AIR CARGO ANALYSIS

DESCRIPTION: Air Cargo is an aviation company that provides air transportation services for passengers and freight. Air Cargo uses its aircraft to provide different services with the help of partnerships or alliances with other airlines. The company wants to prepare reports on regular passengers, busiest routes, ticket sales details, and other scenarios to improve the ease of travel and booking for customers.

PROBLEM STATEMENT: As a DBA expert, I need to focus on identifying the regular customers to provide offers, analyze the busiest route which helps to increase the number of aircraft required and prepare an analysis to determine the ticket sales details. This will ensure that the company improves its operability and becomes more customer-centric and a favorable choice for air travel.

This project includes almost all the concepts of SQL like **Stored Procedure, Window functions, roll up function, group by and having clause, order by clause, IF and CASE, Joins, use of constraints like primary key, foreign key, creating ER diagram etc.**

This project contains 4 tables, whose data description is given below:

Dataset description:

Customer: Contains the information of customers

- customer_id – ID of the customer
- first_name – First name of the customer
- last_name – Last name of the customer
- date_of_birth – Date of birth of the customer
- gender – Gender of the customer

passengers_on_flights: Contains information about the travel details

- aircraft_id – ID of each aircraft in a brand
- route_id – Route ID of from and to location
- customer_id – ID of the customer
- depart – Departure place from the airport
- arrival – Arrival place in the airport
- seat_num – Unique seat number for each passenger
- class_id – ID of travel class
- travel_date – Travel date of each passenger
- flight_num – Specific flight number for each route

ticket_details: Contains information about the ticket details

- p_date – Ticket purchase date

- customer_id – ID of the customer
- aircraft_id – ID of each aircraft in a brand
- class_id – ID of travel class
- no_of_tickets – Number of tickets purchased
- a_code – Code of each airport
- price_per_ticket – Price of a ticket
- brand – Aviation service provider for each aircraft

routes: Contains information about the route details

- Route_id – Route ID of from and to location
- Flight_num – Specific fight number for each route
- Origin_airport – Departure location
- Destination_airport – Arrival location
- Aircraft_id – ID of each aircraft in a brand
- Distance_miles – Distance between departure and arrival location
-

The following are the tasks performed using all the above concepts for analysing the data in the tables consisting data of AirCArgo.:

The screenshot of the output of all the tasks has been pasted after every query performed.

TASKS:

1. Create an ER diagram for the given airlines database.

```
create database aviation;
use aviation;
```

Since none of the table have primary key, so lets define it before joining the table using foreign key

In customer table the primary key is customer_id, in routes table the primary key is route_id. Lets alter the table and add the primary keys respectively

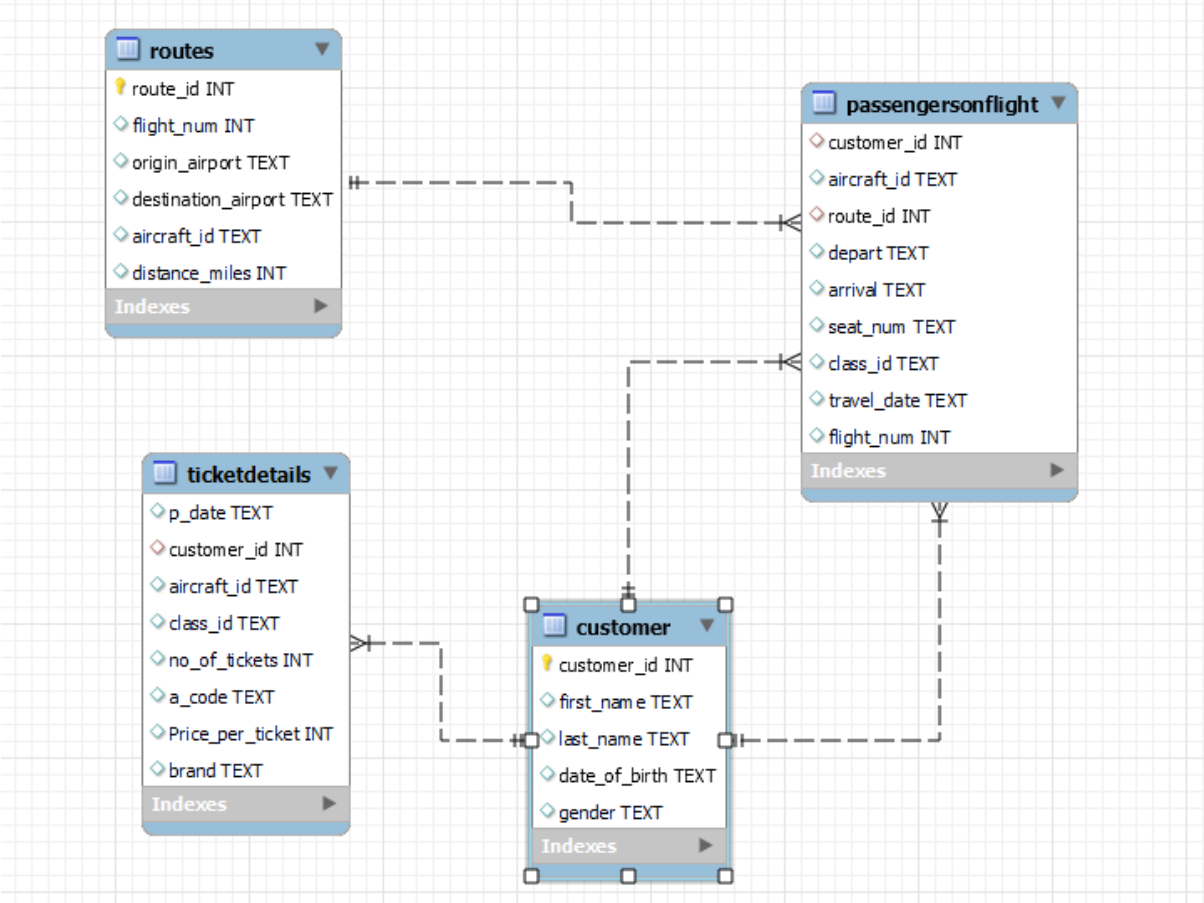
But the tables passengsonflight and ticketdetails do not have specific column for primary key.
So joining all the tables based on the above details.

```
ALTER TABLE customer
ADD CONSTRAINT pk PRIMARY KEY(customer_id);

ALTER TABLE routes
ADD CONSTRAINT pk1 PRIMARY KEY(route_id);

ALTER TABLE passengersonflight
```

ADD CONSTRAINT fk FOREIGN KEY(customer_id)
REFERENCES customer(customer_id);

ALTER TABLE ticketdetails
ADD CONSTRAINT fk1 FOREIGN KEY(customer_id)
REFERENCES customer(customer_id);

ALTER TABLE passengersonflight
ADD CONSTRAINT fk2 FOREIGN KEY(route_id)
REFERENCES routes(route_id);



2. Write a query to create 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.

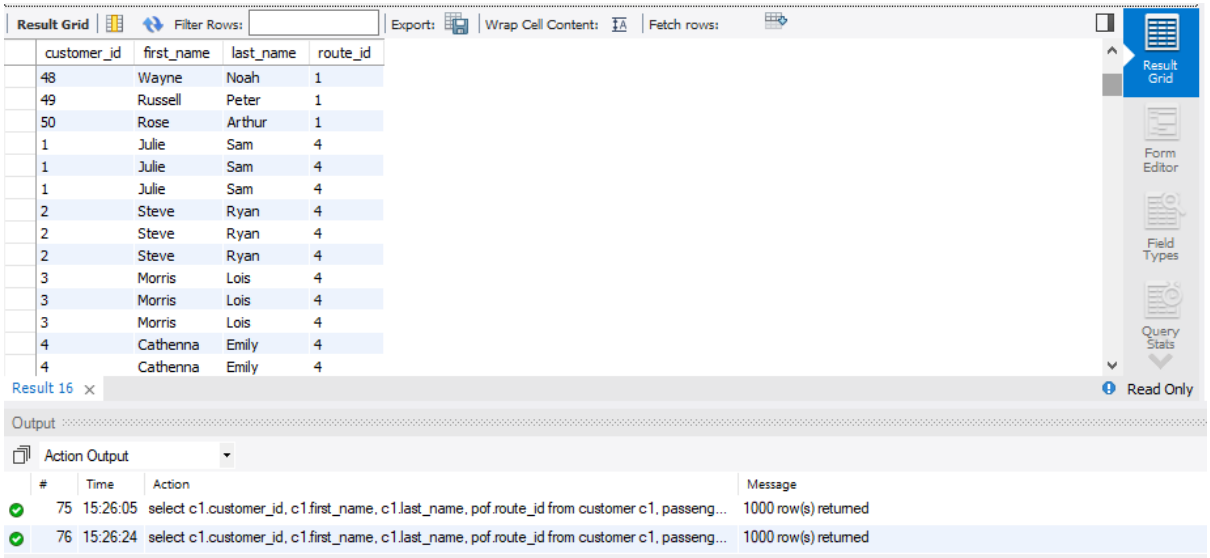
   SELECT *
   FROM routes
   WHERE distance_miles <= 0;

   alter table routes
   add constraint c1

CHECK (distance_miles > 0);

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

```
select c1.customer_id, c1.first_name, c1.last_name, pof.route_id
from customer c1, passengersonflight pof
where route_id between 1 AND 25
order by route_id;
```



customer_id	first_name	last_name	route_id
48	Wayne	Noah	1
49	Russell	Peter	1
50	Rose	Arthur	1
1	Julie	Sam	4
1	Julie	Sam	4
1	Julie	Sam	4
2	Steve	Ryan	4
2	Steve	Ryan	4
2	Steve	Ryan	4
3	Morris	Lois	4
3	Morris	Lois	4
3	Morris	Lois	4
4	Cathenna	Emily	4
4	Cathenna	Emily	4

Result 16

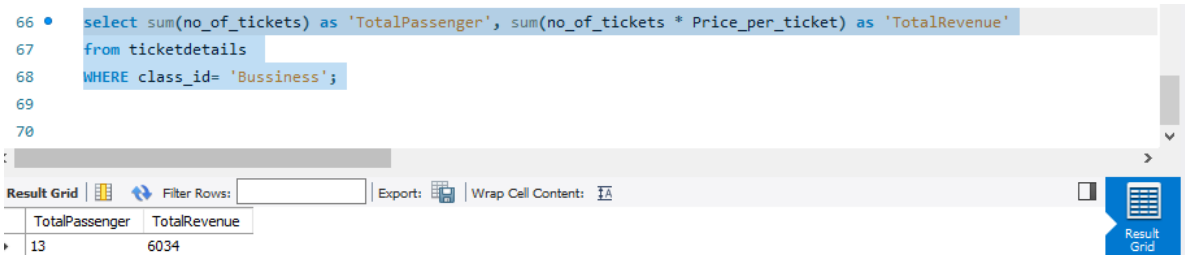
Output

Action Output

#	Time	Action	Message
75	15:26:05	select c1.customer_id, c1.first_name, c1.last_name, pof.route_id from customer c1, passeng...	1000 row(s) returned
76	15:26:24	select c1.customer_id, c1.first_name, c1.last_name, pof.route_id from customer c1, passeng...	1000 row(s) returned

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

```
select sum(no_of_tickets) as 'TotalPassenger', sum(no_of_tickets *
Price_per_ticket) as 'TotalRevenue'
from ticketdetails
WHERE class_id= 'Bussiness';
```



```
select sum(no_of_tickets) as 'TotalPassenger', sum(no_of_tickets * Price_per_ticket) as 'TotalRevenue'
from ticketdetails
WHERE class_id= 'Bussiness';
```

TotalPassenger	TotalRevenue
13	6034

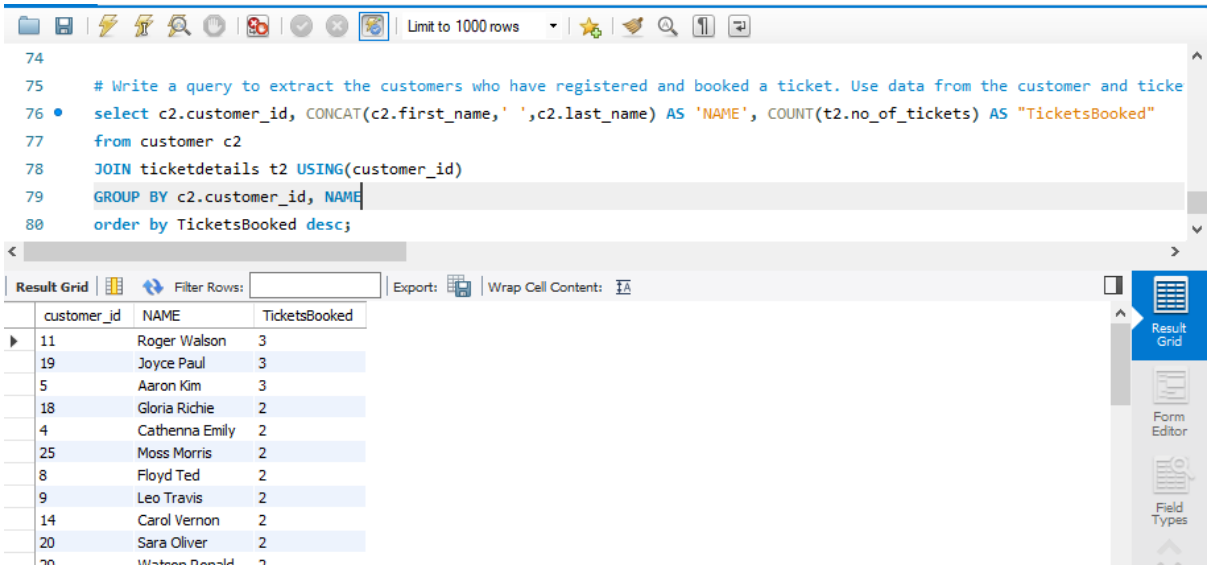
From above it can be seen that there are 13 passengers travelling through Bussiness Class and total revenue earned from them is 6034 unit.

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

```
select CONCAT(first_name,' ',last_name) AS NAME
from customer;
```

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

```
select c2.customer_id, CONCAT(c2.first_name,' ',c2.last_name) AS 'NAME',
COUNT(t2.no_of_tickets) AS "TicketsBooked"
from customer c2
JOIN ticketdetails t2 USING(customer_id)
GROUP BY c2.customer_id, NAME
order by TicketsBooked desc;
```



7. 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.

```
select c3.customer_id, c3.first_name,c3.last_name,t3.brand
from customer c3
JOIN ticketdetails t3 USING(customer_id)
WHERE t3.brand='Emirates'
ORDER BY c3.customer_id;
```

```
select * from ticketdetails;

select c3.customer_id, c3.first_name,c3.last_name,t3.brand
from customer c3
JOIN ticketdetails t3 USING(customer_id)
WHERE t3.brand='Emirates'
ORDER BY c3.customer_id;
```

customer_id	first_name	last_name	brand
2	Steve	Ryan	Emirates
4	Cathenna	Emily	Emirates
4	Cathenna	Emily	Emirates
5	Aaron	Kim	Emirates
7	Anderson	Stewart	Emirates
9	Leo	Travis	Emirates
11	Roger	Walson	Emirates
11	Roger	Walson	Emirates
14	Carol	Vernon	Emirates
18	Gloria	Richie	Emirates
18	Gloria	Richie	Emirates

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

select c4.customer_id, c4.first_name,c4.last_name
from customer c4
JOIN passengersonflight p4 USING(customer_id)
GROUP BY c4.customer_id, c4.first_name,c4.last_name
HAVING SUM(p4.class_id="Economy Plus")>0;

```
select * from passengersonflight;

select c4.customer_id, c4.first_name,c4.last_name
from customer c4
JOIN passengersonflight p4 USING(customer_id)
GROUP BY c4.customer_id, c4.first_name,c4.last_name
HAVING SUM(p4.class_id="Economy Plus")>0;
```

customer_id	first_name	last_name
1	Julie	Sam
8	Floyd	Ted
11	Roger	Walson
17	Catherine	Shad
19	Joyce	Paul
22	Pheny	Eri
32	Chirstoper	Sean
47	Sophia	Carl
50	Rose	Arthur

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

select if(sum(no_of_tickets*Price_per_ticket)>10000,'Revenue Crosses 10000', 'Revenue less than 10000') AS 'REVENUE STATUS'
from ticketdetails;

```
select if(sum(no_of_tickets*Price_per_ticket)>10000,'Revenue Crosses 10000', 'Revenue less than 10000') AS 'REVENUE ST.
from ticketdetails;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

REVENUE STATUS
Revenue Crosses 10000

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

select *,
MAX(Price_per_ticket) OVER (partition by class_id) AS MAX_TICKET_PRICE
from ticketdetails;

```
select *,
MAX(Price_per_ticket) OVER (partition by class_id) AS MAX_TICKET_PRICE
from ticketdetails;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

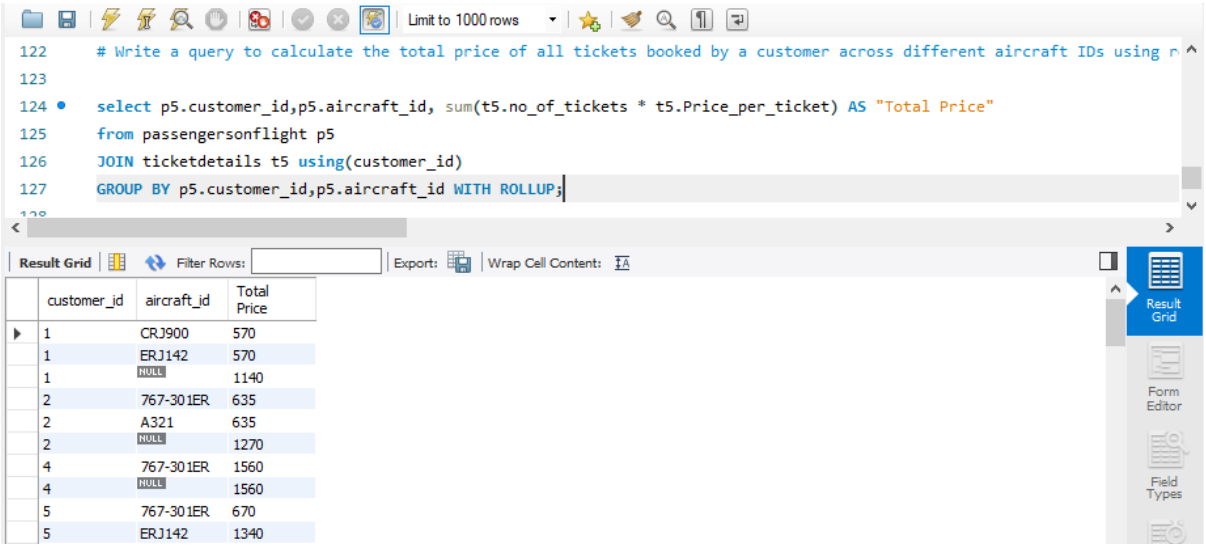
p_date	customer_id	aircraft_id	class_id	no_of_tickets	a_code	Price_per_ticket	brand	MAX_TICKET_PRICE
25-01-2019	2	A321	Bussiness	1	YVR	505	Qatar Airways	510
11-11-2020	11	767-301ER	Bussiness	1	AGB	465	Emirates	510
22-10-2019	29	A321	Bussiness	1	PEK	410	Qatar Airways	510
01-07-2018	5	767-301ER	Bussiness	1	BFS	430	Emirates	510
08-11-2020	11	767-301ER	Bussiness	1	AGB	465	Emirates	510
26-12-2018	27	767-301ER	Economy	1	DAL	130	Emirates	190
01-09-2018	2	767-301ER	Economy	1	DAL	130	Emirates	190
01-12-2018	28	ERJ142	Economy	1	BHX	170	Jet Airways	190
05-05-2020	5	ERJ142	Economy	1	CTM	120	Jet Airways	190
02-02-2019	14	ERJ142	Economy	1	CTM	120	Jet Airways	190
19-12-2018	31	767-301ER	Economy	1	DAL	130	Emirates	190

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

select * from passengersonflight
WHERE route_id=4;

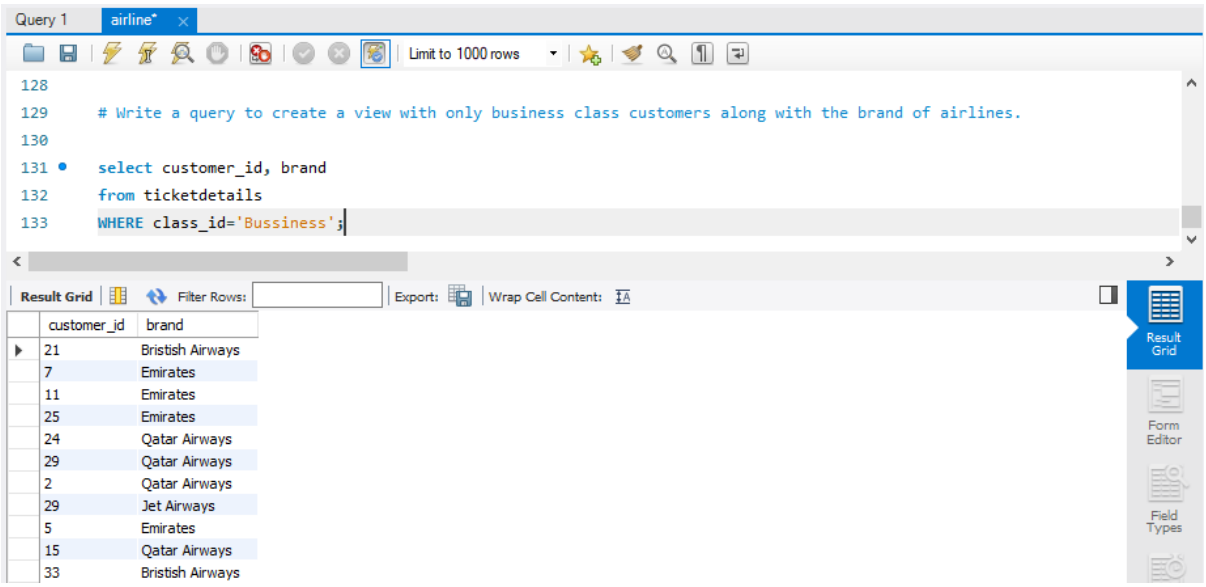
12. Write a query to calculate the total price of all tickets booked by a customer across different aircraft IDs using rollup function.

select p5.customer_id,p5.aircraft_id, sum(t5.no_of_tickets * t5.Price_per_ticket) AS "Total Price"
from passengersonflight p5
JOIN ticketdetails t5 using(customer_id)
GROUP BY p5.customer_id,p5.aircraft_id WITH ROLLUP;



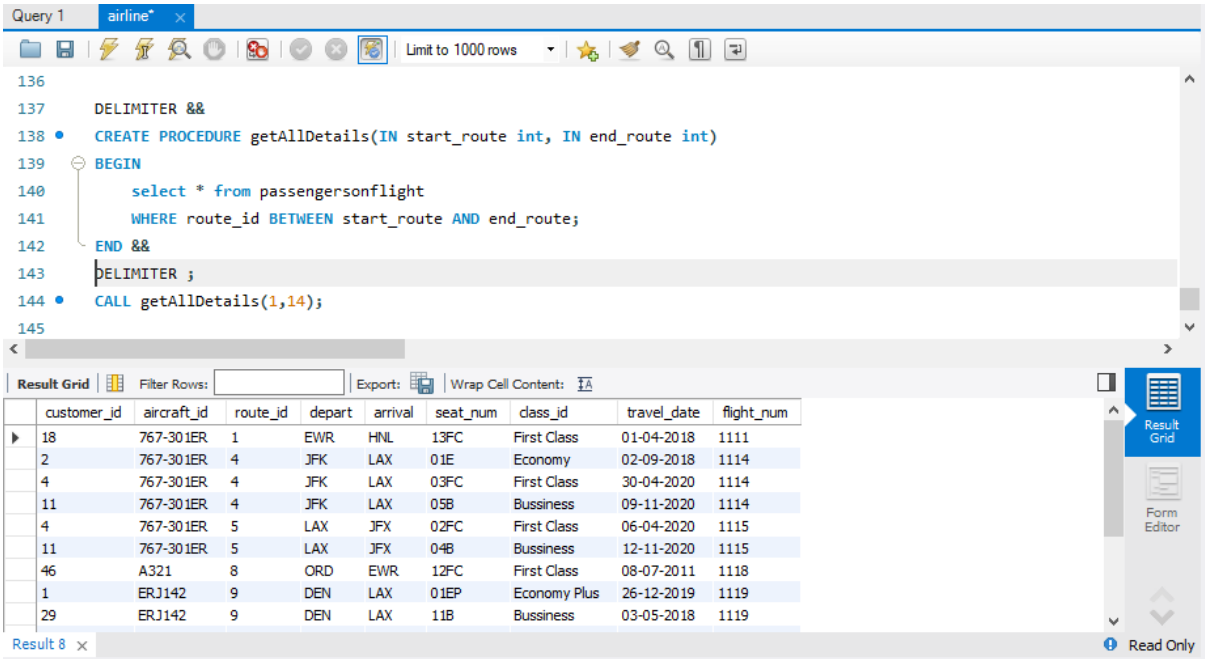
13. Write a query to create a view with only business class customers along with the brand of airlines.

select customer_id, brand
from ticketdetails
WHERE class_id='Bussiness';



14. 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.

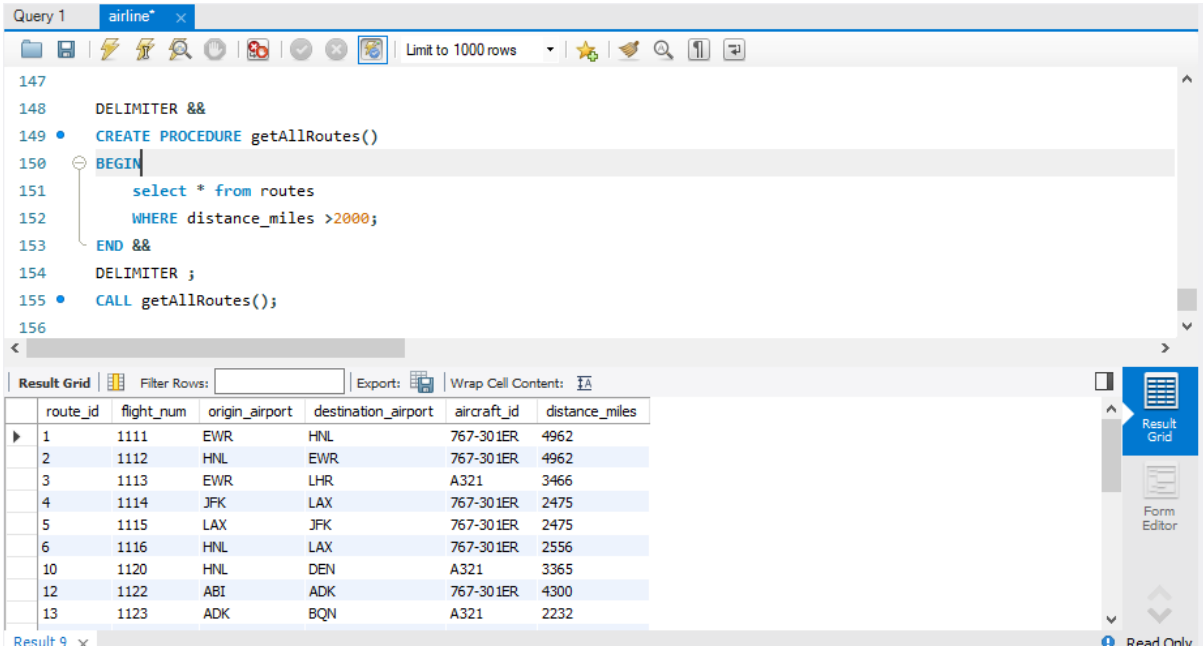
```
DELIMITER &&
CREATE PROCEDURE getAllDetails(IN start_route int, IN end_route int)
BEGIN
        select * from passengersonflight
   WHERE route_id BETWEEN start_route AND end_route;
END &&

DELIMITER ;

CALL getAllDetails(1,14);
```



customer_id	aircraft_id	route_id	depart	arrival	seat_num	class_id	travel_date	flight_num
18	767-301ER	1	EWR	HNL	13FC	First Class	01-04-2018	1111
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
4	767-301ER	5	LAX	JFX	02FC	First Class	06-04-2020	1115
11	767-301ER	5	LAX	JFX	04B	Bussiness	12-11-2020	1115
46	A321	8	ORD	EWR	12FC	First Class	08-07-2011	1118
1	ERJ142	9	DEN	LAX	01EP	Economy Plus	26-12-2019	1119
29	ERJ142	9	DEN	LAX	11B	Bussiness	03-05-2018	1119

15. 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.

```
DELIMITER &&
CREATE PROCEDURE getAllRoutes()
BEGIN
        select * from routes
        WHERE distance_miles >2000;
END &&
DELIMITER ;
CALL getAllRoutes();
```



16. Write a query to create a stored procedure that groups the distance travelled by each flight into three categories. The categories are, short distance travel (SDT) for >=0 AND <= 2000 miles, intermediate distance travel (IDT) for >2000 AND <=6500, and long-distance travel (LDT) for >6500.

```
DELIMITER &&
CREATE PROCEDURE getCategories()
BEGIN
        select flight_num , distance_miles ,
   CASE
                WHEN distance_miles BETWEEN 0 AND 2000 THEN  "Short
Distance"
                WHEN distance_miles BETWEEN 2001 AND 6500 THEN
"Intermediate"
                ELSE "Long Distance"
   END AS "Category"
   from routes;
END &&
DELIMITER ;
CALL getCategories();
```

```
BEGIN
    select flight_num,distance_miles,
    CASE
        WHEN distance_miles BETWEEN 0 AND 2000 THEN  "Short Distance"
        WHEN distance_miles BETWEEN 2001 AND 6500 THEN  "Intermediate"
        ELSE "Long Distance"
    END AS "Category"
    from routes;
END &&
DELIMITER ;
CALL getCategories();
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

flight_num	distance_miles	Category
1114	2475	Intermediate
1115	2475	Intermediate
1116	2556	Intermediate
1117	1745	Short Distance
1118	719	Short Distance
1119	862	Short Distance
1120	3365	Intermediate

17. Write a query to extract ticket purchase date, customer ID, class ID and specify if the complimentary services are provided for the specific class using a stored function in stored procedure on the ticket_details table.
    Condition:

    If the class is Business and Economy Plus, then complimentary services are given as Yes, else it is No

```
DELIMITER &&
CREATE PROCEDURE getComplementaryServiceDetails()
BEGIN
select p_date,customer_id,class_id,
CASE
        WHEN class_id = "Bussiness"  THEN "YES"
   WHEN class_id = "Economy Plus"  THEN "YES"
   ELSE "NO"
END AS "Complimentary Services"
from ticketdetails;
END &&
DELIMITER ;

CALL getComplementaryServiceDetails();
```

```
CASE
    WHEN class_id = "Bussiness"  THEN "YES"
    WHEN class_id = "Economy Plus"  THEN "YES"
    ELSE "NO"
END AS "Complimentary Services"
```

p_date	customer_id	class_id	Complimentary Services
26-12-2018	27	Economy	NO
02-02-2020	22	Economy Plus	YES
03-03-2020	21	Bussiness	YES
04-04-2020	4	First Class	NO
05-05-2020	5	Economy	NO
07-07-2020	7	Bussiness	YES
08-08-2020	8	Economy Plus	YES
09-09-2020	9	First Class	NO
10-10-2020	10	Economy	NO
11-11-2020	11	Bussiness	YES
12-12-2020	19	Economy Plus	YES
01-01-2019	13	First Class	NO
02-02-2019	14	Economy	NO
03-03-2019	25	Bussiness	YES

18. Write a query to extract the first record of the customer whose last name ends with Scott from the customer table.

select * from customer
where last_name = 'Scott'
LIMIT 1;

```
select * from customer
where last_name = 'Scott'
LIMIT 1;
```

customer_id	first_name	last_name	date_of_birth	gender
37	Samuel	Scott	28-01-2000	M
NULL	NULL	NULL	NULL	NULL

CONCLUSION :

In conclusion, this project involved an analysis of an air cargo company's database, focusing on various aspects such as entity-relationship modeling, database querying, stored procedure creation, and data extraction.

The key tasks performed throughout the project are as follows:

1. Created an ER diagram to visualize the relationships within the airlines database, helping to understand the structure of the data.

2. Designed a query to create the route_details table, considering appropriate data types for fields such as route ID, flight number, origin airport, destination airport, aircraft

ID, and distance in miles. Implemented check and unique constraints for flight numbers and route IDs, respectively, and ensured the distance in miles is greater than 0.

3. Wrote a query to display all the passengers who traveled on routes 01 to 25, retrieving data from the passengers_on_flights table.

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

5. Constructed a query to display the full name of the customers by extracting the first name and last name from the customer table.

6. Formulated a query to extract the customers who have registered and booked a ticket, utilizing data from the customer and ticket_details tables.

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

8. Utilized the Group By and Having clauses to write a query that identifies the customers who have traveled by the Economy Plus class, leveraging the passengers_on_flights table.

9. Implemented an IF clause within a query to identify whether the revenue has exceeded 10,000, examining the ticket_details table.

10. Developed a query to create and grant access to a new user, allowing them to perform operations on the database.

11. Utilized window functions in a query to determine the maximum ticket price for each class from the ticket_details table.

12. Optimized the performance of a query to extract passengers with a specific route ID (4) from the passengers_on_flights table.

13. Created a query to view the execution plan of the passengers_on_flights table for the specified route ID (4).

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

15. Created a view containing only business class customers along with the brand of airlines.

16. Designed a stored procedure to retrieve details of all passengers flying between a range of routes.

17. Developed a stored procedure to extract details from the routes table where the traveled distance is more than 2000 miles.

18. Created a stored procedure to group the distance traveled by each flight into three categories: short distance travel (SDT), intermediate distance travel (IDT), and long-distance travel (LDT).

19. Utilized a stored function within a stored procedure to extract the ticket purchase date, customer ID, class ID, and information on whether complimentary services are provided based on the specific class.