Flight Database Management System

Mini Project Report -Database Lab (DSE 2241)

Department of Data Science & Computer Applications

B. Tech Data Science 4th

Semester – Batch: B4- Group:B8

Submitted By

Nayavanth Yogi	220968344
Bhuvan Battu	220968318
Adi Mukherjee	220968328
Mehul Garg	220968332
Ranveer Singh Sayal	220968340
Meghashyam Shenoy	220968352

Mentored By

Vinayak M
Assistant Professor-Senior
DSCA, MIT

Archana H
Assistant Professor-Senior
DSCA, MIT

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

Date:06/04/2024

CERTIFICATE

This is to certify that Bhuvan Battu(220968318), Adi Mukherjee (220968328), Mehul Garg(220968332), Ranveer Singh Sayal(220968340), Nayavanth Yogi(220968344), Meghashyam Shenoy(220968352) have successfully executed a mini project titled “Flight Database Management System” rightly bringing fore the competencies and skill sets they have gained during the course- Database Lab (DSE 2241), thereby resulting in the culmination of this project.

Vinayak M
Assistant Professor-Senior
DSCA, MIT

Archana H
Assistant Professor-Senior
DSCA, MIT

ABSTRACT

More people are flying than ever before. Amid this aviation scene the safety and dependability of air travel is dependent on the efficient management of flight data. The primary goal of this project is to create a flight management database that can manage seating, flight schedule, and more. The aviation industry is changing more by the day and the simultaneous development of a database management system is essential for its continued growth.

The first step of this project is observation and research into the workings of airports and flight patterns. Using the concept of relational databases, we can work together with various aspects of the industry like the flight and ground crew, customers, various airlines and the itinerary. Oracle helps to make it highly scalable and dependable, making it the perfect choice for a rapidly mutating and growing segment of travel.

A flight management database system can help end several problems such as passenger overfitting, unfit crew assignment and incorrect flight schedules. Real time data integration helps in smooth coordination and saves everyone's time, leading to passengers having a more pleasant experience. A database management system durable enough to manage complex flight operations and quick adjustment of flight plans helps in reducing delays and maximizing the resources used.

To put it briefly, a flight management database system that is dependable and well-thought-out from the ground up will contribute to the modernization and simplification of aviation operations. This project is the ideal illustration of how relational databases enhance air travel efficiency, safety, and passenger happiness. We intend to better the lives of airport staff and passengers alike with a real-time database that serves their needs.

Contents

Chapter 1

Introduction

Introducing a comprehensive flight management system tailored for airlines, our project aims to streamline the complex processes involved in managing flights, passengers, reservations, and pilots. With the aviation industry continuously expanding, there's an increasing demand for efficient systems that can handle the myriad of tasks associated with flight operations. Our project offers a robust solution that integrates various functionalities to provide airlines with a centralised platform for managing their operations seamlessly.

At the core of our project lies a sophisticated database schema designed to efficiently store and manage crucial information such as passenger details, flight schedules, reservation records, seat allocations, airport data, and pilot assignments. Leveraging relational database management systems, our schema ensures data integrity, scalability, and performance, enabling airlines to manage large volumes of data with ease. By organising data into structured tables and setting up relationships between entities, our system eases quick and accurate retrieval of information, empowering airlines to make informed decisions and optimise their operations.

Additionally, our system offers real-time updates on flight statuses, enabling airlines to promptly respond to changes and disruptions, thereby improving customer satisfaction and overall operational efficiency. With our flight management system, airlines can navigate the complexities of the aviation industry with confidence, ensuring smooth and reliable operations that meet the demands of modern air travel.

Chapter 2

Synopsis

2.1 Proposed System

The aviation industry faces significant challenges in managing data effectively, leading to operational inefficiencies and compliance issues. Manual processes worsen these problems, making it hard to ensure data accuracy and derive useful insights. To tackle these issues, there's a crucial need for a Flight Database Management System (FDMS) that can streamline operations, improve compliance, and enable predictive analytics. By using advanced technologies and adhering to industry standards, FDMS has the potential to transform aviation data management, promoting efficiency, reliability, and strategic adaptability in this fast-paced industry.

2.2 Objectives

MAIN OBJECTIVES OF THE WORK ARE-

- Uniting aviation related data into one relational database
- Allow customers to find flights easily according to their requirements.
- Empower customers to view flight status.
- Enable consolidation of flight data
- Allow pilots to view their schedule.

Chapter 3

Functional Requirements

3.1 Flight Finder Module

We hope to create a simple view for customers which allows them to make informed decisions on ticket purchases by supplying the necessary flight data.

3.1.1 Source to Destination Flight Finder

Customers must be able to find flights from source to destination.

INPUT	Destination airport, source airport
Processing	The system must check availability of flights from source to destination and fetch all corresponding flight information.
OUTPUT	Flight details of all flights available with departure and arrival time along with the cost of the flight.

3.1.2 Budget Trip Finder

INPUT	budget
Processing	Check for flight trips under the budget providing a list of travel destinations one can choose from.
OUTPUT	List of destinations with their costs for the consumers considerations.

3.2 Flight Status Module

We hope to make a simple view for customers such that they can view the status of their loved ones flights and empower them to efficiently manage their time.

3.2.1 Flight status finder

Enable customers to view the latest flight status.

INPUT	flight unique id OR destination and source
Processing	if unique id is provided the status of the plane is found at once, if destination and source is given all planes between the destination and source on the day is fetched with status
OUTPUT	flights along with their latest status

3.2.2 Flight reservation check

Allow customers to view their reservations and corresponding details.

INPUT	customer id
Processing	find customer id and find all corresponding reservations that match that customer
OUTPUT	Reservation id and other reservation details

3.3 Crew Management Module

The system should ease the management of crew members.

3.3.1 Pilot itinerary

Allow pilots to see which flights have been assigned to them and make their plans accordingly.

INPUT	unique pilot code
Processing	Find all planes assigned to the pilot for the upcoming period, allow pilot access to request for change in their delegated duties
OUTPUT	List of all flights with flight and copilot info respectively and access to request changes

Chapter 4

Detailed Design

4.1 ER Diagram

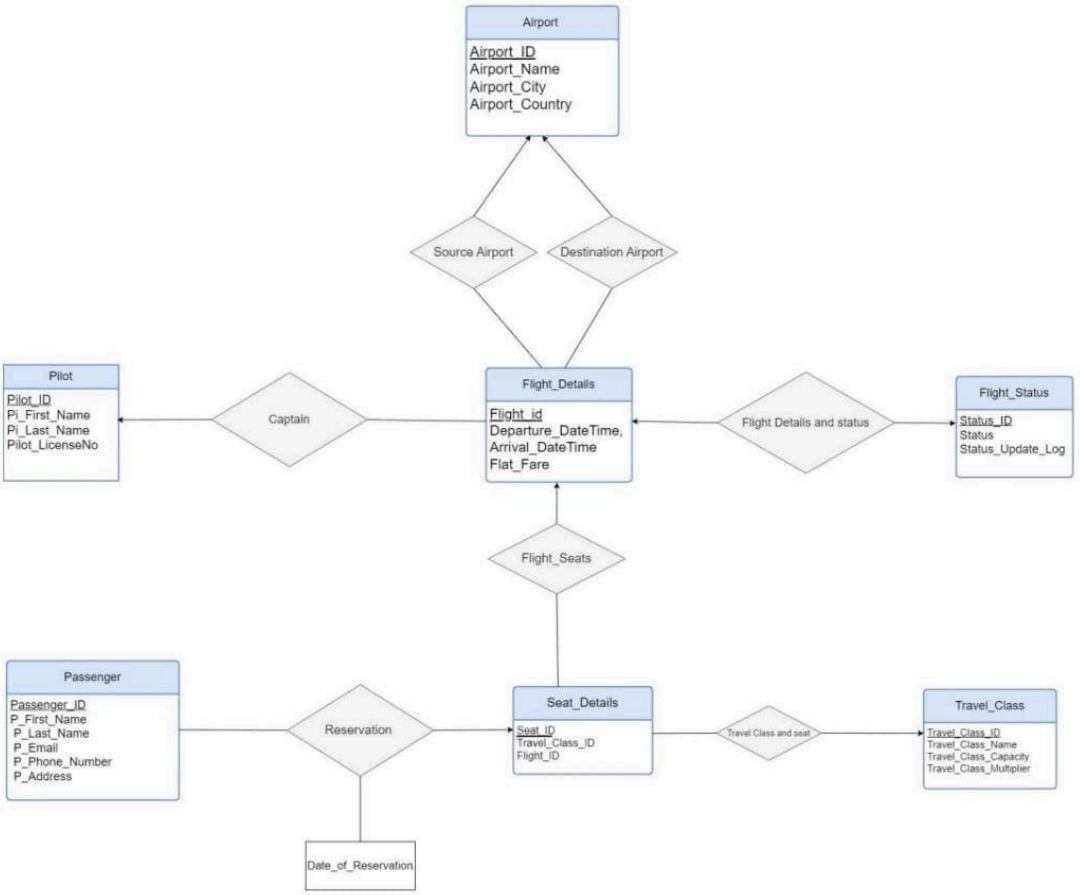

Figure 4.1 ER DIAGRAM

4.2 Schema Diagram

1. **Passenger**(Passenger_ID,P_First_Name, P_Last_Name, P_Email, P_Phone_Number, P_Address)
2. **Flight_Details(**Flight_id, *Source_Airport_ID, Destination_Airport_ID*, *Pilot_ID*, Departure_DateTime, Arrival_DateTime, Flat_Fare)
3. **Airport**(Airport_ID, Airport_Name, Airport_City, Airport_Country)
4. **Reservation**(Reservation_ID, *Passenger_ID, Seat_ID*, Date_of_Reservation)
5. **Seat_Details**(Seat_ID, *Travel_Class_ID, Flight_ID*)
6. **Travel_Class**(Travel_Class_ID, Travel_Class_Name, Travel_Class_Capacity, Travel_Class_Multiplier)
7. **Flight_Status**(*Flight_ID*, Status, Status_Update_Log)
8. **Pilot**(Pilot_ID, Pi_First_Name, Pi_Last_Name, Pilot_LicenseNo)

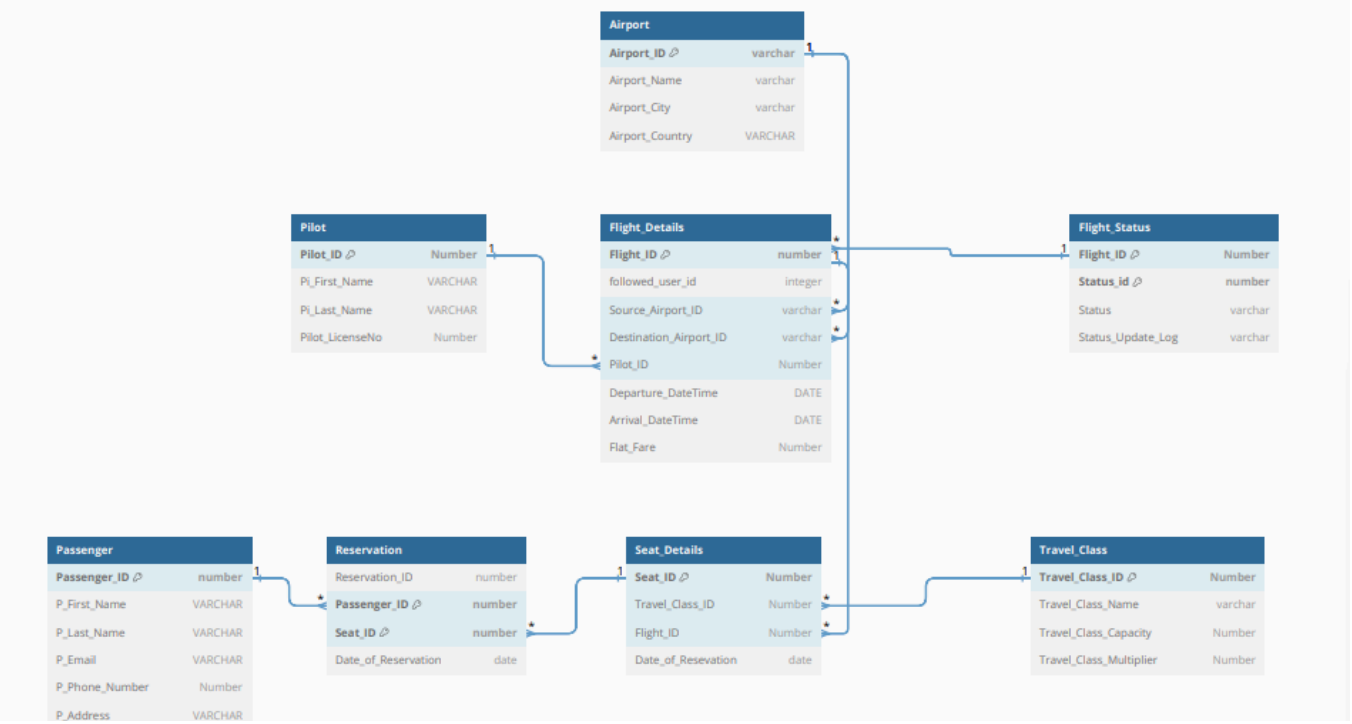

Figure 4.2 Schema Diagram

4.2 Data Dictionary

Airport

Column	Data type (size)	Constraint	Constraint Name
Airport_ID	Varchar(10)	Primary Key	Airport_id_Pr_Key
Airport_Name	Varchar(40)		
Airport_City	Varchar(40)		
Airport_Country	Varchar(40)		

Passenger

Column	Data type (size)	Constraint	Constraint Name
Passenger_ID	Varchar(10)	Primary Key	Pass_id_Pr_key
P_First_Name	Varchar(15)		
P_Last_Name	Varchar(15)		
P_Email	Varchar(50)	Must have “@” and “.”	Em_format
P_Phone_Number	Number(10)	Unique, must have 10 digits.	Phone_length
P_Address	Varchar(50)		

Travel_Class

Column	Data type (size)	Constraint	Constraint Name
Travel_Class_ID	Varchar(10)	Primary Key	Travel_Class_PK
Travel_Class_Name	Varchar(20)	Valid classes-First class, Business class, Premium economy, Economy class, Basic economy	Class_name_check
Travel_Class_Capacity	Number(3)		
Travel_Class_Multiplier	Number(5)		

Pilot

Column	Data type (size)	Constraint	Constraint Name
Pilot_ID	Varchar(10)	Primary Key	Pilot_ID_PK
Pi_First_Name	Varchar(15)		
Pi_Last_Name	Varchar(15)		
Pilot_LicenseNo	Varchar(15)		

Flight_Details

Column	Data type (size)	Constraint	Constraint Name
Flight_ID	Varchar(10)	Primary Key	Flight_ID_PK
Source_Airport_ID	Varchar(10)	Foreign Key-references airport	Source_Airport_ID_FK
Destination_Airport_ID	Varchar(10)	Foreign Key-references airport	Destination_Airport_ID_FK
Pilot_ID	Varchar(10)	Foreign Key-references pilot	Pilot_ID_Number_FK
Departure_DateTime	Date		
Arrival_DateTime	Date	Valid if after Departure_Date Time	TimeAndDate_matching
Flat_Fare	Number	Valid if greater than zero	

Seat_Details

Column	Data type (size)	Constraint	Constraint Name
Seat_ID	Varchar(10)	Primary Key	Seat_ID_PK
Travel_Class_ID	Varchar(10)	Foreign Key-references Travel_Class	Travel_Class_ID_FK
Flight_ID	Varchar(10)	Foreign Key-references Flight_Details	Flight_ID_FK

Reservation

Column	Data type (size)	Constraint	Constraint Name
Reservation_ID	Varchar(10)	Primary Key	Reservation_ID_PK
Passenger_ID	Varchar(10)	Foreign Key-references Passenger	Passenger_ID_FK
Seat_ID	Varchar(10)	Foreign Key-references Seat_Details	Seat_ID_FK
Date_of_Reservation	Date		

Flight_Status

Column	Data type (size)	Constraint	Constraint Name
Flight_ID	Varchar(10)	Foreign Key-references Flight_Details	Flight_ID_FK2
Status	Varchar(15)	Valid statuses- Scheduled, Boarding, Delayed, In air, Arrived, Diverted	Status_Check
Status_Update_Log	Date		

4.3 Relational Model Implementation

4.3.1 Create Commands

```
CREATE TABLE Airport (
   Airport_ID varchar(10) constraint Airport_id_Pr_key PRIMARY
   KEY, Airport_Name VARCHAR(40),
   Airport_City VARCHAR(40),
   Airport_Country
   VARCHAR(40)
);

CREATE TABLE Passenger (
   Passenger_ID varchar(10) constraint Pass_id_Pr_key PRIMARY
   KEY, P_First_Name VARCHAR(15),
   P_Last_Name VARCHAR(15),
   P_Email VARCHAR(50) constraint em_format check (P_Email like '%@%.%'),
   P_Phone_Number Number(10) constraint phone_unq UNIQUE
constraint phone_length check(P_Phone_Number like '__'),
   P_Address VARCHAR(50)
);

CREATE TABLE Travel_Class (
   Travel_Class_ID Varchar(10) constraint Travel_Class_PK PRIMARY
   KEY, Travel_Class_Name VARCHAR(20) constraint class_name_check
   check (
Travel_Class_Name in('First Class','Business Class','Premium
Economy','Economy Class','Basic Economy')),
   Travel_Class_Capacity Number(3),
   Travel_Class_Multiplier Number(5)
);

CREATE TABLE Pilot (
   Pilot_ID varchar(10) constraint Pilot_ID_PK PRIMARY
   KEY, Pi_First_Name VARCHAR(15),
   Pi_Last_Name VARCHAR(15),
   Pilot_LicenseNo varchar(7) constraint UNQ_Licence UNIQUE
);

CREATE TABLE Flight_Details (
   Flight_ID varchar(10) constraint Flight_ID_PK PRIMARY KEY,
   Source_Airport_ID varchar(10) constraint Source_Airport_ID_FK references Airport,
   Destination_Airport_ID varchar(10) constraint Destination_Airport_ID_FK
   references Airport
```

```
    Pilot_ID varchar(10) constraint Pilot_ID_Number_FK references Pilot,
    Departure_DateTime DATE,
    Arrival_DateTime DATE,
    Flat_Fare Number check(Flat_Fare>0),
    constraint source_dest_notsame check(Source_Airport_ID !=
    Destination_Airport_ID), constraint timeanddate_matching
    check(Departure_DateTime<Arrival_DateTime)
    );

CREATE TABLE Seat_Details (
    Seat_ID varchar(10) constraint Seat_ID_PK PRIMARY KEY,
    Travel_Class_ID varchar(10) constraint Travel_Class_ID_FK references Travel_Class ,
    Flight_ID varchar(10) constraint Flight_ID_FK references Flight_Details
    );

CREATE TABLE Reservation (
    Reservation_ID varchar(10) constraint Reservation_ID_PK PRIMARY KEY,
    Passenger_ID varchar(10) constraint Passenger_ID_FK references
    Passenger, Seat_ID varchar(10) constraint Seat_ID_FK references
    Seat_Details , Date_of_Reservation DATE
    );

CREATE TABLE Flight_Status (
    Flight_ID varchar(10) constraint Flight_ID_FK2 References Flight_Details,
    Status VARCHAR(15) constraint status_check check (Status
in ('Scheduled','Boarding','Delayed','In air','Arrived','Diverted')),
    Status_Update_Log Date
);
```

4.3.2 Insert Statements

```
insert all
into passenger values('P1', 'John', 'Doe', 'john.doe@gmail.com', '1234567890', '123 Main St,
New York, USA')
into passenger values('P2', 'Jane', 'Smith', 'jane.smith@yahoo.com', '1987654321', '456 Elm
St, Vancouver, Canada')
into passenger values('P3', 'Alice', 'Johnson', 'alice.johnson@gmail.com', '1122334455', '789
Oak St, Atlanta, USA')
into passenger values('P4', 'Michael', 'Brown', 'michael.brown@yahoo.com', '1654321876', '321
Pine St, Washington, USA')
into passenger values('P5', 'Emily', 'Wilson', 'emily.wilson@outlook.com','1789456123',
'654 Cedar St, Montreal, Canada')
select * from dual;
```

```
INSERT ALL
INTO Reservation VALUES ('R1', 'P1', 'S1', to_date('26-03-2024','dd-mm-yyyy'))
INTO Reservation VALUES ('R2', 'P1', 'S2', to_date('27-03-2024','dd-mm-yyyy'))
INTO Reservation VALUES ('R3', 'P2', 'S3', to_date('28-03-2024','dd-mm-yyyy'))
INTO Reservation VALUES ('R4', 'P2', 'S4', to_date('29-03-2024','dd-mm-yyyy'))
INTO Reservation VALUES ('R5', 'P3', 'S5', to_date('30-03-2024','dd-mm-yyyy'))
INTO Reservation VALUES ('R6', 'P3', 'S6', to_date('31-03-2024','dd-mm-yyyy'))
INTO Reservation VALUES ('R7', 'P4', 'S7', to_date('01-04-2024','dd-mm-yyyy'))
INTO Reservation VALUES ('R8', 'P4', 'S8', to_date('02-04-2024','dd-mm-yyyy'))
INTO Reservation VALUES ('R9', 'P5', 'S9', to_date('03-04-2024','dd-mm-yyyy'))
INTO Reservation VALUES ('R10', 'P5', 'S10',
to_date('04-04-2024','dd-mm-yyyy')) SELECT * FROM dual;

INSERT ALL
INTO Airport VALUES ('A1', 'LAX', 'Los Angeles',
'USA') INTO Airport VALUES ('A2', 'DEN', 'Denver',
'USA')
INTO Airport VALUES ('A3', 'YVR', 'Vancouver',
'Canada') into airport values('A4',’JFK’,’New York’,’USA’)
SELECT * FROM dual;

INSERT ALL
INTO Pilot VALUES ('PL1', 'Michael', 'Johnson',
'ABC123') INTO Pilot VALUES ('PL2', 'Emily', 'Davis',
'DEF456')
INTO Pilot VALUES ('PL3', 'Daniel', 'Martinez',
'GHI789') INTO Pilot VALUES ('PL4', 'Jessica', 'Lopez',
'JKL012')
SELECT * FROM dual;

INSERT ALL
INTO Travel_Class VALUES ('TC1', 'Economy Class', 200,
1.0) INTO Travel_Class VALUES ('TC2', 'Business Class', 50,
1.75) INTO Travel_Class VALUES ('TC3', 'First Class', 20, 2.0)
INTO Travel_Class VALUES('TC4','Premium
Economy',100,1.25) INTO Travel_Class VALUES('TC5','Basic
Economy',75,1.5) SELECT * FROM dual;

INSERT ALL
INTO Seat_Details VALUES ('S1', 'TC1', 'F2')
INTO Seat_Details VALUES ('S2', 'TC2', 'F2')
INTO Seat_Details VALUES ('S3', 'TC2', 'F3')
INTO Seat_Details VALUES ('S4', 'TC1', 'F4')
INTO Seat_Details VALUES ('S5', 'TC1', 'F5')
INTO Seat_Details VALUES ('S6', 'TC1', 'F6')
INTO Seat_Details VALUES ('S7', 'TC2', 'F7')
INTO Seat_Details VALUES ('S8', 'TC2', 'F8')
INTO Seat_Details VALUES ('S9', 'TC1', 'F9')
INTO Seat_Details VALUES ('S10', 'TC1',
'F10')
SELECT * FROM dual;
```

```
INSERT ALL
  INTO Flight_Details VALUES ('F2', 'A2', 'A1', 'PL2', to_date('2024-04-02 09:00:00',
  'yyyy-mm-dd hh24:mi:ss'), to_date('2024-04-02 11:00:00', 'yyyy-mm-dd hh24:mi:ss'),
  250.00) INTO Flight_Details VALUES ('F3', 'A3', 'A4', 'PL3', to_date('2024-04-03 10:00:00',
  'yyyy-mm-dd hh24:mi:ss'), to_date('2024-04-03 12:00:00', 'yyyy-mm-dd hh24:mi:ss'), 300.00)
  INTO Flight_Details VALUES ('F4', 'A4', 'A1', 'PL1', to_date('2024-04-04 11:00:00',
  'yyyy-mm-dd hh24:mi:ss'), to_date('2024-04-04 13:00:00', 'yyyy-mm-dd hh24:mi:ss'), 280.00)
  INTO Flight_Details VALUES ('F5', 'A1', 'A3', 'PL2', to_date('2024-04-05 12:00:00',
  'yyyy-mm-dd hh24:mi:ss'), to_date('2024-04-05 14:00:00', 'yyyy-mm-dd hh24:mi:ss'), 320.00)
  INTO Flight_Details VALUES ('F6', 'A3', 'A1', 'PL3', to_date('2024-04-06 13:00:00',
  'yyyy-mm-dd hh24:mi:ss'), to_date('2024-04-06 15:00:00', 'yyyy-mm-dd hh24:mi:ss'), 270.00)
  INTO Flight_Details VALUES ('F7', 'A2', 'A4', 'PL1', to_date('2024-04-07 14:00:00',
  'yyyy-mm-dd hh24:mi:ss'), to_date('2024-04-07 16:00:00', 'yyyy-mm-dd hh24:mi:ss'), 310.00)
  INTO Flight_Details VALUES ('F8', 'A4', 'A2', 'PL2', to_date('2024-04-08 15:00:00',
  'yyyy-mm-dd hh24:mi:ss'), to_date('2024-04-08 17:00:00', 'yyyy-mm-dd hh24:mi:ss'), 290.00)
  INTO Flight_Details VALUES ('F9', 'A1', 'A4', 'PL3', to_date('2024-04-09 16:00:00',
  'yyyy-mm-dd hh24:mi:ss'), to_date('2024-04-09 18:00:00', 'yyyy-mm-dd hh24:mi:ss'), 330.00)
  INTO Flight_Details VALUES ('F10', 'A4', 'A1', 'PL1', to_date('2024-04-10 17:00:00',
  'yyyy-mm-dd hh24:mi:ss'), to_date('2024-04-10 19:00:00', 'yyyy-mm-dd hh24:mi:ss'), 310.00)
  SELECT * FROM dual;
```

Chapter 5

Implementation

5.1 Triggers

A. Updating flight_status upon insert of records into flight_details. When a record is inserted into flight_details then a new record is inserted into flight_status with the flight_id, status being ‘scheduled’ and current time and date as the status_update_log.

```
CREATE OR REPLACE TRIGGER
update_flight_status_trigger AFTER INSERT ON flight_details
FOR EACH
ROW BEGIN
   INSERT INTO flight_status (flight_id, status, status_update_log)
   VALUES (:NEW.flight_id, 'Scheduled', SYSDATE);
END;
/
```

B. Setting status update log to current system date and time whenever flight status is updated for any flight.

```
CREATE OR REPLACE TRIGGER Update_Status_Log_Trigger
BEFORE UPDATE ON Flight_Status
FOR EACH ROW
BEGIN
UPDATE Flight_Status
SET Status_Update_Log = SYSDATE
WHERE Flight_ID = :NEW.Flight_ID;
END;
/
```

5.2 Stored Procedures

A. Procedure to find all flights from a given source to destination

```
CREATE or REPLACE PROCEDURE GetAvailableFlightsWithSchedule
   ( p_SourceAirport IN VARCHAR2,
   p_DestinationAirport IN VARCHAR2
) as
p_source_airport_id varchar(10);
p_destination_airport_id varchar(10);

BEGIN
 SELECT Airport_ID INTO p_source_airport_id
```

```
    FROM Airport
    WHERE Airport_Name = p_SourceAirport;

    SELECT Airport_ID INTO p_destination_airport_id
    FROM Airport
    WHERE Airport_Name = p_DestinationAirport;

    IF p_source_airport_id IS NOT NULL AND p_destination_airport_id IS NOT NULL THEN
       FOR flight_rec IN (
          SELECT F.Flight_id,
                TC.Travel_Class_Name
                ,
                TC.Travel_Class_Multiplier * F.Flat_Fare AS Cost,
                TO_CHAR(F.Departure_DateTime, 'HH24:MI:SS DD:MM:YYYY') AS
DepartureDateTime,
                TO_CHAR(F.Arrival_DateTime, 'HH24:MI:SS DD:MM:YYYY') AS
ArrivalDateTime
          FROM Flight_Details F INNER JOIN Seat_Details SD ON F.Flight_id = SD.Flight_ID
INNER JOIN Travel_Class TC ON SD.Travel_Class_ID = TC.Travel_Class_ID
          WHERE F.Source_Airport_ID =p_source_airport_id
          AND F.Destination_Airport_ID = p_destination_airport_id
       )
       LOOP
          DBMS_OUTPUT.PUT_LINE('Flight ID: ' || flight_rec.Flight_id ||
                        ', Travel Class: ' || flight_rec.Travel_Class_Name ||
                        ', Cost: ' || flight_rec.Cost ||
                        ', Departure Time: ' || flight_rec.DepartureDateTime ||
                        ', Arrival Time: ' || flight_rec.ArrivalDateTime);
       END
    LOOP; ELSE
       DBMS_OUTPUT.PUT_LINE('Invalid source or destination
    airport'); END IF;
END ;
/
```

EXECUTION OF PROCEDURE PL/SQL BLOCK-

```
SET SERVEROUTPUT ON;
DECLARE
s_a varchar(10) :='&source_aiport';
d_a varchar(10):='&destination_airport';
BEGIN
   GetAvailableFlightsWithSchedule(s_a, d_a);
END;
/
```

B. To find trips under a given budget.

```
CREATE or REPLACE PROCEDURE BudgetFlightFinder(max_cost in number
) as
BEGIN
      FOR flight_rec IN (
```

```
        SELECT F.Flight_id,
            TC.Travel_Class_Name,
            TC.Travel_Class_Multiplier * F.Flat_Fare AS Cost,
            TO_CHAR(F.Departure_DateTime, 'HH24:MI:SS DD:MM:YYYY') AS
DepartureDateTime,
            TO_CHAR(F.Arrival_DateTime, 'HH24:MI:SS DD:MM:YYYY') AS
ArrivalDateTime,
A.Airport_Name as Source_Airport, B.Airport_Name as Destination_Airport
        FROM Flight_Details F  INNER JOIN Seat_Details SD ON F.Flight_id = SD.Flight_ID
INNER JOIN Travel_Class TC ON SD.Travel_Class_ID = TC.Travel_Class_ID, airport A,
airport B
        WHERE (TC.Travel_Class_Multiplier * F.Flat_Fare)<=max_cost and
f.source_airport_id= A.airport_id and f.destination_airport_id=B.airport_id

    )
    LOOP
      DBMS_OUTPUT.PUT_LINE('Flight ID: ' || flight_rec.Flight_id ||
                    ', Travel Class: ' || flight_rec.Travel_Class_Name ||
                    ', Cost: ' || flight_rec.Cost ||
', Source Airport: ' || flight_rec.Source_Airport ||
                    ', Departure Time: ' || flight_rec.DepartureDateTime ||
', Destination Airport: ' || flight_rec.Destination_Airport ||
                    ', Arrival Time: ' || flight_rec.ArrivalDateTime);
    END LOOP;
END ;
/
```

EXECUTION OF PROCEDURE PL/SQL BLOCK-

```
SET SERVEROUTPUT ON;
DECLARE
max_cost NUMBER := &budget;
BEGIN
   BudgetFlightFinder(max_cost);
END;
/
```

C. Find the current flight status of any given flight.

```
CREATE OR REPLACE PROCEDURE FindFlightStatus (
   p_FlightID IN VARCHAR2 DEFAULT NULL,
   p_SourceAirport IN VARCHAR2 DEFAULT NULL,
   p_DestinationAirport IN VARCHAR2 DEFAULT NULL
)
AS
BEGIN
   IF p_FlightID IS NOT NULL THEN
      -- Retrieve flight status by flight ID
      FOR flight_rec IN (
         SELECT FS.Flight_ID, FS.Status, TO_CHAR(FS.Status_Update_Log, 'HH24:MI:SS
```

```
DD:MM:YYYY') AS Status_Update_Log
        FROM Flight_Status FS
        WHERE FS.Flight_ID = p_FlightID
    )
    LOOP
        DBMS_OUTPUT.PUT_LINE('Flight ID: ' || flight_rec.Flight_ID ||
                    ', Status: ' || flight_rec.Status ||
                    ', Status Update Log: ' || flight_rec.Status_Update_Log);
    END LOOP;
  ELSIF p_SourceAirport IS NOT NULL AND p_DestinationAirport IS NOT NULL THEN
    FOR flight_rec IN (
        SELECT F.Flight_ID, FS.Status, TO_CHAR(FS.Status_Update_Log, 'HH24:MI:SS
DD:MM:YYYY') AS Status_Update_Log
        FROM Flight_Status FS
        INNER JOIN Flight_Details F ON FS.Flight_ID = F.Flight_ID
        INNER JOIN Airport ASrc ON F.Source_Airport_ID = ASrc.Airport_ID
        INNER JOIN Airport ADst ON F.Destination_Airport_ID = ADst.Airport_ID
        WHERE ASrc.Airport_Name = p_SourceAirport
        AND ADst.Airport_Name = p_DestinationAirport
    )
    LOOP
        DBMS_OUTPUT.PUT_LINE('Flight ID: ' || flight_rec.Flight_ID ||
                    ', Status: ' || flight_rec.Status ||
                    ', Status Update Log: ' || flight_rec.Status_Update_Log);
    END LOOP;
  ELSE
    DBMS_OUTPUT.PUT_LINE('Please provide either Flight ID or both Source and
Destination Airports.');
  END IF;
END;
/
```

EXECUTION OF PROCEDURE PL/SQL BLOCK-

Input- flightid

```
SET SERVEROUTPUT ON;

DECLARE
   p_FlightID VARCHAR2(100);
   p_SourceAirport VARCHAR2(100);
   p_DestinationAirport VARCHAR2(100);
BEGIN
   p_FlightID := NULL;
   p_SourceAirport := NULL;
   p_DestinationAirport := NULL;

  p_FlightID := '&Enter_Flight_ID';
   FindFlightStatus(p_FlightID, p_SourceAirport, p_DestinationAirport);
END;
/
```

Input- source and destination airport

```
SET SERVEROUTPUT ON;

DECLARE
   p_FlightID VARCHAR2(100);
   p_SourceAirport VARCHAR2(100);
   p_DestinationAirport VARCHAR2(100);
BEGIN
   p_FlightID := NULL;
   p_SourceAirport := NULL;
   p_DestinationAirport := NULL;
 p_SourceAirport := '&Enter_Source_Airport';
 p_DestinationAirport := '&Enter_Destination_Airport';
   FindFlightStatus(p_FlightID, p_SourceAirport, p_DestinationAirport);
END;
/
```

D. Find all reservations for a given passenger

```
CREATE OR REPLACE PROCEDURE FindReservationsForPassenger (
   p_PassengerID IN VARCHAR2
)
AS
BEGIN
   FOR reservation_rec IN (
      SELECT R.Reservation_ID, R.Passenger_ID, R.Seat_ID, R.Date_of_Reservation,
           F.Flight_id, F.Source_Airport_ID, F.Destination_Airport_ID,
           TO_CHAR(F.Departure_DateTime, 'HH24:MI:SS DD:MM:YYYY') AS
DepartureDateTime,
           TO_CHAR(F.Arrival_DateTime, 'HH24:MI:SS DD:MM:YYYY') AS ArrivalDateTime
      FROM Reservation R
      INNER JOIN Seat_Details SD ON R.Seat_ID = SD.Seat_ID
      INNER JOIN Flight_Details F ON SD.Flight_ID = F.Flight_ID
      WHERE R.Passenger_ID = p_PassengerID
   )
   LOOP
      DBMS_OUTPUT.PUT_LINE('Reservation ID: ' || reservation_rec.Reservation_ID
                  || ', Passenger ID: ' || reservation_rec.Passenger_ID ||
                  ', Seat ID: ' || reservation_rec.Seat_ID ||
                  ', Date of Reservation: ' || TO_CHAR(reservation_rec.Date_of_Reservation,
'HH24:MI:SS DD:MM:YYYY') ||
                  ', Flight ID: ' || reservation_rec.Flight_id ||
                  ', Departure Time: ' || reservation_rec.DepartureDateTime ||
                  ', Arrival Time: ' || reservation_rec.ArrivalDateTime);
   END LOOP;
END;
/
```

EXECUTION OF PROCEDURE PL/SQL BLOCK-

```
SET SERVEROUTPUT ON;
DECLARE
```

```
pid varchar(10) :='&passenger_id';
BEGIN
   FindReservationsForPassenger(pid);
END;
/
```

E. Find the itinerary of a given pilot

```
CREATE OR REPLACE PROCEDURE FindFlightsForPilot (
   p_PilotID IN VARCHAR2
)
AS
BEGIN
   FOR flight_rec IN (
      SELECT F.Flight_id, ASource.Airport_Name AS Source_Airport,
ADestination.Airport_Name AS Destination_Airport,
            TO_CHAR(F.Departure_DateTime, 'HH24:MI:SS DD:MM:YYYY')
                                                AS
DepartureDateTime,
          TO_CHAR(F.Arrival_DateTime, 'HH24:MI:SS DD:MM:YYYY') AS ArrivalDateTime
      FROM Flight_Details F
      INNER JOIN Airport ASource ON F.Source_Airport_ID = ASource.Airport_ID
      INNER JOIN Airport ADestination ON F.Destination_Airport_ID =
ADestination.Airport_ID
      WHERE F.Pilot_ID = p_PilotID
   )
   LOOP
      DBMS_OUTPUT.PUT_LINE('Flight ID: ' || flight_rec.Flight_id ||
                  ', Source Airport: ' || flight_rec.Source_Airport ||
                  ', Destination Airport: ' || flight_rec.Destination_Airport ||
                  ', Departure Time: ' || flight_rec.DepartureDateTime ||
                  ', Arrival Time: ' || flight_rec.ArrivalDateTime);
   END LOOP;
END;
/
```

EXECUTION OF PROCEDURE PL/SQL BLOCK-

```
SET SERVEROUTPUT ON;
DECLARE
pid varchar2(10) :=’&pilot_id’;
BEGIN
   FindFlightsForPilot(pid);
END;
/
```

Chapter 6

Result

A. Source to Destination flight finder

```
Enter value for source_aiport: JFK
old   2: s_a varchar(10) :='&source_aiport';
new   2: s_a varchar(10) :='JFK';
Enter value for destination_airport: LAX
old   3: d_a varchar(10):='&destination_airport';
new   3: d_a varchar(10):='LAX';
Flight ID: F4, Travel Class: Economy Class, Cost: 280, Departure Time: 11:00:00
04:04:2024, Arrival Time: 13:00:00 04:04:2024
Flight ID: F10, Travel Class: Economy Class, Cost: 310, Departure Time: 17:00:00
10:04:2024, Arrival Time: 19:00:00 10:04:2024
```

The program takes input of ‘JFK’ as the source airport and ‘LAX’ as destination airport, it gives all flights with the corresponding flight details.

B. Budget Trip Finder

```
Enter value for budget: 400
old   2:     max_cost NUMBER := &budget; -- Prompt for user input for the budget
new   2:     max_cost NUMBER := 400; -- Prompt for user input for the budget
Flight ID: F2, Travel Class: Economy Class, Cost: 250, Source Airport: DEN,
Departure Time: 09:00:00 02:04:2024, Destination Airport: LAX, Arrival Time:
11:00:00 02:04:2024
Flight ID: F6, Travel Class: Economy Class, Cost: 270, Source Airport: YVR,
Departure Time: 13:00:00 06:04:2024, Destination Airport: LAX, Arrival Time:
15:00:00 06:04:2024
Flight ID: F4, Travel Class: Economy Class, Cost: 280, Source Airport: JFK,
Departure Time: 11:00:00 04:04:2024, Destination Airport: LAX, Arrival Time:
13:00:00 04:04:2024
Flight ID: F10, Travel Class: Economy Class, Cost: 310, Source Airport: JFK,
Departure Time: 17:00:00 10:04:2024, Destination Airport: LAX, Arrival Time:
19:00:00 10:04:2024
Flight ID: F5, Travel Class: Economy Class, Cost: 320, Source Airport: LAX,
Departure Time: 12:00:00 05:04:2024, Destination Airport: YVR, Arrival Time:
14:00:00 05:04:2024
Flight ID: F9, Travel Class: Economy Class, Cost: 330, Source Airport: LAX,
Departure Time: 16:00:00 09:04:2024, Destination Airport: JFK, Arrival Time:
18:00:00 09:04:2024
```

The procedure finds all flights under the given max budget of 400 with all important and salient flight details

C. Flight status finder

```
Enter value for enter_flight_id: F4
old  10:     p_FlightID := '&Enter_Flight_ID';
new  10:     p_FlightID := 'F4';
Flight ID: F4, Status: Scheduled, Status Update Log: 07:36:03 05:04:2024
```

The procedure takes in flight id and the latest status for the corresponding flight along with when

the status was last updated is given.

```
Enter value for enter_source_airport: JFK
old   9:  p_SourceAirport := '&Enter_Source_Airport';
new   9:  p_SourceAirport := 'JFK';
Enter value for enter_destination_airport: LAX
old  10:  p_DestinationAirport := '&Enter_Destination_Airport';
new  10:  p_DestinationAirport := 'LAX';
Flight ID: F4, Status: Scheduled, Status Update Log: 07:36:03 05:04:2024
Flight ID: F10, Status: Scheduled, Status Update Log: 07:36:03 05:04:2024
```

The procedure can also take the source and destination airport and output all flights with their corresponding status info.

D. Flight reservation check

```
Enter value for passenger_id: P3
old   2: pid varchar(10) :='&passenger_id';
new   2: pid varchar(10) :='P3';
Reservation ID: R5, Passenger ID: P3, Seat ID: S5, Date of Reservation: 00:00:00
30:03:2024, Flight ID: F5, Departure Time: 12:00:00 05:04:2024, Arrival Time:
14:00:00 05:04:2024
Reservation ID: R6, Passenger ID: P3, Seat ID: S6, Date of Reservation: 00:00:00
31:03:2024, Flight ID: F6, Departure Time: 13:00:00 06:04:2024, Arrival Time:
15:00:00 06:04:2024
```

The procedure gives all reservations made by a passenger given their respective passenger id.

E. Pilot itinerary

```
Enter value for pilot_id: PL2
old   2: pid varchar2(10) :='&pilot_id';
new   2: pid varchar2(10) :='PL2';
Flight ID: F2, Source Airport: DEN, Destination Airport: LAX, Departure Time:
09:00:00 02:04:2024, Arrival Time: 11:00:00 02:04:2024
Flight ID: F5, Source Airport: LAX, Destination Airport: YVR, Departure Time:
12:00:00 05:04:2024, Arrival Time: 14:00:00 05:04:2024
Flight ID: F8, Source Airport: JFK, Destination Airport: DEN, Departure Time:
15:00:00 08:04:2024, Arrival Time: 17:00:00 08:04:2024
```

This procedure can be used by any pilot to find their schedule by entering their pilot id.

Chapter 7

Conclusion And Scope

7.1 Conclusion

In summary, the incorporation of passenger details into our flight database management system sets up a sturdy foundation for efficient operations and exceptional customer service within the airline industry. By ensuring data integrity and providing alternative identifiers like Passenger ID, we have mitigated the risk of inconvenience and potential data loss, thereby preserving a positive passenger experience. The separation of attributes is made in such a way that it is simultaneously easy to access and minimises data integrity breaches thus making it a robust system.

Moreover, the insights derived from analysing passenger behaviours and flight patterns offer invaluable opportunities for informed decision-making within the aviation sector. Finding trends such as preferred routes, peak travel periods, and popular travel classes enables airlines to tailor their services, personalise passenger experiences, and strategically plan for future demand. This initiative-taking approach not only enhances passenger satisfaction but also perfects revenue generation and operational effectiveness, positioning airlines for sustained success in a dynamic and competitive market landscape.

7.2 Scope for future work

This project is but a mere proof of concept upon which a palace must be constructed for this system to be consumer ready, the facets of the project that can be improved upon are:-

1. A user interface for customers to access data and buy tickets with the information provided
2. A portal for airline personnel to update the database easily without intimate knowledge of SQL.
3. Expansion of the database to encompass a holistic view of the aviation industry i.e., including ground operations info, flight crew, airline, flight maintenance, etc.
4. Combining AI to help with efficient route planning and automatic personnel assignment.
5. Security measures to not compromise sensitive information.

The aviation world is broad with massive amounts of crucial data and a database is the best way to store, access, update and manipulate the data. But such a database will be massive, the rise of artificial intelligence and data analytics also make this domain one that can be worked upon and perfected increasingly with time.

Each Team Member Contribution:

Team Member	Contribution
Bhuvan Battu	Code implementation and compilation, Ideation
Adi Mukherjee	Insert commands, Ideation
Mehul Garg	Triggers, Data Dictionary, Documentation
Ranveer Singh Sayal	ER Diagram, Create Commands, Documentation
Nayavanth Yogi	Procedures, Documentation, Ideation
Meghashyam Shenoy	Schema Diagram, Documentation