Khulna University of Engineering and Technology

Project Name : Airlines Database System

Name : Tasfia Zaman Samiha

Roll: 2007078

Submission Date: 07/05/2024

Introduction

The Airlines Database System is a vital component of modern aviation management, enabling airlines to efficiently store, retrieve, and analyze data related to various operational aspects. From flight scheduling to passenger information management and ticketing, this system plays a crucial role in optimizing processes and enhancing overall efficiency in the dynamic airline industry.

This report explores the functionalities and significance of the Airlines Database System in streamlining operations and improving decision-making within the aviation sector. By leveraging relational databases, airlines can effectively manage resources, drive strategic initiatives, and deliver a seamless travel experience for passengers and stakeholders.

Objective

The objectives of the Airlines Database System are to-

- Enhance operational efficiency through streamlined data management and process automation.
- Optimise resource allocation by tracking and analysing flight schedules, passenger loads, and crew assignments.
- Improve passenger services by maintaining accurate passenger records and facilitating seamless booking and check-in processes.
- Ensure safety and compliance by monitoring aircraft maintenance schedules and regulatory requirements.
- Support data-driven decision-making by providing comprehensive insights into flight performance, customer preferences, and market trends.

Implement SQL queries

To achieve this project's objectives, it's essential to deploy a diverse array of SQL queries for robust data management. This aspect is vital for:

• Inserting and modifying records, including student enrollments, course particulars, and exam outcomes.
• Rapidly and effectively accessing particular data to fulfil academic and administrative needs.
• Guaranteeing the database's currency and accuracy, reflecting the most recent academic progress and outcomes.

Database Table

```
create table passengers(
passenger_id number(20),
name varchar2(30),
email varchar2(30),
contact_number number(20),
primary key(passenger_id)
);
```

```
 create table airport(
 aiport_id number(20),
 airport_name varchar2(30),
 city varchar2(30),
 country varchar2(30),
 primary key(aiport_id)
 );
```

```
CREATE TABLE Flights (
    flight_id NUMBER(20),
    departure_date_time DATE,
    arrival_date_time DATE,
    origin_airport_id NUMBER(20),
    PRIMARY KEY (flight_id),
    FOREIGN KEY (origin_airport_id) REFERENCES Airport(aiport_id)
);
```

```
CREATE TABLE Tickets (
   ticket_id NUMBER(20),
   flight_id NUMBER(20),
   passenger_id NUMBER(20),
   seat_number VARCHAR2(10),
   ticket_price NUMBER(10, 2),
   PRIMARY KEY (ticket_id),
   FOREIGN KEY (flight_id) REFERENCES Flights(flight_id),
   FOREIGN KEY (passenger_id) REFERENCES Passengers(passenger_id)
);
```

Detailed explanation of each table

1. Passengers Table:

- This table serves as a repository for passenger information.
- passenger_id: A unique identifier for each passenger, defined as a number with a maximum length of 20 digits.
- name: Stores the name of the passenger, limited to 30 characters.
- email: Stores the email address of the passenger, with a maximum length of 30 characters.
- contact_number: Stores the phone number of the passenger, represented as a numeric value with a maximum length of 20 digits.
- PRIMARY KEY (passenger_id): Defines the `passenger_id` as the primary key, ensuring each record is uniquely identifiable.

2. Airport Table:

- This table holds information about different airports.
- airport_id: Unique identifier for each airport, defined as a number with a maximum length of 20 digits.
- airport_name: Stores the name of the airport, limited to 30 characters.
- city: Stores the city where the airport is located, limited to 30 characters.
- country: Stores the country where the airport is located, limited to 30 characters.

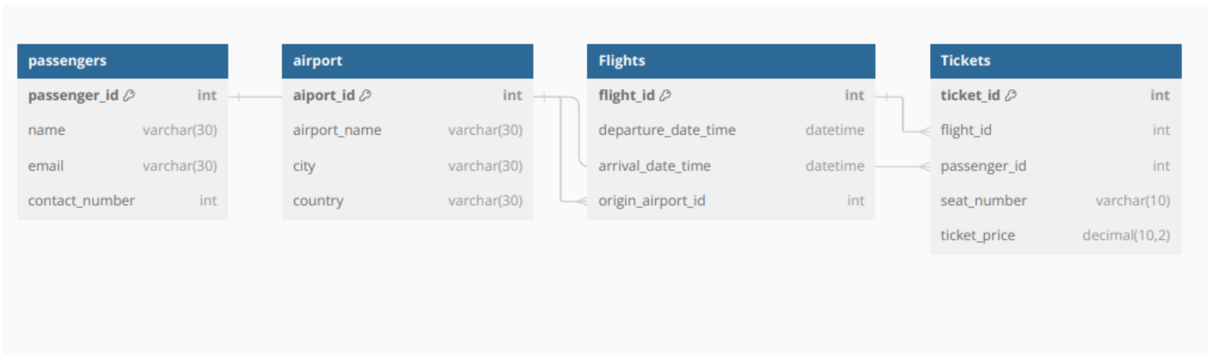
- PRIMARY KEY (airport_id): Defines the `airport_id` as the primary key for unique identification of each airport.

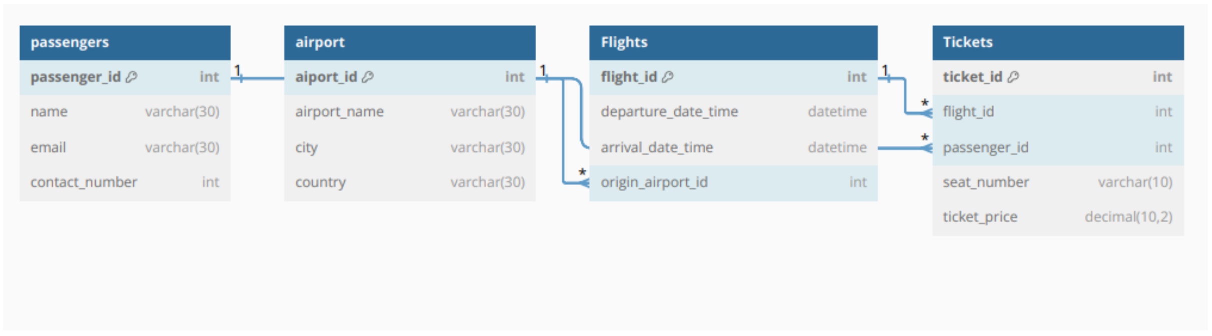
3. Flights Table:

- This table contains details about different flights.
- flight_id: Unique identifier for each flight, defined as a number with a maximum length of 20 digits.
- departure_date_time: Represents the date and time of departure for the flight.
- arrival_date_time: Represents the date and time of arrival for the flight.
- origin_airport_id: Foreign key referencing the `airport_id` in the Airport table, indicating the origin airport for the flight.
- PRIMARY KEY (flight_id): Defines the `flight_id` as the primary key for unique identification of each flight.
- FOREIGN KEY (origin_airport_id) REFERENCES Airport(airport_id): Establishes a relationship with the Airport table, ensuring that the `origin_airport_id` references a valid `airport_id` in the Airport table.

4. Tickets Table:

- This table stores information about tickets issued for flights.
- ticket_id: Unique identifier for each ticket, defined as a number with a maximum length of 20 digits.
- flight_id: Foreign key referencing the `flight_id` in the Flights table, indicating the flight associated with the ticket.
- passenger_id: Foreign key referencing the `passenger_id` in the Passengers table, indicating the passenger associated with the ticket.
- seat_number: Stores the seat number assigned to the passenger for the flight, limited to 10 characters.
- ticket_price: Represents the price of the ticket, defined as a numeric value with a precision of 10 digits and a scale of 2.
- PRIMARY KEY (ticket_id): Defines the `ticket_id` as the primary key for unique identification of each ticket.
- FOREIGN KEY (flight_id) REFERENCES Flights(flight_id): Establishes a relationship with the Flights table, ensuring that the `flight_id` references a valid `flight_id` in the Flights table.
- FOREIGN KEY (passenger_id) REFERENCES Passengers(passenger_id): Establishes a relationship with the Passengers table, ensuring that the `passenger_id` references a valid `passenger_id` in the Passengers table.





Queries on common operation

The database also supports various standard operations to manage airlines data efficiently

Add

INSERT INTO Passengers (passenger_id, name, email, phone_number) VALUES (3, 'Michael Johnson', 'michael@example.com', 5551234567);
INSERT INTO Airport (airport_id, airport_name, city, country) VALUES (2, 'Heathrow Airport', 'London', 'UK');
INSERT INTO Flights (flight_id, departure_date_time, arrival_date_time, origin_airport_id) VALUES (3, '2024-05-03 10:00:00', '2024-05-03 14:00:00', 3);
INSERT INTO Tickets (ticket_id, flight_id, passenger_id, seat_number, ticket_price) VALUES (3, 3, 3, 'C3', 300.00);

Update

UPDATE Passengers
SET passport_number = 'PASS123'
WHERE passenger_id = 3;

ALTER TABLE Passengers
RENAME COLUMN phone_number TO contact_number;

Delete

```
DELETE FROM Airport
WHERE airport_id IN (8, 9);
```

PL/SQL

```
DECLARE
   v_passenger_id Passengers.passenger_id%TYPE;
   v_name Passengers.name%TYPE;
   v_email Passengers.email%TYPE;
   v_phone_number Passengers.contact_number%TYPE;
BEGIN
   -- Initialise cursor to fetch passenger data
   FOR passenger_rec IN (SELECT * FROM Passengers) LOOP
      -- Assign values from cursor to variables
      v_passenger_id := passenger_rec.passenger_id;
      v_name := passenger_rec.name;
      v_email := passenger_rec.email;
      v_phone_number := passenger_rec.contact_number;

      -- Perform some operation with the fetched data
      DBMS_OUTPUT.PUT_LINE('Passenger ID: ' || v_passenger_id);
      DBMS_OUTPUT.PUT_LINE('Name: ' || v_name);
      DBMS_OUTPUT.PUT_LINE('Email: ' || v_email);
      DBMS_OUTPUT.PUT_LINE('Phone Number: ' || v_phone_number);

      -- Here you can add any additional logic or operations

   END LOOP;
END;
/
```

Retrieve Information

```
SELECT *
FROM Flights
WHERE origin_airport_id NOT IN (
    SELECT airport_id
    FROM Airport
    WHERE city = 'New York'
);
```

Conclusion

In conclusion, the database schema designed for the airline system encompasses four key tables: Passengers, Airport, Flights, and Tickets. These tables collectively facilitate the storage and management of passenger information, airport details, flight data, and ticketing records. By organising data into these structured tables and establishing relationships between them using primary and foreign keys, the database ensures data integrity and enables efficient retrieval and manipulation of information. Overall, this database schema forms a robust foundation for supporting various operations and functionalities within the airline system, enhancing operational efficiency and customer service.