Air Cargo Analysis – SQL Project Report

Objective

This SQL project aims to assist Air Cargo, an aviation service provider, in analyzing customer behavior, route usage, ticket sales, and operational performance. The insights generated from this analysis will help the company:

- Recognize regular customers and provide loyalty benefits.
- Identify high-traffic routes for better resource allocation.
- Analyze ticket sales and revenues for profitability.
- Improve booking systems and customer service using database features like stored procedures, functions, and indexing.

Dataset and Tables Used

The project uses the following tables:

- customer: Contains customer personal information.
- routes: Contains flight route details.
- passengers_on_flights: Logs each passenger's travel details.
- ticket_details: Stores ticket booking, pricing, and brand information.

Project Execution and Rationale

1. Database Setup

We began by selecting the working database using USE flight;. This ensures all operations affect the correct schema.

2. Data Validation Constraints

We used ALTER TABLE commands to enforce:

- **Check Constraint**: Ensured flight_num > 0 and distance_miles > 0, which are logical real-world validations.
- **Unique Constraint**: Applied on route_id, flight_num to ensure no duplicate route-flight combination exists.These constraints ensure **data integrity and consistency**.

3. Filtering Passengers by Route Range

Used a SELECT query with BETWEEN to retrieve customers who traveled on routes 1 to 25. This helps identify early or popular routes.

4. Business Class Revenue and Passenger Count

Used aggregation (COUNT, SUM) with a WHERE clause to identify **revenue generated** and the number of **Business Class passengers**, supporting pricing analysis.

5. Customer Full Names

Used CONCAT(first_name, last_name) to generate full names for display or reporting purposes.

6. Ticket Booking Insights

Created a new table customer_ticket_details using JOIN to show customers who have booked tickets, along with booking dates, aircraft, and class. This combines key data into a **single consolidated view**.

7. Brand-based Bookings

Created customer_booking table to show customers who flew with specific airline brands like **Emirates**. Useful for loyalty or brand preference tracking.

8. Economy Plus Flyers

Grouped passengers who traveled in "Economy Plus" using GROUP BY and HAVING. Helps in determining **class popularity** and **targeted marketing**.

9. Revenue Threshold Check

Created a derived revenue table with grouped data and calculated revenue using (COUNT(no_of_tickets) * price_per_ticket). A conditional IF clause was used to check if the revenue crossed ₹10,000. This is a form of **business rule validation**.

10. User Privilege Management

Created a new user 'dev_user' and granted limited permissions (SELECT, INSERT, UPDATE) on company_db. This demonstrates **role-based access control** for secure database interaction.

11. Maximum Ticket Price by Class

Grouped by class_id and used MAX() to identify the most expensive ticket sold in each class. This supports **price analysis and marketing strategy**.

12. Indexing for Query Optimization

Created an index on route_id in the passengers_on_flights table to optimize searches for specific route queries. This **improves performance and speed**.

13. Execution Plan Analysis

Used EXPLAIN to review how MySQL executes the route query, confirming the use of indexing. It aids in **query performance diagnostics**.

14. Revenue Breakdown Using ROLLUP

Used GROUP BY ROLLUP(customer_id, aircraft_id) to generate subtotals and grand totals of ticket prices. A powerful tool for **financial reporting**.

15. Business Class View

Filtered records from passengers_on_flights joined with customer to list only **Business Class flyers**, helping in premium service planning and VIP treatment.

16. Stored Procedure for Dynamic Route Search

Created GetPassengersByRouteRange stored procedure with **dynamic input parameters** and error handling for missing tables. This promotes **modular code** for repetitive queries.

17. Procedure for Distance-Based Route Search

Built travel_dist procedure to find routes longer than a specified distance (e.g., 2000 miles). Helps in **fleet planning for long-haul flights**.

18. Function for Distance Classification

Designed a function dist_type() to label each route as **Short**, **Intermediate**, or **Long Distance**. This enables easy categorization for analytics and reporting.

19. Complimentary Service Function

Created comp_serve() function to determine if complimentary services are provided based on class. Business and Economy Plus receive **"available"**, others **"not available"**.

20. Cursor to Retrieve Specific Customers

Used a **cursor** in a stored procedure to fetch the **first customer whose last name ends with 'Scott'**. It also handles cases where no match is found. Demonstrates use of **cursors and conditional logic** in MySQL.

Conclusion

This project demonstrates the application of advanced SQL techniques to solve real-world business problems in the airline industry. Through joins, grouping, indexing, constraints, views, procedures, and functions, we built a data-driven backend that supports decision-making for:

- Customer loyalty programs
- Fleet and route management
- Revenue monitoring
- Operational efficiency

Each step was executed with the goal of improving performance, data integrity, and usability for the Air Cargo company.

Would you like this write-up compiled into a **PDF file for submission** along with the commented SQL code?