Problem Statement Scenario:

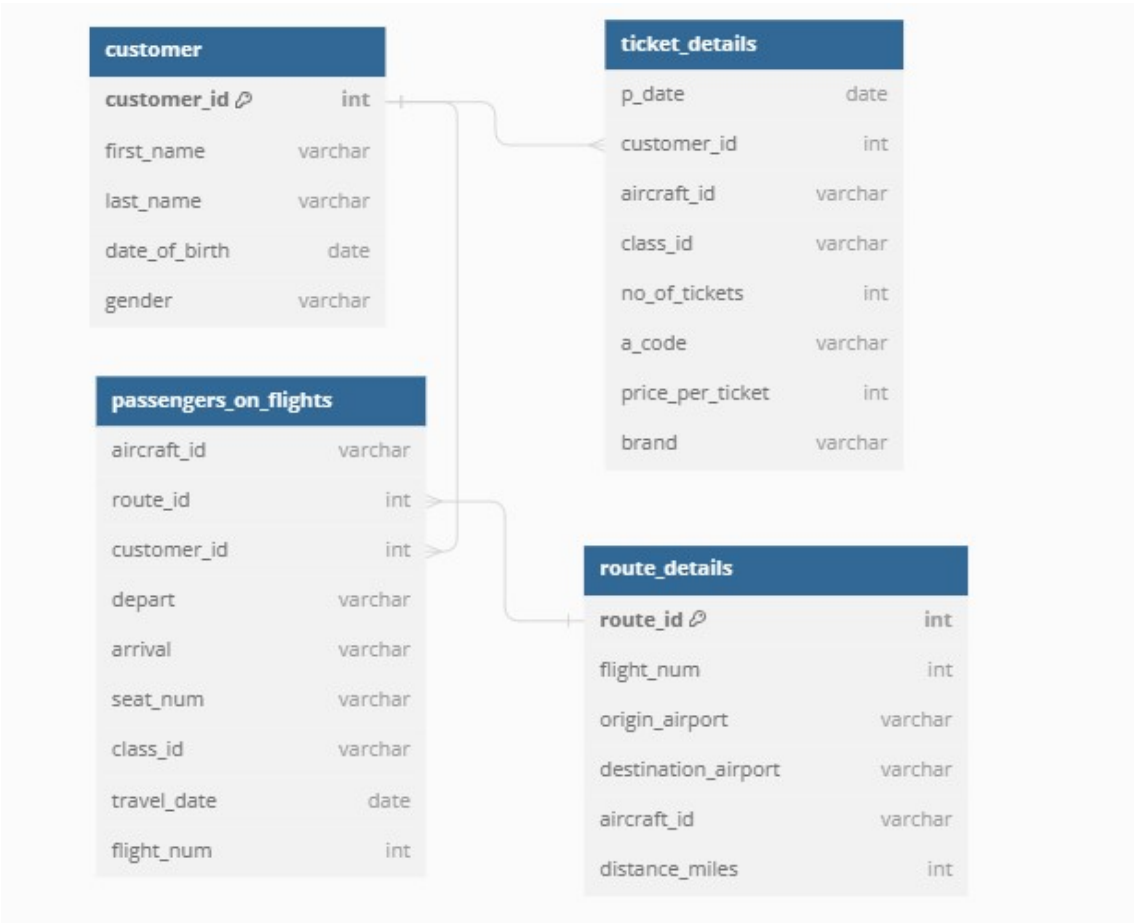
Air Cargo is an aviation company that provides air transportation services for passengers and freight. Air Cargo uses its aircraft to provide different services with the help of partnerships or alliances with other airlines. The company wants to prepare reports on regular passengers, busiest routes,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

Project Objective:

Identify 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.

---

1. Introduction: The Air cargo database consists of 4 tables namely ticket details, customer details, route details and flight details. Create an Entity – Relationship diagram for the relational database, establishing relationships between the entities. Each table should have a primary key or a foreign key to show relationship set with other tables. Used dbdiagram.io to create the ER diagram.



2. Route details table creation: Created a route details table in MySQL, with check constraint for the flight number and unique constraint for the route_id fields and for distance miles greater than 0.

```
use aircargo;

create table route_details (route_id int primary key, flight_num int not null, origin_airport varchar(200), destination_airport varchar(200),
aircraft_id varchar(200), distance_miles int check (distance_miles > 0));
select * from route_details;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

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

3. MySQL Queries for passenger details: Extract the information on passengers travelling between routes 1 to 25. Total 26 entries extracted

```
select * from passengers_on_flights where route_id between 01 and 25;
```

ult Grid | Filter Rows: | 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
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
24	A321	14	BQN	CAK	08B	Bussiness	22-07-2019	1124
25	767-301ER	23	BLV	BFL	09B	Bussiness	07-03-2019	1133
50	A321	21	BFL	BET	10EP	Economy Plus	15-08-2020	1131
29	ERJ142	9	DEN	LAX	11B	Bussiness	03-05-2018	1119

4. **Extract information on the total revenue generated from the business class of the airlines**: Using where clause and group by clause, there are total 13 passengers who travelled in business class in the given time period and total revenue generated is 6034.

```
select t.class_id, count(*) total_passengers, sum(no_of_tickets * Price_per_ticket) total_revenue from ticket_details t where class_id = 'bussiness'
group by class_id;
```

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

class_id	total_passengers	total_revenue
Bussiness	13	6034

5. **Customer details along with full name**: full name of the passengers extracted from first name and last name columns

```
select c.customer_id, concat(first_name,' ', last_name) as full_name, c.date_of_birth, c.gender from customer c;
```

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

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

6. **Join tables to extract customers who have registered and booked a ticket.** Query returned 50 ticket details travelling on different aircrafts from all the classes

```
select c.*, t.aircraft_id, t.no_of_tickets, t.class_id, t.a_code from ticket_details t
left join customer c on c.customer_id = t.customer_id;
```

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

customer_id	first_name	last_name	date_of_birth	gender	aircraft_id	no_of_tickets	class_id	a_code
27	Cherly	Vernon	19-03-1992	F	767-301ER	1	Economy	DAL
22	Pheny	Eri	29-01-1999	M	ERJ142	1	Economy Plus	AGB
21	Chirsty	Josh	10-01-2004	M	CRJ900	1	Bussiness	BOH
4	Cathenna	Emily	14-09-1977	F	767-301ER	1	First Class	AGB
5	Aaron	Kim	18-02-1991	M	ERJ142	1	Economy	CTM
7	Anderson	Stewart	11-01-1992	M	767-301ER	1	Bussiness	BFS
8	Floyd	Ted	21-02-1993	M	A321	1	Economy Plus	DAL
9	Leo	Travis	22-03-1994	M	767-301ER	1	First Class	BOH
10	Melvin	Tracy	23-04-1995	M	A321	1	Economy	MCO
11	Roger	Walson	24-05-1996	M	767-301ER	1	Bussiness	AGB

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**. Total 18 customer entries booked in Emirates airlines

```
select c.* , t.aircraft_id, t.brand from ticket_details t
left join customer c on c.customer_id = t.customer_id
where brand = 'Emirates';
```

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

customer_id	first_name	last_name	date_of_birth	gender	aircraft_id	brand
27	Cherly	Vernon	19-03-1992	F	767-301ER	Emirates
4	Cathenna	Emily	14-09-1977	F	767-301ER	Emirates
7	Anderson	Stewart	11-01-1992	M	767-301ER	Emirates
9	Leo	Travis	22-03-1994	M	767-301ER	Emirates
11	Roger	Walson	24-05-1996	M	767-301ER	Emirates
25	Moss	Morris	18-02-2011	M	767-301ER	Emirates
18	Gloria	Richie	04-12-1989	F	767-301ER	Emirates
25	Moss	Morris	18-02-2011	M	767-301ER	Emirates
14	Carol	Vernon	27-08-1999	F	767-301ER	Emirates
19	Joyce	Paul	02-06-1990	F	767-301ER	Emirates
18	Gloria	Richie	04-12-1989	F	767-301ER	Emirates
5	Aaron	Kim	18-02-1991	M	767-301ER	Emirates

Result 16

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. Total 10 entries booked by customers in Economy Plus

```
select c.customer_id, c.first_name, p.route_id, p.class_id from passengers_on_flights p
left join customer c on c.customer_id = p.customer_id
group by c.customer_id, c.first_name, p.route_id, p.class_id
having class_id = 'Economy Plus';
```

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

customer_id	first_name	route_id	class_id
1	Julie	9	Economy Plus
8	Floyd	38	Economy Plus
11	Roger	31	Economy Plus
17	Catherine	13	Economy Plus
19	Joyce	47	Economy Plus
19	Joyce	30	Economy Plus
22	Pheny	22	Economy Plus
32	Chirstoper	31	Economy Plus
47	Sophia	33	Economy Plus
50	Rose	21	Economy Plus

9. Write a query to identify whether the revenue has crossed 10000 using the IF clause on the ticket_details table. The total revenue for the air cargo inclusive of all the classes has crossed 10K

```
select sum(no_of_tickets * price_per_ticket) total_revenue,
(select if ((sum(no_of_tickets * price_per_ticket)) > 10000, 'Yes', 'No'))total_crossed_10000 from ticket_details;
```

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

total_revenue	total_crossed_10000
15369	Yes

Result Grid

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

```
create user `NewUser` identified by 'NewPassword';
GRANT SELECT ON *.* TO `NewUser`;
```

Output

Action Output

#	Time	Action	Message
1	18:24:27	create user 'NewUser' identified by 'NewPassword'	0 row(s) affected
2	18:24:30	GRANT SELECT ON *.* TO 'NewUser'	0 row(s) affected

11. Write a query to find the maximum ticket price for each class using window functions on the ticket_details table. Window function with partition clause used to extract the ticket price information for each class and the details are given below.

```
select distinct class_id, max(price_per_ticket) over (partition by  class_id order by price_per_ticket desc) Max_price_per_class
from ticket_details;
```

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

class_id	Max_price_per_class
Bussiness	510
Economy	190
Economy Plus	295
First Class	395

12. Write a query to extract the passengers whose route ID is 4 by improving the speed and performance of the passengers_on_flights table. Customers travelling on Route id 4 is extracted from the given data using select and left join on customer table.

```
select p.customer_id, c.first_name, c.last_name, c.gender,
p.aircraft_id, p.route_id, p.depart, p.arrival, p.flight_num from passengers_on_flights p
left join customer c on c.customer_id = p.customer_id
where route_id = 4;
```

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

customer_id	first_name	last_name	gender	aircraft_id	route_id	depart	arrival	flight_num
2	Steve	Ryan	M	767-301ER	4	JFK	LAX	1114
4	Cathenna	Emily	F	767-301ER	4	JFK	LAX	1114
11	Roger	Walson	M	767-301ER	4	JFK	LAX	1114

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

```
explain select p.customer_id, c.first_name, c.last_name, c.gender,
p.aircraft_id, p.route_id, p.depart, p.arrival, p.flight_num from passengers_on_flights p
left join customer c on c.customer_id = p.customer_id
where route_id = 4;
```

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

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	p	NULL	ALL	NULL	NULL	NULL	NULL	50	10.00	Using where
1	SIMPLE	c	NULL	ALL	NULL	NULL	NULL	NULL	50	100.00	Using where; Using join buffer (hash join)

14. Write a query to calculate the total price of all tickets booked by a customer across different aircraft IDs using rollup function. Total tickets booked by all customers calculated and the total price amounts to 15369 using rollup funcion

```
select t.customer_id, sum(no_of_tickets * Price_per_ticket) total_price_all_tickets from ticket_details t
left join customer c on c.customer_id = t.customer_id
group by t.customer_id
with rollup;
```

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

customer_id	total_price_all_tickets
1	570
2	635
4	780
5	670
7	430
8	465
9	770
10	135
11	1225
13	395

15**. Write a query to create a view with only business class customers along with the brand of airlines.** Number of customers travelling in business class across different brands are 13.

```
select t.customer_id, c.first_name, c.last_name,t.class_id, t.brand from ticket_details t
left join customer c on c.customer_id = t.customer_id
where class_id = 'Bussiness';
```

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

customer_id	first_name	last_name	class_id	brand
21	Chirsty	Josh	Bussiness	Bristish Airways
7	Anderson	Stewart	Bussiness	Emirates
11	Roger	Walson	Bussiness	Emirates
25	Moss	Morris	Bussiness	Emirates
24	Calvin	Willis	Bussiness	Qatar Airways
29	Watson	Ronald	Bussiness	Qatar Airways
2	Steve	Ryan	Bussiness	Qatar Airways
29	Watson	Ronald	Bussiness	Jet Airways
5	Aaron	Kim	Bussiness	Emirates
15	Linda	William	Bussiness	Qatar Airways

16. **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.** Stored procedure to get passenger details between route 1 and 50, total 50 entries returned.

```
delimiter //
create procedure range_of_routes(route_id int)
begin
select * from passengers_on_flights
where if (route_id between 1 and 50, True, False) = 1;
end //
delimiter ;

call range_of_routes(50);
```

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

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

Result 62

17. **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**. Stored procedure created, there are 24 flights with distance travelled more than 2000 miles.

```
delimiter //
create procedure routes_2000miles()
begin
    select * from routes
    where distance_miles > 2000;
end //
delimiter ;
call routes_2000miles();
```

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. **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.**

Stored procedure created and the flight travel distance categorized into 3 categories namely short, intermediate and long-distance travel.

```
delimiter //
create procedure distance_category()
begin
    select flight_num, aircraft_id, distance_miles, case when distance_miles >= 0 and distance_miles <= 2000 then 'Short distance travel'
    when distance_miles > 2000 and distance_miles <= 6500 then 'Intermediate distance travel'
    else 'Long distance travel' end distance_category
    from routes
    group by flight_num, aircraft_id, distance_miles;
end //
delimiter ;
call distance_category();
```

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

flight_num	aircraft_id	distance_miles	distance_category
1111	767-301ER	4962	Intermediate distance travel
1112	767-301ER	4962	Intermediate distance travel
1113	A321	3466	Intermediate distance travel
1114	767-301ER	2475	Intermediate distance travel
1115	767-301ER	2475	Intermediate distance travel
1116	767-301ER	2556	Intermediate distance travel
1117	A321	1745	Short distance travel
1118	A321	719	Short distance travel
1119	ERJ142	862	Short distance travel
1120	A321	3365	Intermediate distance travel
1122	767-301ER	4300	Intermediate distance travel
1123	A321	2232	Intermediate distance travel

19. **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

Stored function created to set complimentary service YES or NO based on the classID, Stored procedure created using the function to extract information on complimentary service provided or not to the Business and Economy Plus class passengers.

```
        else set complimentary_service = 'NO';
    end if;
    return (complimentary_service);
end //

create procedure ticket_service()
begin
    select p_date, customer_id, class_id, complimentary_service(class_id) from ticket_details
    group by p_date, customer_id, class_id;
end //
delimiter ;
call ticket_service();
```

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

p_date	customer_id	class_id	complimentary_service(class_id)
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

20. **Write a query to extract the first record of the customer whose last name ends with Scott using a cursor from the customer table.** Customers with last name as Scott filtered with where clause and the first record extracted using window function, the first record found is Samuel Scott

```
select *, first_value(first_name) over (order by customer_id) first_record from customer
where last_name like 'Scott';
```

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

customer_id	first_name	last_name	date_of_birth	gender	first_record
37	Samuel	Scott	28-01-2000	M	Samuel
38	Alexis	Scott	31-10-2001	M	Samuel

Project submitted by

Phebe Prasanthi