SQL Project

Submitted by: Mohit Rohilla

Air Cargo Analysis

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.

Note: You must download the dataset from the course resource section in the LMS and create the tables to perform the above objective.

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

Following operations should be performed:

1. Create an ER diagram for the given airlines database.
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.
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.
4. Write a query to identify the number of passengers and total revenue in business class from the ticket_details table.
5. Write a query to display the full name of the customer by extracting the first name and last name from the customer table.
6. Write a query to extract the customers who have registered and booked a ticket. Use data from the customer and ticket_details tables.
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.
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.
9. Write a query to identify whether the revenue has crossed 10000 using the IF clause on the ticket_details table.
10. Write a query to create and grant access to a new user to perform operations on a database.
11. Write a query to find the maximum ticket price for each class using window functions on the ticket_details table.
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.
13. For the route ID 4, write a query to view the execution plan of the passengers_on_flights table.
14. Write a query to calculate the total price of all tickets booked by a customer across different aircraft IDs using rollup function.
15. Write a query to create a view with only business class customers along with the brand of airlines.

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.
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.
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.
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*

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.

Solution:

A new database air_cargo_analysis is created and tables have been imported using import table wizard:

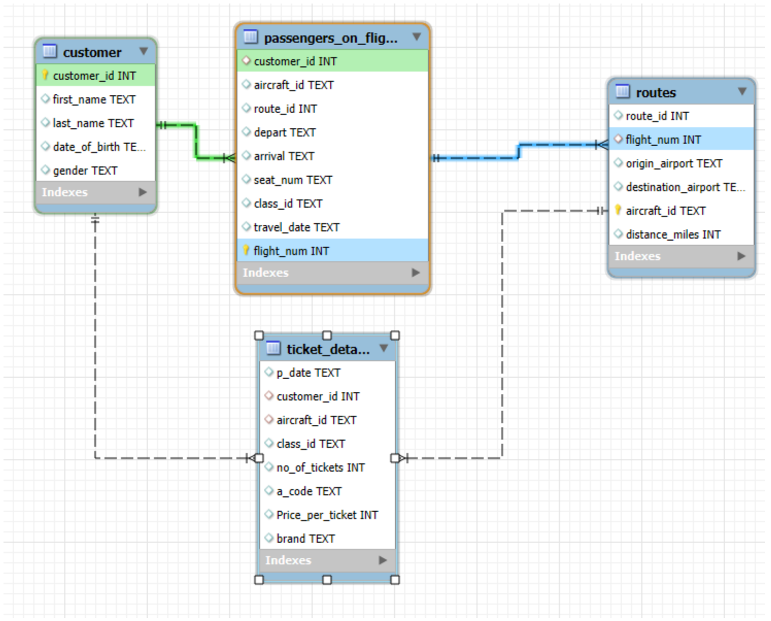
Create database air_cargo_analysis;

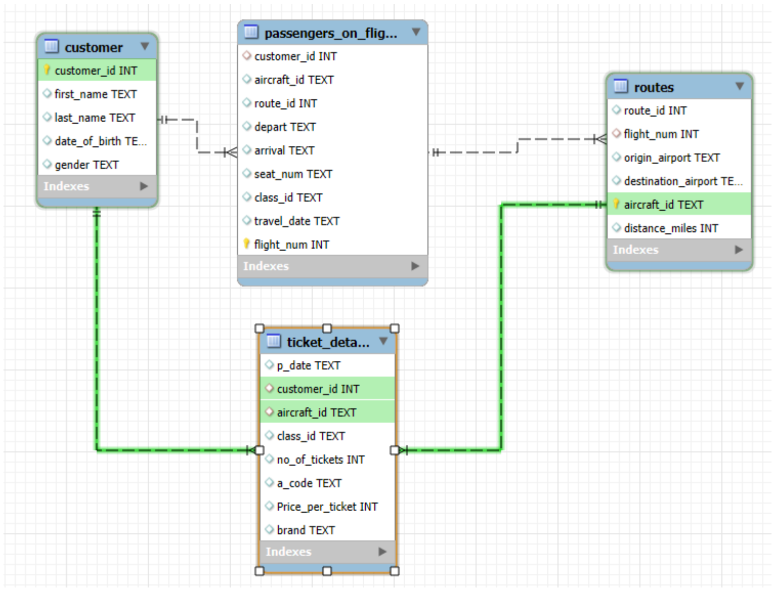
Task 1: Create an ER diagram for the given airlines database.

An ER diagram has been prepared by considering:

- ➢ Customer_id INT as primary key in Customer Table.
- ➢ Flight_num INT as primary key in passanger_on_flight table.
- ➢ Aircraft_id TEXT as primary key in routes table.

The other variations were also possible.

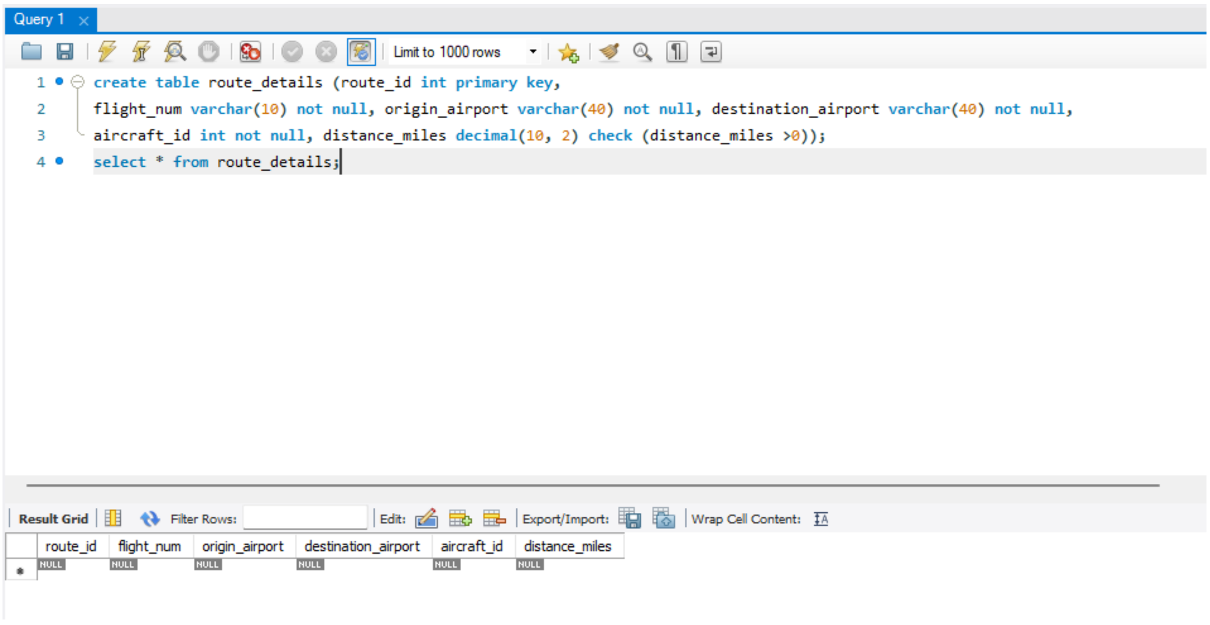




Task 2: Write a query to create route_details table using suitable data types for the fields, such as route_id, flight_num, origin_airport, destination_airport, aircraft_id, and distance_miles. Implement the check constraint for the flight number and unique constraint for the route_id fields. Also, make sure that the distance miles field is greater than 0.

SQL query:

create table route_details (route_id int primary key,
flight_num varchar(10) not null, origin_airport varchar(40) not null, destination_airport varchar(40) not null,
aircraft_id int not null, distance_miles decimal(10, 2) check (distance_miles >0));
select * from route_details;



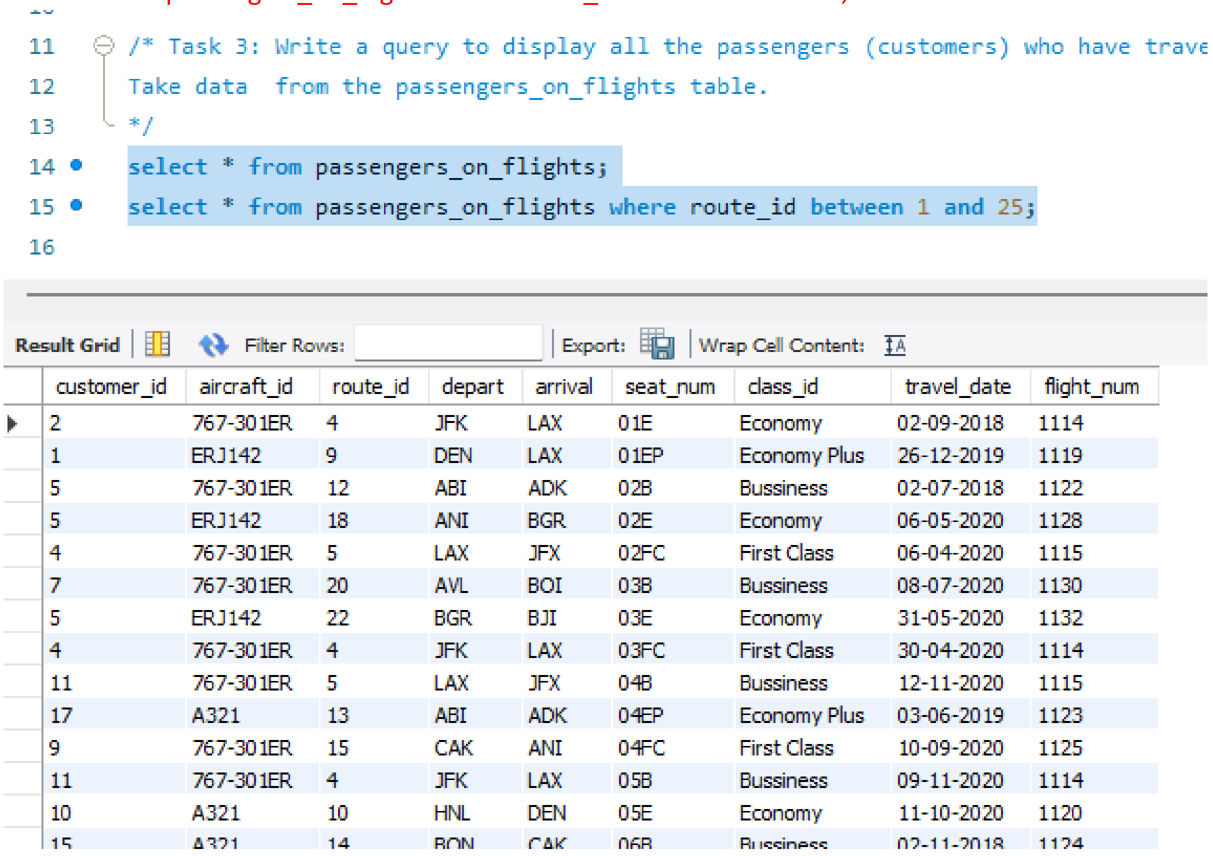
```
create table route_details (route_id int primary key,
flight_num varchar(10) not null, origin_airport varchar(40) not null, destination_airport varchar(40) not null,
aircraft_id int not null, distance_miles decimal(10, 2) check (distance_miles >0));
select * from route_details;
```

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

Task 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.

SQL query:

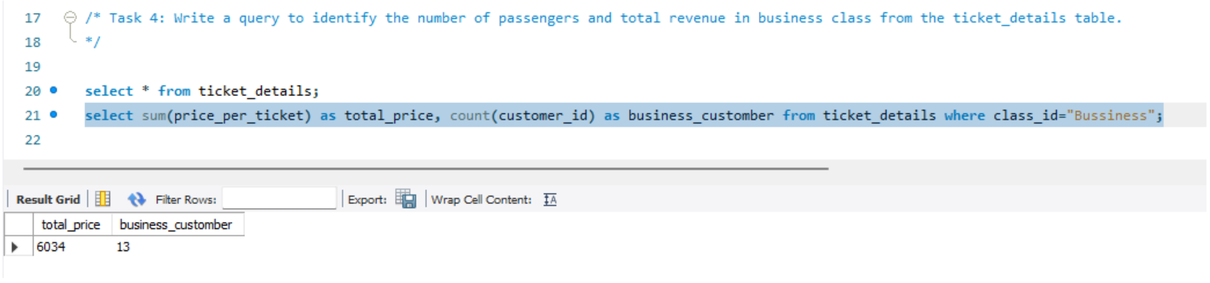
select * from passengers_on_flights where route_id between 1 and 25;



```
/* Task 3: Write a query to display all the passengers (customers) who have trave
Take data  from the passengers_on_flights table.
*/
select * from passengers_on_flights;
select * from passengers_on_flights where route_id between 1 and 25;
```

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	BON	CAK	06B	Bussiness	02-11-2018	1124

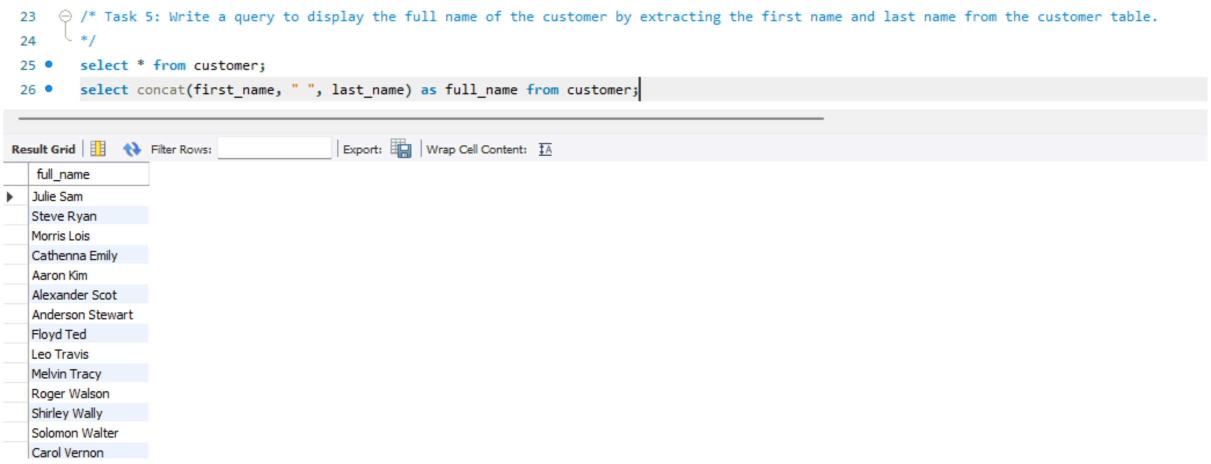
Task 4: Write a query to identify the number of passengers and total revenue in business class from the ticket_details table.
SQL Query:
select sum(price_per_ticket) as total_price, count(customer_id) as business_customber from ticket_details where class_id="Bussiness";



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

select * from ticket_details;
select sum(price_per_ticket) as total_price, count(customer_id) as business_customber from ticket_details where class_id="Bussiness";

```

total_price	business_customber
6034	13

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.
SQL query:
select concat(first_name, " ", last_name) as full_name from customer;

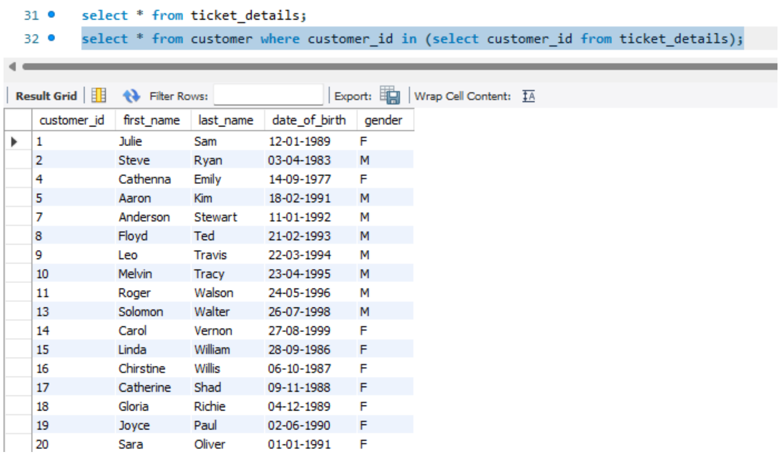


```
/* Task 5: Write a query to display the full name of the customer by extracting the first name and last name from the customer table.
*/
select * from customer;
select concat(first_name, " ", last_name) as full_name from customer;
```

full_name
Julie Sam
Steve Ryan
Morris Lois
Cathenna Emily
Aaron Kim
Alexander Scot
Anderson Stewart
Floyd Ted
Leo Travis
Melvin Tracy
Roger Walson
Shirley Wally
Solomon Walter
Carol Vernon

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.

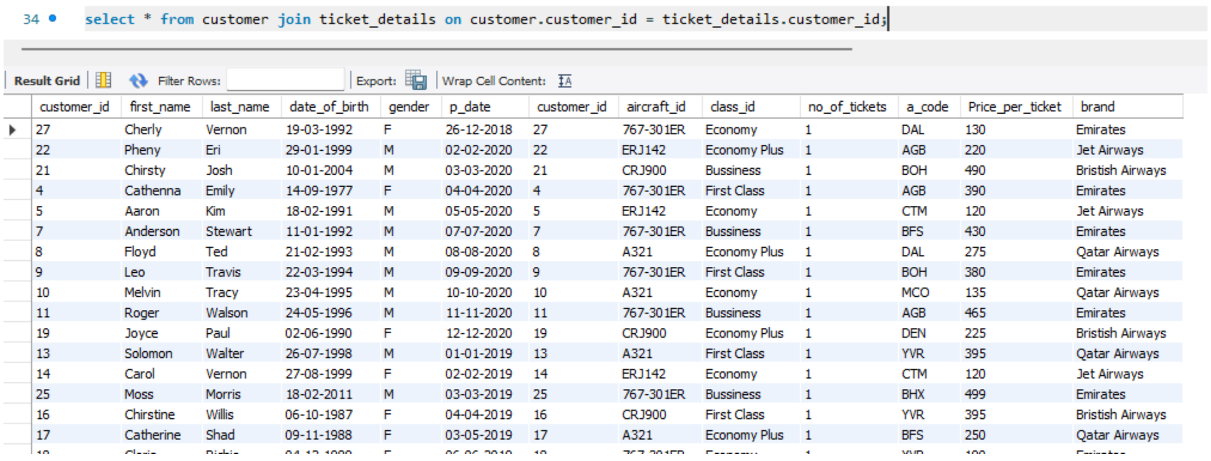
SQL query:
select * from customer where customer_id in (select customer_id from ticket_details);



```
select * from ticket_details;
select * from customer where customer_id in (select customer_id from ticket_details);
```

customer_id	first_name	last_name	date_of_birth	gender
1	Julie	Sam	12-01-1989	F
2	Steve	Ryan	03-04-1983	M
4	Cathenna	Emily	14-09-1977	F
5	Aaron	Kim	18-02-1991	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
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

The other query can be from the join function:
select * from customer where customer_id in (select customer_id from ticket_details);

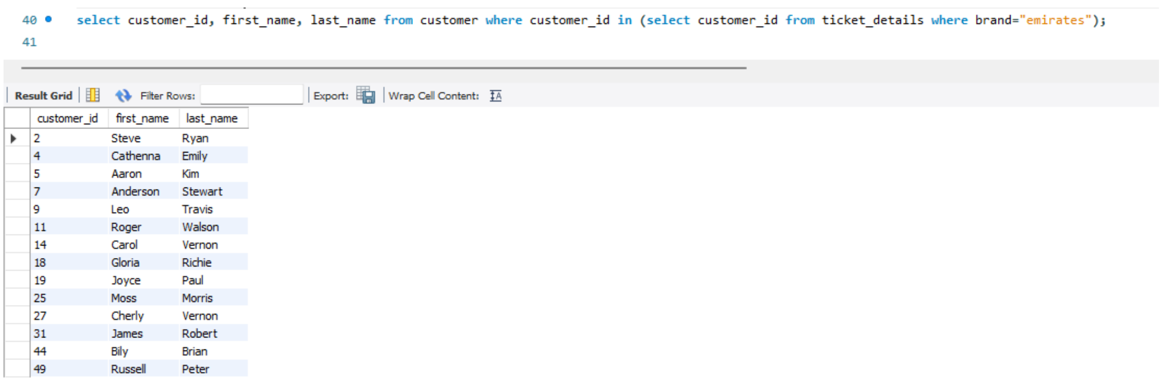


```
select * from customer join ticket_details on customer.customer_id = ticket_details.customer_id;
```

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

customer_id	first_name	last_name	date_of_birth	gender	p_date	customer_id	aircraft_id	class_id	no_of_tickets	a_code	Price_per_ticket	brand
27	Cherly	Vernon	19-03-1992	F	26-12-2018	27	767-301ER	Economy	1	DAL	130	Emirates
22	Pheny	Eri	29-01-1999	M	02-02-2020	22	ERJ142	Economy Plus	1	AGB	220	Jet Airways
21	Chirsty	Josh	10-01-2004	M	03-03-2020	21	CRJ900	Bussiness	1	BOH	490	Bristish Airways
4	Cathenna	Emily	14-09-1977	F	04-04-2020	4	767-301ER	First Class	1	AGB	390	Emirates
5	Aaron	Kim	18-02-1991	M	05-05-2020	5	ERJ142	Economy	1	CTM	120	Jet Airways
7	Anderson	Stewart	11-01-1992	M	07-07-2020	7	767-301ER	Bussiness	1	BFS	430	Emirates
8	Floyd	Ted	21-02-1993	M	08-08-2020	8	A321	Economy Plus	1	DAL	275	Qatar Airways
9	Leo	Travis	22-03-1994	M	09-09-2020	9	767-301ER	First Class	1	BOH	380	Emirates
10	Melvin	Tracy	23-04-1995	M	10-10-2020	10	A321	Economy	1	MCO	135	Qatar Airways
11	Roger	Walson	24-05-1996	M	11-11-2020	11	767-301ER	Bussiness	1	AGB	465	Emirates
19	Joyce	Paul	02-06-1990	F	12-12-2020	19	CRJ900	Economy Plus	1	DEN	225	Bristish Airways
13	Solomon	Walter	26-07-1998	M	01-01-2019	13	A321	First Class	1	YVR	395	Qatar Airways
14	Carol	Vernon	27-08-1999	F	02-02-2019	14	ERJ142	Economy	1	CTM	120	Jet Airways
25	Moss	Morris	18-02-2011	M	03-03-2019	25	767-301ER	Bussiness	1	BHX	499	Emirates
16	Chirstine	Willis	06-10-1987	F	04-04-2019	16	CRJ900	First Class	1	YVR	395	Bristish Airways
17	Catherine	Shad	09-11-1988	F	03-05-2019	17	A321	Economy Plus	1	BFS	250	Qatar Airways

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.
SQL query:
select customer_id, first_name, last_name from customer where customer_id in (select customer_id from ticket_details where brand="emirates");

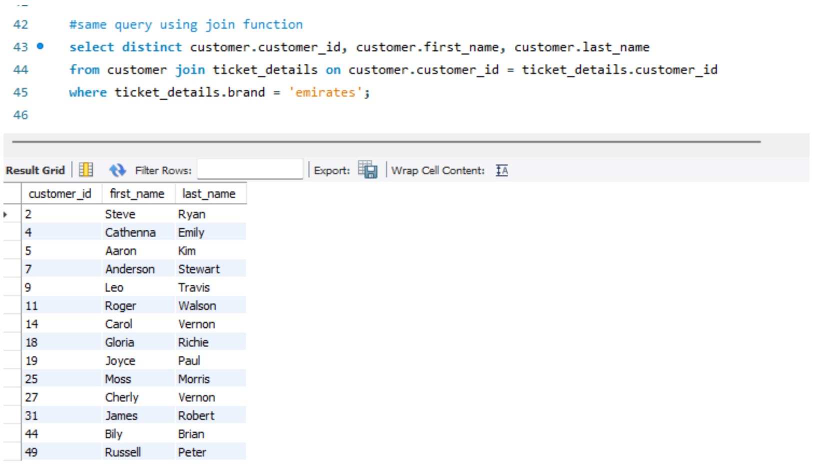


```
select customer_id, first_name, last_name from customer where customer_id in (select customer_id from ticket_details where brand="emirates");
```

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

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

Same problem can be solved by using join function:
SQL query

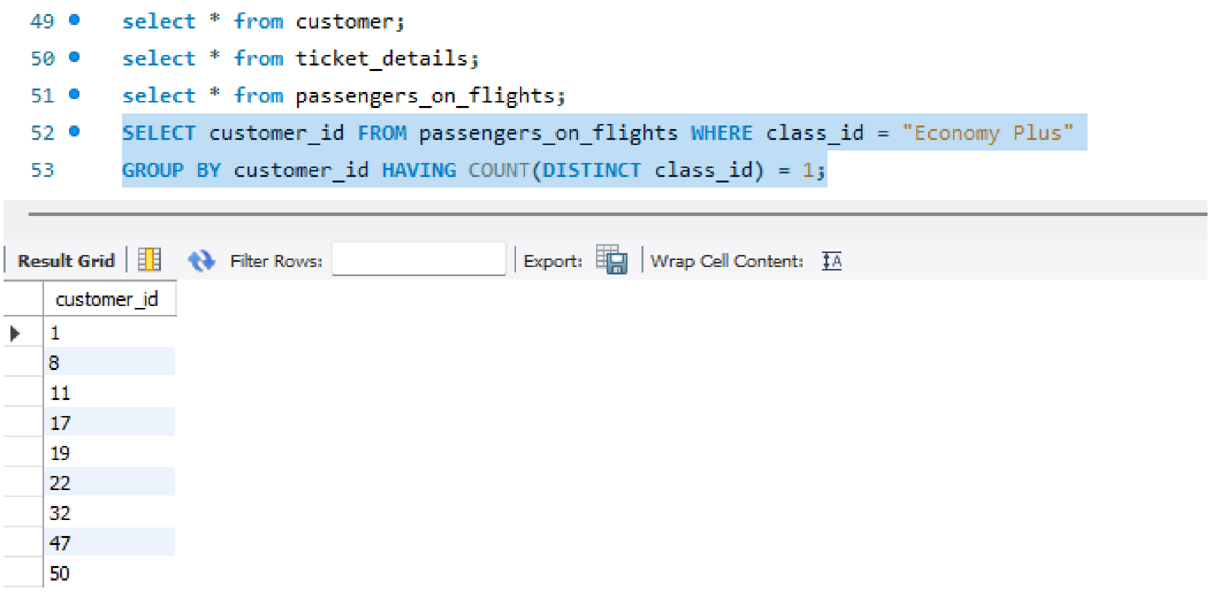


```
#same query using join function
select distinct customer.customer_id, customer.first_name, customer.last_name
from customer join ticket_details on customer.customer_id = ticket_details.customer_id
where ticket_details.brand = 'emirates';
```

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

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

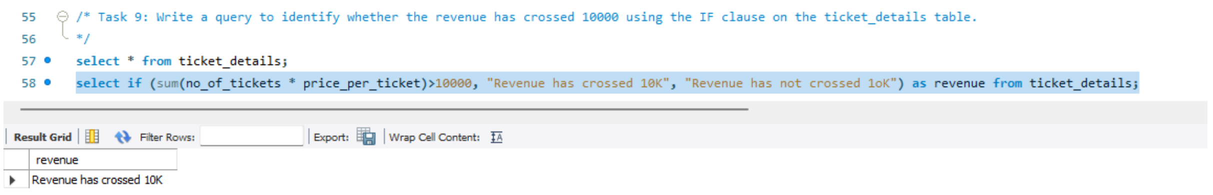
Task 8: Write a query to identify the customers who have travelled by *Economy Plus* class using Group By and Having clause on the passengers_on_flights table.
SQL query:
select customer_id from passengers_on_flights where class_id = "economy plus"
group by customer_id having count(distinct class_id) = 1;



```
select * from customer;
select * from ticket_details;
select * from passengers_on_flights;
SELECT customer_id FROM passengers_on_flights WHERE class_id = "Economy Plus"
GROUP BY customer_id HAVING COUNT(DISTINCT class_id) = 1;
```

```
customer_id
1
8
11
17
19
22
32
47
50
```

Task 9: Write a query to identify whether the revenue has crossed 10000 using the IF clause on the ticket_details table.
SQL query:
select if (sum(no_of_tickets * price_per_ticket)>10000, "Revenue has crossed 10K", "Revenue has not crossed 1oK") as revenue from ticket_details;

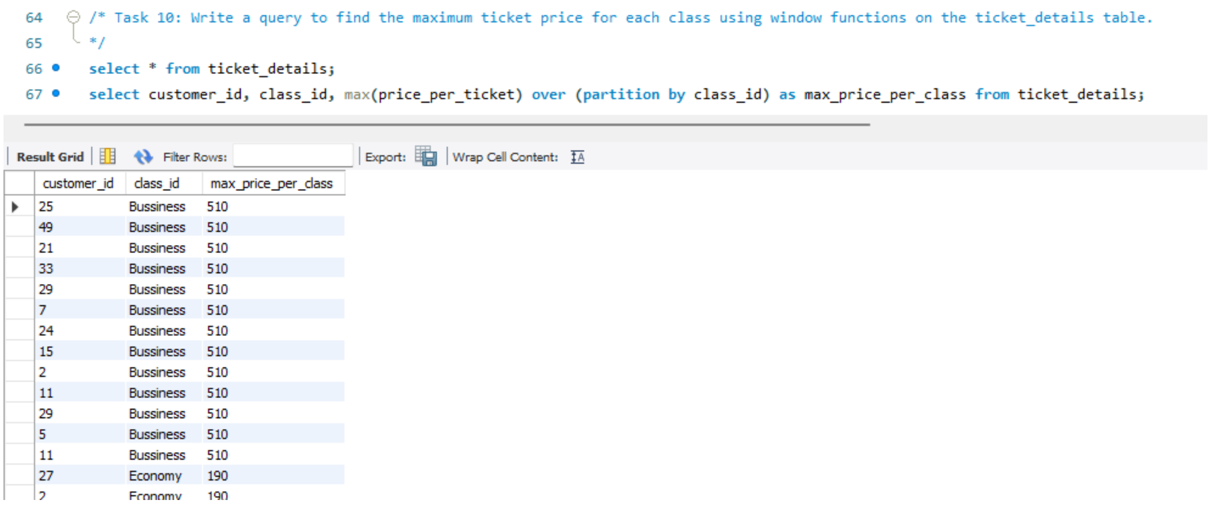


```
/* Task 9: Write a query to identify whether the revenue has crossed 10000 using the IF clause on the ticket_details table.
*/
select * from ticket_details;
select if (sum(no_of_tickets * price_per_ticket)>10000, "Revenue has crossed 10K", "Revenue has not crossed 1oK") as revenue from ticket_details;
```

```
revenue
Revenue has crossed 10K
```

Task 10: Write a query to create and grant access to a new user to perform operations on a database.
SQL query:
CREATE USER 'new_user'@'localhost' IDENTIFIED BY 'password';

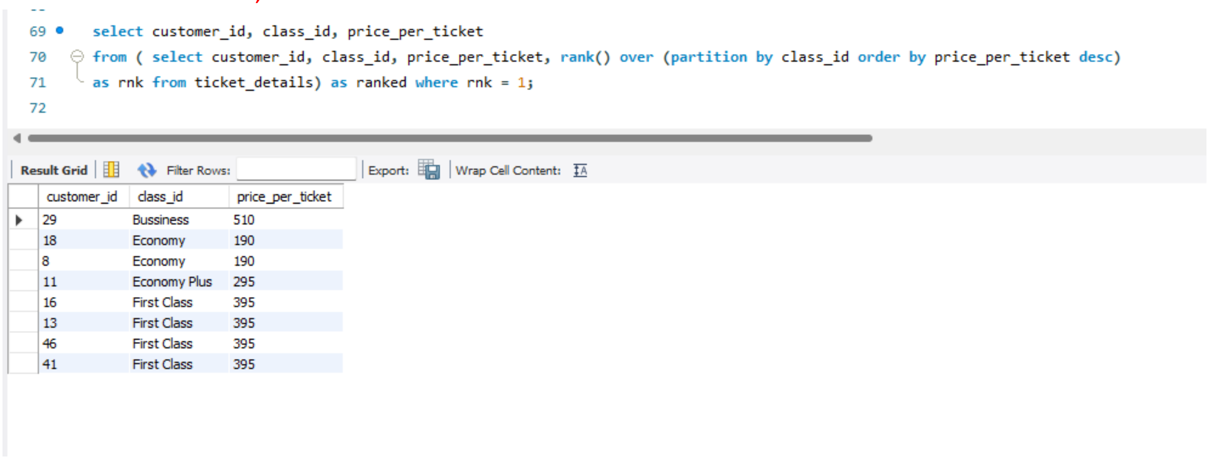
Task 11. Write a query to find the maximum ticket price for each class using window functions on the ticket_details table.
SQL query:
select customer_id, class_id, max(price_per_ticket) over (partition by class_id) as max_price_per_class from ticket_details;



```
/* Task 10: Write a query to find the maximum ticket price for each class using window functions on the ticket_details table.
*/
select * from ticket_details;
select customer_id, class_id, max(price_per_ticket) over (partition by class_id) as max_price_per_class from ticket_details;
```

customer_id	class_id	max_price_per_class
25	Bussiness	510
49	Bussiness	510
21	Bussiness	510
33	Bussiness	510
29	Bussiness	510
7	Bussiness	510
24	Bussiness	510
15	Bussiness	510
2	Bussiness	510
11	Bussiness	510
29	Bussiness	510
5	Bussiness	510
11	Bussiness	510
27	Economy	190
2	Economy	190

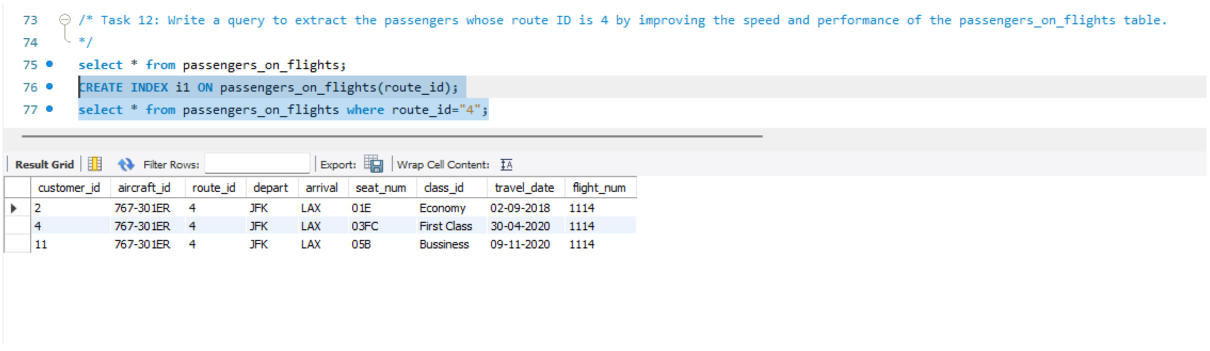
Other way
select customer_id, class_id, price_per_ticket
from (select customer_id, class_id, price_per_ticket,
rank() over (partition by class_id order by price_per_ticket desc) as rnk from ticket_details) as ranked where rnk = 1;



```
select customer_id, class_id, price_per_ticket
from ( select customer_id, class_id, price_per_ticket, rank() over (partition by class_id order by price_per_ticket desc)
as rnk from ticket_details) as ranked where rnk = 1;
```

customer_id	class_id	price_per_ticket
29	Bussiness	510
18	Economy	190
8	Economy	190
11	Economy Plus	295
16	First Class	395
13	First Class	395
46	First Class	395
41	First Class	395

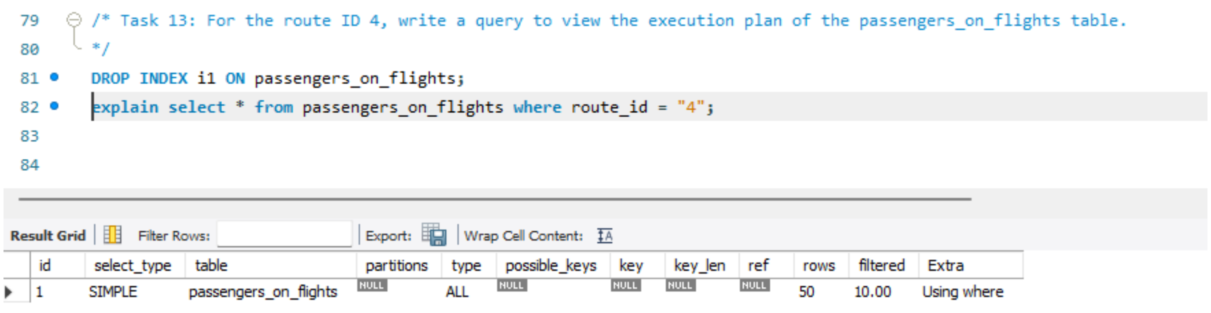
Task 12: Write a query to extract the passengers whose route ID is 4 by improving the speed and performance of the passengers_on_flights table.
SQL query: to solve this query, we have to make an index on the route id
CREATE INDEX i1 ON passengers_on_flights(route_id);
select * from passengers_on_flights where route_id="4";



```
/* 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.
*/
select * from passengers_on_flights;
CREATE INDEX i1 ON passengers_on_flights(route_id);
select * from passengers_on_flights where route_id="4";
```

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

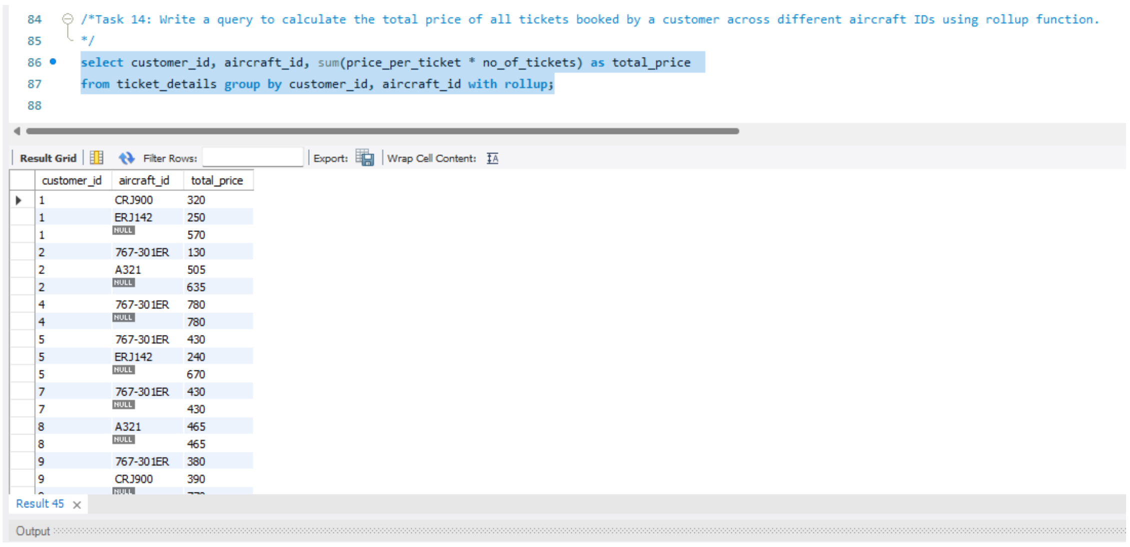
Task 13: For the route ID 4, write a query to view the execution plan of the passengers_on_flights table.
SQL query:
explain select * from passengers_on_flights where route_id = 4;



```
/* Task 13: For the route ID 4, write a query to view the execution plan of the passengers_on_flights table.
*/
DROP INDEX i1 ON passengers_on_flights;
explain select * from passengers_on_flights where route_id = "4";
```

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	passengers_on_flights	NULL	ALL	NULL	NULL	NULL	NULL	50	10.00	Using where

Task 14: Write a query to calculate the total price of all tickets booked by a customer across different aircraft IDs using rollup function.
SQL query:
select customer_id, aircraft_id, sum(price_per_ticket * no_of_tickets) as total_price from ticket_details group by customer_id, aircraft_id with rollup;



```
/*Task 14: Write a query to calculate the total price of all tickets booked by a customer across different aircraft IDs using rollup function.
*/
select customer_id, aircraft_id, sum(price_per_ticket * no_of_tickets) as total_price
from ticket_details group by customer_id, aircraft_id with rollup;
```

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

Result 45

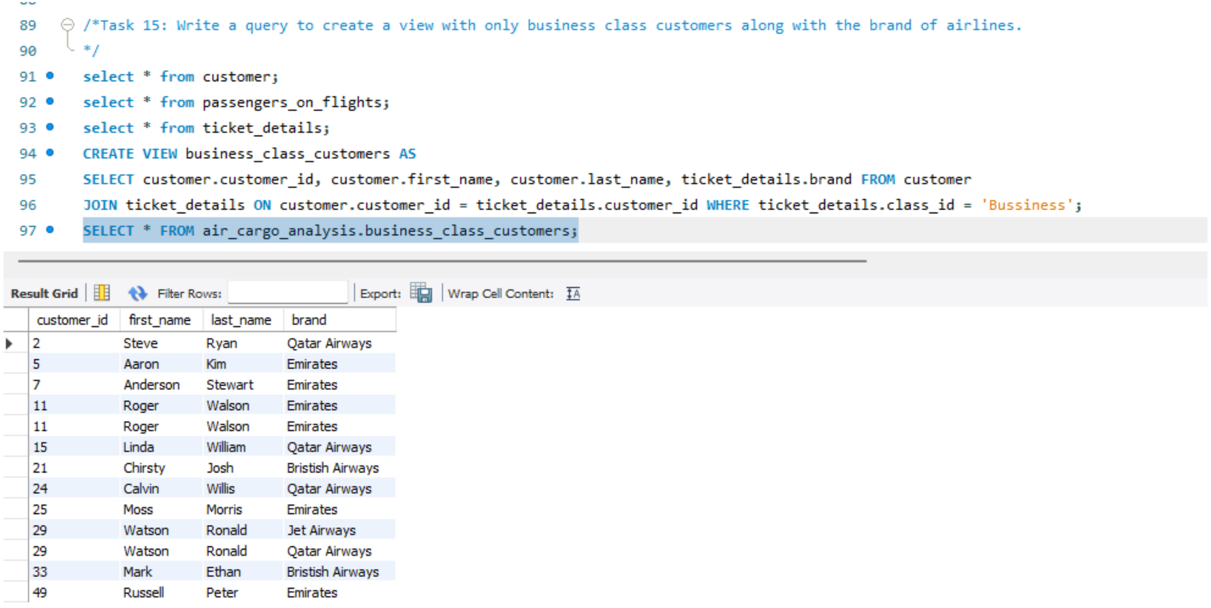
Task 15: Write a query to create a view with only business class customers along with the brand of airlines.

SQL query

CREATE VIEW business_class_customers AS

SELECT customer.customer_id, customer.first_name, customer.last_name, ticket_details.brand FROM customer

JOIN ticket_details ON customer.customer_id = ticket_details.customer_id WHERE ticket_details.class_id = 'Bussiness';



```
/*Task 15: Write a query to create a view with only business class customers along with the brand of airlines.
*/
select * from customer;
select * from passengers_on_flights;
select * from ticket_details;
CREATE VIEW business_class_customers AS
SELECT customer.customer_id, customer.first_name, customer.last_name, ticket_details.brand FROM customer
JOIN ticket_details ON customer.customer_id = ticket_details.customer_id WHERE ticket_details.class_id = 'Bussiness';
SELECT * FROM air_cargo_analysis.business_class_customers;
```

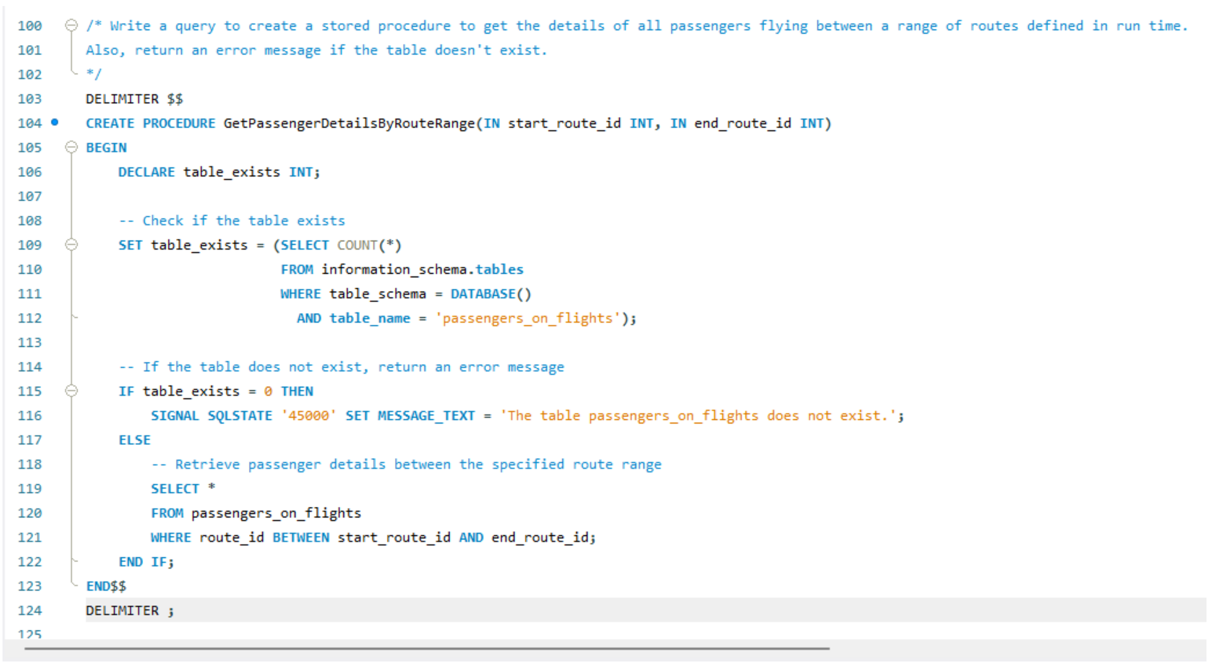
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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

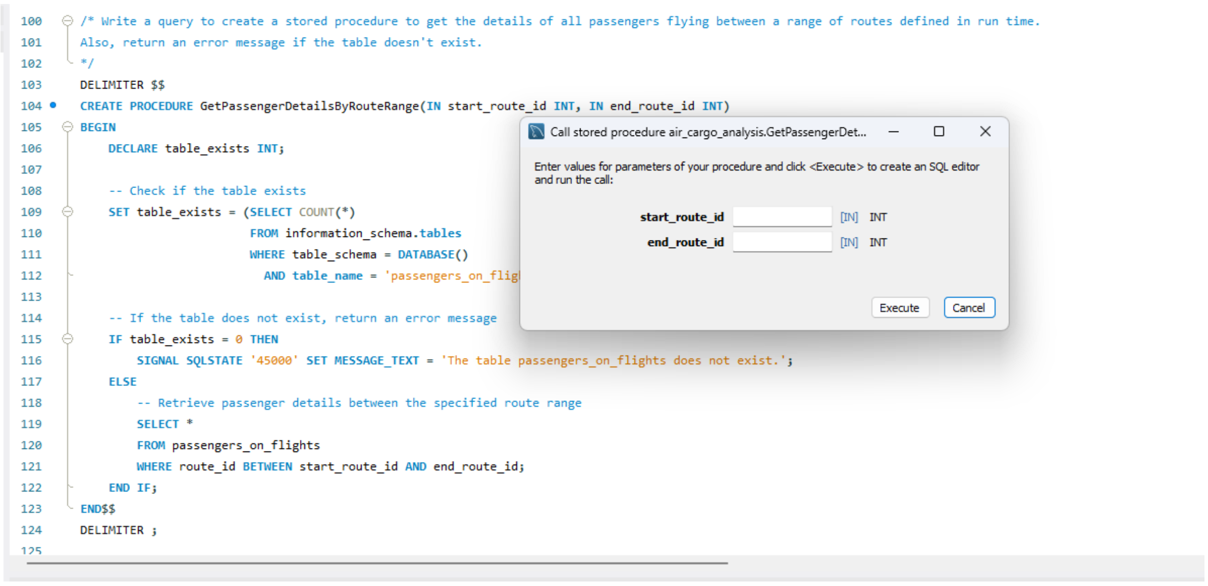
Task 16: Write a query to create a stored procedure to get the details of all passengers flying between a range of routes defined in run time. Also, return an error message if the table doesn't exist.

SQL query

```
DELIMITER $$
CREATE PROCEDURE GetPassengerDetailsByRouteRange(IN start_route_id INT, IN end_route_id INT)
BEGIN
  DECLARE table_exists INT;
  -- Check if the table exists
  SET table_exists = (SELECT COUNT(*)
                   FROM information_schema.tables  WHERE table_schema = DATABASE()
                    AND table_name = 'passengers_on_flights');
 -- If the table does not exist, return an error message
  IF table_exists = 0 THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'The table passengers_on_flights does not
exist.';
  ELSE
    -- Retrieve passenger details between the specified route range
    SELECT *
    FROM passengers_on_flights WHERE route_id BETWEEN start_route_id AND end_route_id;
  END IF;
END$$
DELIMITER ;
```



```sql
/* 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.
*/
DELIMITER $$
CREATE PROCEDURE GetPassengerDetailsByRouteRange(IN start_route_id INT, IN end_route_id INT)
BEGIN
    DECLARE table_exists INT;

    -- Check if the table exists
    SET table_exists = (SELECT COUNT(*)
                        FROM information_schema.tables
                        WHERE table_schema = DATABASE()
                          AND table_name = 'passengers_on_flights');

    -- If the table does not exist, return an error message
    IF table_exists = 0 THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'The table passengers_on_flights does not exist.';
    ELSE
        -- Retrieve passenger details between the specified route range
        SELECT *
        FROM passengers_on_flights
        WHERE route_id BETWEEN start_route_id AND end_route_id;
    END IF;
END$$
DELIMITER ;
```



```sql
/* 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.
*/
DELIMITER $$
CREATE PROCEDURE GetPassengerDetailsByRouteRange(IN start_route_id INT, IN end_route_id INT)
BEGIN
    DECLARE table_exists INT;

    -- Check if the table exists
    SET table_exists = (SELECT COUNT(*)
                        FROM information_schema.tables
                        WHERE table_schema = DATABASE()
                          AND table_name = 'passengers_on_flig

    -- If the table does not exist, return an error message
    IF table_exists = 0 THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'The table passengers_on_flights does not exist.';
    ELSE
        -- Retrieve passenger details between the specified route range
        SELECT *
        FROM passengers_on_flights
        WHERE route_id BETWEEN start_route_id AND end_route_id;
    END IF;
END$$
DELIMITER ;
```

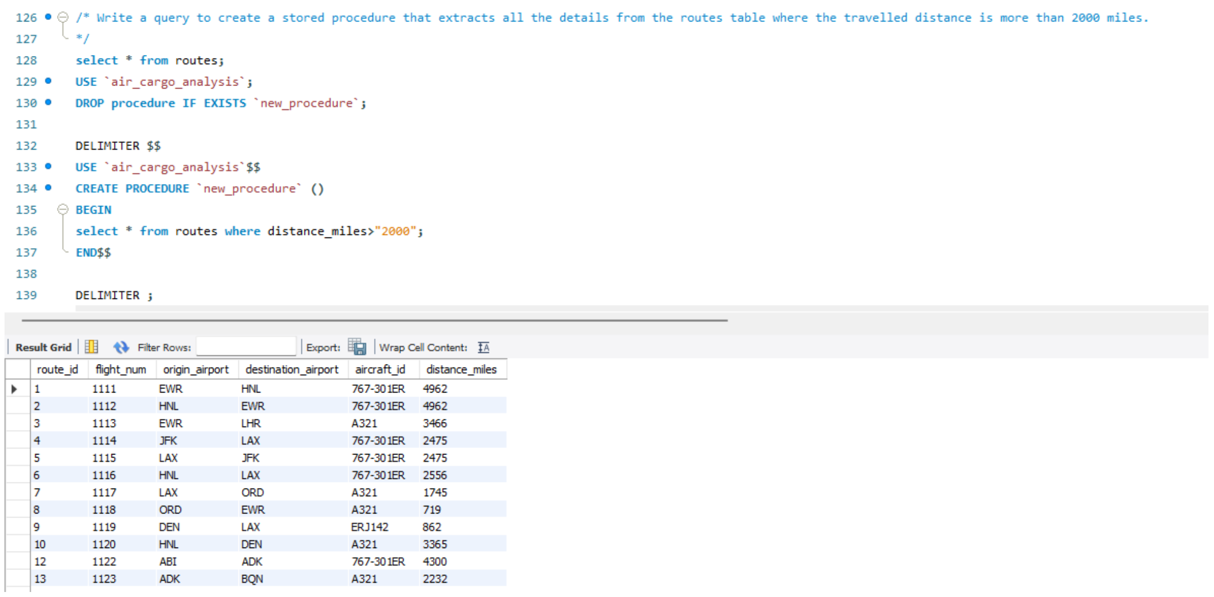
Call stored procedure air_cargo_analysis.GetPassengerDet...

Enter values for parameters of your procedure and click <Execute> to create an SQL editor and run the call:

start_route_id		[IN]	INT
end_route_id		[IN]	INT

Task 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.
SQL query

```
USE `air_cargo_analysis`;
DROP procedure IF EXISTS `new_procedure`;

DELIMITER $$
USE `air_cargo_analysis`$$
CREATE PROCEDURE `new_procedure` ()
BEGIN
select * from routes where distance_miles>"2000";
END$$

DELIMITER ;
```

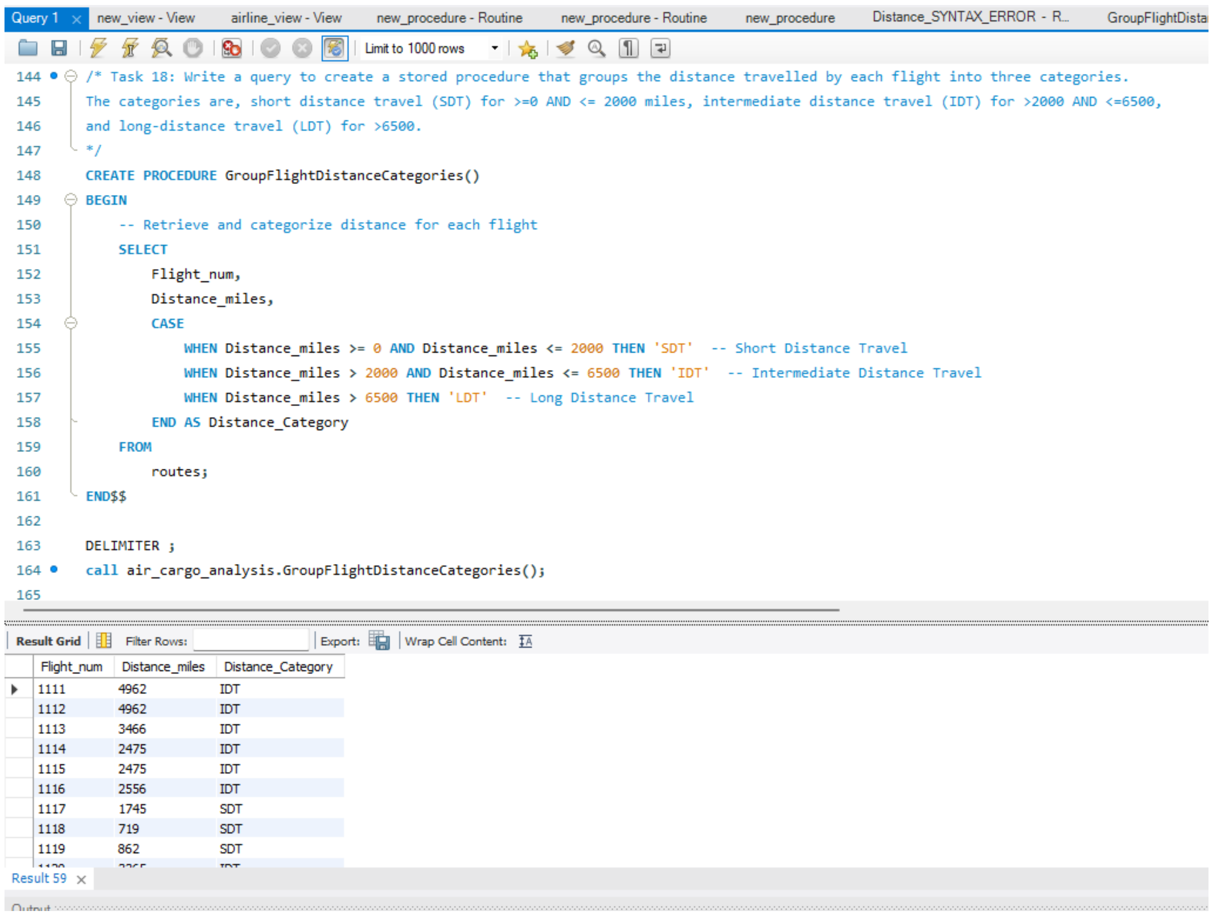


```
/* Write a query to create a stored procedure that extracts all the details from the routes table where the travelled distance is more than 2000 miles.
*/
select * from routes;
USE `air_cargo_analysis`;
DROP procedure IF EXISTS `new_procedure`;

DELIMITER $$
USE `air_cargo_analysis`$$
CREATE PROCEDURE `new_procedure` ()
BEGIN
select * from routes where distance_miles>"2000";
END$$

DELIMITER ;
```

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

Case 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.
SQL code:

```
CREATE PROCEDURE GroupFlightDistanceCategories()
BEGIN
    -- Retrieve and categorize distance for each flight
    SELECT Flight_num,  Distance_miles,
        CASE
            WHEN Distance_miles >= 0 AND Distance_miles <= 2000 THEN 'SDT'  -- Short Distance Travel
            WHEN Distance_miles > 2000 AND Distance_miles <= 6500 THEN 'IDT'  -- Intermediate
Distance Travel
            WHEN Distance_miles > 6500 THEN 'LDT'  -- Long Distance Travel
        END AS Distance_Category
    FROM routes;
END$$
DELIMITER ;
```



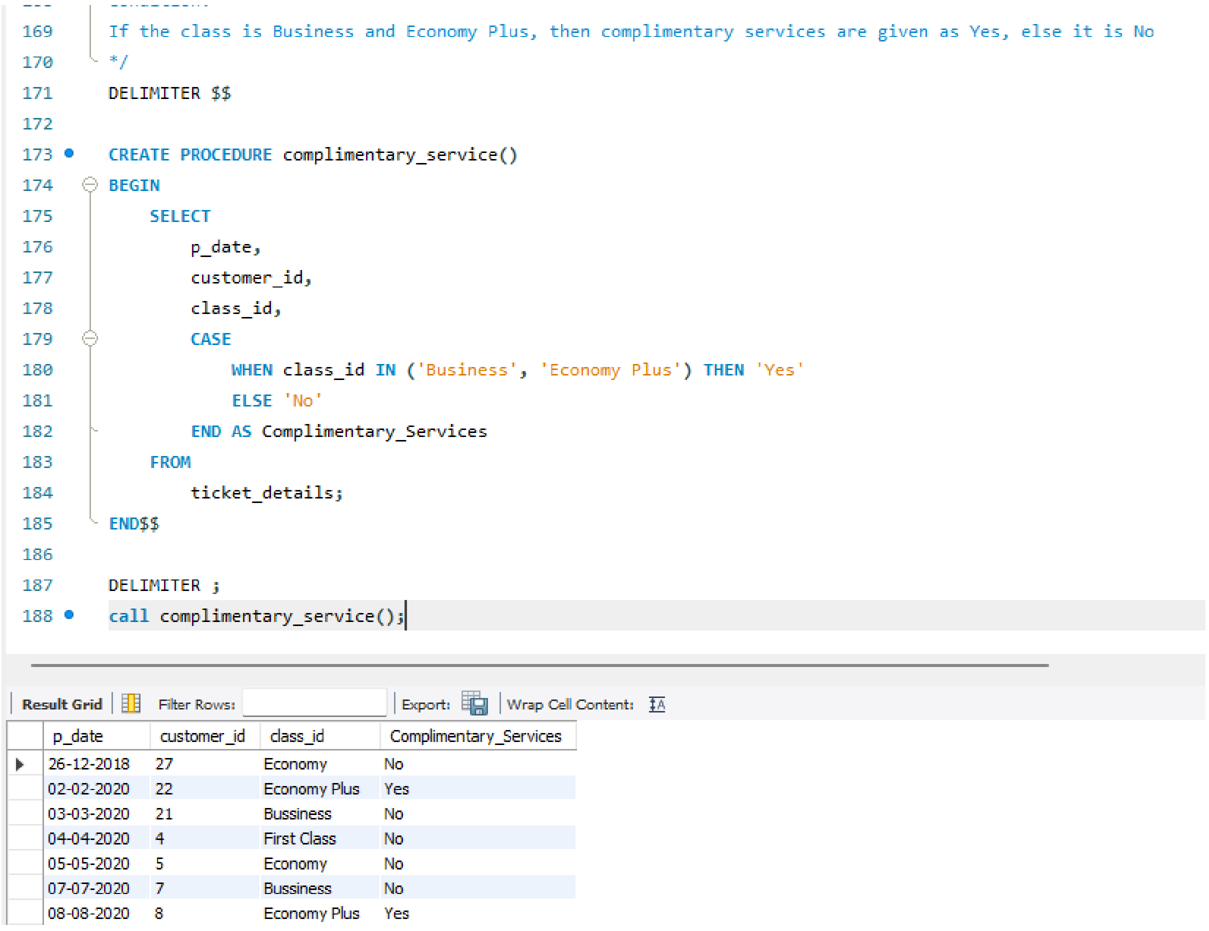
```
/* Task 18: Write a query to create a stored procedure that groups the distance travelled by each flight into three categories.
The categories are, short distance travel (SDT) for >=0 AND <= 2000 miles, intermediate distance travel (IDT) for >2000 AND <=6500,
and long-distance travel (LDT) for >6500.
*/
CREATE PROCEDURE GroupFlightDistanceCategories()
BEGIN
    -- Retrieve and categorize distance for each flight
    SELECT
        Flight_num,
        Distance_miles,
        CASE
            WHEN Distance_miles >= 0 AND Distance_miles <= 2000 THEN 'SDT'  -- Short Distance Travel
            WHEN Distance_miles > 2000 AND Distance_miles <= 6500 THEN 'IDT'  -- Intermediate Distance Travel
            WHEN Distance_miles > 6500 THEN 'LDT'  -- Long Distance Travel
        END AS Distance_Category
    FROM
        routes;
END$$

DELIMITER ;
call air_cargo_analysis.GroupFlightDistanceCategories();
```

Flight_num	Distance_miles	Distance_Category
1111	4962	IDT
1112	4962	IDT
1113	3466	IDT
1114	2475	IDT
1115	2475	IDT
1116	2556	IDT
1117	1745	SDT
1118	719	SDT
1119	862	SDT

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 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 complimentary_service()
BEGIN
  SELECT p_date, customer_id, class_id,
    CASE
      WHEN class_id IN ('Business', 'Economy Plus') THEN 'Yes'
      ELSE 'No'
    END AS Complimentary_Services
  FROM
    ticket_details;
END$$

DELIMITER ;
```



```sql
If the class is Business and Economy Plus, then complimentary services are given as Yes, else it is No
*/
DELIMITER $$

CREATE PROCEDURE complimentary_service()
BEGIN
    SELECT
        p_date,
        customer_id,
        class_id,
        CASE
            WHEN class_id IN ('Business', 'Economy Plus') THEN 'Yes'
            ELSE 'No'
        END AS Complimentary_Services
    FROM
        ticket_details;
END$$

DELIMITER ;
call complimentary_service();
```

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	No
04-04-2020	4	First Class	No
05-05-2020	5	Economy	No
07-07-2020	7	Bussiness	No
08-08-2020	8	Economy Plus	Yes

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.