Air Cargo Analysis

Simplilearn SQL Training

Project- I

By- Anushree Bhargava

Table of Contents

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.

Project Objective:

You, as a DBA expert, 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.

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

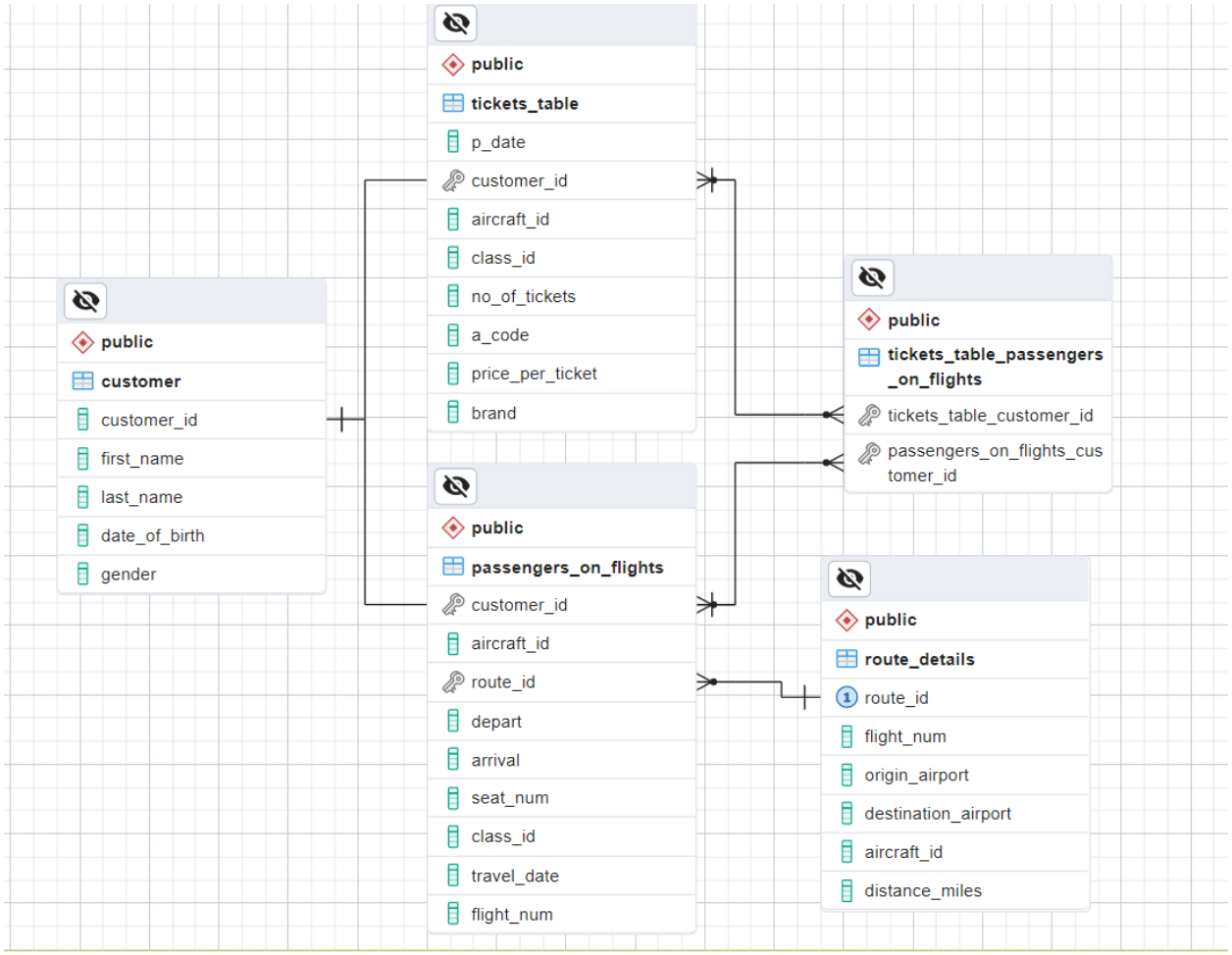
- brand – Aviation service provider for each aircraft

routes: Contains information about the route details

- Route_id – Route ID of from and to location
- Flight_num – Specific flight 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

Following operations should be performed:

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



Task 2: 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 Query History

```
Task 2: Write a query to create a route_info 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.

create table route_info( route_id int unique, flight_num int, origin_airport varchar,
destination_airport varchar, aircraft_id varchar, distance_miles int,
check(flight_num between 1111 and 1160), check(distance_miles>0));

select * from route_info;
```

Air cargo/postgres@PostgreSQL 15

No limit

Data Output Messages Notifications

	route_id integer	flight_num integer	origin_airport character varying	destination_airport character varying	aircraft_id character varying	distance_miles integer
1	1	1111	EWR	HNL	767-301ER	4962
2	2	1112	HNL	EWR	767-301ER	4962
3	3	1113	EWR	LHR	A321	3466
4	4	1114	JFK	LAX	767-301ER	2475
5	5	1115	LAX	JFK	767-301ER	2475
6	6	1116	HNL	LAX	767-301ER	2556
7	7	1117	LAX	ORD	A321	1745
8	8	1118	ORD	EWR	A321	719
9	9	1119	DEN	LAX	ERJ142	862
10	10	1120	HNL	DEN	A321	3365
11	12	1122	ABI	ADK	767-301ER	4300
12	13	1123	ADK	BQN	A321	2232
13	14	1124	BQN	CAK	A321	2445
14	15	1125	CAK	ANI	767-301ER	2000
15	16	1126	ALB	APN	A321	1700
16	17	1127	APN	BLV	767-301ER	1900
17	18	1128	ANI	BGR	ERJ142	2450
18	19	1129	ATW	AVL	A321	2222
19	20	1130	AVL	BOI	767-301ER	3134
20	21	1131	BFL	BET	A321	2425
21	22	1132	BGR	BJI	ERJ142	1242

Total rows: 49 of 49 Query complete 00:00:00.127

Task 3: 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 Query History

```
Task 3:  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.

select passengers_on_flights.customer_id, first_name, last_name, route_id from passengers_on_flights
inner join customer on passengers_on_flights.customer_id= customer.customer_id
where route_id between 1 and 25;
```

Data Output Messages Notifications

	customer_id integer	first_name character varying	last_name character varying	route_id integer
1	1	Julie	Sam	9
2	1	Julie	Sam	9
3	2	Steve	Ryan	4
4	2	Steve	Ryan	4
5	4	Cathenna	Emily	4
6	4	Cathenna	Emily	5
7	4	Cathenna	Emily	4
8	4	Cathenna	Emily	5
9	5	Aaron	Kim	22
10	5	Aaron	Kim	18
11	5	Aaron	Kim	12
12	5	Aaron	Kim	22
13	5	Aaron	Kim	18
14	5	Aaron	Kim	12
15	7	Anderson	Stewart	20
16	7	Anderson	Stewart	20
17	9	Leo	Travis	15
18	9	Leo	Travis	15
19	10	Melvin	Tracy	10
20	10	Melvin	Tracy	10
21	11	Roger	Walson	4
22	11	Roger	Walson	5

Total rows: 52 of 52 Query complete 00:00:00.046

	customer_id integer	first_name character varying	last_name character varying	route_id integer
21	11	Roger	Walson	4
22	11	Roger	Walson	5
23	11	Roger	Walson	4
24	11	Roger	Walson	5
25	13	Solomon	Walter	13
26	13	Solomon	Walter	13
27	15	Linda	William	14
28	15	Linda	William	14
29	17	Catherine	Shad	13
30	17	Catherine	Shad	13
31	18	Gloria	Richie	1
32	18	Gloria	Richie	1
33	22	Pheny	Eri	22
34	22	Pheny	Eri	22
35	24	Calvin	Willis	14
36	24	Calvin	Willis	14
37	25	Moss	Morris	23
38	25	Moss	Morris	23
39	29	Watson	Ronald	9
40	29	Watson	Ronald	9
41	31	James	Robert	20
42	31	James	Robert	20

Total rows: 52 of 52 Query complete 00:00:00.046

42	31	James	Robert	20
43	44	Bily	Brian	15
44	44	Bily	Brian	15
45	46	Louis	Douglas	25
46	46	Louis	Douglas	8
47	46	Louis	Douglas	25
48	46	Louis	Douglas	8
49	49	Russell	Peter	15
50	49	Russell	Peter	15
51	50	Rose	Arthur	21
52	50	Rose	Arthur	21

Total rows: 52 of 52 Query complete 00:00:00.046

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

Query Query History

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


select count(*) as number_of_passengers, sum(price_per_ticket) as total_revenue_in_Business_Class
from tickets_table where class_id='Bussiness';
```

Data Output Messages Notifications

	number_of_passengers bigint	total_revenue_in_business_class bigint
1	13	6034

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

```
/*display the full name of the customer by extracting
the first name and last name from the customer table*/


select first_name || ',' || last_name as full_name from customer;
```

Data Output Messages Notifications

	full_name text
1	Julie,Sam
2	Steve,Ryan
3	Morris,Lois
4	Cathenna,Emily
5	Aaron,Kim
6	Alexander,Scot
7	Anderson,Stew...
8	Floyd,Ted

Total rows: 50 of 50 Query complete 00:00:00.046

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

Air cargo/postgres@PostgreSQL 15

No limit

Query | Query History

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

create table registered_customers_info as select tickets_table.customer_id, first_name, last_name,
p_date, no_of_tickets from customer inner join tickets_table
on tickets_table.customer_id= customer.customer_id;

select * from registered_customers_info;
```

Air cargo/postgres@PostgreSQL 15

No limit

Data Output | Messages | Notifications

	customer_id integer	first_name character varying	last_name character varying	p_date date	no_of_tickets integer
1	1	Julie	Sam	2018-10-01	1
2	1	Julie	Sam	2019-11-23	1
3	2	Steve	Ryan	2018-09-01	1
4	2	Steve	Ryan	2019-01-25	1
5	4	Cathenna	Emily	2020-04-29	1
6	4	Cathenna	Emily	2020-04-04	1
7	5	Aaron	Kim	2020-05-30	1
8	5	Aaron	Kim	2018-07-01	1
9	5	Aaron	Kim	2020-05-05	1
10	7	Anderson	Stewart	2020-07-07	1
11	8	Floyd	Ted	2018-05-01	1
12	8	Floyd	Ted	2020-08-08	1
13	9	Leo	Travis	2018-01-01	1
14	9	Leo	Travis	2020-09-09	1
15	10	Melvin	Tracy	2020-10-10	1
16	11	Roger	Walson	2020-11-08	1
17	11	Roger	Walson	2018-08-01	1
18	11	Roger	Walson	2020-11-11	1
19	13	Solomon	Walter	2019-01-01	1
20	14	Carol	Vernon	2019-12-24	1
21	14	Carol	Vernon	2019-02-02	1
22	15	Linda	William	2018-11-01	1

Total rows: 50 of 50 | Query complete 00:00:00.060

Data Output Messages Notifications

	customer_id integer	first_name character varying	last_name character varying	p_date date	no_of_tickets integer
21	14	Carol	Vernon	2019-02-02	1
22	15	Linda	William	2018-11-01	1
23	16	Chirstine	Willis	2019-04-04	1
24	17	Catherine	Shad	2019-05-03	1
25	18	Gloria	Richie	2018-03-01	1
26	18	Gloria	Richie	2019-06-06	1
27	19	Joyce	Paul	2020-12-13	1
28	19	Joyce	Paul	2018-02-01	1
29	19	Joyce	Paul	2020-12-12	1
30	20	Sara	Oliver	2018-06-01	1
31	20	Sara	Oliver	2019-08-09	1
32	21	Chirsty	Josh	2020-03-03	1
33	22	Pheny	Eri	2020-02-02	1
34	24	Calvin	Willis	2019-07-07	1
35	25	Moss	Morris	2019-09-21	1
36	25	Moss	Morris	2019-03-03	1
37	27	Cherly	Vernon	2018-12-26	1
38	28	Du plesis	Chris	2018-12-01	1
39	29	Watson	Ronald	2018-04-01	1
40	29	Watson	Ronald	2019-10-22	1
41	31	James	Robert	2018-12-19	1
42	32	Chirstoper	Sean	2020-02-04	1

Total rows: 50 of 50 Query complete 00:00:00.060

42	32	Chirstoper	Sean	2020-02-04	1
43	33	Mark	Ethan	2020-03-12	1
44	41	Kyle	Mark	2019-01-11	1
45	44	Bily	Brian	2020-09-05	1
46	46	Louis	Douglas	2019-01-15	1
47	46	Louis	Douglas	2020-10-07	1
48	47	Sophia	Carl	2020-12-09	1
49	49	Russell	Peter	2020-07-17	1
50	50	Rose	Arthur	2020-08-12	1

Total rows: 50 of 50 Query complete 00:00:00.060

Task 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.

Query Query History

```
/* identify the customer's first name and last name based on their
customer ID and brand (Emirates) from the ticket_details table.*/

select first_name, last_name, tickets_table.customer_id from customer
right join tickets_table on customer.customer_id=tickets_table.customer_id
where brand='Emirates';
```

Data Output Messages Notifications

	first_name character varying	last_name character varying	customer_id integer
1	Steve	Ryan	2
2	Cathenna	Emily	4
3	Cathenna	Emily	4
4	Aaron	Kim	5
5	Anderson	Stewart	7
6	Leo	Travis	9
7	Roger	Walson	11
8	Roger	Walson	11
9	Carol	Vernon	14
10	Gloria	Richie	18
11	Gloria	Richie	18
12	Joyce	Paul	19
13	Moss	Morris	25
14	Moss	Morris	25

Total rows: 18 of 18 Query complete 00:00:00.193

Data Output Messages Notifications

	first_name character varying	last_name character varying	customer_id integer
5	Anderson	Stewart	7
6	Leo	Travis	9
7	Roger	Walson	11
8	Roger	Walson	11
9	Carol	Vernon	14
10	Gloria	Richie	18
11	Gloria	Richie	18
12	Joyce	Paul	19
13	Moss	Morris	25
14	Moss	Morris	25
15	Cherly	Vernon	27
16	James	Robert	31
17	Bily	Brian	44
18	Russell	Peter	49

Total rows: 18 of 18 Query complete 00:00:00.193

Task 8: 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.

Air cargo/postgres@PostgreSQL 15

No limit

Query Query History

```
/*identify the customers who have travelled by Economy Plus class using Group By
and Having clause on the passengers_on_flights table*/


select class_id , customer_id
from passengers_on_flights group by class_id, customer_id
having class_id='Economy Plus';
```

Data Output Messages Notifications

	class_id character varying	customer_id integer
1	Economy Plus	19
2	Economy Plus	8
3	Economy Plus	50
4	Economy Plus	22
5	Economy Plus	1
6	Economy Plus	47
7	Economy Plus	17
8	Economy Plus	32
9	Economy Plus	11

Total rows: 9 of 9 Query complete 00:00:00.088

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

Air cargo/postgres@PostgreSQL 15

No limit

Query Query History

```
/* query to identify whether the revenue has crossed 10000 using the
IF clause on the ticket_details table*/


DO $$
BEGIN
  IF sum(price_per_ticket)> 10000 from tickets_table THEN
    RAISE NOTICE 'TRUE';
 ELSE
     RAISE NOTICE 'False';
 END IF;
END $$;
```

Data Output Messages Notifications

```
NOTICE:  TRUE
DO

Query returned successfully in 32 msec.
```

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

Query Query History

```
/*create and grant access to a new user to perform operations on a database*/

create user Calvin Password '1111'
grant all on customer to Calvin;

```

Task 11: Write a query to find the maximum ticket price for each class using window functions on the ticket_details table.

Air cargo/postgres@PostgreSQL 15

No limit

Query Query History

```
/*find the maximum ticket price for each class using
window functions on the ticket_details table.*/

select price_per_ticket,class_id, max(price_per_ticket) over (partition by class_id)
from tickets_table;
```

Data Output Messages Notifications

	price_per_ticket integer	class_id character varying	max integer
1	499	Bussiness	510
2	505	Bussiness	510
3	410	Bussiness	510
4	465	Bussiness	510
5	480	Bussiness	510
6	430	Bussiness	510
7	490	Bussiness	510
8	430	Bussiness	510
9	490	Bussiness	510
10	465	Bussiness	510
11	430	Bussiness	510
12	510	Bussiness	510
13	430	Bussiness	510
14	130	Economy	190
15	120	Economy	190

Total rows: 50 of 50 Query complete 00:00:00.045

Air cargo/postgres@PostgreSQL 15

No limit

Data Output Messages Notifications

	price_per_ticket integer	class_id character varying	max integer
16	135	Economy	190
17	120	Economy	190
18	190	Economy	190
19	150	Economy	190
20	170	Economy	190
21	100	Economy	190
22	190	Economy	190
23	130	Economy	190
24	170	Economy	190
25	130	Economy	190
26	120	Economy	190
27	135	Economy	190
28	225	Economy Plus	295
29	295	Economy Plus	295
30	225	Economy Plus	295
31	250	Economy Plus	295
32	250	Economy Plus	295
33	275	Economy Plus	295
34	220	Economy Plus	295
35	220	Economy Plus	295
36	275	Economy Plus	295

Total rows: 50 of 50 Query complete 00:00:00.045

Data Output Messages Notifications

	price_per_ticket integer	class_id character varying	max integer
29	295	Economy Plus	295
30	225	Economy Plus	295
31	250	Economy Plus	295
32	250	Economy Plus	295
33	275	Economy Plus	295
34	220	Economy Plus	295
35	220	Economy Plus	295
36	275	Economy Plus	295
37	225	Economy Plus	295
38	390	First Class	395
39	380	First Class	395
40	395	First Class	395
41	375	First Class	395
42	395	First Class	395
43	315	First Class	395
44	320	First Class	395
45	380	First Class	395
46	395	First Class	395
47	390	First Class	395
48	365	First Class	395
49	390	First Class	395
50	395	First Class	395

Total rows: 50 of 50 Query complete 00:00:00.045

Task 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.

Air cargo/postgres@PostgreSQL 15

No limit

Query Query History

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

create index index_passengers_on_flights on passengers_on_flights(route_id);

explain analyse select * from passengers_on_flights where route_id=4;
```

Data Output Messages Notifications

	QUERY PLAN text
1	Seq Scan on passengers_on_flights (cost=0.00..2.25 rows=6 width=45) (actual time=0.016..0.022 rows=6 loops...
2	Filter: (route_id = 4)
3	Rows Removed by Filter: 94
4	Planning Time: 0.087 ms
5	Execution Time: 0.033 ms

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

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

select * from passengers_on_flights where route_id='4';

EXPLAIN ANALYZE select * from passengers_on_flights where route_id='4';
```

Data Output Messages Notifications

	QUERY PLAN text
1	Seq Scan on passengers_on_flights (cost=0.00..2.25 rows=6 width=45) (actual time=0.022..0.029 rows=6 loops=1)
2	Filter: (route_id = 4)
3	Rows Removed by Filter: 94
4	Planning Time: 0.129 ms
5	Execution Time: 0.061 ms

Total rows: 5 of 5 Query complete 00:00:00.071

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

Air cargo/postgres@PostgreSQL 15

No limit

Query Query History

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

SELECT aircraft_id,SUM(price_per_ticket)
FROM tickets_table
GROUP BY
    ROLLUP(aircraft_id)
    order by(aircraft_id);
```

Data Output Messages Notifications

	aircraft_id character varying	sum bigint
1	767-301ER	5634
2	A321	4270
3	CRJ900	3440
4	ERJ142	2025
5	[null]	15369

Task 15: Write a query to create a view with only business class customers along with the brand of airlines.

Query Query History

```
/* Task 15: Write a query to create a view with only business class
customers along with the brand of airlines.*/

create view business_class_customers_view as
select customer_id, brand from tickets_table where class_id='Bussiness';
```

Data Output Messages Notifications

	customer_id integer	brand character varying
1	21	Bristish Airways
2	7	Emirates
3	11	Emirates
4	25	Emirates
5	24	Qatar Airways
6	29	Qatar Airways
7	2	Qatar Airways
8	29	Jet Airways
9	5	Emirates
10	15	Qatar Airways
11	33	Bristish Airways
12	49	Emirates
13	11	Emirates

Task 16: Write a query to get the details of all passengers flying between a range of routes defined in run time.

```
/* Task 16:  Write a query to get the details of all passengers flying between a
range of routes defined in run time.*/


select * from passengers_on_flights where route_id between 1 and 10;
```

Data Output Messages Notifications

	customer_id integer	aircraft_id character varying	route_id integer	depart character varying	arrival character varying	seat_num character varying	class_id character varying	travel_date date	flight_num integer
1	2	767-301ER	4	JFK	LAX	01E	Economy	2018-09-02	1114
2	1	ERJ142	9	DEN	LAX	01EP	Economy Plus	2019-12-26	1119
3	4	767-301ER	5	LAX	JFX	02FC	First Class	2020-04-06	1115
4	4	767-301ER	4	JFK	LAX	03FC	First Class	2020-04-30	1114
5	11	767-301ER	5	LAX	JFX	04B	Bussiness	2020-11-12	1115
6	11	767-301ER	4	JFK	LAX	05B	Bussiness	2020-11-09	1114
7	10	A321	10	HNL	DEN	05E	Economy	2020-10-11	1120
8	29	ERJ142	9	DEN	LAX	11B	Bussiness	2018-05-03	1119
9	46	A321	8	ORD	EWR	12FC	First Class	2011-07-08	1118
10	18	767-301ER	1	EWR	HNL	13FC	First Class	2018-04-01	1111
11	2	767-301ER	4	JFK	LAX	01E	Economy	2018-09-02	1114
12	1	ERJ142	9	DEN	LAX	01EP	Economy Plus	2019-12-26	1119
13	4	767-301ER	5	LAX	JFX	02FC	First Class	2020-04-06	1115
14	4	767-301ER	4	JFK	LAX	03FC	First Class	2020-04-30	1114
15	11	767-301ER	5	LAX	JFX	04B	Bussiness	2020-11-12	1115
16	11	767-301ER	4	JFK	LAX	05B	Bussiness	2020-11-09	1114
17	10	A321	10	HNL	DEN	05E	Economy	2020-10-11	1120
18	29	ERJ142	9	DEN	LAX	11B	Bussiness	2018-05-03	1119
19	46	A321	8	ORD	EWR	12FC	First Class	2011-07-08	1118
20	18	767-301ER	1	EWR	HNL	13FC	First Class	2018-04-01	1111

Total rows: 20 of 20 Query complete 00:00:00.048 Ln 39, Col 1

Task 17: Write a query to extract all the details from the routes table where the traveled distance is more than 2000 miles.

```
/*Task 17: Write a query to extract all the details
from the routes table where the travelled distance is more than 2000 miles*/


select * from route_details where distance_miles > 2000;
```

Data Output Messages Notifications

	route_id integer	flight_num integer	origin_airport character varying	destination_airport character varying	aircraft_id character varying	distance_miles integer
1	1	1111	EWR	HNL	767-301ER	4962
2	2	1112	HNL	EWR	767-301ER	4962
3	3	1113	EWR	LHR	A321	3466
4	4	1114	JFK	LAX	767-301ER	2475
5	5	1115	LAX	JFK	767-301ER	2475
6	6	1116	HNL	LAX	767-301ER	2556
7	10	1120	HNL	DEN	A321	3365
8	12	1122	ABI	ADK	767-301ER	4300
9	13	1123	ADK	BQN	A321	2232
10	14	1124	BQN	CAK	A321	2445
11	18	1128	ANI	BGR	ERJ142	2450
12	19	1129	ATW	AVL	A321	2222
13	20	1130	AVL	BOI	767-301ER	3134
14	21	1131	BFL	BET	A321	2425
15	23	1133	BLV	BFL	767-301ER	2354
16	25	1135	RDM	BJI	A321	2425
17	34	1144	CRW	COD	A321	2452
18	35	1145	STT	CDB	ERJ142	2121
19	43	1153	CBM	BOI	A321	8989
20	44	1154	COU	CAK	767-301ER	7676
21	46	1156	CDV	HNL	767-301ER	8668
22	48	1158	SCC	DEN	A321	5645

Total rows: 24 of 24 Query complete 00:00:00.034

22	48	1158	SCC	DEN	A321	5645
23	49	1159	DEC	ABI	A321	4533
24	50	1160	DRT	ORD	A321	2445

Total rows: 24 of 24 Query complete 00:00:00.034

Task 18: Write a query to create groups for the distance traveled 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.

Air cargo/postgres@PostgreSQL 15

No limit

Query | Query History

```
/* Write a query to create groups for 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*/


select flight_num, distance_miles,
case
when distance_miles between 0 and 2000 then 'short_distance_travel'
when distance_miles between 2001 and 6500 then 'intermediate_distance_travel'
when distance_miles > 6500 then 'long_distance_travel'
End
as distance_types
from route_details;
```

Data Output | Messages | Notifications

	flight_num integer	distance_miles integer	distance_types text
1	1111	4962	intermediate_distance_travel
2	1112	4962	intermediate_distance_travel
3	1113	3466	intermediate_distance_travel
4	1114	2475	intermediate_distance_travel
5	1115	2475	intermediate_distance_travel
6	1116	2556	intermediate_distance_travel
7	1117	1745	short_distance_travel
8	1118	719	short_distance_travel
9	1119	862	short_distance_travel
10	1120	3365	intermediate_distance_travel
11	1122	4300	intermediate_distance_travel
12	1123	2232	intermediate_distance_travel
13	1124	2445	intermediate_distance_travel
14	1125	2000	short_distance_travel
15	1126	1700	short_distance_travel
16	1127	1900	short_distance_travel
17	1128	2450	intermediate_distance_travel
18	1129	2222	intermediate_distance_travel
19	1130	3134	intermediate_distance_travel
20	1131	2425	intermediate_distance_travel
21	1132	1242	short_distance_travel
22	1133	2354	intermediate_distance_travel

Total rows: 49 of 49 Query complete 00:00:00.041

Data Output | Messages | Notifications

	flight_num integer	distance_miles integer	distance_types text
22	1133	2354	intermediate_distance_travel
23	1134	1575	short_distance_travel
24	1135	2425	intermediate_distance_travel
25	1136	1311	short_distance_travel
26	1137	578	short_distance_travel
27	1138	246	short_distance_travel
28	1139	909	short_distance_travel
29	1140	780	short_distance_travel
30	1141	660	short_distance_travel
31	1142	246	short_distance_travel
32	1143	1345	short_distance_travel
33	1144	2452	intermediate_distance_travel
34	1145	2121	intermediate_distance_travel
35	1146	1212	short_distance_travel
36	1147	999	short_distance_travel
37	1148	1111	short_distance_travel
38	1149	1579	short_distance_travel
39	1150	909	short_distance_travel
40	1151	898	short_distance_travel
41	1152	890	short_distance_travel
42	1153	8989	long_distance_travel
43	1154	7676	long_distance_travel
44	1155	676	short_distance_travel
45	1156	8668	long_distance_travel
46	1157	675	short_distance_travel
47	1158	5645	intermediate_distance_travel
48	1159	4533	intermediate_distance_travel
49	1160	2445	intermediate_distance_travel

Total rows: 49 of 49 Query complete 00:00:00.041

Task 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 on the ticket_details table. If the class is *Business* and *Economy Plus,* then complimentary services are given as *Yes,* else it is *No*

Air cargo/postgres@PostgreSQL 15

No limit

Query | Query History

```
/* Write a query to extract ticket purchase date, customer ID, class ID and specify if
the complimentary services are provided for the specific class on the ticket_details table.
If the class is Business and Economy Plus, then complimentary services are given as Yes,
else it is No*/


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 tickets_table;
```

Air cargo/postgres@PostgreSQL 15

No limit

Data Output | Messages | Notifications

	p_date date	customer_id integer	class_id character varying	complimentary_services text
1	2018-12-26	27	Economy	no
2	2020-02-02	22	Economy Plus	yes
3	2020-03-03	21	Bussiness	yes
4	2020-04-04	4	First Class	no
5	2020-05-05	5	Economy	no
6	2020-07-07	7	Bussiness	yes
7	2020-08-08	8	Economy Plus	yes
8	2020-09-09	9	First Class	no
9	2020-10-10	10	Economy	no
10	2020-11-11	11	Bussiness	yes
11	2020-12-12	19	Economy Plus	yes
12	2019-01-01	13	First Class	no
13	2019-02-02	14	Economy	no
14	2019-03-03	25	Bussiness	yes
15	2019-04-04	16	First Class	no
16	2019-05-03	17	Economy Plus	yes
17	2019-06-06	18	Economy	no
18	2019-07-07	24	Bussiness	yes
19	2019-08-09	20	First Class	no
20	2019-09-21	25	Economy	no
21	2019-10-22	29	Bussiness	yes

Total rows: 50 of 50 | Query complete 00:00:00.037

Data Output | Messages | Notifications

	p_date date	customer_id integer	class_id character varying	complimentary_services text
21	2019-10-22	29	Bussiness	yes
22	2019-11-23	1	Economy Plus	yes
23	2019-12-24	14	Economy	no
24	2019-01-25	2	Bussiness	yes
25	2018-01-01	9	First Class	no
26	2018-02-01	19	Economy	no
27	2018-03-01	18	First Class	no
28	2018-04-01	29	Bussiness	yes
29	2018-05-01	8	Economy	no
30	2018-06-01	20	First Class	no
31	2018-07-01	5	Bussiness	yes
32	2018-08-01	11	Economy Plus	yes
33	2018-09-01	2	Economy	no
34	2018-10-01	1	First Class	no
35	2018-11-01	15	Bussiness	yes
36	2018-12-01	28	Economy	no
37	2018-12-19	31	Economy	no
38	2020-02-04	32	Economy Plus	yes
39	2020-03-12	33	Bussiness	yes
40	2020-04-29	4	First Class	no
41	2020-05-30	5	Economy	no

Total rows: 50 of 50 | Query complete 00:00:00.037

41	2020-05-30	5	Economy	no
42	2020-07-17	49	Bussiness	yes
43	2020-08-12	50	Economy Plus	yes
44	2020-09-05	44	First Class	no
45	2020-10-07	46	Economy	no
46	2020-11-08	11	Bussiness	yes
47	2020-12-09	47	Economy Plus	yes
48	2019-01-11	41	First Class	no
49	2020-12-13	19	Economy Plus	yes
50	2019-01-15	46	First Class	no
Total rows: 50 of 50	Query complete 00:00:00.037			

Task: 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.

Air cargo/postgres@PostgreSQL 15

No limit

Query Query History

```
/* Task 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*/


begin;

declare
my_cursor cursor for select * from customer where last_name='Scott' limit 1;

fetch from my_cursor;
```

Data Output Messages Notifications

	customer_id integer	first_name character varying	last_name character varying	date_of_birth date	gender character varying
1	37	Samuel	Scott	2000-01-28	M