**ASSIGNMENT - 1**

**DATABASE MANAGEMENT SYSTEM**

**SIDDARTH.P**

**192324264**

**Flight Booking and Passenger Management System**

The **Flight Booking and Passenger Management System** database is designed to handle flight bookings, manage passenger details, track seat availability, and process ticketing. It includes features for booking tickets, processing cancellations, and generating reports for analysis. Automation is incorporated through triggers, and processes are optimized with stored procedures.

**1. Database Tables Design**

**Table: Passengers**

Stores passenger information for identification and contact purposes.

| **Column** | **Data Type** | **Description** |
| --- | --- | --- |
| passenger\_id | INT (PK) | Unique identifier for each passenger |
| first\_name | VARCHAR(50) | Passenger's first name |
| last\_name | VARCHAR(50) | Passenger's last name |
| phone\_number | VARCHAR(15) | Contact number |
| email | VARCHAR(100) | Email address |
| passport\_number | VARCHAR(20) | Passport number (if applicable) |

**Table: Flights**

Tracks flight details, including schedule and capacity.

| **Column** | **Data Type** | **Description** |
| --- | --- | --- |
| flight\_id | INT (PK) | Unique identifier for each flight |
| flight\_number | VARCHAR(20) | Flight number |
| departure\_city | VARCHAR(50) | City of departure |
| arrival\_city | VARCHAR(50) | City of arrival |
| departure\_time | DATETIME | Scheduled departure time |
| arrival\_time | DATETIME | Scheduled arrival time |
| total\_seats | INT | Total number of seats available |
| available\_seats | INT | Current number of available seats |

**Table: Bookings**

Records flight bookings, linking passengers to specific flights and seats.

| **Column** | **Data Type** | **Description** |
| --- | --- | --- |
| booking\_id | INT (PK) | Unique identifier for each booking |
| passenger\_id | INT (FK) | References Passengers.passenger\_id |
| flight\_id | INT (FK) | References Flights.flight\_id |
| seat\_number | VARCHAR(10) | Assigned seat number |
| booking\_date | DATE | Date of booking |
| class | VARCHAR(20) | Travel class (e.g., Economy, Business) |

**Table: Seats**

Tracks individual seat availability per flight.

| **Column** | **Data Type** | **Description** |
| --- | --- | --- |
| seat\_id | INT (PK) | Unique identifier for the seat |
| flight\_id | INT (FK) | References Flights.flight\_id |
| seat\_number | VARCHAR(10) | Seat number |
| is\_available | BOOLEAN | Availability status |

**Table: Payments**

Manages payment records for bookings.

| **Column** | **Data Type** | **Description** |
| --- | --- | --- |
| payment\_id | INT (PK) | Unique identifier for each payment |
| booking\_id | INT (FK) | References Bookings.booking\_id |
| payment\_date | DATE | Date of payment |
| amount | DECIMAL(10,2) | Total amount paid |
| payment\_status | VARCHAR(20) | Status of payment (e.g., Paid, Pending) |

**2. Constraints for Referential Integrity**

* **Foreign Keys:**
  + passenger\_id in Bookings references Passengers(passenger\_id)
  + flight\_id in Bookings references Flights(flight\_id)
  + flight\_id in Seats references Flights(flight\_id)
  + booking\_id in Payments references Bookings(booking\_id)
* **Primary Keys:**
  + Each table has a primary key to uniquely identify records.
* **Check Constraints:**
  + Validate non-negative seat counts and payment amounts.

**3. Stored Procedures**

**a. Check Seat Availability**

Confirms if seats are available on a specific flight for the desired class.

CREATE PROCEDURE CheckSeatAvailability(IN flightId INT)

BEGIN

SELECT available\_seats

FROM Flights

WHERE flight\_id = flightId;

END;

**b. Book Ticket**

Registers a booking, assigns a seat, and updates seat availability.

CREATE PROCEDURE BookTicket(IN passengerId INT, IN flightId INT, IN seatNumber VARCHAR(10), IN class VARCHAR(20))

BEGIN

INSERT INTO Bookings (passenger\_id, flight\_id, seat\_number, booking\_date, class)

VALUES (passengerId, flightId, seatNumber, CURDATE(), class);

UPDATE Flights

SET available\_seats = available\_seats - 1

WHERE flight\_id = flightId;

UPDATE Seats

SET is\_available = FALSE

WHERE flight\_id = flightId AND seat\_number = seatNumber;

END;

**c. Cancel Booking**

Cancels a booking and restores seat availability.

CREATE PROCEDURE CancelBooking(IN bookingId INT)

BEGIN

DECLARE flightId INT;

DECLARE seatNumber VARCHAR(10);

SELECT flight\_id, seat\_number INTO flightId, seatNumber

FROM Bookings

WHERE booking\_id = bookingId;

DELETE FROM Bookings WHERE booking\_id = bookingId;

UPDATE Flights

SET available\_seats = available\_seats + 1

WHERE flight\_id = flightId;

UPDATE Seats

SET is\_available = TRUE

WHERE flight\_id = flightId AND seat\_number = seatNumber;

END;

**4. Triggers**

**a. Update Seat Availability on Booking**

Ensures real-time updates of seat availability.

CREATE TRIGGER AfterBookingInsert

AFTER INSERT ON Bookings

FOR EACH ROW

BEGIN

UPDATE Flights

SET available\_seats = available\_seats - 1

WHERE flight\_id = NEW.flight\_id;

END;

**b. Update Seat Availability on Cancellation**

CREATE TRIGGER AfterBookingDelete

AFTER DELETE ON Bookings

FOR EACH ROW

BEGIN

UPDATE Flights

SET available\_seats = available\_seats + 1

WHERE flight\_id = OLD.flight\_id;

END;

**5. SQL Queries for Reports**

**a. Flight Occupancy Report**

Generates a report of flight occupancy percentages.

SELECT flight\_id,

(total\_seats - available\_seats) / total\_seats \* 100 AS occupancy\_percentage

FROM Flights;

**b. Popular Routes Report**

Identifies the most popular flight routes.

SELECT departure\_city, arrival\_city, COUNT(booking\_id) AS bookings

FROM Flights f

JOIN Bookings b ON f.flight\_id = b.flight\_id

GROUP BY departure\_city, arrival\_city

ORDER BY bookings DESC;

**c. Revenue by Flight Class**

Summarizes revenue by class type for all flights.

SELECT class, SUM(amount) AS total\_revenue

FROM Bookings b

JOIN Payments p ON b.booking\_id = p.booking\_id

GROUP BY class

ORDER BY total\_revenue DESC;