

# Air Cargo Analysis.

## Project 2

### 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.

**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

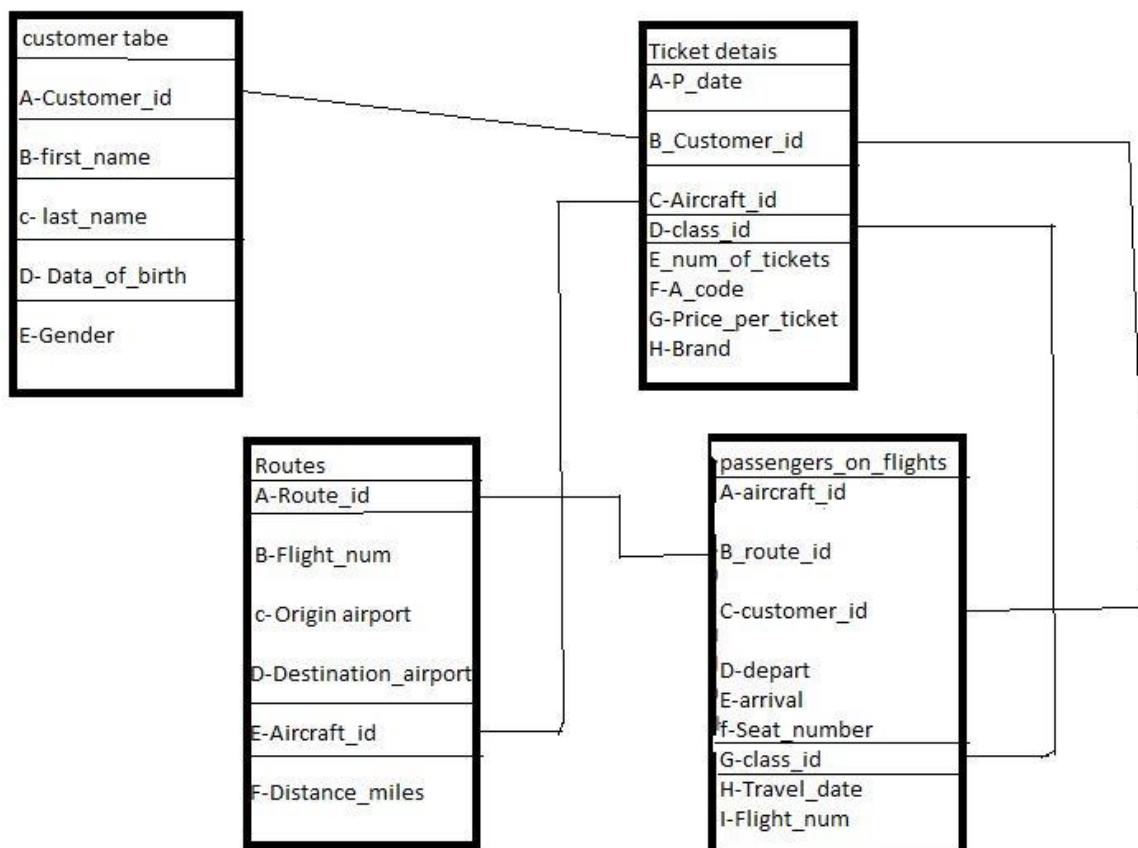
- Route\_id – Route ID of from and to location
- Flight\_num – Specific flight number for each route
- Origin\_airport – Departure location
- Destination\_airport – Arrival location
- Aircraft\_id – ID of each aircraft in a brand
- Distance\_miles – Distance between departure and arrival location

**Following operations should be performed:**

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

---- Diagram is showing the relationship between the different tables through the common columns( attributes) like Customer\_id in Customer table is a primary key which also link with the ticket\_details , so it could be use to access the details of the ticket.

**ENTITY RELATIONSHIP DIAGRAM**



- 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.

The screenshot shows the MySQL Workbench interface. In the center pane, a SQL query is being executed to create a new table named 'route\_details'. The query is as follows:

```

1 • create table route_details
2 (
3   route_id int ,
4   flight_num int,
5   origin_airport varchar(10),
6   destination_airport varchar(10),
7   aircraft_id varchar(10),
8   distance int,
9   check (flight_num is not null),
10  unique (route_id),
11  check (distance > 0)

```

The left pane shows the 'air\_cargo' schema with tables 'customer', 'passengers\_on\_flights', and 'route\_details'. The 'route\_details' table is selected, and its columns are listed: route\_id, flight\_num, origin\_airport, destination\_airport, aircraft\_id, and distance. The right pane shows the 'Result Grid' with the following data:

#	Field	Type	Null	Key	Default	Extra
1	route_id	int	YES	UNI		
2	flight_num	int	YES			
3	origin_airport	varchar(10)	YES			
4	destination...	varchar(10)	YES			
5	aircraft_id	varchar(10)	YES			
6	distance	int	YES			

The status bar at the bottom indicates 'Query Completed'.

- 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.

The screenshot shows the MySQL Workbench interface. In the center pane, a SQL query is being executed to display passengers on flights. The query is as follows:

```

1 • select * from passengers_on_flights
2   where route_id between 1 and 25;
3
4 #second_way
5
6 • select * from passengers_on_flights
7   where route_id >= 1 and route_id <= 25;

```

The left pane shows the 'air\_cargo' schema with tables 'customer', 'passengers\_on\_flights', and 'route\_details'. The 'passengers\_on\_flights' table is selected, and its columns are listed: route\_id, flight\_num, origin\_airport, destination\_airport, aircraft\_id, and distance. The right pane shows the 'Result Grid' with the following data:

#	customer_id	aircraft_id	route_id	depart	arrival	seat_num	class_id	travel_date	flight_num
1	2	767-301ER	4	JFK	LAX	01E	Economy	02-09-2018	1114
2	1	ERJ142	9	DEN	LAX	01EP	Economy Plus	26-12-2019	1119
3	5	767-301ER	12	ABI	ADK	02B	Bussiness	02-07-2018	1122
4	5	ERJ142	18	ANI	BGR	02E	Economy	06-05-2020	1128
5	4	767-301ER	5	LAX	JFX	02FC	First Class	06-04-2020	1115
6	7	767-301ER	20	AVL	BOI	03B	Bussiness	08-07-2020	1130
7	5	ERJ142	22	BGR	RII	03E	Economy	31-05-2020	1132

The status bar at the bottom indicates 'Query Completed'.

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

The screenshot shows the MySQL Workbench interface. The 'Schemas' pane on the left displays the 'air\_cargo' database with tables 'customer', 'passengers\_on\_flights', 'route\_details', 'routes', and 'ticket\_details'. The 'Query 1' window contains the following SQL query:

```
1 select * from air_cargo.ticket_details;
2
3 select count(customer_id) as Total_customers , sum(price_per_ticket) as total_revenue
4 from air_cargo.ticket_details
5 where class_id = "Business";
```

The 'Result Grid' shows the results of the query:

#	Total_customer	total_revenue
1	13	6034

The status bar at the bottom indicates 'Query Completed'.

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

The screenshot shows the MySQL Workbench interface. The 'Schemas' pane on the left displays the 'air\_cargo' database with tables 'customer', 'passengers\_on\_flights', 'route\_details', 'routes', and 'ticket\_details'. The 'Query 1' window contains the following SQL query:

```
1 select concat( first_name , " ", last_name ) as full_name
2 from customer;
```

The 'Result Grid' shows the results of the query:

#	full name
1	Julie Sam
2	Steve Ryan
3	Morris Lois
4	Cathenna Emily
5	Aaron Kim
6	Alexander Scot
7	Anderson Stewart
8	Floyd Ted
9	Leo Travis
10	Melvin Tracy
11	Roger Walson
12	Shirley Wally
13	Solomon Walter

The status bar at the bottom indicates 'Query Completed'.

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

The screenshot shows the MySQL Workbench interface. The 'Schemas' pane on the left shows the 'air\_cargo' database selected, with tables 'customer' and 'ticket\_details' visible. The 'Query Editor' contains the following SQL query:

```
1
2 • select TD.customer_id , C.first_name, C.last_name
3   from ticket_details as TD
4  left join
5   customer as C
6   on TD.customer_id = C.customer_id;
7
8
```

The 'Result Grid' shows the results of the query:

#	customer_id	first_name	last_name
1	27	Cherly	Vernon
2	22	Pheny	Eri
3	21	Chirsty	Josh
4	4	Cathenna	Emily
5	5	Aaron	Kim
6	7	Anderson	Stewart
7	8	Floyd	Ted

The status bar at the bottom indicates 'Query Completed'.

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.

The screenshot shows the MySQL Workbench interface. The 'Schemas' pane on the left shows the 'air\_cargo' database selected, with tables 'customer' and 'ticket\_details' visible. The 'Query Editor' contains the following SQL query:

```
1
2 • select C.first_name , C.last_name , TD.brand
3   from air_cargo.ticket_details TD
4  left join customer C
5   on TD.customer_id= C.customer_id
6  where brand ="Emirates";
```

The 'Result Grid' shows the results of the query:

#	first_name	last_name	brand
1	Cherly	Vernon	Emirates
2	Cathenna	Emily	Emirates
3	Anderson	Stewart	Emirates
4	Leo	Travis	Emirates
5	Roger	Walson	Emirates
6	Moss	Morris	Emirates
7	Gloria	Richie	Emirates
8	Moss	Morris	Emirates
9	Carol	Vernon	Emirates
10	Joyce	Paul	Emirates

The status bar at the bottom indicates 'Query Completed'.

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.

The screenshot shows the MySQL Workbench interface. The 'Schemas' pane on the left shows the 'passengers\_on\_flights' table. The 'Query 1' editor contains the following SQL query:

```

30 select C.first_name , C.last_name , PF.customer_id ,PF.class_id
31 from passengers_on_flights PF
32 left join customer C
33 on PF.customer_id =C.customer_id
34 group by 1,2,3,4
35 having class_id = "Economy plus";

```

The 'Result Grid' shows the following data:

#	first_name	last_name	customer_id	class_id
1	Julie	Sam	1	Economy Plus
2	Floyd	Ted	8	Economy Plus
3	Roger	Walson	11	Economy Plus
4	Catherine	Shad	17	Economy Plus
5	Joyce	Paul	19	Economy Plus
6	Pheny	Eri	22	Economy Plus
7	Chirstoper	Sean	32	Economy Plus
8	Sophia	Carl	47	Economy Plus
9	Rose	Arthur	50	Economy Plus

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

The screenshot shows the MySQL Workbench interface. The 'Schemas' pane on the left shows the 'air\_cargo' database and the 'ticket\_details' table. The 'Query 1' editor contains the following SQL query:

```

6
7 #Method_1
8 select
9 if(sum(price_per_ticket)>10000 ,"yes,Revenue is high" , "No") as Revenue_check
10 from ticket_details;
11 #Method_2
12 select sum(no_of_tickets*price_per_ticket) as Total_revenue,
13 if(sum(no_of_tickets*price_per_ticket) >10000, "yes,Revenue is high" , "No") as Revenue_
14 from ticket_details;
15

```

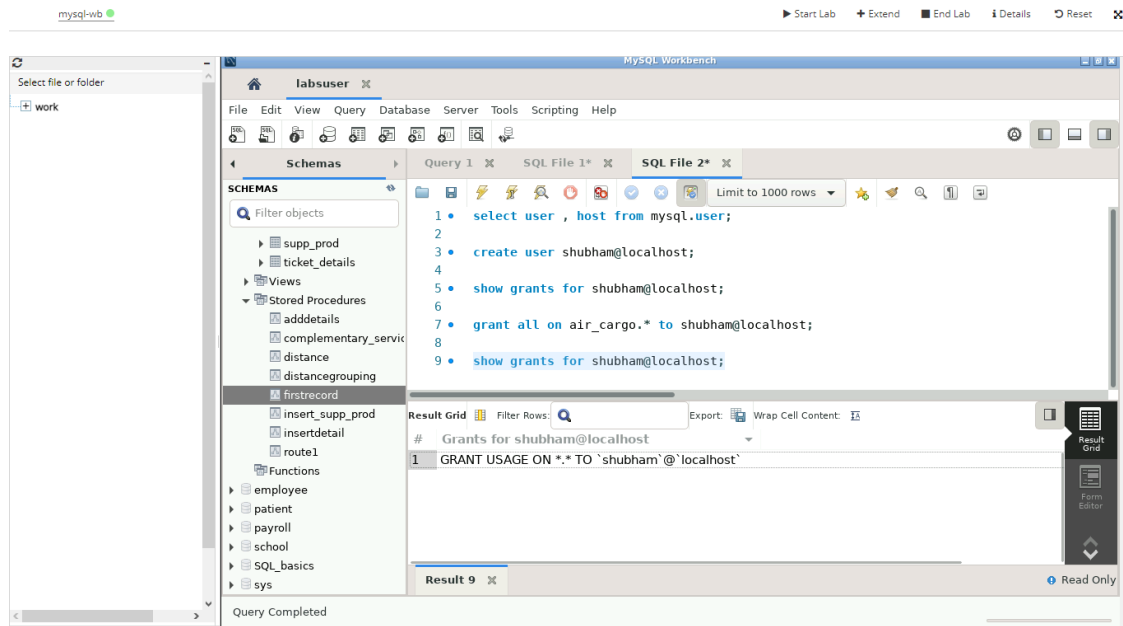
The 'Result Grid' shows the following data:

#	Total_revenue	Revenue_check
1	15369	yes,Revenue is high

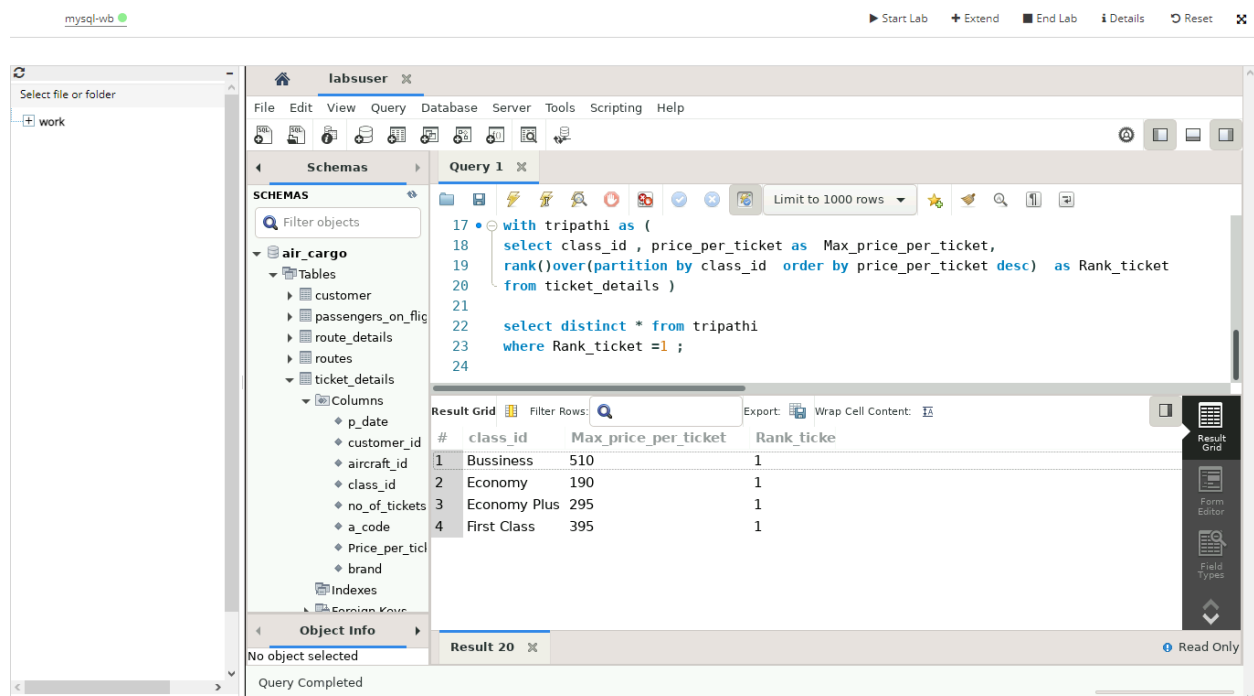
The 'Action Output' pane shows the following messages:

#	Time	Action	Message
19	14:50:03	select * from ticket_details limit 0, 1000	20 row(s) returned
20	14:44:11	select if(sum(price_per_ticket)>10000,"yes,Revenue is hi...	1 row(s) returned
21	14:48:47	select sum(no_of_tickets*price_per_ticket) as Total_reven...	Error Code: 1146. Table 'air_cargo.ticket_details'
22	14:49:14	select sum(no_of_tickets*price_per_ticket) as Total_reven...	Error Code: 1054. Unknown column 'Total_reven...
23	14:49:47	select sum(no_of_tickets*price_per_ticket) as Total_reven...	1 row(s) returned

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.

mysql-wb

Start Lab + Extend End Lab Details Reset

labuser

File Edit View Query Database Server Tools Scripting Help

Schemas

Filter objects

- last\_name
- date\_of\_birth
- gender
- Indexes
- Foreign Keys
- Triggers
- passengers\_on\_flights
  - Columns
    - customer\_id
    - aircraft\_id
    - route\_id
    - depart
    - arrival
    - seat\_num
    - class\_id
    - travel\_date
    - flight\_num
  - Indexes

Object Info Session

No object selected

Query Completed

Query 1 x SQL File 1\* x

Limit to 1000 rows

```
17
18 select * from passengers_on_flights;
19
20 select PF.customer_id , C.first_name , C.last_name
21 from passengers_on_flights PF
22 left join customer C
23 on PF.customer_id = C.customer_id
24 where PF.route_id=4 ;
25
26
27
28
29
```

Result Grid Filter Rows: Export: Wrap Cell Content:

#	customer_id	first_name	last_name
1	2	Steve	Ryan
2	4	Cathenna	Emily
3	11	Roger	Walson

Result 6 x Result 7 x Result 8 x Read Only

13. For the route ID 4, write a query to view the execution plan of the passengers\_on\_flights table.

mysql-wb

Start Lab + Extend End Lab Details Reset

labuser

File Edit View Query Database Server Tools Scripting Help

Schemas

Filter objects

- air\_cargo
  - Tables
    - customer
    - passengers\_on\_flight
    - route\_details
    - routes
    - ticket\_details
      - Columns
        - p\_date
        - customer\_id
        - aircraft\_id
        - class\_id
        - no\_of\_tickets
        - a\_code
        - Price\_per\_ticket
        - brand
      - Indexes
      - Foreign Keys

Object Info Session

No object selected

Query Completed

Query 1 x SQL File 1\* x

Limit to 1000 rows

```
1
2
3 select aircraft_id ,depart , arrival ,travel_date ,flight_num
4 from air_cargo.passengers_on_flights
5 where route_id =4
```

Result Grid Filter Rows: Export: Wrap Cell Content:

#	aircraft_id	depart	arrival	travel_date	flight_num
1	767-301ER	JFK	LAX	02-09-2018	1114
2	767-301ER	JFK	LAX	30-04-2020	1114
3	767-301ER	JFK	LAX	09-11-2020	1114

Result 6 x Result 7 x Result 8 x Read Only



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

The screenshot shows the MySQL Workbench interface with the following details:

- Query 1:**

```

1
2
3 • select sum(price_per_ticket), aircraft_id
4 from air_cargo.ticket_details
5 group by aircraft_id with rollup;

```
- Result Grid:**

#	sum(price_per_ticke	aircraft_id
1	5634	767-301ER
2	4270	A321
3	3440	CRJ900
4	2025	ERJ142
5	15369	NULL

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

The screenshot shows the MySQL Workbench interface with the following details:

- Query 1:**

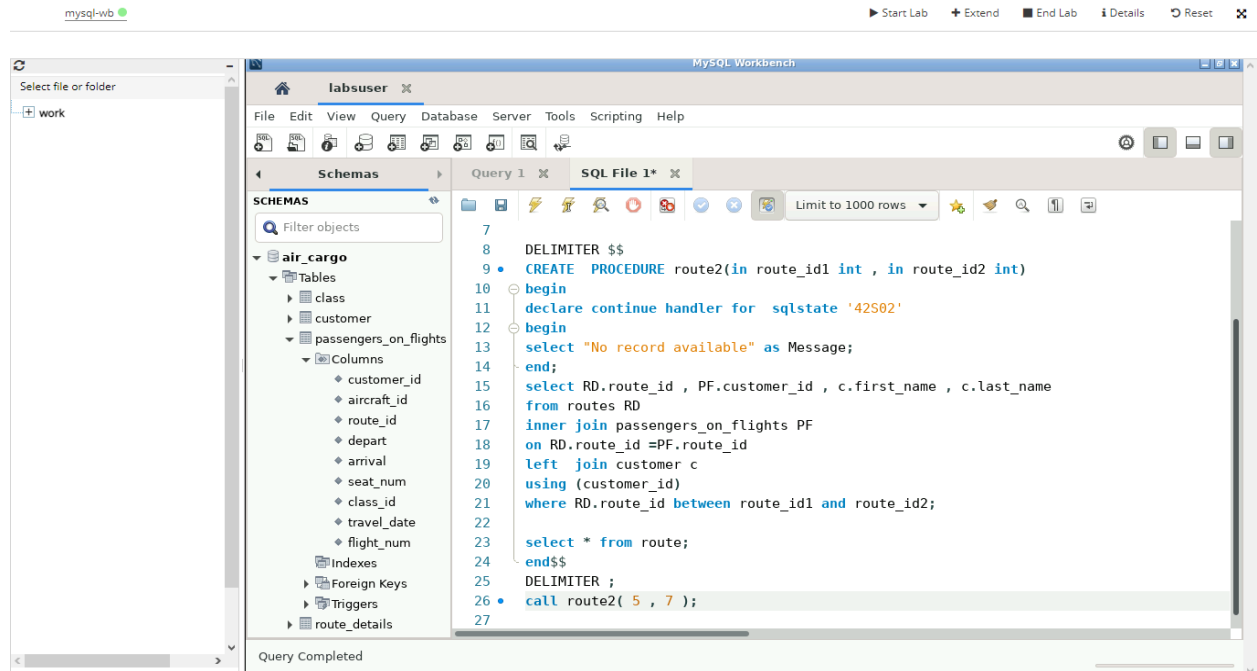
```

1
2 • create view brand
3 as
4 select c.first_name,c.last_name,TD.customer_id , TD.class_id, TD.brand
5 from air_cargo.ticket_details as TD
6 left join customer as c
7 on TD.customer_id= c.customer_id
8 where class_id="Bussiness";

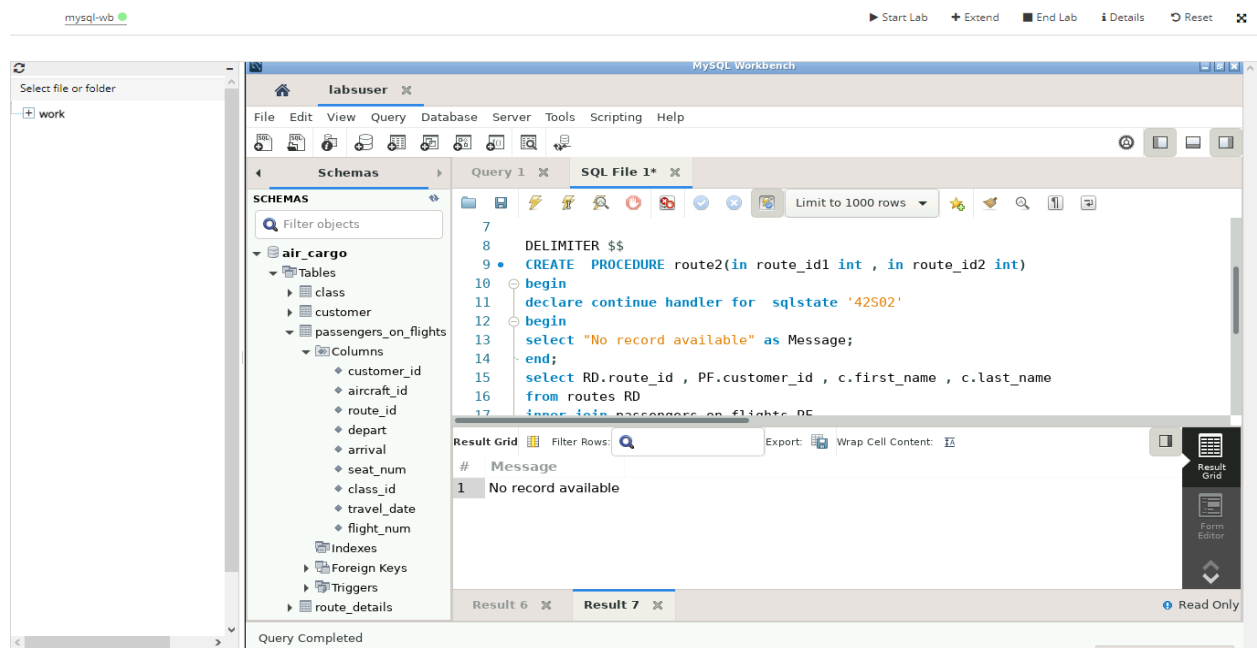
```
- Result Grid:**

#	first_name	last_name	customer_id	class_id	brand
1	Chirsty	Josh	21	Bussiness	Bristish Airways
2	Anderson	Stewart	7	Bussiness	Emirates
3	Roger	Walson	11	Bussiness	Emirates
4	Moss	Morris	25	Bussiness	Emirates
5	Calvin	Willis	24	Bussiness	Qatar Airways
6	Watson	Ronald	29	Bussiness	Qatar Airways
7	Steve	Ryan	2	Bussiness	Qatar Airways
8	Watson	Ronald	29	Bussiness	Jet 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--**CREATION OF STORED PROCEDURE QUERY**→.



---**RESULT SHOWING BECAUSE :ROUTE TABLE: IS NOT IN DB:----**



## --OTHER RESULT WITH CONTINUE ---

The screenshot shows the MySQL Workbench interface. The 'Schemas' pane on the left shows the 'air\_cargo' database with tables 'class', 'customer', and 'passengers\_on\_flights'. The 'Query 1' editor contains the following SQL code:

```
DELIMITER $$
CREATE PROCEDURE route2(in route_id1 int, in route_id2 int)
begin
declare continue handler for sqlstate '42S02'
begin
select "No record available" as Message;
end;
select RD.route_id, PF.customer_id, c.first_name, c.last_name
from routes RD
inner join passengers_on_flights PF
on RD.route_id = PF.route_id
where RD.route_id = route_id1 and PF.route_id = route_id2;
```

The 'Result Grid' shows the following data:

#	route_id	customer_id	first_name	last_name
1	5	4	Cathenna	Emily
2	5	11	Roger	Walson

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.

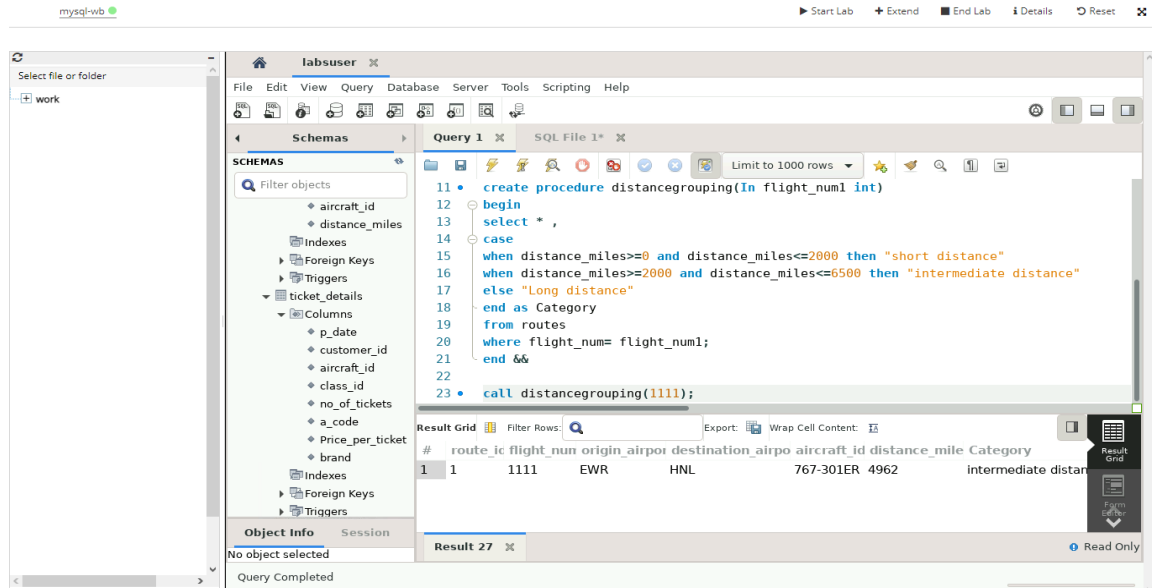
The screenshot shows the MySQL Workbench interface. The 'Schemas' pane on the left shows the 'air\_cargo' database with tables 'class', 'customer', and 'passengers\_on\_flights'. The 'Query 1' editor contains the following SQL code:

```
Delimiter %%
create procedure distance()
begin
select * from routes
where distance_miles > 2000;
end %%
call distance;
```

The 'Result Grid' shows the following data:

#	route_id	flight_num	origin_airport	destination_airport	aircraft_id	distance_mile
1	1	1111	EWB	HNL	767-301ER	4962
2	2	1112	HNL	EWB	767-301ER	4962
3	3	1113	EWB	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

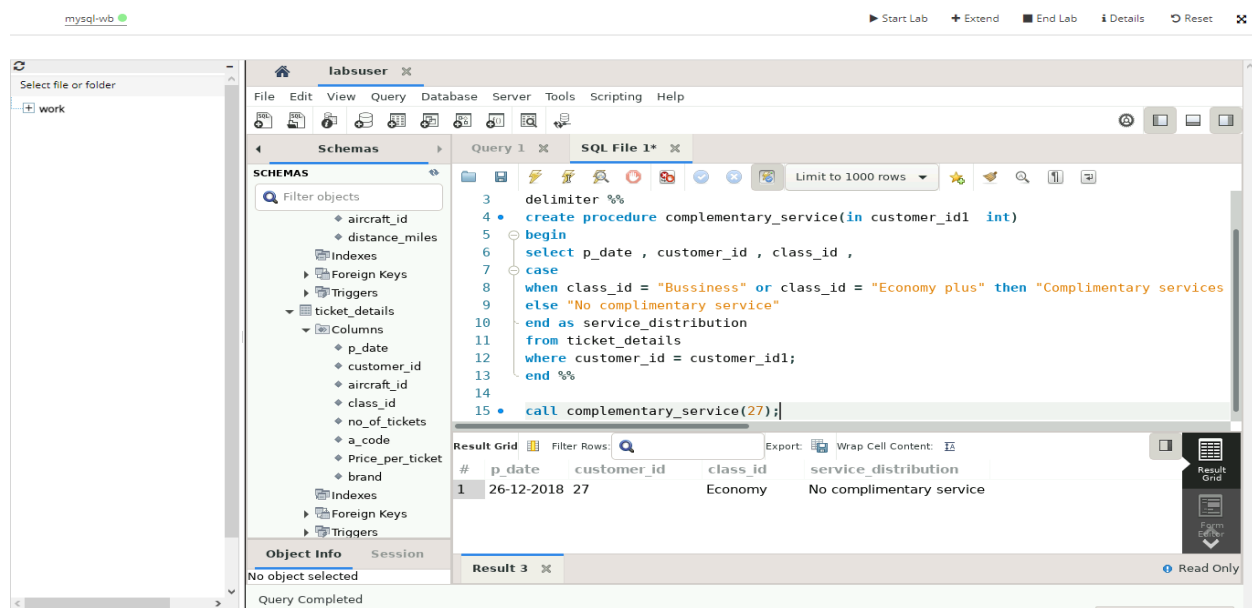
18. Write a query to create a stored procedure that groups the distance travelled by each flight into three categories. The categories are, short distance travel (SDT) for  $\geq 0$  AND  $\leq 2000$  miles, intermediate distance travel (IDT) for  $> 2000$  AND  $\leq 6500$ , and long-distance travel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

mysql-wb

Start Lab + Extend End Lab Details Reset

MySQL Workbench

labuser

File Edit View Query Database Server Tools Scripting Help

Schemas Query 1 SQL File 1\*

Limit to 1000 rows

```
27 DELIMITER $$
28 CREATE PROCEDURE firstrecord()
29 BEGIN DECLARE a VARCHAR(20);
30 DECLARE b VARCHAR(20);
31 DECLARE c INT;
32 DECLARE cursor_1 CURSOR FOR SELECT first_name, last_name, customer_id FROM customer
33 WHERE last_name = "scott";
34 OPEN cursor_1;
35 FETCH cursor_1 INTO a, b, c;
36 SELECT a AS first_name, b AS last_name, c AS customer_id;
37 CLOSE cursor_1;
38 END$$
39 DELIMITER ;
40 CALL firstrecord();
41
```

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

#	first_name	last_name	customer_id
1	Samuel	Scott	37

Result 8 Read Only

Query Completed