

Structured Query Language SQL

Under Guidance

Mr.Veera

Presented By

Sonia A

ABSTRACT

This project aims to develop a robust database management and analysis system for Air Cargo, an aviation company seeking to optimize operations and enhance customer experiences. Leveraging SQL, the project focuses on generating detailed reports to identify and reward regular customers, allocate resources effectively on busy routes, and optimize ticket sales strategies. By utilizing various datasets and performing operations such as creating tables, writing complex queries, and implementing stored procedures, the project aims to improve Air Cargo's operability and maintain its competitive edge in the aviation industry. The insights generated will aid in route optimization, customer satisfaction enhancement, and overall operational efficiency, reinforcing Air Cargo's position as a customer-centric choice for air travel.

INTRODUCTION

In the aviation industry, efficient data management and analysis are critical for maintaining a competitive edge and ensuring customer satisfaction. Air Cargo, an aviation company providing air transportation services for both passengers and freight, recognizes the importance of leveraging data to optimize its operations and enhance customer experiences. Operating through partnerships and alliances with other airlines, Air Cargo is committed to improving the ease of travel and booking for its customers. The company's primary goal is to generate detailed reports on various aspects of its operations, including regular passengers, busiest routes, ticket sales, and other relevant metrics. By analyzing this data, Air Cargo aims to identify and reward its regular customers, allocate resources more effectively on busy routes, and optimize its ticket sales strategies.

As a Database Administrator (DBA) expert, the project's objective is to create a robust data management framework that supports these analyses. The project involves using several datasets containing information about customers, flight details, ticket purchases, and route specifics. The operations to be performed include creating and managing tables, writing complex queries for data retrieval and analysis, optimizing query performance, and implementing stored procedures for specific tasks. By focusing on these objectives, the project aims to enhance Air Cargo's operability, ensuring it remains a customer-centric and favorable choice for air travel.

SQL

Structured Query Language(SQL), is a standardized programming language specifically designed for managing and manipulating relational databases.



Query

A query is a request for data or information from a database. It is a way to interact with the database to retrieve, manipulate, and manage the data stored within it. Queries are used to perform various operations such as selecting, inserting, updating, and deleting data.

How is SQL used in Air Cargo Analysis

- SQL plays a pivotal role in Air Cargo Analysis by enabling the efficient management, manipulation, and analysis of vast amounts of data related to passengers, flights, tickets, and routes by creating tables.
- Once the tables are created, SQL is used to populate them with relevant dataset information, ensuring that all necessary data is accurately recorded.
- SQL queries are crafted to extract actionable insights, such as identifying regular customers who frequently travel and may benefit from loyalty offers, analyzing the busiest routes to optimize aircraft deployment, and determining overall ticket sales to monitor financial performance.

- SQL is also crucial for performance optimization, where indexing and execution plan analysis help in speeding up query performance, ensuring quick and efficient data retrieval.
- SQL is used for user management, creating and granting access to new users, ensuring secure and controlled access to the database.
- SQL's robust capabilities in data handling, analysis, automation, and optimization make it indispensable for Air Cargo Analysis, driving operational efficiency and informed decision-making.

Objective

As a Database Administrator (DBA) expert, your role is to design and implement a robust database system that facilitates detailed analysis of Air Cargo's operations. This includes:

- Identifying regular customers to offer personalized incentives.
- Analyzing the busiest routes to determine the necessary number of aircraft.
- Compiling detailed ticket sales reports to improve business strategies.

Dataset Description

The analysis will be based on several datasets, each containing crucial information:

1. **Customer:** Information about customers (ID, first name, last name, date of birth, gender).
2. **Passengers on Flights:** Details of passengers' travel (aircraft ID, route ID, customer ID, departure, arrival, seat number, class ID, travel date, flight number).
3. **Ticket Details:** Information about ticket purchases (purchase date, customer ID, aircraft ID, class ID, number of tickets, airport code, price per ticket, brand).
4. **Routes:** Route details (route ID, flight number, origin airport, destination airport, aircraft ID, distance in miles).

Operations to be Performed

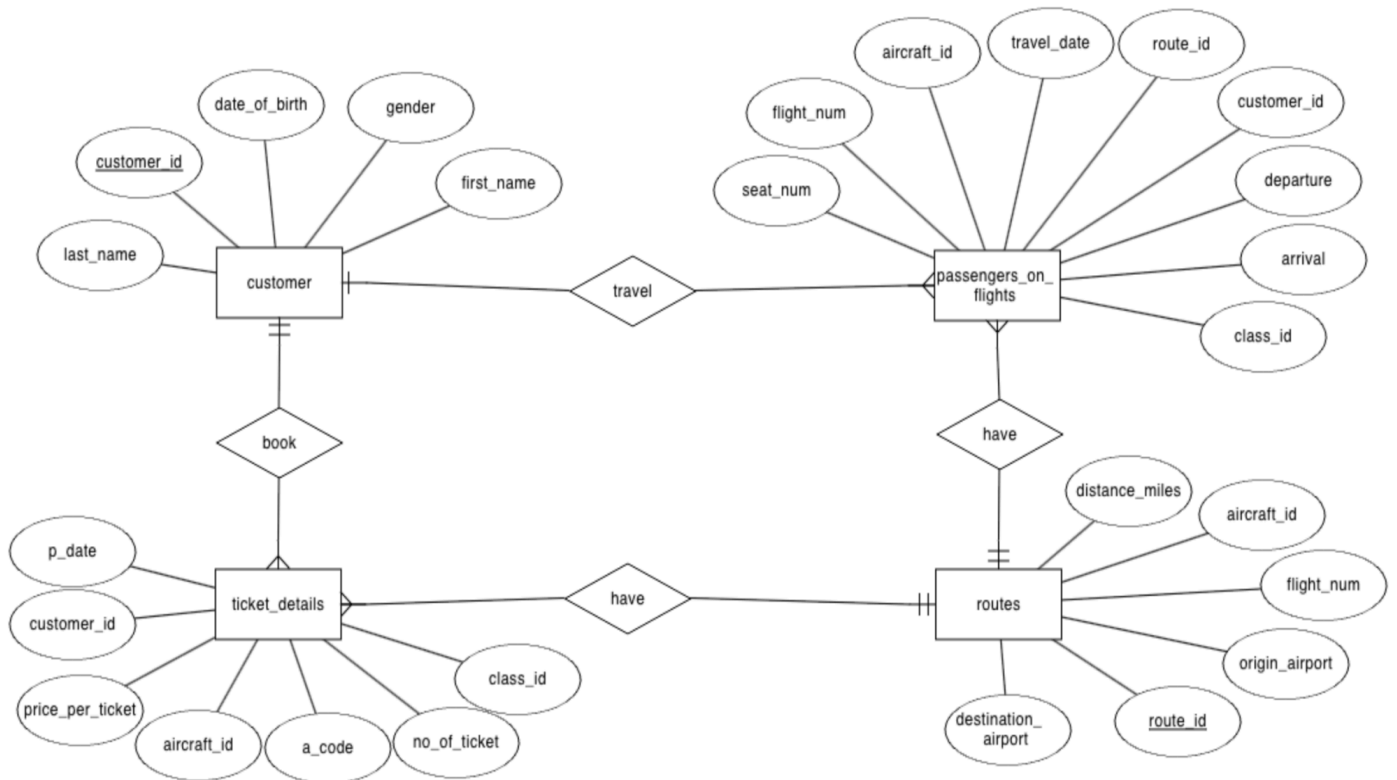
The following operations will be performed to achieve the project objectives:

1. **Create Route Details Table:** Define a table for route details with appropriate constraints.
2. **Passenger Information Retrieval:** Query to display passengers traveling on specific routes.
3. **Business Class Revenue Analysis:** Identify the number of business class passengers and total revenue.
4. **Customer Full Name Display:** Extract and display full names of customers.
5. **Customer and Ticket Information Extraction:** Identify customers who registered and booked tickets.
6. **Emirates Customer Identification:** Retrieve details for customers flying with Emirates.
7. **Economy Plus Passengers Analysis:** Identify Economy Plus passengers using grouping and conditions.
8. **Revenue Threshold Check:** Determine if revenue exceeds a specified threshold.
9. **User Access Management:** Create and grant access to a new database user.
10. **Maximum Ticket Price Calculation:** Calculate the maximum ticket price per class using window functions.
11. **Performance Optimization:** Improve query performance for specific routes.
12. **Execution Plan Viewing:** View the execution plan for specific route queries.
13. **Total Ticket Price Calculation:** Calculate total ticket prices using rollup functions.
14. **Business Class View Creation:** Create a view for business class customers with airline brands.
15. **Passenger Details Stored Procedure:** Develop a procedure to get passenger details for a range of routes.
16. **Route Distance Analysis Procedure:** Create a procedure to extract details of long-distance routes.

By executing these operations, Air Cargo will be able to generate valuable insights to enhance its services and operational efficiency, positioning itself as a leading and customer-friendly aviation company.

ER diagram

The following table is the database design that covers the core functionality of aircargo analysis



CUSTOMER

	customer_id	first_name	last_name	date_of_birth	gender
▶	1	Julie	Sam	12-01-1989	F
	2	Steve	Ryan	03-04-1983	M
	3	Morris	Lois	09-12-1993	M
	4	Cathenna	Emily	14-09-1977	F
	5	Aaron	Kim	18-02-1991	M
	6	Alexander	Scot	12-02-1985	M
	7	Anderson	Stewart	11-01-1992	M
	8	Floyd	Ted	21-02-1993	M
	9	Leo	Travis	22-03-1994	M
	10	Melvin	Tracy	23-04-1995	M
	11	Roger	Walson	24-05-1996	M
	12	Shirley	Wally	25-06-1997	F
	13	Solomon	Walter	26-07-1998	M
	14	Carol	Vernon	27-08-1999	F
	15	Linda	William	28-09-1986	F
	16	Chirstine	Willis	06-10-1987	F
	17	Catherine	Shad	09-11-1988	F
	18	Gloria	Richie	04-12-1989	F
	19	Joyce	Paul	02-06-1990	F
	20	Sara	Oliver	01-01-1991	F

customer 4 ×

Passengers_on_flights

	customer_id	aircraft_id	route_id	depart	arrival	seat_num	class_id	travel_date	flight_num
▶	2	A321	34	CRW	COD	01B	Bussiness	26-01-2019	1117
	2	767-301ER	4	JFK	LAX	01E	Economy	02-09-2018	1114
	1	ERJ142	9	DEN	LAX	01EP	Economy Plus	26-12-2019	1119
	1	CRJ900	30	BUR	STT	01FC	First Class	04-11-2018	1140
	5	767-301ER	12	ABI	ADK	02B	Bussiness	02-07-2018	1122
	5	ERJ142	18	ANI	BGR	02E	Economy	06-05-2020	1128
	8	A321	38	CST	DAL	02EP	Economy Plus	09-08-2020	1148
	4	767-301ER	5	LAX	JFX	02FC	First Class	06-04-2020	1115
	7	767-301ER	20	AVL	BOI	03B	Bussiness	08-07-2020	1130
	5	ERJ142	22	BGR	BJI	03E	Economy	31-05-2020	1132
	11	ERJ142	31	BTM	CHA	03EP	Economy Plus	02-08-2018	1141
	4	767-301ER	4	JFK	LAX	03FC	First Class	30-04-2020	1114
	11	767-301ER	5	LAX	JFX	04B	Bussiness	12-11-2020	1115
	8	A321	43	CBM	BOI	04E	Economy	02-05-2018	1153
	17	A321	13	ABI	ADK	04EP	Economy Plus	03-06-2019	1123
	9	767-301ER	15	CAK	ANI	04FC	First Class	10-09-2020	1125
	11	767-301ER	4	JFK	LAX	05B	Bussiness	09-11-2020	1114
	10	A321	10	HNL	DEN	05E	Economy	11-10-2020	1120
	19	CRJ900	47	DAL	LAX	05EP	Economy Plus	13-01-2021	1157
	9	CRJ900	33	CDC	CST	05FC	First Class	01-02-2018	1143

passengers_on_flights 6 ×

Routes

	route_id	flight_num	origin_airport	destination_airport	aircraft_id	distance_miles
▶	1	1111	EWR	HNL	767-301ER	4962
	2	1112	HNL	EWR	767-301ER	4962
	3	1113	EWR	LHR	A321	3466
	4	1114	JFK	LAX	767-301ER	2475
	5	1115	LAX	JFK	767-301ER	2475
	6	1116	HNL	LAX	767-301ER	2556
	7	1117	LAX	ORD	A321	1745
	8	1118	ORD	EWR	A321	719
	9	1119	DEN	LAX	ERJ142	862
	10	1120	HNL	DEN	A321	3365
	12	1122	ABI	ADK	767-301ER	4300
	13	1123	ADK	BQN	A321	2232
	14	1124	BQN	CAK	A321	2445
	15	1125	CAK	ANI	767-301ER	2000
	16	1126	ALB	APN	A321	1700
	17	1127	APN	BLV	767-301ER	1900
	18	1128	ANI	BGR	ERJ142	2450
	19	1129	ATW	AVL	A321	2222
	20	1130	AVL	BOI	767-301ER	3134
	21	1131	BFL	BET	A321	2425

routes 7 ×

Ticket_details

p_date	customer_id	aircraft_id	class_id	no_of_tickets	a_code	Price_per_ticket	brand
26-12-2018	27	767-301ER	767-301ER	1	DAL	130	Emirates
02-02-2020	22	ERJ142	Economy Plus	1	AGB	220	Jet Airways
03-03-2020	21	CRJ900	Bussiness	1	BOH	490	Bristish Airways
04-04-2020	4	767-301ER	First Class	1	AGB	390	Emirates
05-05-2020	5	ERJ142	Economy	1	CTM	120	Jet Airways
07-07-2020	7	767-301ER	Bussiness	1	BFS	430	Emirates
08-08-2020	8	A321	Economy Plus	1	DAL	275	Qatar Airways
09-09-2020	9	767-301ER	First Class	1	BOH	380	Emirates
10-10-2020	10	A321	Economy	1	MCO	135	Qatar Airways
11-11-2020	11	767-301ER	Bussiness	1	AGB	465	Emirates
12-12-2020	19	CRJ900	Economy Plus	1	DEN	225	Bristish Airways
01-01-2019	13	A321	First Class	1	YVR	395	Qatar Airways
02-02-2019	14	ERJ142	Economy	1	CTM	120	Jet Airways
03-03-2019	25	767-301ER	Bussiness	1	BHX	499	Emirates
04-04-2019	16	CRJ900	First Class	1	YVR	395	Bristish Airways
03-05-2019	17	A321	Economy Plus	1	BFS	250	Qatar Airways
06-06-2019	18	767-301ER	Economy	1	YVR	190	Emirates
07-07-2019	24	A321	Bussiness	1	CTM	480	Qatar Airways
09-08-2019	20	CRJ900	First Class	1	MCO	365	Bristish Airways
21-09-2019	25	767-301ER	Economy	1	BOH	150	Emirates

ticket_details 8 ×

1. Write a query to create a route_details table using suitable data types for the fields, such as route_id, flight_num, origin_airport, destination_airport, aircraft_id, and distance_miles. Implement the check constraint for the flight number and unique constraint for the route_id fields. Also, make sure that the distance miles field is greater than 0.

QUERY:

```
CREATE TABLE routes_details (  
    route_id int,  
    flight_num int,  
    origin_airport varchar(30),  
    destination_airport varchar(70),  
    aircraft_id varchar(70),  
    distance_miles int, PRIMARY KEY (route_id),  
    check (flight_num >= 1111),  
    check(distance_miles > 0 ));
```

OUTPUT:

	route_id	flight_num	origin_airport	destination_airport	aircraft_id	distance_miles
*	NULL	NULL	NULL	NULL	NULL	NULL

routes_details9 x

2. Write a query to display all the passengers (customers) who have traveled in routes 01 to 25. Take data from the passengers_on_flights table.

QUERY:

```
select * from passengers_on_flights WHERE route_id between 1 AND 25;
```

OUTPUT:

	customer_id	aircraft_id	route_id	depart	arrival	seat_num	class_id	travel_date	flight_num
▶	2	767-301ER	4	JFK	LAX	01E	Economy	02-09-2018	1114
	1	ERJ142	9	DEN	LAX	01EP	Economy Plus	26-12-2019	1119
	5	767-301ER	12	ABI	ADK	02B	Bussiness	02-07-2018	1122
	5	ERJ142	18	ANI	BGR	02E	Economy	06-05-2020	1128
	4	767-301ER	5	LAX	JFX	02FC	First Class	06-04-2020	1115
	7	767-301ER	20	AVL	BOI	03B	Bussiness	08-07-2020	1130
	5	ERJ142	22	BGR	BJI	03E	Economy	31-05-2020	1132
	4	767-301ER	4	JFK	LAX	03FC	First Class	30-04-2020	1114
	11	767-301ER	5	LAX	JFX	04B	Bussiness	12-11-2020	1115
	17	A321	13	ABI	ADK	04EP	Economy Plus	03-06-2019	1123
	9	767-301ER	15	CAK	ANI	04FC	First Class	10-09-2020	1125
	11	767-301ER	4	JFK	LAX	05B	Bussiness	09-11-2020	1114
	10	A321	10	HNL	DEN	05E	Economy	11-10-2020	1120
	15	A321	14	BQN	CAK	06B	Bussiness	02-11-2018	1124
	13	A321	13	ADK	BQN	06FC	First Class	05-01-2019	1123
	22	ERJ142	22	BGR	BJI	07EP	Economy Plus	09-02-2020	1132

passengers_on_flights 10 ×

3. Write a query to identify the number of passengers and total revenue in business class from the ticket_details table.

QUERY:

Select Count(customer_id) AS 'NO. OF PASSENGERS' , sum(Price_per_ticket) AS REVENUE from ticket_details Where class_id='Business' ;

OUTPUT:

	NO. OF PASSENGERS	REVENUE
▶	13	6034

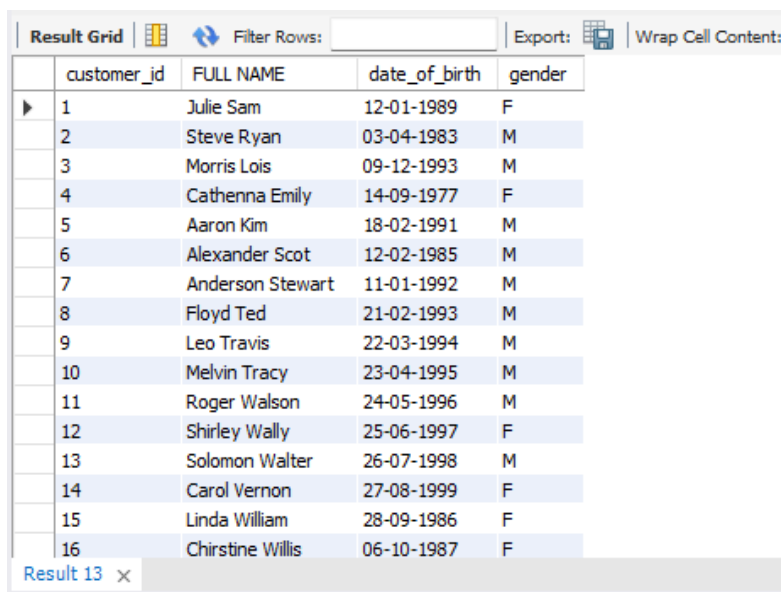
Result 12 ×

4. Write a query to display the full name of the customer by extracting the first name and last name from the customer table.

QUERY:

Select customer_id, concat(first_name, " ", last_name) AS "FULL NAME", date_of_birth, gender from customer;

OUTPUT:



The screenshot shows a database interface with a 'Result Grid' tab. It displays 16 rows of data with columns: customer_id, FULL NAME, date_of_birth, and gender. The data is as follows:

customer_id	FULL NAME	date_of_birth	gender
1	Julie Sam	12-01-1989	F
2	Steve Ryan	03-04-1983	M
3	Morris Lois	09-12-1993	M
4	Cathenna Emily	14-09-1977	F
5	Aaron Kim	18-02-1991	M
6	Alexander Scot	12-02-1985	M
7	Anderson Stewart	11-01-1992	M
8	Floyd Ted	21-02-1993	M
9	Leo Travis	22-03-1994	M
10	Melvin Tracy	23-04-1995	M
11	Roger Walson	24-05-1996	M
12	Shirley Wally	25-06-1997	F
13	Solomon Walter	26-07-1998	M
14	Carol Vernon	27-08-1999	F
15	Linda William	28-09-1986	F
16	Chirstine Willis	06-10-1987	F

5. Write a query to extract the customers who have registered and booked a ticket. Use data from the customer and ticket_details tables.

QUERY:

select c.customer_id , concat(c.first_name, ' ', c.last_name) as Full_Name, count(t.no_of_tickets) as Total_Tickets_booked from customer c

join ticket_details t on c.customer_id = t.customer_id

group by c.customer_id, Full_Name;

OUTPUT:

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
customer_id	Full_Name	Total_Tickets_booked	
27	Cherly Vernon	1	
22	Pheny Eri	1	
21	Chirsty Josh	1	
4	Cathenna Emily	2	
5	Aaron Kim	3	
7	Anderson Stewart	1	
8	Floyd Ted	2	
9	Leo Travis	2	
10	Melvin Tracy	1	
11	Roger Walson	3	
19	Joyce Paul	3	
13	Solomon Walter	1	
14	Carol Vernon	2	
25	Moss Morris	2	
16	Chirstine Willis	1	
17	Catherine Shad	1	

Result 14 ×

6. Write a query to identify the customer's first name and last name based on their customer ID and brand (Emirates) from the ticket_details table.

QUERY:

```
Select c.first_name,c.last_name,c.customer_id from customer c
```

```
join ticket_details t on c.customer_id = t.customer_id
```

```
Where t.brand='Emirates';
```

OUTPUT:

	first_name	last_name	customer_id
▶	Steve	Ryan	2
	Cathenna	Emily	4
	Cathenna	Emily	4
	Aaron	Kim	5
	Anderson	Stewart	7
	Leo	Travis	9
	Roger	Walson	11
	Roger	Walson	11
	Carol	Vernon	14
	Gloria	Richie	18
	Gloria	Richie	18
	Joyce	Paul	19
	Moss	Morris	25
	Moss	Morris	25
	Cherly	Vernon	27
	James	Robert	31

Result 15 ×

7. Write a query to identify the customers who have traveled by *Economy Plus* class using Group By and Having clause on the passengers_on_flights table.

QUERY:

Select sum(customer_id), class_id from passengers_on_flights GROUP BY Class_id

Having class_id = "Economy plus";

OUTPUT:

	sum(customer_id)	class_id
▶	226	Economy Plus

Result 2 ×

8. Write a query to identify whether the revenue has crossed 10000 using the IF clause on the ticket_details table.

QUERY:

```
select * from ticket_details;
```

```
select if(sum(no_of_tickets*price_per_ticket) > 1000, 'Revenue Crossed 10000', 'Revenue less than 10000') as Revenue_Status
```

```
from ticket_details;
```

OUTPUT:

	Revenue_Status
▶	Revenue Crossed 10000

ticket_details 16 Result 17 ×

9. Write a query to create and grant access to a new user to perform operations on a database.

QUERY:





Create user 'sonia' identified by 'Sonia@2003';

grant all privileges on aircargo.*to 'Sonia';

flush privileges;

OUTPUT:

MySQL Connections



Local instance MySQL80  root  localhost:3306	Sonia  sonia  127.0.0.1:3306
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------


10. Write a query to find the maximum ticket price for each class using window functions on the ticket_details table.

QUERY:

```
SELECT DISTINCT class_id, MAX(Price_per_ticket) OVER (PARTITION BY class_id) AS  
Max_class_price  
  
FROM ticket_details;
```

OUTPUT:

Result Grid   Filter Rows: <input type="text"/>		
	class_id	Max_class_price
▶	Bussiness	510
	Economy	190
	Economy Plus	295
	First Class	395

Result 18 

11. Write a query to extract the passengers whose route ID is 4 by improving the speed and performance of the passengers_on_flights table.

QUERY:

```
CREATE INDEX idx_route_id1 ON passengers_on_flights(route_id);
```

```
SELECT customer_id, class_id, seat_num, route_id
```

```
FROM passengers_on_flights
```

```
WHERE route_id = 4;
```

OUTPUT:

	customer_id	class_id	seat_num	route_id
▶	2	Economy	01E	4
	4	First Class	03FC	4
	11	Bussiness	05B	4




passengers_on_flights 19 ×

12. For the route ID 4, write a query to view the execution plan of the passengers_on_flights table

QUERY:

```
Select * from passengers_on_flights where route_id=4;
```

OUTPUT:

Result Grid  Filter Rows: <input type="text"/> Export:  Wrap Cell Content: 									
	customer_id	aircraft_id	route_id	depart	arrival	seat_num	class_id	travel_date	flight_num
▶	2	767-301ER	4	JFK	LAX	01E	Economy	02-09-2018	1114
	4	767-301ER	4	JFK	LAX	03FC	First Class	30-04-2020	1114
	11	767-301ER	4	JFK	LAX	05B	Bussiness	09-11-2020	1114

passengers_on_flights 21 x

13. Write a query to calculate the total price of all tickets booked by a customer across different aircraft IDs using rollup function.

QUERY:

```
SELECT customer_id, aircraft_id, SUM(price_per_ticket) AS total_price
FROM ticket_details
GROUP BY customer_id, aircraft_id WITH ROLLUP;
```

OUTPUT:

	customer_id	aircraft_id	total_price
▶	1	CRJ900	320
	1	ERJ142	250
	1	NULL	570
	2	767-301ER	130
	2	A321	505
	2	NULL	635
	4	767-301ER	780
	4	NULL	780
	5	767-301ER	430
	5	ERJ142	240
	5	NULL	670
	7	767-301ER	430
	7	NULL	430
	8	A321	465
	8	NULL	465
	9	767-301ER	380
	9	CRJ900	390
	9	NULL	770
	10	A321	135
	10	NULL	135

Result 22 ✕

14. Write a query to create a view with only business class customers along with the brand of airlines.

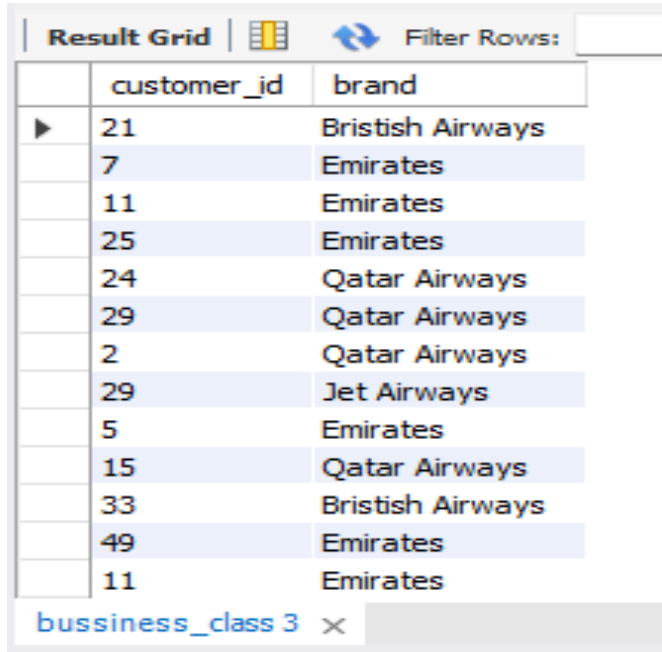
QUERY:

```
CREATE VIEW bussiness_class AS
```

```
SELECT customer_id,brand FROM ticket_details WHERE class_id='Bussiness';
```

```
SELECT * FROM bussiness_class;
```

OUTPUT:



	customer_id	brand
▶	21	Bristish Airways
	7	Emirates
	11	Emirates
	25	Emirates
	24	Qatar Airways
	29	Qatar Airways
	2	Qatar Airways
	29	Jet Airways
	5	Emirates
	15	Qatar Airways
	33	Bristish Airways
	49	Emirates
	11	Emirates

15. Write a query to create a stored procedure to get the details of all passengers flying between a range of routes defined in run time. Also, return an error message if the table doesn't exist.

QUERY:

```
CREATE DEFINER='root'@'localhost' PROCEDURE `new_procedure`( IN route_from  
VARCHAR(100), IN route_to VARCHAR(100))
```

```
BEGIN
```

```
DECLARE table_exists INT;
```

```
SELECT COUNT(*) INTO table_exists FROM
```

```

information_schema.tables
WHERE table_schema = DATABASE()
AND table_name = 'passengers_on_flights';
IF table_exists = 0 THEN
SELECT 'Error: Table passengers_on_flights does not exist';
ELSE
SELECT a.*,b.arrival, b.depart FROM customer a, passengers_on_flights b
WHERE b.route_id AND a.customer_id=b.customer_id
BETWEEN route_from AND route_to;
END IF;
END
CALL new_procedure(4,10);

```

OUTPUT:

Result Grid							
Filter Rows:		Export:		Wrap Cell Content:			
	customer_id	first_name	last_name	date_of_birth	gender	arrival	depart
▶	1	Julie	Sam	12-01-1989	F	CST	CDC
	1	Julie	Sam	12-01-1989	F	DEN	HNL
	1	Julie	Sam	12-01-1989	F	ANI	CAK
	1	Julie	Sam	12-01-1989	F	BOI	CBM
	1	Julie	Sam	12-01-1989	F	LAX	JFK
	1	Julie	Sam	12-01-1989	F	BJI	BGR
	1	Julie	Sam	12-01-1989	F	BOI	AVL
	1	Julie	Sam	12-01-1989	F	JFX	LAX
	1	Julie	Sam	12-01-1989	F	DAL	CST
	1	Julie	Sam	12-01-1989	F	BGR	ANI
	1	Julie	Sam	12-01-1989	F	ADK	ABI

16. Write a query to create a stored procedure that extracts all the details from the routes table where the travelled distance is more than 2000 miles.

QUERY:

```

CREATE DEFINER='root'@'localhost' PROCEDURE `route_details_procedure`()
BEGIN
select* from routes where distance_miles>2000;
END
CALL route_detail_procedure;

```

OUTPUT:

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

	route_id	flight_num	origin_airport	destination_airport	aircraft_id	distance_miles
▶	1	1111	EWR	HNL	767-301ER	4962
	2	1112	HNL	EWR	767-301ER	4962
	3	1113	EWR	LHR	A321	3466
	4	1114	JFK	LAX	767-301ER	2475
	5	1115	LAX	JFK	767-301ER	2475
	6	1116	HNL	LAX	767-301ER	2556
	10	1120	HNL	DEN	A321	3365
	12	1122	ABI	ADK	767-301ER	4300
	13	1123	ADK	BQN	A321	2232
	14	1124	BQN	CAK	A321	2445
	18	1128	ANI	BGR	ERJ142	2450
	19	1129	ATW	AVL	A321	2222
	20	1130	AVL	BOI	767-301ER	3134
	21	1131	BFL	BET	A321	2425

Result 2 ×

CONCLUSION

This project aims to leverage SQL to develop a robust database management and analysis system for Air Cargo. The insights generated will help in optimizing routes, enhancing customer satisfaction, and improving overall operational efficiency. By focusing on key areas such as regular customer identification, route analysis, and ticket sales, the project will enable Air Cargo to become more customer-centric and maintain its competitive edge in the aviation industry.