALUES(2, 24351, 10, 'Small backpack with laptop and
```

documents'); 1 row created.
SQL> INSERT INTO Luggage VALUES(3, 25718, 20, 'Medium-sized bag with sports equipment'); 1 row created.
SQL> INSERT INTO Luggage VALUES(4, 2527, 18, 'Rolling suitcase with toiletries and clothes'); 1 row created.
SQL> INSERT INTO luggage VALUES(5, 62547, 12, 'Gym bag with workout gear and shoes'); 1 row created.

10. Payment_method:

```
CREATE TABLE payment_methods (
    payment_method_id NUMBER PRIMARY KEY,
    passenger_id NUMBER,
    method_type VARCHAR(50),
    card_number VARCHAR(16),
    expiration_date DATE,
    billing_address VARCHAR(255),
    FOREIGN KEY (passenger_id) REFERENCES passenger_details(id_pass)
)
/
INSERT INTO payment_methods VALUES(1, 24351, 'Credit Card', '1234567890123456',
TO_DATE('2024-12-01', 'YYYY-MM-DD'), '123 Main St, City, Country');
INSERT INTO payment_methods VALUES(2, 25718, 'Debit Card', '9876543210987654',
TO_DATE('2025-06-01', 'YYYY-MM-DD'), '456 Elm St, City, Country');
INSERT INTO payment_methods VALUES(3, 2527, 'Credit Card', '5678901234567890',
TO_DATE('2023-10-01', 'YYYY-MM-DD'), '789 Oak St, City, Country');
INSERT INTO payment_methods VALUES(4, 62547, 'PayPal', 'paypal@example.com',
TO_DATE('2026-03-01', 'YYYY-MM-DD'), '321 Pine St, City, Country');
```

11. Passengers_preferences:

```
CREATE TABLE passenger_preferences (
    preference_id NUMBER PRIMARY KEY,
    passenger_id NUMBER,
    seat_preference VARCHAR(20),
    meal_preference VARCHAR(50),
    special_assistance_needed VARCHAR(100),
    FOREIGN KEY (passenger_id) REFERENCES passenger_details(id_pass)
)
```

SQL> INSERT INTO passenger_prêtêrêñĉês VALUES(1, 24351, 'Window', 'Vêgêtarian', 'Whêêlêhàir assistancê');
1 row created.
SQL> INSERT INTO passenger_preferenees VALUES (2; 25718; 'Aiste'; 'Kosher'; 'None');
1 row created:
SQL> INSERT INTO BREEerger_pre‡8FBRES VALUES(3, 3527, Middle', 'Regular', 'Extra Lagreem');
SQL> INSERT INTO passenger preferences VALUES (4, 62547, 'Window', 'Gluten-free', 'Hearing impaired assistance'); 1 row created.
/

12. Passengers_feedback:

```
CREATE TABLE passenger_feedback (
   feedback_id NUMBER PRIMARY KEY,
   passenger_id NUMBER,
   feedback varchar(100),
   feedback_date DATE,
   FOREIGN KEY (passenger_id) REFERENCES passenger_details(id_pass)
)
/
```

SQL> INSERT INTO passenger_feedback values(2,25718, 'The flight was delayed.', TO_DATE('2024-05-15', 'YYYY-MM-DD')); 1 row created.
SQL> INSERT INTO passenger_feedback values (3, 2527, 'The food was excellent.', TO_DATE('2024-06-10', 'YYYY-MM-DD')); 1 row created.
SQL> INSERT INTO passenger_feedback values (4, 62547, 'The staff was very helpful.', TO_DATE('2024-07-20', 'YYYY-MM-DD'))
1 row created.

PROCEDURES:

1. To increase salary of crew details on the basis of commission.

```
create or replace procedure sal_incr(id_no in crew_details.crew_id%TYPE, par in number)
is
c_sal crew_details.sal%TYPE;
begin
select (sal+(sal*(par/100))) into c_sal from crew_details where crew_id=id_no;
COMMIT;
dbms_output.put_line('Crew_id :: '||id_no||' Salary :: '||c_sal);
EXCEPTION
WHEN NO_DATA_FOUND THEN
dbms_output.put_line('no crew found');
when others then
dbms_output.put_line('error occured'||SQLERRM);
ROLLBACK;
end sal_incr;
/

begin
  sal_incr('m169',1000);
end;
/
```

```
SQL> create or replace procedure sal_incr(id_no in crew_details.crew_id%TYPE,par in number)
  2     is
  3     c_sal crew_details.sal%TYPE;
  4     begin
  5
  6     select (sal+(sal*(par/100))) into c_sal from crew_details where crew_id=id_no;
  7     COMMIT;
  8     dbms_output.put_line('Crew_id :: '||id_no||' Salary :: '||c_sal);
  9     EXCEPTION
 10     WHEN NO_DATA_FOUND THEN
 11     dbms_output.put_line('no crew found');
 12     when others then
 13     dbms_output.put_line('error occured'||SQLERRM);
 14     rollback;
 15     end sal_incr;
 16
 17
 18
 19  /

Procedure created.

SQL> begin
  2  sal_incr('m169',1000);
  3  end;
  4  /

PL/SQL procedure successfully completed.
```

```
SQL> set serveroutput on;
SQL> /
Crew_id :: m169 Salary :: 330000

PL/SQL procedure successfully completed.

SQL>
```

2. To Retrieve passenger luggage details.

```
CREATE OR REPLACE PROCEDURE GetPassengerLuggage(
   passenger_id_param IN NUMBER
)
IS
   v_passenger_name passenger_details.passenger_name%TYPE;
   v_luggage_id luggage.luggage_id%TYPE;
   v_weight_kg luggage.weight_kg%TYPE;
   v_description luggage.description%TYPE;
BEGIN
   -- Retrieve passenger name
   SELECT passenger_name INTO v_passenger_name
   FROM passenger_details
   WHERE id_pass = passenger_id_param;
   -- Print passenger name
   DBMS_OUTPUT.PUT_LINE('Passenger: ' || v_passenger_name);
   -- Retrieve luggage information for the passenger
   FOR luggage_rec IN (SELECT luggage_id, weight_kg, description FROM luggage WHERE
 idpassenger = passenger_id_param)
   LOOP
      v_luggage_id := luggage_rec.luggage_id;
      v_weight_kg := luggage_rec.weight_kg;
      v_description := luggage_rec.description;
      -- Print luggage information
      DBMS_OUTPUT.PUT_LINE('Luggage ID: ' || v_luggage_id || ', Weight: ' || v_weight_kg || '
 kg, Description: ' || v_description);
   END LOOP;
END;
/
```

```
 1  CREATE OR REPLACE PROCEDURE GetPassengerLuggage(
 2      passenger_id_param IN NUMBER
 3  )
 4  IS
 5      v_passenger_name passenger_details.passenger_name%TYPE;
 6      v_luggage_id luggage.luggage_id%TYPE;
 7      v_weight_kg luggage.weight_kg%TYPE;
 8      v_description luggage.description%TYPE;
 9  BEGIN
10      -- Retrieve passenger name
11      SELECT passenger_name INTO v_passenger_name
12      FROM passenger_details
13      WHERE id_pass = passenger_id_param;
14      -- Print passenger name
15      DBMS_OUTPUT.PUT_LINE('Passenger: ' || v_passenger_name);
16      -- Retrieve luggage information for the passenger
17      FOR luggage_rec IN (SELECT luggage_id, weight_kg, description FROM luggage WHERE idpassenger = passenger_id_par
am)
18      LOOP
19          v_luggage_id := luggage_rec.luggage_id;
20          v_weight_kg := luggage_rec.weight_kg;
21          v_description := luggage_rec.description;
22          -- Print luggage information
23          DBMS_OUTPUT.PUT_LINE('Luggage ID: ' || v_luggage_id || ', Weight: ' || v_weight_kg || ' kg, Description: '
|| v_description);
24      END LOOP;
25* END;
SQL> /

Procedure created.
```

```
 1  DECLARE
 2      passenger_id_param NUMBER := 2527;
 3  BEGIN
 4      GetPassengerLuggage(passenger_id_param);
 5* END;
SQL> /

PL/SQL procedure successfully completed.

SQL> set serveroutput on;
SQL> /
Passenger: Sagar
Luggage ID: 4, Weight: 18 kg, Description: Rolling suitcase with toiletries and
clothes

PL/SQL procedure successfully completed.
```

FUNCTIONS:

1. To display total number of passengers.

```
create or replace function display_total_passengers
RETURN number IS
total_passengers number(5):=0;
BEGIN
   select count(*) into total_passengers from passenger_details;
   return total_passengers;
end;
/

declare
total number(5);
begin
   total:=display_total_passengers;
   dbms_output.put_line('total passengers are'||total);
end;
```

```
SQL> create or replace function display_total_passengers
  2      RETURN number IS
  3      total_passengers number(5):=0;
  4      BEGIN
  5     select count(*) into total_passengers from passenger_details;
  6     return total_passengers;
  7     end;
  8  /

Function created.

SQL> declare
  2    total number(5);
  3      begin
  4      total:=display_total_passengers;
  5      dbms_output.put_line('total passengers are '||total);
  6     end;
  7  /
total passengers are 5

PL/SQL procedure successfully completed.
```

2. To calculate price of seats selected by the passenger.

```
CREATE OR REPLACE FUNCTION calculate_price(
   f_flight_no IN VARCHAR2,
   f_class IN VARCHAR2,
   f_total_seats IN NUMBER,
   f_total_price OUT NUMBER
) RETURN NUMBER IS
   v_business_seats NUMBER;
   v_economy_seats NUMBER;
   v_business_price NUMBER;
   v_economy_price NUMBER;
BEGIN
   -- Fetch seat availability and prices for the given flight
   SELECT business_class, economy_class, business_class_price, economy_class_price
   INTO v_business_seats, v_economy_seats, v_business_price, v_economy_price
   FROM seat_status
   WHERE flight_no = f_flight_no;
   -- Check seat availability and calculate total price
   IF f_class = 'business' THEN
      IF f_total_seats <= v_business_seats THEN
         f_total_price := f_total_seats * v_business_price;
         RETURN f_total_price; -- Success
      ELSE
         RAISE_APPLICATION_ERROR(-20001, 'Business seats not available');
      END IF;
   ELSE
      IF f_total_seats <= v_economy_seats THEN
         f_total_price := f_total_seats * v_economy_price;
         RETURN f_total_price; -- Success
      ELSE
         RAISE_APPLICATION_ERROR(-20002, 'Economy seats not available');
      END IF;
   END IF;
EXCEPTION
   WHEN NO_DATA_FOUND THEN
      RAISE_APPLICATION_ERROR(-20003, 'Flight not found');
   WHEN OTHERS THEN
      RAISE_APPLICATION_ERROR(-20004, 'An error occurred: ' || SQLERRM);
END;
/
```

```
DECLARE
   flightno VARCHAR(20) := '&flightno';
   totalseats NUMBER := &totalseats;
   totalprice NUMBER := 0;
BEGIN
   totalprice := calculate_price(flightno, 'business', totalseats, totalprice);
   DBMS_OUTPUT.PUT_LINE(totalprice);
END;
/
```

```
 1  CREATE OR REPLACE FUNCTION calculate_price(
 2      f_flight_no IN VARCHAR2,
 3      f_class IN VARCHAR2,
 4      f_total_seats IN NUMBER,
 5      f_total_price OUT NUMBER
 6  ) RETURN NUMBER IS
 7      v_business_seats NUMBER;
 8      v_economy_seats NUMBER;
 9      v_business_price NUMBER;
10      v_economy_price NUMBER;
11  BEGIN
12      -- Fetch seat availability and prices for the given flight
13      SELECT business_class, economy_class, business_class_price, economy_class_price
14      INTO v_business_seats, v_economy_seats, v_business_price, v_economy_price
15      FROM seat_status
16      WHERE flight_no = f_flight_no;
17      -- Check seat availability and calculate total price
18      IF f_class = 'business' THEN
19          IF f_total_seats <= v_business_seats THEN
20              f_total_price := f_total_seats * v_business_price;
21              RETURN f_total_price; -- Success
22          ELSE
23              RAISE_APPLICATION_ERROR(-20001, 'Business seats not available');
24          END IF;
25      ELSE
26          IF f_total_seats <= v_economy_seats THEN
27              f_total_price := f_total_seats * v_economy_price;
28              RETURN f_total_price; -- Success
29          ELSE
```

```
29          ELSE
30              RAISE_APPLICATION_ERROR(-20002, 'Economy seats not available');
31          END IF;
32      END IF;
33  EXCEPTION
34      WHEN NO_DATA_FOUND THEN
35          RAISE_APPLICATION_ERROR(-20003, 'Flight not found');
36      WHEN OTHERS THEN
37          RAISE_APPLICATION_ERROR(-20004, 'An error occurred: ' || SQLERRM);
38* END;
39  /

Function created.

SQL> edit
Wrote file afiedt.buf

  1  DECLARE
  2      flightno VARCHAR(20) := '&flightno';
  3      totalseats NUMBER := &totalseats;
  4      totalprice NUMBER := 0;
  5  BEGIN
  6      totalprice := calculate_price(flightno, 'business', totalseats, totalprice);
  7      DBMS_OUTPUT.PUT_LINE(totalprice);
  8* END;
SQL> /
Enter value for flightno: AII7123
old   2:     flightno VARCHAR(20) := '&flightno';
new   2:     flightno VARCHAR(20) := 'AII7123';
Enter value for totalseats: 4
```

```
  8* END;
SQL> /
Enter value for flightno: AII7123
old   2:     flightno VARCHAR(20) := '&flightno';
new   2:     flightno VARCHAR(20) := 'AII7123';
Enter value for totalseats: 4
old   3:     totalseats NUMBER := &totalseats;
new   3:     totalseats NUMBER := 4;
16000

PL/SQL procedure successfully completed.

SQL> edit
Wrote file afiedt.buf
```

3. To segregate vip privileges on basis of gold, silver and platinum.

```
CREATE OR REPLACE FUNCTION SegregatePrivileges(
   status_param IN VARCHAR2
) RETURN VARCHAR2
IS
   v_privileges VARCHAR2(1000);
BEGIN
   -- Determine privileges based on status
   CASE status_param
      WHEN 'Gold' THEN
         v_privileges := 'Priority boarding, Lounge access';
      WHEN 'Silver' THEN
         v_privileges := 'Extra baggage allowance';
      WHEN 'Platinum' THEN
         v_privileges := 'Upgrade vouchers, Chauffeur service, Premium lounge access';
      ELSE
         v_privileges := 'No privileges defined for this status';
   END CASE;

   RETURN v_privileges;
END;
    /
DECLARE
   gold_privileges VARCHAR2(1000);
   silver_privileges VARCHAR2(1000);
   platinum_privileges VARCHAR2(1000);
BEGIN
   gold_privileges := SegregatePrivileges('Gold');
```

```
    silver_privileges := SegregatePrivileges('Silver');
    platinum_privileges := SegregatePrivileges('Platinum');

    DBMS_OUTPUT.PUT_LINE('Gold Privileges: ' || gold_privileges);
    DBMS_OUTPUT.PUT_LINE('Silver Privileges: ' || silver_privileges);
    DBMS_OUTPUT.PUT_LINE('Platinum Privileges: ' || platinum_privileges);
END;
/
```

```
SQL> CREATE OR REPLACE FUNCTION SegregatePrivileges(
  2      status_param IN VARCHAR2
  3  ) RETURN VARCHAR2
  4  IS
  5      v_privileges VARCHAR2(1000);
  6  BEGIN
  7      -- Determine privileges based on status
  8      CASE status_param
  9          WHEN 'Gold' THEN
 10              v_privileges := 'Priority boarding, Lounge access';
 11          WHEN 'Silver' THEN
 12              v_privileges := 'Extra baggage allowance';
 13          WHEN 'Platinum' THEN
 14              v_privileges := 'Upgrade vouchers, Chauffeur service, Premium lounge access';
 15          ELSE
 16              v_privileges := 'No privileges defined for this status';
 17      END CASE;
 18
 19      RETURN v_privileges;
 20  END;
 21  /

Function created.
```

```
SQL> DECLARE
  2      gold_privileges VARCHAR2(1000);
  3      silver_privileges VARCHAR2(1000);
  4      platinum_privileges VARCHAR2(1000);
  5  BEGIN
  6      gold_privileges := SegregatePrivileges('Gold');
  7      silver_privileges := SegregatePrivileges('Silver');
  8      platinum_privileges := SegregatePrivileges('Platinum');
  9
 10      DBMS_OUTPUT.PUT_LINE('Gold Privileges: ' || gold_privileges);
 11      DBMS_OUTPUT.PUT_LINE('Silver Privileges: ' || silver_privileges);
 12      DBMS_OUTPUT.PUT_LINE('Platinum Privileges: ' || platinum_privileges);
 13  END;
 14  /
Gold Privileges: Priority boarding, Lounge access
Silver Privileges: Extra baggage allowance
Platinum Privileges: Upgrade vouchers, Chauffeur service, Premium lounge access

PL/SQL procedure successfully completed.
```

4. To calculate total number of years crew member has worked.

```
CREATE OR REPLACE FUNCTION CalculateYearsWorked(
   hiredate_crew IN DATE
) RETURN NUMBER
IS
   years_worked NUMBER;
BEGIN
   -- Calculate the number of years worked from the hire date to the current date
   SELECT TRUNC(MONTHS_BETWEEN(SYSDATE, hiredate_crew) / 12) INTO
years_worked FROM DUAL;
   RETURN years_worked;
END;
/

DECLARE
   years_worked NUMBER;
BEGIN
   years_worked := CalculateYearsWorked(TO_DATE('2020-05-11', 'YYYY-MM-DD')); --
Replace with the hire date of the crew member
   DBMS_OUTPUT.PUT_LINE('Years Worked: ' || years_worked);
END;
```

```
 1  CREATE OR REPLACE FUNCTION CalculateYearsWorked(
 2      hiredate_crew IN DATE
 3  ) RETURN NUMBER
 4  IS
 5      years_worked NUMBER;
 6  BEGIN
 7      -- Calculate the number of years worked from the hire date to the current date
 8      SELECT TRUNC(MONTHS_BETWEEN(SYSDATE, hiredate_crew) / 12) INTO years_worked FROM DUAL;
 9      RETURN years_worked;
10* END;
11  /

Function created.
```

```
 1  DECLARE
 2      years_worked NUMBER;
 3  BEGIN
 4      years_worked := CalculateYearsWorked(TO_DATE('2020-05-11', 'YYYY-MM-DD')); -- Replace with the hire date of the crew member
 5      DBMS_OUTPUT.PUT_LINE('Years Worked: ' || years_worked);
 6* END;
SQL>
```

```
/
```

CURSORS:

1. To fetch flight details from table flight_details.

```
declare
cursor fetch1(initial varchar, final varchar, day date)
is
select * from flight_details where origin=initial and destination=final and departure_date=day;
begin
dbms_output.put_line('flightno'||'  '||'flightname'||'  '||'origin'||'  '||'destination'||'
'||'departuredate'||'          '||'departuretime'||'  '||'  '||'arrivaltime'||'  '||'totaldistance'||'
'||'flightduration'||'  '||'status');
for x in fetch1('&origin','&destination','&departure_date')
loop
dbms_output.put_line(x.flight_no||'  '||x.flight_name||'  '||x.origin||'  '||x.destination||'
'||x.departure_date||'  '||x.departure_time||'  '||x.arrival_time||'  '||x.total_distance||'  '||'
'||x.flight_duration||'  '||x.status);
end loop;
end;
```

```
SQL> declare
  2  cursor fetch1(initial varchar,final varchar,day date)
  3  is
  4  select * from flight_details where origin=initial and destination=final and departure_date=day;
  5  begin
  6
  7  dbms_output.put_line('flightno'||'     '||
  8      'flightname'||'   '||'origin'||'    '||'destination'||'   '||
  9      'departuredate'||'      '||'departuretime'||'       '||'arrivaltime'||'   '
 10      ||'totaldistance'||'   '||'flightduration'||'      '||'status');
 11
 12  for x in fetch1('&origin','&destination','&departure_date')
 13  loop
 14  dbms_output.put_line(x.flight_no||'    '||'   '||
 15      x.flight_name||'   '||x.origin||'  '||x.destination||'         '||
 16      x.departure_date||'             '||x.departure_time||'             '||x.arrival_time||'
 17      ||x.total_distance||'       '||x.flight_duration||'    '||x.status);
 18  end loop;
 19  end;
 20  /
Enter value for origin: Delhi
Enter value for destination: Banglore
Enter value for departure_date: 13-may-24
old  12: for x in fetch1('&origin','&destination','&departure_date')
new  12: for x in fetch1('Delhi','Banglore','13-may-24')

PL/SQL procedure successfully completed.
```

```
/
```

2. To fetch passenger details from table passenger_details.

```
declare
cursor fetch2(flightno varchar, day date)
is
select * from passenger_details where flight_no=flightno and booking_date=day;
begin
   dbms_output.put_line('flightno'||'   '||'passenger_name'||'   '||'email'||'   '||'phone'||'   '||'gender'||'
'||'booking_date'||'   ');
   for x in fetch2('&flight_no','&booking_date')
   loop
   dbms_output.put_line(x.flight_no||'   '||x.passenger_name||'   '||x.email||'   '||x.phone||'
'||x.gender||'   '||x.booking_date||'   ');
   end loop;
end;
/
```

```
SQL> declare
  2  cursor fetch2(flightno varchar,day date)
  3  is
  4  select * from passenger_details where flight_no=flightno and booking_date=day;
  5  begin
  6
  7  dbms_output.put_line('flightno'||'     '||
  8     'passenger_name'||'   '||'email'||'    '||'phone'||'   '||
  9     'gender'||'      '||'booking_date'||'       ');
 10
 11  for x in fetch2('&flight_no','&booking_date')
 12  loop
 13  dbms_output.put_line(x.flight_no||'    '||'   '||
 14     x.passenger_name||'   '||x.email||'  '||x.phone||'         '||
 15     x.gender||'             '||x.booking_date||'               ');
 16  end loop;
 17  end;
 18  /
Enter value for flight_no: AI7012
Enter value for booking_date: 2-apr-24
old  11: for x in fetch2('&flight_no','&booking_date')
new  11: for x in fetch2('AI7012','2-apr-24')
flightno     passenger_name   email    phone   gender      booking_date
AI7012       Priya   priya@123  7485784589         female               02-APR-24


PL/SQL procedure successfully completed.
```

3. To calculate total number of vegetarian passengers on flight.

```
DECLARE
v_veg_count NUMBER := 0; -- Variable to store the count of passengers who prefer vegetarian
 meals
CURSOR c_passenger_preferences IS
    SELECT * FROM passenger_preferences WHERE meal_preference = 'Vegetarian';
BEGIN
-- Loop through the cursor and count the number of passengers who prefer vegetarian meals
FOR pref_rec IN c_passenger_preferences
LOOP
    v_veg_count := v_veg_count + 1;
END LOOP;

-- Display the count of passengers who prefer vegetarian meals
DBMS_OUTPUT.PUT_LINE('Number of passengers who prefer vegetarian meals: ' ||
 v_veg_count);
END;
/
```

```
SQL> DECLARE
  2      v_veg_count NUMBER := 0; -- Variable to store the count of passengers who prefer vegetarian meals
  3      CURSOR c_passenger_preferences IS
  4          SELECT * FROM passenger_preferences WHERE meal_preference = 'Vegetarian';
  5  BEGIN
  6      -- Loop through the cursor and count the number of passengers who prefer vegetarian meals
  7      FOR pref_rec IN c_passenger_preferences
  8      LOOP
  9          v_veg_count := v_veg_count + 1;
 10      END LOOP;
 11
 12      -- Display the count of passengers who prefer vegetarian meals
 13      DBMS_OUTPUT.PUT_LINE('Number of passengers who prefer vegetarian meals: ' || v_veg_count);
 14  END;
 15  /
Number of passengers who prefer vegetarian meals: 1

PL/SQL procedure successfully completed.
```

TRIGGERS:

1. To insert details of passengers who cancel their bookings into table details_cancel.

```
create OR replace TRIGGER trig_details
AFTER INSERT OR UPDATE OR DELETE ON passenger_details
for each row
begin
insert into details_cancel(passenger_name, email, phone, gender, booking_date) values
(:OLD.passenger_name, :OLD.email, :OLD.phone, :OLD.gender, :OLD.booking_date);
end;
/
```

```
SQL> EDIT
Wrote file afiedt.buf

  1  create OR replace TRIGGER trig_details
  2  AFTER INSERT OR UPDATE OR DELETE ON  passenger_details
  3  for each row
  4  begin
  5  insert into details_cancel(passenger_name,email,phone,gender,booking_date)  values(:OLD.passenger_name,:OLD.email,:OLD.phone,:OLD.gender,:OLD.booking_d
ate);
  6* end;
SQL> /

Trigger created.

SQL>
```

```
SQL> select * from passenger_details
  2  ;

FLIGHT_NO  PASSENGER_NAME  EMAIL                    PHONE GENDER     BOOKING_D
---------- --------------- -------------------- ---------- ---------- ---------
AI7012     Priya           priya@123            7485784589 female     02-APR-24
AII7123    Rahul           ra123hul             9874561478 male       13-MAY-24
JT4567     Naina           naiiina              7485784584 female     13-MAR-24
NA7890     Sagar           sagar45              9874560078 male       23-MAR-24
SPJ1234    Vandana         123vaniii            8457965485 female     04-APR-24

SQL> delete from passenger_details where booking_date='02-apr-24';

1 row deleted.

SQL> select * from details_cancel;

FLIGHT_NO  PASSENGER_NAME  EMAIL                    PHONE GENDER     BOOKING_D
---------- --------------- -------------------- ---------- ---------- ---------
           Priya           priya@123            7485784589 female     02-APR-24
```

```
FLIGHT_NO  PASSENGER_NAME  EMAIL                    PHONE GENDER     BOOKING_D
---------- --------------- -------------------- ---------- ---------- ---------
AII7123    Rahul           ra123hul             9874561478 male       13-MAY-24
JT4567     Naina           naiiina              7485784584 female     13-MAR-24
NA7890     Sagar           sagar45              9874560078 male       23-MAR-24
SPJ1234    Vandana         123vaniii            8457965485 female     04-APR-24

SQL>
```

2. To raise an error if luggage weight exceeds 20 kg.

```
CREATE OR REPLACE TRIGGER CheckLuggageWeight
BEFORE INSERT OR UPDATE ON luggage
FOR EACH ROW
DECLARE
   v_max_weight CONSTANT NUMBER := 20; -- Maximum allowed weight for luggage
BEGIN
   IF :NEW.weight_kg > v_max_weight THEN
      -- Raise an error if luggage weight exceeds the limit
      RAISE_APPLICATION_ERROR(-20001, 'Luggage weight cannot exceed ' ||
v_max_weight || ' kg.');
   END IF;
END;
/
```

```
 1  CREATE OR REPLACE TRIGGER CheckLuggageWeight
 2  BEFORE INSERT OR UPDATE ON luggage
 3  FOR EACH ROW
 4  DECLARE
 5      v_max_weight CONSTANT NUMBER := 20; -- Maximum allowed weight for luggage
 6  BEGIN
 7      IF :NEW.weight_kg > v_max_weight THEN
 8          -- Raise an error if luggage weight exceeds the limit
 9          RAISE_APPLICATION_ERROR(-20001, 'Luggage weight cannot exceed ' || v_max_weight || ' kg.');
10      END IF;
11* END;
SQL> /

Trigger created.
```

```
SQL> insert into luggage values(7,25718,24,'bags and electronic components');
insert into luggage values(7,25718,24,'bags and electronic components')
            *
ERROR at line 1:
ORA-20001: Luggage weight cannot exceed 20 kg.
ORA-06512: at "SYSTEM.CHECKLUGGAGEWEIGHT", line 6
ORA-04088: error during execution of trigger 'SYSTEM.CHECKLUGGAGEWEIGHT'
```

3. To raise an error if country entered in table airport is not India.

```
CREATE OR REPLACE TRIGGER CheckAirportCountry
BEFORE INSERT OR UPDATE ON airport
FOR EACH ROW
BEGIN
   IF :NEW.country != 'India' THEN
      -- Raise an error if the country is not India
      RAISE_APPLICATION_ERROR(-20001, 'Only airports in India are allowed.');
   END IF;
END;
/
```

```
SQL> insert into airport values('PNB','Jawaharlal nehru airport','Punjab','USA');
insert into airport values('PNB','Jawaharlal nehru airport','Punjab','USA')
            *
ERROR at line 1:
ORA-20001: Only airports in India are allowed.
ORA-06512: at "SYSTEM.CHECKAIRPORTCOUNTRY", line 4
ORA-04088: error during execution of trigger 'SYSTEM.CHECKAIRPORTCOUNTRY'
```

SQL QUERIES

1.SQL Query to select details of crew whose gender is male:

```
SELECT crew_id, crew_name, gender, job, hiredate, sal, comm
FROM crew_details
WHERE gender = 'm'
/
```

```
  1  SELECT crew_id, crew_name, gender, job, hiredate, sal, comm
  2  FROM crew_details
  3* WHERE gender = 'm'
SQL> /

CREW_ CREW_NAME  GENDER     JOB                  HIREDATE         SAL       COMM
----- ---------- ---------- -------------------- --------- ---------- ----------
s789  Saramsh    m          Pilot                11-MAY-20      50000      10000
m169  Member     m          Co-Pilot             10-DEC-21      30000       5000
g123  Gaurav     m          Ticket Agent         01-JAN-22      20000       5000
sj45  Sanjay     m          Flight Instructor    01-APR-23      22000       3000
gk010 Gurkirat   m          Flight Marketing     20-FEB-24      20000       3000
```

2.SQL Query to delete record from vip_passengers where flight_no is AII7123.

```
DELETE FROM vip_passengers
WHERE flightno='AII7123'
 /
```

```
SQL> DELETE FROM vip_passengers
  2  WHERE flightno='AII7123'
  3  /

1 row deleted.
```

3.To update the price of business class seats.

```
UPDATE seat_status
SET business_class_price = 15000
WHERE id=1037
 /
```

```
SQL> UPDATE seat_status
  2  SET business_class_price = 15000
  3  WHERE id=1037;

1 row updated.
```

4.To display flight name and total seats from table flight_details and seat_status.

```
SELECT fd.flight_name, s.total_seats
FROM flight_details fd
JOIN seat_status s ON fd.flight_no = s.flight_no
WHERE s.id = 1003
 /
```

```
SQL> SELECT fd.flight_name, s.total_seats
  2  FROM flight_details fd
  3  JOIN seat_status s ON fd.flight_no = s.flight_no
  4  WHERE s.id = 1003;

FLIGHT_NAME     TOTAL_SEATS
--------------- -----------
Air Indigo              120
```

CONCLUSION

In conclusion, the Flight Management System developed in PL/SQL stands as a testament to the fusion of advanced technology and meticulous engineering in modern aviation.

Through its robust functionality, encompassing precise navigation, efficient flight planning, and seamless integration, it exemplifies the forefront of aviation innovation. By implementing a well-structured database, users can easily access and update data, facilitating efficient decision-making processes. Furthermore, the system offers a user-friendly interface and various querying capabilities, enhancing the overall user experience.

Overall, this project demonstrates the importance of utilizing database management systems in the transport industry to streamline operations and improve data accuracy.

REFERENCES

1. https://www.javatpoint.com/dbms-normalization
2. https://www.geeksforgeeks.org/normal-forms-in-dbms/
3. https://livesql.oracle.com/apex/f?p=590:1:115253254727153
4. https://www.geeksforgeeks.org/plsql-introduction
5. https://www.tutorialspoint.com/plsql/plsql_triggers.htm