DATABASE MANAGEMENT SYSTEM-PROJECT REPORT

FLIGHT MANAGEMENT SYSTEM

SUBMITTEDTO:

DR.SHASHANK

SUBMITTED BY:

GURLEEN KAUR (102203197)

SIMRAT KAUR (102203201)

JAISMEEN KAUR (102203207)

EKLEEN KAUR (102203211)

INDEX

PROBLEM STATEMENT

Design and implement a Flight Management System (FMS) using PL/SQL for efficient airline operations. The system will manage flight information such as schedules, bookings, seat availability, customer data, and flight status updates. Key features include a booking system with search and seat reservation capabilities, customer management functionalities, and reporting tools for generating flight schedules, passenger lists, and revenue reports. The project will focus on ensuring data integrity, handling concurrency, and providing a user-friendly interface for airline staff and customers. Evaluation criteria include functionality, performance, reliability, usability, and comprehensive documentation.

PROJECT DESCRIPTION

Airlines Integration: Design the FMS to seamlessly integrate with multiple airlines, allowing for efficient management of flights across different carriers.

Booking Management: Develop a robust booking system that enables passengers to make reservations, modify bookings, and cancel flights as needed.

Seat Allocation: Implement algorithms for automatic seat allocation based on passenger preferences and availability.

Check-in Processes: Streamline check-in processes by generating boarding passes and managing passenger documentation electronically.

Payment Integration: Integrate payment gateways to facilitate secure and convenient transactions for ticket purchases and additional services.

Customer Communication: Implement features for automated notifications and alerts to keep passengers informed about flight changes, upgrades, and promotions

Reporting and Analytics: Develop reporting tools to generate insights into booking trends, revenue analysis, and passenger demographics for strategic decision-making.

Mobile Accessibility: Ensure mobile compatibility for the FMS, allowing passengers to access their bookings, check-in, and receive updates via mobile devices.

Security Measures: Implement robust security measures to protect passenger data, financial transactions, and system integrity against cyber threats.

ENTITIES

CITY

Cname	State	Country

AIRPORT

Ap_name	State	Country

AIRLINE

Airline_ID	AI_Name	Three_digit_code

EMPLOYEE

SSN	Name	Address	Phone	Age	Job_type	Shift

PASSENGER

PID	Passport_No	Fname	Address	Phone	Age

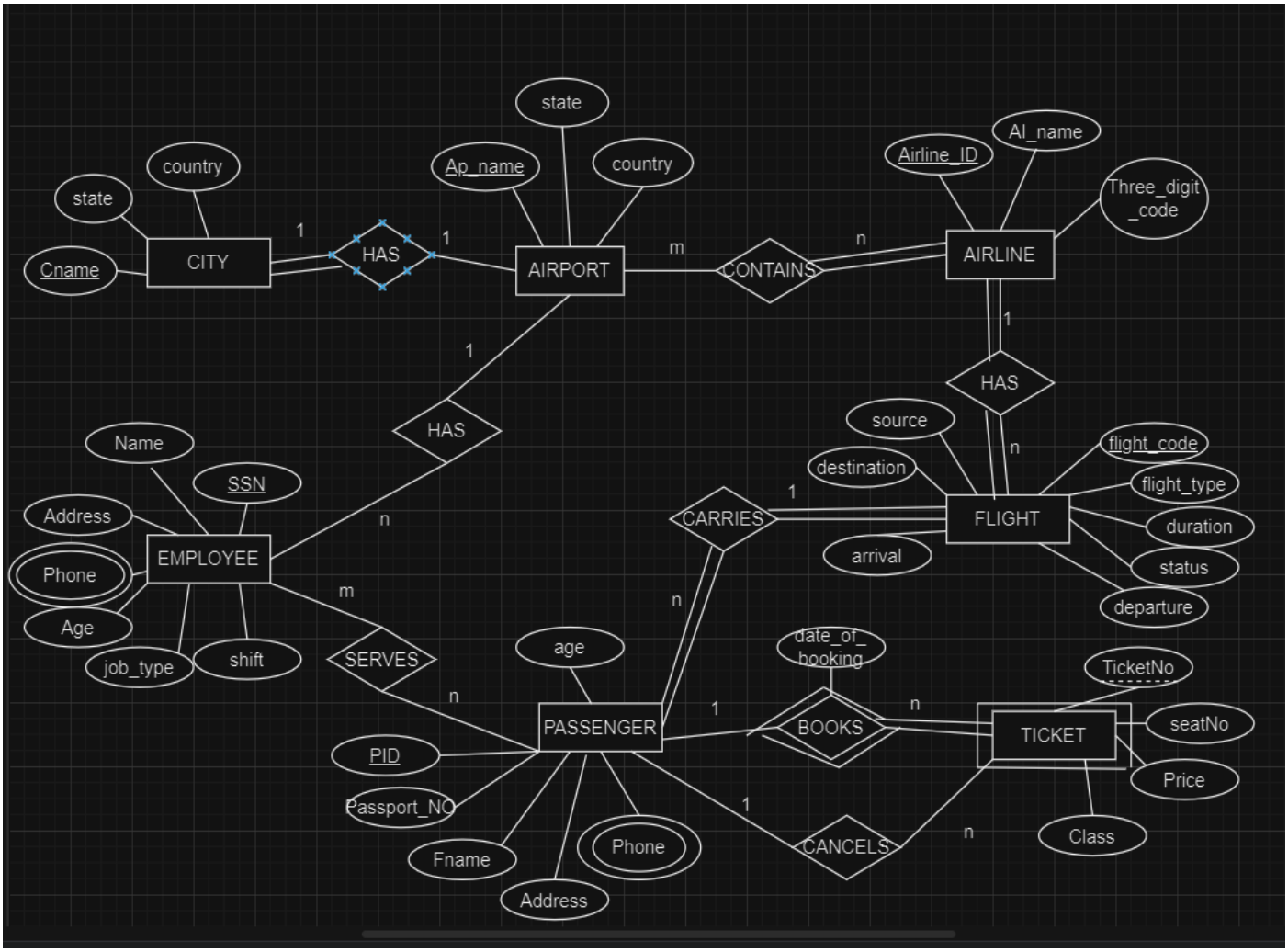
FLIGHT

Flight_code	Source	Destination	arrival	departure	Flight_type	Status	Duration

TICKET

Ticket_no	SeatNo	Price	Class

ENTITY-RELATIONSHIP DIAGRAM

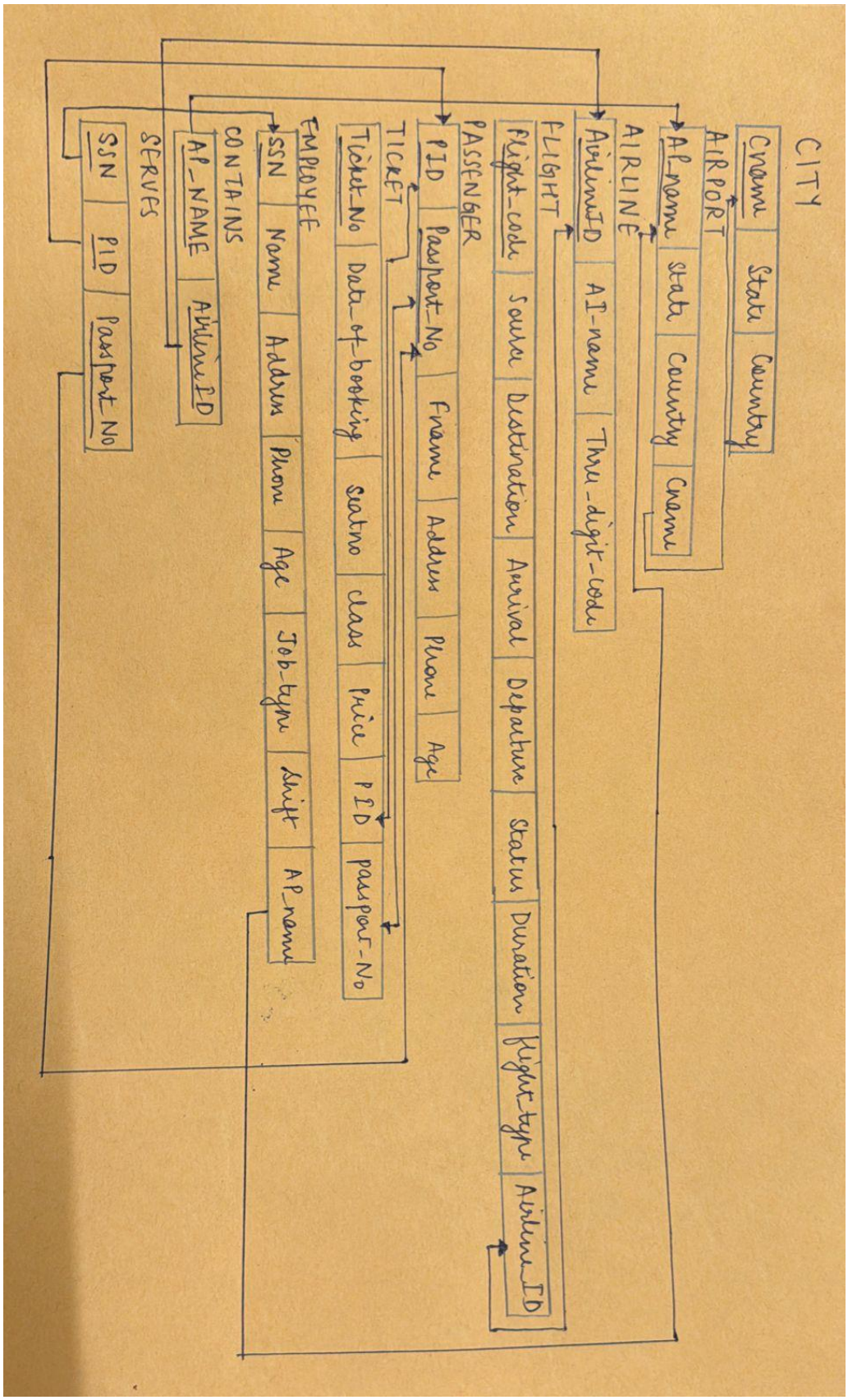


ER Diagram has following relationships:

Entity 1	Name of the Relationship	Entity 2	Cardinality
City	has	Airport	1:1
Airport	contains	Airline	m : n
Airport	has	Employee	1 : n
Airline	has	Flight	1 : n
Flight	carries	Passengers	1 : n
Employee	serves	Passengers	m : n
Passenger	books	Ticket	1 : n
Passenger	cancels	Ticket	1 : n

Type of the binary relationship	Relationships in the system
one-to-one	(1 A city has only one airport.
one-to-many	(1) An airline has multiple flights, that is many flights belong to the same airline company. (2) A flight carries many passengers. (3) A passenger can book one or more tickets. (4) A passenger can cancel one or more tickets.
many-to-many	All International airlines operating through various countries across the world have their offices located in all major cities and airports they cover. Hence, an airport may have many airline offices.

ER to relational schema mapping*:*



Functional Dependencies:

1. **CNAME -> Airport** (A city determines a unique airport, assuming there's only one airport per city)
2. **AIRPORT -> CNAME** (An airport belongs to a specific city)
3. **AIRPORT, AIRLINE -> Contains** (An airport and an airline together determine if the airline uses that airport)
4. **AIRLINE -> Contains** (An airline can use multiple airports, but we can't infer which airports an airline uses from the airline information alone)
5. **FLIGHT -> Flies** (A flight uses at least one airport) **[This may not hold if a flight has not been assigned to any airports yet]**
6. **Airport (source/destination)** -> Flight (The source/destination airport determines a flight) **[Assuming a flight has a unique source and destination combination]**
7. **PASSENGER -> Books** (A passenger can book multiple tickets)
8. **TICKET -> Passenger** (A ticket belongs to a specific passenger)
9. **EMPLOYEE -> Serves** (An employee can serve on multiple flights)
10. **FLIGHT -> Serves** (A flight is served by multiple employees)

Normalization Rules on Database

First Normal Form (1NF):

For a table to be in the First Normal Form, it should follow the following 4 rules:

1. It should only have single(atomic) valued attributes/columns.
2. Values stored in a column should be of the same domain
3. All the columns in a table should have unique names.
4. And the order in which data is stored, does not matter.

Normalization into 1NF:

A table is in 1NF if it meets the following conditions:
Each cell contains a single value.
There are no repeating groups of columns.
The tables derived from the ER diagram are already be in 1NF. Here, each table would contain the following attributes:

CITIES table:
Cname (City name)
state
country

AIRPORT table:
Ap name (Airport name)
Three_digit_code (Unique code for the airport)
Cname (foreign key referencing the Cname attribute in the CITIES table)

AIRLINE table:
Al name (Airline name)
Airline ID (Unique code for the airline)

FLIGHT table:
source (foreign key referencing the Three_digit_code attribute in the AIRPORT table)
destination (foreign key referencing the Three_digit_code attribute in the AIRPORT table)
flight_code (Unique code for the flight)
flight_type (e.g., commercial, cargo)
duration (flight time)

PASSENGER table:
PID (Unique passenger ID)

Name
SSN (Social Security Number)
Address
Phone
... other passenger attributes

TICKET table:
TicketNo (Unique ticket number)
PID (foreign key referencing the PID attribute in the PASSENGER table)
FLIGHT_source (foreign key referencing the source attribute in the FLIGHT table) [if applicable]
FLIGHT_destination (foreign key referencing the destination attribute in the FLIGHT table) [if applicable]
date_of_booking
Price
Class

EMPLOYEE table:
Employee ID (Unique employee ID)
Name
Age
job_type
shift
Address
Phone
Al name (foreign key referencing the Al name attribute in the AIRLINE table)

2. Second Normal Form (2NF):

For a table to be in the Second Normal Form
1. It should be in the First Normal form.
2. And, it should not have Partial Dependency.

1.FLIGHT table:
The FLIGHT_source and FLIGHT_destination attributes in the 1NF FLIGHT table are partially dependent on the composite key (source, destination) of the AIRPORT table.
To achieve 2NF, we can decompose the FLIGHT table into two tables:

FLIGHT table:
flight_code (Unique code for the flight)
flight_type (e.g., commercial, cargo)
duration (flight time)

FLIGHT_ROUTE table:
flight_code (foreign key referencing the flight_code attribute in the FLIGHT table)
source (foreign key referencing the Three_digit_code attribute in the AIRPORT table)
destination (foreign key referencing the Three_digit_code attribute in the AIRPORT table)

3. Third Normal Form (3NF):

A table is in 3NF if it meets the following conditions:
1.It is in 2NF.
2.No non-key attribute is dependent on another non-key attribute.

Normalization into 3NF:

Currently, the tables in 2NF don't have any transitive dependencies between non-key attributes. Therefore, they are also in 3NF.

SQL QUERIES

--Creating tables and inserting values—

1) CITY TABLE

```
SQL> CREATE table city(cname varchar(15) not null, state varchar2(15), country varchar(30),primary key(cname));
CREATE table city(cname varchar(15) not null, state varchar2(15), country varchar(30),primary key(cname))
             *
```

```
SQL>  insert into city(cname, state,country) values ('Louisville
','Kentucky','United States');

1 row created.

SQL> INSERT INTO CITY (CNAME, STATE, COUNTRY) VALUES ('Chandigarh','Chandigarh','lndia');

1 row created.

SQL> INSERT INTO CITY (CNAME, STATE, COUNTRY) VALUES ('Fort Worth','Texas','United States');

1 row created.

SQL> INSERT INTO CITY (CNAME, STATE, COUNTRY)VALUES('Delhi','Delhi','lndia');

1 row created.

SQL> INSERT INTO CITY (CNAME, STATE, COUNTRY) VALUES('Mumbai','Maharashtra','lndia');

1 row created.

SQL> INSERT INTO CITY (CNAME, STATE, COUNTRY) VALUES('San Francisco', 'California', 'United States');

1 row created.

SQL> INSERT INTO CITY (CNAME, STATE, COUNTRY) VALUES('Frankfurt','Hesse','Germany');

1 row created.

SQL> INSERT INTO CITY (CNAME, STATE, COUNTRY) VALUES('Houston','Texas','United States');

1 row created.
```

```
SQL> select * from city;

CNAME           STATE           COUNTRY
--------------- --------------- ------------------------------
Louisville      Kentucky        United States
Chandigarh      Chandigarh      lndia
Fort Worth      Texas           United States
Delhi           Delhi           lndia
Mumbai          Maharashtra     lndia
San Francisco   California      United States
Frankfurt       Hesse           Germany
Houston         Texas           United States
New York City   New York        United States
Tampa           Florida         United States

10 rows selected.
```

2) AIRPORT TABLE

```
SQL> CREATE TABLE AIRPORT(AP_NAME VARCHAR2(100) NOT NULL,STATE VARCHAR2(15), COUNTRY VARCHAR(30), CNAME VARCHAR2(15), PRIMARY KEY(AP_
NAME),FOREIGN KEY(CNAME) REFERENCES CITY(CNAME) ON DELETE CASCADE);

Table created.
```

```
SQL> INSERT INTO AIRPORT (AP_NAME, STATE, COUNTRY, CNAME) VALUES('Louisville InternationalAirport','Kentucky',' United  States','Loui
sville');

1 row created.

SQL> INSERT INTO AIRPORT (AP_NAME, STATE, COUNTRY, CNAME) VALUES('Chandigarh InternationalAirport','Chandigarh','India','Chandigarh')
;

1 row created.

SQL> INSERT INTO AIRPORT (AP_NAME, STATE, COUNTRY, CNAME) VALUES('Dallas/Fort Worth InternationalAirport','Texas','United States','Fo
rt  Worth');
INSERT INTO AIRPORT (AP_NAME, STATE, COUNTRY, CNAME) VALUES('Dallas/Fort Worth InternationalAirport','Texas','United States','Fort  W
orth')
*
ERROR at line 1:
ORA-02291: integrity constraint (COE203201.SYS_C0026659) violated - parent key
not found


SQL> INSERT INTO AIRPORT (AP_NAME, STATE, COUNTRY, CNAME) VALUES('lndira Gandhilnternational Airport',' Delhi','India','Delhi');


1 row created.

SQL> INSERT INTO AIRPORT (AP_NAME, STATE, COUNTRY, CNAME) VALUES('Chhatrapati Shivaji International Airport','Maharashtra  ','India',
'Mumbai');

1 row created.
```

```
SQL> select * from airport;

AP_NAME
--------------------------------------------------------------------------------
STATE           COUNTRY                        CNAME
--------------- ------------------------------ ---------------
Louisville InternationalAirport
Kentucky         United  States                Louisville

Chandigarh InternationalAirport
Chandigarh      India                          Chandigarh

lndira Gandhilnternational Airport
 Delhi          India                          Delhi
```

```
AP_NAME
--------------------------------------------------------------------------------
STATE           COUNTRY                        CNAME
--------------- ------------------------------ ---------------
Chhatrapati Shivaji International Airport
Maharashtra     India                          Mumbai

San Francisco InternationalAirport
California      United States                  San Francisco

FrankfurtAirport
Hesse           Germany                        Frankfurt
```

```
AP_NAME
--------------------------------------------------------------------------------
STATE           COUNTRY                        CNAME
--------------- ------------------------------ ---------------
George Bush IntercontinentalAirport
Texas           United States                  Houston

John F. Kennedy International_Airport
New York        United States                  New York City

Tampa International_Airport
Florida         United States                  Tampa
```

3) AIRLINE TABLE

```
SQL>  CREATE TABLE AIRLINE(AIRLINEID VARCHAR(3) NOT NULL, AL_NAME VARCHAR2(50),THREE_DIGIT_CODE VARCHAR(3), PRIMARY KEY(AIRLINEI
D));

Table created.
```

```
SQL> INSERT INTO AIRLINE (AIRLINEID, AL_NAME, THREE_DIGIT_CODE) VALUES('AA','American Airlines','001');

1 row created.

SQL> INSERT INTO AIRLINE (AIRLINEID, AL_NAME, THREE_DIGIT_CODE) VALUES('Al','Air India Limited','098');

1 row created.

SQL> INSERT INTO AIRLINE (AIRLINEID, AL_NAME, THREE_DIGIT_CODE) VALUES('LH','Lufthansa', '220');

1 row created.

SQL> INSERT INTO AIRLINE (AIRLINEID, AL_NAME, THREE_DIGIT_CODE) VALUES('BA','British Airways','125');

1 row created.

SQL> INSERT INTO AIRLINE (AIRLINEID, AL_NAME, THREE_DIGIT_CODE) VALUES('QR','Qatar Airways','157');

1 row created.

SQL> INSERT INTO AIRLINE (AIRLINEID, AL_NAME, THREE_DIGIT_CODE) VALUES('9W','Jet Airways','589');

1 row created.

SQL> INSERT INTO AIRLINE (AIRLINEID, AL_NAME, THREE_DIGIT_CODE) VALUES('EK','Emirates','176');

1 row created.

SQL> INSERT INTO AIRLINE (AIRLINEID, AL_NAME, THREE_DIGIT_CODE) VALUES('EY','Ethiad Airways','607');

1 row created.
```

```
SQL> select * from airline;

AIR AL_NAME                                           THR
--- ------------------------------------------------- ---
AA  American Airlines                                 001
Al  Air India Limited                                 098
LH  Lufthansa                                         220
BA  British Airways                                   125
QR  Qatar Airways                                     157
9W  Jet Airways                                       589
EK  Emirates                                          176
EY  Ethiad Airways                                    607

8 rows selected.
```

4) CONTAINS TABLE

```
SQL> CREATE TABLE CONTAINS (
  2      AIRLINEID VARCHAR(2) NOT NULL,
  3      AP_NAME VARCHAR(100) NOT NULL,
  4      PRIMARY KEY (AIRLINEID, AP_NAME),
  5      FOREIGN KEY (AIRLINEID) REFERENCES AIRLINE (AIRLINEID),
  6      FOREIGN KEY (AP_NAME) REFERENCES AIRPORT (AP_NAME)
  7  );

Table created.
```

```
SQL> INSERT INTO CONTAINS (AIRLINEID, AP_NAME) VALUES ('AA', 'Louisville InternationalAirport');

1 row created.

SQL> INSERT INTO CONTAINS (AIRLINEID, AP_NAME) VALUES ('Al', 'Chandigarh InternationalAirport');

1 row created.

SQL> INSERT INTO CONTAINS (AIRLINEID, AP_NAME) VALUES ('LH', 'lndira Gandhilnternational Airport');

1 row created.

SQL> INSERT INTO CONTAINS (AIRLINEID, AP_NAME) VALUES ('BA', 'Chhatrapati Shivaji International Airport');

1 row created.

SQL> INSERT INTO CONTAINS (AIRLINEID, AP_NAME) VALUES ('QR', 'San Francisco InternationalAirport');

1 row created.

SQL> INSERT INTO CONTAINS (AIRLINEID, AP_NAME) VALUES ('9W', 'FrankfurtAirport');

1 row created.

SQL> INSERT INTO CONTAINS (AIRLINEID, AP_NAME) VALUES ('EK', 'George Bush IntercontinentalAirport');

1 row created.
```

```
AI
--
AP_NAME
--------------------------------------------------------------------------------
9W
FrankfurtAirport

AA
Louisville InternationalAirport

Al
Chandigarh InternationalAirport


AI
--
AP_NAME
--------------------------------------------------------------------------------
BA
Chhatrapati Shivaji International Airport

EK
George Bush IntercontinentalAirport

LH
lndira Gandhilnternational Airport
```

5) FLIGHTS TABLE

```
SQL> CREATE TABLE FLIGHTS (
  2       FLIGHT_CODE VARCHAR(10) NOT NULL,
  3       SOURCE VARCHAR(3),
  4       DESTINATION VARCHAR(3),
  5       ARRIVAL VARCHAR2(10),
  6       DEPARTURE VARCHAR2(10),
  7       STATUS VARCHAR(10),
  8       DURATION VARCHAR2(30),
  9       FLIGHTTYPE VARCHAR(10),
 10       LAYOVER_TIME VARCHAR2(30),
 11       NO_OF_STOPS INT,
 12       AIRLINEID VARCHAR(3),
 13       PRIMARY KEY(FLIGHT_CODE),
 14       FOREIGN KEY(AIRLINEID) REFERENCES AIRLINE(AIRLINEID) ON
 DELETE CASCADE
 15   );

Table created.
```

```
SQL> INSERT INTO FLIGHTS(FLIGHT_CODE, SOURCE, DESTINATION, ARRIV
AL, DEPARTURE, STATUS, DURATION, FLIGHTTYPE, LAYOVER_TIME, NO_OF
_STOPS, AIRLINEID) VALUES('Al2014','BOM','DFW','02:10','03:15','
On-time','24hr','Connecting','N/A',3,'AA');

1 row created.

SQL> INSERT INTO FLIGHTS(FLIGHT_CODE, SOURCE, DESTINATION, ARRIV
AL, DEPARTURE, STATUS, DURATION, FLIGHTTYPE, LAYOVER_TIME, NO_OF
_STOPS, AIRLINEID) VALUES('QR2305','BOM','DFW','13:00','13:55','
Delayed','21hr','Non-stop','0',0,'QR');

1 row created.

SQL> INSERT INTO FLIGHTS(FLIGHT_CODE, SOURCE, DESTINATION, ARRIV
AL, DEPARTURE, STATUS, DURATION, FLIGHTTYPE, LAYOVER_TIME, NO_OF
_STOPS, AIRLINEID) VALUES('EY1234','JFK','TPA','19:20','20:05','
On-time','16hrs','Connecting','N/A',5,'EY');

1 row created.

SQL> INSERT INTO FLIGHTS(FLIGHT_CODE, SOURCE, DESTINATION, ARRIV
AL, DEPARTURE, STATUS, DURATION, FLIGHTTYPE, LAYOVER_TIME, NO_OF
_STOPS, AIRLINEID) VALUES('LH9876','JFK','BOM','05:50','06:35','
On-time','18hrs','Non-stop','0',0,'LH');

1 row created.
```

```
SQL> INSERT INTO FLIGHTS(FLIGHT_CODE, SOURCE, DESTINATION, ARRIV
AL, DEPARTURE, STATUS, DURATION, FLIGHTTYPE, LAYOVER_TIME, NO_OF
_STOPS, AIRLINEID) VALUES('BA1689','FRA','DEL','10:20','10:55','
On-time','14hrs','Non-stop','0',0,'BA');

1 row created.

SQL> INSERT INTO FLIGHTS(FLIGHT_CODE, SOURCE, DESTINATION, ARRIV
AL, DEPARTURE, STATUS, DURATION, FLIGHTTYPE, LAYOVER_TIME, NO_OF
_STOPS, AIRLINEID) VALUES('AA4367','SFO','FRA','18:10','18:55','
On-time','21hrs','Non-stop','0',0,'AA')
  2  ;

1 row created.

SQL> INSERT INTO FLIGHTS(FLIGHT_CODE, SOURCE, DESTINATION, ARRIV
AL, DEPARTURE, STATUS, DURATION, FLIGHTTYPE, LAYOVER_TIME, NO_OF
_STOPS, AIRLINEID) VALUES('QR1902','IXC','IAH','22:00','22:50','
Delayed','28hrs','Non-stop','5',1,'QR');

1 row created.

SQL> INSERT INTO FLIGHTS(FLIGHT_CODE, SOURCE, DESTINATION, ARRIV
AL, DEPARTURE, STATUS, DURATION, FLIGHTTYPE, LAYOVER_TIME, NO_OF
_STOPS, AIRLINEID) VALUES('BA3056','BOM','DFW','02:15','02:55','
On-time','29hrs','Connecting','N/A',3,'BA');

1 row created.
```

```
SQL> DURATION, FLIGHTTYPE, LAYOVER_TIME, NO_OF_STOPS, AIRLINEID)
 VALUES('EK3456','BOM','SFO','18:50','19:40','On-time','30hrs','
Non-stop','0',0,'EK');
SP2-0734: unknown command beginning "DURATION, ..." - rest of li
ne ignored.
SQL> INSERT INTO FLIGHTS(FLIGHT_CODE, SOURCE, DESTINATION, ARRIV
AL, DEPARTURE, STATUS, DURATION, FLIGHTTYPE, LAYOVER_TIME, NO_OF
_STOPS, AIRLINEID) VALUES('EK3456','BOM','SFO','18:50','19:40','
On-time','30hrs','Non-stop','0',0,'EK');

1 row created.

SQL> INSERT INTO FLIGHTS(FLIGHT_CODE, SOURCE, DESTINATION, ARRIV
AL, DEPARTURE, STATUS, DURATION, FLIGHTTYPE, LAYOVER_TIME, NO_OF
_STOPS, AIRLINEID) VALUES('9W2334','IAH','DEL','23:00','13:45','
On-time','23hrs','Direct','0',0,'9W');

1 row created.

SQL> select * from flights;
```

```
--------
NO_OF_STOPS AIR
----------- ---

EY1234     JFK TPA 19:20      20:05      On-time
16hrs                          Connecting N/A
         5 EY

LH9876     JFK BOM 05:50      06:35      On-time
18hrs                          Non-stop   0

FLIGHT_COD SOU DES ARRIVAL    DEPARTURE  STATUS
---------- --- --- ---------- ---------- ----------
DURATION                       FLIGHTTYPE LAYOVER_TIME
------------------------------ ---------- ----------------------
--------
NO_OF_STOPS AIR
----------- ---
          0 LH

BA1689     FRA DEL 10:20      10:55      On-time
14hrs                          Non-stop   0
         0 BA

AA4367     SFO FRA 18:10      18:55      On-time

FLIGHT_COD SOU DES ARRIVAL    DEPARTURE  STATUS
---------- --- --- ---------- ---------- ----------
DURATION                       FLIGHTTYPE LAYOVER_TIME
------------------------------ ---------- ----------------------
```

```
--------
NO_OF_STOPS AIR
---------- ---
21hrs                          Non-stop   0
         0 AA

QR1902     IXC IAH 22:00      22:50      Delayed
28hrs                          Non-stop   5
         1 QR


FLIGHT_COD SOU DES ARRIVAL    DEPARTURE  STATUS
---------- --- --- ---------- ---------- ----------
DURATION                       FLIGHTTYPE LAYOVER_TIME
------------------------------ ---------- ----------------------
--------
NO_OF_STOPS AIR
---------- ---
BA3056     BOM DFW 02:15      02:55      On-time
29hrs                          Connecting N/A
         3 BA

EK3456     BOM SFO 18:50      19:40      On-time
30hrs                          Non-stop   0
         0 EK

FLIGHT_COD SOU DES ARRIVAL    DEPARTURE  STATUS
---------- --- --- ---------- ---------- ----------
DURATION                       FLIGHTTYPE LAYOVER_TIME
------------------------------ ---------- ----------------------
```

6) PASSENGER TABLE

```
SQL> CREATE TABLE PASSENGER (
  2      PID INT NOT NULL,
  3      PASSPORT_NO VARCHAR(20) NOT NULL,
  4      FNAME VARCHAR(50),
  5      ADDRESS VARCHAR(100),
  6      PHONE VARCHAR(15),
  7      AGE INT,
  8      SSN INT,
  9      AP_NAME VARCHAR(100),
 10      PRIMARY KEY (PID, PASSPORT_NO),
 11      FOREIGN KEY (SSN) REFERENCES EMPLOYEE(SSN),
 12      FOREIGN KEY (AP_NAME) REFERENCES AIRPORT(AP_NAME)
 13  );

Table created.
```

```
SQL> INSERT INTO PASSENGER (PID, PASSPORT_NO, FNAME, ADDRESS, PHONE, AGE, SSN, AP_NAME)
  2  VALUES (1, 'R1234567', 'John Doe', '123 Main St, Anytown', '555-1234', 30, 111111111, 'Louisville InternationalAirport');

1 row created.

SQL> INSERT INTO PASSENGER (PID, PASSPORT_NO, FNAME, ADDRESS, PHONE, AGE, SSN, AP_NAME)
  2  VALUES (2, 'R2345678', 'Jane Smith', '456 Elm St, Othertown', '555-5678', 25, 987654321, 'Chhatrapati Shivaji International Airp
ort');

1 row created.

SQL> INSERT INTO PASSENGER (PID, PASSPORT_NO, FNAME, ADDRESS, PHONE, AGE, SSN, AP_NAME)
  2  VALUES (3, 'R3456789', 'Alice Johnson', '789 Oak St, Anytown', '555-1111', 28, 222222222, 'Louisville InternationalAirport');

1 row created.

SQL> INSERT INTO PASSENGER (PID, PASSPORT_NO, FNAME, ADDRESS, PHONE, AGE, SSN, AP_NAME)
  2  VALUES (4, 'R4567890', 'Bob Smith', '456 Pine St, Othertown', '555-2222', 35, 333333333, 'Chhatrapati Shivaji International Airp
ort');

1 row created.

SQL> INSERT INTO PASSENGER (PID, PASSPORT_NO, FNAME, ADDRESS, PHONE, AGE, SSN, AP_NAME)
  2  VALUES (5, 'R5678901', 'Charlie Brown', '123 Maple St, Anothertown', '555-3333', 40, 444444444, 'Louisville InternationalAirport
');

1 row created.
```

```
       PID PASSPORT_NO
---------- --------------------
FNAME
--------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
PHONE                AGE        SSN
---------------- ---------- ----------
AP_NAME
--------------------------------------------------------------------------------
         1 R1234567
John Doe
123 Main St, Anytown

       PID PASSPORT_NO
---------- --------------------
FNAME
--------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
PHONE                AGE        SSN
---------------- ---------- ----------
AP_NAME
--------------------------------------------------------------------------------
555-1234              30  111111111
Louisville InternationalAirport
```

```
       PID PASSPORT_NO
---------- --------------------
FNAME
--------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
PHONE                AGE        SSN
---------------- ---------- ----------
AP_NAME
--------------------------------------------------------------------------------
         2 R2345678
Jane Smith
456 Elm St, Othertown

       PID PASSPORT_NO
---------- --------------------
FNAME
--------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
PHONE                AGE        SSN
---------------- ---------- ----------
AP_NAME
--------------------------------------------------------------------------------
555-5678              25  987654321
Chhatrapati Shivaji International Airport
```

```
       PID PASSPORT_NO
---------- --------------------
FNAME
--------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
PHONE                  AGE        SSN
---------------- ---------- ----------
AP_NAME
--------------------------------------------------------------------------------
         3 R3456789
Alice Johnson
789 Oak St, Anytown

       PID PASSPORT_NO
---------- --------------------
FNAME
--------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
PHONE                  AGE        SSN
---------------- ---------- ----------
AP_NAME
--------------------------------------------------------------------------------
555-1111                28  222222222
Louisville InternationalAirport
```

```
       PID PASSPORT_NO
---------- --------------------
FNAME
--------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
PHONE                  AGE        SSN
---------------- ---------- ----------
AP_NAME
--------------------------------------------------------------------------------
         4 R4567890
Bob Smith
456 Pine St, Othertown

       PID PASSPORT_NO
---------- --------------------
FNAME
--------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
PHONE                  AGE        SSN
---------------- ---------- ----------
AP_NAME
--------------------------------------------------------------------------------
555-2222                35  333333333
Chhatrapati Shivaji International Airport
```

```
       PID PASSPORT_NO
---------- --------------------
FNAME
--------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
PHONE                  AGE        SSN
---------------- ---------- ----------
AP_NAME
--------------------------------------------------------------------------------
         5 R5678901
Charlie Brown
123 Maple St, Anothertown

       PID PASSPORT_NO
---------- --------------------
FNAME
--------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
PHONE                  AGE        SSN
---------------- ---------- ----------
AP_NAME
--------------------------------------------------------------------------------
555-3333                40  444444444
Louisville InternationalAirport
```

```
       PID PASSPORT_NO
---------- --------------------
FNAME
--------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
PHONE                AGE        SSN
---------------- ---------- ----------
AP_NAME
--------------------------------------------------------------------------------
         6 R6789012
David Lee
789 Elm St, Othercity

       PID PASSPORT_NO
---------- --------------------
FNAME
--------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
PHONE                AGE        SSN
---------------- ---------- ----------
AP_NAME
--------------------------------------------------------------------------------
555-4444                 45  555555555
Chhatrapati Shivaji International Airport
```

7) EMPLOYEE

```
SQL> CREATE TABLE EMPLOYEE (
  2      SSN INT PRIMARY KEY,
  3      NAME VARCHAR(50),
  4      ADDRESS VARCHAR(100),
  5      PHONE VARCHAR(20),
  6      AGE INT,
  7      JOB_TYPE VARCHAR(50),
  8      SHIFT VARCHAR(20),
  9      AP_NAME VARCHAR(50),
 10      FOREIGN KEY (AP_NAME) REFERENCES AIRPORT(AP_NAME)
 11  );

Table created.
```

```
SQL> INSERT INTO EMPLOYEE (SSN, NAME, ADDRESS, PHONE, AGE, JOB_TYPE, SHIFT, AP_NAME)
  2  VALUES (123456789, 'John Doe', '123 Main St, Anytown', '555-1234', 30, 'Manager', 'Day', 'Louisville InternationalAirport');

1 row created.

SQL> INSERT INTO EMPLOYEE (SSN, NAME, ADDRESS, PHONE, AGE, JOB_TYPE, SHIFT, AP_NAME)
  2  VALUES (987654321, 'Jane Smith', '456 Elm St, Othertown', '555-5678', 25, 'Assistant', 'Night', 'Chhatrapati Shivaji Internation
al Airport');

1 row created.

SQL> INSERT INTO EMPLOYEE (SSN, NAME, ADDRESS, PHONE, AGE, JOB_TYPE, SHIFT, AP_NAME)
  2  VALUES (111111111, 'Alice Johnson', '789 Oak St, Anytown', '555-1111', 28, 'Receptionist', 'Day', 'Louisville InternationalAirpo
rt');

1 row created.

SQL> INSERT INTO EMPLOYEE (SSN, NAME, ADDRESS, PHONE, AGE, JOB_TYPE, SHIFT, AP_NAME)
  2  VALUES (222222222, 'Bob Smith', '456 Pine St, Othertown', '555-2222', 35, 'Security', 'Night', 'Chhatrapati Shivaji Internationa
l Airport');

1 row created.

SQL> INSERT INTO EMPLOYEE (SSN, NAME, ADDRESS, PHONE, AGE, JOB_TYPE, SHIFT, AP_NAME)
  2  VALUES (333333333, 'Charlie Brown', '123 Maple St, Anothertown', '555-3333', 40, 'Janitor', 'Day', 'Louisville InternationalAirp
ort');

1 row created.
```

```
SQL> INSERT INTO EMPLOYEE (SSN, NAME, ADDRESS, PHONE, AGE, JOB_TYPE, SHIFT, AP_NAME)
  2  VALUES (444444444, 'David Lee', '789 Elm St, Othercity', '555-4444', 45, 'Pilot', 'Night', 'Chhatrapati Shivaji International Ai
rport');

1 row created.

SQL> INSERT INTO EMPLOYEE (SSN, NAME, ADDRESS, PHONE, AGE, JOB_TYPE, SHIFT, AP_NAME)
  2  VALUES (555555555, 'Emma Garcia', '456 Walnut St, Yetanothertown', '555-5555', 32, 'Flight Attendant', 'Day', 'Louisville Intern
ationalAirport');

1 row created.
```

```
SQL> INSERT INTO EMPLOYEE (SSN, NAME, ADDRESS, PHONE, AGE, JOB_TYPE, SHIFT, AP_NAME)
  2  VALUES (444444444, 'David Lee', '789 Elm St, Othercity', '555-4444', 45, 'Pilot', 'Night', 'Chhatrapati Shivaji International Ai
rport');

1 row created.

SQL> INSERT INTO EMPLOYEE (SSN, NAME, ADDRESS, PHONE, AGE, JOB_TYPE, SHIFT, AP_NAME)
  2  VALUES (555555555, 'Emma Garcia', '456 Walnut St, Yetanothertown', '555-5555', 32, 'Flight Attendant', 'Day', 'Louisville Intern
ationalAirport');

1 row created.
```

8) SERVES TABLE

```
SQL> CREATE TABLE SERVES (
  2      SSN INT NOT NULL,
  3      PID INT NOT NULL,
  4      PASSPORT_NO VARCHAR(10) NOT NULL,
  5      PRIMARY KEY(SSN, PID, PASSPORT_NO),
  6      FOREIGN KEY(SSN) REFERENCES EMPLOYEE(SSN),
  7      FOREIGN KEY(PID, PASSPORT_NO) REFERENCES PASSENGER(PID, PASSPORT_NO)
  8  );

Table created.
```

```
SQL> INSERT INTO SERVES (SSN, PID, PASSPORT_NO) VALUES (123456789, 1, 'R1234567');

1 row created.

SQL> INSERT INTO SERVES (SSN, PID, PASSPORT_NO) VALUES (987654321, 2, 'R2345678');

1 row created.

SQL> INSERT INTO SERVES (SSN, PID, PASSPORT_NO) VALUES (111111111, 3, 'R3456789');

1 row created.

SQL> INSERT INTO SERVES (SSN, PID, PASSPORT_NO) VALUES (222222222, 4, 'R4567890');

1 row created.

SQL> INSERT INTO SERVES (SSN, PID, PASSPORT_NO) VALUES (333333333, 5, 'R5678901');

1 row created.

SQL> INSERT INTO SERVES (SSN, PID, PASSPORT_NO) VALUES (444444444, 6, 'R6789012');

1 row created.
```

```
SQL> select * from serves;

       SSN        PID PASSPORT_N
---------- ---------- ----------
 111111111          3 R3456789
 123456789          1 R1234567
 222222222          4 R4567890
 333333333          5 R5678901
 444444444          6 R6789012
 987654321          2 R2345678

6 rows selected.

SQL>
```

9) TICKET TABLE

```
SQL> CREATE TABLE TICKET (
  2      TICKETNO INT PRIMARY KEY,
  3      PID INT,
  4      PASSPORT_NO VARCHAR(20),
  5      SEAT_NO VARCHAR(10),
  6      CLASS VARCHAR(10),
  7      PRICE DECIMAL(10, 2),
  8      DATE_OF_BOOKING DATE,
  9      FOREIGN KEY (PID, PASSPORT_NO) REFERENCES PASSENGER (PID, PASSPORT_NO)
 10  );

Table created.
```

```
SQL> INSERT INTO TICKETS (TICKETNO, PID, PASSPORT_NO, SEAT_NO, CLASS, PRICE, DATE_OF_BOOKING) VALUES (1, 1, 'R1234567', 'A1', 'Econom
y', 500.00, TO_DATE('2024-05-10', 'YYYY-MM-DD'));

1 row created.

SQL>   2
SQL> INSERT INTO TICKETS (TICKETNO, PID, PASSPORT_NO, SEAT_NO, CLASS, PRICE, DATE_OF_BOOKING) VALUES (2, 2, 'R2345678', 'B2', 'Busine
ss', 1200.00, TO_DATE('2024-05-11', 'YYYY-MM-DD'));

1 row created.

SQL> INSERT INTO TICKETS (TICKETNO, PID, PASSPORT_NO, SEAT_NO, CLASS, PRICE, DATE_OF_BOOKING) VALUES (3, 3, 'R3456789', 'C3', 'First
Class', 2000.00, TO_DATE('2024-05-12', 'YYYY-MM-DD'));
INSERT INTO TICKETS (TICKETNO, PID, PASSPORT_NO, SEAT_NO, CLASS, PRICE, DATE_OF_BOOKING) VALUES (3, 3, 'R3456789', 'C3', 'First Class
', 2000.00, TO_DATE('2024-05-12', 'YYYY-MM-DD'))
                                                                                                                             *

ERROR at line 1:
ORA-12899: value too large for column "COE203201"."TICKETS"."CLASS" (actual:
11, maximum: 10)


SQL> INSERT INTO TICKETS (TICKETNO, PID, PASSPORT_NO, SEAT_NO, CLASS, PRICE, DATE_OF_BOOKING) VALUES (4, 4, 'R4567890', 'D4', 'Econom
y', 550.00, TO_DATE('2024-05-13', 'YYYY-MM-DD'));

1 row created.

SQL> INSERT INTO TICKETS (TICKETNO, PID, PASSPORT_NO, SEAT_NO, CLASS, PRICE, DATE_OF_BOOKING) VALUES (5, 5, 'R5678901', 'E5', 'Busine
ss', 1250.00, TO_DATE('2024-05-14', 'YYYY-MM-DD'));

1 row created.
```

```
SQL> INSERT INTO TICKETS (TICKETNO, PID, PASSPORT_NO, SEAT_NO, CLASS, PRICE, DATE_OF_BOOKING) VALUES (8, 2, 'R2345678', 'H8', 'Busine
ss', 1220.00, TO_DATE('2024-05-17', 'YYYY-MM-DD'));

1 row created.

SQL> INSERT INTO TICKETS (TICKETNO, PID, PASSPORT_NO, SEAT_NO, CLASS, PRICE, DATE_OF_BOOKING) VALUES (9, 3, 'R3456789', 'I9', 'First
Class', 2050.00, TO_DATE('2024-05-18', 'YYYY-MM-DD'));
INSERT INTO TICKETS (TICKETNO, PID, PASSPORT_NO, SEAT_NO, CLASS, PRICE, DATE_OF_BOOKING) VALUES (9, 3, 'R3456789', 'I9', 'First Class
', 2050.00, TO_DATE('2024-05-18', 'YYYY-MM-DD'))
                                                                                                                             *

ERROR at line 1:
ORA-12899: value too large for column "COE203201"."TICKETS"."CLASS" (actual:
11, maximum: 10)


SQL> INSERT INTO TICKETS (TICKETNO, PID, PASSPORT_NO, SEAT_NO, CLASS, PRICE, DATE_OF_BOOKING) VALUES (10, 4, 'R4567890', 'J10', 'Econ
omy', 560.00, TO_DATE('2024-05-19', 'YYYY-MM-DD'));

1 row created.
```

```
SQL> select * from tickets;

  TICKETNO        PID PASSPORT_NO          SEAT_NO    CLASS          PRICE
---------- ---------- -------------------- ---------- ---------- ----------
DATE_OF_B
---------
         1          1 R1234567             A1         Economy           500
10-MAY-24

         2          2 R2345678             B2         Business         1200
11-MAY-24

         4          4 R4567890             D4         Economy           550
13-MAY-24


  TICKETNO        PID PASSPORT_NO          SEAT_NO    CLASS          PRICE
---------- ---------- -------------------- ---------- ---------- ----------
DATE_OF_B
---------
         5          5 R5678901             E5         Business         1250
14-MAY-24

         7          1 R1234567             G7         Economy           520
16-MAY-24

         8          2 R2345678             H8         Business         1220
17-MAY-24
```

```
  TICKETNO        PID PASSPORT_NO          SEAT_NO    CLASS          PRICE
---------- ---------- -------------------- ---------- ---------- ----------
DATE_OF_B
---------
        10          4 R4567890             J10        Economy           560
19-MAY-24
```

STORED PRODECURE:

Procedure to insert a new passenger.

Procedure to Insert a New Passenger (insert_passenger):
This procedure allows inserting a new passenger into the passenger table.
It takes parameters such as PID (Passenger ID), Passport Number, Full Name, Address, Phone Number, Age, and Airport Name.
Inside the procedure, an INSERT statement is used to add the new passenger details to the passenger table.
Finally, a COMMIT statement is used to save the changes to the database.

```
SQL> CREATE OR REPLACE PROCEDURE insert_passenger (
  2      p_pid IN INT,
  3      p_passport_no IN VARCHAR2,
  4      p_fname IN VARCHAR2,
  5      p_address IN VARCHAR2,
  6      p_phone IN VARCHAR2,
  7      p_age IN INT,
  8      p_ap_name IN VARCHAR2
  9  ) AS
 10  BEGIN
 11      INSERT INTO passenger (PID, PASSPORT_NO, FNAME, ADDRESS, PHONE, AGE, AP_NAME)
 12      VALUES (p_pid, p_passport_no, p_fname, p_address, p_phone, p_age, p_ap_name);
 13      COMMIT;
 14  END insert_passenger;
 15  /

Procedure created.
```

Output:

```
SQL> -- Insert a new passenger
SQL> EXECUTE insert_passenger(7, 'R7890123', 'Michael Smith', '789 Oak St, Anytown', '555-7777', 35, 'Louisville InternationalAirport
');

PL/SQL procedure successfully completed.

SQL>
SQL> -- Retrieve the newly inserted passenger
SQL> SELECT * FROM passenger WHERE PID = 7;
```

```
       PID PASSPORT_NO
---------- --------------------
FNAME
--------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
PHONE                AGE        SSN
---------------- ---------- ----------
AP_NAME
--------------------------------------------------------------------------------
         7 R7890123
Michael Smith
789 Oak St, Anytown

       PID PASSPORT_NO
---------- --------------------
FNAME
--------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
PHONE                AGE        SSN
---------------- ---------- ----------
AP_NAME
--------------------------------------------------------------------------------
555-7777                 35
Louisville InternationalAirport
```

Procedure to update passenger details.

Procedure to Update Passenger Details (update_passenger_details):

This procedure updates the phone number and address of an existing passenger based on their PID.

It takes the PID, new phone number, and new address as parameters.

Inside the procedure, an UPDATE statement is used to modify the phone number and address of the passenger in the passenger table.

Again, a COMMIT statement is used to save the changes to the database.

```
SQL> CREATE OR REPLACE PROCEDURE update_passenger_details (
  2      p_pid IN INT,
  3      p_phone IN VARCHAR2,
  4      p_address IN VARCHAR2
  5  ) AS
  6  BEGIN
  7      UPDATE passenger
  8      SET PHONE = p_phone,
  9          ADDRESS = p_address
 10      WHERE PID = p_pid;
 11      COMMIT;
 12  END update_passenger_details;
 13  /

Procedure created.
```

Output:

```
SQL> -- Update passenger details
SQL> EXECUTE update_passenger_details(5, '555-5555', '123 Maple St, Anothertown');

PL/SQL procedure successfully completed.

SQL>
SQL> -- Retrieve the updated passenger details
SQL> SELECT * FROM passenger WHERE PID = 5;
```

```
       PID PASSPORT_NO
---------- --------------------
FNAME
--------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
PHONE                  AGE        SSN
---------------- ---------- ----------
AP_NAME
--------------------------------------------------------------------------------
         5 R5678901
Charlie Brown
123 Maple St, Anothertown

       PID PASSPORT_NO
---------- --------------------
FNAME
--------------------------------------------------
ADDRESS
--------------------------------------------------------------------------------
PHONE                  AGE        SSN
---------------- ---------- ----------
AP_NAME
--------------------------------------------------------------------------------
555-5555                40  444444444
Louisville InternationalAirport
```

Procedure to delete a ticket.

Procedure to Delete a Ticket (delete_ticket):

This procedure allows deleting a ticket from the tickets table based on the ticket number (TICKETNO).

It takes the ticket number (TICKETNO) as a parameter.

Inside the procedure, a DELETE statement is used to remove the ticket from the tickets table.

Finally, a COMMIT statement is used to save the changes to the database.

```
SQL> CREATE OR REPLACE PROCEDURE delete_ticket (
  2      p_ticketno IN INT
  3  ) AS
  4  BEGIN
  5      DELETE FROM tickets
  6      WHERE TICKETNO = p_ticketno;
  7      COMMIT;
  8  END delete_ticket;
  9  /

Procedure created.
```

Output:

```
SQL> -- Delete a ticket
SQL> EXECUTE delete_ticket(4);

PL/SQL procedure successfully completed.

SQL>
SQL> -- Confirm the ticket deletion
SQL> SELECT * FROM tickets WHERE TICKETNO = 4;

no rows selected
```

FUNCTIONS:

1)Function to Calculate Total Price for a Flight:

Function to Calculate Total Price for a Flight (calculate_total_price):
This function calculates the total price of all tickets for a given flight.
It takes the flight code (FLIGHT_COD) as input.
Inside the function, a SELECT statement is used to sum up the prices of all tickets associated with the provided flight code.
The total price is returned as the output of the function.

```
SQL> CREATE OR REPLACE FUNCTION calculate_total_price (
  2      p_flight_cod IN VARCHAR2
  3  ) RETURN DECIMAL AS
  4      v_total_price DECIMAL(10, 2);
  5  BEGIN
  6      SELECT SUM(PRICE) INTO v_total_price
  7      FROM tickets
  8      WHERE FLIGHT_COD = p_flight_cod;
  9
 10      RETURN v_total_price;
 11  END calculate_total_price;
 12  /

Warning: Function created with compilation errors.
```

2)Function to Get Employee Details by SSN:

Function to Get Employee Details by SSN (get_employee_details):

This function retrieves details of an employee from the employee table based on their Social Security Number (SSN).

It takes the SSN as input.

Inside the function, a SELECT statement is used to fetch the employee details matching the provided SSN.

The employee details are returned as the output of the function.

```
SQL> CREATE OR REPLACE FUNCTION get_employee_details (
  2      p_ssn IN VARCHAR2
  3  ) RETURN employee%ROWTYPE AS
  4      v_employee employee%ROWTYPE;
  5  BEGIN
  6      SELECT * INTO v_employee
  7      FROM employee
  8      WHERE SSN = p_ssn;
  9
 10      RETURN v_employee;
 11  END get_employee_details;
 12  /

Function created.
```

3)Function to Get Airport Name by City Name:

Function to Get Airport Name by City Name (get_airport_name):

This function retrieves the name of the airport associated with a given city name.

It takes the city name (CNAME) as input.

Inside the function, a SELECT statement is used to fetch the airport name (AP_NAME) from the airport table based on the provided city name.

The airport name is returned as the output of the function.

```
SQL> CREATE OR REPLACE FUNCTION get_airport_name (
  2      p_city_name IN VARCHAR2
  3  ) RETURN VARCHAR2 AS
  4      v_ap_name VARCHAR2(100);
  5  BEGIN
  6      SELECT AP_NAME INTO v_ap_name
  7      FROM airport
  8      WHERE CNAME = p_city_name;
  9
 10      RETURN v_ap_name;
 11  END get_airport_name;
 12  /

Function created.
```

TRIGGERS:

1)Trigger to Update Flight Status:

Trigger to Update Flight Status (update_flight_status):

This trigger fires before updating a row in the flights table.

It checks if the arrival time of the flight is earlier than the previous arrival time. If so, it sets the flight status to 'Delayed'; otherwise, it sets it to 'On-time'.

```
SQL> CREATE OR REPLACE TRIGGER update_flight_status
  2  BEFORE UPDATE ON flights
  3  FOR EACH ROW
  4  BEGIN
  5      IF :new.ARRIVAL < :old.ARRIVAL THEN
  6          :new.STATUS := 'Delayed';
  7      ELSE
  8          :new.STATUS := 'On-time';
  9      END IF;
 10  END update_flight_status;
 11  /

Trigger created.
```

2)Trigger to Enforce Passenger Age Constraint:

Trigger to Enforce Passenger Age Constraint (check_passenger_age):

This trigger fires before inserting or updating a row in the passenger table.

It checks if the age of the passenger exceeds a maximum allowed age (set to 120). If the age exceeds this limit, it raises an error, preventing the insertion or update operation

```
SQL> CREATE OR REPLACE TRIGGER check_passenger_age
  2  BEFORE INSERT OR UPDATE ON passenger
  3  FOR EACH ROW
  4  DECLARE
  5      v_max_age CONSTANT INT := 120;
  6  BEGIN
  7      IF :new.AGE > v_max_age THEN
  8          RAISE_APPLICATION_ERROR(-20001, 'Passenger age exceeds maximum allowed age.');
  9      END IF;
 10  END check_passenger_age;
 11  /

Trigger created.
```

3)Trigger to Log Ticket Deletion:

Trigger to Log Ticket Deletion (log_ticket_deletion):

This trigger fires after deleting a row from the tickets table.

It inserts a record into the deleted_tickets table, containing the ticket number (TICKETNO) and the deletion date (DELETION_DATE). This log helps in tracking deleted tickets for auditing purposes.

```
SQL> CREATE OR REPLACE TRIGGER log_ticket_deletion
  2  AFTER DELETE ON tickets
  3  FOR EACH ROW
  4  BEGIN
  5      INSERT INTO deleted_tickets (TICKETNO, DELETION_DATE)
  6      VALUES (:old.TICKETNO, SYSDATE);
  7      COMMIT;
  8  END log_ticket_deletion;
  9  /

Warning: Trigger created with compilation errors.
```

CURSORS IN OUR TABLE:

1)Fetching data about tickets and their prices:

DECLARE: This keyword marks the beginning of the PL/SQL block. It's used to declare variables, cursors, and other program objects.

Cursor Declaration: A cursor named ticket_prices_cur is declared. It selects data from the tickets table.

BEGIN: Marks the beginning of the executable part of the block.

OPEN: This statement opens the cursor ticket_prices_cur, making it ready to fetch rows from the result set.

LOOP: This begins a loop that iterates through the rows returned by the cursor.

FETCH: This statement retrieves the next row from the result set into the variables specified (v_ticket_no, v_pid, etc.).

EXIT WHEN: This condition checks if there are no more rows to fetch from the cursor's result set. If no more rows are found, the loop is exited.

DBMS_OUTPUT.PUT_LINE: Inside the loop, this statement prints information about each fetched row. In this case, it prints the ticket number, price, and class.

CLOSE: After all rows have been processed, the cursor is closed to release associated resources.

END: Marks the end of the PL/SQL block.

The cursor is essentially a pointer to the result set returned by the SQL query. It allows you to fetch rows one by one and process them within the PL/SQL block. This example demonstrates how to use a cursor to iterate through the result set and perform actions on each row, in this case, printing information about tickets.

```
SQL> DECLARE
  2      v_ticket_no NUMBER;
  3      v_pid VARCHAR2(20);
  4      v_passport_no VARCHAR2(20);
  5      v_seat_no VARCHAR2(10);
  6      v_class VARCHAR2(20);
  7      v_price NUMBER;
  8      v_date_of_booking DATE;
  9      CURSOR ticket_prices_cur IS
 10          SELECT TICKETNO, PID, PASSPORT_NO, SEAT_NO, CLASS, PRICE, DATE_OF_BOOKING
 11          FROM tickets;
 12  BEGIN
 13      OPEN ticket_prices_cur;
 14      LOOP
 15          FETCH ticket_prices_cur INTO v_ticket_no, v_pid, v_passport_no, v_seat_no, v_class, v_price, v_date_of_book
ing;
 16          EXIT WHEN ticket_prices_cur%NOTFOUND;
 17          -- Process the data here, for example, print or manipulate it
 18          DBMS_OUTPUT.PUT_LINE('Ticket No: ' || v_ticket_no || ', Price: ' || v_price || ', Class: ' || v_class);
 19      END LOOP;
 20      CLOSE ticket_prices_cur;
 21  END;
 22  /

PL/SQL procedure successfully completed.
```

2) Fetching employees and their job details:

```
SQL> DECLARE
  2      v_ssn employee.SSN%TYPE;
  3      v_name employee.NAME%TYPE;
  4      v_address employee.ADDRESS%TYPE;
  5      v_phone employee.PHONE%TYPE;
  6      v_age employee.AGE%TYPE;
  7      v_job_type employee.JOB_TYPE%TYPE;
  8      v_shift employee.SHIFT%TYPE;
  9      v_ap_name employee.AP_NAME%TYPE;
 10      CURSOR employee_details_cur IS
 11          SELECT SSN, NAME, ADDRESS, PHONE, AGE, JOB_TYPE, SHIFT, AP_NAME
 12          FROM employee;
 13  BEGIN
 14      OPEN employee_details_cur;
 15      LOOP
 16          FETCH employee_details_cur INTO v_ssn, v_name, v_address, v_phone, v_age, v_job_type, v_shift, v_ap_name;
 17          EXIT WHEN employee_details_cur%NOTFOUND;
 18          -- Process the data here, for example, print or manipulate it
 19          DBMS_OUTPUT.PUT_LINE('Employee: ' || v_name || ', Job Type: ' || v_job_type || ', Shift: ' || v_shift);
 20      END LOOP;
 21      CLOSE employee_details_cur;
 22  END;
 23  /
```

SOME SQL QUERIES:

1) Retrieve the total number of tickets sold for each class.

```
SQL> SELECT CLASS, COUNT(*) AS TOTAL_TICKETS_SOLD
  2  FROM tickets
  3  GROUP BY CLASS;

CLASS      TOTAL_TICKETS_SOLD
---------- ------------------
Economy                     3
Business                    3
```

2) Retrieve the number of passengers served by each employee.

```
SQL> SELECT e.NAME, COUNT(s.PID) AS PASSENGER_COUNT
  2  FROM employee e
  3  LEFT JOIN serves s ON e.SSN = s.SSN
  4  GROUP BY e.NAME;

NAME                                               PASSENGER_COUNT
-------------------------------------------------- ---------------
Alice Johnson                                                    1
Bob Smith                                                        1
David Lee                                                        1
Jane Smith                                                       1
Charlie Brown                                                    1
John Doe                                                         1
Emma Garcia                                                      0

7 rows selected.
```

3) Retrieve the average age of passengers traveling in each class.

```
SQL> SELECT t.CLASS, AVG(p.AGE) AS AVERAGE_AGE
  2  FROM tickets t
  3  JOIN passenger p ON t.PID = p.PID
  4  GROUP BY t.CLASS;

CLASS      AVERAGE_AGE
---------- -----------
Economy     31.6666667
Business            30
```

4) Retrieve the names of airlines operating in cities with airports in India.

```
SQL> SELECT DISTINCT a.AL_NAME
  2  FROM airline a
  3  JOIN contains AI ON a.AL_NAME = AI.AIRLINEID
  4  JOIN airport ap ON AI.AP_NAME = ap.AP_NAME
  5  JOIN city c ON ap.CNAME = c.CNAME
  6  WHERE c.COUNTRY = 'India';

no rows selected
```

CONCLUSION

The Airport Management System project outlined comprehensive requirements and specifications aimed at facilitating efficient management of airports, airlines, and passengers. Through meticulous planning and analysis, we have devised a system that addresses key operational factors influencing airport management, such as flight scheduling, booking management, and passenger services.

By capturing data on commercial service airports, airlines, flights, and passengers, the system provides a holistic view of airport operations, enhancing decision-making and operational efficiency. The integration of airline codes, flight details, and passenger information ensures accurate tracking and management of flight services, while also facilitating seamless booking and ticketing processes.

Moreover, the inclusion of employee management features acknowledges the vital role played by airport staff in ensuring smooth operations and passenger satisfaction. From administrative support to traffic control, the system caters to diverse roles within the airport ecosystem, streamlining workflows and enhancing productivity.

LEARNING OUTCOMES:

1. Database Design: Gain proficiency in designing a relational database schema that accurately models the entities, relationships, and attributes relevant to airport management, airlines, flights, passengers, and employees.

2. SQL Skills: Develop competence in writing SQL queries to perform various operations on the database, including data retrieval, insertion, deletion, and modification.

3. Normalization Techniques: Understand and apply normalization rules to ensure that the database schema is structured efficiently, minimizing redundancy and dependency issues while maintaining data integrity.

4. Entity-Relationship Modeling: Learn to create entity-relationship diagrams (ERDs) to visually represent the relationships between different entities in the database and translate them into a normalized relational schema.

5. System Integration: Explore the integration of different components within the airport management system, such as airlines, flights, passengers, and employees, to ensure seamless data flow and functionality across the entire system.

6. Real-world Application: Gain insight into real-world scenarios and requirements of airport management systems, including handling flight schedules, bookings, passenger information, and employee management, and apply database concepts to address these challenges effectively.

7. Problem-solving Skills: Develop problem-solving skills by analyzing complex requirements and translating them into actionable database designs, queries, and functionalities that meet the needs of users and stakeholders.

8. Team Collaboration: Enhance collaboration and teamwork skills by working together with team members to brainstorm ideas, divide tasks, share responsibilities, and coordinate efforts to successfully complete the project within the specified timeline.