DB design - Airline Management System

Nikhil Reddy Karukonda NUID: 002774917

Summary: Why did you create this database?

The objective of creating this database is to design a comprehensive Airline Management System that effectively handles all aspects of airline operations. This system includes **flight scheduling**, **ticket reservations**, **cancellations**, **payment**, **and notifications**, enabling airlines to optimize their processes, minimize expenses, and make decisions based on data. By adopting this database, airlines can boost their performance efficiency and elevate customer contentment while addressing the needs of various system users such as **customers**, **passengers**, **airline staff**, **and system administrators**. The database design encompasses crucial entities like **Airport**, **Airline**, **Flight**, **Payment**, **Notification**, **Customer**, **Passenger**, **Seat**, **and more**, ensuring a solid and well-organized system that tackles all essential requirements and use cases.

List of use cases:

This database serves numerous use cases for an airline management system, including:

- **a.** Storing and maintaining data for all airlines, enabling easy access to airline information.
- **b.** Allowing customers to search for flights based on the desired date and source/destination airport criteria, ensuring a seamless booking experience.
- **c.** Recording customers' personal information, facilitating ticket reservations, cancellations, and flight status checks for any scheduled flight.
- **d.** Enabling customers to create itineraries that consist of multiple flights, offering greater flexibility in travel planning.
- **e.** Allowing customers to make single reservations for multiple passengers under one itinerary, streamlining the reservation process.
- **f.** Providing customers with the option to cancel their itineraries and reservations, ensuring a hassle-free experience.
- **g.** Assigning crew members to flights effectively, optimizing staff allocation for each flight instance.
- **h.** Facilitating payment processing for reservations and sending notifications to customers regarding their reservations and flight status updates, enhancing communication and transparency.

The database can be further extended to include additional functionalities such as baggage tracking, in-flight meals, phone number format validation, and user password encryption, further enhancing the capabilities of the Airline Management System.

System Users: Customers, Passengers, Airline Staff (Pilot, Attendant), and System Admin.

PK airport code (PK) PK flight number (PK) airport_name airport_address_id (FK) PK itinerary id (FK) PK flight instance id (FK) schedul Cabin Crev PK <u>crew member id (PK)</u> flight instance id (PK) Bridge Table PK aircraft code (PK) email address_id (FK) PK itinerary id PK airline code (PK) Custome PK payment id (PK) PK reservation id (PK) PK customer id (PK) flight_instance_id (FK) first_name last_name phone_number email address_id(FK) Gender DOB PK notification id (PK) account id (PK) PK passenger id (PK) first_name last_name phone_numb Gender DOB

ER Diagrams - final version [after normalization]

Stored Procedures:

sp_search_flights: This stored procedure allows users to search for available flights based on departure date, departure airport code, and arrival airport code. It returns a list of active or scheduled flights that match the search criteria.

sp_get_customer_reservations: This stored procedure retrieves all reservations made by a particular customer along with the passengers associated with each reservation.

sp_assign_crew_to_flight: This stored procedure checks if a crew member is available for a given flight instance before assigning them to the flight. If the crew member is not available, an error message is returned.

sp_cancel_reservation: This stored procedure cancels a reservation if it is not already canceled or checked-in. If the payment status is 'COMPLETED', it also updates the payment status to 'REFUNDED'. If the reservation is already canceled or checked-in, an error message is returned.

Triggers:

payment_completed_reservation_confirmed: This trigger is executed after an update on the Payment table. If the updated payment status is 'COMPLETED', it checks if there's a reservation with the corresponding reservation_id and the status 'CONFIRMED'. If such a reservation exists, it inserts a new entry into the Notification table with the current date, the notification type 'email', the appropriate content, and the reservation id.

Views:

FlightStatusView: This view retrieves the flight status information for all flight instances, along with the associated flight details. Users can query this view to get the flight status for a specific flight instance.